The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

For more information on the *Stata Journal*, including information for authors, see the web page

http://www.stata-journal.com

The *Stata Journal* is indexed and abstracted in the following:

- Science Citation Index Expanded (also known as SciSearch®)
- CompuMath Citation Index®

The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press. Stata and Mata are registered trademarks of StataCorp LP.

Subscriptions are available from StataCorp, 4905 Lakeway Drive, College Station, Texas 77845, telephone 979-696-4600 or 800-STATA-PC, fax 979-696-4601, or online at

http://www.stata.com/bookstore/sj.html

Subscription rates

The listed subscription rates include both a printed and an electronic copy unless otherwise mentioned.

Subscriptions mailed to U.S. and Canadian addresses:

3-year subscription	$195
2-year subscription	$135
1-year subscription	$ 69
1-year student subscription	$ 42
1-year university library subscription	$ 89
1-year institutional subscription	$195

Subscriptions mailed to other countries:

3-year subscription	$285
2-year subscription	$195
1-year subscription	$ 99
3-year subscription (electronic only)	$185
1-year student subscription	$ 69
1-year university library subscription	$119
1-year institutional subscription	$225

Back issues of the *Stata Journal* may be ordered online at

http://www.stata.com/bookstore/sjj.html

Individual articles three or more years old may be accessed online without charge. More recent articles may be ordered online.

http://www.stata-journal.com/archives.html

The *Stata Journal* is published quarterly by the Stata Press, College Station, Texas, USA.

Address changes should be sent to the *Stata Journal*, StataCorp, 4905 Lakeway Drive, College Station, TX 77845, USA, or emailed to sj@stata.com.

Volume 8 Number 1 2008

The Stata Journal

Articles and Columns 3

Notes and Comments 134

Software Updates 146

The Stata Journal (2008)
8, Number 1, pp. 1–2

Editorial announcements

1 Changes to Editorial Board

The *Stata Journal* is greatly dependent on its Associate Editors, who carry much of the burden of reviewing papers or suggesting reviewers and who provide the Editors with invaluable advice, whether solicited or spontaneous. From this issue we bid good-bye, with warmest thanks for their many contributions over several years, to three long-standing Associate Editors:

- David Clayton, Cambridge Institute for Medical Research (2001–2007)
- Charles Franklin, University of Wisconsin−Madison (2001–2007)
- Joanne Garrett, University of North Carolina (2001–2007 and *Stata Technical Bulletin* 1994–2001)

We also welcome five new Associate Editors and look forward to working with them:

- Nathaniel Beck, New York University
- Maarten L. Buis, Vrije Universiteit, Amsterdam
- David Epstein, Columbia University
- Frauke Kreuter, University of Maryland−College Park
- Austin Nichols, Urban Institute, Washington DC

2 Stata Journal is now accessible online

We are delighted to announce on behalf of the publisher that back issues of the *Stata Journal* are now available online, regardless of whether you or your institution subscribe. Articles will become available based on a moving wall: articles three or more years old may be accessed without charge, while more recent articles may be accessed following a payment of $7.50 per article. Point your browser to

http://www.stata-journal.com/archives.html

to access a simple search engine (search by keyword, author, or title) or to find individual articles. All articles are delivered in PDF form. This URL also leads to information about the terms under which articles are made available. We believe that this online access will be enormously useful to all subscribers—and not only because your subscription may not date back to volume 1, number 1 in 2001. Do spread the information to all those who may be interested, both within and beyond the Stata community.

3 Stata Technical Bulletin is also accessible online

The entire back run of the *Stata Technical Bulletin* (STB) is also now available without charge online. The STB preceded the *Stata Journal* and appeared in 61 issues from May 1991 to May 2001. Visit

http://www.stata.com/bookstore/stbj.html

to be able to access each issue as a separate PDF. Much of the STB's contents have been superseded by developments in Stata itself or in later user-written work, but equally many programs still in use are documented in the STB, and many useful expository articles may be found in back issues.

Thus whenever you come across a reference to a previous article in the *Stata Journal* or *Stata Technical Bulletin*—say, through use of the `search` command on a keyword, through an article in the *Stata Journal*, or through a Statalist posting—easy access is now possible for you, your colleagues, and your students.

H. Joseph Newton and Nicholas J. Cox
Editors, *Stata Journal*

The Stata Journal (2008)
8, Number 1, pp. 3–28

metan: fixed- and random-effects meta-analysis

Ross J. Harris
Department of Social Medicine
University of Bristol
Bristol, UK
epzrgh@bristol.ac.uk

Michael J. Bradburn
Health Services Research Center
University of Sheffeld
Sheffield, UK

Jonathan J. Deeks
Department of Primary Care Medicine
University of Birmingham
Birmingham, UK

Roger M. Harbord
Department of Social Medicine
University of Bristol
Bristol, UK

Douglas G. Altman
Centre for Statistics in Medicine
University of Oxford
Oxford, UK

Jonathan A. C. Sterne
Department of Social Medicine
University of Bristol
Bristol, UK

Abstract. This article describes updates of the meta-analysis command `metan` and options that have been added since the command's original publication (Bradburn, Deeks, and Altman, metan – an alternative meta-analysis command, *Stata Technical Bulletin Reprints*, vol. 8, pp. 86–100). These include version 9 graphics with flexible display options, the ability to meta-analyze precalculated effect estimates, and the ability to analyze subgroups by using the `by()` option. Changes to the output, saved variables, and saved results are also described.

Keywords: sbe24_2, metan, meta-analysis, forest plot

1 Introduction

Meta-analysis is a two-stage process involving the estimation of an appropriate summary statistic for each of a set of studies followed by the calculation of a weighted average of these statistics across the studies (Deeks, Altman, and Bradburn 2001). Odds ratios, risk ratios, and risk differences may be calculated from binary data, or a difference in means obtained from continuous data. Alternatively, precalculated effect estimates and their standard errors from each study may be pooled, for example, adjusted log-odds ratios from observational studies. The summary statistics from each study can be combined by using a variety of meta-analytic methods, which are classified as fixed-effect models in which studies are weighted according to the amount of information they contain; or random-effects models, which incorporate an estimate of between-study variation (heterogeneity) in the weighting. A meta-analysis will customarily include a forest plot, in which results from each study are displayed as a square and a horizontal line, representing the intervention effect estimate together with its confidence interval. The area of the square reflects the weight that the study contributes to the meta-analysis. The combined-effect estimate and its confidence interval are represented by a diamond.

Here we present updates to the `metan` command and other previously undocumented additions that have been made since its original publication (Bradburn, Deeks, and Altman 1998). New features include

- Version 9 graphics
- Flexible display of tabular data in the forest plot
- Results from a second type of meta-analysis displayed in the same forest plot
- `by()` group processing
- Analysis of precalculated effect estimates
- Prediction intervals for the intervention effect in a new study from random-effects analyses

There are a substantial number of options for the `metan` command because of the variety of meta-analytic techniques and the need for flexible graphical displays. We recommend that new users not try to learn everything at once but to learn the basics and build from there as required. Clickable examples of `metan` are available in the help file, and the dialog box may also be a good way to start using `metan`.

2 Example data

The dataset used in subsequent examples is taken from the meta-analysis published as table 1 in Colditz et al. (1994, 699). The aim of the analysis was to quantify the efficacy of BCG vaccine against tuberculosis, and data from 11 trials are included here. There was considerable between-trial heterogeneity in the effect of the vaccine; it has been suggested that this might be explained by the latitude of the region in which the trial was conducted (Fine 1995).

▷ Example

Details of the dataset are shown below by using `describe` and `list` commands.

```
. use bcgtrial
(BCG and tuberculosis)

. describe

Contains data from bcgtrial.dta
  obs:            11                          BCG and tuberculosis
 vars:           12                           31 May 2007 17:11
 size:           693 (99.9% of memory free)   (_dta has notes)
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
trial           byte   %8.0g                  Trial number
trialnam        str14  %14s                   Trial name
authors         str20  %20s                   Authors of trial
startyr         int    %8.0g                  Year trial started
latitude        byte   %8.0g                  Latitude of trial area
alloc           byte   %33.0g      alloc      Allocation method
tcases          int    %8.0g                  BCG vaccinated cases
tnoncases       float  %9.0g                  BCG vaccinated noncases
ccases          int    %8.0g                  Unvaccinated cases
cnoncases       float  %9.0g                  Unvaccinated noncases
ttotal          long   %12.0g                 BCG vaccinated population
ctotal          long   %12.0g                 Unvaccinated population
-------------------------------------------------------------------------------
Sorted by:  startyr  authors

. list trialnam startyr tcases tnoncases ccases cnoncases, clean noobs
> abbreviate(10)

          trialnam   startyr   tcases   tnoncases   ccases   cnoncases
            Canada      1933        6         300       29         274
      Northern USA      1935        4         119       11         128
           Chicago      1941       17        1699       65        1600
     Georgia (Sch)      1947        5        2493        3        2338
       Puerto Rico      1949      186       50448      141       27197
    Georgia (Comm)      1950       27       16886       29       17825
       Madanapalle      1950       33        5036       47        5761
                UK      1950       62       13536      248       12619
      South Africa      1965       29        7470       45        7232
             Haiti      1965        8        2537       10         619
            Madras      1968      505       87886      499       87892
```

Trial name and number identify each study, and we have information on the authors and the year the trial started. There are also two variables relating to study characteristics: the latitude of the area in which the trial was carried out, and the method of allocating patients to the vaccine and control groups—either at random or in some systematic way. The variables `tcases`, `tnoncases`, `ccases`, and `cnoncases` contain the data from the 2 × 2 table from each study (the number of cases and noncases in the vaccination group and nonvaccination group). The variables `ttotal` and `ctotal` are the total number of individuals (the sum of the cases and noncases) in the vaccine and control groups. Displayed below is the 2 × 2 table for the first study (Canada, 1933):

	cases	noncases	total
treated	6	300	306
control	29	274	303

The risk ratio (RR), log-risk ratio (log-RR), standard error of log-RR (SE log-RR), 95% confidence interval (CI) for log-RR, and 95% CI for RR may be calculated as follows (see, for example, Kirkwood and Sterne 2003).

$$\text{Risk in treated population} = \frac{\texttt{tcases}}{\texttt{ttotal}} = \frac{6}{306} = 0.0196$$

$$\text{Risk in control population} = \frac{\texttt{ccases}}{\texttt{ctotal}} = \frac{29}{303} = 0.0957$$

$$\text{RR} = \frac{\text{Risk in treated population}}{\text{Risk in control population}} = \frac{0.0196}{0.0957} = 0.2049$$

$$\log \text{RR} = \log(\text{RR}) = -1.585$$

$$\begin{aligned} \text{SE}(\log \text{RR}) &= \sqrt{\frac{1}{\texttt{tcases}} + \frac{1}{\texttt{ccases}} - \frac{1}{\texttt{ttotal}} - \frac{1}{\texttt{ctotal}}} \\ &= \sqrt{\frac{1}{6} + \frac{1}{29} - \frac{1}{306} - \frac{1}{303}} = 0.441 \end{aligned}$$

$$95\%\ \text{CI for} \log \text{RR} = \log \text{RR} \pm 1.96 \times \text{SE}(\log \text{RR}) = -2.450 \text{ to } -0.720$$

$$95\%\ \text{CI for RR} = \exp(-2.450) \text{ to } \exp(-0.720) = 0.086 \text{ to } 0.486$$

◁

3 Syntax

`metan` *varlist* [*if*] [*in*] [,
[*binary_data_options* | *continuous_data_options* | *precalculated_effect_estimates_options*]
measure_and_model_options *output_options* *forest_plot_options*]

binary_data_options

`or rr rd fixed random fixedi randomi peto cornfield chi2 breslow nointeger cc(#)`

continuous_data_options

`cohen hedges glass nostandard fixed random nointeger` *precalculated_effect_estimates_options*

fixed random

measure_and_model_options

wgt(*wgtvar*) second(*model* | *estimates_and_description*)
first(*estimates_and_description*)

output_options

by(*byvar*) nosubgroup sgweight log eform efficacy ilevel(#)
olevel(#) sortby(*varlist*)
label([namevar = *namevar*], [yearvar = *yearvar*]) nokeep notable nograph
nosecsub

forest_plot_options

xlabel(#, ...) xtick(#, ...) boxsca(#) textsize(#) nobox nooverall
nowt nostats counts group1(*string*) group2(*string*) effect(*string*) force
lcols(*varlist*) rcols(*varlist*) astext(#) double nohet summaryonly rfdist
rflevel(#) null(#) nulloff favours(*string* # *string*) firststats(*string*)
secondstats(*string*) boxopt(*marker_options*) diamopt(*line_options*)
pointopt(*marker_options* | *marker_label_options*) ciopt(*line_options*)
olineopt(*line_options*) classic nowarning *graph_options*

For a full description of the syntax, see Bradburn, Deeks, and Altman (1998). We will focus on the new options, most of which come under *forest_plot_options*; previously undocumented options such as `by()` (and related options), `breslow`, `cc()`, `nointeger`; and changes to the output such as the display of the I^2 statistic. Syntax will be explained in the appropriate sections.

4 Basic use

4.1 2×2 data

For binary data, the input variables required by `metan` should contain the cells of the 2×2 table; i.e., the number of individuals who did and did not experience the outcome event in the treatment and control groups for each study. When analyzing 2×2 data a range of methods are available. The default is the Mantel–Haenszel method (`fixed`). The inverse-variance fixed-effect method (`fixedi`) or the Peto method for estimating summary odds ratios (`peto`) may also be chosen. The DerSimonian and Laird random-effects method may be specified with `random`. See Deeks, Altman, and Bradburn (2001) for a discussion of these methods.

4.2 Display options

Previous versions of the `metan` command used the syntax `label(namevar =` *namevar*`, yearvar =` *yearvar*`)` to specify study information in the table and forest plot. This syntax still functions but has been superseded by the more flexible `lcols(`*varlist*`)` and `rcols(`*varlist*`)` options. The use of these options is described in more detail in section 5. The option `favours(`*string* `#` *string*`)` allows the user to display text information about the direction of the treatment effect, which appears under the graph (e.g., exposure good, exposure bad). `favours()` replaces the option `b2title()`. The `#` is required to split the two strings, which appear to either side of the null line.

▷ Example

Here we use `metan` to derive an inverse-variance weighted (fixed effect) meta-analysis of the BCG trial data. Risk ratios are specified as the summary statistic, and the trial name and the year the trial started are displayed in the forest plot using `lcols()` (see section 5).

```
. metan tcases tnoncases ccases cnoncases, rr fixedi lcols(trialnam startyr)
> xlabel(0.1, 10) favours(BCG reduces risk of TB # BCG increases risk of TB)
           Study     |     RR    [95% Conf. Interval]     % Weight
---------------------+---------------------------------------------------
Canada               |  0.205       0.086     0.486          1.11
Northern USA         |  0.411       0.134     1.257          0.66
Chicago              |  0.254       0.149     0.431          2.96
Georgia (Sch)        |  1.562       0.374     6.528          0.41
Puerto Rico          |  0.712       0.573     0.886         17.42
Georgia (Comm)       |  0.983       0.582     1.659          3.03
Madanapalle          |  0.804       0.516     1.254          4.22
UK                   |  0.237       0.179     0.312         10.81
South Africa         |  0.625       0.393     0.996          3.83
Haiti                |  0.198       0.078     0.499          0.97
Madras               |  1.012       0.895     1.145         54.58
---------------------+---------------------------------------------------
I-V pooled RR        |  0.730       0.667     0.800        100.00
---------------------+---------------------------------------------------

  Heterogeneity chi-squared = 125.63 (d.f. = 10) p = 0.000
  I-squared (variation in RR attributable to heterogeneity) =  92.0%
  Test of RR=1 : z=   6.75 p = 0.000
```

The output table contains effect estimates (here, RRs), CIs, and weights for each study, followed by the overall (combined) effect estimate. The results for the Canada study are identical to those derived in section 2. Heterogeneity statistics relating to the extent that RRs vary between studies are displayed, including the I^2 statistic, which is a previously undocumented addition. The I^2 statistic (see section 9.1) is the percentage of between-study heterogeneity that is attributable to variability in the true treatment effect, rather than sampling variation (Higgins and Thompson 2004, Higgins et al. 2003). Here there is substantial between-study heterogeneity. Finally, a test of the null hypothesis that the vaccine has no effect (RR=1) is displayed. There is strong evidence against the null hypothesis, but the presence of between-study heterogeneity means that the fixed-effect assumption (that the true treatment effect is the same in each study) is incorrect. The forest plot displayed by the command is shown in figure 1.

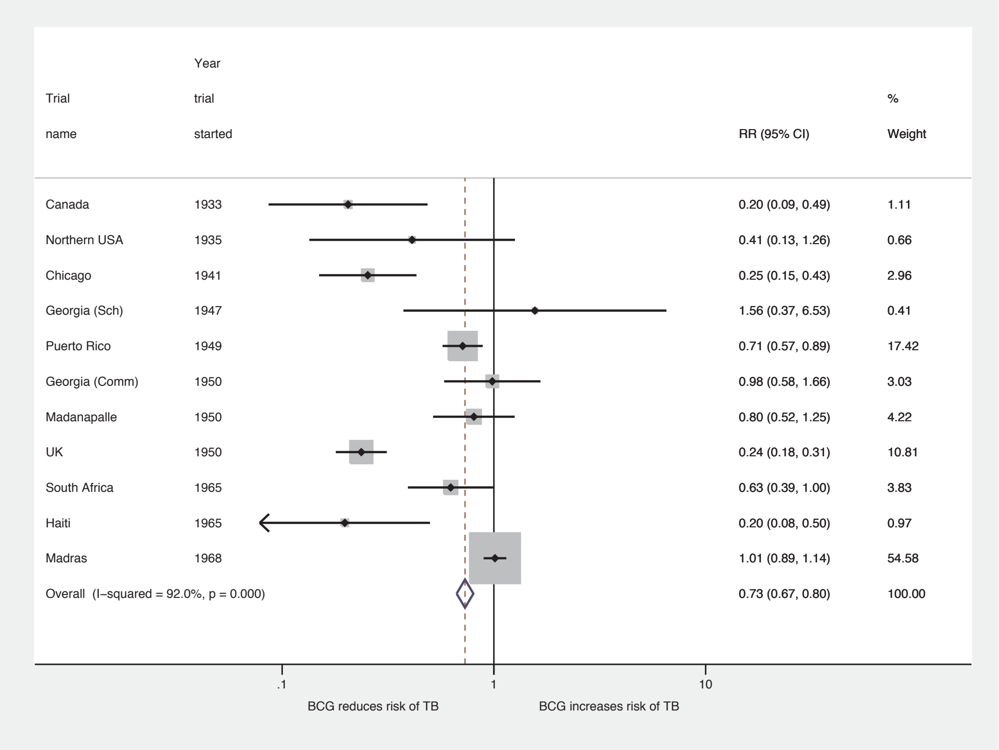


Figure 1. Forest plot displaying an inverse-variance weighted fixed-effect meta-analysis of the effect of BCG vaccine on incidence of tuberculosis.

◁

4.3 Precalculated effect estimates

The `metan` command may also be used to meta-analyze precalculated effect estimates, such as log-odds ratios and their standard errors or 95% CI, using syntax similar to the alternative Stata meta-analysis command `meta` (Sharp and Sterne 1997). Here only the inverse-variance fixed-effect and DerSimonian and Laird random-effects methods are available, because other methods require the 2 × 2 cell counts or the means and standard deviations in each group. The `fixed` option produces an inverse-variance weighted analysis when precalculated effect estimates are analyzed.

When analyzing ratio measures (RRs or odds ratios), the log ratio with its standard error or 95% CI should be used as inputs to the command. The `eform` option can then be used to display the output on the ratio scale (as for the `meta` command).

▷ Example

We will illustrate this feature by generating the log-RR and its standard error in each study from the 2 × 2 data, and then by meta-analyzing these variables.

```
. gen logRR = ln( (tcases/ttotal) / (ccases/ctotal) )
. gen selogRR = sqrt( 1/tcases +1/ccases -1/ttotal -1/ctotal )
. metan logRR selogRR, fixed eform nograph
           Study     |     ES     [95% Conf. Interval]     % Weight
---------------------+---------------------------------------------------
                     (table of study results omitted)
---------------------+---------------------------------------------------
I-V pooled ES        |   0.730        0.667     0.800         100.00
---------------------+---------------------------------------------------

  Heterogeneity chi-squared = 125.63 (d.f. = 10) p = 0.000
  I-squared (variation in ES attributable to heterogeneity) =  92.0%
  Test of ES=1 : z=   6.75 p = 0.000
```

The results are identical to those derived directly from the 2 × 2 data in section 4.1; we would have observed minor differences if the default Mantel–Haenszel method had been used previously. When analyzing precalculated estimates, `metan` does not know what these measures are, so the summary estimate is named “ES” (effect size) in the output.

◁

4.4 Specifying two analyses

`metan` now allows the display of a second meta-analytic estimate in the same output table and forest plot. A typical use is to compare fixed-effect and random-effects analyses, which can reveal the presence of small-study effects. These may result from publication or other biases (Sterne, Gavaghan, and Egger 2000). See Poole and Greenland (1999) for a discussion of the ways in which fixed-effect and random-effects analyses may differ. The syntax is to specify the method for the second meta-analytic estimate as `second(`*method*`)`, where *method* is any of the standard `metan` options.

▷ Example

Here we use `metan` to analyze 2 × 2 data as in section 4.1, specifying an inverse-variance weighted (fixed effect) model for the first method and a DerSimonian and Laird (random effects) model for the second method:

```
. metan tcases tnoncases ccases cnoncases, rr fixedi second(random)
> lcols(trialnam startyr) nograph
           Study        |     RR    [95% Conf. Interval]     % Weight
------------------------+-------------------------------------------------

                      (table of study results omitted)

------------------------+-------------------------------------------------
I-V pooled RR           |   0.730       0.667     0.800        100.00
D+L pooled RR           |   0.508       0.336     0.769        100.00
------------------------+-------------------------------------------------

  Heterogeneity chi-squared = 125.63 (d.f. = 10) p = 0.000
  I-squared (variation in RR attributable to heterogeneity) =  92.0%
  Test of RR=1 : z=   6.75 p = 0.000
```

The results of the second analysis are displayed in the table: a forest plot using the `second()` option is derived in the next section and displayed in figure 2. The protective effect of BCG against tuberculosis appears greater in the random-effects analysis than in the fixed-effect analysis, although CI is wider. This reflects the greater uncertainty in the random-effects analysis, which allows for the true effect of the vaccine to vary between studies. Random-effects analyses give relatively greater weight to smaller studies than fixed-effect analyses, and so these results suggest that the estimated effect of BCG was greater in the smaller studies. It is also possible to supply a precalculated pooled-effect estimate with `second()`; see section 7.2 for details.

◁

5 Displaying data columns in graphs

The options `lcols(`*varlist*`)` and `rcols(`*varlist*`)` produce columns to the left or right of the forest plot. String (character) or numeric variables can be displayed. If numeric variables have value labels, these will be displayed in the graph. If the variable itself is labeled, this will be used as the column header, allowing meaningful names to be used. Up to four lines are used for the heading, so names can be long without taking up too much graph width.

The first variable in `lcols()` is used to identify studies in the table output, and summary statistics and study weight are always the first columns on the right of the forest plot. These can be switched off by using the options `nostats` and `nowt`, but the order cannot be changed.

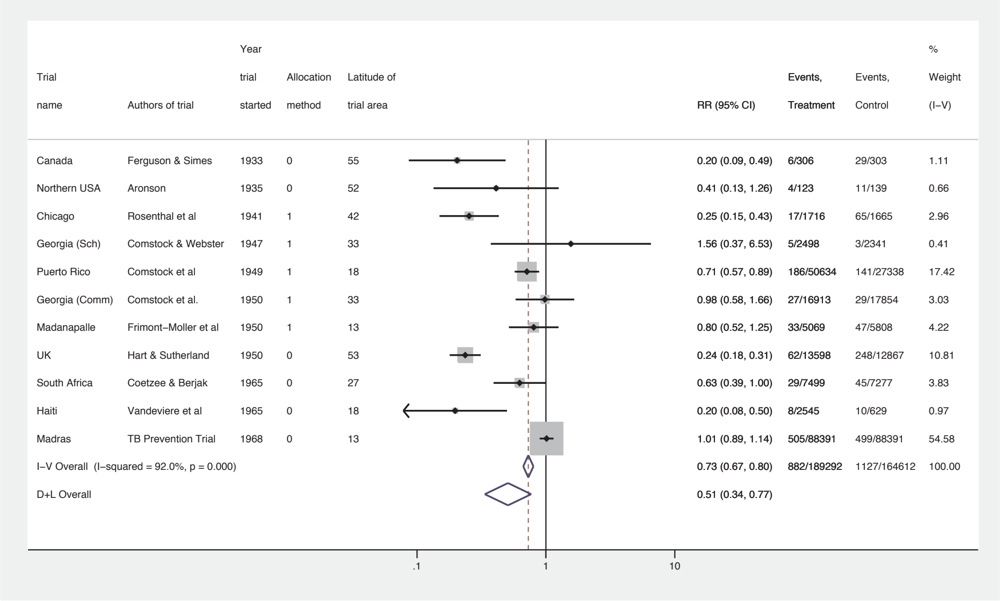
If lengthy string variables are to be displayed, the `double` option may be used to allow output to spread over two lines per study in the forest plot. The percentage of the forest plot given to text may be adjusted using `astext(`*#*`)`, which can be between 10 and 90 (the default is 50).

A previously undocumented option that affects columns is `counts`. When this option is specified, more columns will appear on the right of the graph displaying the raw data; either the 2×2 table for binary data or the sample size, mean, and standard deviation in each group if the data are continuous. The groups may be labeled by using `group1(`*string*`)` and `group2(`*string*`)`, although the defaults *Treatment* and *Control* will often be acceptable for the analysis of randomized controlled trials (RCTs).

▷ Example

We now present an example command that uses these features, as well as the `second()` option. The resulting forest plot is displayed in figure 2:

```
. metan tcases tnoncases ccases cnoncases, rr fixedi second(random)
> lcols(trialnam authors startyr alloc latitude) counts astext(70)
> textsize(200) boxsca(80) xlabel(0.1,10) notable xsize(10) ysize(6)
```



		Year						%
Trial		trial	Allocation	Latitude of		Events,	Events,	Weight
name	Authors of trial	started	method	trial area	RR (95% CI)	Treatment	Control	(I–V)
Canada	Ferguson & Simes	1933	0	55	0.20 (0.09, 0.49)	6/306	29/303	1.11
Northern USA	Aronson	1935	0	52	0.41 (0.13, 1.26)	4/123	11/139	0.66
Chicago	Rosenthal et al	1941	1	42	0.25 (0.15, 0.43)	17/1716	65/1665	2.96
Georgia (Sch)	Comstock & Webster	1947	1	33	1.56 (0.37, 6.53)	5/2498	3/2341	0.41
Puerto Rico	Comstock et al	1949	1	18	0.71 (0.57, 0.89)	186/50634	141/27338	17.42
Georgia (Comm)	Comstock et al.	1950	1	33	0.98 (0.58, 1.66)	27/16913	29/17854	3.03
Madanapalle	Frimont–Moller et al	1950	1	13	0.80 (0.52, 1.25)	33/5069	47/5808	4.22
UK	Hart & Sutherland	1950	0	53	0.24 (0.18, 0.31)	62/13598	248/12867	10.81
South Africa	Coetzee & Berjak	1965	0	27	0.63 (0.39, 1.00)	29/7499	45/7277	3.83
Haiti	Vandeviere et al	1965	0	18	0.20 (0.08, 0.50)	8/2545	10/629	0.97
Madras	TB Prevention Trial	1968	0	13	1.01 (0.89, 1.14)	505/88391	499/88391	54.58
I–V Overall (I–squared = 92.0%, p = 0.000)					0.73 (0.67, 0.80)	882/189292	1127/164612	100.00
D+L Overall					0.51 (0.34, 0.77)			

Figure 2. Forest plot displaying an inverse-variance weighted fixed-effect meta-analysis of the effect of BCG vaccine on incidence of tuberculosis. Columns of data are displayed in the plot.

Note the specification of x-axis labels and text and box sizes. The graph is also reshaped by using the standard Stata graph options `xsize()` and `ysize()`; see section 10.2 for more details. Box and text sizes are expressed as a percentage of standard size with the default as 100, such that 50 will halve the size and 200 will double it.

◁

6 by() processing

A major addition to `metan` is the ability to perform stratified or subgroup analyses. These may be used to investigate the possibility that treatment effects vary between subgroups; however, formal comparisons between subgroups are best performed by using meta-regression; see Harbord and Higgins (2008) or Higgins and Thompson (2004). We may also want to display results for different groups of studies in the same plot, even though it is inappropriate to meta-analyze across these groups.

6.1 Syntax and options for by()

`nooverall` specifies that the overall estimate not be displayed, for example, when it is inappropriate to meta-analyze across groups.

`sgweight` requests that weights be displayed such that they sum to 100% within each subgroup. This option is invoked automatically with `nooverall`.

`nosubgroup` specifies that studies be arranged by the subgroup specified, but estimates for each subgroup not be displayed.

`nosecsub` specifies that subestimates using the method defined by `second()` not be displayed.

`summaryonly` specifies that individual study estimates not be displayed, for example, to produce a summary of different groups in a compact graph.

▷ Example

Fine (1995) suggested that there is a relationship between the effect of BCG and the latitude of the area in which the trial was conducted. Here we may want to use meta-regression to further investigate this tendency (see Harbord and Higgins 2008). To illustrate the `by()` option, we will classify the studies into three groups defined by latitude. We define these groups as tropical (≤23.5 degrees), midlatitude (between 23.5 and 40 degrees) and northern (≥40 degrees).

```
. gen lat_cat = ""
(11 missing values generated)

. replace lat_cat = "Tropical, < 23.5 latitude" if latitude <= 23.5
lat_cat was str1 now str27
(4 real changes made)

. replace lat_cat = "23.5-40 latitude" if latitude > 23.5 & latitude < 40
(3 real changes made)

. replace lat_cat = "Northern, > 40 latitude" if latitude >= 40 & latitude < .
(4 real changes made)

. assert lat_cat != ""

. label var lat_cat "Latitude region"
```

(*Continued on next page*)

```
. metan tcases tnoncases ccases cnoncases, rr fixedi second(random) nosecsub
> lcols(trialnam startyr latitude) astext(60) by(lat_cat) xlabel(0.1,10)
> xsize(10) ysize(8)
           Study     |     RR    [95% Conf. Interval]     % Weight
-------------------------------------------------------------------------

     Northern, > 40 lat
Canada               |    0.205     0.086     0.486          1.11
Northern USA         |    0.411     0.134     1.257          0.66
Chicago              |    0.254     0.149     0.431          2.96
UK                   |    0.237     0.179     0.312         10.81
 Sub-total           |
  I-V pooled RR      |    0.243     0.193     0.306         15.54
-------------------------------------------------------------------------
     23.5-40 latitude
Georgia (Sch)        |    1.562     0.374     6.528          0.41
Georgia (Comm)       |    0.983     0.582     1.659          3.03
South Africa         |    0.625     0.393     0.996          3.83
 Sub-total           |
  I-V pooled RR      |    0.795     0.567     1.114          7.27
-------------------------------------------------------------------------
     Tropical, < 23.5 l
Puerto Rico          |    0.712     0.573     0.886         17.42
Madanapalle          |    0.804     0.516     1.254          4.22
Haiti                |    0.198     0.078     0.499          0.97
Madras               |    1.012     0.895     1.145         54.58
 Sub-total           |
  I-V pooled RR      |    0.904     0.815     1.003         77.19
-------------------------------------------------------------------------
Overall              |
  I-V pooled RR      |    0.730     0.667     0.800        100.00
  D+L pooled RR      |    0.508     0.336     0.769
-------------------------------------------------------------------------

Test(s) of heterogeneity:
               Heterogeneity  degrees of
                 statistic     freedom      P    I-squared**
Northern, > 40 lat  1.06         3      0.787      0.0%
23.5-40 latitude   2.51          2      0.285     20.2%
Tropical, < 23.5 l 18.42         3      0.000     83.7%
Overall            125.63         10      0.000     92.0%
Overall Test for heterogeneity between sub-groups:
                   103.64          2      0.000

** I-squared: the variation in RR attributable to heterogeneity)

Considerable heterogeneity observed (up to 83.7%) in one or more sub-groups,
Test for heterogeneity between sub-groups likely to be invalid

Significance test(s) of RR=1

Northern, > 40 lat  z= 12.00     p = 0.000
23.5-40 latitude    z=  1.33     p = 0.183
Tropical, < 23.5 l  z=  1.90     p = 0.058
Overall               z=  6.75     p = 0.000
-------------------------------------------------------------------------
```

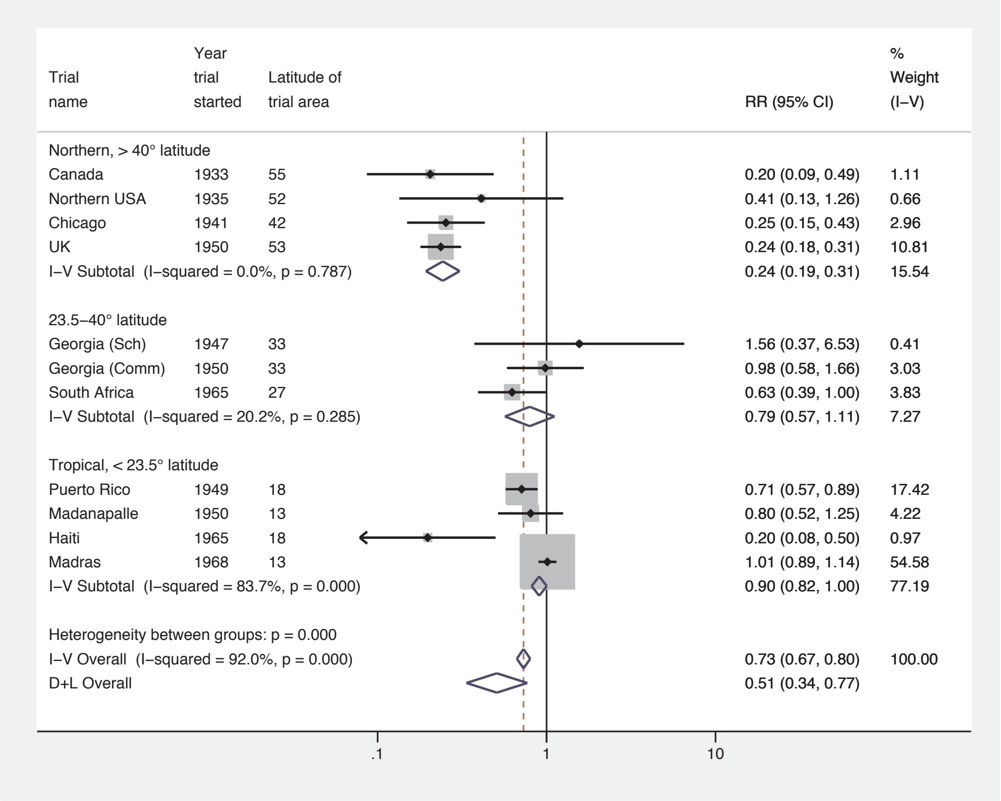


Figure 3. Forest plot displaying an inverse-variance weighted fixed-effect meta-analysis of the effect of BCG vaccine on incidence of tuberculosis. Results are stratified by latitude region, and the overall random-effects estimate is also displayed.

The output table is now stratified by latitude group, and pooled estimates for each group are displayed. Tests of heterogeneity and the null hypothesis are displayed for each group and overall. With the inverse-variance method, a test of heterogeneity between groups is also displayed; note the warning in the output that the test may be invalid because of within-subgroup heterogeneity. Output is similar in the forest plot, displayed in figure 3. Examining each subgroup in turn, it appears that much of the heterogeneity is accounted for by latitude: for two of the groups there is little or no evidence of heterogeneity. The only group to show a strong treatment effect is the $\geq$40 degree group.

◁

The test for between-group heterogeneity is an issue of current debate, as it is strictly valid only when using the fixed-effect inverse-variance method, and *p*-values will be too small if there is heterogeneity within any of the subgroups. Therefore, the test is performed only with the inverse-variance method (`fixedi`), and warnings will appear if there is evidence of within-group heterogeneity. Despite these caveats, this method is better than other, seriously flawed, methods such as testing the significance of a treatment effect in each group rather than testing for differences between the groups. As explained at the start of this section, meta-regression is the best way to examine and test for between-group differences.

7 User-defined analyses

7.1 Study weights

The `wgt(`*wgtvar*`)` option allows the studies to be combined by using specific weights that are defined by the variable *wgtvar*. The user must ensure that the weights chosen are meaningful. Typical uses are when analyzing precalculated effect estimates that require weights that are not based on standard error or to assess the robustness of conclusions by assigning alternative weights.

7.2 Pooled estimates

Pooled estimates may be derived by using another package and presented in a forest plot by using the `first()` option to supply these to the `metan` command. Here `wgt(`*wgtvar*`)` is used merely to specify box sizes in the forest plot, no heterogeneity statistics are produced, and no values are returned. When using this feature, stratified analyses are not allowed.

An alternative method is to provide the user-supplied meta-analytic estimate by using the `second()` option. Data are analyzed by using standard methods, and the resulting pooled estimate is displayed together with the user-defined estimate (which need not be derived by using `metan`), allowing a comparison. When using this feature, the option `nosecsub` is invoked, as stratification using the user-defined method is not possible.

When these options are specified, the user must supply the pooled estimate with its standard error or CI and a method label. The user may also supply text to be displayed at the bottom of the forest plot, in the position normally given to heterogeneity statistics, using `firststats(`*string*`)` and `secondstats(`*string*`)`.

▷ Example

The BCG data were analyzed by using a fully Bayesian random-effects model with WinBUGS software (Lunn et al. 2000). This analysis used the methods described by Warn, Thompson, and Spiegelhalter (2002) to deal with RRs. The chosen model incorporated a noninformative prior (mean 0, precision 0.001). The resulting RR of 0.518 (95% CI: 0.300, 0.824) is similar to that derived from a DerSimonian and Laird random-effects analysis. However, the CI from the Bayesian analysis is wider, because it allows for the uncertainty in estimating the between-study variance. The following syntax supplies the summary estimates in `second()` and compares this result with the random-effects analysis. The resulting forest plot is displayed in figure 4.

```
. metan logRR selogRR, random second(-.6587 -1.205 -.1937 Bayes)
> secondstats(Noninformative prior: d~dnorm(0.0, 0.001)) eform
> notable astext(60) textsize(130) lcols(trialnam startyr latitude)
> xlabel(0.1,10)
```

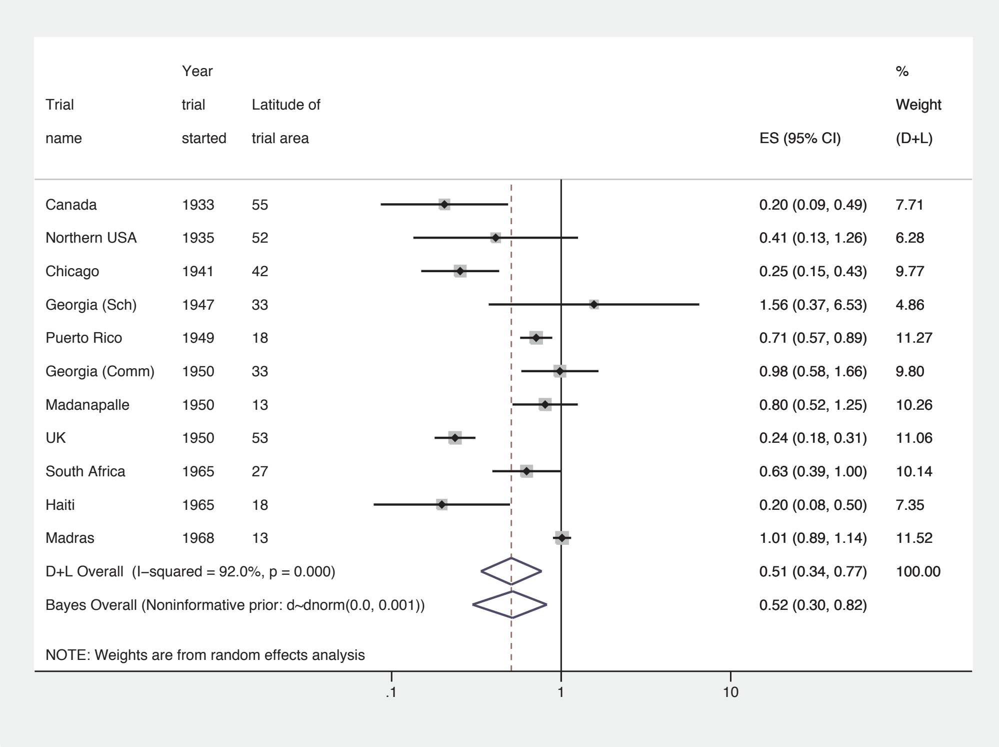


Figure 4. Forest plot displaying a fully Bayesian meta-analysis of the effect of BCG vaccine on incidence of tuberculosis. A noninformative prior has been specified, resulting in a pooled-effect estimate similar to the random-effects analysis.

