STATA TECHNICAL BULLETIN

May 1993
STB-13

A publication to promote communication among Stata users

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

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

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

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

Contents of this issue

an1.1 STB categories and insert codes

Inserts in the STB are presently categorized as follows:

General Categories:			
an	announcements	*ip*	instruction on programming
cc	communications & letters	*os*	operating system, hardware, & interprogram communication
dm	data management		
dt	data sets	*qs*	questions and suggestions
gr	graphics	*tt*	teaching
in	instruction	*zz*	not elsewhere classified
Statistical Categories:			
sbe	biostatistics & epidemiology	*srd*	robust methods & statistical diagnostics
sed	exploratory data analysis	*ssa*	survival analysis
sg	general statistics	*ssi*	simulation & random numbers
smv	multivariate analysis	*sss*	social science & psychometrics
snp	nonparametric methods	*sts*	time-series, econometrics
sqc	quality control	*sxd*	experimental design
sqv	analysis of qualitative variables	*szz*	not elsewhere classified

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

an31 Statement from the new editor

Sean Becketti, Editor, STB, FAX 913-888-6708

This issue marks the beginning of the third year of the *Stata Technical Bulletin*. It also marks the beginning of my term as editor of the STB. I want to take this opportunity to express my appreciation to Joseph Hilbe, the founding editor of the STB, and the associate editors for their work in producing the first two years of the STB. I'd also like to give you, the readers of the STB, an idea of what you can expect in future issues.

Having just completed my first issue as editor of the STB, I now appreciate Joseph Hilbe's accomplishments more than ever. The work of communicating with authors, reviewing submissions, testing software, and so on is time-consuming and exacting. In performing the job of editor, though, I have the advantage of relying on the existing organization of the STB and on the network of contributors who have sustained the publication. Dr. Hilbe and his associate editors began with only the idea of the STB and developed this idea into the publication you enjoy today. Longtime STB readers can take comfort from the knowledge that Dr. Hilbe and the other associate editors will continue to guide the progress of the STB.

While the position of editor involves considerable work, it is fascinating work as the authors are distinguished individuals and the articles are both useful and intellectually intriguing. The article on tests for normality in the very first issue of the STB touched off a heated debate between leading statisticians—a debate that produced seven additional articles, several Stata programs to perform improved tests of normality, and comprehensive Monte Carlo tests of these competing programs. This is an example of the STB at its best. And there have certainly been other examples in the last two years. My admittedly personal list of favorites includes Salgado-Ugarte and Curts-Garcia's articles on resistant smoothing (STB-7 and STB-11), Royston's nonlinear regression program (STB-7, STB-8, STB-11, and STB-12), Danuso's continuous-time dynamic system simulation package (STB-8), and Gould and Hadi's piece on identifying multivariate outliers (STB-11), to name but a few. As a reader of the STB, I have made good use of these and many of the other programs published in the first two volumes.

As you can tell, I am a satisfied STB reader, and I intend to retain the features that have made the STB useful and successful. Nonetheless I have two specific goals for improving the STB. First, I want to expand the range of topics covered in the STB. In the statistical areas, I want to publish more articles on time series analysis and econometrics, more on nonparametric and semiparametric statistics, more on exploratory data analysis, and more on structural model estimation and simulation. At the same time, I want to retain the contributions in such other areas as biostatistics, epidemiology, and survival analysis. Aside from the purely statistical areas, I want to publish more articles on data management, more articles on interesting data sets, and more articles on teaching. I particularly want to increase the use of the STB as a clearinghouse for questions, problems, and solutions submitted by Stata users.

The second goal is an outgrowth of the first: I want to increase the number of authors publishing in the STB. As the masthead of each issue declares, the STB is "A publication to promote communication among Stata users." The more readers who are represented in these pages, the better. And many different types of articles can help the STB accomplish this mission. The STB already excels at distributing programs that perform specialized tasks. The STB also publishes helpful articles on programming techniques. I believe that readers would also benefit from reading first-person accounts of how Stata helped or hindered them in carrying out a real-life project. In addition, I'd like to see readers submit their unsolved research problems as a way of mobilizing the resources of the community of Stata users.

If you want to influence the character of the STB, contact me with your articles, your ideas, your suggestions, your questions, your complaints, your Stata programs—even your half-finished Stata programs. Encourage your colleagues, your coworkers, and your students to send in submissions as well. Much of what we know as Stata today was developed in response to just these types of interactions with Stata users. I encourage you to make the STB—and, as a consequence, Stata—more and more useful to you as time goes on.

an32 STB-7—STB-12 available in bound format

Sean Becketti, Editor, STB, FAX 913-888-6708

The second year of the *Stata Technical Bulletin* (issues 7–12) has been reprinted in a 240+ page bound book called *The Stata Technical Bulletin Reprints, Volume 2*. These issues of the STB and the book of reprints were prepared under the editorship of Joseph Hilbe. The volume of reprints is available from CRC for $25—$20 for STB subscribers—plus shipping. Authors of inserts in STB-7—STB-12 will automatically receive the book at no charge and need not order.

This book of reprints includes everything that appeared in issues 7–12 of the STB. As a consequence, you do not need to purchase the reprints if you saved your STBs. However, many subscribers find the reprints useful since they are bound in a volume that matches the Stata manuals in size and appearance. Our primary reason for reprinting the STB, though, is to make it easier and cheaper for new users to obtain back issues. For those not purchasing the reprints, note that *zz2* in this issue provides a cumulative index for the second year of the original STBs.

crc30 Linearly interpolate (extrapolate) values

The syntax of `ipolate` is

`ipolate` *yvar xvar*`, generate(`*newvar*`)` [`by(`*varnames*`) epolate`]

`ipolate` creates *newvar* = *yvar* where *yvar* is not missing and fills in *newvar* with linearly interpolated (and optionally extrapolated) values of *yvar* where *yvar* is missing.

Options

`generate()` is not optional; it specifies the name of the new variable to be created.

`by()` specifies that interpolation (and extrapolation) is to be performed separately for the groups designated by equal values of *varnames*.

`epolate` specifies values are to be both interpolated and extrapolated. Interpolation only is the default.

Example 1

```
. list x y

              x          y
   1.         0          .
   2.         1          3
   3.       1.5          .
   4.         2          6
   5.         3          .
   6.       3.5          .
   7.         4         18

. ipolate x y, gen(y1)
(1 missing value generated)

. ipolate y x, gen(y2) epolate

. list

             x          y         y1         y2
  1.         0          .          .          0
  2.         1          3          3          3
  3.       1.5          .        4.5        4.5
  4.         2          6          6          6
  5.         3          .         12         12
  6.       3.5          .         15         15
  7.         4         18         18         18
```

In the above example, the third observation on `y1`, corresponding to $x = 1.5$, is the linearly interpolated value between (x, y) values $(1, 3)$ and $(2, 6)$. The x value 1.5 is halfway between x values 1 and 2 and thus the interpolated value is also halfway between the corresponding y values of 3 and 6: $(3 + 6)/2 = 4.5$.

The first observation of `y1` is missing because it cannot be interpolated (values of (x, y) do not exist on both sides of $x = 0$). Specifying the `epolate` option, as we did when creating `y2`, allows extrapolation as well as interpolation. The value of y was linearly extrapolated from the two nearest (x, y) pairs ($x = 1$ and 2) to obtain 0.

Example 2

You have a data set of the circulation of a magazine for 1970 through 1973. Circulation is recorded in a variable called `circ` and the year in `year`. In a few of the years, the circulation is not known and you want to fill it in by linear interpolation:

```
. ipolate circ year, gen(icirc)
```

Now assume you have data on the circulations for 50 magazines; the identity of the magazine is recorded in a variable called `magazine` (which might be a string variable—it does not matter):

```
. ipolate circ year, gen(icirc) by(magazine)
```

If `by()` is specified, interpolation is performed separately for each group.

Example 3

You have data on y and x, although some values of y are missing. You wish to smooth the $y(x)$ relation using lowess (see [5s] ksm) and then fill in missing values of y using interpolated values:

```
. ksm y x, gen(yls) lowess
. ipolate yls x, gen(iyls)
```

crc31 Categorical variable histogram

The syntax of `hist` is

`hist` *varname* [*weight*] [`if` *exp*] [`in` *range*] [`,` `incr(`*#*`)` *graph_options*]

`fweight`s are allowed.

`hist` graphs a histogram of *varname*, the result being quite similar to '`graph` *varname*`, histogram`'. `hist`, however, is intended for use with integer-coded categorical variables. `hist` determines the number of bins automatically, the x-axis is automatically labeled, and the labels are centered below the corresponding bar. `hist` may be used with categorical variables taking on up to 50 unique values.

Options

`incr(`*#*`)` specifies how the x-axis is to be labeled. `incr(1)`, the default if *varname* reflects 25 or few categories, labels the minimum, minimum + 1, minimum + 2, . . ., maximum. `incr(2)`, the default if there are more than 25 categories, would label the minimum, minimum + 2, . . ., etc.

graph_options refers to any of the options allowed with `graph`'s `histogram` style excluding `bin()`, `xlabel()`, and `xscale()`. These do include, for instance, `freq`, `ylabel()`, `by()`, `total`, and `saving()`.

Example

You have a categorical variable `rep78` reflecting the repair records of automobiles. It is coded 1 = Poor, 2 = Fair, 3 = Average, 4 = Good, and 5 = Excellent. You could type

```
. graph rep78, histogram bin(5)
```

to obtain a histogram. You should specify `bin(5)` because your categorical variable takes on 5 values and you want one bar per value. (You could omit the option in this case, but only because the default value of `bin()` is 5; if you had 4 or 6 bars, you would have to specify it.) In any case, the resulting graph, while technically correct, is aesthetically displeasing because the numeric code 1 is on the left edge of the first bar while the numeric code 5 is on the right edge of the last bar.

Using `hist` is easier:

```
. hist rep78
```

`hist` not only centers the numeric codes underneath the corresponding bar, it also automatically labels all the bars.

Figure 1 was drawn by typing

```
. hist rep78, by(foreign) total
```

The `by()` and `total` options—allowed with `graph, histogram` or with `hist`—drew separate histograms for domestic and foreign cars along with a third histogram for the combined group.

Figure 2 was drawn by typing

```
. hist candi [freq=pop], by(inc) total ylab yline noaxis
                              title(Exit Polling By Family Income)
```

using data collected by Voter Research and Surveys based on questionnaires completed by 15,490 voters from 300 polling places on election day. The data was originally printed in the *New York Times*, November 5, 1992 and was reprinted in Lipset (1993).

In both of these examples, each bar is labeled; if your categorical variable takes on many values, you may not want to label them all. Typing

```
. hist myvar, incr(2)
```

would label every other bar. Specifying `incr(3)` would label every third bar, and so on.

Caution

`hist` is not a general replacement for `graph, histogram`. `hist` is intended for use with categorical data only, which is to say, noncontinuous data. If you wanted a histogram of automobile prices, for instance, you would still want to use the more general `graph, histogram` command.

