STATA
TECHNICAL
BULLETIN

November 1991
STB-4

A publication to promote communication among Stata users

Editor

Joseph Hilbe
Stata Technical Bulletin
10952 North 128th Place
Scottsdale, Arizona 85259-4464
602-860-1446 FAX
stb@stata.com EMAIL

Associate Editors

J. Theodore Anagnoson, Cal. State Univ., LA
Richard DeLeon, San Francisco State Univ.
Paul Geiger, USC School of Medicine
Lawrence C. Hamilton, Univ. of New Hampshire
Stewart West, Baylor College of Medicine

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.

Copyright Statement. The Stata Technical Bulletin (STB) and the contents of the supporting files (programs, datasets, and help files) are copyright © by StataCorp. The contents of the supporting files (programs, datasets, and help files), may be copied or reproduced by any means whatsoever, in whole or in part, as long as any copy or reproduction includes attribution to both (1) the author and (2) the STB.

The insertions appearing in the STB may be copied or reproduced as printed copies, in whole or in part, as long as any copy or reproduction includes attribution to both (1) the author and (2) the STB. Written permission must be obtained from Stata Corporation if you wish to make electronic copies of the insertions.

Users of any of the software, ideas, data, or other materials published in the STB or the supporting files understand that such use is made without warranty of any kind, either by the STB, the author, or Stata Corporation. In particular, there is no warranty of fitness of purpose or merchantability, nor for special, incidental, or consequential damages such as loss of profits. The purpose of the STB is to promote free communication among Stata users.

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.

an12 CRC has new area code

Charles Fox, Computing Resource Center, 800-782-8272

Our telephone and fax numbers changed as of November 2. The new Area Code is 310. It is replacing 213 in many areas of Los Angeles County—so you might want to check the number of anyone else you call in this area.

New numbers: 310-393-9893 phone, 310-393-7551 fax. The 800 numbers stay the same: 800-782-8272 if calling from the U. S. (except Los Angeles); 800-248-8272 if calling from Canada.

an13 TCL is now marketing Stata in the UK

Ana Timberlake, Managing Director, Timberlake Clark Limited, London

Timberlake Clark Limited (TCL) are a statistical and economic modelling consultancy, based in Greenwich. Over its ten years existence, TCL have marketed and supported many statistical analysis packages. One of our non-executive directors (Patrick Royston) has been a very happy user of Stata for over a year.

Our support of Stata will be user driven and may include

- Over the telephone free support
- Training courses (public or in-house)
- User meetings
- and others, e.g. demonstrations, participation in technical meetings

Please contact me and let me know what activities you would like us to get involved with in the UK. I am particularly interested to know if you would benefit from a CHEST agreement for Stata.

We are also responsible for the organization of STATSIG (*S*pecial *I*nterest *G*roup in *STAT*istics), a group with activities sponsored by the IBM PC User Group. Let us know if you are interested in receiving details of our meetings and activities.

Address: Timberlake Clark Limited
40b Royal Hill
Greenwich
London SE 10 8RT
UK
Phone: 44-81-692 6636
Fax: 44-81-691 3916

an14 SAS's new technical journal

Joseph Hilbe, Editor, STB, FAX 602-860-1446

SAS users will be pleased to learn that the Institute is publishing a new technical journal called *Observations: The Technical Journal for SAS Software Users*. The first issue was just published (Fourth Quarter, 1991). Subscription cost is $49 per year for the quarterly. No diskettes are provided.

If you are interested contact The SAS Institute, SAS Circle, Box 8000, Cary, NC 27512-8000. The telephone number is 919-677-8000 and the FAX number is 919-677-8123.

an15 Regression with Graphics released

Joseph Hilbe, Editor, STB, FAX 602-860-1446

Brooks/Cole Publishing Company, a division of Wadsworth, will release Lawrence Hamilton's new text entitled *Regression with Graphics* (ISBN 0-534-15900-1) this month (anticipated release date of November 18). Written as a sequel to his earlier *Modern Data Analysis: A First Course in Applied Statistics*, this text is aimed at a second semester undergraduate course in applied statistics or data analysis. It is also appropriate for graduate social science courses where students should become more intimately familiar with regression diagnostics, EDA and graphical techniques, robust methods, logistic regression, and elementary principal components and factor analysis. Text material is heavily supplemented with numerous graphs that were produced with Stata and Stage. STB-1, -2, and -3 inserts provided by Dr. Hamilton were based on parts of the text.