◁

8 New analysis options

Here we discuss previously undocumented options added to `metan` since its original publication.

8.1 Dealing with zero cells

The cc(#) option allows the user to choose what value (if any) is to be added to the cells of the 2×2 table for a study in which one or more of the cell counts equals zero. Here the default is to add 0.5 to all cells of the 2×2 table for the study (except for the Peto method, which does not require a correction). This approach has been criticized, and other approaches (including making no correction) may be preferable (see Sweeting, Sutton, and Lambert [2004] for a discussion). The number declared in cc(#) must be between zero and one and will be added to each cell. When no events are recorded and RRs or odds ratios are to be combined the study is omitted, although for risk differences the effect is still calculable and the study is included. If no adjustment is made in the presence of zero cells, odds ratios and their standard errors cannot be calculated. Risk ratios and their standard errors cannot be calculated when the number of events in either the treatment or control group is zero.

8.2 Noninteger sample size

The nointeger option allows the number of observations in each arm (cell counts for binary data or the number of observations for continuous data) to be noninteger. By default, the sample size is assumed to be a whole number for both binary and continuous data. However, it may make sense for this not to be so, for example, to use a more flexible continuity correction with a different number added to each cell or when the meta-analysis incorporates cluster randomized trials and the effective-sample size is less than the total number of observations.

8.3 Breslow and Day test for heterogeneity

The breslow option can be used to perform the Breslow–Day test for heterogeneity of the odds ratio (Breslow and Day 1980). A review article by Reis, Hirji, and Afifi (1999) compared several different tests of heterogeneity and found this test to perform well in comparison to other asymptotic tests.

9 New output

9.1 The $\mathbf{I^2}$ statistic

metan now displays the I^2 statistic as well as Cochran's Q to quantify heterogeneity, based on the work by Higgins and Thompson (2004) and Higgins et al. (2003). Briefly, I^2 is the percentage of variation attributable to heterogeneity and is easily interpretable. Cochran's Q can suffer from low power when the number of studies is low or excessive power when the number of studies is large. I^2 is calculated from the results of the meta-analysis by

$$I^2 = 100\% \times \frac{(Q - \mathrm{df})}{Q}$$

where Q is Cochran's heterogeneity statistic and df is the degrees of freedom. Negative values of I^2 are set to zero so that I^2 lies between 0% and 100%. A value of 0% indicates no observed heterogeneity, and larger values show increasing heterogeneity. Although there can be no absolute rule for when heterogeneity becomes important, Higgins et al. (2003) tentatively suggest adjectives of low for I^2 values between 25%–50%, moderate for 50%–75%, and high for $\geq$75%.

9.2 Prediction interval for the random-effects distribution

The presentation of summary random-effects estimates may sometimes be misleading, as the CI refers to the average true treatment effect, but this is assumed under the random-effects model to vary between studies. A CI derived from a larger number of studies exhibiting a high degree of heterogeneity could be of similar width to a CI derived from a smaller number of more homogeneous studies, but in the first situation, we will be much less sure of the range within which the treatment effect in a new study will lie (Higgins and Thompson 2001). The prediction interval for the treatment effect in a new trial may be approximated by using the formula

$$\text{mean} \pm t_{\text{df}} \times \sqrt{(\text{se}^2 + \tau^2)}$$

where t is the appropriate centile point (e.g., 95%) of the t distribution with $k-2$ degrees of freedom, se^2 is the squared standard error, and τ^2 the between-study variance. This incorporates uncertainty in the location and spread of the random-effects distribution. The approximate prediction interval can be displayed in the forest plot, with lines extending from the summary diamond, by using the option `rfdist`. With $\leq$2 studies, the distribution is inestimable and effectively infinite; thus the interval is displayed with dotted lines. When heterogeneity is estimated to be zero, the prediction interval is still slightly wider than the summary diamond as the t statistic is always greater than the corresponding normal deviate. The coverage (e.g., 90%, 95%, or 99%) for the interval may be set by using the command `rflevel(#)`.

▷ Example

Here we display the prediction intervals corresponding to the stratified analyses derived in section 6.1. The resulting forest plot is displayed in figure 5.

```
. metan tcases tnoncases ccases cnoncases, rr random rfdist
> lcols(trialnam startyr latitude) astext(60) by(lat_cat) xlabel(0.1,10)
> xsize(10) ysize(8) notable
```

(Continued on next page)

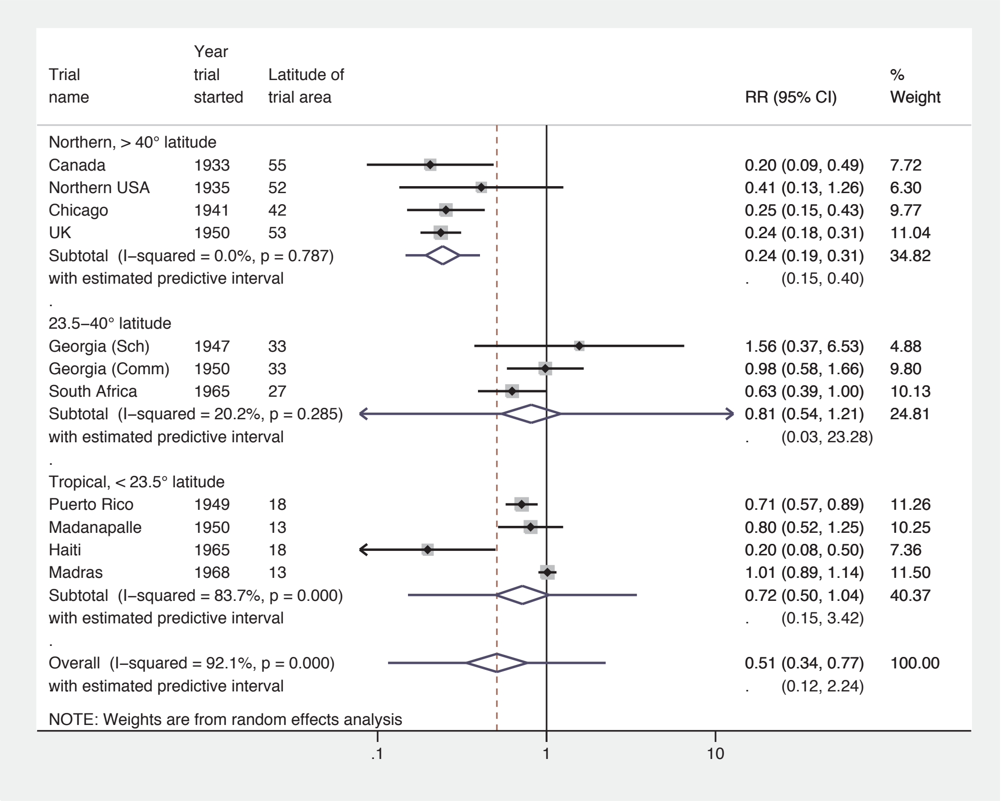


Figure 5. Forest plot displaying a random-effects meta-analysis of the effect of BCG vaccine on incidence of tuberculosis. Results are stratified by latitude region and the prediction interval for a future trial is displayed for each and overall.

◁

9.3 Vaccine efficacy

Results from the analysis of 2×2 data from vaccine trials may be reexpressed as the *vaccine efficacy* (also known as the *relative-risk reduction*); defined as the proportion of cases that would have been prevented in the placebo group had they received the vaccination (Kirkwood and Sterne 2003). The formula is

$$\begin{aligned}\text{Vaccine efficacy (VE)} &= 100\% \times \left(1 - \frac{\text{risk of disease in vaccinated}}{\text{risk of disease in unvaccinated}}\right)\\ &= 100\% \times (1 - \text{RR})\end{aligned}$$

In `metan`, data are entered in the same way as any other analysis of 2 × 2 data and the option `efficacy` added. Results are displayed as odds ratios or RRs in the table and forest plot, but another column is added to the plot showing the results reexpressed as vaccine efficacy.

▷ Example

The BCG data are reanalyzed here, with results also displayed in terms of vaccine efficacy. The resulting forest plot is displayed in figure 6.

```
. metan tcases tnoncases ccases cnoncases, rr random efficacy
> lcols(trialnam startyr) textsize(150) notable xlabel(0.1, 10)
```

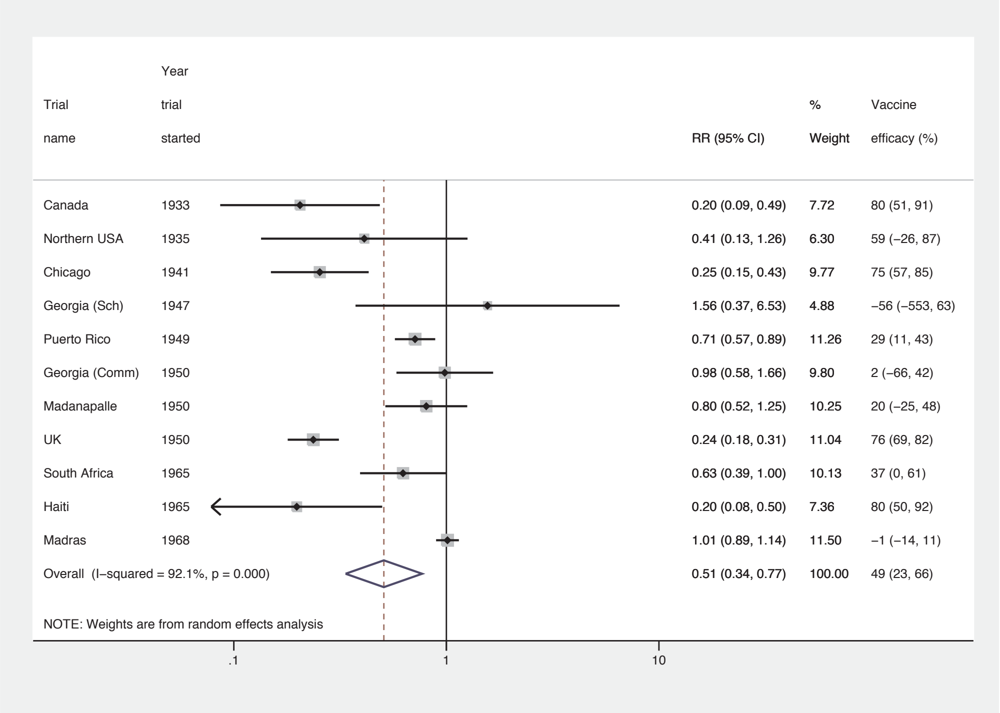


Figure 6. Forest plot displaying a random-effects meta-analysis of the effect of BCG vaccine on incidence of tuberculosis. Results are also displayed in terms of vaccine efficacy; estimates with a RR of greater than 1 produce a negative vaccine efficacy.

◁

10 More graph options

10.1 metan graph options

Previous users of `metan` may find that they do not like the new box style and prefer a solid black box without the point estimate marker. The option `classic` changes back to this style. There are also options available to change the boxes, diamonds, and other lines. This is achieved by using options that change the standard graph commands that `metan` uses. For instance, the vertical line representing the overall effect may be changed using `olineopt()`, which can take standard Stata *line_options* such as `lwidth()`, `lcolor()`, and `lpattern()`. Boxes are weighted markers and not much can be changed, although shape and color may be modified by using *marker_options* in the `boxopt()` option, such as `msymbol()` and `mcolor()`, or we can dispense with the boxes entirely by using the option `nobox`. The point estimate markers have more flexibility and may also be modified by using *marker_options* in the `pointopt()` option; for instance, labels may by attached to them by using `mlabel()`. The CIs and diamonds may be changed by using *line_options* in the options `ciopt()` and `diamopt()`. For more details, see the `metan` help file and the Stata *Graphics Reference Manual* ([G] **graph**).

▷ Example

Here many aspects of the graph are changed and a raw data variable is defined (as in `counts`) and attached to the point estimates in the graph. The resulting graph is not shown here, but a similar application is shown in section 10.3.

```
. gen counts = string(tcases) + "/" + string(tcases+tnoncases) + "," +
> string(ccases) + "/" + string(ccases+cnoncases)
. metan tcases tnoncases ccases cnoncases, rr fixedi second(random) nosecsub
> notable olineopt(lwidth(thick) lcolor(navy) lpattern(dot))
> boxopt(msymbol(triangle) mcolor(dkgreen))
> pointopt(mlabel(counts) mlabsize(tiny) mlabposition(5))
```

◁

10.2 Overall graph options

Any graph options that come under the *overall*, *note*, and *caption* sections of Stata's `graph twoway` command may be added to a `metan` command, and the x axis (and y axis if required) may have a title added. The options `aspect()` or `xsize()` and `ysize()` may be used to specify different aspect ratios (e.g., portrait). The default aspect ratio of a Stata graph is around 0.7 (height/width), and `metan` tries to stick to this shape; although graphs that are more naturally displayed as long or wide will be reshaped to some degree. Use of the above options will control this more precisely.

Finally, the use of schemes is also supported. As colors of boxes and so on are defined within `metan`, these will not always give the desired result but may produce some interesting effects. Try, for example, using the scheme `economist`. More on schemes can be found in [G] **schemes intro**.

10.3 Notes on graph building

It can be useful to declare local or global macros that contain portions of code that are frequently used. For example, if the forest plot always has triangular "boxes" in forest green, contains the same columns of data, and so on, global macros may be declared for these bits of code. These can then be reused for a series of meta-analyses to specify the look and contents of the graphs. These could also be declared in an ado-file so that they are ready to use in every Stata session. This idea is similar to using Stata graph schemes.

▷ Example

Macros are defined to control various aspects of the graph and then used in the `metan` command. The resulting forest plot is displayed in figure 7.

```
. global metamethod rr fixedi second(random) nosecsub
. global metacolumns lcols(trialnam startyr latitude) astext(60)
. global metastyle boxopt(mcolor(forest_green) msymbol(triangle))
> pointopt(msymbol(smtriangle) mcolor(gold) msize(tiny)
> mlabel(counts) mlabsize(tiny) mlabposition(2) mlabcolor(brown))
> diamopt(lcolor(black) lwidth(medthick)) graphregion(fcolor(gs10)) boxsca(80)
. global metaopts favours(decreases TB # increases TB)
> xlabel(0.1, 0.2, 0.5, 2, 5, 10) notable
. metan tcases tnoncases ccases cnoncases,
> $metamethod $metacolumns $metastyle $metaopts by(lat_cat) xsize(10) ysize(8)
```

(*Continued on next page*)

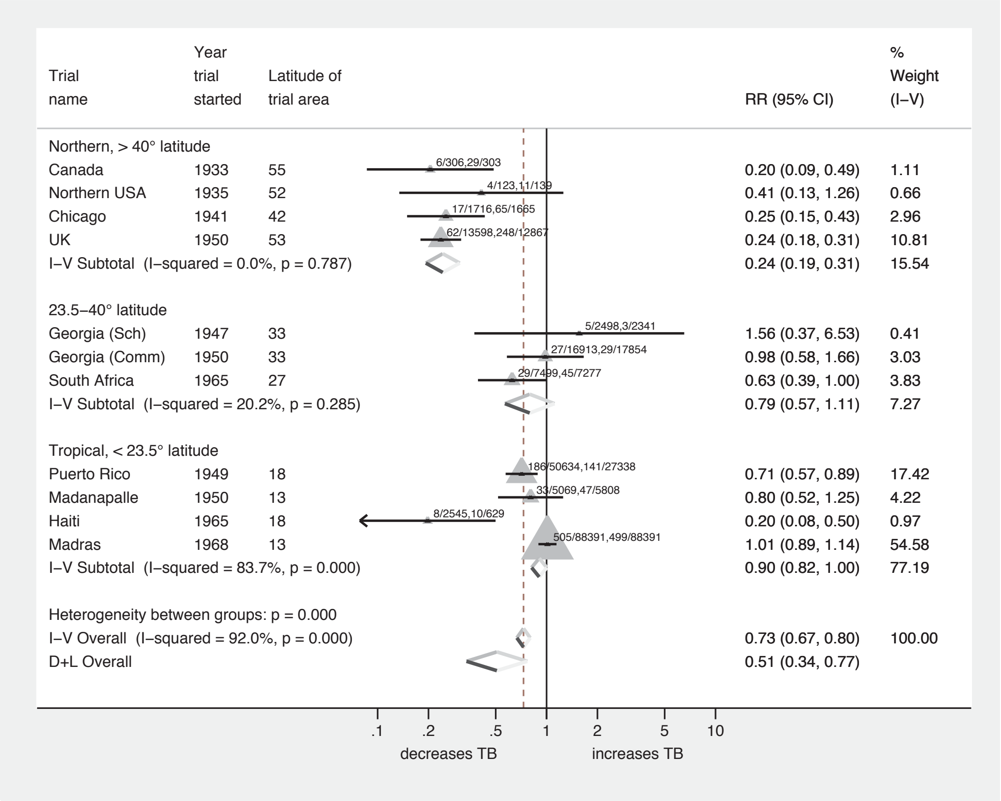


Figure 7. Forest plot displaying an inverse-variance weighted fixed-effect meta-analysis of the effect of BCG vaccine on incidence of tuberculosis. Results are stratified by latitude region, and the overall random-effects estimate is also displayed. Various options have been used to change the display of the graph.

◁

11 Variables and results produced by metan

11.1 Variables generated

When odds ratios (OR) or RRs are combined from 2×2 data and the `log` option is not used, the SE log-OR or log-RR is saved in a variable named `_selogES`, to make clear that it is the SE log-OR or RR and not on the same scale. If the `log` option is used, the standard error is named `_seES`, as it is on the same scale as the estimate itself. In both cases, the estimate is called `_ES`.

It is possible to calculate the standard error of ORs and RRs by the delta method; this is what Stata does, for example, with the results reported by the logistic command.

However, the distribution of ratios is in general highly skewed, and for this reason, `metan` does not attempt to record the standard error of either the OR or RR.

Absolute measures (risk differences or mean differences) are symmetric and may be assumed to be normally distributed via the central limit theorem. Here `metan` stores these quantities in `_ES` and their standard errors in `_seES`. The derived variables incorporate the correction for zero cells (see section 8.1).

`_ES`	Effect size (ES)
`_seES`	Standard error of ES
`_selogES`	Standard error of log ES
`_LCI`	Lower confidence limit for ES
`_UCI`	Upper confidence limit for ES
`_WT`	Study percentage weight
`_SS`	Study sample size

11.2 Saved results (macros)

As with many Stata commands, macros are left behind containing the results of the analysis. If two methods are specified by using the option `second()`, some of these are repeated; for example, `r(ES)` and `r(ES_2)` give the pooled-effects estimates for each method. Subgroup statistics when using the `by()` option are not saved; if these are required for storage, it is recommended that a program be written that analyzes subgroups separately (perhaps using the `nograph` and `notable` options).

(*Continued on next page*)

Name	Second	Description
r(ES)	r(ES_2)	pooled-effect size (if the log option is specified with or or rr, this is the pooled log-OR or log-RR)
r(seES)	r(seES_2)	standard error of pooled-effect size with symmetrical CI, i.e., mean differences, risk difference, log-OR, and log-RR using log option
r(selogES)	r(selogES_2)	standard error of log-OR or log-RR when ORs or RRs are combined without the log option
r(ci_low)	r(ci_low_2)	lower CI of pooled-effect size
r(ci_upp)	r(ci_upp_2)	upper CI of pooled-effect size
r(z)		Z-value of effect size
r(p_z)		p-value for significance of effect size
r(het)		chi-squared test for heterogeneity
r(df)		degrees of freedom (number of informative studies minus 1)
r(p_het)		p-value for significance of test for heterogeneity
r(i_sq)		the I^2 statistic
r(tau2)		estimated between-study variance (random-effects analyses only)
r(chi2)		chi-squared test for significance of odds ratio (fixed-effect OR only)
r(p_chi2)		p-value for the above test
r(rger)		overall event rate, group 1 (if binary data are combined)
r(cger)		overall event rate, group 2 (see above)
r(measure)		effect measure (e.g., RR, SMD)
r(method_1)	r(method_2)	analysis method (e.g., M-H, D+L)

12 References

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 *Stata Technical Bulletin Reprints*, vol. 8, pp. 86–100. College Station, TX: Stata Press.

Breslow, N. E., and N. E. Day. 1980. *Statistical Methods in Cancer Research: Volume I—The Analysis of Case-Control Studies.* Lyon, UK: International Agency for Research on Cancer.

Colditz, G. A., T. F. Brewer, C. S. Berkey, M. E. Wilson, E. Burdick, H. V. Fineberg, and F. Mosteller. 1994. Efficacy of BCG vaccine in the prevention of tuberculosis.

Meta-analysis of the published literature. *Journal of the American Medical Association* 271: 698–702.

Deeks, J. J., D. G. Altman, and M. J. Bradburn. 2001. Statistical methods for examining heterogeneity and combining results from several studies in meta-analysis. In *Systematic Reviews in Health Care: Meta-analysis in context*, ed. M. Egger, G. D. Smith, and D. G. Altman, 285–321.

Fine, P. E. 1995. Variation in protection by BCG: implications of and for heterologous immunity. *Lancet* 346: 1339–1345.

Harbord, R. M., and J. P. T. Higgins. 2008. Meta-regression in Stata. *Stata Journal*. Forthcoming.

Higgins, J. P. T., and S. G. Thompson. 2001. Presenting random effects meta-analyses: where are we going wrong? In *9th International Cochrane Colloquium*. Lyon, France.

———. 2004. Controlling the risk of spurious findings from meta-regression. *Statistics in Medicine* 23: 1663–1682.

Higgins, J. P. T., S. G. Thompson, J. J. Deeks, and D. G. Altman. 2003. Measuring inconsistency in meta-analyses. *British Medical Journal* 327: 557–560.

Kirkwood, B. R., and J. A. C. Sterne. 2003. *Essential Medical Statistics.* 2nd ed. Oxford: Blackwell Science.

Lunn, D. J., A. Thomas, N. Best, and D. Spiegelhalter. 2000. WinBUGS – A Bayesian modelling framework: concepts, structure and extensibility. *Statistics and Computing* 10: 325–337.

Poole, C., and S. Greenland. 1999. Random-effects meta-analyses are not always conservative. *American Journal of Epidemiology* 150: 469–475.

Reis, I., K. F. Hirji, and A. Afifi. 1999. Exact and asymptotic tests for homogeneity in several 2 × 2 tables. *Statistics in Medicine* 18: 893–906.

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

Sterne, J. A. C., D. Gavaghan, and M. Egger. 2000. Publication and related bias in meta-analysis: power of statistical tests and prevalence in the literature. *Journal of Clinical Epidemiology* 53: 1119–1129.

Sweeting, M. J., A. J. Sutton, and P. C. Lambert. 2004. What to add to nothing? Use and avoidance of continuity corrections in meta-analysis of rare events. *Statistics in Medicine* 23: 1351–1375.

Warn, D. E., S. G. Thompson, and D. J. Spiegelhalter. 2002. Bayesian random effects meta-analysis of trials with binary outcomes: methods for absolute risk difference and relative risk scales. *Statistics in Medicine* 21: 1601–1623.

About the authors

Ross Harris is a research associate in medical statistics at the Department of Social Medicine, University of Bristol, Bristol, UK. His research interests include meta-analysis, particularly summarizing dose–response relationships from published data and examining sources of bias in randomized trials; epidemiology of HIV and AIDS and methods for dealing with missing data.

Michael Bradburn is a medical statistician in the Health Services Research Unit, University of Sheffield, Sheffield, UK, and wrote most of the original `metan` code. His current research is focused on randomized trials.

Jonathan Deeks is professor of health statistics at the University of Birmingham, Birmingham, UK, and head of the Medical Statistics Group and Diagnostic Evaluation Support Unit in the Department of Public Health. His work has focused on issues related to meta-analysis and more recently diagnostic test evaluations, including both clinical applications and methodological developments. Currently, Jon Deeks is the elected representative for Methods Groups on the Steering Group of the Cochrane Collaboration and is leading the implementation of Reviews of Diagnostic Test Accuracy in the Cochrane Collaboration.

Roger Harbord is a research associate in medical statistics in the Department of Social Medicine, University of Bristol, Bristol, UK. He is a coconvenor of the Cochrane Collaborations Screening and Diagnostic Tests Methods Group.

Douglas Altman is professor of statistics in medicine at the University of Oxford, Oxford, UK, and founding director of the Centre for Statistics in Medicine. His research interests include the use and abuse of statistics in medical research, studies of prognosis, regression modeling, systematic reviews and meta-analysis, randomized trials, reporting guidelines, and studies of medical measurement.

Jonathan Sterne is professor of medical statistics and epidemiology in the Department of Social Medicine, University of Bristol, Bristol, UK. His research interests include statistical methods for epidemiology and health services research, meta-analysis and systematic reviews, clinical epidemiology of HIV and AIDS in the era of antiretroviral therapy, and the epidemiology of asthma and allergic diseases.

The Stata Journal (2008)
8, Number 1, pp. 29–48

A tool for deterministic and probabilistic sensitivity analysis of epidemiologic studies

Nicola Orsini
Division of Nutritional Epidemiology
Institute of Environmental Medicine
Karolinska Institutet
Stockholm, Sweden
nicola.orsini@ki.se

Rino Bellocco
Department of Statistics
University of Milano-Bicocca
Milano, Italy

Matteo Bottai
Department of Epidemiology and Biostatistics
Arnold School of Public Health
University of South Carolina
Columbia, SC

Alicja Wolk
Division of Nutritional Epidemiology
Institute of Environmental Medicine
Karolinska Institutet
Stockholm, Sweden

Sander Greenland
Departments of Epidemiology and Statistics
University of California, Los Angeles
Los Angeles, CA

Abstract. Classification errors, selection bias, and uncontrolled confounders are likely to be present in most epidemiologic studies, but the uncertainty introduced by these types of biases is seldom quantified. The authors present a simple yet easy-to-use Stata command to adjust the relative risk for exposure misclassification, selection bias, and an unmeasured confounder. This command implements both deterministic and probabilistic sensitivity analysis. It allows the user to specify a variety of probability distributions for the bias parameters, which are used to simulate distributions for the bias-adjusted exposure–disease relative risk. We illustrate the command by applying it to a case–control study of occupational resin exposure and lung-cancer deaths. By using plausible probability distributions for the bias parameters, investigators can report results that incorporate their uncertainties regarding systematic errors and thus avoid overstating their certainty about the effect under study. These results can supplement conventional results and can help pinpoint major sources of conflict in study interpretations.

Keywords: st0138, episens, episensi, sensitivity analysis, unmeasured confounder, misclassification, bias, epidemiology

1 Introduction

Conventional statistical methods to estimate exposure–disease associations from observational studies are based on several assumptions, such as no measurement error and no selection bias (i.e., selection, participation, and retention of subjects are purely random). If the associations are interpreted as causal effects, another assumption of random-exposure assignment within levels of controlled covariates is also implicitly made. When such assumptions are not met, tests and estimates for the association between exposure and disease are likely to be biased and may fail to capture most of the uncertainty about the estimated parameter (Greenland 2005).

There are many proposed methods to adjust uncertainty assessments for unmeasured sources of bias or systematic error (Chu et al. 2006; Eddy, Hasselblad, and Shachter 1992; Fox, Lash, and Greenland 2005; Greenland 2001; Greenland 2003b; Greenland 2005; Greenland and Lash 2008; Hoffman and Hammonds 1994; Lash and Fink 2003; Phillips 2003; and Steenland and Greenland 2004). Nonetheless, few published papers in epidemiologic journals use quantitative methods to investigate the role of potential bias in the observed findings (Jurek et al. 2006). To facilitate the use of both deterministic and probabilistic sensitivity analysis, we present a flexible and easy-to-use tool to assess the uncertainty of exposure–disease associations due to misclassification of the exposure, selection bias, and unmeasured confounding. The proposed tool is implemented as a one-line Stata command. Here we illustrate the use of the tool by analyzing a published medical study reporting a positive association between occupational resin exposure and lung-cancer deaths in a case–control study.

2 Methods

We consider the simplest situation in which there are only two factors: the disease and the exposure. Each factor is considered as being either present or absent, and so the data can be summarized in a 2×2 table. The term *relative risk* (RR) will be used as a generic term for the risk ratio (ratio of proportions getting disease), rate ratio (ratio of person–time incidence rates), and odds ratio (ratio of odds, most often used in case–control data). The formulas implemented for correction of the observed RR due to misclassification of the exposure, selection bias, and a binary unmeasured or uncontrolled confounder are described in detail elsewhere (Greenland 1996; Greenland and Lash 2008).

Deterministic (ordinary or classical) sensitivity analysis provides an external adjustment of the observed RR upon specification of a list of hypothetical values for the bias parameters. The main limitation of this approach is related to the lack of explicit accounting for uncertainty about the bias parameters (Greenland 1998). To account for this uncertainty, probabilistic sensitivity analysis allows the user to specify a variety of probability densities for the bias parameters and use these densities to obtain simulation limits for the bias-adjusted exposure–disease relative risk. The accompanying Stata tool allows the user to specify a variety of probability density functions for the bias parameters (table 1). Probabilistic sensitivity analysis through Monte Carlo (random-number–based) simulations involves two iterated steps: 1) draw a random sample (one set of bias parameters) from the user-specified probability density functions of the bias parameters, and 2) back-calculate a bias-adjusted ("corrected) RR from the drawn parameters. These two steps are repeated several times to obtain a distribution of bias-adjusted RR.

Table 1. Probability distributions for the bias parameters used in adjustment for misclassification of the exposure, selection bias, and unmeasured or uncontrolled confounding

Type of systematic error and bias parameters	Description	Probability density functions (pdf)
Misclassification of the exposure		
`dseca()`	Sensitivity cases	`constant(`$\#$`)`
`dspca()`	Specificity cases	`uniform(`$a\ b$`)`
`dsenc()`	Sensitivity noncases	`triangular(`$a\ b\ c$`)`
`dspnc()`	Specificity noncases	`trapezoidal(`$a\ b\ c\ d$`)`
		`logit-logistic(`$m\ s\ \left[lb\ ub\right]$`)`
		`logit-normal(`$m\ s\ \left[lb\ ub\right]$`)`
Selection bias		
`dpscex()`	Pr selection cases exposed	`constant(`$\#$`)`
`dpscun()`	Pr selection cases unexposed	`uniform(`$a\ b$`)`
`dpsnex()`	Pr selection noncases exposed	`triangular(`$a\ b\ c$`)`
`dpsnun()`	Pr selection noncases unexposed	`trapezoidal(`$a\ b\ c\ d$`)`
		`logit-logistic(`$m\ s\ \left[lb\ ub\right]$`)`
		`logit-normal(`$m\ s\ \left[lb\ ub\right]$`)`
`dsbfactor()`	Selection bias factor	`constant(`$\#$`)`
		`log-logistic(`$m\ s$`)`
		`log-normal(`$m\ s$`)`
Unmeasured confounding		
`dpexp()`	Pr confounder exposed	`constant(`$\#$`)`
`dpunexp()`	Pr confounder unexposed	`uniform(`$a\ b$`)`
		`triangular(`$a\ b\ c$`)`
		`trapezoidal(`$a\ b\ c\ d$`)`
		`logit-logistic(`$m\ s\ \left[lb\ ub\right]$`)`
		`logit-normal(`$m\ s\ \left[lb\ ub\right]$`)`
`drrcd()`	RR counfounder–disease	`constant(`$\#$`)`
`dorce()`	OR confounder–exposure	`log-logistic(`$m\ s$`)`
		`log-normal(`$m\ s$`)`

The results of the simulation can be summarized by descriptions of the distribution of bias-adjusted RR. For example, the median (50th percentile) and the 2.5th and 97.5th percentiles can serve as analogues of the point and interval estimate for the bias-adjusted RR. To take into account uncertainty due to random error, we subtract from the distribution of the bias-adjusted ln(RR) a random draw from a normal distribution with zero mean and standard deviation equal to the standard error of the conventional ln(RR) estimate.

In a situation where more than one systematic error occurred during the study (uncontrolled confounding, selection bias, misclassification of the exposure) and these errors can be treated as independent, we can perform a multiple probabilistic bias analysis with adjustment made in the reverse order of their occurrence. Suppose that the order of events is as follows: confounded associations arise in the population used as the source of study subjects; subjects are selected; and finally, subjects are classified by exposure (with no misclassification of disease). Then, at each iteration of the simulation, adjustment of the observed exposure–disease RR follows this order: adjustment for misclassification of the exposure, then adjustment for selection bias, and finally, adjustment for uncontrolled confounders.

The rest of the article is organized as follows: section 3 presents the syntax of the command `episens` and its immediate form `episensi`; section 4 provides some examples in which the command is applied to published data; and section 5 contains a discussion of strengths and limitations of sensitivity analysis.

3 The episens command

3.1 Syntax

`episens` *var_case* *var_exposed* [*var_time*] [*if*] [*in*] [*weight*] [, *options*]

`episensi` *#a* *#b* *#c* *#d* [, *options*]

3.2 Description

`episens` performs deterministic and probabilistic sensitivity analysis of the exposure–disease relative risk for misclassification of the exposure, selection bias, and unmeasured or uncontrolled confounding.

`episensi` is the immediate form of `episens`.

3.3 Options

The probability distribution function (pdf) of each bias parameter is specified as an argument of an option. The list of probability distributions is presented in *pdf for the bias parameter (pdf_options)* below, as well as in table 1 organized by type of systematic error.

Misclassification of the exposure

dseca(*pdf_options*)	define the sensitivity among the cases
dspca(*pdf_options*)	define the specificity among the cases
dsenc(*pdf_options*)	define the sensitivity among the noncases
dspnc(*pdf_options*)	define the specificity among the noncases
corrsens(#)	set the correlation between case and noncase sensitivities to #
corrspec(#)	set the correlation between case and noncase specificities to #

Selection bias

dpscex(*pdf_options*)	define the selection probability among cases exposed
dpscun(*pdf_options*)	define the selection probability among cases unexposed
dpsnex(*pdf_options*)	define the selection probability among noncases exposed
dpsnun(*pdf_options*)	define the selection probability among noncases unexposed
dsbfactor(*pdf_options*)	define the selection-bias factor

Uncontrolled confounder

dpexp(*pdf_options*)	define the prevalence of the confounder among the exposed
dpunexp(*pdf_options*)	define the prevalence of the confounder among the unexposed
drrcd(*pdf_options*)	define the confounder-disease relative risk
dorce(*pdf_options*)	define the confounder-exposure odds ratio
corrprev(#)	set the correlation between exposure-specific confounder prevalences to #

(*Continued on next page*)

pdf for the bias parameter (*pdf_options*)

`constant(#)`	constant value equal to #
`uniform(`*a b*`)`	uniform between min = a and max = b
`triangular(`*a b c*`)`	triangular with min = a, mode = b, and max = c
`trapezoidal(`*a b c d*`)`	trapezoidal with min = a, modes between b and c, and max = d
`logit-logistic(`*m s* [*lb ub*]`)`	logit–logistic with mean = m and scale = s, shifted between [*lb ub*]
`logit-normal(`*m s* [*lb ub*]`)`	logit-normal with mean = m and scale = s, shifted between [*lb ub*]
`log-logistic(`*m s*`)`	loglogistic with mean = m and scale = s
`log-normal(`*m s*`)`	lognormal with mean = m and scale = s

Simulations

`reps(#)`	specify the number of replications to be performed
`nodots`	suppress the replication dots
`seed(#)`	set the random-number seed to #
`ndraw(#)`	number of observations drawn at each replication
`saving(`*filename*`)`	save results to *filename*
`grprior`	histogram of the priors
`grarrsys`	histogram of the adjusted relative risk (systematic error)
`grarrtot`	histogram of the adjusted relative risk (systematic error plus random error)

Study design, format, combined analysis

`study(cc|cs|ir)`	specify the type of study
`format(%`*fmt*`)`	set the display format for numbers
`combined`	specify combined analyses of multiple biases

3.4 Saved results

`episens` saves the following in `r()`:

Scalars

Deterministic sensitivity analysis

`r(bias_mie)` percentage of bias due to misclassification of the exposure

`r(rrdx_mie)` exposure–disease relative risk adjusted for misclassification of the exposure

`r(bias_sel)` percentage of bias due to selection bias

`r(rrdx_sel)` exposure–disease relative risk adjusted for selection bias

`r(bias_unc)` percentage of bias due to unmeasured confounding

`r(rrdx_unc)` exposure–disease relative risk adjusted for unmeasured confounding

Probabilistic sensitivity analysis

`r(rrdx_mie_pm)` median of the distribution of exposure–disease relative risks adjusted for misclassification of the exposure

`r(rrdx_mie_plb)` 2.5th percentile of the distribution of exposure–disease risks adjusted for misclassification of the exposure

`r(rrdx_mie_pub)` 97.5th percentile of the distribution of exposure–disease risks adjusted for misclassification of the exposure

`r(rrdx_sel_pm)` median of the distribution of exposure–disease relative risks adjusted for selection bias

`r(rrdx_sel_plb)` 2.5th percentile of the distribution of exposure–disease risks adjusted for selection bias

`r(rrdx_sel_pub)` 97.5th percentile of the distribution of exposure–disease risks adjusted for selection bias

`r(rrdx_unc_pm)` median of the distribution of exposure–disease relative risks adjusted for unmeasured confounding

`r(rrdx_unc_plb)` 2.5th percentile of the distribution of exposure–disease risks adjusted for unmeasured confounding

`r(rrdx_unc_pub)` 97.5th percentile of the distribution of exposure–disease risks adjusted for unmeasured confounding

`r(rrdx_all_pm)` median of the distribution of exposure–disease relative risks adjusted for all user-specified biases

`r(rrdx_all_plb)` 2.5th percentile of the distribution of exposure–disease risks adjusted for all user-specified biases

`r(rrdx_all_pub)` 97.5th percentile of the distribution of exposure–disease risks adjusted for all user-specified biases

4 Examples

4.1 Deterministic sensitivity analysis

To illustrate how to perform a sensitivity analysis using the command `episens`, we used the crude data from a case–control study comparing cases of lung-cancer deaths with controls based on occupational exposure to resins (Greenland et al. 1994).

```
. cci 45 94 257 945, woolf
                                                         Proportion
                 |   Exposed   Unexposed  |      Total     Exposed
-----------------+------------------------+------------------------
           Cases |        45          94  |        139       0.3237
        Controls |       257         945  |       1202       0.2138
-----------------+------------------------+------------------------
           Total |       302        1039  |       1341       0.2252
                 |                        |
                 |      Point estimate    |    [95% Conf. Interval]
                 |------------------------+------------------------
      Odds ratio |         1.760286       |    1.202457    2.576898 (Woolf)
 Attr. frac. ex. |         .4319106       |    .1683693    .6119365 (Woolf)
 Attr. frac. pop |         .1398272       |
                 +-------------------------------------------------
                               chi2(1) =     8.63  Pr>chi2 = 0.0033
```

The authors found a positive association between occupational exposure and lung-cancer deaths (OR=1.76, 95% CI, 1.20–2.58). Further adjustment for age or year did not substantially change this association. Nonetheless, the measured exposure to resins must be misclassified to some extent.