Figures

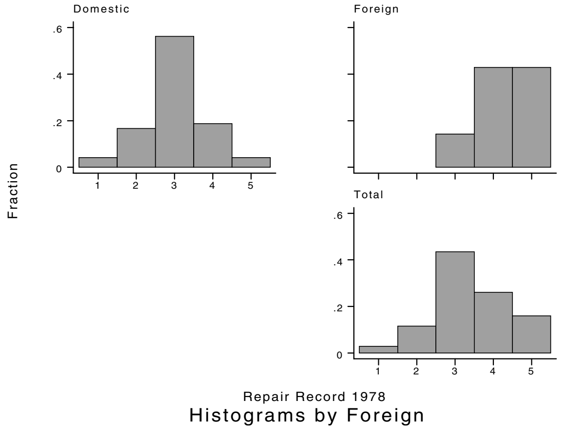

Figure 1

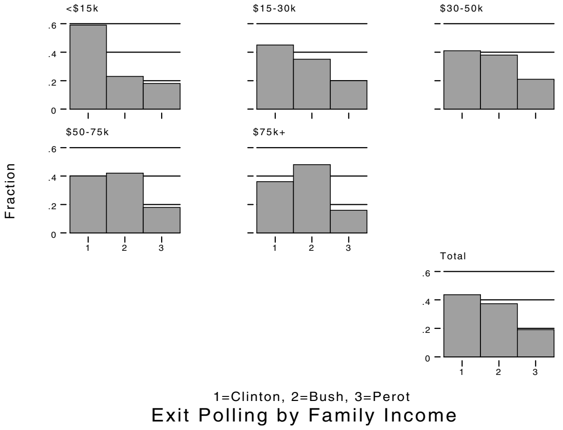

Figure 2

References

Lipset, S. M. 1993. The significance of the 1992 election. *Political Science and Politics* 26(1): 7–16.

dm12.1 | Selecting claims from medical claims data bases

Robert J. Vaughn, MPH, Clinical Projects Manager, Utah Peer Review Organization, FAX 801-487-2296

In Vaughn (1993), I published two ado files, `anyproc` and `anydx` for selecting claims from medical claims data bases. `anyproc` selects claims based on procedure codes and `anydx` selects claims based on diagnosis codes. For instance,

```
. anydx spinal 721.42 721.91 724.00 724.02 724.09
```

creates a new variable called `spinal` that is equal to 1 if any of the five diagnosis codes listed above are found and is equal to 0 otherwise.

Charles Chapin of Health Services Advisory Group, Inc., Phoenix, Arizona discovered a small bug in these programs and also suggested a useful extension. I have fixed the bug and added the suggested extension. The corrected programs are available on the STB-13 diskette along with updated help files.

The original versions of `anydx` and `anyproc` would not accept requests for a single procedure or diagnosis code. For example, the command

```
. anydx spinal 721.42
```

would fail with the error message "`invalid syntax`". An effective, though inelegant, solution to this problem was to repeat the code, that is, to type

```
. anydx spinal 721.42 721.42
```

This command would produce the appropriate result. However, the corrected versions of `anydx` and `anyproc` fix the problem, and single codes are now accepted.

The suggested extension allows a range of codes to be selected easily. Imagine that you wish to select all the diagnosis codes between 721.42 and 722.53 inclusive. Rather than type each of these codes, the new programs allow you to type

```
. anydx spinal 721.42 722.53, range
```

In this case, the "spinal" variable is set to 1 if any of the diagnosis codes are between 721.42 and 722.53 and to 0 otherwise. The syntax for this extension is

$$\left\{ \texttt{anydx} \mid \texttt{anyproc} \right\} \textit{newvar mincode maxcode} \texttt{ , range}$$

Both *mincode* and *maxcode* must be numbers, that is, no alphabetic or other non-numeric characters can appear. However, the diagnosis and procedure code variables in the data base may contain non-numeric codes. Such codes will simply be ignored. To select non-numeric codes, the original forms of `anydx` and `anyproc` must be used, that is, each of the codes must be typed separately.

References

Vaughn, Robert J. 1993. Selecting claims from medical claims data bases. *Stata Technical Bulletin* 12: 11–12.

dm13 | Person name extraction

William Gould, CRC, FAX 310-393-7551

Anyone who works directly with collected data must occasionally work with the names of individuals. Sometimes the names must be put on output. Sometimes the names are the only identifier or one of a handful of identifiers, all of which are untrustworthy, and must actually be put to use in merging data sets. Anyone who has worked with names knows how unpleasant this task can be. One can spend literally hours combing through printouts searching for the "keypunch" errors.

`extrname` is a partial solution to this problem. The syntax of `extrname` is

$$\texttt{extrname}\ \textit{varname}\ \left[\texttt{if}\ \textit{exp}\right]\ \left[\texttt{in}\ \textit{range}\right]\ \Big[\texttt{, all prefix(}\textit{newvar}_1\texttt{) first(}\textit{newvar}_2\texttt{)}$$
$$\texttt{middle(}\textit{newvar}_3\texttt{) last(}\textit{newvar}_4\texttt{) suffix(}\textit{newvar}_5\texttt{) affil(}\textit{newvar}_6\texttt{) odd(}\textit{newvar}_7\texttt{)}\ \Big]$$

`extrname` attempts to extract American-style person names into new variables corresponding to titles, first, middle, and last name.

At least one option must be specified and, generally, you will want to specify all of them; `all` provides a convenient way to do this.

Options

`all` is equivalent to specifying `prefix(prefix) first(first) middle(middle) last(last) suffix(suffix) affil(affil) odd(odd)`.

`prefix(`*newvar*$_1$`)` declares that *newvar*$_1$ is to contain the prefix (Mr., Dr., etc.) of the extracted name.

`first(`*newvar*$_2$`)` declares that *newvar*$_2$ is to contain the first name or initial.

`middle(`*newvar*$_3$`)` declares that *newvar*$_3$ is to contain the middle name or initial.

`last(`*newvar*$_4$`)` declares that *newvar*$_4$ is to contain the last name.

`suffix(`*newvar*$_5$`)` declares that *newvar*$_5$ is to contain the suffix (such as Jr., Sr., III, etc.)

`affil(`*newvar*$_6$`)` declares that *newvar*$_6$ is to contain the affiliations (such as M.D., Esq., Ph.D., etc.).

`odd(`*newvar*$_7$`)` declares that *newvar*$_7$ is to contain nonzero values where `extrname` believes it had problems extracting the name. See *Codes* below for a list of the values and their meaning. It is strongly advised that you specify this option if you do not specify `all`.

Remarks

A name is defined as consisting of parts called the prefix, first, middle, last, suffix, and affil (short for affiliation). For instance:

Name	prefix	first	middle	last	suffix	affil
Smith				Smith		
Roger Smith		Roger		Smith		
Roger A. Smith		Roger	A.	Smith		
Roger A. Smith, Jr.		Roger	A.	Smith	Jr.	
Dr. Roger A. Smith, Jr.	Dr.	Roger	A.	Smith	Jr.	
Dr. Roger A. Smith, Jr., MD	Dr.	Roger	A.	Smith	Jr.	M.D.

`extrname` attempts to extract names of the form:

[*first* [*middle*]] *last*
last [, *first* [*middle*]]

`extrname` would understand any of the following as well as the above:

Smith, Roger
Smith, Roger A.
Dr. Smith, Roger A.
Smith, Dr. Roger A.

In general, however, `extrname` will not understand

last [*first* [*middle*]]

that is, last name first with no comma between the last and the first name. “Smith Roger” would be interpreted as first name Smith and last name Roger.

I say in general because there are some cases where, even with the omitted comma, `extrname` will be able to determine that the last name came first (as in “Smith Roger A.” where the hanging middle initial makes clear that Smith is the last name).

The point of `extrname` is to divide names into their components even when the name is “messy.” For instance, `extrname` will properly process:

input	resulting last, first middle
`Smith R`	`"Smith", "R" ""`
`Roger St. Craig`	`"St. Craig", "Roger" ""`
`John Mc Call`	`"McCall", "John" ""`
`AB Smith`	`"Smith", "A." "B."`
`NY YON`	`"Yon", "Ny" ""`
`B van Hooser`	`"Van Hooser", "B." ""`
`A van der sleuss`	`"Van Der Sleuss", "A." ""`
`D de Bolt`	`"De Bolt", D." ""`
`T de la Rosa`	`"De La Rosa", "T." ""`

The input names do not have to be of mixed case (`"SMITH R"` would be okay) but, if they are, the casing information will be exploited (e.g., `"MR SMITH"` could be Mr. Smith or M. R. Smith and will be interpreted as the latter by `extrname`, but `"Mr Smith"` would be correctly interpreted as Mr. Smith).

Prefix

The resulting `prefix()` can contain

```
Mr.
Ms.        Miss     Mrs.
Dr.        Prof.    Prof. Dr.    drs.
Sgt.       Lt.      Lt. Cmdr.    Cmdr.    Cap.
Lt. Col.   Col.     Gen.
```

I emphasize, this is what `prefix()` may contain on output, not how the *prefix* must appear in the input. Dr. might be spelled out or missing the period, Prof. Dr. might be Dr. Prof. or even Doctor Professor, and so on.

Suffix

The resulting `suffix()` can contain

```
Jr.    Sr.    II    III    IV
```

Affiliation

The resulting `affil()` can contain:

```
M.D.    Ph.D.    Esq.
```

Codes

In addition to extracting the name, `odd()` produces a flag indicating whether `extrname` thought it had problems. This flag is only suggestive; just because `extrname` does not think it has problems does not mean it does not (e.g., input `"SMITH ROBERT"` resulting in last name Robert). Similarly, the "problem" cases may not be problems at all.

`odd()` contains

−1. There are multiple problems (listed below).

0. No problems were flagged. There may be problems `extrname` did not detect.

11. First name may be first initial and middle initial: The first name has two letters and the middle initial is blank. This was interpreted as a first name, however, because it contains vowels and, among all the two-letter names found, the consonant-only names were less than 60% of the sample. (Logic: the vowels a, e, i, o, u, and y amount to 6/26 or 23% of the alphabet. Assuming the first letters of names are randomly distributed over the alphabet—which they are not—the probability of both initials being consonants is $(1-.23)^2 = .5929$. Thus, we look across all two-letter "names" found in the data and ask if the distribution of consonant-only "names" is what we would expect if the "names" were really initials. We then either treat the vowel-containing subset as names or initials.)

21. First and middle initials may need to be combined to form real first name: An apparently two-letter first name that contained vowels was treated as a first and second initial because, among all the two-letter names found, the consonant-only names were more than 59% of the sample.