There are few general texts on regression that directly address robust and logistic regression. Hamilton has done excellent work in making otherwise rather difficult material accessible to the undergraduate reader. His discussion of logistic regression follows that of Hosmer & Lemeshow's *Applied Logistic Regression*, John Wiley & Sons, 1989. Stata users will find it particularly useful when using `logiodd2` (see *sqv1.3*).

The following Table of Contents is based on a pre-publication manuscript copy. Expect minor changes with the final release. Each chapter ends with a Conclusion, Exercises, and Notes (not shown).

Chapter 1: VARIABLE DISTRIBUTIONS— The Concord Water Study; Mean, Variance, and Standard Deviation; Normal Distributions; Median and Interquartile Range; Boxplots; Symmetry Plots; Quantile Plots; Quantile-Quantile Plots; Quantile-Normal Plots; Power Transformations; Selecting an Appropriate Power

Chapter 2: BIVARIATE REGRESSION ANALYSIS— The Basic Linear Model; Ordinary Least Squares; Scatterplots and Regression; Predicted Values and Residuals; R^2, Correlation, and Standardized Regression Coefficients; Reading Computer Output; Hypothesis Tests for Regression Coefficients; Confidence Intervals; Regression through the Origin; Problems with Regression; Residual Analysis; Power Transformations; Understanding Curvilinear Regression

Chapter 3: BASICS OF MULTIPLE REGRESSION— Multiple Regression Models; A Three-Variable Example; Partial Effects; Variable Selection; A Seven-Variable Example; Standardized Regression Coefficients; t-Tests and Confidence Intervals for Individual Coefficients; F-Tests for Sets of Coefficients; Multicollinearity; Search Strategies; Interaction Effects; Intercept Dummy Variables; Slope Dummy Variables; Oneway Analysis of Variance; Twoway Analysis of Variance

Chapter 4: REGRESSION CRITICISM— Assumptions of Ordinary Least Squares; Correlation and Scatterplot Matrices; Residual versus Predicted Y Plots; Autocorrelation; Nonnormality; Influence Analysis; More Case Statistics; Symptoms of Multicollinearity

Chapter 5: REGRESSION WITH TRANSFORMED VARIABLES— Transformations and Curves; Choosing Transformations; Diagnostics; Conditional Effect Plots; Comparing Curves

Chapter 6: ROBUST REGRESSION— A Two-Variable Example; Goals of Robust Regression; M-Estimation and Iteratively Reweighted Least Squares; Calculation by IRLS; Standard Errors and Tests for M-Estimates; Using Robust Estimation; A Robust Multiple Regression; Bounded-Influence Regression

Chapter 7: LOGIT REGRESSION— Limitations of Linear Regression; The Logit Regression Model; Estimation; Hypothesis Tests and Confidence Intervals; Interpretation; Statistical Problems; Influence Statistics for Logit Regression; Diagnostic Graphs

Chapter 8: PRINCIPAL COMPONENTS AND FACTOR ANALYSIS— Introduction to Principal Components and Factor Analysis; A Principal Components Analysis; How Many Components?; Rotation; Factor Scores; Graphical Applications: Detecting Outliers and Clusters; Principal Factor Analysis; An Example of Principal Factor Analysis; Maximum Likelihood Factor Analysis

Appendices: Population and Sampling Distributions; Computer-Intensive Methods; Matrix Algebra; Statistical Tables

Anyone interested in obtaining a copy of the book should contact Wadsworth at 800-354-9706.

crc11 Drawing random samples

The syntax of `sample` is

`sample` # [`if` *exp*] [`in` *range*] [`, by(`*groupvars*`)`]

`sample` draws a # percent sample of the data in memory, thus discarding 100–# percent of the observations. Observations not

meeting the optional `if` and `in` criteria are kept (sampled at 100%). If `by()` is specified, a # percent sample is drawn within each set of values of *groupvars*, thus maintaining the proportion of each group.

Sample sizes are calculated as the closest integer to $(\#/100)N$, where N is the number of observations in the data or group. Thus, a 10 percent sample of 52 observations will select 5 observations, while a 10 percent sample of 56 observations will select 6. Note that a 10 percent sample of 4 or fewer observations selects nothing.

Sampling is defined as drawing observations without replacement. The previously released `bootsamp` (see '`help bootsamp`') will draw observations with replacement. If you are serious about drawing random samples, you must first set the random number seed with the `set seed` command.