Exposure misclassification

The sensitivities and specificities of classification among the cases and noncases would allow us to adjust the observed data for classification error (Greenland 1996; Greenland and Lash 2008). We can perform a deterministic sensitivity analysis assuming nondifferential misclassification of the exposure and assigning a specific (fixed) value to the sensitivity and specificity among cases and noncases, say, 0.9.

```
. episensi 45 94 257 945, st(cc) dseca(c(.9)) dspca(c(.9)) dsenc(c(.9))
> dspnc(c(.9))
Se|Cases   : Constant(.9)
Sp|Cases   : Constant(.9)
Se|No-Cases: Constant(.9)
Sp|No-Cases: Constant(.9)
Observed Odds Ratio [95% Conf. Interval]= 1.76 [1.20, 2.58]
Deterministic sensitivity analysis for misclassification of the exposure
   External adjusted Odds Ratio = 2.34
   Percent bias = -25%
```

The odds ratio (OR adjusted for misclassification of the exposure is 2.34, with a percentage of bias of $(1.76 - 2.34)/2.34 * 100 = -25\%$. However, under the assumption that the sensitivity among the cases (0.9) is higher than the sensitivity among the noncases (0.8) with specificities at 0.8, the OR adjusted for misclassification of the exposure would be 9.11.

```
. episensi 45 94 257 945, st(cc) dseca(c(.9)) dspca(c(.8)) dsenc(c(.8))
> dspnc(c(.8))
Se|Cases   : Constant(.9)
Sp|Cases   : Constant(.8)
Se|No-Cases: Constant(.8)
Sp|No-Cases: Constant(.8)
Observed Odds Ratio [95% Conf. Interval]= 1.76 [1.20, 2.58]
Deterministic sensitivity analysis for misclassification of the exposure
   External adjusted Odds Ratio = 9.11
   Percent bias = -81%
```

One can repeat this procedure for various likely combinations of sensitivities and specificities among the cases and noncases and present the adjusted ORs in a table (Greenland 1996; Greenland and Lash 2008).

Selection bias

Because of lack of adequate job records for exposure reconstruction, some data available from this study indicate that the probabilities of selecting a case and a noncase are 0.7 and 0.6, respectively.

If selection is associated with both exposure to resins and lung-cancer death, considerable selection bias could result. The selection-bias factor `dsbfactor` is given by the exposed versus unexposed selection probabilities comparing cases (`dpscex`/`dpscun`) and noncases (`dpsnex`/`dpsnun`). If the selection probabilities among cases and noncases do not differ across exposure status, there is no bias [`dsbfactor` = (`dpscex`/`dpscun`)/(`dpsnex`/`dpsnun`) = 1].

```
. episensi 45 94 257 945, st(cc) dpscex(c(.7)) dpscun(c(.7)) dpsnex(c(.6))
> dpsnun(c(.6))
Pr Case Selection Exposed: Constant(.7)
Pr Case Selection No-Exposed: Constant(.7)
Pr No-Case Selection Exposed: Constant(.6)
Pr No-Case Selection No-Exposed: Constant(.6)
Observed Odds Ratio [95% Conf. Interval]= 1.76 [1.20, 2.58]
Deterministic sensitivity analysis for selection bias
   External adjusted Odds Ratio = 1.76
   Percent bias =   0%
```

However, if the probabilities of selecting a case and a noncase are different with respect to the exposure status, the selection-bias factor will be greater than 1 if (`dpsecx`/`dpscun`) > (`dpsnex`/`dpsnun`) and lower than 1 if (`dpsecx`/`dpscun`) < (`dpsnex`/`dpsnun`). For instance, lets suppose that the probability of selecting a case exposed is 0.9, a case unexposed is 0.5, a noncase exposed is 0.5, and a noncase unexposed is 0.7. The selection-bias factor is equal to $(.9/.5)/(.5/.7) = 2.5$.

```
. episensi 45 94 257 945, st(cc) dpscex(c(.9)) dpscun(c(.5)) dpsnex(c(.5))
> dpsnun(c(.7))
Pr Case Selection Exposed: Constant(.9)
Pr Case Selection No-Exposed: Constant(.5)
Pr No-Case Selection Exposed: Constant(.5)
Pr No-Case Selection No-Exposed: Constant(.7)
Observed Odds Ratio [95% Conf. Interval]= 1.76 [1.20, 2.58]
Deterministic sensitivity analysis for selection bias
   External adjusted Odds Ratio = 0.70
   Percent bias = 152%
```

The selection-bias adjusted OR is equal to $1.76/2.5 = 0.7$. In an opposite scenario, the probability of selecting a case exposed is 0.5, a case unexposed is 0.9, a noncase exposed is 0.7, and a noncase unexposed is 0.5. The selection-bias factor is equal to $(.5/.9)/(.7/.5) = 0.4$.

```
. episensi  45 94 257 945, st(cc) dpscex(c(.5)) dpscun(c(.9)) dpsnex(c(.7))
> dpsnun(c(.5))
Pr Case Selection Exposed: Constant(.5)
Pr Case Selection No-Exposed: Constant(.9)
Pr No-Case Selection Exposed: Constant(.7)
Pr No-Case Selection No-Exposed: Constant(.5)
Observed Odds Ratio [95% Conf. Interval]= 1.76 [1.20, 2.58]
Deterministic sensitivity analysis for selection bias
   External adjusted Odds Ratio = 4.44
   Percent bias = -60%
```

The selection-bias adjusted OR is equal to $1.76/0.4 = 4.4$. These two extreme scenarios, however, do not take into account that there is no reason to expect big differences comparing the case and noncase selection probabilities with respect to the exposure; that is, `dpscex` should be similar to `dpscun`, and `dpsnex` should be similar to `dpsnun`.

Uncontrolled confounders

In the case–control study of occupational exposure to resins and lung-cancer mortality, the authors had no data on smoking. Therefore, we want to quantify the potential bias introduced by ignoring smoking in the published analysis. To back-calculate the smoking adjusted OR, we assume that the RR relating smoking to lung-cancer death is 5, and the smoking prevalences among the resins exposed and unexposed are 0.7 and 0.5, respectively.

```
. episensi 45 94 257 945, dpexp(c(.7)) dpunexp(c(.5)) drrcd(c(5))
Pr(c=1|e=1): Constant(.7)
Pr(c=1|e=0): Constant(.5)
RR_cd      : Constant(5)
Observed Odds Ratio [95% Conf. Interval]= 1.76 [1.20, 2.58]
Deterministic sensitivity analysis for unmeasured confounding
   External adjusted Odds Ratio = 1.39
   Percent bias =  27%
```

The resin lung-cancer death OR adjusted for smoking is 1.39, which is lower than the observed OR because we assumed positive associations between the confounder and the outcome ($5 > 1$) as well as between the confounder and the exposure ($0.7 > 0.5$). For sensitivity analysis, one can repeat the above command using other plausible values for the resins-specific smoking prevalences and the smoking lung-cancer OR (Greenland 1996; Greenland and Lash 2008).

4.2 Probabilistic sensitivity analysis

The main limitation of deterministic sensitivity analyses is that they treat the bias parameters as if they were known or as if they can assume only certain fixed values. It also fails to discriminate among the different scenarios in terms of their likelihood, and it is not straightforward to summarize all the bias-adjusted RR calculated under a variety of possible values for the bias parameters. Therefore, we next assume that we can specify prior probability distributions for the bias parameters that capture our uncertainty about those parameters and then use these distributions in a probabilistic sensitivity analysis.

Exposure misclassification

We first assume nondifferential misclassification of the exposure with probability density functions for sensitivities and specificities among cases and noncases equal to trapezoidal distributions with a minimum of 0.75 and a maximum of 1, and an interval of equally probable values between 0.85 and 0.95.

A technical issue is that the formulas used to back-calculate the relative risk can yield negative adjusted counts, which are impossible. To avoid negative adjusted counts, the prior distributions for sensitivity and specificity must be bounded by `dsenc()` $\geq$ (number of noncases classified exposed / total number of noncases) and `dspnc()` $\geq$ (number of noncases classified unexposed / total number of noncases) among noncases and by `dseca()` $\geq$ (number of cases classified exposed / total number of cases) and `dspca()` $\geq$ (number of cases classified unexposed / total number cases) among cases. The command `episens` automatically discards draws of sensitivities and specificities from user-specified distributions falling into the region of negative adjustment. It is the user's decision whether to check that the resulting truncated distribution still appears to be reasonable.

Here negative adjustments would occur whenever `dsenc()` $<$ ($257/1202 = 0.214$) and `dspnc()` $<$ ($945/1202 = 0.786$) among noncases, and `dseca()` $<$ ($45/139 = 0.324$) and `dspca()` $<$ ($94/139 = 0.676$) among cases. Among these four bounds, however, only one is of interest (`dspnc()` < 0.786) because we specified trapezoidal distributions between 0.75 and 1.

```
. episensi 45 94 257 945, st(cc) reps(20000) nodots
> dseca(trap(.75 .85 .95 1)) dspca(trap(.75 .85 .95 1))
> dsenc(trap(.75 .85 .95 1)) dspnc(trap(.75 .85 .95 1)) seed(123)
Se|Cases   : Trapezoidal(.75,.85,.95,1)
Sp|Cases   : Trapezoidal(.75,.85,.95,1)
Se|No-Cases: Trapezoidal(.75,.85,.95,1)
Sp|No-Cases: Trapezoidal(.75,.85,.95,1)
Probabilistic sensitivity analysis for misclassification of the exposure
                                      Percentiles          Ratio
                             2.5      50        97.5       97.5/2.5
                             ----------------------------------------
Conventional                 1.20     1.76      2.58       2.14
Systematic error             1.87     2.47      14.71      7.86
Systematic and random error  1.49     2.57      15.00      10.07
```

The 2.5th and 97.5th percentiles of the simulated distribution of bias-adjusted OR are 1.9 and 14.7, and the median estimate is 2.5. Including random error in the distribution of bias-adjusted OR, the 2.5th and 97.5th percentiles of the simulated distribution become 1.5 and 15. Unsurprisingly, given the high uncertainty about the bias parameters, the ratio of the bias-adjusted simulation limits (15/1.5) is about 5 times the ratio of the conventional limits (2.6/1.2). The option `grprior` helps to visualize the assumed prior probability distributions by showing histograms of the draws of the bias parameters from those distributions (figure 1). The specificity distribution among noncases is truncated at 0.786 because the command `episens` discards draws leading to negative adjustments. The option `saving(`*filename*`)` can be useful to inspect the sampled distributions of the bias parameters and the bias-adjusted odds ratios and to control various aspects of the graphs.

We can allow for differential misclassification by drawing the sensitivities and specificities from different trapezoidal distributions for cases and controls. Because the sensitivities/specificities among the cases are not independent of the sensitivities/specificities among the noncases, we should specify a high correlation between sensitivities and specificities respectively, say, 0.8. The options `corrsens()` and `corrspec()` can help to control the degree of differentiality. Assuming the same priors for cases and noncases, a correlation of 1 means no difference between sensitivities/specificities among cases and noncases (nondifferential misclassification).

```
. episensi 45 94 257 945, st(cc) reps(20000) nodots dseca(trap(.75 .85 .95 1))
> dspca(trap(.75 .85 .95 1)) dsenc(trap(.7 .8 .9 .95))
> dspnc(trap(.7 .8 .9 .95)) corrsens(.8) corrspec(.8) seed(123) grprior
Se|Cases   : Trapezoidal(.75,.85,.95,1)
Sp|Cases   : Trapezoidal(.75,.85,.95,1)
Se|No-Cases: Trapezoidal(.7,.8,.9,.95)
Sp|No-Cases: Trapezoidal(.7,.8,.9,.95)
Corr Se|Cases and Se|No-Cases : .8
Corr Sp|Cases and Sp|No-Cases : .8
Probabilistic sensitivity analysis for misclassification of the exposure
                                         Percentiles          Ratio
                             2.5       50         97.5      97.5/2.5
                             ----------------------------------------
Conventional                 1.20      1.76       2.58      2.14
Systematic error             1.81      3.48       48.19     26.57
Systematic and random error  1.61      3.60       48.92     30.47
```

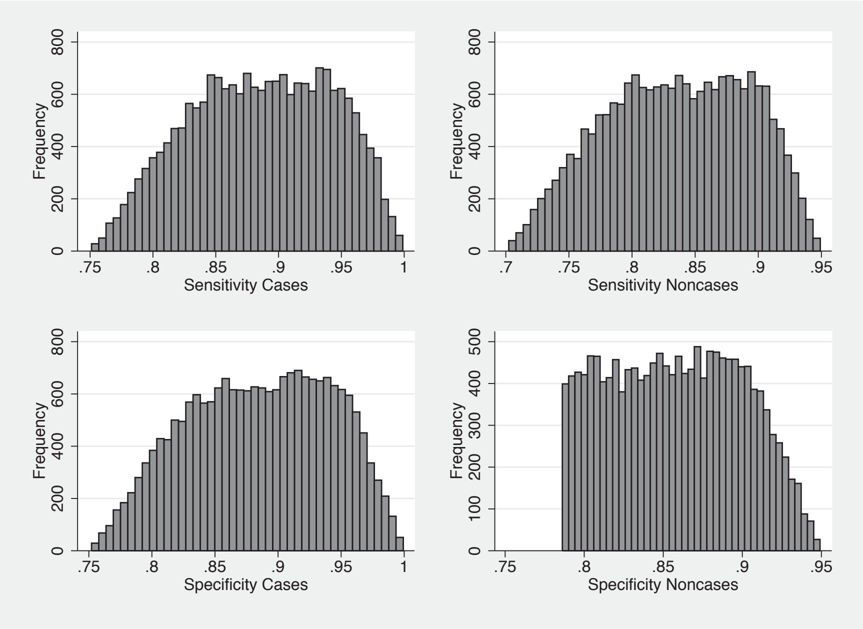


Figure 1. Histograms of 20,000 draws from trapezoidal prior distributions ($a = 0.75$, $b = 0.85$, $c = 0.95$, $d = 1$) for the sensitivity and specificity among cases and noncases.

The 95% simulation limits including systematic and random error were 1.6 and 49, with a median estimate of 3.6.

Selection bias

Although *Selection bias* of section 4.1 shows how sensitive the resins lung-cancer death OR is to different scenarios of selection bias, these scenarios are of no help because only a small association (if any) between lack of records and lung-cancer death is expected (`dsbfactor()` = 1). Instead of assigning a distribution to each selection probability (`dpscex()`, `dpscun()`, `dpsnex()`, `dpsnun()`), we can directly assign a prior distribution to the selection-bias factor (figure 2). Particularly, we assume a lognormal distribution with mean 0 and standard deviation 0.21, which yields 95% prior probability of the bias factor falling between $\exp(0 - 1.96 * 0.21) = 0.7$ and $\exp(0 + 1.96 * 0.21) = 1.5$.

```
. episensi 45 94 257 945, st(cc) reps(20000) nodots dsbfactor(log-n(0 0.21))
> seed(123) grprior

selection bias factor: Log-Normal(0.00,0.21)

Probabilistic sensitivity analysis for selection bias

                                           Percentiles         Ratio
                             2.5       50         97.5      97.5/2.5
                             ---------------------------------------
Conventional                 1.20       1.76       2.58       2.14
Systematic error             1.16       1.76       2.66       2.29
Systematic and random error  1.01       1.75       3.08       3.05
```

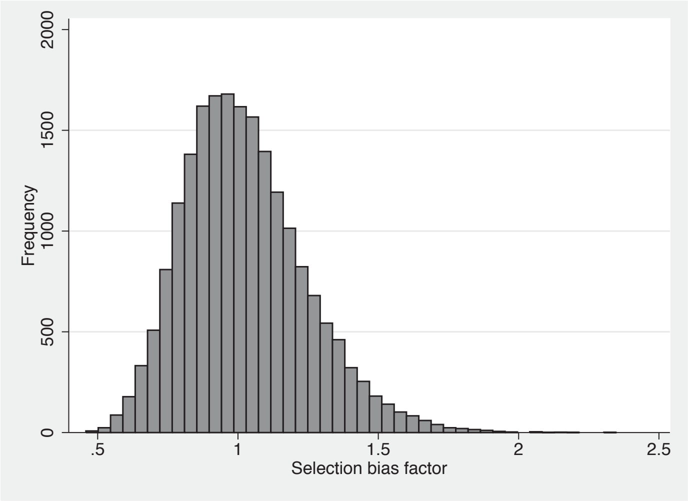


Figure 2. Histogram of 20,000 draws from a lognormal distribution for the selection-bias factor ($m = 0, s = 0.21$).

As expected, the median estimate of the selection-bias adjusted OR 1.75 is not practically different from the conventional 1.76, but the ratio of 95% simulation limits including systematic and random error (3.05) is 43% higher than the conventional one (2.14).

Uncontrolled confounder

As a starting example, we specify two uniform independent distributions for the smoking prevalences among exposed and unexposed between 0.4 and 0.7. We also independently specify a prior probability distribution for the smoking lung-cancer mortality RR that is lognormal with 95% confidence limits of ln(5) and ln(15). These limits imply that the mean of this prior RR distribution is $\{\ln(15)+\ln(5)\}/2 = 2.159$ with standard deviation $\{\ln(15)-\ln(5)\}/(2*1.96) = 0.280$. Figure 3 shows the draws from these prior probability distributions for the bias parameters (option `grprior`).

```
. episensi 45 94 257 945, st(cc) reps(20000) nodots dpexp(uni(.4 .7))
> dpunexp(uni(.4 .7)) drrcd(log-n(2.159 .280)) seed(123)
> grarrtot grprior
Pr(c=1|e=1): Uniform(.4,.7)
Pr(c=1|e=0): Uniform(.4,.7)
RR_cd      : Log-Normal(2.16,0.28)
Probabilistic sensitivity analysis for unmeasured confounding
                                       Percentiles          Ratio
                             2.5       50         97.5      97.5/2.5
                             ---------------------------------------
Conventional                 1.20      1.76       2.58      2.14
Systematic error             1.25      1.76       2.49      2.00
Systematic and random error  1.05      1.76       2.96      2.83
```

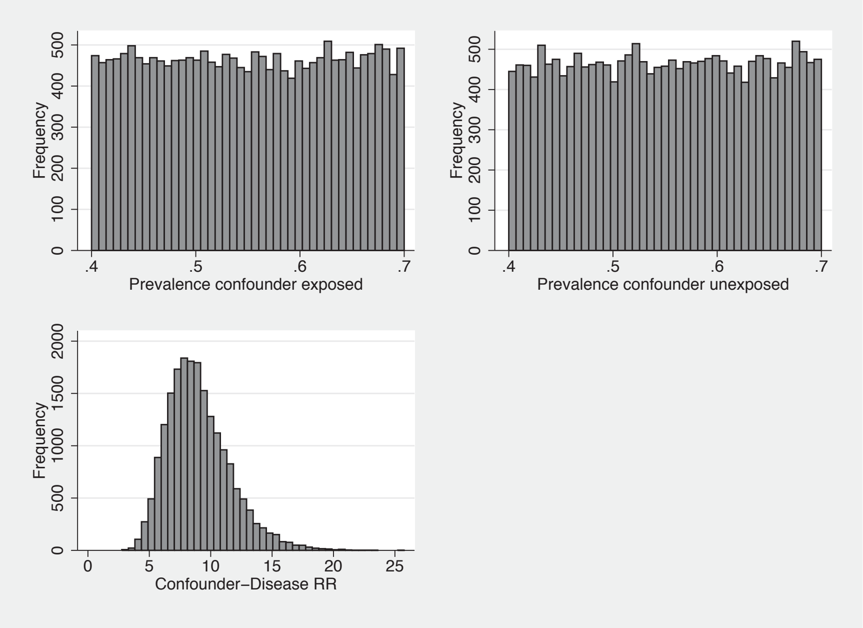


Figure 3. Histograms of 20,000 draws each from prior distributions for the smoking-exposure specific prevalences and the confounder-disease RR.

From the 20,000 draws for each bias parameter, the median smoking-adjusted resins lung-cancer OR is 1.76 with 2.5th and 97.5th percentiles of 1.05 and 2.96. As expected, the ratio of the smoking-adjusted simulation limits (2.83) is 32% higher than the ratio of the conventional limits (2.14). The distribution of the bias-adjusted OR, including both systematic and random error is shown in figure 4 (option `grarrtot`).

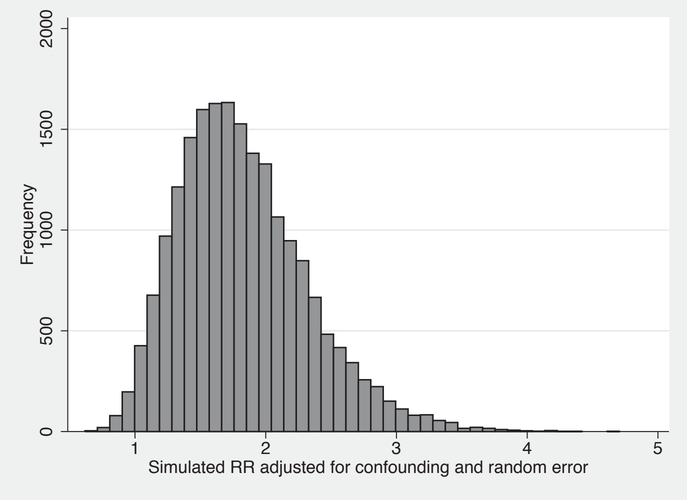


Figure 4. Distribution of the 20,000 smoking-adjusted resins lung-cancer odds ratios derived from the data and the prior distributions in figure 3.

Given that there is no reason to expect great differences in the prevalence of smoking among resins exposed and unexposed, small differences are more likely than large ones. Therefore, it is unrealistic to assume two independent priors for the two prevalences of smoking `dpexp()` and `dpunexp()`. A way to incorporate this consideration in the probabilistic sensitivity analysis is to specify a probability distribution for the confounder-exposure OR (option `dorce()`) instead of the prevalence of the confounder among the exposed (option `dpexp()`). Using independent priors for the confounder-exposure OR and the prevalence of the confounder among the unexposed is more reasonable and easier to specify realistically than using independent priors for the confounder prevalences among the exposed and unexposed.

Suppose that we assign to the confounder-exposure OR a lognormal distribution with mean 0 (that is, `dpexp()` is expected to be similar to `dpunexp()`) and 95% prior limits equal to $\{(1-.7)*.4\}/\{0.7*(1-0.4)\} = 0.286$ and $\{.7*(1-.4)\}/\{(1-.7)*0.4\} = 3.5$. These limits are derived calculating a confounder-exposure OR at the extreme values of 0.4 and 0.7 for `dpexp()` and `dpunexp()`. The standard deviation for the lognormal distribution is equal to the standard error calculated from the prior limits $\{\ln(3.5)-\ln(0.286)\}/(1.96*2) = 0.639$.

```
. episensi 45 94 257 945, st(cc) reps(20000) nodots dpunexp(uni(.4 .7))
> dorce(log-n(0 0.639)) drrcd(log-n(2.159 .280)) seed(123) grprior
Pr(c=1|e=0): Uniform(.4,.7)
RR_cd      : Log-Normal(2.16,0.28)
OR_ce      : Log-Normal(0.00,0.64)
Probabilistic sensitivity analysis for unmeasured confounding
                                      Percentiles          Ratio
                             2.5      50        97.5      97.5/2.5
                             ----------------------------------------
Conventional                 1.20     1.76      2.58      2.14
Systematic error             1.25     1.76      3.02      2.42
Systematic and random error  1.04     1.79      3.36      3.23
```

Given the priors graphically presented in figure 5, the median bias-adjusted OR is equal to 1.79 with 95% simulation limits 1.04 and 3.36, which have a ratio 3.2 or 14% higher than the earlier ratio of 2.8 based on unrealistic independent priors of the smoking prevalences.

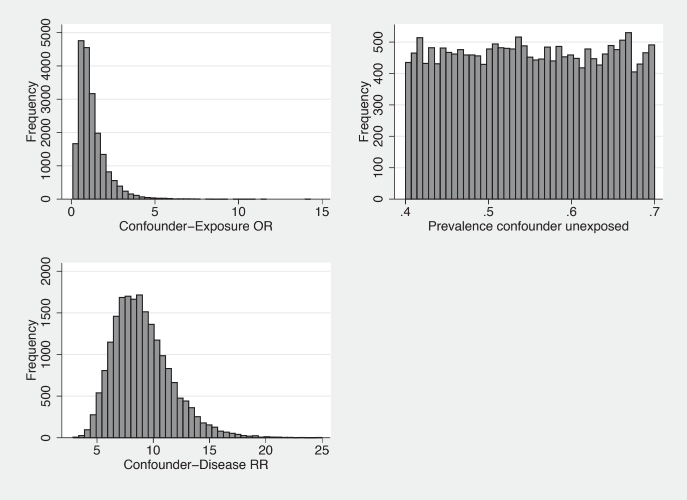


Figure 5. Histograms of 20,000 draws each from prior distributions for the confounder-exposure odds ratio, the prevalence of smoking among the unexposed to resins, and the confounder-disease RR.

4.3 Combined analysis of biases

Adjustment for multiple biases can be done by specifying the option `combined`. To illustrate, we will adjust the observed OR for differential misclassification of the resins exposure and the selection bias and for uncontrolled confounding by smoking using the probability density functions for the bias parameters specified in the above sections.

```
. episensi 45 94 257 945, st(cc) reps(20000) nodots dseca(trap(.75 .85 .95 1))
> dspca(trap(.75 .85 .95 1)) dsenc(trap(.7 .8 .9 .95)) dspnc(trap(.7 .8 .9 .95))
> corrsens(.8) corrspec(.8) dsbfactor(log-n(0 0.21)) dpunexp(uni(.4 .7))
> dorce(log-n(0 0.639)) drrcd(log-n(2.159 .280)) seed(123) combined

Se|Cases   : Trapezoidal(.75,.85,.95,1)
Sp|Cases   : Trapezoidal(.75,.85,.95,1)
Se|No-Cases: Trapezoidal(.7,.8,.9,.95)
Sp|No-Cases: Trapezoidal(.7,.8,.9,.95)
Corr Se|Cases and Se|No-Cases : .8
Corr Sp|Cases and Sp|No-Cases : .8
selection bias factor: Log-Normal(0.00,0.21)
Pr(c=1|e=0): Uniform(.4,.7)
RR_cd      : Log-Normal(2.16,0.28)
OR_ce      : Log-Normal(0.00,0.64)

Probabilistic sensitivity analysis - Combined corrections
    Misclassification of the exposure
    Selection bias
    Unmeasured confounding

                                       Percentiles         Ratio
                             2.5       50        97.5      97.5/2.5
                             ----------------------------------------
Conventional                 1.20      1.76      2.58      2.14
Systematic error             1.47      3.87      56.17     38.14
Systematic and random error  1.34      3.92      57.56     42.98
```

A comparison of the combined analysis with the single-bias analyses of the previous sections shows that, under the given priors confounding by smoking and selection bias have little impact on the observed resins lung-cancer OR, and that the greatest source of uncertainty is misclassification of the exposure.

5 Discussion

We have presented a new Stata command, `episens`, to perform both deterministic and probabilistic sensitivity analysis to assess the potential impact of systematic errors on observed exposure–disease associations. To illustrate, we applied `episens` to a case–control study regarding occupational resin exposure and lung-cancer deaths.

The advantages of a probabilistic sensitivity analysis have been discussed previously (Greenland 2001; Greenland 2003a; Greenland 2005; Greenland and Lash 2008; Lash and Fink 2003; Phillips 2003; Phillips and LaPole 2003; and Steenland and Greenland 2004). Briefly, a probabilistic sensitivity analysis requires the investigator to make explicit this uncertainty about bias parameters. This explication is done by using prior distributions for the parameters, which reflect background information and judgments of the investigator about sources of systematic error. The resulting distribution of bias-adjusted estimates captures the uncertainty about bias that is ignored by conventional statistics (such as confidence intervals). Under certain common conditions, this distribution can be viewed as an approximation to the more computationally demanding posterior distribution of Bayesian analysis (Greenland 2005).

Concerns have been raised about the arbitrariness in the particular distributions assumed for the bias parameters. The important point however is that changing the prior distributions can result in different distributions for the bias-adjusted exposure–disease RR. This relation corresponds to the fact that if different investigators have different opinions about sources of bias, it should be no surprise if their conclusions differ.

Differences of opinion about bias sources may be represented by different prior distributions. The different bias-adjusted RR distributions that result then reflect the differences in conclusions (final opinions) we should expect when prior opinions differ and decisive data are lacking (as is usually the case in epidemiology). Thus, by varying the input prior distributions for probabilistic sensitivity analyses, we can illustrate the extent to which differences in prior opinions about various sources of bias may contribute to conflicting interpretations of the study. The possibility of conflicting outputs may encourage analysts to provide the best available evidence or arguments to support their own choices for prior distributions. As with earlier, more specialized SAS macros (Fox, Lash, and Greenland 2005), the Stata command presented in this paper greatly eases such variation by automating the transformation of the input priors to the output bias-adjusted distributions.

In conclusion, we have provided a user-friendly command suitable for both deterministic and probabilistic sensitivity analysis to evaluate bias due to misclassification of a binary exposure variable, selection bias, and bias due to an uncontrolled confounder. We hope that future refinements will provide extensions to variables with multiple levels, and allow for misclassification of multiple variables.

6 References

Chu, H., Z. Wang, S. R. Cole, and S. Greenland. 2006. Sensitivity analysis of misclassification: A graphical and a Bayesian approach. *Annals of Epidemiology* 16: 834–841.

Eddy, D. M., V. Hasselblad, and R. D. Shachter. 1992. *Meta-Analysis by the Confidence Profile Method: The Statistical Synthesis of Evidence*. Boston: Academic Press.

Fox, M. P., T. L. Lash, and S. Greenland. 2005. A method to automate probabilistic sensitivity analyses of misclassified binary variables. *International Journal of Epidemiology* 34: 1370–1376.

Greenland, S. 1996. Basic methods for sensitivity analysis of biases. *International Journal of Epidemiology* 25: 1107–1116.

———. 1998. The sensitivity of a sensitivity analysis (invited paper). In *1997 Proceedings of the Biometrics Section*, 19–21. Alexandria, VA: American Statistical Association.

———. 2001. Sensitivity analysis, Monte Carlo risk analysis, and Bayesian uncertainty assessment. *Risk Analysis* 21: 579–583.

———. 2003a. Generalized conjugate priors for Bayesian analysis of risk and survival regressions. *Biometrics* 59: 92–99.

———. 2003b. The impact of prior distributions for uncontrolled confounding and response bias: A case study of the relation of wire codes and magnetic fields to childhood leukemia. *Journal of the American Statistical Association* 98: 47–54.

———. 2005. Multiple-bias modeling for analysis of observational data (with discussion). *Journal of the Royal Statistical Society, Series A* 168: 267–306.

Greenland, S., and T. L. Lash. 2008. *Bias analysis.* In *Modern Epidemiology*, ed. K. J. Rothman, S. Greenland, and T. L. Lash, 3rd ed., in press. Philadelphia: Lippincott–Raven.

Greenland, S., A. Salvan, D. H. Wegman, M. F. Hallock, and T. J. Smith. 1994. A case–control study of cancer mortality at a transformer-assembly facility. *Interntional Archives of Occupational and Environmental Health* 66: 49–54.

Hoffman, F. O., and J. S. Hammonds. 1994. Propagation of uncertainty in risk assessments: The need to distinguish between uncertainty due to lack of knowledge and uncertainty due to variability. *Risk Analysis* 14: 707–712.

Jurek, A. M., G. Maldonado, S. Greenland, and T. R. Church. 2006. Exposure-measurement error is frequently ignored when interpreting epidemiologic study results. *European Journal of Epidemiology* 21: 871–876.

Lash, T. L., and A. K. Fink. 2003. Semiautomated sensitivity analysis to assess systematic errors in observational data. *Epidemiology* 14: 451–458.

Phillips, C. V. 2003. Quantifying and reporting uncertainty from systematic errors. *Epidemiology* 14: 459–466.

Phillips, C. V., and L. M. LaPole. 2003. Quantifying errors without random sampling. *BMC Medical Research Methodology* 3: 9.

Steenland, K., and S. Greenland. 2004. Monte Carlo sensitivity analysis and Bayesian analysis of smoking as an unmeasured confounder in a study of silica and lung cancer. *American Journal of Epidemiology* 160: 384–392.

About the authors

Nicola Orsini is a Ph.D. student, Division of Nutritional Epidemiology, the National Institute of Environmental Medicine, Karolinska Institutet, Stockholm, Sweden.

Rino Bellocco is associate professor of biostatistics, Department of Medical Epidemiology and Biostatistics, Karolinska Institutet, Stockholm, Sweden, and Associate Professor of Biostatistics, Department of Statistics, University of Milano Bicocca, Milano, Italy.

Matteo Bottai is assistant professor of biostatistics, Department of Epidemiology and Biostatistics, Arnold School of Public Health, University of South Carolina.

Alicja Wolk is professor of nutritional epidemiology, the National Institute of Environmental Medicine, Karolinska Institutet, Stockholm, Sweden.

Sander Greenland is professor of epidemiology, UCLA School of Public Health, and professor of statistics, UCLA College of Letters and Science, Los Angeles, CA.

The Stata Journal (2008)
8, Number 1, pp. 49–67

A new framework for managing and analyzing multiply imputed data in Stata

John B. Carlin
Clinical Epidemiology & Biostatistics Unit
Murdoch Children's Research Institute &
University of Melbourne
Parkville, Australia
john.carlin@mcri.edu.au

John C. Galati
Clinical Epidemiology & Biostatistics Unit
Murdoch Children's Research Institute &
University of Melbourne
Parkville, Australia

Patrick Royston
Cancer and Statistical Methodology Groups
MRC Clinical Trials Unit
London, UK

Abstract. A new set of tools is described for performing analyses of an ensemble of datasets that includes multiple copies of the original data with imputations of missing values, as required for the method of multiple imputation. The tools replace those originally developed by the authors. They are based on a simple data management paradigm in which the imputed datasets are all stored along with the original data in a single dataset with a vertically stacked format, as proposed by Royston in his `ice` and `micombine` commands. Stacking into a single dataset simplifies the management of the imputed datasets compared with storing them individually. Analysis and manipulation of the stacked datasets is performed with a new prefix command, `mim`, which can accommodate data imputed by any method as long as a few simple rules are followed in creating the imputed data. `mim` can validly fit most of the regression models available in Stata to multiply imputed datasets, giving parameter estimates and confidence intervals computed according to Rubin's results for multiple imputation inference. Particular attention is paid to limiting the available postestimation commands to those that are known to be valid within the multiple imputation context. However, the user has flexibility to override these defaults. Features of these new tools are illustrated using two previously published examples.

Keywords: st0139, mim, mimstack, ice, micombine, miset, mifit, multiple imputation, missing data, missing at random

1 Introduction

The presence of missing data raises challenges for many statistical analyses, especially those based on multivariable methods where the absence of values on just one or two variables for a case will, in general, render that observation unusable in standard methods of analysis. Loss of observations from the analysis dataset in this way raises two potential threats: first, that of bias due to selection processes that may be related to the variables or—more importantly—to the associations of interest, and second, that of loss of precision (or power) due to reduction in the available sample size. Over the past two decades, considerable literature has arisen on statistical approaches to handling missing data: in particular, see the influential texts by Little and Rubin (2002) and Schafer (1997).

The leading general approach to the problem now appears to be the method of multiple imputation (MI). Briefly, this method has two distinct stages. First, a set of copies of the original dataset must be created, in which each of the missing values is imputed using an appropriate modeling procedure. Second, standard analyses are performed on each of these completed or imputed datasets, and the results (in the form of whatever parameter estimates are of substantive interest—typically regression coefficients) are then combined according to Little and Rubin's theory (2002) to obtain a set of final estimates and standard errors. This process has been outlined in more detail in an earlier article (Carlin et al. 2003).

Although at first glance MI may appear cumbersome for use in everyday data analysis, with appropriate software tools the method is not difficult to apply. Recent publication of software both for performing imputation and for analyzing the imputed datasets has led to an upsurge of usage in applied research papers. The boundaries for safe application of the method have not been fully delineated although it is well understood that the standardly available approaches all rely on an assumption that the missing data can be regarded as missing at random. This assumption is a tricky one to characterize clearly in many applications, especially those involving large datasets with many variables (Potthoff et al. 2006). Furthermore, there are a number of possible approaches for performing imputation (Schafer 1997; van Buuren, Boshuizen, and Knook 1999). Although we think the method is very effective and reliable in many situations, the underlying assumptions need to be considered carefully in any application, and further research is needed to better define the types of problems where reliable answers can be expected.

In order to facilitate better research on MI and its validity in the context of departures from assumptions, as well as to facilitate more widespread adoption of the method in practice, we have developed a comprehensive new architecture for managing the process of data analysis using MI in Stata. Within this new framework, users are able to apply all the commonly used estimation commands available in Stata, including those based on the `svy` prefix, providing a substantial extension of the previously available tools for MI analysis. We have also refined and extended the available postestimation commands, including an implementation of `predict` for multiply imputed data. These advances are provided in the form of a new prefix command, `mim`, which this article introduces.

2 Overview

In this section, we give an outline of our approach and relate the new structure to previous work by the authors.

2.1 Background

Earlier publications described a system for managing imputed datasets and performing combined analyses in Stata (Carlin et al. 2003) and described a command for creating imputations using the method of "chained equations" (`ice`) along with another command (`micombine`) for combined analyses (Royston 2004, 2005a, 2005b). The latter publications were a substantial advance on the former for two reasons: (1) they provided a method for performing the imputations, and (2) they highlighted the fact that the MI process could be handled in Stata by storing imputed versions of a dataset in a stacked format within a single dataset.

The earlier `mitools` package of commands (Carlin et al. 2003) had no facility for generating imputed values (which had to be generated externally, for example, by using the freeware NORM from http://www.stat.psu.edu/~jls/misoftwa.html) and assumed that the imputed datasets were to remain distinct files. The `mifit` command in that package performed combined analysis for a range of regression commands by repeatedly loading each imputed dataset, storing the results obtained, and performing the combined calculations at the end. The package also introduced methods for postestimation in the MI framework (commands `milincom` and `mitestparm`) and was a first attempt to create a general environment for flexible management of imputed datasets. However, the architecture adopted meant that a special-purpose command needed to be written to perform manipulations (recoding, transformation, etc.) within each imputed dataset because this required successive reloading of the datasets.

Royston's focus was on developing the `ice` command as an implementation of the method of "multiple imputation using chained equations", or "MICE" (Van Buuren, Boshuizen, and Knook 1999), with the `micombine` command provided to allow inferences to be obtained by combining analyses over the resulting imputations. Again, a wide range of regression estimation commands was accommodated. Stata's `ereturn` commands were used to allow the standard Stata postestimation commands `test` and `testparm` to work as might be expected, using estimated regression coefficients and the variance–covariance matrix obtained by pooling across the imputed datasets.

It seems clear that the best environment for managing the method of MI in Stata is based on storing the imputed datasets in stacked form in a single dataset, as in `ice` and `micombine`. The `mim` prefix command described in this article provides a new integrated framework for MI in Stata using this paradigm. To be compatible with `mim`, a dataset must contain two variables `_mj` and `_mi` that index, respectively, the individual datasets within the stack and observations within the datasets. Thus `_mi` should contain the same value i for each observation from the ith individual across datasets, with the datasets being identified by `_mj` taking the values $0, 1, \ldots, m$ for the original data (`_mj` $= 0$) and each of the imputed datasets. Most mim subcommands use only the imputed data (so ignore cases for which _mj = 0), but retaining the original data in the stack enables parallel manipulation and transformation of variables within incomplete and imputed data. Retaining the original data also allows complete-case analyses to be performed by applying the restriction if _mj==0.

2.2 Estimation for MI datasets

mim is designed mainly for the creation of combined parameter estimates from an ensemble of imputed datasets. It allows the creation of combined estimates for regression coefficients obtained from any command that has the standard Stata estimation command structure. All commonly used commands,[1] including those taking the svy prefix, are recognized directly by the mim prefix command, and others can also be used by specifying the category(fit) option with mim (see section 3). In the latter case, the user must take responsibility for the results because mim will not automatically reflect any nonstandard characteristics of commands that are not in the recognized list. While most Stata estimation commands—including those using multiple-equation models—should work seamlessly with mim, the user should pay attention to a command's handling of any ancillary parameters. Often these are calculated on the log scale but back-transformed for display purposes, and the associated t or z statistics, and their p-values, are sometimes suppressed. When a command that has these characteristics but is not in the recognized list is used with mim, all parameters will be displayed on the same scale in which they are calculated, and the corresponding t statistics and p-values will be displayed, whether or not they are valid. This behavior is consistent with Stata's ereturn display command.