101. The first name contains embedded blanks.

102. The middle name contains embedded blanks.

103. The last name contains embedded blanks.

111. The first name contains periods in odd places.

112. The second name contains periods in odd places.

113. The last name contains periods.

Examples

```
. list
                             name
  1.             michael de rohan
  2.                   carlin, al
  3.                  gerald kamp
  4.                   mabery, al
  5.                     ed tatum
  6.                ed fitzgerald
  7.              paul e. abraham
  8.                     perez m.
  9.                      fox, al
 10.              st. aubin, mark
 11.                kristan, edw.
 12.                jose de jesus
 13.    van meter, kenneth w. jr.
 14.            gemeren, john van
 15.                gregory olson
 16.              ramon garcia j.
 17.        st. germain, james w.
 18.            michael van marek
 19.          kenneth r. neal, sr
 20.        st. pierre, donald j.
 21.               h. j. h. bacon
 22.          la rosa, michael l.
 23.                john van beek
 24.             herron, garry r.
 25.                 john van lom
 26.                d. j. johnson
 27.        burke, joseph st. jr.
 28.           boster, donald st.
 29.                   fosen, ken
 30.            pavek, thomas st.
 31.                     lu clark
 32.         lapinsk, michael st.
 33.             robert de napoli
 34.               betty van dyke
 35.                  hoffman, al
 36.         peterson, loreen an.
 37.           van swol, james h.
 38.          st. george, john m.
 39.            o´dell stephen l.
 40.       markwenas, bradley st.
 41.          vander tuuk, tom n.
 42.          charles latiker, jr
 43.                   meyers, ed
 44.                   shelby, al
 45.           ethridge, mark st.
 46.                stefanich, ed
 47.                  al williams
 48.                 riley, james
 49.            mark van de linde
 50.         bakelaar, robert st.

. extrname name, all
                                        ------ lengths ------
                defined       blank     shortest      longest
-------------------------------------------------------------
Original             50           0            7           25
Prefix                0          50
First name           50           0            2            7
Middle name          15          35            2            5
Last name            50           0            3           13
Suffix                4          46            3            3
Affil                 0          50

. tab odd
         odd|      Freq.     Percent        Cum.
------------+-----------------------------------
         -1 |         11       22.00       22.00
          0 |         12       24.00       46.00
         11 |         11       22.00       68.00
        102 |          1        2.00       70.00
        103 |         13       26.00       96.00
        111 |          1        2.00       98.00
        112 |          1        2.00      100.00
------------+-----------------------------------
      Total |         50      100.00
```

```
. list first-suffix
         first     middle           last     suffix
  1.   Michael                  De Rohan
  2.        Al                    Carlin
  3.    Gerald                      Kamp
  4.        Al                    Mabery
  5.        Ed                     Tatum
  6.        Ed                Fitzgerald
  7.      Paul         E.        Abraham
  8.        M.                     Perez
  9.        Al                       Fox
 10.      Mark                 St. Aubin
 11.      Edw.                   Kristan
 12.      Jose                  De Jesus
 13.   Kenneth         W.      Van Meter        Jr.
 14.      John               Van Gemeren
 15.   Gregory                     Olson
 16.    Garcia         J.          Ramon
 17.     James         W.    St. Germain
 18.   Michael                 Van Marek
 19.   Kenneth         R.           Neal        Sr.
 20.    Donald         J.     St. Pierre
 21.        H.      J. H.          Bacon
 22.   Michael         L.        La Rosa
 23.      John                  Van Beek
 24.     Garry         R.         Herron
 25.      John                   Van Lom
 26.        D.         J.        Johnson
 27.    Joseph                 St. Burke        Jr.
 28.    Donald                St. Boster
 29.       Ken                     Fosen
 30.    Thomas                 St. Pavek
 31.        Lu                     Clark
 32.   Michael               St. Lapinsk
 33.    Robert                 De Napoli
 34.     Betty                  Van Dyke
 35.        Al                   Hoffman
 36.    Loreen        An.       Peterson
 37.     James         H.       Van Swol
 38.      John         M.     St. George
 39.   Stephen         L.         O´dell
 40.   Bradley             St. Markwenas
 41.       Tom         N.    Vander Tuuk
 42.   Charles                   Latiker        Jr.
 43.        Ed                    Meyers
 44.        Al                    Shelby
 45.      Mark              St. Ethridge
 46.        Ed                 Stefanich
 47.        Al                  Williams
 48.     James                     Riley
 49.      Mark              Van De Linde
 50.    Robert              St. Bakelaar
```

This example was constructed from a larger data set of 61,080 distinct names; 50% of the above data is a random sample from the larger data and the remaining 50% is from a subset thought to be more difficult than average. Thus, the fact that only 12 cases (24% of the sample) were designated as nonproblems should not be taken as representative.

When run on the full sample, `tabulate odd` reports

```
. tab odd
         odd|      Freq.     Percent        Cum.
------------+-----------------------------------
         -1 |         46        0.08        0.08
          0 |      60881       99.67       99.75
         21 |         38        0.06       99.81
        102 |         18        0.03       99.84
        103 |         80        0.13       99.97
        111 |          5        0.01       99.98
        112 |         10        0.02      100.00
        113 |          2       0.00      100.00
------------+-----------------------------------
      Total |      61080      100.00
```

Execution speed

`extrname` is slow. Samples of 91 to 2,912 names were used to estimate

$$\text{seconds} = 20.48 + .0409n$$

on a SPARCstation IPC. Running the full data of 61,080 observations took 2,896 seconds (or 48 minutes); the equation would have predicted 2,519 seconds.

Request for improvements

As already noted, during development, `extrname` was tested on a (messy) data set of 61,080 different names. Nevertheless, there are probably many names that `extrname` will process incorrectly. This can be because it is logically impossible, based on the input, to know what correctly is (e.g., `"MR SMITH"`) or because `extrname` is not sufficiently sophisticated. As Adams (1992, 135) put it, "A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools."

If you encounter a name that is not of the form last-name-first-with-missing-comma that `extrname` should be able to correctly extract but does not, please fax it to me.

References

Adams, D. 1992. *Mostly Harmless.* New York: Harmony Books.

dm13.1 String manipulation functions

William Gould, CRC, FAX 310-393-7551

In writing `extrname` (*dm13*, above) I found myself needing various string manipulation functions. The commands below will also be installed when you install `extrname`:

```
replstr  "oldstr" "newstr" # varname [if exp] [in range]
replword "oldwrd" "newwrd" # varname [if exp] [in range]
trimblnk varname [if exp] [in range]
mixcase  varname [if exp] [in range]
exchstr varname varname [if exp] [in range]
splitstr newvar varname [character]
minlen   [str]# varlist
```

`replstr` replaces, in each observation, up to # occurrences of *oldstr* with *newstr*. `replword` replaces, in each observation, up to # occurrences of *oldwrd* with *newrd*. # may be 0, 1, 2, …, or `.` (missing value). If # is 0, no action is taken. If # is `.`, all occurrences are replaced.

`trimblnk` removes all leading and trailing blanks and removes all embedded multiple blanks.

`mixcase` changes *varname* to be mostly lower case with the beginning of each word capitalized.

`exchstr` exchanges the contents of two string variables.

`splitstr` parses *varname* on *character* (default being the space character), moves the first token into *newvar*, and eliminates the token from the original *varname*.

`minlen` recasts each variable in *varlist* to `str#` if `str#` is longer than how the variable is currently stored; otherwise, it takes no action.

Example: replstr

To change every '`(`' and '`)`' to '`[`' and '`]`' in variable `desc`:

```
. replstr "(" "[" . desc
. replstr ")" "]" . desc
```

To change only the first occurrence in each observation:

```
. replstr "(" "[" 1 desc
. replstr ")" "]" 1 desc
```

To change '`&`' to '`and`' with blanks around it:

```
. replstr "&" " and " . desc
```

The storage type of string variable `desc` (`str`#) will be made longer if necessary. See example of `trimblnk` below to now remove possible multiple blanks.

To remove all occurrences of the character '`#`' in `desc`:

```
. replstr "#" "" . desc
```

Example: replword

A word is a special case of string: a word either has blanks around it or is at the beginning or end of a string. Changing all occurrences of string "is" to "are" in "this is a test" results in "thare are a test"; changing the word "is" to "are" results in "this are a test".

To change all occurrences of the word "`Mr`" to "`Mr.`" in `name`

```
. replword "Mr" "Mr." . name
```

This will not change "`Mr.`" to "`Mr..`" because "`Mr.`" is not equivalent to the word "`Mr`".

Example: trimblnk

To remove all multiple blanks in variable `name`:

```
. trimblnk name
```

This is not equivalent to

```
. replstr  "  "   " "  .  name  /* there are 2 blanks in the 1st
                                    pair of quotes and one in the second */
```

because `replstr` is not recursive (a sequence of three blanks would become two after `replstr`).

To change '`&`' to "`and`" with blanks around it and then remove the (possibly) introduced multiple blanks in variable `desc`:

```
. replstr "&" " and " . desc
. trimblnk desc
```

Example: mixcase

To capitalize the first letter of each word and lowercase the rest in variable name:

```
. mixcase name
```

If the first observation of name contained "`MR. robert E. SMITH`", it would now contain "`Mr. Robert E. Smith`".

Example: exchstr

To exchange the contents of the two string variables `first` and `last` wherever variable `swap` is not zero:

```
. exchstr first last if swap~=0
```

The storage types (`str`#) of `first` and `last` will be changed if necessary.

Example: splitstr

You have a string variable `pname`; you wish to remove the first word (defined as the characters up to a blank) and put those characters in a new string variable called `first`:

```
. splitstr first pname " "
```

If the first observation of `pname` was "`Roger E. Smith`", the first observation of `first` now contains "`Roger`" and `pname` now contains "`E. Smith`".

Example: minlen

You have a string variable `ttl`. You are about to replace its contents with "`Doctor`" everywhere it contains "`Dr.`". This is occurring in a program you have written and you are not sure that string variable `ttl` is of sufficient width to hold the word "`Doctor`":

```
. minlen 6 ttl
. replace ttl="Doctor" if ttl=="Dr."
```

More generally, you have two string variables `ttl1` and `ttl2`. You are about to replace `ttl1` with `ttl2` everywhere variable `assign` is not zero. You want to be sure that `ttl1` is long enough to contain what might come from `ttl2` and, afterwards, you want to make the variable `ttl1` consume as little memory as possible:

```
. local stype : type ttl2
. minlen `stype´ ttl1
. replace ttl1 = ttl2 if assign~=0
. compress ttl1
```

ip4 Program debugging command

Sean Becketti, Editor, STB, FAX 913-888-6708

The syntax of `pause` is

`pause` { `on` | `off` | [*message*] }

If pause is on, a `pause` [*message*] displays *message* and temporarily suspends execution of the program, returning control to the keyboard. Execution of keyboard commands continues until you type `end` or `q`, at which time execution of the program resumes. Typing `BREAK` in pause mode (as opposed to pressing the Break key) also resumes program execution, but the Break signal is sent back to the calling program.

If pause is off, `pause` does nothing.

Pause is off by default. Type `pause on` to turn pause on. Type `pause off` to turn it back off.

Remarks

`pause` assists in debugging Stata programs. The line '`pause`' or '`pause` *message*' is placed in the program where problems are suspected (more than one `pause` may be placed in a program). For instance, you have a program that is not working properly. A piece of this program reads

```
gen `tmp´=exp(`1´)/`2´
summarize `tmp´
local mean=_result(3)
```

You think the error may be in the creation of `` `tmp´ ``. You change the program to read

```
gen `tmp´=exp(`1´)/`2´
pause Just created tmp        /* this line is new */
summarize `tmp´
local mean=_result(3)
```

Let's pretend your program is named `myprog`; interactively, you now type

```
. myprog
(output from your program appears)
```

That is, `pause` does nothing. It does nothing because pause is off and so `pause`s in your program are ignored. If you turn pause on:

```
. pause on
. myprog
(any output myprog creates up to the pause appears)
pause:  Just created tmp
-> . describe
         (output omitted )
-> . list
         (output omitted )
-> . end
execution resumes...
(remaining output from myprog appears)
```

The "`->`" is called the pause-mode prompt. You can give any Stata command. You can examine variables and, if you wish, even change them. If, while in pause mode, you wish to terminate execution of your program, you type '`BREAK`' (in capitals):

```
. myprog
(any output myprog creates up to the pause appears)
pause:  Just created tmp
-> . list
              (output omitted )
-> . BREAK
sending Break to calling program...
--Break--
r(1);
. _
```

The results are the same as if you pressed Break while your program were executing. If you press the Break key in pause mode (as opposed to typing `BREAK`), however, it means only that the execution of the command you have just given interactively is to be interrupted.

Notes

1. You may put many `pause`s in your programs.
2. By default, pause is off, so the `pause`s will not do anything. Even so, you should remove the pauses after your program is debugged because each execution of a do-nothing `pause` will slow your program slightly.
3. `pause` is implemented as an ado-file; this means the definitions of local macros in your program are unavailable to you. To see the value of local macros, display them in the `pause` message; for instance:

   ```
           pause Just created tmp, i=`i´
   ```

   Then, when the line is executed, you will see something like:

   ```
   pause:  Just created tmp, i=1
   -> . _
   ```

4. Remember, temporary variables (e.g., `tempvar tmp` ... `gen `tmp´=`...) are assigned real names such as `__00424` by Stata. Thus, in pause mode you want to examine `__00424` and not `tmp`. Generally, you can determine the real name of your temporary variables from `describe`'s output, but in the example above, it would be better had `pause` been invoked with

   ```
           pause Just created tmp, called `tmp´, i=`i´
   ```

   Then, when the line is executed, you will see something like:

   ```
   pause:  Just created tmp, called __00424, i=1
   -> . _
   ```

5. When giving commands that include double quotes, you may occasionally see the error message "`type mismatch`" but then the command will work properly:

   ```
   pause:  Just created tmp, called __00424, i=1
   -> . list if __00424=="male"
   type mismatch
   (output from request appears as if nothing is wrong)
   -> . _
   ```

   This is a problem in `pause` that I am still struggling to fix. In any case, you know the problem is not with what you typed and that what you typed executed correctly because the return code was not nonzero (no "`r(101)`" or some such message.)

os8 Stata and Lotus 123

Patrick Royston, Royal Postgraduate Medical School, London, FAX (011)-44-81-740 3119
EMAIL proyston@rpms.ac.uk
William Gould, CRC, FAX 310-393-7551

Below, we provide three Stata commands for DOS users to make going between Lotus$^{\text{tm}}$ 123 and Stata easier. The first command, `lotus`, saves the Stata data currently in memory in a 123 spreadsheet and invokes the Lotus program; when you exit Lotus, it reimports the data from the spreadsheet, thus turning Lotus into a spreadsheet editor for Stata. The remaining two commands, `limport` and `lexport`, make importing and exporting data from and to Lotus easier.