Say you have data on the characteristics of patients, You wish to draw a 10 percent sample of the data in memory. You type '`sample 10`'.

Assume that among the variables is `race`. `race==0` are whites and `race==1` are nonwhites. You wish to keep 100% of the nonwhite patients but only 10% of the white patients. You type '`sample 10 if race==0`'.

If instead you wish to draw a 10% sample of white and a 10% sample of nonwhite patients, you type '`sample 10, by(race)`'. This differs from typing simply '`sample 10`' in that, with `by()`, `sample` holds constant the ratio of white to nonwhite patients.

dm3 | Automatic command logging for Stata

D. H. Judson, Dept. of Sociology, Washington State University

Although Stata's interactive command language system is particularly useful for exploratory data analysis and instant response, interesting analyses are often lost (or must be laboriously repeated) because the user forgets to log commands and/or output. More importantly, sweeping data changes cannot be easily repeated, and such changes, at best, are dangerous. However, the ability to rapidly generate new variables, predicted values, and the like is a useful one for many purposes. Thus, we are faced with the problem: How do we perform analyses and data management while retaining a record of our work?

A simple solution is to revert to batch (ado) file programming, but this defeats the whole purpose of interactive and exploratory data analysis. The solution, of course, is automatic logging. If log files can be generated automatically at the start of a work session, the user never needs to worry that an analysis of data change cannot be remembered or repeated.

The following additions to the file `profile.do` accomplish automatic logging. They can be appended to the end of a standard `profile.do` file. This implementation works in DOS and assumes that the logs are collected in the `c:\logs` subdirectory.

```
* AUTOMATIC LOG PROGRAM
capture program drop autolog
program define autolog
mac def _Name=1
mac def _retc=602
while %_retc==602 {
   cap log using c:\logs\auto`%_Name`.log, noproc
   mac def _retc=_rc
   mac def _Name=%_Name+1
}
mac drop _Name
mac drop _retc
end
autolog
program drop autolog
di in bl "  log on..."
```

The commands perform the following steps:

1) The macro `_Name` is set to 1.
2) Stata attempts to create the log file `c:\logs\auto1.log`. This log file uses the option `noproc` to log only the commands entered at the keyboard, thus reducing its size.
3) If the file `c:\logs\auto1.log` already exists, the program increments the macro `_Name` by 1 and repeats the process at step 2 with the log file `c:\logs\auto2.log`.
4) This process continues until a file is created.
5) The program `autolog` is created and executed.

Note that legal DOS file names are eight characters long, so that log files can range from `auto1.log` to `auto9999.log`.

Long before the theoretical upper limit is reached, however, the time required for the program to search for a non-existent log file becomes prohibitive. On a 10-Mhz 80286 IBM-compatible computer with a 28-millisecond hard drive using the regular DOS version of Stata, if the program needs to search ten times to find a non-existent log file (i.e., `auto1.log` through `auto9.log` already exist), it will take about 15 seconds. Therefore, the user will want to periodically check these log files and erase superfluous or redundant logs. BiTurbo and Intercooled Stata are faster.

The automatically created log files can be edited in any word processor or text editor, or can be seen via the DOS `type` command. As indicated in the Stata manual, they also can be executed as a do-file, if necessary.

dm4 | A duplicate value identification program

Marc Jacobs, Social Sciences, University of Utrecht, The Netherlands FAX (011)-31-30-53 4405

Some time ago I needed to know if all values of a variable were unique. `tabulate` did not help because there were usually too many values involved. Although the manuals do not make it obvious, one solution is

```
. sort x
. quietly by x: assert _N==1
```

The `assert` command verifies a claim is true and complains otherwise. `_N` is the number of observations in the "data set" which, when prefixed with `by`, means the number of observations in the by-group. The `quietly` suppresses all the output, which would include the identity of each by-group. If the claim is false, however, the return code is nonzero.

I have constructed a small utility program to implement this. The syntax is

`chkdup` *varname*

`chkdup` can be found in the `dm4` directory on the STB-4 diskette.

gr8 | Printing a series of Stata graphs

John A. Anderson, Dept. of Civil Engineering, University of New Hampshire

[Prof. Anderson has created two `.bat` *files related to this article. They may be found in the* `gr8` *directory on the STB-4 diskette. To enhance speed, I have made both* `.bat` *files into binary* `.com` *files. They are also located in the* `gr8` *directory—Ed.]*

I print most of my Stata graphs at 300 dpi resolution on an HP LaserJet Series II. Depending on the complexity of the graph, it can take quite some time for the output. The waiting is multiplied when printing a series of graphs in a single sitting. Production, however, can be enhanced by using appropriate MS-DOS commands, creating a `BAT` file, and/or running `GPHDOT.EXE` or `GPHPEN.EXE` under Windows 3.0. Depending on your needs one approach may be better than another at a particular time. Below I have outlined procedures that I have found helpful when printing Stata graphs.