2.3 Postestimation with MI

The method of MI was developed with a focus on the canonical activity of estimating regression models. We have maintained this focus here although Rubin's rules can be applied to any estimand for which approximate normality of the estimate is reasonably assured (and in a later release of mim, we plan to provide a more generic capability to create combined estimates for any user-defined scalar estimator). Rubin's combination rules have been shown to work well for scalar estimands, especially when a small-sample adjustment is applied to the degrees of freedom used for the t reference distribution (Barnard and Rubin 1999) (and assuming that the method of imputation is *proper*; Rubin [1996]). For standard fitting of regression models, the scalar approach is adequate because the estimation of each coefficient in the linear predictor may be treated separately from the other coefficients, using each coefficient's estimated standard error or variance.

[1] These are regress, mean, proportion, ratio, logistic, logit, ologit, mlogit, probit, oprobit, poisson, glm, binreg, nbreg, gnbreg, blogit, clogit, cnreg, mvreg, rreg, qreg, iqreg, sqreg, bsqreg, stcox, streg, xtgee, xtreg, xtlogit, xtnbreg, xtpoisson, xtmixed, svy: regress, svy: mean, svy: proportion, svy: ratio, svy: logistic, svy: logit, svy: ologit, svy: mlogit, svy: probit, svy: oprobit, and svy: poisson.

However, several subsidiary estimation tasks or hypothesis tests that involve more than one coefficient are often of interest. These are managed for all standard estimation commands in Stata with a range of auxiliary commands under the heading of postestimation. We believe that some, but not all, of these standard postestimation commands can be validly translated to the MI context with our current understanding of MI. `mim` currently has the facility to handle `lincom`, `testparm`, and `predict`, which respectively provide estimates for linear combinations of the regression parameters, Wald-type hypothesis tests for groups of regression coefficients considered simultaneously, and estimates of predicted values for the units of the original dataset. (Note that postestimation methods relying on likelihood comparisons (`lrtest`) are not applicable because MI does not involve calculation of likelihood functions for the data.)

The MI version of `lincom` is straightforward; it simply requires application of Rubin's rules to the (scalar) linear combination that is of interest. However, multiparameter hypothesis testing is less straightforward because it is not clear that a valid pooled variance–covariance matrix (in a multiparameter problem) can always be obtained by a simple averaging process (Schafer 1997). We have implemented an MI version of `testparm` using the method of Li, Raghunathan, and Rubin (1991), but in this first release we have not provided a full translation of the `test` command, of which `testparm` is a special case with a more limited range of syntax. Users may apply any of Stata's postestimation commands that rely on the standard structure of Stata's returned results (in particular the vector of estimates `e(b)` and variance–covariance matrix `e(V)`) by requesting that MI values of these quantities be placed in the standard returned results. `mim` does not do this by default because we do not believe that there is adequate theory to support all the possible resulting calculations and, in particular, because of the difficulty just mentioned of ensuring a valid variance–covariance matrix. The user is referred to `help mim` for details of the objects returned in `e()` when `mim` is used with an estimation command.

We have also provided a limited implementation of Stata's `predict` command under the `mim` prefix. This produces estimates of predicted values at each observation in the estimation dataset by treating the estimand $X_i\beta$ for each observation i as a scalar parameter to which the Rubin combination formulas are applied. This calculation will often use values of X_i that are missing in the original data. A more general approach to prediction, which would allow predictions to be created for "synthetic" observations (appended as new rows of data), is a more complex task that we have not yet addressed. It requires a method for creating a joint inference for the vector β of regression coefficients in a linear predictor and then applying this to whatever set or sets of X values are specified.

2.4 Data manipulation with MI datasets

The final category of subcommands that `mim` handles are those that manipulate and transform data. Our experience is that, for practical work with complex datasets, it is essential to have the capacity to work flexibly with data after imputation has been performed. For example, imputation may be performed on raw variables that must then be categorized or transformed in various ways to be used in planned analyses. With the previous `mitools`, imputed datasets were stored separately, so a command for managing manipulation of each imputed dataset in an ensemble was needed. In the `mim` environment, most data manipulations (`generate`, `replace`, `recode`, etc.) can be simply applied to the single stacked dataset. Assuming that the original data with missing values has been retained in the stacked `mim` dataset (with `_mj` = 0), the specification of data transformations should appropriately allow for any missing values, i.e., in general by explicit exclusions such as "`if var!=.`".

`mim` was specifically programmed for three data manipulation commands (`reshape`, `append`, and `merge`) that cannot simply be applied to the stacked dataset because they require that proper attention be paid to the repeated dataset structure. The `sortorder()` option is required for the use of `merge`, in order to guarantee preservation of the observation identifier across merged datasets, because the `_mi` index must be dropped while the data manipulation is performed.

As with estimation commands, other data manipulation commands may be applied at the user's discretion by specifying the option `category(manip)`, which essentially allows a command to be applied to each dataset separately, with the resulting datasets stacked back into the same structure as used originally. Note, however, that certain data transformations, such as those that generate new observations (e.g., `expand`), may produce meaningless results in the context of an MI dataset.

The `mim` prefix also supports two newly written utility subcommands: `check` and `genmiss`. The former provides a check as to whether the dataset in memory has a `mim`-compatible structure containing the indexing variables `_mj` and `_mi`. The main checks are that nonmissing values must be constant across imputed datasets and that all missing values must have been imputed. `genmiss` creates an indicator variable to contain the missing/observed status of a selected variable. These utility subcommands require that the original dataset with missing values has been included in the stacked dataset.

While `mim` is designed to facilitate the handling of multiply imputed datasets, the user should be aware of a number of other utilities that are available in Stata for managing and manipulating missing data more generally. These range from the user-written command `mvpatterns`, which enables a detailed summary of patterns of missing data, to various usages of standard Stata functions. In particular, the `rowmiss` function of `egen` is a handy tool for identifying the extent to which missing data affect observations in a dataset, as for example in `egen int nmiss = rowmiss(`*varlist*`) if _mj==0`, which would create a variable containing the number of missing values in *varlist*.

Finally, the `mim` package includes one auxiliary command, `mimstack`, which creates a `mim`-compatible dataset from an appropriate set of imputed datasets, with or without the original incomplete data.

3 Syntax

mim [, *mim_options*] : *command*

mim [, *replay_options*]

mim_options	description
General	
category(fit|manip)	specify whether *command* is estimation or data manipulation
noisily	display output from execution of *command* within each of the imputed datasets
Estimation (valid only for estimation commands)	
dots	display progress dots during model fitting
noindividual	suppress capture of results from each application of *command*
storebv	fill the standard list (e(b), e(V), etc.) of returned results for estimation commands with MI estimates
Manipulation (valid only for data manipulation commands)	
sortorder(*varlist*)	one or more variables that uniquely identify the observations in a given imputed dataset following each execution of *command*

replay_options	description
clearbv	clears the standard list (e(b), e(V), etc.) of returned results for estimation commands, but leaves intact all other items returned by mim
j(#)	fills the standard list (e(b), e(V), etc.) of returned results for estimation commands with the estimates corresponding to imputed dataset #
reporting_options	level() and eform options supported by *command*
storebv	same as for estimation, unless j() option is specified

xi is allowed as a prefix to mim but not as a prefix to *command*.
svy is allowed as a prefix to *command*.
version is allowed as a prefix to *command*.

4 Options

4.1 General

`category(fit|manip)` is not required for the estimation and data manipulation commands that are listed in section 2. However, it is required when any other command is used to specify the type of command that is being passed to `mim:` either estimation (`category(fit)`) or data manipulation (`category(manip)`).

`noisily` specifies that the results of the application of *command* to each of the individual imputed datasets should be displayed.

4.2 Estimation

`dots` specifies that progress dots should be displayed.

`noindividual` specifies that capture of the estimation results corresponding to the fitting of the given estimation command to each of the individual imputed datasets should be suppressed.

`storebv` specifies that the standard list of returned results for estimation commands be filled using the MI results, forcing the MI coefficient and covariance matrix estimates into `e(b)` and `e(V)`, respectively. This enables subsequent application, at the user's discretion, of Stata postestimation commands that use these quantities directly.

4.3 Manipulation

`sortorder(`*varlist*`)` must specify a list of one or more variables that uniquely identifies the observations in each of the datasets in a `mim`-compatible dataset after *command* has been applied to the given dataset (*varlist* cannot include `_mi` because the `_mj` and `_mi` variables are dropped from each dataset prior to the call to *command*). This option is not valid for `append` and `reshape` but is *mandatory* for all other data manipulation commands.

4.4 Replay

`clearbv` specifies that the standard list (`e(b)`, `e(V)`, etc.) of returned results for estimation commands be cleared. All other (`eclass`) items returned specifically by `mim` are left intact.

`j(`#`)` specifies that the standard list (`e(b)`, `e(V)`, etc.) of returned results for estimation commands be filled with the estimates from the #th imputed dataset.

reporting_options may include any `level()` and `eform` options supported by *command*.

`storebv` specifies that the standard list (`e(b)`, `e(V)`, etc.) of returned results for estimation commands be filled with the MI estimates, unless the `j()` option is specified.

(There are no *mim_options* for `mim: predict`, `mim: check`, and `mim: genmiss`.)

5 Example: Adolescent health cohort study

Our first illustration uses a dataset adapted from an adolescent health cohort study that was used by Carlin et al. (2003) to introduce the original `mitools` commands. Imputation was performed for this study using the stand-alone package NORM, based on fitting a multivariate normal distribution (followed by appropriate rounding of categorical variables). This produces imputations in separate files, which we may combine into a `mim` format by applying `mimstack`. Imputations were performed separately for males and females in order to preserve interactions with gender, so we first load and stack five imputed datasets (`smiF`*`.dta`) with the female participants, followed by a similar process with the male participants:

```
. mimstack, m(5) sortorder(id wave) istub(smiF) clear
. save smifimp5, replace
file smifimp5.dta saved
. mimstack, m(5) sortorder(id wave) istub(smiM) clear
. save smimimp5, replace
file smimimp5.dta saved
```

We then join the two `mim` datasets into one by using `mim: append`, at the same time creating a variable `sex` to identify the two genders. The `check` utility is used to verify that we have created a dataset in `mim`-compatible format.

```
. use smifimp5, clear
. gen byte sex = 1
. mim: append using smimimp5
. replace sex = 0 if sex == .
(3420 real changes made)
. label define sexlb 0 "male" 1 "female"
. label values sex sexlb
. mim: check
..........
PASS
. save smiall, replace
file smiall.dta saved
```

The `mim` dataset `smiall.dta` is now ready for analysis and will remain in memory during the course of subsequent `mim` commands.

(*Continued on next page*)

```
. describe
Contains data from smiall.dta
  obs:         7,020
 vars:            12                          13 Mar 2008 07:48
 size:       140,400 (86.6% of memory free)   (_dta has notes)
--------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
--------------------------------------------------------------------------------
_mj             byte   %8.0g                  imputation identifier
_mi             int    %8.0g                  observation identifier
id              long   %9.0g
wave            byte   %9.0g                  survey wave
mmetro          byte   %9.0g                  school in metro area
parsmk          byte   %9.0g                  either parent smokes
drkfre          byte   %16.0g      drkfre     drinking frequency
alcdos          byte   %21.0g      alcdos     av units/drinking day
alcdhi          byte   %9.0g                  drank >=5 units at least once
smk             byte   %13.0g      smk        smoking status
cistot          byte   %9.0g                  CIS total score
sex             byte   %8.0g       sexlb
--------------------------------------------------------------------------------
Sorted by:  _mj  _mi
```

When using MI, it is sometimes useful to informally examine the variation in values across imputed datasets. This can be done with standard Stata syntax by using the `_mj` index. For example, one could examine the distribution of drinking frequency (a four-category variable) among imputed and nonimputed cases by running tables as follows:

```
. mim: genmiss drkfre
. by _mj: tabulate drkfre _mim_drkfre, col
  (output omitted)
```

To illustrate a more targeted analysis, we generate a binary variable `drkreg` and obtain estimates of the frequency of regular drinking at each wave by using the command `mim: proportion`. (`proportion` is a Stata estimation command—available from release 9—and so has been incorporated into the standard `mim` structure, making it unnecessary to have a separate command such as `mici` in the previous `mitools`.)

```
. gen drkreg = (drkfre >= 2) if drkfre != .
(163 missing values generated)
. forvalues num = 1/6 {
  2. dis "wave: " `num´
  3. mim: proportion drkreg if wave==`num´
  4. }
wave: 1
Multiple-imputation estimates (proportion)              Imputations =       5
Proportion estimation                                   Minimum obs =     195
                                                        Minimum dof =   180.3

------------------------------------------------------------------------------
             |     Coef.  Std. Err.      t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
           0 |   .876923   .023918   36.66   0.000    .829728  .924118   180.3
           1 |   .123077   .023918    5.15   0.000    .075882  .170272   180.3
------------------------------------------------------------------------------
  (output omitted)
```

Issuing the command `mim` on its own replays the last set of results produced by a `mim` estimation command, in this case for `wave==6`:

```
. mim
Multiple-imputation estimates (proportion)              Imputations =       5
Proportion estimation                                   Minimum obs =     195
                                                        Minimum dof =    35.2

------------------------------------------------------------------------------
             |     Coef.  Std. Err.      t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
           0 |   .644103   .040777   15.80   0.000    .561336  .726869    35.2
           1 |   .355897   .040777    8.73   0.000    .273131  .438664    35.2
------------------------------------------------------------------------------
```

The MI (combined) estimates are displayed using a standard Stata format with a few variations to convey important information about the MI results. The number of imputed datasets is shown, and under this we have the *minimum* number of observations available for each of the separate analyses. In many cases (including the example shown here), the number of observations will be identical across imputed datasets, but this is not the case if the estimation is performed on a subset of the data defined by restriction according to a variable that is subject to missing values. In that case, the sample used for estimation will generally differ across imputations. Displaying the minimum sample size is a conservative approach; for some purposes, the user may prefer to obtain the average to display in tables of results. The final column in the table contains the approximate degrees of freedom (Barnard and Rubin 1999) that are used for defining the t multiplier underlying the confidence interval calculation. This column also gives a useful index of the extent to which missingness has affected the information available for the estimation of each parameter. The value "Minimum dof" gives the minimum of the "`MI.df`" across the effects that have been estimated (as well as across the datasets of varying size, if applicable). In this example of `proportion` applied to a binary variable, the standard error and associated degrees of freedom are identical for each of the two complementary proportions.

The variation in results underlying the combined estimate, across imputed datasets, could be examined by replaying the single imputation results, as follows:

```
. forvalues num = 1/5 {
  2. mim, j(`num´)
  3. }
  (output omitted)
```

An important feature is that `mim` can take the `xi` prefix to generate interactions and dummy variables in the standard way. We illustrate this with a logistic regression that examines evidence for a different rate of change with `wave` between the `sex`es by fitting an interaction model. The (incorrect) independent-observations likelihood is used for estimation (i.e., the standard `logistic` command) with standard errors obtained by the robust sandwich method in order to allow for correlation between repeated measures on the same subjects.

```
. xi: mim: logistic drkreg i.sex*wave, cluster(id)
i.sex             _Isex_0-1           (naturally coded; _Isex_0 omitted)
i.sex*wave        _IsexXwave_#        (coded as above)

Multiple-imputation estimates (logistic)                 Imputations =      5
Logistic regression                                      Minimum obs =   1170
                                                         Minimum dof =  228.7

------------------------------------------------------------------------------
      drkreg | Odds Rat.  Std. Err.     t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
     _Isex_1 |   .522541   .203362   -1.67   0.096    .243466  1.12151   975.9
        wave |   1.22544   .071734    3.47   0.001    1.09194  1.37526   228.7
_IsexXwave_1 |   1.03796   .084476    0.46   0.647    .884479  1.21807   391.7
------------------------------------------------------------------------------
```

This model may be used to illustrate the use of `mim: lincom`; we estimate the odds ratio for regular drinking among males as follows:

```
. mim: lincom wave + _IsexXwave_1
Multiple-imputation estimates for lincom                       Imputations = 5

 ( 1)  wave + _IsexXwave_1 = 0

------------------------------------------------------------------------------
      drkreg | Odds Rat.  Std. Err.     t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
         (1) |   1.27195   3.37316    0.09   0.928    .006988  231.506   997.9
------------------------------------------------------------------------------
```

We note again that `mim` recognizes the `logistic` command and so by default returns estimates in exponentiated form, labeled appropriately as odds ratios. When a similar logistic regression model is estimated using the generalized estimating equations method, the default display of the estimates is in the log scale, i.e., as the coefficients in the linear predictor. However, exponentiated coefficients may be obtained as usual by using the `eform` option.

```
. mim: xtgee drkreg sex wave, fam(binom) i(id)
Multiple-imputation estimates (xtgee)                   Imputations =      5
                                                        Minimum obs =   1170
                                                        Minimum dof =   78.6

------------------------------------------------------------------------------
      drkreg |     Coef.  Std. Err.     t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
         sex |  -.493965    .23835   -2.07   0.040   -.964602 -.023328   163.7
        wave |   .219717   .038719    5.67   0.000    .142644   .29679    78.6
       _cons |  -1.67304   .224338   -7.46   0.000   -2.11679 -1.22929   132.3
------------------------------------------------------------------------------
```

We illustrate the multiparameter postestimation capabilities of `mim` by including further covariates in the model:

```
. gen cisgp = cistot
(165 missing values generated)
. recode cisgp 0/5=1 6/11=2 12/100=3
(cisgp: 6300 changes made)
. xi: mim: xtgee drkreg sex wave i.cisgp, fam(binom) i(id) eform
i.cisgp           _Icisgp_1-3         (naturally coded; _Icisgp_1 omitted)
Multiple-imputation estimates (xtgee)                   Imputations =      5
                                                        Minimum obs =   1170
                                                        Minimum dof =   74.1

------------------------------------------------------------------------------
      drkreg |    exp(b)  Std. Err.     t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
         sex |   .558385   .133963   -2.43   0.016    .348033  .895874   226.0
        wave |   1.28066   .052581    6.03   0.000    1.18006  1.38983    74.1
   _Icisgp_2 |   1.00571   .192688    0.03   0.976    .689186   1.4676   189.2
   _Icisgp_3 |   1.77553   .353741    2.88   0.004    1.19932  2.62857   255.8
------------------------------------------------------------------------------

. mim: testparm _Icis*
 ( 1)  _Icisgp_2 = 0
 ( 2)  _Icisgp_3 = 0
       F(  2, 221.4) =    4.76
            Prob > F =    0.0095
```

A test of the overall null hypothesis of no differences between the three groups defined by the `cisgp` variable (a categorical indicator of mental health) was obtained by using the `mim: testparm` command.

Because these data relate to measures taken on repeated occasions, some analyses may best be handled by reshaping the data to wide form. This is accomplished by using `mim: reshape`:

```
. gen drkany = (drkfre >= 1) if drkfre != .
(163 missing values generated)
. keep _mj _mi id wave drkany cisgp sex
. mim: reshape wide drkany cisgp, i(id) j(wave)
```

We can now obtain an estimate of the incidence of alcohol use between waves 1 and 2:

```
. mim: proportion drkany2 if drkany1 == 0
Multiple-imputation estimates (proportion)                Imputations =       5
Proportion estimation                                     Minimum obs =     136
                                                          Minimum dof =   122.6

------------------------------------------------------------------------------
             |      Coef.  Std. Err.      t    P>|t|    [95% Conf. Int.]  MI.df
-------------+----------------------------------------------------------------
           0 |    .696501   .040309   17.28   0.000    .616709  .776292  122.6
           1 |    .303499   .040309    7.53   0.000    .223708  .383291  122.6
------------------------------------------------------------------------------
```

Logistic regression can be used to examine the association between incidence and covariates of interest:

```
. xi: mim: logistic drkany2 i.cisgp1 if drkany1 == 0
i.cisgp1          _Icisgp1_1-3        (naturally coded; _Icisgp1_1 omitted)
Multiple-imputation estimates (logistic)                  Imputations =       5
Logistic regression                                       Minimum obs =     136
                                                          Minimum dof =   524.7

------------------------------------------------------------------------------
     drkany2 | Odds Rat.  Std. Err.      t    P>|t|    [95% Conf. Int.]  MI.df
-------------+----------------------------------------------------------------
  _Icisgp1_2 |    .864098   .406523   -0.31   0.756    .343045  2.17658  642.2
  _Icisgp1_3 |    1.06857   .484563    0.15   0.884    .438557  2.60365  595.8
------------------------------------------------------------------------------
```

This analysis provides an example where the size of the imputed dataset used in each of the single-imputation analyses varies because the condition `drkany1==0` produces a different set of observations (because `drkany` was subject to missingness and so varies across imputations).

6 Example: Breast cancer

We use a second example, taken from Royston (2004), to illustrate the use of `ice` to obtain multiply imputed data, followed by `mim` to handle and analyze the imputations.

First, the raw data containing missing values is loaded, and `stset` is used to specify a survival time structure for later analysis. This could have been done subsequently although the summary information provided would be potentially misleading because it would reflect the number of imputed datasets that were created. Second, five imputations of the missing values are created using `ice` (version 1.4.0; Royston 2007), saving the imputations to a new file, `brcaeximp2b.dta`. We use the `match()` option for the variable `mx6` because it has an extremely skewed, semicontinuous distribution that makes it difficult to impute using a parametric model.

```
. use brcaex, clear
(German breast cancer data)

. stset rectime, fail(censrec)

     failure event:  censrec != 0 & censrec < .
obs. time interval:  (0, rectime]
 exit on or before:  failure

------------------------------------------------------------------------------
      686  total obs.
        0  exclusions
------------------------------------------------------------------------------
      686  obs. remaining, representing
      299  failures in single record/single failure data
   771400  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =      2659

. ice mx1 mx4a mx5e mx6 mhormon lnt _d using brcaeximp2b, match(mx6) m(5)
> genmiss(m_) seed(101) replace

   #missing |
     values |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 |        231       33.67       33.67
          1 |        290       42.27       75.95
          2 |        126       18.37       94.31
          3 |         33        4.81       99.13
          4 |          6        0.87      100.00
------------+-----------------------------------
      Total |        686      100.00

   Variable | Command | Prediction equation
------------+---------+-------------------------------------------------------
        mx1 | regress | mx4a mx5e mx6 mhormon lnt _d
       mx4a | logit   | mx1 mx5e mx6 mhormon lnt _d
       mx5e | regress | mx1 mx4a mx6 mhormon lnt _d
        mx6 | regress | mx1 mx4a mx5e mhormon lnt _d
    mhormon | logit   | mx1 mx4a mx5e mx6 lnt _d
        lnt |         | [No missing data in estimation sample]
         _d |         | [No missing data in estimation sample]
------------------------------------------------------------------------------

Imputing 1..2..3..4..5..(note: file brcaeximp2b.dta not found)
file brcaeximp2b.dta saved
```

A plot of the distributions of observed and imputed values of one of the variables subject to missing data (`mx1`) illustrates the variability between imputations but reveals a similar distribution for the imputed values as for the observed, although one of the imputed distributions is somewhat different from the others (figure 1). Slightly different results will be obtained each time the imputation procedure is performed (unless the `seed()` option is used in `ice`); this is a natural feature of the method.

(*Continued on next page*)

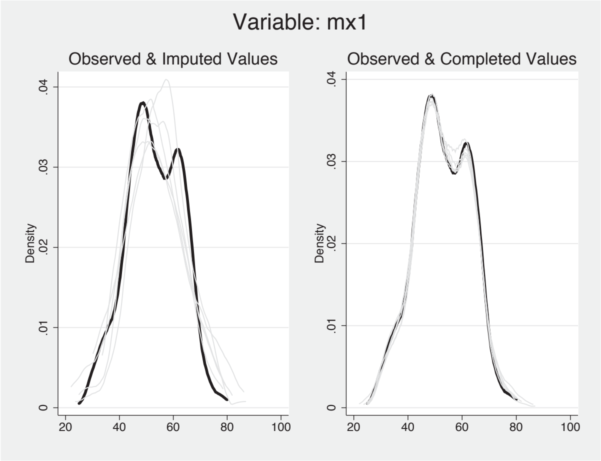


Figure 1. Frequency plots of the observed and imputed values of the variable `mx1` in the breast cancer example. The left-hand panel superimposes the distribution of the five sets of imputed values (light-gray lines) on the distribution of the observed values (black lines), while the right-hand panel displays the distribution of the completed data—observed and imputed values combined—along with the incomplete observed data distribution.

Fractional polynomial transformations are applied to `mx1` and `mx6` for modeling purposes:

```
. use brcaeximp2b, clear
(German breast cancer data)

. fracgen mx1 -2 -0.5
-> gen double mx1_1 = X^-2
-> gen double mx1_2 = X^-0.5
   (where: X = mx1/10)

. fracgen mx6 0.5
-> gen double mx6_1 = X^0.5
   (where: X = (mx6+1)/1000)
```

The model is fitted in each imputed dataset and combined estimates are obtained:

```
. mim: stcox mx1_1 mx1_2 mx4a mx5e mx6_1 mhormon, nohr
Multiple-imputation estimates (stcox)                Imputations =       5
                                                     Minimum obs =     686
                                                     Minimum dof =     8.8

------------------------------------------------------------------------------
          _t |     Coef.  Std. Err.      t    P>|t|    [95% Conf. Int.]   MI.df
-------------+----------------------------------------------------------------
       mx1_1 |    36.459   19.3132    1.89   0.092   -7.38798  80.3059     8.8
       mx1_2 |  -14.9337   8.25602   -1.81   0.103   -33.5639  3.69646     9.1
        mx4a |     .5223   .290443    1.80   0.075   -.053744  1.09834   102.7
        mx5e |  -1.86353   .273425   -6.82   0.000     -2.417 -1.31005    38.1
       mx6_1 |  -1.97985   .440288   -4.50   0.000   -2.87119  -1.0885    38.0
     mhormon |  -.422055   .163597   -2.58   0.016   -.757682 -.086429    27.1
------------------------------------------------------------------------------
```

7 Conclusions

The field of MI data analysis is still young, but it is quickly growing and increasingly offers the possibility of more efficient and more informative analyses of important datasets, particularly in the social and health sciences. Following the success of our earlier package `mitools` (Carlin et al. 2003) and of the package `ice` for multiple imputation of missing values (Royston 2004, 2005b), our new package `mim` further rationalizes and advances the management and analysis of MI datasets. The approach used by `mim` requires all imputed copies of the dataset to be stored together in stacked format, allowing all analysis to take place using the single dataset in memory. This approach is conceptually appealing in that it reminds the analyst that the individual imputed datasets should not be taken too seriously on their own: it is only by analyzing the multiply imputed datasets and appropriately combining results that valid inferences may be obtained. In this sense, the imputed data are naturally viewed as an extension of the original data. Use of the `mim` framework does, however, require that the user not forget that they are using a multiply imputed dataset; it is easy to mistakenly apply commands to the entire stacked dataset with the illusion of having several times more observations than actually exist. Clearly, there is an inherent complexity in using MI, which requires that the user always needs to be alert to such issues.

While there is certainly room for further development of `mim` (for example, to extend the `test` postestimation command), we believe the current version already provides a rich set of facilities for the analysis of MI data and for research on MI inference. Examples of research questions with MI data include how to build multivariable models from a set of candidate variables and how to construct suitable model performance summaries and diagnostics. More broadly, important questions remain unanswered about the use of MI: for example, how sensitive are results to the use of inappropriate imputation methods, and are there ways in which users can check the validity of their imputations and resulting analytic conclusions? As mentioned in this article's introduction, the only imputation methods that are widely available in standard software assume that the data are "missing at random" according to Rubin's technical definition (Little and Rubin 2002). Although this assumption cannot, by definition, be tested in the data being analyzed, the user should consider whether it has a good basis in the context of his or her application, and it would be helpful to have more research on the sensitivity of results to departures from missing at random (e.g., Carpenter, Kenward, and White 2007).

We hope to be able to update `mim` on a regular basis as relevant research on handling statistical issues with MI data is published and in response to user queries and suggestions. We also hope that users will develop Stata implementations of alternative methods for imputation and make them compatible with the `mim` environment so that comparative analyses are facilitated. (For example, it would be valuable to have a Stata version of Schafer's NORM.) Therefore, we welcome user input to help us further develop `mim`.

8 Acknowledgments

We are grateful to Ian White and Carolyn Coffey for testing earlier versions of `mim` and for making helpful suggestions during its development, and to Mark Lunt for suggesting a substantial improvement in the coding of `mim: predict`.

9 References

Barnard, J., and D. B. Rubin. 1999. Small-sample degrees of freedom with multiple imputation. *Biometrika* 86: 948–955.

Carlin, J. B., N. Li, P. Greenwood, and C. Coffey. 2003. Tools for analyzing multiple imputed datasets. *Stata Journal* 3: 226–244.

Carpenter, J. R., M. G. Kenward, and I. R. White. 2007. Sensitivity analysis after multiple imputation under missing at random: a weighting approach. *Statistical Methods in Medical Research* 16: 259–275.

Li, K. H., T. E. Raghunathan, and D. B. Rubin. 1991. Large-sample significance levels from multiply imputed data using moment-based statistics and an F reference distribution. *Journal of the American Statistical Association* 86: 1065–1073.

Little, R. J. A., and D. B. Rubin. 2002. *Statistical Analysis with Missing Data*. 2nd ed. New York: Wiley.

Potthoff, R. F., G. E. Tudor, K. S. Pieper, and V. Hasselblad. 2006. Can one assess whether missing data are missing at random in medical studies? *Statistical Methods in Medical Research* 15: 213–234.

Royston, P. 2004. Multiple imputation of missing values. *Stata Journal* 4: 227–241.

———. 2005a. Multiple imputation of missing values: update. *Stata Journal* 5: 188–201.

———. 2005b. Multiple imputation of missing values: update of ice. *Stata Journal* 5: 527–536.

———. 2007. Multiple imputation of missing values: further update of ice, with an emphasis on interval censoring. *Stata Journal* 7: 445–464.

Rubin, D. B. 1996. Multiple imputation after 18+ years. *Journal of the American Statistical Association* 91: 473–489.

Schafer, J. L. 1997. *Analysis of Incomplete Multivariate Data*. London: Chapman & Hall.

van Buuren, S., H. C. Boshuizen, and D. L. Knook. 1999. Multiple imputation of missing blood pressure covariates in survival analysis. *Statistics in Medicine* 18: 681–694.

About the authors

John Carlin is a biostatistician with broad experience across a range of collaborative research relating mainly to child and adolescent health, and with current methodological research interests in the handling of missing data in large longitudinal studies. He is the director of the Clinical Epidemiology and Biostatistics Unit, Murdoch Children's Research Institute, at the Royal Children's Hospital in Melbourne, Australia, and has professorial appointments in the Department of Paediatrics and School of Population Health at the University of Melbourne.

John Galati was a research officer within the Clinical Epidemiology and Biostatistics Unit, Murdoch Children's Research Institute, at the time this work was completed. He has a PhD in pure mathematics and has worked on various projects in health and medical research since 2004, as well as having substantial computer programming and systems analysis experience.

Patrick Royston is a medical statistician with 30 years of experience, with a strong interest in biostatistical methodology and in statistical computing and algorithms. He works in clinical trials and related research issues in kidney cancer and other cancers. Currently, he is focusing on problems of model building and validation with survival data, including prognostic factors studies; on complex sample size problems in clinical trials with a survival-time endpoint; on writing a book on multivariable regression modeling; and on new trial designs.

The Stata Journal (2008)
8, Number 1, pp. 68–78

Tests for unbalanced error-components models under local misspecification

Walter Sosa-Escudero
Department of Economics
Universidad de San Andrés
Buenos Aires, Argentina
wsosa@udesa.edu.ar

Anil K. Bera
Department of Economics
University of Illinois at Urbana–Champaign
Champaign, IL

Abstract. This paper derives *unbalanced* versions of the test statistics for first-order serial correlation and random individual effects summarized in Sosa-Escudero and Bera (2001, *Stata Technical Bulletin Reprints*, vol. 10, pp. 307–311), and updates their `xttest1` routine. The derived test statistics should be useful for applied researchers faced with the increasing availability of panel information where not every individual or country is observed for the full time span. The test statistics proposed here are based on ordinary least-squares residuals and hence are computationally very simple.

Keywords: sg164_1, xttest1, error-components model, unbalanced panel data, testing, misspecification

1 Introduction

A standard specification check that accompanies the output of almost every estimated error-components model is a simple test for the presence of random individual effects. The well-known Breusch–Pagan statistic (Breusch and Pagan 1980), based on the Rao-score (RS) principle, is a frequent choice. Bera, Sosa-Escudero, and Yoon (2001) demonstrated that, in the presence of first-order serial correlation, the test too often rejects the correct null hypothesis of no random effects. Consequently, they propose a modified version that is not affected by the presence of local serial correlation. A similar concern affects the standard test for first-order serial correlation derived by Baltagi and Li (1991), which overrejects the true null hypothesis when random effects are present. For this case, an adjusted RS test was also derived by Bera, Sosa-Escudero, and Yoon (2001). These test statistics, along with their `xttest1` routine in Stata and some empirical illustrations, are presented in Sosa-Escudero and Bera (2001). For a textbook exposition, see Baltagi (2005, 96–97).

These test procedures were originally derived for the *balanced* case, that is, in the panel-data terminology, the case where all individuals are observed for the same number of periods, and in every period all individuals are observed. On the other hand, in applied work the availability of *unbalanced* panels is far from being an uncommon situation. Though in some cases statistical procedures designed for the balanced case can be straightforwardly extended to accommodate unbalanced panels, many estimation or test procedures require less trivial modifications.

Baltagi and Li (1990) derived an unbalanced version of the Breusch–Pagan statistic. The purpose of this paper is to derive unbalanced versions of the test for first-order serial correlation originally proposed by Baltagi and Li (1991) and of the modified tests proposed by Bera, Sosa-Escudero, and Yoon (2001). As a simple extension, we also derive an unbalanced version of the joint test of serial correlation and random effects proposed by Baltagi and Li (1991). The derived test statistics, being based on ordinary least-squares residuals after pooled estimation, are computationally very simple. Finally, the Sosa-Escudero and Bera (2001) `xttest1` routine is appropriately updated to handle unbalanced panels.

2 Tests for the unbalanced case

Consider a simple linear model for panel data allowing for the presence of random individual effects and first-order serial correlation:

$$\begin{aligned} y_{it} &= x_{it}'\beta + u_{it} \\ u_{it} &= \mu_i + \nu_{it} \\ \nu_{it} &= \lambda\nu_{i,t-1} + \epsilon_{it}, \qquad |\lambda| < 1 \end{aligned}$$

where x_{it} is a $k \times 1$ vector of explanatory variables with 1 in its first position, β is a $k \times 1$ vector of parameters including an intercept, $\mu_i \sim N(0, \sigma_\mu^2)$, and $\epsilon_{it} \sim N(0, \sigma_\epsilon^2)$. We will assume $\nu_{i,0} \sim N\left\{0, \sigma_\epsilon^2/(1-\lambda^2)\right\}$.

We will be interested in testing for the absence of random effects ($H_0 : \sigma_\mu^2 = 0$) and/or first-order serial correlation ($H_0 : \lambda = 0$). The panel will be unbalanced in the sense that for every individual $i = 1, \ldots, N$ we will observe, possibly, a different number of time observations T_i. We will restrict the analysis to the cases where missing observations occur either at the beginning or at the end of the sample period for each individual (that is, there are no "gaps" in the series), and the starting and final periods are determined randomly. Hence, without loss of generality and to avoid complicating the notation too much, we can safely assume that the series for each individual starts at the same period ($t = 1$) and finish randomly at period $t = T_i$.

Let $m = \sum_{i=1}^N T_i$ be the total number of observations. Let u be an $m \times 1$ vector with typical element u_{it} where observations are sorted first by individuals and then by time, so the time index is the faster one. Then in our setup, $V(u) \equiv \Omega$ can be written as

$$V(u) = \sigma_\mu^2 \widetilde{H} + \sigma_\epsilon^2 \widetilde{V}$$

where $\widetilde{H}$ is an $m \times m$ block diagonal matrix with blocks H_i equal to matrices of ones, each with dimensions $T_i \times T_i$. Similarly, $\widetilde{V}$ will be a block diagonal $m \times m$ matrix with blocks V_i equal to

$$V_i = \begin{bmatrix} 1 & \lambda & \lambda^2 & \dots & \lambda^{T_i-1} \\ \lambda & 1 & \lambda & \dots & \lambda^{T_i-2} \\ \vdots & \vdots & \vdots & & \vdots \\ \lambda^{T_i-1} & \lambda^{T_i-2} & \lambda^{T_i-3} & \dots & 1 \end{bmatrix}$$

For the purpose of deriving the test statistics, the log-likelihood function will be

$$L(\beta, \lambda, \sigma_\epsilon^2, \sigma_\mu^2) = \text{constant} - \frac{1}{2}\log|\Omega| - \frac{1}{2}u'\Omega^{-1}u$$

The information matrix for this problem is known to be block diagonal between β and the remaining parameters. Therefore, for the purposes of this paper, we will concentrate only on the parameters λ, σ_μ^2, and σ_ϵ^2. Under a more general setup, suppose the log likelihood can be characterized by a three-parameter vector $\theta = (\psi, \phi, \gamma)'$. Let $d(\theta)$ be the score vector and $J(\theta)$ the information matrix. If it can be assumed that $\phi = 0$, the standard Rao-score (RS) test statistic for the null hypothesis $H_0 : \psi = 0$ is given by

$$\mathrm{RS}_\psi = d_\psi(\widehat{\theta}) J_{\psi\cdot\gamma}^{-1}(\widehat{\theta}) d_\psi(\widehat{\theta}) \tag{1}$$

where d_ψ is the element of the score corresponding to the parameter ψ, $J_{\psi\cdot\gamma}(\theta) = J_\psi - J_{\psi\gamma}J_\gamma^{-1}J_{\gamma\psi}$, and $\widehat{\theta}$ is the maximum likelihood estimator (MLE) of θ under the restriction implied by the null hypothesis and the assumption $\phi = 0$. Asymptotically, this test statistic under the null hypothesis H_0: $\psi = 0$ is known to have a central chi-squared distribution. In the context of our error-components model, if $\gamma = \sigma_\epsilon^2$ and if we set $\psi = \sigma_\mu^2$ and $\phi = \lambda$, (1) is a test for random effects assuming no serial correlation; and if we set $\psi = \lambda$ and $\phi = \sigma_\mu^2$, (1) gives a test for serial correlation assuming no random effects. The standard Breusch–Pagan test for random effects (assuming no serial correlation) and the Baltagi–Li test for first-order serial correlation (assuming no random effects) are derived from this principle.

Bera and Yoon (1993) showed that the test statistic (1) is invalid when $\phi \neq 0$, in the sense that the test tends to reject the null hypothesis too frequently even when it is correct. More specifically, the RS_ψ statistic is found to have an asymptotic *noncentral* chi-squared distribution under $H_0 : \psi = 0$, when $\phi = \delta/\sqrt{n}$, that is, when the alternative is *locally misspecified*. In particular, this implies that when the null is correct, the Breusch–Pagan test tends to reject the true null of absence of random effects if the error term is serially correlated, even in a local sense. A similar situation arises for the test for serial correlation of Baltagi and Li (1991) in the local presence of random effects. In order to remedy this problem, Bera and Yoon (1993) proposed the following modified RS statistic:

$$
\begin{aligned}
\mathrm{RS}^{*}_{\psi} \quad = \quad & \frac{1}{n}\big\{d_{\psi}(\widehat{\theta}) - J_{\psi\phi\cdot\gamma}(\widehat{\theta})J^{-1}_{\phi\cdot\gamma}(\widehat{\theta})d_{\phi}(\widehat{\theta})\big\}' \\
& \big\{J_{\psi\cdot\gamma}(\widehat{\theta}) - J_{\psi\phi\cdot\gamma}(\widehat{\theta})J^{-1}_{\phi\cdot\gamma}(\widehat{\theta})J_{\phi\psi\cdot\gamma}(\widehat{\theta})\big\}^{-1} \\
& \big\{d_{\psi}(\widehat{\theta}) - J_{\psi\phi\cdot\gamma}(\widehat{\theta})J^{-1}_{\phi\cdot\gamma}(\widehat{\theta})d_{\phi}(\widehat{\theta})\big\} \qquad (2)
\end{aligned}
$$

where $\widehat{\theta}$ is the MLE of θ under the joint null $\psi = \phi = 0$. This modified test statistic has an asymptotic *central* χ^2_1 distribution under the null hypothesis $H_0 : \psi = 0$ and when $\phi = \delta/\sqrt{n}$, that is, the modified test statistic has the correct size even when the underlying model is locally misspecified. Based on this principle, Bera, Sosa-Escudero, and Yoon (2001) derived modified tests for random effects (serial correlation), which are valid in the presence of local first-order serial correlation (random effects) assuming that the panel is balanced.