The syntax of the three commands are

```
lotus
limport filename [, clear|replace ]
lexport filename [, replace ]
```

These three commands make the following assumptions:

1. You own Lotus 123.
2. You have installed Lotus in `C:\123`.
3. You own Stat/Transfer (see [0] transfer).
4. You have installed Stat/Transfer in `C:\ST`.

Ways around assumptions 2 and 4 are detailed under the heading *Technical assumptions* below.

The lotus command

`lotus` temporarily saves the data in memory in a Lotus 123 `wk1` file using Stat/Transfer and enters 123 with the data loaded. When you exit Lotus 123, you are put back into Stata and the Lotus data set is reloaded (the conversion back to Stata format is automatic). The only thing that you, as a Lotus user, must do is to save the data under the same name it had when you entered Lotus (which will be `__123.wk1`) before exiting Lotus back to Stata. You do this by typing `/fs` followed by `r`. To exit Lotus, you type `/qy`.

The limport and lexport commands

`limport` converts the Lotus 123 file *filename*`.wk1` to a Stata-format file *filename*`.dta` and loads the file into Stata. `lexport` converts the Stata data in memory into a Lotus 123 file *filename*`.wk1`.

The `clear` and `replace` options mean the same thing. `replace`, used with `limport`, means *filename*`.dta` may be replaced if it already exists. Used with `lexport`, it means *filename*`.wk1` may be replaced if it already exists.

Examples: lotus

You are using Stata and would now like to look at your data and, perhaps, change it using Lotus 123. You type

```
. lotus
```

You are now in Lotus with the data loaded. If you make changes to the data and want them automatically exported back to your Stata session, you type `/fs` followed by `r`. Whether you change the data or not, you type `/qy` to exit Lotus and return to Stata.

There is one problem with `lotus` of which you should be aware: variable types (`int`s, `long`s, `float`s, etc.) may be changed.

Examples: limport

You are using Stata and want to load the data saved in a Lotus 123 spreadsheet. In Lotus, you previously saved the data as `myfile.wk1`. You type

```
. limport myfile
```

This command not only loads the data, it creates `myfile.dta` so that, in the future, you can simply type '`use myfile`'.

Assume you later update your spreadsheet and, in Lotus, save it again. In Stata, you attempt to re-import the data:

```
. limport myfile
file myfile.dta already exists
r(602);
```

To re-import the data, `limport` must recreate the Stata version of the data and that version already exists. To have `limport` replace the existing file anyway, type

```
. limport myfile, replace
```

This is equivalent to typing

```
. erase myfile.dta
. limport myfile
```

Now assume the Lotus data you wish to import is saved in a different directory, say `C:\123`. To import the data, you type

```
. limport c:\123\myfile
```

The newly created `myfile.dta` will also be saved in that directory, so in the future, to reload the data, you must type

```
. use c:\123\myfile
```

If you do not want to keep the Stata-format data there, you might type

```
. erase c:\123\myfile.dta
. save myfile
```

Examples: lexport

You are using Stata and wish to save the data in memory for subsequent use in Lotus. If you type

```
. lexport myfile
```

`myfile.wk1` will be created in the current directory. If you have done this before with the same filename, rather than `myfile.wk1` being created, you will see

```
. lexport myfile
file myfile.wk1 already exists
r(602);
```

If you are willing to replace that file, you can type

```
. lexport myfile, replace
```

If you want to save `myfile.wk1` in `C:\123`, you can type

```
. lexport c:\123\myfile
```

Technical assumptions

As we said at the outset, `lotus`, `limport`, and `lexport` assume you own both Lotus 123 and Stat/Transfer and that you have installed Lotus in `C:\123` and have installed Stat/Transfer in `C:\ST`. Below we will cover how to vary the two installation-location assumptions and even how these commands can be varied to use a program other than Stat/Transfer to perform the translation.

`lotus` needs to invoke Lotus 123; it assumes the command is `c:\123\123`. If you have installed Lotus someplace other than `C:\123`, you can define the global macro `$S_LOTUS` with the full identity of the Lotus 123 program. For example, if Lotus were installed in `D:\123`, you would type

```
. mac def S_LOTUS "d:\123\123"
```

To automate this definition every time Stata is invoked, see the last technical note in [1] start/stop.

The remaining customization concerns `limport` and `lexport` which, even if you do not use them directly, are used indirectly by `lotus`. By default, `limport` and `lexport` assume Stat/Transfer has been installed in `C:\ST`. To change `fn.wk1` into `fn.dta`, `limport` issues the command

```
c:\st\transfer fn.wk1 fn.dta > nul
```

to DOS. To change `fn.dta` into `fn.wk1`, `lexport` issues the command:

```
c:\st\transfer fn.dta fn.wk1 > nul
```

Four global macros control the command actually issued: `$S_IMPO`, `$S_IMPO2`, `$S_EXPO`, and `$S_EXPO2`. The actual commands issued by `limport` and `lexport` are

```
$S_IMPO fn.wk1 fn.dta $S_IMPO2
$S_EXPO fn.dta fn.wk1 $S_EXPO2
```

If the four macros are not defined, `limport` (`lexport`) pretends `$S_IMPO` and `$S_EXPO` contain `"c:\st\transfer"` and that `$S_IMPO2` and `$S_EXPO2` contain `"> nul"`. Thus, `limport`'s

```
$S_IMPO fn.wk1 fn.dta $S_IMPO2
```

is the same as

```
c:\st\transfer fn.wk1 fn.dta > nul
```

and similarly for `lexport`.

Assume Stat/Transfer is installed not in `C:\ST` but in `D:\ST`. Typing

```
. mac def S_EXPO "d:\st\transfer"
. mac def S_IMPO "d:\st\transfer"
```

will make `limport` and `lexport` work anyway. When `limport`, for instance, translates a data set, it will issue the command:

```
$S_IMPO fn.wk1 fn.dta $S_IMPO2
```

Since `$S_IMPO` has been redefined, this command will be:

```
d:\st\transfer fn.wk1 fn.dta > nul
```

(If you have Stat/Transfer installed someplace other than `C:\ST`, you will want to automate the definition of these macros. See the last technical note in [1] start/stop for a way to do this.)

Now assume that you do not own Stat/Transfer but that you own some other file translation utility which has the capability to convert Lotus to Stata data sets and vice-versa. To make the problem difficult, we will assume that this other translation utility has the command syntax:

```
xlate lotus=fn.wk1 to stata=fn.dta
```

To fix `limport`, you first write a `.BAT` file which we will call `l_to_s.bat`:

```
@ECHO OFF
xlate lotus=%1 to stata=%2
```

In Stata, you reset `$S_IMPO`:

```
. mac def S_IMPO "l_to_s"
```

Now, typing `limport` will invoke `l_to_s` which, in turn, will invoke `xlate`.

os9 Printing Stata log files

Sean Becketti, Editor, STB, FAX 913-888-6708

`printlog.ado` formats and prints Stata log files.

The syntax of `printlog` is

`printlog` *list-of-log-files*

Discussion

Stata provides a utility program called `fsl`—pronounced *fizzle* and short for *Format Stata Log*—to format Stata log files in a readable format. `fsl` is similar to `gphdot` and `gphpen` in that `fsl` is an external program rather than a Stata command. The `fsl` command can be issued during a Stata session by prefixing it with the Stata shell command, for example,

```
. ! fsl session
```

will format the file `session.log` using the default layout and save the formatted version as `session.prn`. A second shell command can now be given to print `session.prn`, then a third command can be given to erase `session.prn`.

`printlog` automates this process. Typing

```
. printlog session
```

will format `session.log`, print `session.prn`, then erase `session.prn`. Multiple log files can be specified:

```
. printlog session1 session2 session3
```

will complete the same process for all three log files.

`printlog` is a simple program, but, because it formats and prints files on your specific printer, it needs to be customized for your computer before it can be used. The entire `printlog` program follows:

```
program define printlog
        version 3.0
        while ("`1´" != "") {
                local fn "`1´"
                local j = index("`fn´",".log")
                if ((`j´)>0 & (`j´==length("`fn´")-3)) {
                        local fn=substr("`fn´",1,(`j´-1))
}
                !fsl -lhp `fn´
                !lp `fn´.prn
                erase `fn´.prn
                mac shift
}
end
```

You need to modify the two lines with the shell command prefix (`!`) to make `printlog` work for you, that is, you need to modify the lines

```
!fsl -lhp `fn´
!lp `fn´.prn
```

I use the Unix version of Stata, so my version of the `fsl` command uses the Unix option switch (`-`). I have an HP LaserJet, so I use the (`-lhp`) layout option. Change that option to match your printer (or delete it to use the default layout) and add any other options you use regularly. (See [5u] fsl in the *Stata Reference Manual* for information on `fsl` options.) My print command is `lp`, the Unix command for sending files to the printer. Change that line to the print command for your computer. No other changes are needed.

Two technical notes: First, if you use a word processor, such as WordPerfect, to make the changes, remember to save your version of `printlog` as an ASCII file and not in the special format normally used by your word processor. Second, if you use a print spooler, such as the DOS `print` command, the `.prn` file may be erased before the spooler is ready to print it. If this is a problem for you, delete the line

```
erase `fn´.prn
```

and the `.prn` will not be erased.

sg5.1 Correlation coefficients with significance levels

Sean Becketti, Editor, STB, FAX 913-888-6708

The `corrprob` command (*sg5*) displays the correlation between two variables along with a normal approximation to the marginal significance level of the test that the correlation is zero. `corrprob` offers three varieties of correlation coefficients: ordinary (i.e., Pearson product-moment), Spearman rank, and Kendall's τ_β. Unfortunately, as several readers have pointed out, the support routines to calculate the Spearman and τ_β correlation coefficients were omitted from the STB-5 diskette.

I have chosen to replace `corrprob` rather than to repair it. After a period of using `corrprob`, I found it more convenient to break this single command into three different commands, one for each type of correlation coefficient. The command to calculate Kendall's τ_β, `ktau`, has already been incorporated into Stata ([5s] spearman). The `spearman` command provided in Stata does not display the marginal significance level, so I have provided a replacement. The two new commands on this diskette, then, are `pearson` and `spear`. They use the same syntax and store the same results as `ktau`.

sg11.2 Calculation of quantile regression standard errors

William Rogers, CRC, FAX 310-393-7551

Stata's `qreg` command estimates quantile regression and presents standard errors for the coefficients. In [5s] qreg it states that the standard errors are obtained using a method suggested by Koenker and Bassett (1982), but the explanation is not complete. In Rogers (1992), I attempted to complete the explanation and then went on to explore the robustness of these standard errors (finding that, while they appear adequate in the case of homoscedastic errors, they are probably understated if the errors are heteroscedastic). I suggested the use of bootstrap standard errors and Gould (1992) provided one such routine.