Using a single MS-DOS command line

When each graph you wish to print requires the same Stata print options, you can print all of the graphs using one MS-DOS command line.

1. Use the `COPY` command to ensure that all the files you wish to print and no other files are in a single subdirectory.
2. Know where your `GPHDOT.EXE` and/or `GPHPEN.EXE` are located.
3. Use the `FOR` command at the `DOS` prompt outside of a batch file. (In the example below, the command is invoked from the subdirectory containing `GPHDOT.EXE`—you can do the same thing with `GPHPEN.EXE`.)

```
Example:  FOR %P IN (C:\STATA\MYGRAPH\*.*) DO GPHDOT %p
```

This command will perform `gphdot` (located in `C:\STATA`) on each file in the subdirectory `C:\STATA\MYGRAPH`.

Using a "self-calling" BAT file

When each graph you wish to print requires the same Stata print options and no more than six options, the `PG.BAT` or `PG1.BAT` files on the STB-4 disk can be used.

1. If you have MS-DOS version 3.3 or later, copy the `PG.BAT` file (copy `PG1.BAT` for MS-DOS 3.2 or earlier) from the STB-4 disk to the subdirectory where you keep `GPHDOT.EXE` and/or `GPHPEN.EXE` (usually `C:\STATA`).
2. Use the `COPY` command to ensure that all the files you wish to print and no other files are in a single subdirectory.
3. From the subdirectory containing the graphs, type the command `PG` and press <ENTER>.

Using this BAT file eliminates the need for remembering a somewhat unusual MS-DOS command line. Also, it is "friendly" and easy for beginners and students to use. If the command is not entered correctly, an explanation will be provided to help the user reenter the command correctly.

Note: More information on the use of FOR and CALL can be found in Wolverton (1989). It should be noted, however, that I was unable to successfully obtain the desired output when following Wolverton's self-calling BAT example exactly. A comparison of the PG.BAT and the BAT on page 83 will reveal the discrepancies.

Using a modifiable BAT file

When you require different Stata options or Stata print commands to be used with each graph in a series, a modifiable BAT file can be effectively used.

1. Using a text editor, create a BAT file (e.g., PJOB.BAT) and list "line-by-line" the necessary Stata command for each graph (in this example, store PJOB.BAT in the subdirectory that contains GPHDOT.EXE and/or GPHPEN.EXE).

```
Example:  GPHDOT C:\STATA\MYGRAPH\ONE.GPH /N
          GPHDOT C:\STATA\MYGRAPH\TWO.GPH /N /+ /C2
          GPHPEN C:\STATA\MYGRAPH\THREE.GPH
```

2. Execute the BAT file.
3. Using your text editor, modify the BAT file each time you wish to print a different set of graphs.

Using Windows 3.0 Enhanced Mode to increase productivity

When a printing session is expected to take up a fair amount of time you have two choices—go to lunch or use Windows 3.0 in enhanced mode to print your graphs in the background while doing other work on your word processor, spreadsheet, or Stata.

1. To use enhanced mode with Windows, you need an 80386 or 80386SX processor—if you have a 386 choose any of the printing methods described above.
2. Access your DOS COMMAND.COM by double clicking on the DOS Prompt icon (if you do not have a DOS Prompt icon, use your file manager to run COMMAND.COM).
3. When DOS becomes active, hold down the ALT key and press the SPACEBAR.
4. Click on "Settings..." in the drop own menu that is displayed.
5. Under "Tasking Options" click on "Background" and then click on "OK".
6. Enter the command for the Stata print method of your choice at the DOS prompt—you can now minimize the COMMAND window into an icon and access another program such as your word processor—your graphs will continue printing in the background until they are completed.

References

Wolverton, Van 1989. *Supercharging MS-DOS*. Redmond, WA: Microsoft Press

os2.1 Questions and answers about Stat/Transfer—an addendum

Ted Anagnoson, California State University, Los Angeles

In STB-2, there were a series of questions and answers about Stat/Transfer. While they are complete as far as they go, I have discovered several characteristics of Stat/Transfer that will make life much easier for those of you who use it to move datasets around.