To derive tests for the unbalanced case, let $\theta = (\lambda, \sigma^2_{\mu}, \sigma^2_{\epsilon})'$ and $\widehat{\theta} = (0, 0, \widehat{\sigma}^2_{\epsilon})'$ be the MLE of θ under the joint null hypothesis $H_0 : \lambda = \sigma^2_{\mu} = 0$. The following formula by Hemmerle and Hartley (1973) will be useful to derive the score vector for the problem:

$$
d_{\theta_r} \equiv \frac{\partial L}{\partial \theta_r} = -\frac{1}{2}\,\mathrm{tr}\left(\Omega^{-1}\frac{\partial \Omega}{\partial \theta_r}\right) + \frac{1}{2}\left(u'\Omega^{-1}\frac{\partial \Omega}{\partial \theta_r}\Omega^{-1}u\right) \qquad (3)
$$

where θ_r denotes the rth element of θ, $r = 1, 2, 3$. Note that $\partial\Omega/\partial\sigma^2_{\mu} = \widetilde{H}$ with $\mathrm{tr}(\widetilde{H}) = m$. Similarly, $\partial\Omega/\partial\sigma^2_{\epsilon} = \widetilde{V}$, which under the restricted MLE is an $m \times m$ identity matrix with trace equal to m. Also $\partial\Omega/\partial\lambda = \sigma^2_{\epsilon}\widetilde{G}$, where $\widetilde{G}$ is a block diagonal matrix with blocks equal to G_i, with $G_i = \partial V_i/\partial\lambda$ given by

$$
G_i = \begin{bmatrix}
0 & 1 & 2\lambda & \cdots & (T_i - 1)\lambda^{T_i - 2} \\
1 & 0 & 1 & \cdots & (T_i - 2)\lambda^{T_i - 3} \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & 1 & 0 & 1 \\
(T_i - 1)\lambda^{T_i - 2} & \cdots & \cdots & 1 & 0
\end{bmatrix}
$$

Under the restricted MLE, G_i is a bidiagonal matrix as follows:

$$
G_i(\widehat{\theta}) = \begin{bmatrix}
0 & 1 & 0 & & \cdots & 0 \\
1 & 0 & 1 & & \cdots & 0 \\
\vdots & \vdots & \vdots & & \vdots & \vdots \\
0 & \cdots & & 1 & 0 & 1 \\
0 & \cdots & & \cdots & 1 & 0
\end{bmatrix}
$$

Hence, $\mathrm{tr}\left\{G_i(\widehat{\theta})\right\} = 0$. Replacing these results in (3) and evaluating the expression under the restricted MLE, we obtain

$$\begin{aligned} d_{\sigma^2_\mu}(\widehat{\theta}) &= -\frac{1}{2}\mathrm{tr}\left(\frac{1}{\widehat{\sigma}^2_\epsilon} I_m \widetilde{H}\right) + \frac{1}{2} e' \frac{1}{\widehat{\sigma}^2_\epsilon} I_m \widetilde{H} \frac{1}{\widehat{\sigma}^2_\epsilon} e \\ &= -\frac{1}{2}\frac{1}{\widehat{\sigma}^2_\epsilon} m + \frac{1}{2}\frac{1}{\widehat{\sigma}^4_\epsilon} e'\widetilde{H} e = -\frac{m}{\widehat{\sigma}^2_\epsilon}\ A \end{aligned}$$

where e is an $m \times 1$ vector with typical element $e_{it} = x'_{it}\widehat{\beta}$, and $\widehat{\beta}$ is the restricted MLE of β. Similarly, $\widehat{\sigma}^2_\epsilon = e'e/m$ is the restricted MLE of σ^2_ϵ, and $A \equiv 1 - e'\widetilde{H}e/(e'e)$. In a similar fashion,

$$\begin{aligned} d_\lambda(\widehat{\theta}) &= -\frac{1}{2}\mathrm{tr}\left\{\frac{1}{\widehat{\sigma}^2_\epsilon}\widehat{\sigma}^2_\epsilon \widetilde{G}(\widehat{\theta})\right\} + \frac{1}{2} 2 \frac{1}{\widehat{\sigma}^2_\epsilon} e'\widetilde{G}(\widehat{\theta}) e \\ &= \frac{1}{\widehat{\sigma}^2_\epsilon} e'\widetilde{G}(\widehat{\theta})\ e = m\ B \end{aligned}$$

where $B \equiv e'\widetilde{G}e/e'e$.

To derive the elements of the information matrix, we will use the following formula from Baltagi (2005, 59–60):

$$J_{r,s}(\theta) = E\left(-\frac{\partial^2 L}{\partial\theta_r \partial\theta_s}\right) = \frac{1}{2}\mathrm{tr}\left(\Omega^{-1}\frac{\partial\Omega}{\partial\theta_r}\Omega^{-1}\frac{\partial\Omega}{\partial\theta_s}\right)$$

Then

$$\begin{aligned} J_{\sigma^2_\epsilon,\sigma^2_\epsilon}(\widehat{\theta}) &= \frac{1}{2}\mathrm{tr}\left\{\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{V}(\widehat{\theta})\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{V}(\widehat{\theta})\right\} = \frac{1}{2}\mathrm{tr}\left(\frac{1}{\widehat{\sigma}^4_\epsilon} I_m\right) = \frac{m}{2\widehat{\sigma}^4_\epsilon} \\ J_{\hat{\sigma}^2_\mu,\hat{\sigma}^2_\mu}(\widehat{\theta}) &= \frac{1}{2}\mathrm{tr}\left(\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{H}\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{H}\right) = \frac{1}{2}\frac{1}{\widehat{\sigma}^4_\epsilon}\mathrm{tr}\left(\widetilde{H}\widetilde{H}\right) = \frac{\sum_{i=1}^N T_i^2}{2\widehat{\sigma}^4_\epsilon} \\ J_{\lambda,\lambda}(\widehat{\theta}) &= \frac{1}{2}\mathrm{tr}\left\{\frac{1}{\widehat{\sigma}^2_\epsilon}\widehat{\sigma}^2_\epsilon\widetilde{G}(\widehat{\theta})\frac{1}{\widehat{\sigma}^2_\epsilon}\widehat{\sigma}^2_\epsilon\widetilde{G}(\widehat{\theta})\right\} = \frac{1}{2}\mathrm{tr}\left\{\widetilde{G}(\widehat{\theta})\widetilde{G}(\widehat{\theta})\right\} \\ &= \frac{1}{2}\sum_{i=1}^N 2(T_i - 1) = m - N \\ J_{\hat{\sigma}^2_\epsilon,\hat{\sigma}^2_\mu}(\widehat{\theta}) &= \frac{1}{2}\mathrm{tr}\left\{\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{V}(\widehat{\theta})\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{V}(\widehat{\theta})\right\} = \frac{1}{2}\frac{1}{\widehat{\sigma}^4_\epsilon}\mathrm{tr}\left(\widetilde{H}\right) = \frac{m}{2\widehat{\sigma}^4_\epsilon} \\ J_{\hat{\sigma}^2_\epsilon,\lambda}(\widehat{\theta}) &= \frac{1}{2}\mathrm{tr}\left\{\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{V}(\widehat{\theta})\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{G}(\widehat{\theta})\right\} = \frac{1}{2}\frac{1}{\widehat{\sigma}^4_\epsilon}\mathrm{tr}\left\{\widetilde{G}(\widehat{\theta})\right\} = 0 \\ J_{\lambda,\hat{\sigma}^2_\mu}(\widehat{\theta}) &= \frac{1}{2}\mathrm{tr}\left\{\frac{1}{\widehat{\sigma}^2_\epsilon}\widehat{\sigma}^2_\epsilon\widetilde{G}(\widehat{\theta})\frac{1}{\widehat{\sigma}^2_\epsilon}\widetilde{H}\right\} = \frac{1}{2}\frac{1}{\widehat{\sigma}^2_\epsilon}\mathrm{tr}\left\{\widetilde{G}(\widehat{\theta})\widetilde{H}\right\} \\ &= \frac{1}{2}\frac{2}{\widehat{\sigma}^2_\epsilon}\left(\sum_{i=1}^N T_i - N\right) = \frac{1}{\widehat{\sigma}^2_\epsilon}(m - N) \end{aligned}$$

where we have used the facts that $\mathrm{tr}\left\{\widetilde{G}_i(\widehat{\theta})\widetilde{G}_i(\widehat{\theta})\right\} = \mathrm{tr}\left\{\widetilde{G}_i(\widehat{\theta})\widetilde{H}_i\right\} = 2(T_i - 1)$, and $\mathrm{tr}(\widetilde{H}_i\widetilde{H}_i) = T_i^2$.

Collecting all the elements, the information matrix evaluated at the restricted MLE under the joint null can be expressed as

$$J(\widehat{\theta}) = \frac{1}{2\widehat{\sigma}_\epsilon^4}\begin{bmatrix} m & m & 0 \\ m & a & 2\widehat{\sigma}_\epsilon^2(m-N) \\ 0 & 2\widehat{\sigma}_\epsilon^2(m-N) & 2\widehat{\sigma}_\epsilon^4(m-N) \end{bmatrix}$$

where $a \equiv \sum_{i=1}^N T_i^2$. For the balanced case $T_i = T$, we get exactly the same expression for $J(\widehat{\theta})$ as in Baltagi and Li (1991, 279). From the above expression of $J(\widehat{\theta})$, we can show that

$$\begin{aligned} J_{\mu\lambda\cdot\sigma_\epsilon^2} &= \frac{m-N}{\widehat{\sigma}_\epsilon^2} \\ J_{\mu\cdot\sigma_\epsilon^2} &= \frac{a-m}{2\widehat{\sigma}_\epsilon^4} \\ J_{\lambda\cdot\sigma_\epsilon^2} &= m-N \end{aligned}$$

Substituting these results in (2), we obtain the unbalanced version of the modified test for random effects as

$$\mathrm{RS}_\mu^* = \frac{m^2\left(A+2B\right)^2}{2\left(a-3m+2N\right)}$$

When $T_i = T$ (the balanced case), the above expression boils down to

$$\mathrm{RS}_\mu^* = \frac{NT\left(A+2B\right)^2}{2(T-1)\left\{1-(2/T)\right\}}$$

as in Bera, Sosa-Escudero, and Yoon (2001) for the balanced case.

Similarly, the modified test statistic for serial correlation is

$$\mathrm{RS}_\lambda^* = \left(B + \frac{m-N}{a-m}A\right)^2 \frac{(a-m)m^2}{(m-N)(a-3m+2N)}$$

and when $T_i = T$, we get

$$\mathrm{RS}_\lambda^* = \left(B + \frac{A}{T}\right)^2 \frac{NT^2}{(T-1)(1-2/T)}$$

which is the expression in Bera, Sosa-Escudero, and Yoon (2001) for the balanced case.

For computational purposes, it is interesting to see that

$$A = 1 - \frac{\sum_{i=1}^{N}\left(\sum_{t=1}^{T_i} e_{it}^2\right)^2}{\sum_{i=1}^{N}\sum_{t=1}^{T_i} e_{it}^2}$$

and

$$B = \frac{\sum_{i=1}^{N}\sum_{t=2}^{T_i} e_{i,t}e_{i,t-1}}{\sum_{i=1}^{N}\sum_{t=1}^{T_i} e_{it}^2}$$

and, therefore, there is no need to construct the $\widetilde{G}$ or $\widetilde{H}$ matrices; hence, the test statistics can be easily computed right after ordinary least-squares estimation without constructing any matrices.

The previous derivations allow us to obtain the unbalanced version of the test for serial correlation assuming no random effects:

$$\mathrm{RS}_\lambda = \frac{m^2B^2}{m-N}$$

which again reduces to $NT^2B^2/(T-1)$, originally derived by Baltagi and Li (1991) for balanced panels. Also, for completeness, the unbalanced version of the test for random effects assuming no serial correlation is given by

$$\mathrm{RS}_\mu = \frac{\frac{1}{2}m^2A^2}{a-m}$$

This test statistic is a particular case of the Baltagi–Li test for the two-way error-components model.

Suppose that we are interested in the joint null hypothesis of no random effects and no first-order serial correlation. Let $\mathrm{RS}_{\phi,\psi}$ be the RS test statistic for the joint null hypothesis $H_0 : \phi = \psi = 0$. Bera and Yoon (2001) show that the following identities hold:

$$\mathrm{RS}_{\phi\psi} = \mathrm{RS}^*_\psi + \mathrm{RS}_\phi = \mathrm{RS}^*_\phi + \mathrm{RS}_\psi$$

This simplifies computations, as illustrated in Sosa-Escudero and Bera (2001). Then, as a simple byproduct of the previous derivations, we can obtain a statistic for jointly testing serial correlation and random effects, as

$$\mathrm{RS}_{\lambda\mu} = m^2\left\{\frac{A^2+4AB+4B^2}{2(a-3m+2N)} + \frac{B^2}{m-N}\right\}$$

When $T_i = T$, $\mathrm{RS}_{\lambda\mu}$ simplifies to

$$\mathrm{RS}_{\lambda\mu} = \frac{NT^2}{2(T-1)(T-2)} \left(A^2 + 4AB + 2TB^2\right)$$

which is the original joint test statistic of Baltagi and Li (1991).

Finally, because $\sigma_\mu^2 \geq 0$, it is natural to consider one-sided versions of the tests for the null $H_0 : \sigma_\mu^2 = 0$. As in Bera, Sosa-Escudero, and Yoon (2001), appropriate test statistics can be readily constructed by taking the signed square roots of the original two-sided tests RS_μ and RS_μ^*. Denoting their one-sided versions, respectively, as RSO_μ and RSO_μ^*, we have

$$\mathrm{RSO}_\mu = -\sqrt{\frac{\frac{1}{2}\, m^2}{a-m}}\, A$$

and

$$\mathrm{RSO}_\mu^* = -\sqrt{\frac{m^2}{2\,(a-3m+2N)}}\, (A+2B)$$

3 Empirical illustration

As an illustration of these procedures, we provide an empirical exercise that is based on Gasparini, Marchionni, and Sosa-Escudero (2001). It consists of a simple linear panel-data model where the dependent variable is the Gini coefficient for 17 regions of Argentina. The vector of explanatory variables includes mean income and its square (`ie` and `ie2`); proportion of the population employed in the manufacturing industry (`indus`) and in public administration, health, or education (`adpubedsal`); unemployment rate (`desempleo`); activity rate (`tactiv`); public investment as percentage of GDP (`invipib`); degree of openness (`apertura`); social assistance (`pyas4`); proportion of population older than 64 (`e64`); proportion of population that completed high school (`supc`); and average family size (`tamfam`); for details see Gasparini, Marchionni, and Sosa-Escudero (2001). Models of this type have been used extensively in the literature exploring the links between inequality and development, usually to study the so-called “Kuznets hypothesis”, which postulates an inverted U-shaped relationship between these two variables (for example, see Anand and Kanbur [1993] and Gustafsson and Johansson [1999]).

Income-related variables, including the Gini coefficients, are constructed using Argentina’s *Permanent Household Survey* (Encuesta Permamente de Hogares), which surveys several socioeconomic variables at the household level for several regions of the country. Because of certain administrative deficiencies, the panel is largely unbalanced, so the number of available temporal observations ranges from 5 to 8 years in the period 1992–2000.

First, we `tsset` the data and then use `xtreg` to estimate the parameters of a one-way error-components model with region-specific random effects:

```
. use ginipanel5

. tsset naglo ano
       panel variable:  naglo (unbalanced)
        time variable:  ano, 1992 to 2000, but with a gap
                delta:  1 unit

. xtreg gini ie ie2 indus adpubedsal desempleo tactiv invipib apertura pyas4
> e64 supc tamfam, re i(naglo)

Random-effects GLS regression                   Number of obs      =       128
Group variable: naglo                           Number of groups   =        17

R-sq:  within  = 0.5096                         Obs per group: min =         6
       between = 0.6153                                        avg =       7.5
       overall = 0.5344                                        max =         8

Random effects u_i ~ Gaussian                   Wald chi2(12)      =    121.30
corr(u_i, X)       = 0 (assumed)                Prob > chi2        =    0.0000

------------------------------------------------------------------------------
        gini |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          ie |  -.0000995   .0001823    -0.55   0.585    -.0004568    .0002578
         ie2 |   1.64e-08   2.19e-07     0.08   0.940    -4.12e-07    4.45e-07
       indus |   -.041974   .0704982    -0.60   0.552    -.1801478    .0961999
  adpubedsal |  -.0635789   .0531777    -1.20   0.232    -.1678053    .0406475
   desempleo |  -.1177452   .0638999    -1.84   0.065    -.2429868    .0074963
      tactiv |   .0999584   .0737997     1.35   0.176    -.0446864    .2446031
     invipib |  -.3307239   .1912258    -1.73   0.084    -.7055197    .0440718
    apertura |   .4289793   .0768693     5.58   0.000     .2783183    .5796404
       pyas4 |   2.884162   1.626136     1.77   0.076    -.3030061    6.071331
         e64 |  -.1339182   .1505384    -0.89   0.374    -.4289681    .1611316
        supc |   .2417907   .0946423     2.55   0.011     .0562952    .4272861
      tamfam |   .0169905   .0174328     0.97   0.330    -.0171771    .0511581
       _cons |   .3084864   .1031351     2.99   0.003     .1063453    .5106274
-------------+----------------------------------------------------------------
     sigma_u |  .01370805
     sigma_e |  .01377936
         rho |  .49740589   (fraction of variance due to u_i)
------------------------------------------------------------------------------
```

Next the command `xttest1` with the `unadjusted` option presents the following output:

```
. xttest1, unadjusted
Tests for the error component model:
        gini[naglo,t] = Xb + u[naglo] + v[naglo,t]
           v[naglo,t] = lambda v[naglo,(t-1)] + e[naglo,t]
        Estimated results:
                         |       Var     sd = sqrt(Var)
                ---------+-----------------------------
                    gini |   .0006167       .0248335
                       e |   .0001899      .01377936
                       u |   .0001879      .01370805
        Tests:
           Random Effects, Two Sided:
           LM(Var(u)=0)         =   13.50 Pr>chi2(1) =  0.0002
           ALM(Var(u)=0)        =    6.03 Pr>chi2(1) =  0.0141
           Random Effects, One Sided:
           LM(Var(u)=0)         =    3.67 Pr>N(0,1)  =  0.0001
           ALM(Var(u)=0)        =    2.46 Pr>N(0,1)  =  0.0070
           Serial Correlation:
           LM(lambda=0)         =    9.32 Pr>chi2(1) =  0.0023
           ALM(lambda=0)        =    1.86 Pr>chi2(1) =  0.1732
           Joint Test:
           LM(Var(u)=0,lambda=0) =   15.35 Pr>chi2(2) =  0.0005
```

The unadjusted version of the tests for random effects (`LM(Var(u)=0)`) and serial correlation (`LM(lambda=0)`), and the test for the joint null (`LM(Var(u)=0,lambda=0)`) suggest rejecting their nulls at the 5% significance level. Care must be taken in deriving conclusions about the direction of the misspecification because, in light of the results in Bera, Sosa-Escudero, and Yoon (2001), rejections may arise because of the presence of random effects, serial correlation, or both. To explore the possible nature of the misspecification, we restore the modified versions of the test. The adjusted version of the test for serial correlation `ALM(lambda=0)` now fails to reject the null hypothesis while the adjusted version of the test for random effects `ALM(Var(u)=0)` still does. This suggests that the possible misspecification is likely due to the presence of random effects rather than the serial correlation. Consequently, and to stress the main usefulness of these procedures, in this example the presence of the random effects seems to confound the unadjusted test for serial correlation, making it spuriously reject its null.

4 Acknowledgments

We thank an anonymous referee for useful suggestions that helped improve the routine considerably, and we thank Javier Alejo for excellent research assistance.

5 References

Anand, S., and R. Kanbur. 1993. Inequality and development: A critique. *Journal of Development Economics* 41: 19–43.

Baltagi, B., and Q. Li. 1990. A Lagrange multiplier test for the error components model with incomplete panels. *Econometric Reviews* 9: 103–107.

———. 1991. A joint test for serial correlation and random individual effects. *Statistics and Probability Letters* 11: 277–280.

Baltagi, B. H. 2005. *Econometric Analysis of Panel Data*. 3rd ed. New York: Wiley.

Bera, A., W. Sosa-Escudero, and M. Yoon. 2001. Tests for the error component model in the presence of local misspecification. *Journal of Econometrics* 101: 1–23.

Bera, A., and M. Yoon. 1993. Specification testing with locally misspecified alternatives. *Econometric Theory* 9: 649–658.

———. 2001. Adjustments of Rao's score test for distributional and local parametric misspecifications. Mimeo: University of Illinois at Urbana–Champaign.

Breusch, T., and A. Pagan. 1980. The Lagrange multiplier test and its applications to model specification in econometrics. *Review of Economic Studies* 47: 239–253.

Gasparini, L., M. Marchionni, and W. Sosa-Escudero. 2001. *Distribucion del Ingreso en la Argentina: Perspectivas y Efectos sobre el Bienestar*. Cordoba: Fundacion Arcor-Triunfar.

Gustafsson, B., and M. Johansson. 1999. In search of smoking guns: what makes income inequality vary over time in different countries? *American Sociological Review* 64: 585–605.

Hemmerle, W. J., and H. Hartley. 1973. Computing maximum likelihood estimates for the mixed AOV model using the W transformation. *Technometrics* 15: 819–831.

Sosa-Escudero, W., and A. Bera. 2001. sg164: Specification tests for linear panel data models. *Stata Technical Bulletin* 61: 18–21. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 307–311. College Station, TX: Stata Press.

About the authors

Walter Sosa-Escudero is an associate professor and the chairman of the Department of Economics, Universidad de San Andrés, Buenos Aires, Argentina.

Anil K. Bera is a professor in the Department of Economics, University of Illinois at Urbana–Champaign.

The Stata Journal (2008)
8, Number 1, pp. 79–104

fuzzy: A program for performing qualitative comparative analyses (QCA) in Stata

Kyle C. Longest
Department of Sociology
University of North Carolina at Chapel Hill
Chapel Hill, NC
klongest@email.unc.edu

Stephen Vaisey
Department of Sociology
University of North Carolina at Chapel Hill
Chapel Hill, NC

Abstract. Qualitative comparative analysis (QCA) is an increasingly popular analytic strategy, with applications to numerous empirical fields. This article briefly discusses the substantive motivation and technical details of QCA, as well as fuzzy-set QCA, followed by an in-depth discussion of how the new program `fuzzy` performs these techniques in Stata. An empirical example is presented that demonstrates the full suite of tools contained within `fuzzy`, including creating configurations, performing a series of statistical tests of the configurations, and reducing the identified configurations.

Keywords: st0140, fuzzy, cmvom, cnfgen, coincid, coverage, fzplot, mavmb, reduce, setgen, suffnec, truthtab, yavyb, yvn, yvo, yvv, yvy, qualitative comparative analysis, QCA, fuzzy sets, Boolean logic, Boolean data, postestimation command

1 Introduction

In recent years, researchers in a number of fields have begun using qualitative comparative analysis (QCA) or its fuzzy-set variant to analyze multivariate data (Ragin 1987, 2000). For examples, see Kalleberg and Vaisey (2005), Mahoney (2003), Roscigno and Hodson (2004), and Vaisey (2007). Rather than estimate the net effects of single variables, QCA employs Boolean logic to examine the relationship between an outcome and all binary combinations of multiple predictors. The advantage of QCA is that it allows the researcher to find distinct combinations of causal variables that, in turn, suggest different theoretical pathways to given outcomes. Although early versions of QCA (Ragin 1987) were criticized on the grounds that they were deterministic and that they bore little relation to commonly used variants of the general linear model, recent developments are now integrating QCA-based strategies with more formal statistical distributions and procedures (Ragin 2006; Smithson and Verkuilen 2006).

2 Why QCA?

Because QCA is still a relatively new strategy, we illustrate its utility by describing one substantive topic that might benefit from its application. The stress process model has inspired a wealth of research in the mental health field that has shown the deleterious consequences of an accumulation of stressors on increased negative health outcomes. Yet personal resources, active coping strategies, and social support can buffer these stressors, thereby reducing their harmful impact (for reviews, see Lin and Ensel [1989] and Thoits [1995]). The strength of this research notwithstanding, Thoits (1995) argued that there may be important pathways to negative health outcomes that have remained unseen because investigators typically apply linear estimation models to the stress process.

> Just as there may be different combinations of conditions across countries which lead to political revolution, there may be different configurations of factors across individuals which lead to heart attack or to the onset of major depression.... The assumption of one process for becoming depressed or ill and the concomitant use of the general linear model to test it requires us to reject or ignore other possible processes which are less frequently observed and do not manage to achieve statistical significance. (Thoits 1995, 68)

In this observation, Thoits was advocating the use of QCA in tests of the stress process model. That is, high levels of conjunction among stressors and buffering agents could define multiple routes to the same level of distress.

In addition to finding multiple paths to an outcome, QCA is especially appropriate for testing models, like that underlying stress theory, that involve a multitude of "interacting" factors. More precisely, QCA effectively addresses theoretical hypotheses that predict multiple variables will operate in tandem at specific levels (e.g., high stress events, low coping resources, and low social support) to produce particular outcomes (e.g., high distress). For example, stress theory predicts that numerous negative events should interact with chronic strain to produce high levels of distress, but the use of several active coping strategies should moderate this interaction, reducing the likelihood of elevated distress. Furthermore, high levels of mastery may enhance the buffering influence of active coping efforts, and high social support might further increase mastery's moderating influence on the interaction of active coping with stressors. This hypothesis modeled in a regression framework would involve a five-way interaction term, which would have to be interpreted along with all of its component interactions, an obviously difficult and inefficient task. QCA, on the other hand, explicitly and straightforwardly tests each possible combination of factors at specific levels with a given outcome. The results then can be interpreted more clearly, making QCA a potentially more effective analytic strategy for complex theoretical processes such as those posed by stress theory.

The stress process model is just one specific case that would benefit from the application of QCA.[1] QCA's utility as an analytic strategy stands to augment research in numerous fields. Greckhamer and his colleagues (2007) have recently argued for QCA's

[1] Longest and Thoits (2007) have tested this model with QCA and found several intriguing results.

application in strategic management research because of its ability to analyze complex relationships between different industry- and corporate-level mechanisms in predicting business success. Similarly, researchers in epidemiology would benefit from QCA's capacity to capture holistically individuals' experience of risk and protective characteristics (Schuit et al. 2002). Finally, Shanahan et al. (2007) have demonstrated how QCA can be employed to examine the complex relationship between environmental and genetic factors leading to adult success.

We believe that part of the reason QCA has not been utilized more widely in empirical research across fields is because no software presently exists that easily combines QCA with conventional data management and statistical tests. The primary stand-alone program, fuzzy-set QCA, is able to compute logical truth tables for both fuzzy-set and dichotomous-set data, but it has no built-in capacity for probabilistically testing logical necessity and sufficiency, as advocated by Ragin (2006). There is also a program for the R statistics language that can perform logical reductions of Boolean data, but it lacks the ability to perform useful statistical tests. Further, there is no program in Stata that can perform the necessary analyses or reductions in fuzzy-set analyses.

In this paper, we present and outline `fuzzy`, a new Stata command we have developed that is capable of creating, testing, and performing logical reductions on both fuzzy and dichotomous (crisp) set-theoretic data. We will first outline some of the background of the technique and then provide a detailed explanation of the functionality of the command.

3 Statistical background

QCA evaluates the relationship between an outcome and all possible Boolean combinations of predictors. For example, given an outcome set `Y` and predictor sets `A` and `B`, QCA examines which combinations of `A` and `B` (i.e, $\mathtt{A} \cdot \mathtt{B}$, $\mathtt{A} \cdot \mathtt{b}$, $\mathtt{a} \cdot \mathtt{B}$, $\mathtt{a} \cdot \mathtt{b}$) are most likely to produce `Y`. In a QCA framework, the term "set" is used rather than "variable" to emphasize the idea that each variable has been transformed to represent the individual's level of membership in a given condition, for example, his or her level of membership in "heavy alcohol users". The combination of individual "sets"—for example, high depression and low self esteem—is then referred to as a "configuration". Sets are labeled, according to convention, with capital and lowercase letters. In the crisp-set case (i.e., all sets are dichotomous indicators) capital letters signify 1 (i.e., fully in `A`) and lowercase letters signify 0 (i.e., fully out of `A`). When using fuzzy sets, where set membership can take on any value between 0 and 1, uppercase simply means the level of set membership (e.g., value of `A`) and lowercase means 1 minus the set membership (e.g., $1-\mathtt{A}$). The operator "·" stands for the Boolean "and".

In the crisp-set case, the relationship between the predictors and the outcome can be evaluated using conditional probabilities—e.g., $\Pr(\mathtt{Y} \,|\, \mathtt{A} \cdot \mathtt{B})$. In set-theoretic terms, higher conditional probabilities indicate greater empirical correspondence with the statement "$\mathtt{A} \cdot \mathtt{B}$ is a subset of `Y`", or, in logical terms, "if $\mathtt{A} \cdot \mathtt{B}$, then `Y`". Evaluating this logical or subset relationship becomes more problematic in the fuzzy-set case, however, because unlike crisp sets, fuzzy sets can range between 0 (completely exclusive) and 1 (completely inclusive). Thus individuals can be more or less a member of a particular set (e.g., 0.33 would indicate something like "more out than in, but still somewhat in" the set, whereas 0.7 would signify something like "more in than out, but not entirely in" the set). Combining fuzzy sets into configurations is usually done using the minimum operator, so $\mathtt{A} \cdot \mathtt{B} = \min(\mathtt{A},\mathtt{B})$, or $\mathtt{a} \cdot \mathtt{B} = \min\{(1-\mathtt{A})\ ,\ \mathtt{B}\}$.

The advantage of fuzzy sets over crisp sets is that we can transform our original measures without losing the variation associated with dichotomizing categorical or continuous measures. Using the minimum operation to calculate configuration membership more precisely defines the degree to which an individual experiences the combination of factors (i.e., individuals do not have to be completely in or completely out of every possible configuration). But this added nuance prohibits the use of a simple conditional probability to evaluate the degree of subsetness of each configuration in a given outcome. The most common approach to evaluating this relationship when using fuzzy sets is the inclusion ratio:

$$I_{XY} = \Sigma \min(x_i, y_i)/\Sigma x_i \tag{1}$$

where X signifies the predictor configuration (e.g., $\mathtt{A} \cdot \mathtt{B}$), Y signifies the outcome set, x_i stands for each case's membership in the configuration X, and y_i stands for each case's membership in the set Y (see Ragin [2000, 2006] and Smithson and Verkuilen [2006] for discussions of other methods). As with conditional probabilities, the closer the value of I_{XY} to unity, the greater the consistency of the data with the assertion that X is a subset of Y or, in logical terms, with the statement "if X, then Y". (For this reason, this value is often referred to below as the "consistency score".)

Also there are a number of methods for deciding whether each configuration of predictors (X) should "count" as a (probabilistically) sufficient condition for Y. One way, advocated by Ragin (2000, 2006), is to determine a numeric benchmark (say, 0.8) and code all configurations, for which $I_{XY} > 0.8$, as sufficient. We take no position on any particular method here, and the `fuzzy` program allows multiple types of tests of probabilistic sufficiency. This flexibility is beneficial because the methods are still being refined and because several types of tests can support the robustness of claims of sufficiency.

The ultimate classification of some configurations as sufficient, however, is an important part of QCA. Once the sufficient configurations have been determined, one can use Boolean algebra to reduce the configurations into a more parsimonious solution. For example, if both $\mathtt{a} \cdot \mathtt{B} \cdot \mathtt{C}$ and $\mathtt{A} \cdot \mathtt{B} \cdot \mathtt{C}$ were coded as sufficient, this would reduce to $\mathtt{B} \cdot \mathtt{C}$. This type of logical reduction can be extended to more complicated solution sets of configurations through the use of the Quine–McCluskey algorithm (see Ragin [1987]). In this way, one can obtain a logical description of the conditions sufficient to produce (probabilistically speaking) a particular outcome.

Finally, each final solution is evaluated with respect to its coverage of the outcome. Coverage is simply an indicator of how much of Y is covered by X; it is computed as follows:

$$C_{XY} = \Sigma \text{min}(x_i, y_i)/\Sigma y_i$$

Although computationally similar, `coverage` addresses a different aspect than does the consistency score. Primarily, it helps to answer how much of the outcome is understood by taking into account the final solution set. For example, the set of skydiving parachute failures would be a near-perfect subset (i.e., high consistency) of the set of deaths, but this combination might not be very helpful (i.e., low coverage) in determining the most common or meaningful pathways to mortality in a given population.

The `fuzzy` program allows the user to create configurations from single sets coded as dichotomous or as fuzzy, to evaluate the sufficiency of these configurations statistically by using a variety of benchmarks, and to reduce the configurations determined sufficient to their common logical elements. The remainder of this paper describes the functionality of the program.

4 Creating, testing, and reducing sets

4.1 Syntax

```
fuzzy varlist [if] [weight] [, label(capital_letter_list) keepsets drop
   settest(testlist) group(varname) conval(#) sigonly slevel(#)
   greater(col1|col2) common cluster(varname) necessity matx(matlist)
   standardized altdisplay reduce remainders(#) dnc(configlist)
   truthtab(filename) keepconfigs]
```

where *testlist* is `yvn`, `yvo`, `yvv`, `yavyb`, `cmvom`, or `mavmb`; and *matlist* is `suffnec` or `coincid`.

`fweight`s, `iweight`s, and `pweight`s are allowed with `fuzzy`; see [U] **11.1.6 weight**.

The weights are applied with the `ratio` command, which is used to calculate the consistency score for each configuration, but are not used in any other parts of the routine (such as the creation of the `bestfit` variable).

4.2 Description

`fuzzy` is a suite of tools to perform QCA, as previously described. Without any options specified, it will create the `bestfit` variable that displays the number of cases that score greater than 0.500 on each configuration (which each case can only do for one configuration). The varlist should be treated similarly to other Stata commands, such that the first variable listed is the outcome variable followed by the individual set variables. All variables entered must range from 0 to 1 (or be dichotomous coded 0/1).[2]

To create the configurations, `fuzzy` requires that all of the variables in the varlist be named with single, capital letters. If the user enters variables named as such, then `fuzzy`, without any options, will simply create the `bestfit` variable. But if any of the variables in the varlist are not named with single, capital letters, `fuzzy` will generate a copy variable of each variable in the varlist, naming them with single, capital letters, which are automatically deleted when the program is terminated. The user can control what letters are used to designate these copies by invoking the `label()` option, and when done the new variables will remain in the dataset, unless the `drop` option also is specified. Additionally, specifying `keepconfigs` will prevent the deletion of the generated configurations.

The primary advantages of `fuzzy`, compared to other QCA programs, lie in its options, most notably `settest()` and `reduce`. The `settest()` option defines the tests (each to be fully explained later) to be performed on the configurations' means or consistency scores. These tests help determine each configuration's degree of inclusion with the given outcome. `sigonly`, `slevel()`, `conval()`, and `greater()` all alter the configurations that are displayed by the given test and determine which configurations enter the reduction (if `reduce` is specified). At least one of these options (hereafter referred to as `settest()` options) must be specified for `reduce` to work, otherwise it would try to logically reduce every possible configuration (i.e., it would reduce to a logical contradiction). `common` can be used when multiple tests are run in a single call and will display (and send to `reduce`) only the configurations that pass all the tests designated.

`reduce` uses the Quine–McCluskey algorithm (Ragin 1987) to logically reduce the configurations specified by `settest()` and its options. Further, it displays the coverage statistics for each of the reduced configurations and for the total final solution set. If nonsingle, capital letter variables are entered in *varlist*, `reduce` will display its output using the original variable names.

Finally, the option `matx()` can be used to produce matrices of descriptive statistics. `matx(coincid)` will display the coincidence matrix for the varlist, and invoking the `standardized` option will produce the standardized scores. Similarly, `matx(suffnec)` produces a matrix showing the sufficiency and necessity scores for each variable entered into the varlist, and using the `altdisplay` option will produce a "flipped" matrix, placing the values where they are generally visualized graphically (i.e., sufficiency in the upper left and necessity in the lower right).

4.3 Options

`label(`*capital_letter_list*`)` allows the user to specify what sets should be named when used in creating and displaying the resulting configurations. If all variables in *varlist* are already named as single, capital letters, then there is no reason to specify this option (unless the user would like copies of the variables with new designations).

[2] See `setgen` for a useful way to create such variables.

Further, this option is not required, but if any of the variables in *varlist* are not named with a single, capital letter and this option is not specified, then the generated copies of those variables will be named with a random single, capital letter, which will be used in displaying the configurations but dropped when `fuzzy` is terminated.

`keepsets` prevents the generic single, capital letter version of each variable from being deleted. This is only applicable when the `label()` option has not been specified and the original variables are not already named as single, capital letters.

`drop` automatically deletes any single, capital letter copies of variables entered in *varlist* when the program is terminated (only applicable if `label()` is also specified).

`settest(`*testlist*`)` defines which tests will be run and displayed:

`yvn` performs and displays the results (configuration; y consistency; n consistency; F distribution; p-value; number of best-fitting observations) of the test between each configuration's y consistency (inclusion in y) versus its n consistency (inclusion in not-y, or $1-y$). The test is performed using a Wald test (which uses an F distribution) comparing the consistency scores [i.e., equation (1)] derived using the `ratio` command; a similar test procedure is used for all the tests available in `settest()`. Thus a significant p-value means that the y consistency and the n consistency of a particular configuration are statistically different.

`yvo` performs and displays the results (configuration; y consistency; all other sets' y consistency; F distribution; p-value; number of best-fitting observations) of a test between each set's y consistency and the y consistency of all other configurations (excluding only the configuration in question). This "other" y consistency is calculated by taking the maximum value of every other configuration, excluding the configuration to be tested, and computing its inclusion in the outcome set. This test is generally applicable only if the configurations comprise binary crisp sets.

`yvv` performs a test of each configuration's y consistency versus a given numerical value (default is 0.800) and displays the configuration, y consistency, test value, F distribution, p-value, and number of best-fitting observations.

`yavyb` tests each configuration's y consistency for each of the subgroups defined in `group()`. The y consistency for each configuration is calculated separately for each subgroup, and then they are tested against each other. Hence this test will indicate, for each configuration, whether the y consistency of the first group defined in `group()` is significantly different from the y consistency of the second group. It displays the configuration; the first group's y consistency; the second group's y consistency; the F distribution; and the p-value. `group()` must be specified with `settest(yavyb)`.

`cmvom` operates similarly to `yvo`, but rather than using the configuration's consistency, it calculates and displays each configuration's weighted mean. The configuration's mean on the outcome is weighted by the membership in that configuration. This value is then tested against the mean as weighted by the maximum value of the other configurations. It displays each configuration's mean;

the "other" cumulative configuration's mean; the *t* value; the *p*-value; and the number of best-fitting observations.

`mavmb` operates similarly to `yavyb`, but it conducts the test using the configuration's weighted mean by each group specified by `group()`. The adjusted mean is calculated using the membership in that configuration for each subgroup. It then displays the weighted mean for the first group, weighted mean for the second group, *t* value, and *p*-value. `group()` must be specified with `settest(mavmb)`.