It has been pointed out to me that the documentation of the method "suggested by" Koenker and Bassett has still not been fully described. In particular, I have provided no clue as to how $f_{\text{errors}}(0)$ is calculated. The answer is that we use an estimate of our own devising based on a variation of the k-nearest-neighbor idea which I will now describe.

The variance–covariance matrix is estimated by $\mathbf{R}_2^{-1}\mathbf{R}_1\mathbf{R}_2^{-1}$. $\mathbf{R}_1$ is estimated as $\mathbf{X}'\mathbf{W}\mathbf{W}'\mathbf{X}$, where $\mathbf{W}$ is a $n \times n$

diagonal matrix with elements

$$W_{ii} = \begin{cases} q/f_{\text{errors}}(0) & \text{if } r > 0 \\ (1-q)/f_{\text{errors}}(0) & \text{if } r < 0 \\ 0 & \text{otherwise} \end{cases}$$

and $\mathbf{R}_2$ is the design matrix $\mathbf{X}'\mathbf{X}$. This is derived from formula 3.11 in Koenker and Bassett, although their notation is much different. $f_{\text{errors}}()$ refers to the density of the true residuals. There are many things that Koenker and Bassett leave unspecified, including how one should obtain a density estimate for the errors in real data. It is at this point that we offer our contribution.

We first sort the residuals and locate the observation in the residuals corresponding to the quantile in question, taking into account weights if they are applied. We then calculate w_n, the square root of the sum of the weights. Unweighted data is equivalent to weighted data where each observation has weight 1, resulting in $w_n = \sqrt{n}$. For analytically weighted data, the weights are rescaled so that the sum of the weights is the sum of the observations, resulting in $\sqrt{n}$ again. For frequency weighted data, w_n literally is the square of the sum of the weights.

We locate the closest observation in each direction such that the sum of weights for all closer observations is w_n. If we run off the end of the dataset, we stop. We calculate w_s, the sum of weights for all observations in this middle space. Typically, w_s is slightly greater than w_n.

The residuals obtained after quantile regression have the property that if there are k parameters, then exactly k of them must be zero. Thus, we calculate an adjusted weight $w_a = w_s - k$. The density estimate is the distance spanned by these observations divided by w_a. Because the distance spanned by this mechanism converges toward zero, this estimate of the density converges in probability to the true density.

References

Gould, W. 1992. Quantile regression and bootstrapped standard errors. *Stata Technical Bulletin* 9: 19–21.

Koenker, R. and G. Bassett, Jr. 1982. Robust tests for heteroscedasticity based on regression quantiles. *Econometrica* 50: 43–61.

Rogers, W. H. 1992. Quantile regression standard errors. *Stata Technical Bulletin* 9: 16–19.

sg17 | Regression standard errors in clustered samples

William Rogers, CRC, FAX 310-393-7551

Stata's **hreg**, **hlogit** and **hprobit** commands estimate regression, maximum-likelihood logit, and maximum-likelihood probit models based on Huber's (1967) formula for individual-level data and they produce consistent standard errors even if there is heteroscedasticity, clustered sampling, or the data is weighted. The description of this in [5s] hreg might lead one to believe that Huber originally considered clustered data, but that is not true. I developed this approach to deal with cluster sampling problems in the RAND Health Insurance Experiment in the early 1980s (Rogers 1983; Rogers and Hanley 1982; Brook, et al. 1983). What is true is that with one simple assumption, the framework proposed by Huber can be applied to produce the answer we propose. That assumption is that the clusters are drawn as a simple random sample from some population. The observations must be obtained within each cluster by some repeatable procedure.

Ordinary linear regression applied to the observations of a cluster is a nonstandard maximum-likelihood estimate; that is, a maximum of the "wrong" likelihood, given this sampling procedure. This is an important special case of the more general problem that Huber's article addresses.

The special case can be obtained by recognizing that a cluster can play the same role as an observation. For instance, the Huber regularity conditions require that the importance of any one observation vanishes as the number of observations becomes infinite. Huber's method is not good when there are only a few observations. In this special case, the Huber regularity conditions require that the importance of any one *cluster* vanishes as the number of clusters (and therefore observations) becomes infinite. Thus, Huber's reinterpreted method is not good when there are only a few clusters.

To apply Huber' formula directly, one would want to calculate a score value and a Hessian value for each cluster. This is described in [5s] huber. Although elegant, this application of Huber's result does not provide much insight into why the idea works. For the case of linear regression, the matrix algebra provides more insight.

Let p be the number of parameters and n the number of observations. Let $\mathbf{X}$ be the $n \times p$ design matrix and $\mathbf{y}$ be the $n \times 1$ vector of dependent values. The ordinary linear regression estimates are $(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$. The variance of this estimate is

$$\text{var}(\mathbf{b}) = \text{E}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\mathbf{y} - \text{E}\mathbf{y})(\mathbf{y} - \text{E}\mathbf{y})'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$$

If we adopt the usual practice of replacing unknown errors with residuals, the inner matrix is an $n \times n$ rank 1 matrix, which is not very helpful. The original solution to this problem is to assume that the $\mathbf{X}$ matrix is fixed and move the expectation inside, and take advantage of the independent and identically distributed assumption to assert that $\mathrm{E}(\mathbf{y}-\mathrm{E}\mathbf{y})(\mathbf{y}-\mathrm{E}\mathbf{y})' = \sigma^2\mathbf{I}$. All of the off-diagonal terms are zero, and all of the diagonal terms are the same. The estimate of σ^2 is obtained by substituting residuals for the errors $(\mathbf{y}-\mathrm{E}\mathbf{y})$. After reduction, the familiar variance estimate $(\mathbf{X}'\mathbf{X})^{-1}\sigma^2$ is obtained.

In the revised solution, we do not assume that the diagonal terms are identical (White 1980). Also, we do not assume off-diagonal terms are zero unless they come from different clusters. Observations from different clusters are independent, so their off-diagonal elements must be zero. We simply let all these nonzero terms be represented by the appropriate products of the residuals.

Ordinarily, estimating n parameters, or even more with clustering, would be a bad idea. However, with pre- and post-multiplication by $\mathbf{X}$, a convergent averaging effect is obtained provided that no cluster is too large.

If weights are present, these weights appear in the equation and are treated as part of the observations. The variance estimate now becomes:

$$\mathrm{var}(\mathbf{b}) = \mathrm{E}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}(\mathbf{y}-\mathrm{E}\mathbf{y})(\mathbf{y}-\mathrm{E}\mathbf{y})'\mathbf{W}\mathbf{X}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}$$

Since linear regression is not the maximum-likelihood answer, most statisticians would presume that it does not give an answer we would want. However, it is worth pointing out that the "wrong" answer given by linear regression is the answer that would be given if the entire population of clusters were sampled in the manner prescribed. In some applications this is the desired answer, and other answers converge to something else. In each case, the user must decide if the linear regression answer is wanted or not, on a theoretical basis. For example, if we sample families and then take one family member (without weighting), family members in large families will be undersampled.

Two advantages of this framework over other approaches to the cluster sampled problem are (1) that the nature of the within-cluster dependence does not have to be specified in any way, and (2) familiar estimation methods can be used. Since the method can be thought of as "correcting" linear regression, the user is free to conduct other types of sensitivity analysis in parallel. For example, he might also consider a sample selection model using the linear regression results as a common point of comparison.

Although the mathematics guarantees that the large sample behavior of this estimate will be good, what about small-sample behavior? A few Monte-Carlo experiments will give a good idea of what is going on. Simple experiments will suffice since Huber covariance estimates respond to affine transformations of the $\mathbf{X}$ matrix or $\mathbf{y}$ observation vector just as regular covariance estimates do.

Experiment 1

First, we verify that in large samples the answers obtained by the Huber algorithm is okay. We draw 2,500 observations clustered in groups of 5 as follows:

```
. clear
. set obs 2500
. gen x = (_n-1250.5)/1249.5
. gen y = invnorm(uniform())
. gen u = uniform()
. gen g = int((_n-1)/5)
```

We then run 1,000 examples of this and examine the collective results.

The known covariance matrix for the ordinary linear regression coefficients is

$$\begin{pmatrix} 0.0004000 & 0 \\ 0 & 0.001199 \end{pmatrix}$$

For the standard regression estimates, these covariances are obtained up to a multiplier that is distributed $\chi^2(2498)/2498$, which has expectation 1 and variance $2/2498$.

We will look at two Huber estimates. The first Huber estimate assumes no clustering and is equivalent to White's method. The second Huber estimate assumes clustering in 500 clusters of 5 observations each. Each cluster contains nearby values of x.

There are two things we would want to know. First, do these variance estimation procedures estimate the right variance on the average, and second, how well do the estimates reflect known population behavior?

		Huber/White method	
	Usual formula	Unclustered	Clustered
Average estimated variance of the coefficient			
var(_cons) $\times 10^4$	4.000	3.995	3.979
var(_b[x]) $\times 10^4$	11.99	11.98	11.93
correlation	0.	0.	0.
RMS error of the variance estimate			
var(_cons) $\times 10^8$	32	115.	2439.
var(_b[x]) $\times 10^8$	96	4973.	9730.
correlation	0	.028	.063
Percent of cases marked as significant			
var(_cons)	5.0	4.3	4.3
var(_b[x])	5.0	4.7	4.9

All three methods produce essentially the same variance estimates. There is no cost here for added robustness. The unclustered and clustered Huber estimates of the variance are more variable, but it does not matter. Asymptotics have taken hold.

Experiment 2

Next, we verify the desirable properties of the Huber estimate for clustered data in large samples. We draw 2,500 observations clustered in groups of 5 as follows:

```
. clear
. set obs 2500
. gen x = (_n-1250.5)/1249.5
. gen y = invnorm(uniform())
. gen yy = invnorm(uniform())
. gen u = uniform()
. gen g = int((_n-1)/5)
. sort g
. qui by g: replace y = sqrt(.8)*y + sqrt(.2)*yy[_N] if _n < _N
```

We then run 1,000 examples of this and examine the collective results. The intracluster correlation is 0.2, and the group size is 5, so the design effect DEFF (see [5s] deff) for this problem is 1.8, meaning that the F-ratios of the standard problem are too high by a multiple of 1.8.

The results are

		Huber/White method	
	Usual formula	Unclustered	Clustered
Average estimated variance of the coefficient			
var(_cons) $\times 10^4$	4.000	3.986	5.883
var(_b[x]) $\times 10^4$	11.99	11.93	17.53
correlation	0.	0.	0.
RMS error of the variance estimate			
var(_cons) $\times 10^8$	32	32.	4.
var(_b[x]) $\times 10^8$	96	96.	14.
correlation	0	.030	.064
Percent of cases marked as significant			
var(_cons)	10.3	10.3	5.0
var(_b[x])	10.5	10.7	5.3

The usual estimates did not change, and neither did the Huber results uncorrected for clustering. However, they are no longer correct. The Huber estimates with correction for clustering got the right answers. The impact on the Type I errors is remarkable.

It is noteworthy that the DEFF approach to this problem did not predict the relative loss of precision. Evidently, this problem—although seemingly simple—is too hard for DEFF. The Huber answers did a much better job.

Experiment 3

Now we will try a small-sample model. We will draw 25 observations in a manner similar to Experiment 1. The results favor the usual estimates.

		Huber/White method	
	Usual formula	Unclustered	Clustered
Average estimated variance of the coefficient			
var(_cons) $\times 10^4$	.040	.036	.023
var(_b[x]) $\times 10^4$	.111	.096	.051
correlation	0.	0.	0.
RMS error of the variance estimate			
var(_cons) $\times 10^4$	35	111.	245.
var(_b[x]) $\times 10^4$	96	396.	751.
correlation	0	.265	.522
Percent of cases marked as significant			
var(_cons)	5.0	5.7	19.6
var(_b[x])	5.0	6.4	26.3

The Huber standard errors are notably smaller then the usual ones. Once again, we see that the Huber covariance estimates are more variable, the clustered ones more than the unclustered. Since the Huber estimates are too low, they are too likely to declare a result significant.