Q. I am using Borland's Quattro Pro Spreadsheet. Can I use Stat/Transfer to move my data into Stata?

A. The answer given was to save the spreadsheet in the .wk1 format that Borland and most spreadsheets provide and then move it. This is OK, but we had a case this summer in our workshops on exploratory data analysis and Stata where Quattro had saved the worksheet with a blank line in row 1, the variable names in row 2, a blank line in row 3, and the data starting in row 4. Since Stat/Transfer assumes that the top row of the spreadsheet is the variable names and the next row is the first case of data, it would not transfer over this worksheet.[1] So you might need to look at the spreadsheet either in Quattro format or in Lotus format and make sure that there are no blank lines or other inconsistencies.

Q. How can I control the size and data type of my Stata variables?

A. The information given is correct, but for Lotus 1-2-3 files, I have found the following: First, Stat/Transfer looks only at the top row of the data (the second row of the spreadsheet) to decide if a variable is transferable. If that second row is blank, a character variable, and perhaps a numeric variable, will not be transferred (actually, Stat/Transfer won't put the variable in the variable list from which you select variables to be transferred). So the top row of data is crucial. If you have a dataset with a lot of missing data, you might want to make sure that the top row or first case is complete and has the correct variable format (character or numeric) for a successful transfer. We have inserted 999s or strings of "xxxxxxxxx" for character variables into the first few rows of our files when those rows have missing data in them.

The second addendum concerns Stat/Transfer's ability to minimize the size of data files. I have found that with Lotus 1-2-3 even variables which are formatted with fixed format, zero decimal places, transfer over as `long` or `float` variables. Here the problem is probably with Lotus rather than Stat/Transfer. I have not been able to get a variable to transfer over as a `byte` or an `int` in spite of several attempts using various features of Lotus. While Stat/Transfer may minimize the size of variables in other statistical packages like Systat or Gauss, it does NOT do so with Lotus 1-2-3.

In spite of this fact, I have found over time that for all but the smallest datasets, inputting the data via Lotus 1-2-3 and transferring over the file is by far the easiest way of inputting data, and I strongly recommend the use of Stat/Transfer in those circumstances.

Q. How do I move large mainframe datasets into Stata?

A. We had a 2.1 megabyte SAS file which we attempted to import into Stata via dbms/copy. Unfortunately dbms/copy gave us gibberish. We finally got dbms/copy to produce a 7.0+ megabyte ASCII file which we reread into Stata as raw data (dbms/copy's raw data format has 13 columns per numeric variable, which accounts for a lot of white space in that file[2]).

Bottom line: the advice that the "TOSPSS" procedure be used for SAS datasets seems to be solid. In defense of dbms/copy, they do state that SAS/PC files have an "encrypted field" within them.

Notes

1. We have been unable to determine whether the blank lines got into the spreadsheet from user error or from some characteristic of Quattro when it saves a `.wk1` file.
2. PK-ZIP took that 7.0+ megabyte ASCII file and produced a zipped version of just over 500K.

os3.1 | Comment on os3: using Intercooled Stata within DOS 5.0

Marc Jacobs, Social Sciences, University of Utrecht, The Netherlands, FAX (011)-31-30-53 4405

I am certainly a fan of both Stata, preferably the Intercooled version, and Windows 3.0. Running regular Stata Professional under Windows 3.0 in 386 enhanced mode never gave me problems, except that it is a bit slower than running Stata directly under DOS. There are two major advantages to running it under Windows: Windows can emulate expanded memory; and using the advanced settings with the PIF editor, a user can easily switch from Stata to the editor, where various do-files can be accessed. In my work, I find that data analyses, do-files, and other Stata programs need a lot of testing. What is more comfortable than to edit a do-file, run it, find out what went wrong, and then start editing at the same point where you left it?

I too have noticed that Intercooled Stata and the enhanced mode of Windows 3.0 do not work together. Running the former in the Windows real mode is not satisfactory—it's just like working on an IBM 6 Mhz PC again! Several times the Windows just crashed. Moreover, using the `emm386.sys` manager prohibited me from running Intercooled Stata in standard mode. The solution: small jobs I run in regular Stata Professional under Windows; big jobs I run Intercooled Stata directly under DOS 5.0.