`group(`*varname*`)` defines the group variable used when invoking the `yavyb` or `mavmb` test. *varname* must be a 2-category variable.

`conval(`#`)` changes the value against which to test each configuration's *y* consistency if `settest(yvv)` is specified. The value can be any number between 0 and 1. The default is `conval(.800)`.

`sigonly` restricts the display of any tests specified in `settest()` to only those with significant *p*-values.

`slevel(`#`)` changes the significance level to be used in determining `sigonly`. The default is `slevel(.05)`.

`greater(col1|col2)` restricts the display of any test specified by `settest()` to only those in which the value in the designated column (of the output) is greater than the other column. The first column is always the consistency (or mean) for each configuration, while the second column is what that consistency is being tested against.

Note: `sigonly` and `greater(col1|col2)` can be used in conjunction. Thus, for example, if `sigonly` and `greater(col1)` were specified, any test designated in `settest()` would display only the results for those configurations that had a significant *p*-value on the test and that had a first column value greater than the second column.

`common` displays only the configurations that pass all of the tests and conditions specified by `settest()`. For example, if `settest(yvn yvv) sigonly common` was entered, `common` would display the configurations that had a significantly greater *y* consistency than *n* consistency and a *y* consistency significantly greater than 0.800.

Note: At least one of the `settest()` options must be used if `reduce` is invoked, in order to specify which configurations should be entered into the reduction. Further, if one (or more) of these restrictions is invoked along with `reduce`, only those configurations that are displayed will be entered into the reduction. Finally, if multiple tests are specified in `settest()`, without `common`, the last displayed configurations (regardless of the order in which the tests are specified) will be entered into the reduction (the order of the display will follow the order listed in the syntax above). Specifying `common` will send those common configurations to `reduce`.

`cluster(`*varname*`)` allows the standard errors produced by the `ratio` command when calculating consistencies to adjust for intragroup correlation.

`necessity` produces a table of each configuration's necessity value (similar to consistency except the denominator is the sum of the outcome instead of the sum of the configuration).

`matx(`*matlist*`)` defines which matrix to produce:

`suffnec` produces a sufficiency and necessity matrix for all the variables entered in *varlist*. Thus this option can be used to help determine the relationship between individual sets with each other and with the outcome. Sufficiency, in this case, is equivalent to computing individual set consistency scores.

`coincid` produces a coincidence matrix for all the variables entered in *varlist*. Again this is useful to help understand the relationship between the independent variables by using methods in line with fuzzy-set theory. Coincidence measures the amount of overlap or coincidence between the two sets or configurations (see Ragin [2006] for full details).

`standardized` alters the coincidence scores by taking into account the size of the sets. Coincidence is, in part, determined by the size of the sets because the larger they are the more room there is to overlap. This standardizing procedure divides the coincidence score by the total membership of the smaller set. The values in this matrix thus indicate the proportion of the total possible overlap between the sets.

`altdisplay` flips the `suffnec` matrix so that the sufficiency scores are in the upper left and the necessity scores are in the lower right (which is how these values are generally portrayed graphically).

`reduce` uses the elements passed by `settest()` to implement the Quine–McCluskey algorithm to produce a reduced final solution set and its accompanying coverage statistics. For example, if the input

```
. fuzzy Y A B C, settest(yvn) sigonly
```

displayed the configurations `ABc` and `ABC`, then `reduce` would produce `AB` and display this reduced configuration's coverage statistics. Again, when invoking `reduce`, it is also necessary to give some criteria to prevent the total possible configuration set from being entered into the reduction, which would reduce to a logical contradiction. Thus, when `reduce` is specified, `settest()` and `sigonly` or `greater()` must be specified.

`remainders(`#`)` runs the reduction twice: once with those configurations specified as remainders included in the reduction as "do-not-care" configurations, and once with them excluded. The # determines that any configuration with fewer or equal to # best-fitting observations are treated as remainders. This is highly advisable when the configuration contains many sets or when the sample size is small.

`dnc(`*configlist*`)` specifies configurations as "do-not-care" configurations. The entered configurations are treated as do-not-care configurations regardless of whether they pass any specified tests. These configurations are used in the first step of the reduction but not in the second step (see Ragin [2000] for a full explanation of do-not-care configurations).

`truthtab(`*filename*`)` outsheets (and replaces) a file containing the resulting truth table. If no options are specified, it will outsheet the entire truth table; otherwise, it operates similarly to `reduce` in how it defines which configurations are included. The *filename* should end in the desired output type (`.dta`, `.csv`, etc.).

`keepconfigs` prevents the generated configuration variables from being deleted when `fuzzy` is terminated.

4.4 Saved results

Macros

`r(y)`	outcome variable
`r(sets)`	total possible configuration set
`r(start)`	independent variable sets
`r(colsig)`	configurations passing the last displayed results (if `settest()` is specified)
`r(comm)`	common configurations (if `common` is specified)
`r(reducsols)`	final reduced configurations (if `reduce` is specified)

Matrices

`r(coincid)`	coincidence matrix (if `matx(coincid)` is specified)
`r(suffnec)`	sufficiency and necessity matrix (if `matx(suffnec)` is specified)

4.5 Postestimation commands and stand-alones

In addition to the base command and its options, the `fuzzy` program also includes many of the same options as stand-alone programs, to be used in testing specific configurations or as "postestimation" commands. Specifically, all the possible tests specified in `settest()`, the matrices in `matx()`, `truthtab()`, `coverage`, and `reduce` can all be used as stand-alone or post`fuzzy` commands. In both cases, they accept all the options that would pertain to them if specified in `fuzzy`.

When run as stand-alone programs, the user can enter specific configurations that do not have to exist in the dataset, but if they do not, their set components (named with single, capital letters) must. For example, if the user was specifically interested in the consistencies of two particular configurations ($\mathtt{A} \cdot \mathtt{B} \cdot \mathtt{C}$ and $\mathtt{a} \cdot \mathtt{B} \cdot \mathtt{C}$), the command

```
. yvn Y ABC abc
```

could be run, which would produce the typical `yvn` output, but only for `ABC` and `abc`. Variables representing the configurations `ABC` and `abc` do not need to be present in the dataset for this command to work, but the individual set variables `Y`, `A`, `B`, and `C` do. The user could also specify the `sigonly`, the `greater()`, or both options, just as would be done if `yvn` was used in the `fuzzy` command.

These commands can also be used as postestimation commands. So, for example, if the user ran a simple `fuzzy` command with no options, `yvn` could be entered subsequently, which would produce the same output as if `yvn` had been included in `settest()` with the original `fuzzy` command. When these postestimation commands are used in this manner, they use the full varlist from the last run `fuzzy` command unless `last` is specified as an option, which will then use the last "displayed" configurations resulting from any invoked `settest()` options. If the user wishes to run the test on the final reduced solution set (assuming `reduce` had been specified in the previous command), `usereduction` should be entered as an option.

Additionally, `reduce`, `coverage`, and `truthtab` can be used as postestimation commands and will automatically perform their operations on the last displayed configurations (`last` does not need to be specified because these programs can be used only if `settest()` was used in the last run `fuzzy` or stand-alone command). They also can be used as stand-alones to manually run their operations. For example, a user could perform a reduction on a specified set of configurations. When used in this manner, full configurations must be entered (i.e., it will not reduce `ABc aB` because the latter does not contain all of the constituent sets; but it would reduce `ABC aBc aBC`, which is logically equivalent). Finally, `reduce` as a stand-alone accepts the `dncare()` option for specifying configurations as do-not-care configurations, and `truthtab` accepts its `outsheet()` option.

Next there are a few extensions of `fuzzy` that only work as stand-alone programs.

`cnfgen` *newconfigurationlist* generates specific configuration variables. The individual sets must be included in the dataset, named with single, capital letters, and range from 0 to 1. For example, the user could enter

```
. cnfgen aBc ABC abc
```

to produce variables representing these three configurations.

`yvy` *outcomevar configuration1 configuration2* tests the *y* consistency of the first configuration entered against the *y* consistency of the second configuration entered. The configurations do not need to exist in the dataset, but their individual single, capital letter versions must. It is also possible to test individual sets against each other, but they must exist in the dataset (and be individual letters) and range from 0 to 1.

`fzplot` *outcomevar configuration* [`, mlabel(`*varname*`)`] will produce a sufficiency plot of the specified configuration and the outcome variable. The configuration does not need to exist in the dataset, but its individual single, capital letter versions must exist.

`mlabel(`*varname*`)` is an available option and is used just as it would be with `scatter`; see [G] **graph twoway scatter**.

`coverage` [*outcomevar configurations*] will produce the coverage statistics for the given configurations. The configurations do not need to exist in the dataset, but their individual single, capital letter versions must exist.

`setgen` *newvar* `=` *fcn*`(`*arguments*`)` [`,` *option*] works like `egen` but specifically for creating fuzzy sets that range from 0 to 1. The function *fcn* is one of the following:

`stdrank(`*varname*`)` rank orders the variable and then standardizes this ranking to range from 0 to 1. The equation for this standardization is

$$\frac{rankedvar - \min(rankedvar)}{\max(rankedvar) - \min(rankedvar)}$$

`direct(`*varname*`), anchors(`*numlist*`)` performs the “direct” transformation, outlined in Ragin (2008), which uses set values (the `anchors()` option) to calibrate the membership scores as levels of deviation from the anchors, in terms of log odds. The anchors should be entered in ascending order (i.e., mimimum threshold, crossover point, maximum threshold).

`ndrect(`*varname*`), grpdvar(`*varname*`)` performs the “indirect” transformation, outlined in Ragin (2008), which uses a regression based on the qualitative cutpoints in the `grpdvar()` option to determine membership scores. The `grpdvar()` option should be a categorization of the original variables into groups according to their levels of membership (e.g., more in than out = 0.33). This new, qualitatively grouped variable is regressed on the original in order to reshape the original variable into a set, ranging from 0 to 1, that is based on the knowledge-based grouping.

`crisp(`*varname*`)` [`, cutpt(`#`)`] will produce a “crispy” copy of the variable (i.e., a binary 0/1 variable). The default is to split the new variable at the original variable’s median, such that the median value and below will be coded as 0, otherwise 1 (unless the median and the maximum value are the same in the original variable, in which case the median will be coded 1). If the median is not an appropriate delineation, the `cutpt()` option can be used to specify the value at which to split the variable, such that all those values equal to or less than the cutpoint will be coded 0 and the rest 1.

5 Remarks and examples

To illustrate the capabilities and functionality of the `fuzzy` command, we present analyses using the National Study of Youth and Religion (Smith and Denton 2003). The study is a nationally representative survey of 3,390 teenagers and their parents. For simplicity, 5th and 6th grade adolescents are dropped and listwise deletion is employed for all variables in analyses, resulting in a sample size of $n = 3{,}112$.

For the purposes of these examples, we will be using the teen’s reported grades (`grds`) during the past academic semester as the outcome variable (higher scores indicate better grades). The independent variables will include their number of work hours (`workhrs`), their number of friends reported to use drugs or drink alcohol (`peerus`), an index of parental monitoring (`pmonit`), and their reported alcohol usage (`alcuse`). All of the variables have been transformed into sets using the `stdrank()` function in `setgen`.

This transformation is one of many that could be used to convert variables into sets. To illustrate its general properties, we have presented the frequency distributions from the original grade variable and its new set version.

```
. use nsyr_example_data
. tabulate grds, nol
       grds |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |         13        0.42        0.42
          2 |         11        0.35        0.77
          3 |         19        0.61        1.38
          4 |         88        2.83        4.21
          5 |        387       12.44       16.65
          6 |        503       16.16       32.81
          7 |        402       12.92       45.73
          8 |      1,088       34.96       80.69
          9 |        325       10.44       91.13
         10 |        276        8.87      100.00
------------+-----------------------------------
      Total |      3,112      100.00
. setgen G = stdrank(grds)
. tabulate G
    rank of |
     (grds) |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 |         13        0.42        0.42
   .0040438 |         11        0.35        0.77
   .0090986 |         19        0.61        1.38
   .0271272 |         88        2.83        4.21
   .1071609 |        387       12.44       16.65
   .2571188 |        503       16.16       32.81
    .409604 |        402       12.92       45.73
   .6606571 |      1,088       34.96       80.69
   .8987363 |        325       10.44       91.13
          1 |        276        8.87      100.00
------------+-----------------------------------
      Total |      3,112      100.00
```

The distribution of cases has not changed, but the scale has been "fuzzified" to range between 0 and 1, with the values now representing the level of membership in the set "good grades". The similarity of distributions is not required, and in fact there are situations that may call for more user-knowledge-based coding (for a discussion of this type of transformation, see Ragin [2000]). In the current example, we have chosen to use the standardized rank transformation because it is a relatively straightforward conversion.

The following is the distribution of each variable and its corresponding set:

Variable	Original range	Original mean	Set mean
Grades	1–10	7.26	0.52
Work hours	0–40	3.31	0.19
Peer substance use	0–5	0.70	0.25
Parent monitoring	0–4	2.62	0.51
Alcohol use	0–5.5	0.60	0.27

5.1 Configuration testing

The first step in the fuzzy analysis might be to see which configurations contain the greatest number of individuals.

```
. use nsyr_example_data, clear
. foreach var of varlist grds workhrs peerus pmonit alcuse {
  2.         setgen st`var´ = stdrank(`var´)
  3. }
. fuzzy stgrds stworkhrs stpeerus stpmonit stalcuse, label(G W P M A)
. tabulate bestfit
    bestfit |      Freq.     Percent        Cum.
------------+-----------------------------------
       WPMA |         45        1.45        1.45
       WPMa |         32        1.03        2.47
       WPmA |        198        6.36        8.84
       WPma |         47        1.51       10.35
       WpMA |         37        1.19       11.54
       WpMa |        148        4.76       16.29
       WpmA |         83        2.67       18.96
       Wpma |        113        3.63       22.59
       wPMA |        110        3.53       26.12
       wPMa |        115        3.70       29.82
       wPmA |        318       10.22       40.04
       wPma |        131        4.21       44.25
       wpMA |        158        5.08       49.33
       wpMa |        882       28.34       77.67
       wpmA |        220        7.07       84.74
       wpma |        475       15.26      100.00
------------+-----------------------------------
      Total |      3,112      100.00
```

Thus 1.45% of adolescents are likely to experience all of the independent measures at above-median levels (WPMA), while the most common configuration (wpMa), with 28.34% of the sample best fitting it, indicates low work hours, not many friends who use subtances, high parent monitoring, and low levels of individual alcohol use. This command would also have produced five new individual variables (G, W, P, M, A), all copies of the original variables for which they were named. If the user did not wish to keep these new variables, `drop` could be specified.

The `label()` option, however, is not necessary. Had it not been specified, we would have seen the following:

```
. fuzzy stgrds stworkhrs stpeerus stpmonit stalcuse
. tabulate bestfit
    bestfit |      Freq.     Percent        Cum.
------------+-----------------------------------
       BDEF |         45        1.45        1.45
       BDEf |         32        1.03        2.47
       BDeF |        198        6.36        8.84
       BDef |         47        1.51       10.35
       BdEF |         37        1.19       11.54
       BdEf |        148        4.76       16.29
       BdeF |         83        2.67       18.96
       Bdef |        113        3.63       22.59
       bDEF |        110        3.53       26.12
       bDEf |        115        3.70       29.82
       bDeF |        318       10.22       40.04
       bDef |        131        4.21       44.25
       bdEF |        158        5.08       49.33
       bdEf |        882       28.34       77.67
       bdeF |        220        7.07       84.74
       bdef |        475       15.26      100.00
------------+-----------------------------------
      Total |      3,112      100.00
```

The only difference in the two commands, therefore, is the naming of the configurations. Of course, using `label()` may be helpful in keeping straight the sets involved in creating the configurations. Also, because `label()` was not used, the variables `B`, `D`, `E`, and `F` are, by default, deleted when the program is terminated, but `keepsets` could be specified to prevent this if the user wanted to use these generic names in future calls. In the remaining examples, we will use the single letter designations with the `fuzzy` command (treating it as though the user had already run the first invocation of `fuzzy` above). But all of the commands could be run with original variables (with or without the `label()` option).

❑ Technical note

While the total in the `bestfit` variable does add up to the total number of nonmissing cases, this may not always be true. Cases scoring 0.5 on all individual predictor sets will not appear because they belong equally to all configurations.

❑

Next one might want to get a sense of the relationship between the independent variable sets by using fuzzy-set methods.

(*Continued on next page*)

```
. fuzzy G W P M A, matx(coincid suffnec) standardized
Coincidence Matrix
             |        G          W          P          M          A
-------------+-------------------------------------------------------
           G |    1.000
           W |    0.603      1.000
           P |    0.574      0.422      1.000
           M |    0.699      0.485      0.453      1.000
           A |    0.595      0.459      0.630      0.482      1.000
Sufficiency and Necessity Matrix
             |        G          W          P          M          A
-------------+-------------------------------------------------------
           G |    1.000      0.214      0.276      0.677      0.311
           W |    0.603      1.000      0.422      0.485      0.459
           P |    0.574      0.311      1.000      0.453      0.630
           M |    0.699      0.178      0.225      1.000      0.261
           A |    0.595      0.311      0.579      0.482      1.000
```

The high work hours and high grades sets overlap by 60% of their possible shared area, as shown by their 0.603 coincidence score. The standardization option is especially helpful because several of the variables (e.g., work hours and alcohol use) do not contain many members at high degrees. In fact, the coincidence score between grades and work hours was one of the lowest when the size of the variables was not accounted for by the standardization option, which demonstrates the utility of also invoking `standardized` with the coincidence matrix. We also see that high parent monitoring is the single set that—alone—is most sufficient for predicting the outcome (consistency = 0.699). Both of these matrices are returned and could be used to run further tests.

Seeing that the variable sets are indeed related, it would now be helpful to examine their resulting configurations' sufficiency with the outcome. To do so, we will run a series of tests, the first of which is the most basic, to get a sense of each configuration's consistency with the outcome.

```
. fuzzy G W P M A, settest(yvv)
Y-Consistency vs. Set Value
Set      YConsist  Set Value    F         P         NumBestFit
wpma       0.790     0.800     1.78     0.182       475
wpmA       0.771     0.800     4.76     0.029       220
wpMa       0.739     0.800    69.15     0.000       882
wpMA       0.805     0.800     0.14     0.705       158
wPma       0.770     0.800     4.11     0.043       131
wPmA       0.650     0.800    83.20     0.000       318
wPMa       0.801     0.800     0.01     0.926       115
wPMA       0.759     0.800     5.51     0.019       110
Wpma       0.804     0.800     0.06     0.804       113
WpmA       0.759     0.800     3.12     0.077        83
WpMa       0.790     0.800     0.41     0.523       148
WpMA       0.843     0.800     3.95     0.047        37
WPma       0.819     0.800     0.76     0.385        47
WPmA       0.648     0.800    47.15     0.000       198
WPMa       0.835     0.800     1.95     0.162        32
WPMA       0.796     0.800     0.03     0.869        45
```

Each configuration's consistency is displayed, as well as the resulting test against 0.800. The results indicate that only the configuration WpMA is significantly more consistent than 0.800 at the 0.05 level. Of course, one of the primary advantages of the fuzzy command is that it can perform more stringent tests of each configuration's consistency value. We look for the configurations that have y consistencies significantly greater than 0.700, as well as significantly greater than their n consistencies.

```
. fuzzy G W P M A, settest(yvv yvn) sigonly greater(col1) conval(.700) common
Y-CONSISTENCY vs N-CONSISTENCY
Set         YCons     NCons       F         P     NumBestFit
wpma        0.790     0.734     16.49     0.000       475
wpMa        0.739     0.646     45.83     0.000       882
Wpma        0.804     0.727      6.79     0.009       113
WpMa        0.790     0.698      9.46     0.002       148
WPma        0.819     0.715      5.70     0.017        47
Y-Consistency vs. Set Value
Set       YConsist  Set Value    F         P         NumBestFit
wpma        0.790     0.700    136.51     0.000       475
wpmA        0.771     0.700     27.69     0.000       220
wpMa        0.739     0.700     27.20     0.000       882
wpMA        0.805     0.700     68.45     0.000       158
wPma        0.770     0.700     21.90     0.000       131
wPMa        0.801     0.700     48.95     0.000       115
wPMA        0.759     0.700     11.33     0.001       110
Wpma        0.804     0.700     43.85     0.000       113
WpmA        0.759     0.700      6.45     0.011        83
WpMa        0.790     0.700     31.27     0.000       148
WpMA        0.843     0.700     43.19     0.000        37
WPma        0.819     0.700     29.96     0.000        47
WPMa        0.835     0.700     28.48     0.000        32
WPMA        0.796     0.700     12.50     0.000        45
Common Sets
wpma wpMa Wpma WpMa WPma
```

Notice that the sigonly and greater() options apply to both the yvv and yvn tests, while conval() pertains only to yvv. Using the common option is a quick way to see the configurations that pass both tests. From the given output, it appears that wpma, wpMa, Wpma, WpMa, and WPma are the most highly consistent configurations with good grades. It is possible though that these configurations may logically overlap. To perform the reduction:

(Continued on next page)

```
. fuzzy G W P M A, settest(yvv yvn) sigonly greater(col1) conval(.700)
> common reduce
  (output omitted)
Common Sets
wpma wpMa Wpma WpMa WPma

5 Solutions Entered as True

Minimum Configuration Reduction Set
Wma pa

Final Reduction Set

Coverage
Set         Raw Coverage     Unique Coverage     Solution Consistency
W*m*a          0.113             0.017                   0.778
p*a            0.732             0.636                   0.603

Total Coverage = 0.749
Solution Consistency = 0.604
```

The five initial configurations have been collapsed into two. (The "Minimum Configuration Reduction Set" displays the reduced configurations from the initial step. In certain cases, this will be different than the "Final Reduction Set", which results from the second step—employing prime implicants—of the Quine–McCluskey algorithm.) We also can tell that low personal alcohol use (`a`) is key to higher grades. When this base set is conjoined with either low peer substance use (`p`) or high work hours combined with low parent monitoring (`W * m`), the adolescent is also likely to be achieving higher grades.[3] Additionally, this example displays the benefit of fuzzy methods more generally, as we find a somewhat surprising relationship between work hours and a positive outcome. Normally, higher work hours have been found to increase the likelihood of a number of delinquent activities (Bachman and Schulenberg 1993; Safron, Schulenberg, and Bachman 2001; and Paschall, Ringwalt, and Flewelling 2002). When this high work intensity, however, is concurrently combined with low personal alcohol use and low parent monitoring, it is associated with higher academic achievement. Understanding why low parent monitoring is included in this configuration is difficult without further examination although it may indicate independent youth (i.e., working many hours breaks them from parent monitoring, but they still do not participate in delinquent activities, which all conjoins to be associated with positive academic outcomes).

Finally, it would have been possible to run the entire set of analyses in the following single command:

[3] We recognize that the reduced configuration `p*a` has a consistency below the 0.7 set value. This drop in value (i.e., increased coverage but reduced effectiveness) is due to the increased amount of people who belong to the minimized configuration. We recognize this difference as an important methodological (and substantive) issue that should be addressed in future research.

```
. drop G W P M A

. fuzzy stgrds stworkhrs stpeerus stpmonit stalcuse, label(G W P M A)
> matx(coincid suffnec) standard settest(yvv yvn) sigonly greater(col1)
> conval(.700) common reduce
  (output omitted)
Common Sets
wpma wpMa Wpma WpMa WPma

5 Solutions Entered as True

Minimum Configuration Reduction Set
Wma pa

Final Reduction Set

Coverage
Set                            Raw Coverage   Unique Coverage   Solution Consistency
STWORKHRS*stpmonit*stalcuse       0.113           0.017                0.778
stpeerus*stalcuse                 0.732           0.636                0.603

Total Coverage = 0.749
Solution Consistency = 0.604
```

When the original variables are used along with `reduce`, the reduction output uses the original variable names, making it possible to use this output in potential tables.

❑ Technical note

It is possible to pass configurations to that have consistency scores that are greater with the negation (i.e., $1 - \textit{outcome}$) than the outcome to reduce (e.g., `fuzzy` *varlist*`, settest(yvn) greater(col2) reduce`), but if this is done, `reduce` still uses the outcome to compute the coverage and consistency scores of the reduced configurations. If these values are desired for the negation, we suggest the user create a specific variable, to be used as the outcome, that represents $1 - \textit{outcome}$ and revert to specifying `greater(col1)` with the `yvn` test.

❑

5.2 Postestimation testing and stand-alones

Many of the options within `fuzzy` have been constructed to be used as stand-alone programs, which may be highly useful to test specific configurations. For example, perhaps one is interested in the coincidence of the configurations, in addition to the individual variables:

(*Continued on next page*)

```
. fuzzy G W P M A
. coincid `r(sets)´
Coincidence Matrix
             │       c1        c2        c3        c4        c5        c6
─────────────┼────────────────────────────────────────────────────────────
          r1 │    1.000
          r2 │    0.220     1.000
          r3 │    0.432     0.133     1.000
          r4 │    0.180     0.550     0.162     1.000
          r5 │    0.156     0.168     0.096     0.145     1.000
          r6 │    0.074     0.202     0.046     0.159     0.284     1.000
          r7 │    0.128     0.140     0.113     0.171     0.535     0.215
          r8 │    0.066     0.189     0.056     0.237     0.275     0.407
          r9 │    0.097     0.079     0.065     0.074     0.074     0.042
         r10 │    0.039     0.115     0.025     0.106     0.059     0.064
         r11 │    0.085     0.066     0.076     0.077     0.060     0.034
         r12 │    0.035     0.104     0.028     0.126     0.053     0.056
         r13 │    0.033     0.053     0.021     0.049     0.128     0.074
         r14 │    0.020     0.055     0.013     0.048     0.069     0.089
         r15 │    0.028     0.047     0.023     0.056     0.109     0.062
         r16 │    0.018     0.054     0.014     0.064     0.069     0.082
             │       c7        c8        c9       c10       c11       c12
─────────────┼────────────────────────────────────────────────────────────
          r7 │    1.000
          r8 │    0.343     1.000
          r9 │    0.065     0.046     1.000
         r10 │    0.052     0.073     0.269     1.000
         r11 │    0.069     0.048     0.464     0.180     1.000
         r12 │    0.061     0.087     0.213     0.482     0.237     1.000
         r13 │    0.109     0.086     0.207     0.189     0.139     0.170
         r14 │    0.057     0.090     0.098     0.195     0.071     0.149
         r15 │    0.131     0.100     0.158     0.158     0.175     0.220
         r16 │    0.080     0.133     0.098     0.201     0.102     0.287
             │      c13       c14       c15       c16
─────────────┼────────────────────────────────────────
         r13 │    1.000
         r14 │    0.242     1.000
         r15 │    0.464     0.176     1.000
         r16 │    0.264     0.315     0.369     1.000
```

The `fuzzy` command is used to generate all the possible configurations and then create a full coincidence matrix of these configurations. We could have included the outcome variable in this matrix, but we do not have to in this case. If we had not entered `r(sets)` after the `coincid` command in this case, a coincidence matrix for just the individual set variables would have been reproduced. There are also two alternative methods for producing similar results: (1) in the `fuzzy` command line, the user could have invoked the `keepconfigs` option and then entered

```
. coincid wpma - WPMA
```

or (2) the user could have, without running the first `fuzzy` command, manually entered every possible (or desired) configuration:

```
. coincid wpma wpmA wpMA
```

Again, when these postestimation commands are used, only the individual sets (named as capital letters) must exist in the dataset. For example, we could run the following command as long as the set variables G, W, P, M, and A existed in the dataset (even if the configuration variables did not):

```
. yvn G wpma Wpma WPma
Y-CONSISTENCY vs N-CONSISTENCY
Set         YCons     NCons        F          P     NumBestFit
wpma        0.790     0.734     16.49     0.000        475
Wpma        0.804     0.727      6.79     0.009        113
WPma        0.819     0.715      5.70     0.017         47
```

When using the options as primary programs, they will accept all of the options associated with them in the main fuzzy (e.g., we could have specified sigonly, slevel(), greater(), or a combination of these in the above example). When using the tests in this manner, it is still necessary to specify the dependent variable in addition to the configurations to be tested.

❏ Technical note

If yvo or cmvom is used in this manner, the "other" configuration that each configuration is tested against consists only of those configurations that are entered into the command line. For example, had yvo been used instead of yvn in the last example, each configuration would have been tested against the other two configurations instead of the full possible configuration set. Additionally, if yvo or cmvom is used in this manner, at least two configurations must be specified.

❏

(Continued on next page)

Many of the described extensions work in this manner as well. For example, in figure 1, we can visually see the relationship between a specific configuration and the outcome.

```
. fzplot G Wpma
```

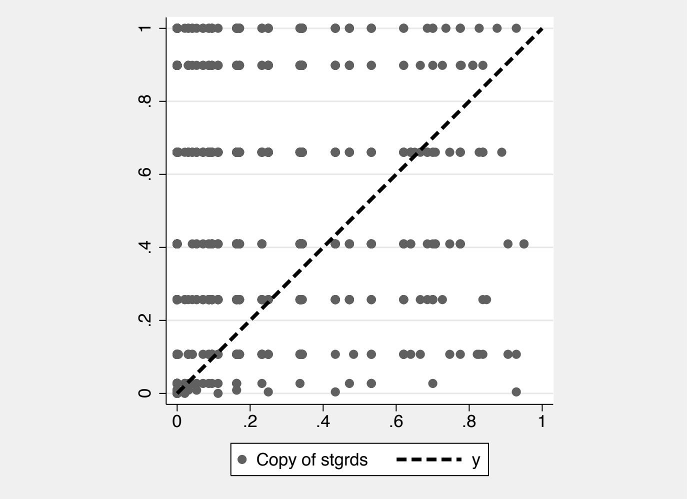


Figure 1. Graph of the relationship between outcome (`stgrds`) and configuration (`Wmpa`).

Or it may be beneficial to test two configurations' consistencies against one another. For example, given the reduction above, we might have concluded that the configuration `Wma` was perhaps the most important because of its higher consistency. But we can test if this difference is significant with

```
. yvy G Wma pa
Y-CONSISTENCY vs Y-CONSISTENCY
WmaYcons    paYCons           F         P
0.778        0.603         142.28     0.000
```

This test shows that, in fact, the two configurations' consistencies are significantly different. Further, the configurations do not have to be "full" in the sense that they use all of the original sets. Rather, the configurations are generated by taking the minimum value of the specified combination. `reduce` used as a stand-alone program does require "full" configurations to perform the reduction properly.

Finally, these commands can be used following `fuzzy` (and each other) as postestimation commands. When this is done, the default is to use the original variable list from the last run `fuzzy` (or other postestimation command). For example,

```
. fuzzy G W P M A
. yvv
Y-Consistency vs. Set Value
Set      YConsist  Set Value    F         P         NumBestFit
wpma       0.790     0.800     1.78     0.182       475
wpmA       0.771     0.800     4.76     0.029       220
wpMa       0.739     0.800    69.15     0.000       882
wpMA       0.805     0.800     0.14     0.705       158
wPma       0.770     0.800     4.11     0.043       131
wPmA       0.650     0.800    83.20     0.000       318
wPMa       0.801     0.800     0.01     0.926       115
wPMA       0.759     0.800     5.51     0.019       110
Wpma       0.804     0.800     0.06     0.804       113
WpmA       0.759     0.800     3.12     0.077        83
WpMa       0.790     0.800     0.41     0.523       148
WpMA       0.843     0.800     3.95     0.047        37
WPma       0.819     0.800     0.76     0.385        47
WPmA       0.648     0.800    47.15     0.000       198
WPMa       0.835     0.800     1.95     0.162        32
WPMA       0.796     0.800     0.03     0.869        45
```

This is similar to running the command on one line. But it is also possible to run post hoc examination of limited sets of configurations. For example, if in the last run `fuzzy` command the display was altered using one of the `settest()` options or `reduce`, then it is possible to specify `last` or `usereduction` as an option with the postestimation commands to restrict their analyses to the last displayed set of configurations.

```
. fuzzy G W P M A, settest(yvv yvn) sigonly greater(col1) conval(.700) common
> reduce
  (output omitted)
. yvv, usered conv(.700)
Y-Consistency vs. Set Value
Set      YConsist  Set Value    F         P         NumBestFit
Wma        0.778     0.700    27.39     0.000       160
pa         0.603     0.700   233.46     0.000      1618
```

The use of the options as postestimation commands also allows for more complex and flexible sets of tests. For example, if the user wanted to identify and reduce all the configurations having consistencies with the outcome that were greater than with the negation (but not necessarily significantly so) and that were also significantly greater than 0.700, it would be impossible to do in one call to `fuzzy` (because the `settest()` options apply to all of the tests in `settest()`). But the user could accomplish this goal using a simple series of commands:

(*Continued on next page*)

```
. fuzzy G W P M A, settest(yvn) greater(col1)
Y-CONSISTENCY vs N-CONSISTENCY
Set        YCons     NCons       F         P     NumBestFit
wpma       0.790     0.734     16.49     0.000       475
wpMa       0.739     0.646     45.83     0.000       882
wpMA       0.805     0.801      0.03     0.855       158
wPMa       0.801     0.768      1.50     0.220       115
Wpma       0.804     0.727      6.79     0.009       113
WpmA       0.759     0.756      0.00     0.949        83
WpMa       0.790     0.698      9.46     0.002       148
WpMA       0.843     0.796      1.48     0.223        37
WPma       0.819     0.715      5.70     0.017        47
WPMa       0.835     0.753      3.07     0.080        32
WPMA       0.796     0.791      0.01     0.925        45
. yvv, last conval(.700) greater(col1) sigonly
Y-Consistency vs. Set Value
Set      YConsist  Set Value    F         P          NumBestFit
wpma       0.790     0.700    136.51     0.000       475
wpMa       0.739     0.700     27.20     0.000       882
wpMA       0.805     0.700     68.45     0.000       158
wPMa       0.801     0.700     48.95     0.000       115
Wpma       0.804     0.700     43.85     0.000       113
WpmA       0.759     0.700      6.45     0.011        83
WpMa       0.790     0.700     31.27     0.000       148
WpMA       0.843     0.700     43.19     0.000        37
WPma       0.819     0.700     29.96     0.000        47
WPMa       0.835     0.700     28.48     0.000        32
WPMA       0.796     0.700     12.50     0.000        45
. reduce
11 Solutions Entered as True
Minimum Configuration Reduction Set
pa pM Ma Wp Wa WM

Final Reduction Set
Coverage
Set       Raw Coverage     Unique Coverage     Solution Consistency
p*a          0.732             0.134                 0.603
p*M          0.602             0.026                 0.717
M*a          0.611             0.034                 0.721
W*p          0.152             0.013                 0.666
W*a          0.153             0.012                 0.696
W*M          0.129             0.011                 0.750
Total Coverage = 0.847
Solution Consistency = 0.598
```

Finally, if the user knew the specific configurations that should be treated as true, then `reduce` could be used as its own stand-alone command. As a simple illustrative example, if the user had some reason to believe only those configurations with high work hours and high parent monitoring should be considered true, the following command could be used:

```
. reduce G WpMa WPMa WpMA WPMA
4 Solutions Entered as True
Minimum Configuration Reduction Set
WM

Final Reduction Set
Coverage
Set       Raw Coverage     Unique Coverage     Solution Consistency
W*M               0.129               0.129                   0.750
Total Coverage = 0.129
Solution Consistency = 0.750
```

As expected, the configuration set reduces to just high work hours (W) and high parent monitoring (M).

6 Acknowledgments

We would like to thank Nick Cox (University of Durham, UK) for early help with the programming that got the larger program off the ground. We also appreciate Charles Ragin and Sarah Strand (University of Arizona) for useful testing and feedback of earlier versions. The development of this program was supported in part by the Center for Developmental Sciences, University of North Carolina at Chapel Hill.

7 References

Bachman, J. G., and J. E. Schulenberg. 1993. How part-time work intensity relates to drug use, problem behavior, time use, and satisfaction among high school seniors: Are these consequences or merely correlates? *Developmental Psychology* 29: 220–235.

Greckhamer, T., V. F. Misangyi, H. Elms, and R. Lacey. 2007. Using Qualitative Comparative Analysis in Strategic Management Research: An Examination of Combinations of Industry, Corporate, and Business-unit Effects. *Organizational Research Methods OnlineFirst* 1–32.

Kalleberg, A. L., and S. Vaisey. 2005. Pathways to a good job: Perceived work quality among the machinists in North America. *British Journal of Industrial Relations* 43: 431–454.

Lin, N., and W. M. Ensel. 1989. Life stress and health: Stressors and resources. *American Sociological Review* 54: 382–399.

Longest, K. C., and P. Thoits. 2007. The stress process and physical health: A configurational approach. Paper presented at the American Sociological Annual Meeting.

Mahoney, J. 2003. Long-run development and the legacy of colonialism in Spanish America. *American Journal of Sociology* 109: 50–106.

Paschall, M. J., C. Ringwalt, and R. L. Flewelling. 2002. Explaining higher levels of alcohol use among working adolescents: An analysis of potential explanatory variables. *Journal of Studies on Alcohol* 63: 169–178.

Ragin, C. 2000. *Fuzzy-Set Social Science*. Chicago: University of Chicago Press.

———. 2006. Set relations in social research: Evaluating the consistency and coverage. *Political Analysis* 14: 291–310.

———. 2008. Fuzzy set analysis: Calibration versus measurement. In *Oxford Handbook of Political Methodology*, ed. J. Box-Steffensmeier, H. Brady, and D. Collier. Oxford: Oxford University Press.

Ragin, C. C. 1987. *The Comparative Method: Moving Beyond Qualitative and Quantitative Strategies*. Berkeley: University of California Press.

Roscigno, V. J., and R. Hodson. 2004. The organizational and social foundations of worker resistance. *American Sociological Review* 69: 14–39.

Safron, D. J., J. E. Schulenberg, and J. G. Bachman. 2001. Part-time work and hurried adolescence: The links among work intensity, social activities, health behaviors, and substance use. *Journal of Health and Social Behavior* 42: 425–449.

Schuit, A. J., A. J. M. Van Loon, M. Tijhuis, and M. C. Ocké. 2002. Clustering of lifestyle risk factors in a general adult population. *Preventive Medicine* 35: 219–224.

Shanahan, M. J., L. D. Erikson, S. Vaisey, and A. Smolen. 2007. Helping relationships and genetic propensities: A combinatoric study of DRD2, mentoring, and educational continuation. *Twin Research and Human Genetics* 10: 285–298.

Smith, C., and M. L. Denton. 2003. Methodological design and procedures for the National Study of Youth and Religion. Technical report, University of North Carolina, Chapel Hill, NC. http://www.youthandreligion.org/.

Smithson, M., and J. Verkuilen. 2006. *Fuzzy Set Theory: Applications in the Social Sciences*. Thousand Oaks, CA: Sage.

Thoits, P. A. 1995. Stress, coping, and social support processes: Where are we? What next? *Journal of Health and Social Behavior* 35: 53–79.

Vaisey, S. 2007. Culture, structure, and community: The search for belonging in 50 urban communes. *American Sociological Review* 72: 851–873.

About the authors

Kyle C. Longest is a PhD candidate in the department of sociology at the University of North Carolina at Chapel Hill. His research interests focus on adolescent development with special attention to identity, education, substance use, and the transition out of high school.

Stephen Vaisey is a doctoral student in the department of sociology at the University of North Carolina at Chapel Hill. His research focuses on the relationship between culture and cognition.

The Stata Journal (2008)
8, Number 1, pp. 105–121

Speaking Stata: Spineplots and their kin

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

Abstract. The term *spineplot* has been applied over the last decade or so to a type of bar chart used particularly for showing frequencies, proportions, or percentages of two cross-classified categorical variables. The principle is that the areas of rectangular tiles are proportional to the frequencies in the cells of a contingency table. Often both coarse and fine structure are easy to see, including departures from independence. The main idea has, in fact, been rediscovered repeatedly over at least the last 130 years. In its most general form, it has been widely publicized under the name *mosaic plots*. This column introduces, discusses, and exemplifies a Stata implementation of spineplots. It is noted that a restriction to two variables is more apparent than real, as either axis of a spineplot can show a composite variable defined by cross combinations of two or more variables.