The reason for this is that there are mathematical constraints on the residuals. Formally, they need to add to zero and be orthogonal to each x. In the current problem, it is as if we were doing the regression on only 5 values; the sum of squared residuals would be only $3/5$ of the actual variance for those sums. In fact, this is what we observe for the intercept (.023 is about $3/5$ of .040).

A correction can be worked out as follows. For each observation, the variance of the residual is $\sigma^2(1-h_i)$, where $h_i = \mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}'_i$. For two observations, the covariance of the residuals is $\sigma^2(-\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}'_j)$.

Thus, when the IID model holds, but a covariance matrix is estimated via Huber's method, the underestimation bias is

$$-(\mathbf{X}'\mathbf{X})^{-1}\Big(\sum_{j=1}^{C}\sum_{i=1}^{N_j}\sum_{m=1}^{N_j}\mathbf{x}'_{ji}\mathbf{x}_{ji}\sigma^2(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}'_{jm}\mathbf{x}_{jm}\Big)(\mathbf{X}'\mathbf{X})^{-1}$$

This formula cannot be computed by Stata, but a useful bound can be obtained by noting that the interior diagonal terms ($i=m$) are h_i; the off-diagonal terms will be less than $\sqrt{h_i h_m}$; and so a simple approximation to the result would be to add to each residual $\sigma\sqrt{h_i}$ before applying _huber.

I have modified hreg, creating hreg2 to calculate this quantity. A further useful observation is that we can bound the asymptotic convergence rate for the Huber estimates. That bound is

$$O\Big(\sum_{j=1}^{C}\frac{N_j^2}{N^2}\Big)$$

So, if no cluster is larger than 5% or so of the total sample, the standard errors will not be too far off because each term will be off by less than 1 in 400.

Experiment 4

Experiment 4 is a repeat of Experiment 3 except that I used the `hreg2` command included on the STB diskette:

		Huber/White method	
	Usual formula	Unclustered	Clustered
Average estimated variance of the coefficient			
var(`_cons`) $\times 10^4$	.040	.039	.039
var(`_b[x]`) $\times 10^4$	.111	.108	.108
correlation	0.	0.	0.
RMS error of the variance estimate			
var(`_cons`) $\times 10^4$	35	114.	207.
var(`_b[x]`) $\times 10^4$	96	436.	872.
correlation	0	.242	.310
Percent of cases marked as significant			
var(`_cons`)	5.0	4.7	7.6
var(`_b[x]`)	5.0	5.2	11.4

Much better! This does a reasonable job of correcting the answer for this problem, but may be an overcorrection for variables where there is not a lot of intracluster correlation.

A further problem arises now that the variance is correct on average. In some sense we only have 5 observations—one for each cluster—so perhaps the *t*-statistic ought to have 5 degrees of freedom instead of 23. Recalculating the percent of cases in the clustered case using 5 would result in the last part of the table reading:

		Huber/White method	
	Usual formula	Unclustered	Clustered
Percent of cases marked as significant			
var(`_cons`)	5.0	4.7	3.4
var(`_b[x]`)	5.0	5.2	6.3

This further adjustment works well in this case, bringing the Type I error probabilities back into line.

Conclusions

The formulas above imply that the bias exists in proportion to the square of cluster size in relation to sample size. As long as the largest cluster is 5 percent or less of the sample, this bias should be negligible.

In the case where the 5-percent rule does not hold, an adjustment is possible. On the STB diskette, I provide `hreg2.ado` as an alternative to `hreg` that makes this adjustment.

I have also shown a formula which in principle corrects this bias in all cases. However, this formula is not easily implemented at present.

References

Brook, R. H., J. E. Ware, W. H. Rogers, et al. 1983. Does free care improve adults' health? *New England Journal of Medicine* 309: 1426–1434.

Huber, P. J. 1967. The behavior of maximum likelihood estimates under non-standard conditions. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 1: 221–233.

Rogers, W. H. 1983. Analyzing complex survey data. Santa Monica, CA: Rand Corporation memorandum.

Rogers, W. H. and J. Hanley. 1982. Weibull regression and hazard estimation. *SAS Users Group International Proceedings.*

White, H. 1980. A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity. *Econometrica* 48: 817–830.

sqv8 Interpreting multinomial logistic regression

Lawrence C. Hamilton, Dept. of Sociology, Univ. of New Hampshire, l_hamilton@unhh.unh.edu
Carole L. Seyfrit, Dept. of Sociology and Criminal Justice, Old Dominion Univ., cls100f@oduvm.cc.odu.edu

Social and biological scientists widely use logit (logistic) regression to model binary dependent variables such as move/stay or live/die. Techniques for modeling multiple-category dependent variables are a relatively recent development, however. Asking Stata to perform multinomial logistic regression is easy; given a Y with three or more unordered categories, predicted by X_1 and X_2, you type '`mlogit` Y X_1 X_2'. If Y has only two categories, `mlogit` fits the same model as `logit` or `logistic`. Otherwise, though, an `mlogit` model is more complex. This insert, a sort of "beginners guide to multinomial logit" written while stormbound at the Nullagvik Hotel, illustrates several ways to interpret `mlogit` output.

Data set `arctic.dta` contains information from a recent survey of high school students in Alaska's Northwest Arctic.[1]

```
. use arctic
(Hamilton/Seyfrit:Arctic student)
. describe

Contains data from arctic.dta
  Obs:   259 (max= 10484)                  Hamilton/Seyfrit:Arctic student
 Vars:     3 (max=    99)
Width:     6 (max=   200)
  1. migrate      byte   %8.0g    migrate  Expect to live most of life
  2. ties         float  %9.0g             Social ties to community
  3. kotz         byte   %8.0g    kotz     Attend Kotzebue High School
Sorted by:
```

Variable `migrate` indicates where students say they expect to live most of the rest of their lives: the same area (Northwest Arctic), elsewhere in Alaska, or out of Alaska.

```
. tabulate migrate, plot
   Expect to|
live most of|
        life|      Freq.
------------+------------+-----------------------------------------------------
       same |         92 |****************************************
   other AK |        120 |*****************************************************
   leave AK |         47 |********************
------------+------------+-----------------------------------------------------
      Total |        259
```

Kotzebue (population near 3,000) is the Northwest Arctic's regional hub and largest city. More than a third of these students attend Kotzebue High School; the rest attend smaller schools in bush villages of 200–600 people. The relatively cosmopolitan Kotzebue students less often expect to stay where they are, and lean more towards leaving the state:

```
. tabulate migrate kotz, chi2

  Expect to| Attend Kotzebue High School
  live most|
    of life|     other   Kotzebue |     Total
-----------+----------------------+----------
      same |        75         17 |        92
  other AK |        80         40 |       120
  leave AK |        11         36 |        47
-----------+----------------------+----------
      Total|       166         93 |       259

          Pearson chi2(2) =  46.2992   Pr = 0.000
```

`mlogit` can replicate this simple analysis (though its likelihood-ratio chi-square need not exactly equal `tabulate`'s usual Pearson chi-square):

```
. mlogit migrate kotz, base(1) nolog rrr

Multinomial regression                                  Number of obs =    259
                                                        chi2(2)       =  46.23
                                                        Prob > chi2   = 0.0000
Log Likelihood = -244.64465                             Pseudo R2     = 0.0863

------------------------------------------------------------------------------
 migrate |        RRR   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
other AK |
    kotz |   2.205882   .7304664      2.389   0.018       1.149125    4.234454
```

```
---------+--------------------------------------------------------------------
leave AK |
    kotz |   14.4385   6.307555      6.112   0.000       6.107956    34.13095
------------------------------------------------------------------------------
(Outcome migrate==same is the comparison group)
```

`base(1)` specifies that category 1 of Y (`migrate` = `"same"`) is the base category for comparison. `nolog` suppresses printing of the iteration log. The `rrr` option instructs `mlogit` to show relative risk ratios, which resemble the odds ratios given by `logistic` (instead of the coefficients given by `logit`).

Referring back to the tabulate output, we can calculate that among Kotzebue students the odds favoring “leave Alaska” over “stay in the same area” are

$$\text{P(leave AK)}/\text{P(same)} = (36/93)/(17/93) = 2.1176471$$

Among other students the odds favoring “leave Alaska” over “same area” are

$$\text{P(leave AK)}/P(\text{same}) = (11/166)/(75/166) = .1466667$$

Thus the odds favoring “leave Alaska” over “same area” are

$$2.1176471/.1466667 = 14.4385$$

times higher for Kotzebue students than for others. This multiplier, a ratio of two odds, equals the relative risk ratio (14.4385) displayed by `mlogit`.

In general, the relative risk ratio for category j of Y, and predictor X_k, equals the amount by which predicted odds favoring $Y = j$ (compared with $Y = \text{base}$) are multiplied, per 1-unit increase in X_k, other things being equal. In other words, the relative risk ratio rrr_{jk} is a multiplier such that, if all X variables except X_k stay the same:

$$\text{rrr}_{jk} \times \frac{\text{P}(Y = j|X_k)}{\text{P}(Y = \text{base}|X_k)} = \frac{\text{P}(Y = j|X_k + 1)}{\text{P}(Y = \text{base}|X_k + 1)}$$

`ties` is a continuous scale indicating the strength of students’ social ties to family and community.[2] Including `ties` as a second predictor:

```
. mlogit migrate kotz ties, rrr nolog base(1)

Multinomial regression                              Number of obs =     259
                                                    chi2(4)       =   91.96
                                                    Prob > chi2   = 0.0000
Log Likelihood = -221.77969                         Pseudo R2     = 0.1717

------------------------------------------------------------------------------
 migrate |        RRR   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
other AK |
    kotz |   2.214184   .7724996      2.278   0.024       1.113816    4.401636
    ties |   .4802486   .0799184     -4.407   0.000       .3460481    .6664932
---------+--------------------------------------------------------------------
leave AK |
    kotz |   14.84604   7.146824      5.604   0.000       5.752758    38.31291
    ties |    .230262    .059085     -5.723   0.000       .1389168    .3816715
------------------------------------------------------------------------------
(Outcome migrate==same is the comparison group)
```

Asymptotic t tests here indicate that the four relative risk ratios, describing two X variables’ effects, all differ significantly from 1.0. If a Y variable has J categories, then `mlogit` describes the effect of each predictor with $J - 1$ relative risk ratios or coefficients, and hence also prints $J - 1$ t tests—evaluating two or more separate null hypotheses for each predictor. Likelihood-ratio tests evaluate the overall effect of each predictor. First, save as “`0`” the results from the full model:

```
. lrtest, saving(0)
```

Then estimate a simpler model with one X variable omitted, and perform a likelihood-ratio test. For example, to test the effect of `ties`:

```
. quietly mlogit migrate kotz

. lrtest
Mlogit:  likelihood-ratio test                   chi2(2)     =      45.73
                                                 Prob > chi2 =     0.0000
```

To then test the effect of kotz:

```
. quietly mlogit migrate ties

. lrtest
Mlogit:  likelihood-ratio test                          chi2(2)     =      39.05
                                                        Prob > chi2 =     0.0000
```

Caution: If our data contained missing values, the three mlogit commands just shown might have analyzed three overlapping subsets of observations: the full model would use only observations with nonmissing migrate, kotz and ties values; the ties-omitted model would bring back in any observations missing only ties values; and the kotz-omitted model would bring back observations missing only kotz values. When this happens, Stata will say "Warning: observations differ" but still present the likelihood-ratio test. In this case, the test is invalid. Analysts must either screen observations with if qualifiers attached to modeling commands, such as

```
. mlogit migrate kotz ties, rrr nolog base(1)
. lrtest, saving(0)
. quietly mlogit migrate kotz if ties!=.
. lrtest
. quietly mlogit migrate ties if kotz!=.
. lrtest
```

or simply drop all observations having missing values before proceeding:

```
. drop if migrate==. | kotz==. | ties==.
. mlogit ...
(etc.)
```

Data set arctic.dta contains no missing values because we already dropped them.

Both kotz and ties significantly predict migrate. What else can we say from this output? To interpret specific effects, recall that migrate = "same" is the base category. The relative risk ratios tell us that:

> Odds that a student expects migration to elsewhere in Alaska rather than staying in the same area are 2.21 times greater (increase about 121%) among Kotzebue High School students (kotz = 1), adjusting for social ties to community.
>
> Odds that a student expects to leave Alaska rather than stay in the same area are 14.85 times greater (increase about 1385%) among Kotzebue High School students (kotz = 1), adjusting for social ties to community.
>
> Odds that a student expects migration to elsewhere in Alaska rather than staying are multiplied by .48 (decrease about 52%) with each 1-unit (since ties is standardized, its units equal standard deviations) increase in social ties, controlling for Kotzebue/village schools.
>
> Odds that a student expects to leave Alaska rather than staying are multiplied by .23 (decrease about 77%) with each 1-unit increase in social ties, controlling for Kotzebue/village schools.

predict can calculate predicted probabilities from mlogit. The outcome(*#*) option specifies for which Y category we want probabilities. For example, to get predicted probabilities that migrate = "leave AK" (category 3):

```
. quietly mlogit migrate kotz ties
. predict PleaveAK, outcome(3)
. label variable PleaveAK "P(migrate=3 | kotz, ties)"
```

Tabulating predicted probabilities for each value of the dependent variable shows how the model fits:

```
. tab migrate, summ(PleaveAK)
   Expect to| Summary of P(migrate=3 | kotz, ties)
live most of|
        life|        Mean   Std. Dev.       Freq.
------------+-----------------------------------
       same |   .08112671   .10995358          92
   other AK |   .17702251   .18187067         120
   leave AK |   .38922644   .20137434          47
------------+-----------------------------------
      Total |   .18146718   .19548242         259
```

A minority of these students ($47/259$ = 18%) expect to leave Alaska. The model averages only a .39 probability of leaving Alaska even for those who actually chose this response—reflecting the fact that although our predictors have significant effects, most variation in migration plans remains unexplained (pseudo $R^2 = 0.1717$).

Conditional effect plots help to visualize what a model implies regarding continuous predictors. We can draw them using estimated coefficients (not risk ratios) to calculate predicted probabilities:

```
. mlogit migrate kotz ties, base(1) nolog
Multinomial regression                                  Number of obs =    259
                                                        chi2(4)       =  91.96
                                                        Prob > chi2   = 0.0000
Log Likelihood = -221.77969                             Pseudo R2     = 0.1717
------------------------------------------------------------------------------
 migrate |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
other AK |
    kotz |    .794884   .3488868      2.278   0.024       .1077917    1.481976
    ties |  -.7334513   .1664104     -4.407   0.000      -1.061177   -.4057253
   _cons |    .206402   .1728053      1.194   0.233      -.1339182    .5467221
---------+--------------------------------------------------------------------
leave AK |
    kotz |   2.697733   .4813959      5.604   0.000       1.749679    3.645787
    ties |  -1.468537   .2565991     -5.723   0.000       -1.97388    -.963195
   _cons |  -2.115025   .3758163     -5.628   0.000      -2.855152   -1.374898
------------------------------------------------------------------------------
(Outcome migrate==same is the comparison group)
```

`L2villag` represents the predicted log odds of `migrate` = 2 (other Alaska) over `migrate` = 1 (same area) for village students. `L3kotz` is the predicted log odds of `migrate` = 3 (leave Alaska) over `migrate` = 1 for Kotzebue students, and so forth:

```
. generate L2villag=.206402+.794884*0-.7334513*ties
. generate L2kotz=.206402+.794884*1-.7334513*ties
. generate L3villag=-2.115025+2.697733*0-1.468537*ties
. generate L3kotz=-2.115025+2.697733*1-1.468537*ties
```

From these logits we calculate predicted probabilities:

```
. generate P1villag=1/(1+exp(L2villag)+exp(L3villag))
. label variable P1villag "same area"
. generate P2villag=exp(L2villag)/(1+exp(L2villag)+exp(L3villag))
. label variable P2villag "other Alaska"
. generate P3villag=exp(L3villag)/(1+exp(L2villag)+exp(L3villag))
. label variable P3villag "leave Alaska"
. generate P1kotz=1/(1+exp(L2kotz)+exp(L3kotz))
. label variable P1kotz "same area"
. generate P2kotz=exp(L2kotz)/(1+exp(L2kotz)+exp(L3kotz))
. label variable P2kotz "other Alaska"
. generate P3kotz=exp(L3kotz)/(1+exp(L2kotz)+exp(L3kotz))
. label variable P3kotz "leave Alaska"
```

Figures 1 and 2 show conditional effect plots for village and Kotzebue students separately:

```
. graph P1villag P2villag P3villag ties, symbol(pOT) l1(probability)
        b2("social ties to place (village schools)") ylabel(0,.2,.4,.6,.8,1)
        xlabel(-2,-1,0,1,2) yline(0,1) xline(0) noaxis
```

(see figure 1)

```
. graph P1kotz P2kotz P3kotz ties, symbol(pOT) l1(probability)
        b2("social ties to place (Kotzebue High School)") ylabel(0,.2,.4,.6,.8,1)
        xlabel(-2,-1,0,1,2) yline(0,1) xline(0) noaxis
```

(see figure 2)

The plots indicate that among village students, social ties increase the probability of staying rather than moving elsewhere in Alaska. Relatively few village students expect to leave Alaska. In contrast, among Kotzebue students ties particularly affect the probability of leaving Alaska rather than moving elsewhere. Only if they feel very strong ties do Kotzebue students tend to favor staying put.

Notes

1. This research was supported by a grant from the National Science Foundation (DPP-9111675); see C. L. Seyfrit and L. C. Hamilton (1992) "Social impacts of resource development on Arctic adolescents," in *Arctic Research of the United States* 6(Fall): 57–61.
2. `ties` represents the first principal component of fourteen survey questions about family and community, similar to those in Table 3 of C. L. Seyfrit and L. C. Hamilton (1992) "Who will leave? Oil, migration, and Scottish island youth," in *Society and Natural Resources* 5(3): 263–276.

Figures

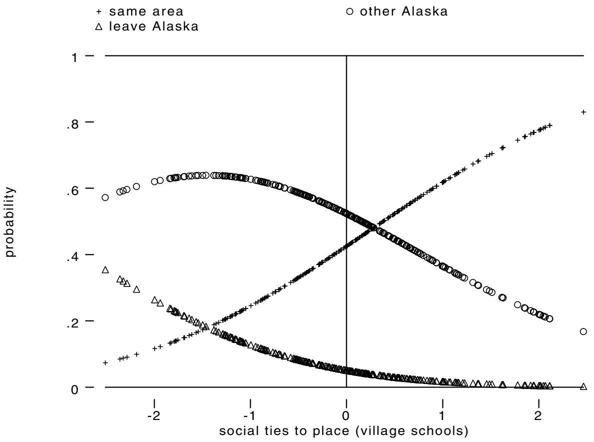

Figure 1

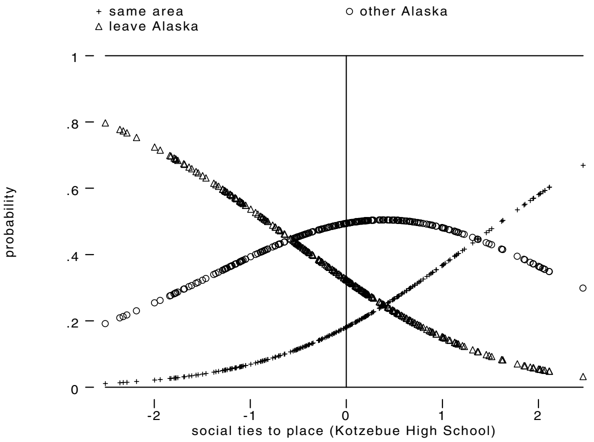

Figure 2

sts3 Cross correlations

Sean Becketti, Editor, STB, FAX 913-888-6708

`xcorr.ado` calculates cross correlations—that is, correlations between two variables at different lags—and displays them in a compact, readable form.

The syntax of `xcorr` is

`xcorr` *var1 var2* [`if` *exp*] [`in` *range*] [`, lags(`*# of lags*`)` { `kendall` | `pearson` | `spearman` }]

Discussion

Cross correlations are a natural extension of the concept of correlation to pairs of time series variables. For cross section variables x and y, the correlation is a measure of the linear association between the variables. For time series variables x_t and y_t, however, the contemporaneous correlation must be supplemented with measures of the correlations between x_t and y_t at various lags.

As an example, suppose that y_t is the growth rate of real (that is, adjusted for inflation) national output at time t and x_t is the growth rate of the money stock. A variety of economic theories hold that increases in the growth of the money stock tend to increase real output temporarily after a lag of 6 to 12 months. These theories predict that the correlation between lagged values of the money stock (x_{t-l} where l is the number of periods before t) should initially grow with l, the number of lags, but eventually decline as l grows very large and the temporary boost to output dissipates. The cross correlations between output growth and money growth provide a preliminary indication of the strength of these theories.

Cross correlations are also used as a diagnostic tool when fitting time series models. A common and important identifying assumption is that two variables are uncorrelated at any lag. The cross correlations provide a direct measure of the reasonableness of this assumption. Box and Jenkins provide examples of using cross correlations for this purpose.

Cross correlations can be calculated without using `xcorr`. The appropriate lagged variables can be generated, then correlated. For example:

```
. gen xlag = x[_n-1]
. gen ylag = y[_n-1]
. correlate x xlag y ylag
```

would display cross correlations of the variables `x` and `y`. This method is tedious for more than one or two lagged correlations. The `lag` command (from *sts2*) can be used to reduce the tedium.

```
. lag 4 x
. lag 4 y
. correlate x y L*
```

would display cross correlations of x and y with four lags of each variable (labeled L.x, L2.x, and so on). This method is easy, but the output is needlessly messy. For stationary variables, the correlation between x_t and y_t is the same as the correlation between x_{t-1} and y_{t-1}. However, the correlate command doesn't exploit this fact to reduce the number correlations displayed. Thus xcorr has two key advantages: it reduces the work required to produce cross correlations, and it presents the cross correlations in a compact, readable form.

Options

lags(*# of lags*) specifies the maximum lag to use in calculating cross correlations, that is, the maximum value of l. If this option is unspecified, xcorr calls the period program (supplied in *sts2*) to determine the maximum lag. Thus, by default, xcorr will use up to four lags for quarterly data, up to twelve lags for monthly data, and so on. In order for xcorr to determine the correct number of lags to use, the frequency of the data must have already been set with the period command or the number must be explicitly specified with the lags() option.

{**kendall | pearson | spearman**} selects the type of correlation coefficient to calculate. The default is **pearson**, the ordinary product-moment correlation. **spearman** selects the rank correlation coefficient, and **kendall** selects Kendall's τ_β.