Keywords: gr0031, spineplots, mosaic plots, bar charts, graphics, categorical data

1 Introduction

The recent history of categorical data analysis within statistical science has been marked by increasing convergence with what might reasonably be dubbed continuous data analysis. Even a generation ago, categorical data analysis was little more to practitioners in many quantitative fields than a ragbag of chi-squared tests and measures of bivariate association. (Indeed, even now many introductory texts appear to offer little more.) In stark contrast, those looking at continuous response variables could exploit a steadily more coherent and powerful toolbox based on regression and ANOVA, seen as members of a family of linear models. However, much greater focus in categorical data analysis over the last few decades on models of various kinds, including log linear and logit models and their several relatives, has greatly lessened the contrasts between the two major parts of statistical practice (see, for example, Agresti [2002]).

One facet of categorical data analysis which continues to receive uneven attention is the use of graphical methods. It is often argued (for example, by Tufte [2001]) that tabular displays, whether of data or summaries or model results, may be more effective or informative than graphs for many categorical problems. Nevertheless, various plotting methods have been suggested for such problems. Friendly (2000) surveys many recent innovations, but none yet appear to challenge bar charts as the most popular graphical method for categorical data.

Bar charts provoke a range of reactions from statistically minded people. Some charts showing only a few frequencies may strike readers as a waste of space in any outlet supposedly aimed at intelligent adults or as too elementary or trivial to deserve much coverage in professional literature. Yet there are many reasons for thinking that bar charts may complement tables helpfully, particularly when the bar charts are well designed and well chosen.

In a previous column, I reviewed some ways of producing such charts in Stata for categorical data (Cox 2004). In this column, I focus on what are now widely known as *spineplots*, discussing the main ideas of spineplots and showing a Stata implementation. The term may be new to you, but the idea may yet be familiar; in any case, it will not appear strange. Spineplots grow out of the basic graphical notion that area may usefully encode frequency, which underlies several other standard forms, including histograms.

2 Spineplots

Names should not matter, but they do. Labels should matter much less than the underlying ideas. A wind rose or a stem-and-leaf plot by any other name is just as sweet, or as prickly, an idea. Yet across times and places and disciplines, all sorts of minor and major confusions can arise when the same name is used for different things, different names are used for the same thing, or authors unthinkingly assume that readers have had the same education and experience and possess the same terminology. Explaining what is, and what is not, a spineplot—or more precisely what is and is not done by the Stata program `spineplot`—thus requires attention to usages in the literature.

The name *spineplot* is credited to Hummel (1996). The term is gaining in popularity but appears already to be differently understood. In the strictest definition, spineplots are one-dimensional, horizontal stacked bar charts, but many discussions and implementations allow vertical subdivision (e.g., by highlighting) into two or possibly more categories. Some literature treats spineplots, as understood here, under the heading of *mosaic plots* (or *mosaicplots*), variously with and without also using the term *spineplot*.

The Stata implementation `spineplot` discussed here adopts a broad interpretation of the term. It works on two categorical variables—not one—and conveys the frequencies shown in a two-way contingency table. (One-dimensional, horizontal stacked bar charts have long been possible in Stata; in Stata 8 the official command `graph hbar` became available.) Conversely, the implementation here does not purport to be a general mosaic plot program capable of producing mosaic plots given three or more categorical variables.

Textbooks and monographs with examples of spineplots and related plots include Friendly (2000); Venables and Ripley (2002); Robbins (2005); Unwin, Theus, and Hofmann (2006); Young, Valero-Mora, and Friendly (2006); and Cook and Swayne (2007). Among several papers, Hofmann's (2000) discussion is clear, concise, and well illustrated.

Mosaic plots, including spineplots as a special case, have been reinvented several times under different names. Hartigan and Kleiner (1981, 1984) introduced them, or reintroduced them, into mainstream statistics. Friendly (2002) cites earlier examples, including the work of Georg von Mayr (1877), Karl G. Karsten (1923), and Erwin J.

Raisz (1934). Hofmann (2007) discusses a mosaic by Francis A. Walker (1874). Other early examples are those of Willard C. Brinton (1914, quoting earlier work), Berend G. Escher (1924), and Hans Zeisel (1947, 1985).[1] Further, independent reinventions of the idea continue to appear (e.g., Bertin [1983]; Feinstein and Kwoh [1988]; and Feinstein [2002]).

3 First examples

Examples will convey the essence far better than a word description. With a nod to Stata tradition, fire up Stata with the **auto** data, and look at the cross classification of two categorical variables: whether cars are foreign (from outside the United States) **foreign** and their 1978 repair record **rep78**. Repair record may be considered to be a response variable; hence, as with scatter plots and the Stata command **scatter**, it is named first to **spineplot** as the variable to be shown on the y axis. **spineplot** does not try to be smart about colors, nor does it know whether a categorical variable is ordered (ordinal) or not (nominal). Thus we here skip the default and move directly to specifying an ordered series of gray scales for bar colors (figure 1):

```
. sysuse auto
(1978 Automobile Data)
. spineplot rep78 foreign, bar1(bcolor(gs14)) bar2(bcolor(gs11))
> bar3(bcolor(gs8)) bar4(bcolor(gs5)) bar5(bcolor(gs2))
```

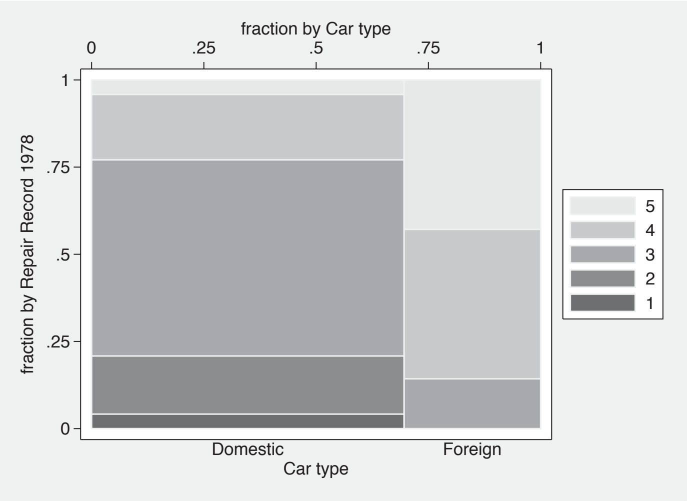


Figure 1. Spineplot of repair record and whether foreign for 74 cars, as produced by **spineplot**

[1] See Anonymous (1967), Robinson (1970), Sills (1992), Anderson (2001), and Hertz (2001) for biographical pieces on several of these pioneers. Karl Karsten has been credited with the idea of hedge funds. Berend Escher is now better known as a brother of Maurits C. Escher, whose own mosaics are immensely more intricate and intriguing than any to be discussed here.

As you might guess, options like `bar1()` and `bar2()` override defaults for the first, second, and subsequent bars. Counting is from the top downward. Here the darkest gray scales show poor repair records. Adopting the reverse choice, or indeed any other choice of colors, is naturally at your discretion. Whatever the choice, the spineplot makes clear that foreign and domestic cars had very different distributions of repair record in 1978.

The graph structure is similar to the structure of a standard two-way contingency table, such as the one `tabulate rep78 foreign` would produce. One detailed difference is that high response values are in the last rows but toward the top of the y axis, reflecting table and graph conventions, respectively. Another detailed difference is that cells with zero frequency are represented in the spineplot by tiles of zero area, that is, not at all.

For interpretation of spineplots, note that cross classification of independent variables would yield tiles that align consistently, as the resulting conditional distributions would be identical. Conversely, departures from independence, or relationships between variables, are shown by failure of alignment. The fine structure of such departures is open to inspection, although limits are imposed by the low visibility of cells with low frequencies and thus low tile areas. Spineplots are especially useful when considering a null hypothesis of independence.

However, in some cases where independence is highly implausible, spineplots may not be particularly effective. A common example is assessing categorical agreement of observers or methods, the problem which to many users is that addressed by the `kappa` command ([R] **kappa**). Here the usual expectation is that the diagonal or near-diagonal cells of the contingency table would show much higher frequencies than those near the opposite corners. Such a pattern would indeed be obvious on a spineplot, but the coloring used in `spineplot` does not make further scrutiny especially helpful.

Be that as it may, let us consider how this spineplot differs from more conventional bar charts. Surprising although it may seem, official Stata offers no direct and obvious command for bar charts of categorical data. Two user-written commands, `catplot` and `tabplot`, are among the alternatives (Cox 2004). Both may be downloaded from the Statistical Software Components archive by using the `ssc` command (see [R] **ssc** for further information).

With `catplot`, there is considerable choice of format. Two close relatives of the spineplot are particularly pertinent. The first shows frequencies (figure 2):

```
. tabulate rep78 foreign
  (output omitted)
. catplot bar rep78 foreign, asyvars stack bar(1, bcolor(gs2))
> bar(2, bcolor(gs5)) bar(3, bcolor(gs8)) bar(4, bcolor(gs11))
> bar(5, bcolor(gs14)) legend(pos(3) col(1))
```

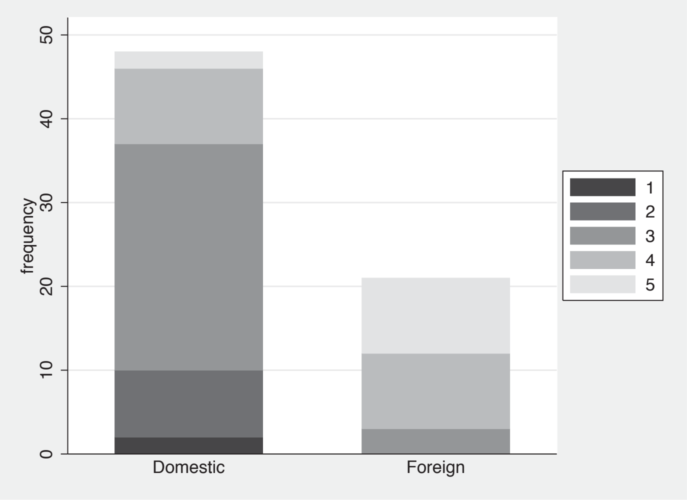


Figure 2. Bar chart of repair record and whether foreign for 74 cars, as produced by `catplot`

The second shows stacked percentages (figure 3):

```
. catplot bar rep78 foreign, asyvars stack percent(foreign) bar(1, bcolor(gs2))
> bar(2, bcolor(gs5)) bar(3, bcolor(gs8)) bar(4, bcolor(gs11))
> bar(5, bcolor(gs14)) legend(pos(3) col(1))
```

(*Continued on next page*)

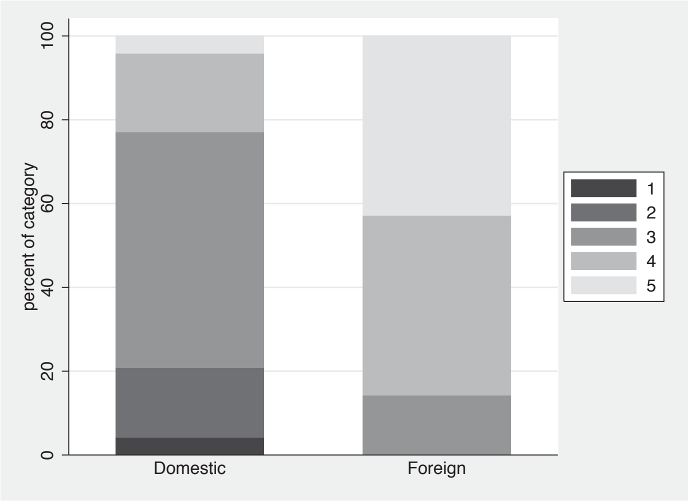


Figure 3. Bar chart of repair record and whether foreign for 74 cars, as produced by `catplot`, showing column percentages

With `catplot`, therefore, as with most bar chart software, it is easy to get a display of stacked frequencies. In that display, proportions or percentages are tacit and so often difficult to read off precisely. It is also easy to get a display of stacked percentages. In that display, the underlying frequencies are not in view. (In this case, `catplot` is a wrapper for `graph bar`, which might suggest the use of the `blabel()` option. But `blabel()` shows numerically what is being shown graphically, and we would want to show something else, so `blabel()` would not help.)

`tabplot` is another possibility. Here the percentage breakdown is shown in figure 4. Omitting the `percent()` option would yield a display of frequencies instead.

```
. tabplot rep78 foreign, percent(foreign) showval(format(%2.1f))
```

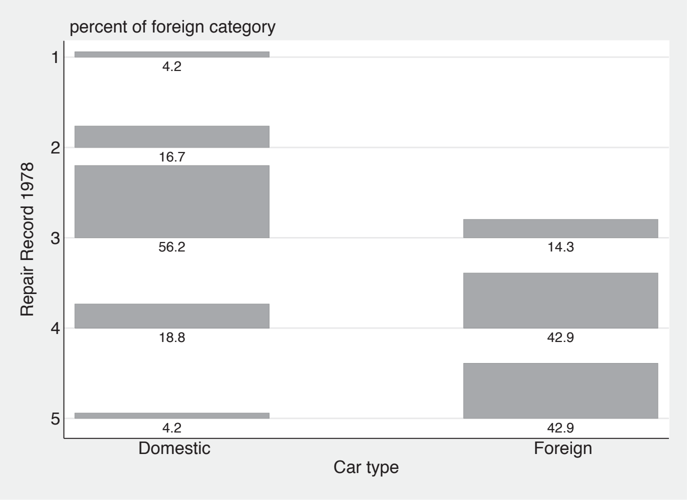


Figure 4. Tabular bar chart of repair record and whether foreign for 74 cars, as produced by `tabplot`, showing column percentages

This plot echoes the structure of a two-way contingency table even more clearly than does a spineplot. A glance at the code shows that much of the work within `tabplot` is done by a call to `twoway rbar`. But again there is a choice between showing frequencies and showing percentages. There is no scope for showing both simultaneously.

In sum: Spineplots show conditional distributions on both axes simultaneously. We can easily add information on absolute frequencies using the `text()` option (figure 5):

```
. by foreign rep78, sort: gen N = _N
. spineplot rep78 foreign, bar1(bcolor(gs14)) bar2(bcolor(gs11))
> bar3(bcolor(gs8)) bar4(bcolor(gs5)) bar5(bcolor(gs2)) text(N)
```

(*Continued on next page*)

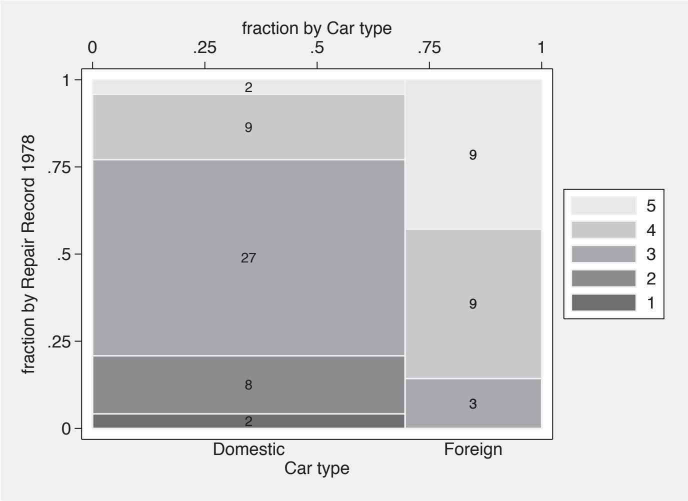


Figure 5. Spineplot of repair record and whether foreign for 74 cars, as produced by `spineplot`, with cell frequencies shown

Missing values in either of the two variables do not perturb the frequencies produced by the `generate` command above. The resulting frequencies are assigned but then ignored by `spineplot`. Conversely, empty cells of the contingency table do not, by definition, correspond to any observations, so counts of zero will not be shown. Combining the count with an `if` or `in` condition would require more care, but the details need not detain us now. Plotting something else, such as standardized residuals given some model, is another possibility. It would often be a good idea to impose a particular numeric format before display, say, by `string(`*residual*`, "%4.3f")`.

Most implementations of spineplots (and, more generally, mosaic plots) in other software omit axes and numerical scales and convey a recursive subdivision according to what may be several categorical variables by a hierarchy of gaps of various sizes. As the graphs produced by `spineplot` are restricted to two variables, axes and numerical scales are kept as defaults. The distinction between categories is conveyed by bar boundaries rather than explicit gaps. Naturally, there is scope for omitting graph elements not desired using standard `graph` options, or, in Stata 10 upward, the Graph Editor. Similarly, users may vary the thickness of bar boundaries, although thick boundaries would distort the relative sizes of what are perceived as bar areas.

The examples already seen raise other small matters of presentation.

First, note the possibility of using `plotregion(margin(zero))` to place axes alongside the plot region. Having a margin is often useful for scatterplots and their kin but is perhaps distracting for spineplots.

As with scatterplots, response variables are usually better shown on the y axis of spineplots. But as with scatter plots, there can be reasons for overriding that convention. (In the Earth or environmental sciences, plotting height above or depth below the land surface on the vertical axis is common and indeed often expected.) If one variable is binary, it is often better to plot that one on the y axis. The `foreign` variable is a case in point. Even though `foreign` is arguably a predictor of `rep78` rather than vice versa, I suggest that the spineplot with `foreign` on the y axis is more congenial. See figure 6 and judge for yourself. Notice that ordering of colors is now less of an issue, as any two distinct colors are ordered one way or the other.

```
. spineplot foreign rep78
```

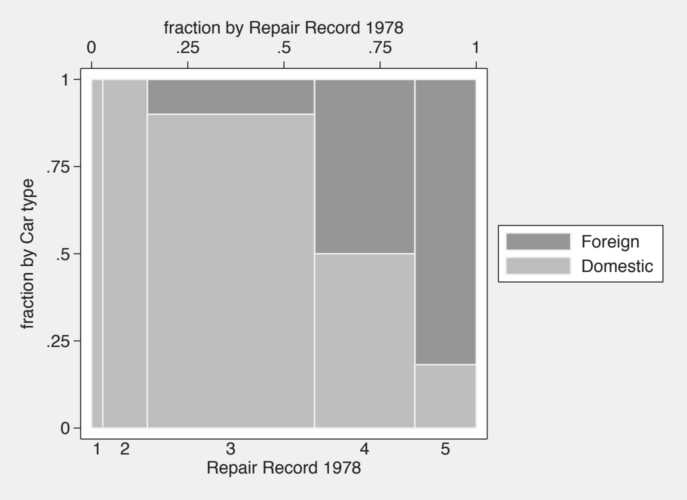


Figure 6. Spineplot of whether foreign and repair record for 74 cars, as produced by `spineplot`, with cell frequencies shown

Even more mundane, but very possibly troublesome in practice, is that if one or more cells have very small frequencies, then a squeeze of some sort is inevitable with `spineplot`. There is no way to show the corresponding tiles, or descriptive labels, or added text, without some difficulty. There are no easy solutions to this problem. You may decide to amalgamate cells; or to use the Graph Editor to ease crowding by moving text, adding arrows, and so forth; or just to use some other kind of graph. Manifestly, all kinds of graphs have some limitations on what they can show easily and effectively, and spineplots are no exception.

4 Discrimination at Berkeley?

A now classic problem among categorical analysts concerns the success or failure of applications for admittance as graduate students at the University of California, Berkeley. The problem was first discussed by Bickel, Hammel, and O'Connell (1975) and since then worked over in various ways in many articles and texts (e.g., Freedman, Pisani, and Purves [1978; 2007]; Friendly [2000]; and Agresti [2002]). Here we use a subset of the data presented by Friendly (2000) and Agresti (2002). The response is decision—admitted or rejected—and the covariates are intended major (masked by identifiers A, B, C, D, E, F) and sex of applicants. The data are available with the files for this column as `berkeley.dta`. They come as frequencies of the various cross combinations, so we must specify weights when we call up `spineplot`. (Alternatively, `expand`ing the dataset on the frequencies so that every individual application became an observation would make that unnecessary; see [D] **expand** for more.)

```
    . use berkeley, clear
    . spineplot decision sex [fw=frequency]
```

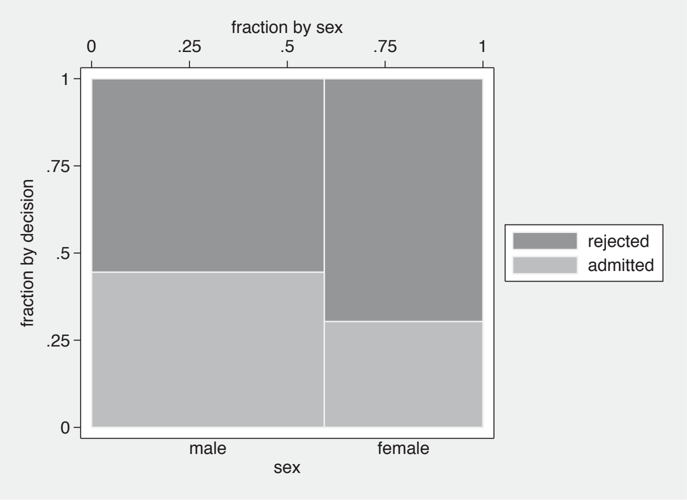


Figure 7. Spineplot of decision versus sex for admissions to various Berkeley graduate majors. At first sight, substantial discrimination against females is evident.

A spineplot of decision versus gender shows apparent discrimination against females (figure 7). However, majors are by no means equally easy to get into (figure 8). A corresponding `tabulate` shows that admission rates vary from 64% for A to 6.4% for F.

```
. spineplot decision major [fw=frequency]
```

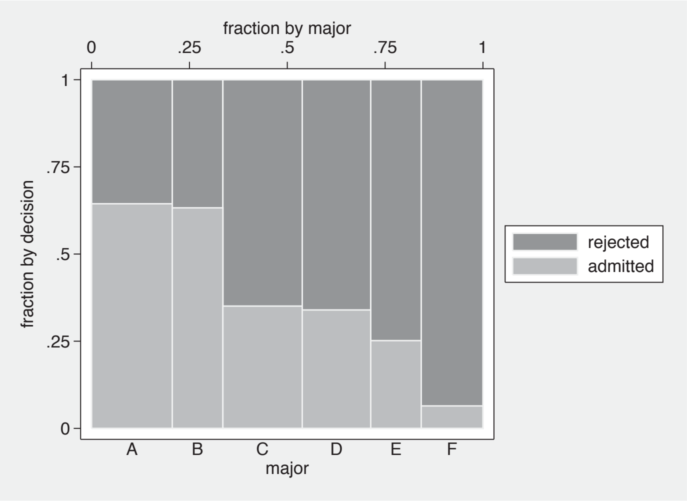


Figure 8. Spineplot of decision versus major for admissions to various Berkeley graduate majors. Acceptance rates vary over a tenfold range.

These are just two-dimensional representations of three-dimensional data. We need to see what structure may exist in three dimensions, including whether there are interactions between the covariates. How can we do that with a two-dimensional display? The answer lies in a composite categorical variable, defined by the cross combinations of two or more categorical variables (Cox 2007). Although not the only method, `egen`'s `group()` function is fine for this purpose:

```
. egen group = group(major sex), label
```

The `label` option is essential for graphs and tables to make sense. Without it, the resulting groups would just show as groups 1 to 12. Further, the order of variables fed to the function is crucial. `group(major sex)` aligns male and female for each major. `group(sex major)` would align majors for each sex. The first is what we need here. In other problems, experimentation with group order may be needed to see what works best.

```
. spineplot decision group [fw=frequency], xlabel(, angle(v) axis(2))
> xtitle("", axis(2)) xtitle(fraction by major and sex, axis(1))
```

Figure 9 shows the result. Vertical axis labels are the lesser of two evils, as there is far too little room for horizontal labels to be legible. Some readers may prefer to try a compromise, say, an angle of 45°. The default title for the bottom x axis would be the variable label for `group`, `group(major sex)`, which we prefer to blank out. Similarly, the title for the top x axis improves on the default.

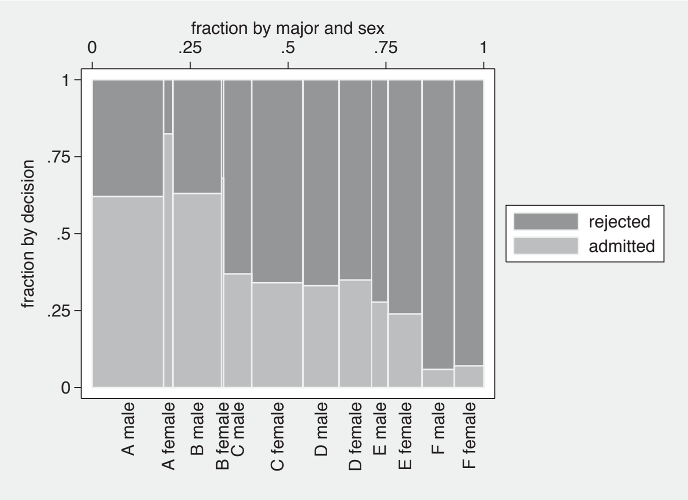


Figure 9. Spineplot of decision versus major and sex for admissions to various Berkeley graduate majors. Females are admitted proportionally more than males to four majors and proportionally less to the remaining two.

The fine structure of the display allows focus on the key question. Major by major, a higher proportion of females than males is admitted to A, B, D, and F, and a lower proportion to C and E. (Admittedly, the comparison for B is not clear on the graph given the small frequencies concerned; for that result, a peek at a table is needed.) Hence, the appearance of discrimination against females appears very much an artifact of the sex and major composition of the applicants or, in other terminology, an example of the amalgamation paradox often named for E. H. Simpson, despite its earlier elucidation by G. U. Yule and several others (Agresti 2002).

A lesson for other examples is that the restriction of spineplots to two variables is more apparent than real given the scope for creating composite variables. Compare what Hofmann (2001) calls "double-decker plots" (for binary responses) and what Wilkinson (2005) calls "region trees".

5 Spineplot details

5.1 Syntax

`spineplot` *yvar xvar* [*if*] [*in*] [*weight*] [,

  `bar1(`*twoway_bar_options*`)` ... `bar20(`*twoway_bar_options*`)`

  `barall(`*twoway_bar_options*`)` `missing` `percent`

  `text(`*textvar* [, *marker_label_options*]`)` *twoway_options*]

`fweight`s and `aweight`s may be specified; see [U] **11.1.6 weight**.

5.2 Description

`spineplot` produces a spineplot for two-way categorical data. The fractional breakdown of the categories of the first-named variable *yvar* is shown for each category of the second-named variable *xvar*. Stacked bars are drawn with vertical extent showing fraction in each *yvar* category given each *xvar* category and horizontal extent showing fraction in each *xvar* category. Thus the areas of tiles formed represent the frequencies, or more generally totals, for each cross combination of *yvar* and *xvar*.

5.3 Options

`bar1(`*twoway_bar_options*`) ... bar20(`*twoway_bar_options*`)` allow specification of the appearance of the bars for each category of *yvar* using options of `twoway bar`.

`barall(`*twoway_bar_options*`)` allows specification of the appearance of the bars for all categories of *yvar* using options of `twoway bar`.

`missing` specifies that any missing values of either of the variables specified should also be included within their own categories. The default is to omit them.

`percent` specifies labeling as percentages. The default is labeling as fractions.

`text(`*textvar* [`,` *marker_label_options*]`)` specifies a variable to be shown as text at the center of each tile. *textvar* may be a numeric or string variable. It should contain identical values for all observations in each cross combination of *yvar* and *xvar*. A simple example is the frequency of each cross combination. To show nothing in particular tiles, use a variable with missing values (either numeric missing or empty strings) for those tiles. A numeric variable with fractional part will typically look best converted to string as, for example, `string(`*residual*`,"%4.3f")`. The user is responsible for choice of tile colors so that text is readable. `text()` may also include *marker_label_options* for tuning the display.

twoway_options refers to options of `twoway`; see [G] ***twoway_options***. By default there are two *x* axes, `axis(1)` on top and `axis(2)` on bottom, and two *y* axes, `axis(1)` on right and `axis(2)` on left.

5.4 Inside the program

You may wish to know more about how the program works. The code, naturally, is open for inspection in your favorite text editor.

The program works by calculating cumulative frequencies. The plot is then produced by overlaying distinct graphs, each being a call to `twoway bar, bartype(spanning)` for one category of *yvar*. By default, each bar is shown with `blcolor(bg) blw(medium)`, which should be sufficient to outline each bar distinctly but delicately. By default also, the categories of *yvar* will be distinguished according to the graph scheme you are using. With the default `s2color` scheme, the effect is reminiscent of canned fruit salad (which may be fine for exploratory work). For a publishable graph, you might want to use something more subdued, such as various gray scales or different intensities, as in this column.

Options `bar1()` to `bar20()` are provided to allow overriding the defaults on up to 20 categories, the first, second, etc., shown. The limit of 20 is plucked out of the air as more than any user should really want. The option `barall()` is available to override the defaults for all bars. Any `bar#()` option always overrides `barall()`. Thus if you wanted thicker `blwidth()` on all bars, you could specify `barall(blwidth(thick))`. If you wanted to highlight the first category only, you could specify `bar1(blwidth(thick))` or a particular color.

Other defaults include `legend(col(1) pos(3))`. At least with `s2color`, a legend on the right implies an approximately square plot region, which can look quite good. A legend is supplied partly because there is no guarantee that all *yvar* categories will be represented for extreme categories of *xvar*. However, it will often be possible and tasteful to omit the legend and show categories as axis label text.

6 Conclusion

Spineplots offer an alternative to more conventional bar charts for showing the data in a two-way contingency table. Their particular merit arises from the fact that frequencies are encoded by tile areas so that, in principle, spineplots convey the information in both marginal and conditional distributions. Departure from independence is shown by failure of tiles to align, which is easily seen. Spineplots can also be extended to higher-order contingency tables, in so far as two or more categorical variables may be combined to form a single composite variable to be shown on either axis.

However, what is a key feature of spineplots can also be a limitation. Cells with small frequencies will be represented by small tiles, and cells with zero frequencies will not be represented at all, so the fine structure associated with such cells may be difficult to discern. Hence, other kinds of bar charts remain complementary for showing the structure of contingency tables.

7 Acknowledgments

Matthias Schonlau, Scott Merryman, and Maarten Buis provoked the writing of the `spineplot` command through challenging Statalist postings. A suggestion from Peter Jepsen led to the `text()` option. Private emails from Matthias Schonlau and Antony Unwin highlighted different senses of spineplots and the importance of sort order. Antony suggested standardizing on “spineplot” rather than “spine plot”. Maarten verified for me that the spineplot in my copy of Escher (1934) also appears in Escher (1924). Vince Wiggins originally told me about the undocumented `bartype(spanning)` option.

8 References

Agresti, A. 2002. *Categorical Data Analysis*. 2nd ed. Hoboken, NJ: Wiley.

Anderson, M. J. 2001. Francis Amasa Walker. In *Statisticians of the Centuries*, ed. C. C. Heyde and E. Seneta, 216–218. New York: Springer.

Anonymous. 1967. In memoriam Prof. Dr. B. G. Escher. *Geologie en Mijnbouw* 46: 417–422.

Bertin, J. 1983. *Semiology of Graphics: Diagrams, Networks, Maps*. Madison: University of Wisconsin Press.

Bickel, P. J., E. A. Hammel, and J. W. O'Connell. 1975. Sex bias in graduate admissions: Data from Berkeley. *Science* 187: 398–404.

Brinton, W. C. 1914. *Graphic Methods for Presenting Facts*. New York: Engineering Magazine Company.

Cook, D., and D. F. Swayne. 2007. *Interactive and Dynamic Graphics for Data Analysis: With R and GGobi*. New York: Springer.

Cox, N. J. 2004. Speaking Stata: Graphing categorical and compositional data. *Stata Journal* 4: 190–215.

———. 2007. Stata tip 52: Generating composite categorical variables. *Stata Journal* 7: 582–583.

Escher, B. G. 1924. *De Methodes der Grafische Voorstelling*. Amsterdam: Wereldbibliotheek.

———. 1934. *De Methodes der Grafische Voorstelling*. 2nd ed. Amsterdam: Wereldbibliotheek.

Feinstein, A. R. 2002. *Principles of Medical Statistics*. Boca Raton, FL: Chapman & Hall/CRC.

Feinstein, A. R., and C. K. Kwoh. 1988. A box-graph method for illustrating relative size relationships in a 2×2 table. *International Journal of Epidemiology* 17: 222–224.

Freedman, D., R. Pisani, and R. Purves. 1978. *Statistics*. New York: W. W. Norton.

———. 2007. *Statistics*. 4th ed. New York: W. W. Norton.

Friendly, M. 2000. *Visualizing Categorical Data*. Cary, NC: SAS Institute.

———. 2002. A brief history of the mosaic display. *Journal of Computational and Graphical Statistics* 11: 89–107.

Hartigan, J. A., and B. Kleiner. 1981. Mosaics for contingency tables. In *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, ed. W. F. Eddy, 268–273. New York: Springer.

———. 1984. A mosaic of television ratings. *American Statistician* 38: 32–35.

Hertz, S. 2001. Georg von Mayr. In *Statisticians of the Centuries*, ed. C. C. Heyde and E. Seneta, 219–222. New York: Springer.

Hofmann, H. 2000. Exploring categorical data: Interactive mosaic plots. *Metrika* 51: 11–26.

———. 2001. Generalized odds ratios for visual modeling. *Journal of Computational and Graphical Statistics* 10: 628–640.

———. 2007. Interview with a centennial chart. *Chance* 20(2): 26–35.

Hummel, J. 1996. Linked bar charts: Analysing categorical data graphically. *Computational Statistics* 11: 23–33.

Karsten, K. G. 1923. *Charts and Graphs: An Introduction to Graphic Methods in the Control and Analysis of Statistics*. New York: Prentice-Hall.

Raisz, E. J. 1934. The rectangular statistical cartogram. *Geographical Review* 24: 292–296.

Robbins, N. M. 2005. *Creating More Effective Graphs*. Hoboken, NJ: Wiley.

Robinson, A. H. 1970. Erwin Josephus Raisz, 1893–1968. *Annals of the Association of American Geographers* 60: 189–193.

Sills, D. L. 1992. In memoriam: Hans Zeisel, 1905–1992. *Public Opinion Quarterly* 56: 536–537.

Tufte, E. R. 2001. *The Visual Display of Quantitative Information*. 2nd ed. Cheshire, CT: Graphics Press.

Unwin, A., M. Theus, and H. Hofmann. 2006. *Graphics of Large Datasets: Visualizing a Million*. New York: Springer.

Venables, W. N., and B. D. Ripley. 2002. *Modern Applied Statistics with S*. New York: Springer.

von Mayr, G. 1877. *Die Gesetzmässigkeit im Gesellschaftsleben*. München: Oldenbourg.

Walker, F. A. 1874. *Statistical Atlas of the United States Based on the Results of the Ninth Census 1870*. New York: Census Office.

Wilkinson, L. 2005. *The Grammar of Graphics*. 2nd ed. New York: Springer.

Young, F. W., P. M. Valero-Mora, and M. Friendly. 2006. *Visual Statistics: Seeing Data with Dynamic Interactive Graphics*. Hoboken, NJ: Wiley.

Zeisel, H. 1947. *Say It with Figures*. New York: Harper.

———. 1985. *Say It with Figures*. 6th ed. New York: Harper & Row.

About the author

Nicholas Cox is a statistically minded geographer at Durham University. He contributes talks, postings, FAQs, and programs to the Stata user community. He has also coauthored 15 commands in official Stata. He wrote several inserts in the *Stata Technical Bulletin* and is an editor of the *Stata Journal*.

The Stata Journal (2008)
8, Number 1, pp. 122–128

Review of Applied Health Economics by Jones, Rice, Bago d'Uva, and Balia

Stephen P. Jenkins
Institute for Social and Economic Research
University of Essex
Colchester, UK
stephenj@essex.ac.uk

Abstract. This article reviews *Applied Health Economics* by Jones, Rice, Bago d'Uva, and Balia.

Keywords: gn0038, health economics, Stata texts

1 Introduction

Jones et al. (2007) provide an excellent introduction to the methods used by health economists for the statistical analysis of survey data. The book is not about health economics concepts or about econometric principles but, instead, about the combination of the two. It provides a practical guide to how to do applied research: it might be more accurately titled *Applied Health Econometrics*. Notwithstanding the health focus, the book will be a useful handbook for advanced undergraduates, graduate students, and researchers in many fields in addition to health.

The authors have well-established reputations in health economics research and teaching. They are all associated with the leading center for health economics in the United Kingdom (at the University of York) and have published in leading health economics and statistics journals using the methods described. Another benefit is that the material has already been "road tested": it is based on their short course and revised in the light of feedback received. The case studies are largely based on research published by the authors.

2 Coverage

The authors group their chapters according to the nature of their outcome variable and the types of survey data available. Thus chapter 1 introduces the five cross-sectional and panel datasets that are used in the case studies in the subsequent chapters: eight waves from the British Household Panel Survey (BHPS), cross-sectional and follow-up data from the British Health and Lifestyle Survey (HALS), cross-sectional data from wave 1 of the Canadian National Population Health Survey (NPHS), cross-sectional data from the World Health Organization Multi-Country Survey for the Indian state of Andhra Pradesh (WHO-MCS), and data from waves 2–4 of the Portuguese component of the European Community Household Panel (ECHP). Chapters 2 and 3 continue the data description theme. The remaining three parts of the book consider categorical data (chapters 4 and 5), survival data (6 and 7), and panel data (8–11). Each chapter contains a case study analyzing one or more health outcome variables with empirical illustrations that are entirely Stata based. (Familiarity with basic Stata structure and syntax is assumed.) Indeed, the text contains substantial amounts of verbatim do-file and log file text (in Courier font to distinguish it from surrounding commentary). All the do-file code presented in the text is downloadable from the authors' web site, http://www.york.ac.uk/res/herc/hedg.html, together with the BHPS and HALS data files used in the analysis. The version of Stata used is not mentioned, but it appears to be Stata 8.

Table 1 shows a more detailed summary of the topics and methods covered, with chapter-by-chapter classifications of the dataset used, technique illustrated, and the corresponding Stata tools used. The table shows that the book's emphasis is on multivariable regression models of the many types most common among microeconometricians. And so, for instance, probit models and their generalizations are favored over logit-type models. The panel and survival data models allow for random intercepts but not random slopes (`xtreg`, `xtprobit`, `xtcloglog`, `pgmhaz8`, `reprob`). Multilevel models, popular among some noneconomics social science disciplines and which might be estimated by using `gllamm` or (in Stata 10) `xtmixed`, `xtmelogit`, or `xtmepoisson`, are not considered. On the other hand, applied econometrics books rarely discuss issues of nonresponse (panel attrition), how to test for attrition bias, or how to apply inverse-probability weighted estimators; this book does. Another feature I like is that the book does not simply fit models; it also shows how to undertake specification tests and how to draw out the implications of parameter estimates.

(*Continued on next page*)

Table 1. Topics and methods covered in *Applied Health Economics*

Chapter	Dataset	Topic	Stata commands and tools
Part I: Data description			
1. Data and survey design		Introduction to datasets and variables	
2. Describing the dynamics of health	BHPS	Data description	do, log, iis, tis, generate, replace, recode, label, egen, sort, by, xtile, cumul, graph bar, graph export, tabulate, summarize
3. Inequality in health utility and self-assessed health	NPHS	Description of distributions and basic regression methods	graph twoway, centile, _pctile, table, glcurve[a], kdensity, global, sktest, regress, oprobit, intreg, predict, test
Part II: Categorical data			
4. Bias in self-reported data	WHO-MCS	Generalized ordered probit models	reshape, program, ml, gop[b], hopit[b], matrix
5. Health and lifestyles	HALS	Regression models for multiple binary outcomes	label, tabulate, describe, foreach, pcorr, icd9, fitstat[a], probit, mvprobit[a], mvppred[a], meffcon[b], meffdum[b]
Part III: Survival data			
6. Smoking and mortality	HALS	Survival analysis with continuous time duration data	quietly, date functions, list, count, stset, stsum, stdes, sts graph, graph combine, streg, stcurve, stsgen, estimates store
7. Health and retirement	BHPS	Survival analysis with discrete time duration data	forvalues, xtdes, ltable, pgmhaz8[a], cloglog, xtloglog
Part IV: Panel data			
8. Health and wages	BHPS	Linear panel data models	local, while, regress, xtreg, hausman, xthtaylor
9. Modelling the dynamics of health	BHPS	Limited dependent variable panel models	dprobit, xtprobit, quadchk, clogit, reprob[a]
10. Non-response and attrition	BHPS	Testing for attrition inverse-probability bias; weighted estimators	Same as chapter 9
11. Models for health-care use	ECHP	Count data models	nbreg, gnbreg, zip, zinb, ztnb, lcnb2_pan[b], lc_hurdle_pan[b]

Note: Stata commands and tools are listed only once, in the chapter in which first used.