Example

The prime rate, the benchmark rate banks use for their commercial lending, is set to reflect the general level of short-term interest rates. Some analysts believe that the prime rate adjusts only sluggishly to changes in market conditions. A preliminary examination of the relationship between the prime rate (rprime) and the rate on 3-month Treasury bills (rtb3) follows:

```
. use interest
(Monthly interest rates)

. describe

Contains data from interest.dta
  Obs:   877 (max= 12434)                   Monthly interest rates
 Vars:     5 (max=    99)
Width:    16 (max=   200)
  1. year         int    %8.0g              Year
  2. month        int    %8.0g    month     Month
  3. date         float  %9.0g              Date
  4. rprime       float  %9.0g              Prime rate
  5. rtb3         float  %9.0g              3-month T-bill yield
Sorted by:  year  month

. period 12
12 (monthly)

. xcorr rprime rtb3 if year>1982
           lags of            lags of
           rprime             rtb3
Lag      r    p-value       r    p-value
---    -----  -------     -----  -------
 0      0.96    0.00
 1      0.92    0.00       0.97    0.00
 2      0.88    0.00       0.95    0.00
 3      0.83    0.00       0.92    0.00
 4      0.77    0.00       0.87    0.00
 5      0.71    0.00       0.83    0.00
 6      0.64    0.00       0.77    0.00
 7      0.58    0.00       0.70    0.00
 8      0.53    0.00       0.63    0.00
 9      0.50    0.00       0.56    0.00
10      0.47    0.00       0.50    0.00
11      0.45    0.00       0.44    0.00
12      0.44    0.00       0.39    0.00
```

In this example, correlations are calculated using data after 1982. The period from 1979 through 1982 was marked by record high interest rates, anomalous patterns of prime rates and bank lending, and significant adjustments in monetary policy. As a result, data from this period could give misleading information about the typical correlations between the prime rate and the Treasury bill rate.

As the listing shows, the prime rate and the yield on 3-month Treasury bills are highly and significantly correlated at many lags. However nominal interest rates are typically nonstationary. These rates are usually differenced once to induce stationarity before they are analyzed. We use the dif command from *sts2*.

```
. dif rprime
. dif rtb3
. xcorr D.rprime D.rtb3 if year>1982, lags(4)
           lags of            lags of
          D.rprime           D.rtb3
Lag      r    p-value       r    p-value
---    -----  -------     -----  -------
 0      0.61    0.00
 1      0.25    0.00       0.68    0.00
 2      0.07    0.49       0.43    0.00
 3      0.07    0.48       0.21    0.03
 4     -0.00    0.92       0.14    0.14
```

After differencing, the prime rate and Treasury bill rates are still significantly correlated, but the correlation is substantially reduced and disappears after three quarters. Three lagged values of the Treasury bill yield are significantly correlated with the current value of the prime rate, consistent with the view that the prime rate adjusts sluggishly to changes in other market interest rates.

Note

xcorr uses the programs lag, period, and _ts_peri from *sts2* (STB-7), and the programs pearson, and spear from *sg5.1* (STB-13).

Saved Results

xcorr saves in the S_# macros:

S_1	correlation between *var1* and *var2*
S_2	p-value of correlation between *var1* and *var2*
S_3	correlation between L.*var1* and *var2*
S_4	p-value of correlation between L.*var1* and *var2*
S_5	correlation between *var1* and L.*var2*
S_6	p-value of correlation between *var1* and L.*var2*
.	
.	
.	

References

Box, G. E. P., and G. M. Jenkins. 1976. *Time Series Analysis: forecasting and control*. revised ed. Oakland, CA: Holden–Day.

Becketti, S. 1992. sts2: Using Stata for time series analysis. *Stata Technical Bulletin* 7: 18–26.

zz2 Cumulative index for STB-7—STB-12

[an] Announcements

STB-7	2	an17	Stata seminars announced *J. Hilbe*
STB-7	3	an18	STB-1–STB-6 available in bound format *J. Hilbe*
STB-7	3	an19	Stand-alone statistical tools *G. Dallal*
STB-8	2	an20	Stata U.K. distributor *C. Fox*
STB-8	2	an21	Stata for the Apple Macintosh *T. Anderson*
STB-8	3	an22	Stata for the IBM RS/6000 workstation *T. Anderson*
STB-9	2	an23	CPS labor extracts available *D. Feenberg*
STB-10	2	an24	Stata for the Macintosh ships *T. Anderson*
STB-10	2	an25	Complementary STB subscription for school libraries *P. Branton*
STB-10	2	an26	Request for data management problems *J. Anagnoson*
STB-11	2	an27	Pagano and Gauvreau text available *T. Anderson*
STB-11	3	an28	Spanish analysis of biological data text published *I. Salgado-Ugarte*
STB-12	2	an29	New edition of Statistics with Stata published *T. Anderson*
STB-30	3	an30	Editorship turnover *J. Hilbe*

[crc] CRC-Provided Support Materials

STB-7	3	crc12.1	Oops!, again
STB-8	3	crc13	Short describes, finding variables, and codebooks
STB-8	5	crc14	Pairwise correlation coefficients
STB-8	7	crc15	Parse bug
STB-8	7	crc16	Encapsulated PostScript driver
STB-9	3	crc17	Bug fixes
STB-9	3	crc18	Important difference between regular and Intercooled Stata
STB-9	4	crc19	Nonlinear regression command
STB-10	3	crc20	Stata expression-processing bug
STB-10	3	crc21	True Kaplan–Meier estimates for ltable
STB-10	3	crc22	Error fixed in rchart
STB-10	3	crc23	Minor bug fixes
STB-11	4	crc24	Error in corc
STB-11	4	crc25	Problem with tobit
STB-11	4	crc26	Improvement to poisson
STB-12	3	crc27	More extensions to generate: categorical variables
STB-12	4	crc28	Repeat command with specified arguments
STB-12	5	crc29	Macintosh report

[dm] Data Management

STB-7	3	dm7	Utility to reverse variable coding	*M. Jacobs*
STB-7	4	dm8	Command to unblock data sets	*J. Hilbe*
STB-7	4	dm9	An ANOVA blocking utility	*P. Lachenbruch*
STB-9	4	dm10	Infiling data: Automatic dictionary creation	*W. Gould*
STB-12	7	dm11	Matching the Current Population Surveys	*F. Welch*
STB-12	11	dm12	Selecting claims from medical claims data bases	*R. Vaughn*

[gr] Graphics

STB-12	12	gr3.1	3-dimensional graphics revisited	*W. Gould*
STB-7	5	gr10.1	Printing graphs and creating WordPerfect graph files	*M. Jacobs*
STB-9	8	gr11	Using CorelDraw as a Stata graphics editor	*M. Jacobs*
STB-12	13	gr12	Graphs of means and medians by categorical variables	*W. Gould*

[ip] Instruction on Programming

STB-9	9	ip2	A keyboard shortcut	*P. Lachenbruch*
STB-10	3	ip3	Stata programming	*W. Gould*
STB-11	5	ip3.1	Stata programming	*W. Gould*

[os] Operating System, etc.

STB-7	5	os4	Stata icon for Microsoft Windows 3.1	*J. Hilbe*
STB-8	7	os5	Running Stata under OS/2 2.0	*J. Hilbe*
STB-8	8	os6	Importing and exporting text files with Stata	*P. Goldberg*
STB-10	18	os7	Stata and windowed operating systems	*W. Gould*
STB-11	9	os7.1	Stata and windowed interfaces	*W. Rising*
STB-11	10	os7.2	Response	*W. Gould*
STB-11	10	os7.3	CRC committed to Stata's command language	*W. Gould*

[sbe] Biostatistics and Epidemiology

STB-12	14	sbe2.1	Bailey–Makeham survival models	*W. Rogers*
STB-12	14	sbe2.2	Faster hpredict.exe	*H. Krakauer*
STB-7	5	sbe5	Calculating person-years and incidence rates	*J. Hilbe*
STB-7	7	sbe6	3 × 3 matched case–control tests	*J. Hilbe*
STB-9	9	sbe7	Hyperbolic regression analysis in biomedical applications	*P. Geiger*
STB-11	10	sbe7.1	Hyperbolic regression correction	*P. Geiger*
STB-9	12	sbe8	Left-censored survival data	*W. Rogers*
STB-10	20	sbe9	Brier score decomposition	*W. Rogers*
STB-11	11	sbe10	An improvement to poisson	*G. Rodríguez*

[sed] Exploratory Data Analysis

STB-7	8	sed7	Resistant smoothing using Stata	*I. Salgado-Ugarte & J. Curts-Garcia*
STB-8	9	sed7.1	Resistant nonlinear smoothing using Stata	*W. Gould*
STB-11	14	sed7.2	Twice reroughing procedure for resistant nonlinear smoothing	*I. Salgado-Ugarte & J. Curts-Garcia*

[sg] General Statistics

STB-7	11	sg1.2	Nonlinear regression command	*P. Royston*
STB-8	12	sg1.3	Nonlinear regression command, bug fix	*P. Royston*
STB-11	17	sg1.4	Standard nonlinear curve fits	*P. Royston*
STB-12	14	sg1.5	Standard nonlinear curve fits update	*P. Royston*
STB-8	12	sg7	Centile estimation command	*P. Royston*
STB-8	15	sg8	Probability weighting	*W. Rogers*
STB-9	14	sg8.1	Huber exponential regression	*W. Rogers*
STB-9	14	sg9	Similarity coefficients for 2×2 binary data	*J. Hilbe*
STB-10	22	sg9.1	Additional statistics to similari output	*J. Hilbe*
STB-9	16	sg10	Confidence limits in bivariate linear regression	*P. Geiger*
STB-9	16	sg11	Quantile regression standard errors	*W. Rogers*
STB-9	19	sg11.1	Quantile regression with bootstrapped standard errors	*W. Gould*
STB-10	22	sg12	Extended tabulate utilities	*D. Judson*
STB-10	23	sg13	Is a transformation of the dependent variable necessary?	*R. Goldstein*
STB-10	26	sg14	Is a transformation of an independent variable necessary?	*R. Goldstein*
STB-11	17	sg15	Sample size determination for means and proportions	*J. Hilbe*
STB-11	20	sg16	Generalized linear models	*J. Hilbe*
STB-12	15	sg16.1	Generalized linear models using power links	*J. Hilbe*

[smv] Multivariate Analysis

STB-8	18	smv5.1	Loglinear analysis of cross classifications, update	*D. Judson*
STB-11	28	smv6	Identifying multivariate outliers	*W. Gould & A. Hadi*

[snp] Nonparametric methods

STB-9	21	snp4	Non-parametric test for trend across ordered groups	*K. Stepniewska and D. Altman*

[sqv] Analysis of Qualitative Variables

STB-7	18	sqv3	Wald and Atkinson's R extensions to logistic	*J. Hilbe*
STB-9	22	sqv3.1	Graphical display of Atkinson's R values	*M. Jacobs*
STB-8	18	sqv4	Calculation of the deviance goodness-of-fit statistic after logistic	*J. Hilbe*
STB-9	22	sqv4.1	Correction to ldev command output	*J. Hilbe*
STB-9	22	sqv5	Univariate log-likelihood tests for model identification	*J. Hilbe*
STB-10	27	sqv6	Smoothed partial residual plots for logistic regression	*J. Hilbe*
STB-12	16	sqv7	Cusum plots and tests for binary variables	*P. Royston*

[srd] Robust Methods and Statistical Diagnostics

STB-9	24	srd13	Maximum R-squared and pure error lack-of-fit test	*R. Goldstein*
STB-10	27	srd14	Cook–Weisberg test of heteroscedasticity	*R. Goldstein*
STB-10	29	srd15	Restricted cubic spline functions	*R. Goldstein*

[ssi] Simulation and Random Numbers

STB-8	19	ssi3	Continuous dynamic system modeling and simulation with Stata	*F. Danuso*

[sts] Time Series and Econometrics

STB-7	18	sts2	Using Stata for time series analysis	*S. Becketti*

[tt] Teaching

STB-7	26	tt4	Teaching ecology with Stata	*J. Taylor*
STB-7	28	tt5	Simple chemical equilibrium	*P. Geiger*
STB-12	17	tt6	Courseware	*W. Gould and W. Rogers*
STB-12	28	tt6.1	Courseware guidelines	*W. Gould and W. Rogers*
STB-12	32	tt6.2	Duxbury Press looking for courseware authors	*S. Loll*