[a] User-written program available from SSC and/or *Stata Journal* archives.

[b] Program written by the authors with code shown in the text (also downloadable from their web site).

Some readers might be bemused by the classification of linear regression (regress), ordered probit regression (oprobit), and interval-censored regression (intreg) as part of the bundle of tools for data description rather than modeling. On the other hand, contrary to some social scientists' prejudices about econometricians' obsessions, there is also much space given to nonregression-based numerical and graphical methods for data description and data checking and cleaning. This forms the bulk of chapters 2 and 3. I also like the introduction to basic elements of Stata programming. Throughout the book, the authors show how tools such as local and global macros and foreach and forvalues loop constructions make analysis more effective. This could have been flagged more as a feature.

Reflecting the fact that the book is an offshoot from a real-life research program, the Stata tools used are a mixture of built-in commands and other tools and user-written commands. Taken all together, the portfolio used illustrates two important strengths of Stata, namely, its extensive suite of canned routines and its extensibility. Interestingly, none of the user-written programs drawn on here that are downloadable from SSC or the *Stata Journal* archives were originally written for analysis of health survey data, but clearly they are useful in this context. The authors themselves also develop several special maximum likelihood estimators with program and ml code, notably for generalized ordered probit models (allowing for nonparallel cutpoints, with heterogeneity, i.e., "hopit" models) and for panel-data count models (latent-class hurdle and negative binomial models). They also show how to derive average partial effects after multivariate probit estimation. So, not only can readers learn more about advanced Stata programming from the commentaries on the construction of the code, but they may also apply the econometric techniques in other contexts. The code is in the public domain.

3 Discussion

The book meets its stated aims well. But having had my appetite whetted for this sort of material for teaching and training, I would like more. A second edition might address several matters.

The book could benefit from greater overall editorial control providing greater cross-chapter consistency in style—both in the text and in the Stata programming. For example, some chapters use global macros to hold sets of covariate names, whereas others use local macros for the same task. Both methods work, but there is no explanation of the difference. I have a few other minor quibbles with the code used. For example, tis and iis are used instead of tsset, which was introduced in version 7. stsplit is cited as the means for episode splitting when preparing discrete-time survival data for analysis. It works, but in my experience, its use misleads students who become confused about the differences between estimation of continuous-time and discrete-time survival models. For the latter, I prefer to avoid all reference to st commands and implement episode splitting by using expand.

The book contains inconsistent references to user-written programs and how to obtain them. Chapter 2 uses glcurve to draw Lorenz and concentration curves, but the chapter does not name the authors (Jenkins and Van Kerm 1999, 2007) nor, more importantly for readers, does it explain where to download the command (the latest version is a *Software Update* to *Stata Journal* volume 7, number 2). Chapter 3 does cite a *Stata Journal* article about `mvprobit` and `mvppred` (Cappellari and Jenkins 2003), but only several pages later is the `ssc` command referred to as a means to obtain them. And in the intervening pages, the `findit` command is cited as the means to obtain the `fitstat` command, but the authors are not mentioned. (They are Long and Freese [2006], who have now incorporated the `fitstat` command into their `spost` package.) The source for Mark Stewart's `reprob` command for dynamic binary dependent variable panel regression is cited as an unpublished paper. `findit` will not find it. In fact, the command has become the more powerful `redpace` (Stewart 2006), available from the *Stata Journal* archives.

Of course I am not complaining about the use of commands that I have co-written! The authors could have explained in one place (e.g., the *Preface*) the several ways to obtain user-written commands and how to get the latest version. Then they could have referred back to this whenever required later in the book. The *Preface* would also have been the place to cite the version of Stata being used and the implications of using later versions. A more systematic introduction to the use of do-files and log files would have been well-placed there, too.

The discussion of datasets (chapter 1) should be clearer about the relationship between the original datasets and the ones that are actually used later in the book. For example, it is helpful to know from the start that only the first 8 waves of the BHPS are used (15 are now available to registered users from the UK Data Archive). Similarly, the discussion of the ECHP is general, referring to all 14 participating countries and up to 8 waves of data (the maximum); in chapter 11, we finally learn that the case study is based on only 3 waves of data from Portugal. The authors could also be clearer about how users might obtain the Stata datasets used in their case studies. Five different surveys are discussed in chapter 1, and readers may gain the impression that Stata datasets based on them are also available. In fact, only data from the BHPS and HALS are downloadable from the authors' web site. Two is better than none, but readers would benefit from knowing precisely how to obtain the other datasets.

A related and more substantial point is that one important stage of analysis is rather downplayed in the book. There is a useful distinction between the dataset(s) that is available from a data distributor and an analysis dataset that is derived from it. For example, each wave of BHPS data consists of more than 10 files, each of which keys on different identifiers (distinguishing respondents, enumerated individuals, households, spells, income sources, etc.) and which contains different types of variables. To construct a panel dataset for analysis, one usually has to merge data from different files within each wave, and then append or merge files across different waves (depending on whether one wants long- or wide-format data). In my experience, it is this stage of analysis that is the most problematic for researchers (even BHPS experts) and most likely to lead to errors. The authors use one `bhps.dta` file for analysis, with no discussion of how it was created from the original files. Similar remarks apply to the other datasets. Even if the book itself has no space for the do-file code showing how to create the analysis datasets, it would be useful to have this information on the authors' web site alongside the other code.

Although the authors emphasize their focus on hands-on empirical analysis, eschewing discussion of the microeconometric methods, I think the book would benefit from more systematic referencing of such discussions elsewhere. Most chapters have a final short *Overview* section, which is little more than a brief recap of the topics covered. This section would be more useful if it included "further reading"—citations to the relevant chapters in econometric and statistics texts and perhaps also to related health economics literature.

The book's *Introduction* could also be expanded to provide readers with a greater awareness of the microeconometric and health economics topics that the book does not cover. I am not a health economist and so am poorly placed to make suggestions about this. Nonetheless, I observe that all the case studies here refer to individuals and their health. None of the studies considers data about health providers (e.g., hospitals or doctors) or health insurers and which of these may have provided a greater role in the discipline of health economics outside Europe. On the methods side, I am aware that there is a substantial microeconometric literature within health economics that has debated the relative merits of the "Heckman selection" model and the "two-part" model for modeling data comprising a significant minority of observations with zero value for the outcome variable (e.g., health care expenditure) and the remainder with a positive value. (See Jones 2000 for a review.) Although the book considers only empirical research on topics that the authors themselves have undertaken, and it is an impressively wide range, it would be useful for readers to get a feel for what topics and methods have not been covered.

The text is remarkably free of typographical errors. One suggestion for the future is that the bullet points used to preface do-file command lines be omitted; they are redundant.

4 Conclusions

Applied Health Economics is a good practical guide to the application of microeconometric methods to survey data on health and related variables. It could be used as a complementary text for econometrics courses at the advanced undergraduate or postgraduate level or for specialist training courses, and it will be a useful reference for applied researchers in health and other fields.

5 References

Cappellari, L., and S. P. Jenkins. 2003. Multivariate probit regression using simulated maximum likelihood. *Stata Journal* 3: 278–294.

Jenkins, S. P., and P. Van Kerm. 1999. sg107: Generalized Lorenz curves and related graphs. *Stata Technical Bulletin* 48: 25–29. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 274–278. College Station, TX: Stata Press.

———. 2007. Software Update: sg107: Generalized Lorenz curves and related graphs. *Stata Journal* 7: 280.

Jones, A. M. 2000. Health Econometrics. In *Handbook of Health Economics*, ed. A. J. Culyer and J. P. Newhouse, 265–344. Amsterdam: North Holland.

Jones, A. M., N. Rice, T. Bago d'Uva, and S. Balia. 2007. *Applied Health Economics*. London: Routledge.

Long, J. S., and J. Freese. 2006. *Regression Models for Categorical Dependent Variables Using Stata*. 2nd ed. College Station, TX: Stata Press.

Stewart, M. B. 2006. Maximum simulated likelihood estimation of random effects dynamic probit models with autocorrelated errors. *Stata Journal* 6: 256–272.

About the author

Stephen Jenkins is the Director of the Institute for Social and Economic Research, University of Essex, UK (the home of the British Household Panel Survey), a research professor at DIW Berlin, and an associate editor of the *Stata Journal*. He is an applied economist who has contributed several often-downloaded commands to the SSC archive.

The Stata Journal (2008)
8, Number 1, pp. 129–133

Review of Event History Analysis with Stata by Blossfeld, Golsch, and Rohwer

Frank Kalter
Institute of Sociology
University of Leipzig
Leipzig, Germany
fkalter@sozio.uni-leipzig.de

Abstract. This article reviews *Event History Analysis with Stata* by Blossfeld, Golsch, and Rohwer.

Keywords: gn0039, causal modeling, event history analysis, TDA, teaching

1 Introduction

This is a book for which I have been waiting many years! I was introduced to event history analysis as a young research assistant in the early '90s, and I was lucky enough to attend a workshop by Hans-Peter Blossfeld and Götz Rohwer once in Cologne, Germany. Here we received a week-long introduction to event history analysis, and I have seldom again profited from such a short time investment. First, we got used to the statistical package Transition Data Analysis (TDA), written by Götz Rohwer. TDA is a freeware program that, at the time, by far outperformed alternative (expensive) software packages for event history analysis in terms of features and speed. Second, we learned to apply event history analysis. The workshop was designed in an extremely motivating, hands-on style, using data and examples from actual research. We moved step by step, always linking the formal content to precise TDA command files. You could improve your understanding by simply rerunning the commands, and later on, you could easily modify the original files to satisfy your own research needs. This step-by-step process is how methods should be taught.

Fortunately, *Techniques of Event History Modeling* appeared in 1995, with a second edition in 2002 (Blossfeld and Rohwer 1995, 2002), presenting an update and extension of the workshop in book form and again delivering all the necessary material. I still use *Techniques of Event History Modeling* to refresh or extend my knowledge, and I know many self-learners who have learned successfully from it. Having found that this textbook is perfect for teaching students, I have used it for this purpose very effectively.

In spite of its unquestionable advantages, TDA has always produced some discomfort. As just one example, the production of graphics for survivor and hazard curves as well as their ability to be imported into text files has never been easy or convincing. Most importantly, however, TDA alone was never enough. The major hurdle of event history analysis does not lie in running the models but in constructing the necessary data files.

To do that, everybody—including me—referred to her or his usual software package. The export and import involved with this process did cost a lot of time when modifying analyses in daily research, but teaching fluency in both software packages was an even more severe difficulty.

Then, fortunately, Stata began to implement more and more event history features and soon became a serious alternative for event history analysis. People began to *translate* the TDA command files into Stata do-files, thus saving the advantages of the book while avoiding the transaction costs of switching between statistical packages. With the increased use of Stata and all the advantages Stata provides, this translation was an obvious thing to do. And this translation is exactly what Blossfeld, Golsch, and Rohwer have addressed in the book reviewed here.

2 Contents

Chapter 1 starts with demonstrating the usefulness of event history analysis both in general and specifically for the social sciences. The chapter explains the relative advantages of event history data as compared to cross-sectional or panel data, justifying the reasoning with references to recent research. Blossfeld, Golsch, and Rohwer put specific emphasis on the need for investigating causal relationships in the social sciences and on the role of event history analysis within this attempt. Building on these discussions, the authors introduce basic terminologies and statistical concepts of event history analysis.

Chapter 2 continues with an explanation of the basic structure of event history data files and how to use them with Stata. The reader becomes familiar with the training data file (a subsample of 600 episodes from the German Life History Study), which provides the basis for all following analyses. This chapter provides some rudimentary information on how to handle Stata to allow even Stata beginners to rerun all the do-files used as examples throughout the book. Chapter 3 discusses the two basic nonparametric descriptive methods, i.e., life table and product-limit (Kaplan–Meier) estimation, along with methods to compare survivor functions. The authors helpfully demonstrate how the output resulting from Stata do-files could also be calculated by hand according to the formulas—this way the reader acquires an especially deep understanding of both the output and the underlying algorithms.

The next five chapters focus on parametric models. Chapter 4 starts with the simplest one, the exponential transition rate model, which assumes that the baseline transition rate is constant over time. Step by step, the reader learns about models without covariates, models with time-constant covariates, models with multiple events, and models with repeated events. Building on this information, chapter 5 introduces the piecewise-constant exponential model. This model allows the baseline transition rate to vary between different time intervals while being constant within. This often provides a very flexible fit to the data, and piecewise-constant exponential models turn out to be very useful tools in practical research. Chapter 6 then adds another important step by introducing time-dependent covariates and the technique of episode splitting. The authors' general ideas and precise Stata handling are embedded into a broader theoretical discussion of the causal approach to interdependent processes. The chapter examines four examples from actual research in order to illustrate the utility of time-dependent covariates and the flexibility of the episode-splitting method. Chapter 7 further enriches the toolbox by adding four parametric models that can be fitted by Stata: Gompertz models, Weibull models, loglogistic models, and lognormal models. In each of these models, the general shape of the baseline transition rate over time is determined by two basic parameters. The sections give a short discussion of the underlying formulas, followed by estimations of models without covariates, covariates linked to one of the model's parameters, and covariates linked to both of the parameters. Chapter 8 rounds off the treatment of parametric models, suggesting methods to check parametric assumptions either by direct graphical methods or by checking the pseudoresiduals.

Chapter 9 treats semiparametric models, specifically the Cox model. It gives an idea of the partial-likelihood estimation, shows how to introduce time-dependent covariates, discusses the proportionality assumption together with consequences of its violation, and shows how to get graphical insights into the underlying baseline transition rates. The final chapter is devoted to problems of model specification. The discussion begins with a look at the consequences of unobserved heterogeneity on the time-dependent shape of the transition rate by showing examples and segues to models with a gamma mixture as possible strategies to this problem.

3 Assessment

As *Event History Analysis with Stata* (Blossfeld, Golsch, and Rohwer 2007) is basically a Stata "translation" of the TDA-based *Techniques of Event History Modeling* (Blossfeld and Rohwer 1995, 2002), it automatically inherits all the strengths of the latter. Above all, it is the book's general didactical concept that makes it a convincing introduction and distinguishes it from rival books. The most basic characteristics in this respect have already been implicitly mentioned in this review: in this book you learn by doing and you learn step-by-step, from simple things to more complicated ones and in a well-designed structure both within chapters and from chapter to chapter. Each necessary step is documented in a do-file, which you can download from http://web.uni-bamberg.de/sowi/soziologie-i/eha/stata/, together with the basic dataset. Running and modifying the do-files gives you immediate and easy access to your own research problems, especially because the discussion of outputs is closely linked to the underlying statistical concepts. Moreover, the authors wrote the book in a very clear language, kept formulas and statistical theory within necessary limits, and cared about general methodological issues and the link to sociological theory. You learn not only to apply event history analysis but also about its general and practical drawbacks and opportunities in the social sciences.

Event History Analysis with Stata takes over not only many strengths but also some shortcomings from the former TDA-based version. For example, what has often been criticized is Blossfeld and Rohwer's (1995, 2002) tendency to concentrate only on continuous-time models while not treating discrete-time models and their tendency to put a bit too much emphasis on parametric models—these tendencies have not changed in *Event History Analysis with Stata*. One may also miss some recent, more elaborate developments in event history modeling.

In addition to these inherited limitations, the update itself seems to contain one basic shortcoming. Certainly, marrying *Techniques of Event History Modeling* to Stata is worth doing, as argued in the introduction. But what we find here is only the mere intersection of the former contents with the features and possibilities of Stata. This could have been much more! One may have wished, for example, that the authors would have chanced to enrich Stata by delivering ado-files for procedures that are still unique for TDA, e.g., the parametric sickle model or the generalized loglogistic model. But the respective pages of the former book were simply cut. Conversely, specific features and possibilities of Stata are not picked up to enrich the content of the former book and to enhance some of its procedures; some are minor but nice Stata options (for example, built-in smoothers to get graphical representations of hazard rates in the product-limit estimator), while others are helpful Stata commands that lead to more fundamental changes in the logic of certain techniques (like using the elegant combination of `stsplit` and `stjoin` to generate time-dependent dummy variables instead of the clumsy solution suggested in section 6.4). As in spoken languages, software translating is often not preferable in a straight one-to-one way—sometimes it is not even possible.

It also seems that, now and then, the Stata version would have required a slight change in the organization of the book: while, for example, the piecewise-constant exponential model is specified by changing only a model selection parameter in TDA, it here uses the `stsplit` command in section 5.2. Episode splitting, however, is not introduced until chapter 6 of the book.

Given the strength of the book, many of these critiques are minor or could be regarded as a matter of taste in the end. Nevertheless, the authors missed a chance to improve the splendid course even further. And it surely is an unquestionable shortcoming that—apart from not using respective features itself—the book does not refer to additional Stata options or alternative Stata solutions.

4 Conclusion

As was the case with its TDA-based predecessors, *Event History Analysis with Stata* offers a wonderful introduction to survival analysis for practicing social scientists who want to learn to apply the techniques successfully in their fields of interest. The systematic and practical approach makes it an ideal textbook for students or a perfect course for self-learners. It is also well-suited as a standard reference book for active researchers. But in all three cases you will—occasionally, at least—have to use an additional reference in order to learn more about the event history–specific features of Stata or to fully employ all the possibilities it offers. For example, *An Introduction to Survival Analysis using Stata* (Cleves et al. 2008) will surely be a helpful complement. In contrast to what the authors state in the preface, however, *Techniques of Event History Analysis* (Blossfeld and Rohwer 1995, 2002) is not a necessary complement—it is simply replaced. Basically, *Event History Analysis with Stata* is a third edition—now, finally, in Stata. Therefore, it would have been more appropriate to stay with the old title. If you already own one of the former editions, you do not need to buy the new one—just download the do-files; you will only miss some updates to the references to research examples. If you do not yet own one of the former editions, you should definitely buy the new Stata-based book simply because it is more convenient to have the information all together. Certainly, you will recommend this book to your students or other event-history beginners and advanced learners as the most important reference.

5 References

Blossfeld, H., K. Golsch, and G. Rohwer. 2007. *Event History Analysis with Stata*. Mahwah, NJ: Lawrence Erlbaum.

Blossfeld, H., and G. Rohwer. 1995. *Techniques of Event History Modeling: New Approaches to Causal Analysis*. Mahwah, NJ: Lawrence Erlbaum.

———. 2002. *Techniques of Event History Modeling: New Approaches to Causal Analysis*. 2nd ed. Mahwah, NJ: Lawrence Erlbaum.

Cleves, M., W. Gould, R. Gutierrez, and Y. Marchenko. 2008. *An Introduction to Survival Analysis Using Stata*. 2nd ed. College Station, TX: Stata Press, in press.

About the author

Frank Kalter is professor of sociology at the University of Leipzig, Germany. His major teaching and research interests include methods and statistics, rational choice theory, and the sociology of migration and integration. His work has appeared in journals like *European Sociological Review*, *Journal of Mathematical Sociology*, *Rationality and Society*, and *Research in Social Stratification and Mobility*. He is currently conducting several comparative research projects on the causes of migration and on mechanisms of immigrants' structural assimilation, collecting primary data and applying longitudinal data analysis in each.

The Stata Journal (2008)
8, Number 1, pp. 134–136

Stata tip 56: Writing parameterized text files

Rosa Gini
Regional Agency for Public Health of Tuscany
Florence, Italy
rosa.gini@arsanita.toscana.it

Stata includes several commands for text file manipulation. A good example is the `copy` command ([D] **copy**). Typing

```
. copy filename1 filename2
```

simply copies `filename1` to `filename2`, regardless of its content.

Often when dealing with text files, you need greater flexibility. Stata can also read and write text files using the `file` suite of commands ([P] **file**). Thereby, you can rewrite a text file by first reading and then writing it with modifications.

```
local filetarget "filename2"
local filesource "filename1"
local appendreplace "replace" /* append or replace */
tempname target source
file open `target´ using `filetarget´, write `appendreplace´ text
file open `source´ using `filesource´, read text
file read `source´ textline
while r(eof) == 0 {
        file write `target´ `"`textline´"´ _n
        file read `source´ textline
}
file close `source´
file close `target´
```

A notable feature of this second way of copying text files is that you can append files to existing files. Even more importantly, you need not copy the file character for character. While rewriting, Stata may substitute the values of local or global macros that have been defined. This allows users to work with a template and produce text with elements substituted for each occasion. Such files may be called "parameterized", as they contain elements constant within a document but variable from document to document.

As an example of the many applications of this simple device, consider the needs of those who periodically access big databases to produce standard reports. The structure of the database is fixed; hence, access will be by a cascade of queries. The queries will be the same every time except for some parameters that change, such as the date (e.g., month, quarter, or year). Stata can access such databases without intermediaries, as SQL code for the queries can be stored in a text file with global macros and be rewritten and executed periodically using the `odbc` command ([D] **odbc**). A file `query_para.sql` might contain the following simple SQL parameterized code:

pr0037

```
CREATE TABLE ${year}_AMI AS
SELECT H.ID, H.CODE_PATIENT, H.YEAR, H.SEX, H.AGE
FROM HOSPITALIZATIONS  H, PATHOLOGIES  H_PATHOL
WHERE H.ID=H_PATHOL.ID AND H_PATHOL.DIAGNOSIS="410" AND ${conditions} AND
> H.YEAR=${year} AND H_PATHOL.YEAR=${year};
CREATE INDEX ${year}_ID ON ${year}_AMI (CODE_PATIENT)
TABLESPACE epidemiology;
ANALYZE TABLE ${year}_AMI compute statistics;
CREATE TABLE ${year}_MORTALITY AS
SELECT DISTINCT CASES.CODE_PATIENT, MOR.DEATH_DATE
FROM ${year}_AMI CASES, MORTALITY  MOR
WHERE MOR.CODE_PATIENT=CASES.CODE_PATIENT;
```

This sequence of queries looks for patients hospitalized for AMI (acute myocardial infarction, or heart attack) in a given year and then links the list of patients to the mortality records to obtain data on survival. As the list of patients may be very long, the code computes an index to perform better linkage.

The following code is a template for one Stata session. For example, substitute any `connect_options` desired for `odbc`.

```
/* set parameters */
global year = 2005
global conditions "H.AGE>64"
/* rewrite text */
local filetarget "query.sql"
local filesource "query_para.sql"
local appendreplace "replace" /* append or replace */
tempname target source
file open `target´ using `filetarget´, write `appendreplace´ text
file open `source´ using `filesource´, read text
local i = 1
file read `source´ textline
while r(eof) == 0 {
        file write `target´ `"`textline´"´ _n
        local ++i
        file read `source´ textline
}
file close `source´
file close `target´
/* execute queries */
odbc sqlfile("query.sql"), dsn("DataSourceName") [connect_options]
/* load and save generated tables */
foreach table in AMI MORTALITY {
        odbc load table("${year}_`table´"), clear [connect_options]
        save ${year}_`table´, replace
}
```

The code will make Stata

1. Write a text file of actual (nonparameterized) SQL code, where `${year}` is substituted by `2005` and `${conditions}` is substituted by `"H.AGE>64"`.

2. Execute the SQL code via `odbc`. This may take some time. If an SQL client is available, a practical alternative is to make Stata call that client and ask it

to execute the SQL using the `shell` command ([D] **shell**). This will make the execution of the queries independent of the Stata session.

3. Load the generated tables.

4. Save each of the generated tables as a Stata `.dta` file for later analysis.

The Stata Journal (2008)
8, Number 1, pp. 137–138

Stata tip 57: How to reinstall Stata

Bill Gould
StataCorp
College Station, TX
wgould@stata.com

Sometimes disaster, quite unbidden, may strike your use of Stata. Computers break, drives fail, viruses attack, you or your system administrator do silly—even stupid—things, and Stata stops working or, worse, only partially works, recognizing some commands but not others. What is the solution? You need to reinstall Stata. It will be easier than you may fear.

1. If Stata is still working, get a list of the user-written ado-files you have installed. Bring up your broken Stata and type

   ```
   . ado dir
   ```

   That should list the packages. `ado dir` is a built-in command of Stata, so even if the ado-files are missing, `ado dir` will work. Assuming it does work, let's type the following:

   ```
   . log using installed.log, replace
   . ado dir
   . log close
   ```

   Exit Stata and store the new file `installed.log` in a safe place. Print the file, too. In the worst case, we can use the information listed to reinstall the user-written files.

2. With both your original CD and your paper license handy, take a deep breath and reinstall Stata. If you cannot find your original license codes, call Stata Technical Services.

3. Launch the newly installed Stata. Type `ado dir`. That will either (1) list the files you previously had installed or (2) list nothing. Almost always, the result will be (1).

4. Regardless, `update` your Stata: Type `update query` and follow the instructions.

5. Now you are either done, or, very rarely, you still need to reinstall the user-written files. In that case, look at the original `ado dir` listing we obtained in step 1. One line might read

   ```
   [1] package mf_invtokens from http://fmwww.bc.edu/RePEc/bocode/m
         ´MF_INVTOKENS´: module (Mata) to convert ...
   ```

   so you would type

   ```
   . net from http://fmwww.bc.edu/RePEc/bocode/m
   . net install mf_invtokens
   ```

dm0035

In the case where the package is from http://fmwww.bc.edu, easier than the above is to type

```
. ssc install mf_invtokens
```

Both will do the same thing. `ssc` can be used only to install materials from http://fmwww.bc.edu/. In other cases, type the two `net` commands.

Anyway, work the list starting at the top.

Unless you have had a disk failure, it is exceedingly unlikely that you will lose the user-written programs. If you do not have a backup plan in place for your hard disk, it is a good idea to periodically log the output of `ado dir` and store the output in a safe place.

The Stata Journal (2008)
8, Number 1, pp. 139–141

Stata tip 58: nl is not just for nonlinear models

Brian P. Poi
StataCorp
College Station, TX
bpoi@stata.com

1 Introduction

The `nl` command makes performing nonlinear least-squares estimation almost as easy as performing linear regression. In this tip, three examples are given where `nl` is preferable to `regress`, even when the model is linear in the parameters.

2 Transforming independent variables

Using the venerable `auto` dataset, suppose we want to predict the weight of a car based on its fuel economy measured in miles per gallon. We first plot the data:

```
. sysuse auto
. scatter weight mpg
```

Clearly, there is a negative relationship between `weight` and `mpg`, but is that relationship linear? The engineer in each of us believes that the amount of gasoline used to go one mile should be a better predictor of weight than the number of miles a car can go on one gallon of gas, so we should focus on the reciprocal of `mpg`. One way to proceed would be to create a new variable, `gpm`, measuring gallons of gasoline per mile and then to use `regress` to fit a model of `weight` on `gpm`. However, consider using `nl` instead:

```
. nl (weight = {b0} + {b1}/mpg)
(obs = 74)
Iteration 0:  residual SS =  1.19e+07
Iteration 1:  residual SS =  1.19e+07
      Source |       SS       df       MS
-------------+------------------------------
       Model |  32190898.6     1  32190898.6         R-squared     =    0.7300
    Residual |  11903279.8    72   165323.33         Adj R-squared =    0.7263
-------------+------------------------------         Root MSE      =  406.5997
       Total |  44094178.4    73  604029.841         Res. dev.     =  1097.134
------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         /b0 |   415.1925   192.5243     2.16   0.034     31.40241    798.9826
         /b1 |   51885.27   3718.301    13.95   0.000     44472.97    59297.56
------------------------------------------------------------------------------
  Parameter b0 taken as constant term in model & ANOVA table
```

st0141

(You can verify that R^2 from this model is higher than that from a linear model of `weight` on `mpg`. You can also verify that our results match those from `regress`ing `weight` on `gpm`.)

Here a key advantage of `nl` is that we do not need to create a new variable containing the reciprocal of `mpg`. When doing exploratory data analysis, we might want to consider using the natural log or square root of a variable as a regressor, and using `nl` saves us some typing in these cases. In general, instead of typing

```
. generate sqrtx = sqrt(x)
. regress y sqrtx
```

we can type

```
. nl (y = {b0} + {b1}*sqrt(x))
```

3 Marginal effects and elasticities

Using `nl` has other advantages as well. In many applications, we include not just the variable x in our model but also x^2. For example, most wage equations express log wages as a function of experience and experience squared. Say we want to fit the model

$$y_i = \alpha + \beta_1 x_i + \beta_2 x_i^2 + \epsilon_i$$

and then determine the elasticity of y with respect to x; that is, we want to know the percent by which y will change if x changes by one percent.

Given the interest in an elasticity, the inclination might be to use the `mfx` command with the `eyex` option. We might type

```
. generate xsq = x^2
. regress y x xsq
. mfx compute, eyex
```

These commands will not give us the answer we expect because `regress` and `mfx` have no way of knowing that `xsq` is the square of `x`. Those commands just see two independent variables, and `mfx` will return two "elasticities", one for `x` and one for `xsq`. If x changes by some amount, then clearly x^2 will change as well; however, `mfx`, when computing the derivative of the regression function with respect to `x`, holds `xsq` fixed!

The easiest way to proceed is to use `nl` instead of `regress`:

```
. nl (y = {a} + {b1}*x + {b2}*x^2), variables(x)
. mfx compute, eyex
```

Whenever you intend to use `mfx` after `nl`, you must use the `variables()` option. This option causes `nl` to save those variable names among its estimation results.

4 Constraints

nl makes imposing nonlinear constraints easy. Say you have the linear regression model

$$y_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \epsilon_i$$

and for whatever reason you want to impose the constraint that $\beta_2\beta_3 = 5$. We cannot use the constraint command in conjunction with regress because constraint only works with linear constraints. nl, however, provides an easy way out. Our constraint implies that $\beta_3 = 5/\beta_2$, so we can type

```
. nl (y = {a} + {b1}*x1 + {b2=1}*x2 + (5/{b2})*x3)
```

Here we initialized β_2 to be 1 because if the product of β_2 and β_3 is not 0, then neither of those parameters can be 0, which is the default initial value used by nl.

The Stata Journal (2008)
8, Number 1, pp. 142–145

Stata tip 59: Plotting on any transformed scale

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

Using a transformed scale on one or the other axis of a plot is a standard graphical technique throughout science. The most common example is the use of a logarithmic scale. This possibility is wired into Stata through options `yscale(log)` and `xscale(log)`; see [G] ***axis_scale_options***. The only small difficulty is that Stata is not especially smart at reading your mind to discern what axis labels you want. When values range over several orders of magnitude, selected powers of 10 are likely to be convenient. When values range over a shorter interval, labels based on multiples of 1 2 5 10, 1 4 7 10, or 1 3 10 may all be good choices.

No other scale receives such special treatment in Stata. However, other transformations such as square roots (especially for counts) or reciprocals (e.g., in chemistry or biochemistry [Cornish-Bowden 2004]) are widely used in various kinds of plots. The aim of this tip is to show that plotting on *any* transformed scale is straightforward. As an example, we focus on logit scales for continuous proportions and percents.

Given proportions p, logit $p = \ln\{p/(1-p)\}$ is perhaps most familiar to many readers as a link function for binary response variables within logit modeling. Such logit modeling is now over 60 years old, but before that lies a century over which so-called logistic curves were used to model growth or decay in demography, ecology, physiology, chemistry, and other fields. Banks (1994), Kingsland (1995), and Cramer (2004) give historical details, many examples, and further references.

The growth of literacy and its complement—the decline of illiteracy—provide substantial examples. In a splendid monograph, Cipolla (1969) gives fascinating historical data but no graphs. Complete illiteracy and complete literacy provide asymptotes to any growth or decay curve, so even without any formal modeling we would broadly expect something like S-shaped or sigmoid curves. Logit scales in particular thus appear natural or at least convenient for plotting literacy data (Sopher 1974, 1979). More generally, plotting on logit scales goes back at least as far as Wilson (1925).

Figure 1 shows how many newly married people could not write their names in various countries during the late nineteenth century, as obtained with data from Cipolla (1969, 121–125) and the following commands:

```
. local yti "% newly married unable to write their names"
. line Italy_females Italy_males France_females France_males Scotland_females
> Scotland_males year, legend(pos(3) col(1) size(*0.8)) xla(1860(10)1900)
> xtitle("") yla(, ang(h)) ytitle(`yti´)
```

gr0032

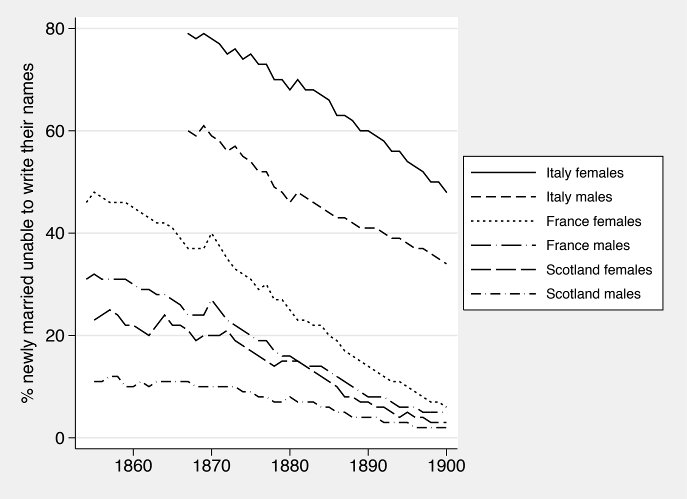


Figure 1. Line plot of illiteracy by sex for various countries in the nineteenth century.

How do we show such data on a logit scale? There are two steps. First, calculate the coordinates you want shown before you graph them. Here we loop over a bunch of variables and apply the transform `logit(`*percent*`/100)`:

```
. foreach v of var *males {
.       gen logit_`v´ = logit(`v´/100)
.       label var logit_`v´ "`: var label `v´´"
. }
```

For tutorials on `foreach` and the machinery used here in looping, see Cox (2002, 2003). Note that, at the same time, we copy variable labels across so that they will show up automatically on later graph legends.

Second, and just slightly more difficult, is to get axis labels as we want them (and axis ticks also, if needed). Even people who work with logits all the time usually do not want to decode that a logit of 0 means 50%, or a logit of 1 means 73.1%, and so forth, even if the `invlogit()` function makes the calculation easy. Logit scales stretch percents near 0 or 100 compared with those near 50. Inspection of figure 1 suggests that `2 5 10(10)80` would be good labels to show for percents within the range of the data. So we want text like `50` to be shown where the graph is showing `logit(50/100)`. The key trick is to pack all the text we want to show and where that text should go into a local macro.

```
. foreach n of num 2 5 10(10)80 {
.       local label `label´ `= logit(`n´/100)´ "`n´"
. }
```

To see what is happening, follow the loop: First time around, local macro `‘n’` takes on the value 2. `logit(2/100)` is evaluated on the fly (the result is about −3.8918) and that is where on our *y* axis the text “2” should go. Second time around, the same is done for 5 and `logit(5/100)`. And so forth over the numlist `2 5 10(10)80`.

Now we can get our graph with logit scale:

```
. line logit_Italy_females logit_Italy_males logit_France_females
> logit_France_males logit_Scotland_females logit_Scotland_males year,
> legend(pos(3) col(1) size(*0.8)) xla(1860(10)1900) xtitle("")
> yla(`label´, ang(h)) ytitle(`yti´)
```

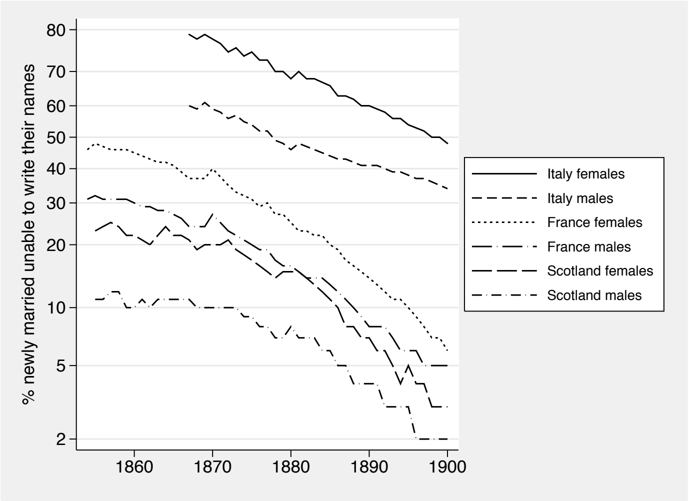


Figure 2. Line plot of illiteracy by sex for various countries in the nineteenth century. Note the logit scale for the response.

Specifically in this example, we now can see data on a more natural scale, complementing the original raw scale. The granularity of the data (rounded to integer percents) is also evident.

Generally, some small details of macro handling deserve flagging. You may be accustomed to a tidy form of local macro definition:

```
. local macname "contents"
```

But the delimiters `" "` used here would, for this problem, complicate processing given that we do want double quotes inside the macro. Note from the previous `foreach` loop that they can be left off, to advantage.

In practice, you might need to iterate over several possible sets of labels before you get the graph you most like. Repeating the whole of the `foreach` loop would mean that the local macro would continue to accumulate material. Blanking the macro out with

```
. local label
```

will let you start from scratch.

The recipe for ticks is even easier. Suppose we want ticks at `15(10)75`, that is, at `15 25 35 45 55 65 75`. We just need to be able to tell Stata exactly where to put them:

```
. foreach n of num 15(10)75 {
.       local ticks `ticks´ `= logit(`n´/100)´
}
```

Then specify an option such as `yticks(‘ticks’)` in the graph command.

Finally, note that the local macros you define must be visible to the graph command you issue, namely within the same interactive session, do-file, or program. That is what local means, after all.

In a nutshell: Showing data on any transformed scale is a matter of doing the transformation in advance, after which you need only to fix axis labels or ticks. The latter is best achieved by building graph option arguments in a local macro.

References

Banks, R. B. 1994. *Growth and Diffusion Phenomena: Mathematical Frameworks and Applications*. Berlin: Springer.

Cipolla, C. M. 1969. *Literacy and Development in the West*. Harmondsworth: Penguin.

Cornish-Bowden, A. 2004. *Fundamentals of Enzyme Kinetics*. 3rd ed. London: Portland Press.

Cox, N. J. 2002. Speaking Stata: How to face lists with fortitude. *Stata Journal* 2: 202–222.

———. 2003. Speaking Stata: Problems with lists. *Stata Journal* 3: 185–202.

Cramer, J. S. 2004. The early origins of the logit model. *Studies in History and Philosophy of Biological and Biomedical Sciences* 35: 613–626.

Kingsland, S. E. 1995. *Modeling Nature: Episodes in the History of Population Ecology*. 2nd ed. Chicago: University of Chicago Press.

Sopher, D. E. 1974. A measure of disparity. *Professional Geographer* 26: 389–392.

———. 1979. Temporal disparity as a measure of change. *Professional Geographer* 31: 377–381.

Wilson, E. B. 1925. The logistic or autocatalytic grid. *Proceedings of the National Academy of Sciences* 11: 451–456.

The Stata Journal (2008)
8, Number 1, p. 146

Software Updates

st0126_1: QIC program and model selection in GEE analyses. J. Cui. *Stata Journal* 7: 209–220.

This program has been updated to include the general negative binomial distribution while only a special case of the negative binomial distribution was previously considered. The help file has also been updated to reflect this extension in example 4.

sxd1_3: Random allocation of treatments in blocks. P. Ryan. *Stata Technical Bulletin* 54: 49–53; 50: 36–37; 49: 43–46. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, pp. 353–358; vol. 9, pp. 352–353; vol. 7, pp. 297–300.

`ralloc` has been enhanced to support up to 10 (previously only 5) treatments in a one-way design; 4×4 two-way factorial designs; and $2 \times 2 \times 2$, $2 \times 2 \times 3$, $2 \times 3 \times 3$, and $3 \times 3 \times 3$ three-way factorial designs. The help file has three new examples showing how to implement three-way factorial designs and how to specify more complex treatment allocation ratios than is apparent from the simple `ratio()` option.