I would like to give a warning about using the `noems` switch when emulating expanded memory under Windows. Do not do it. Stata files may become corrupted as a result. Set the device driver in such a way that specific emulated RAM is provided. In other words

```
device=c:\bin\dos\emm386.exe 1024 ram
```

rather than

```
device=c:\bin\dos\emm386.exe noems
```

os3.2 | A follow-up question to os3: using Intercooled Stata within DOS 5.0

Arnold Katz, Department of Economics, University of Pittsburgh, FAX 412-648-1793

Q. Editor: I am under the impression you believe that it is conceptually feasible to run Intercooled Stata from Windows 3.0 in enhanced mode with the QEMM/386 memory manager. Being able to run Intercooled Stata under Windows would greatly

simplify my work. At present I use Windows Standard mode under QEMM with the QEMM parameter EMBMEM. If `x` is one's maximum RAM and `y` is the RAM to be reserved for Intercooled Stata, EMBMEM is set to `y`. The result is fairly crippled. Not only does Windows lose access to `x-y` RAM, but it is also impossible to exploit the task switching option that would be available if Intercooled Stata could be run in enhanced mode. Is there a way to run Intercooled Stata in enhanced mode?

A. The present version of Intercooled Stata is unable to run under Windows 3 in enhanced mode—you must use standard or real mode. In order to run in protected mode, a program must be compiled to support the DOS Protected Mode Interface (DPMI) standard, something that can only be done by CRC. Many protected mode programs, for example, AutoCAD, have a similar problem. CRC tells me that they will soon release a version of Intercooled Stata that will run in enhanced mode.

In the meantime, depending on your hardware constraints as well as on other software considerations, you may employ any of the following:

1. In real mode, call Windows by using the `win/r` command. Then click on "file" and "run." Type, for example, `c:\stata\istata.exe`.
2. In standard mode, call Windows by using the `win/s` command. Use the same procedure as in 1. Following your suggestion, however, most users may need to use the QEMM EMBMEM parameter as you described in your question. Always check the documentation that came with your computer. Research, experimentation, and patience seem critical.
3. Opt out of Windows and run Intercooled Stata from DOS.

sg3.6 A response to sg3.3: comment on tests of normality

Patrick Royston, Royal Postgraduate Medical School, London, FAX (011)-44-81-740 3119

In my opinion, the distribution of the D'Agostino-Pearson K^2 statistic for testing for non-normality is not close enough to chi-square(2) for practical use. As I showed in *sg3.1* (Royston 1991a), it rejects the null hypothesis of normality too often, particularly for small samples (n<100, say) and for stringent tests (e.g. significance levels of 0.01 or less). However, in *sg3.5* (Royston 1991b), I supplied a correction to K^2 which overcame the problem, so that it is no longer an issue.

I certainly concur with D'Agostino et al.'s (1990, 1991) stated aim of replacing the Kolmogorov test with better tests. D'Agostino et al. (1990) recommended both K^2 and the Shapiro–Wilk W as good "omnibus" tests. One of their problems with W was that it was unavailable for n>50; this is not so, see Royston (1982) and `swilk.ado` in *sg3.2* (Royston 1991c). They also complained that if W rejected the null hypothesis, it provided no information on the nature of the departure from normality. Meaningful indices of non-normality can in fact be derived by linear transformation of the Shapiro–Wilk and Shapiro–Francia statistics (Royston 1991d). The index for the Shapiro–Francia test is called V' and is proportional to the variance of the difference between the ordered data and expected normal order statistics. It provides an intuitively reasonable link between the normal plot and the test of non-normality.

My approach to testing for non-normality lines up with that of D'Agostino et al. (1990): "A good complete normality analysis would consist of the use of the [normal probability] plot plus the statistics." In that order! The normal plot shows all the data and gives more information about possible non-normality than any number of summary statistics. The information includes the presence of outliers and whether the data are skew and/or short- or long-tailed. However, it may be useful to know whether non-linearities in the normal plot are more likely to be "real" than to be caused by chance fluctuation. That is the job of tests like K^2, W, etc.

D'Agostino's comment "as sample sizes increase, as any applied researcher knows, these tests will reject the null hypothesis ..." applies to any test of significance whatever. It is a well-recognised drawback of the hypothesis-testing approach and is one reason why modern statistical practice leans towards estimates with confidence intervals rather than P values. It is true that $\sqrt{b_1}$ and b_2 do estimate population skewness and kurtosis. Unfortunately, however, the estimates may be highly biased (especially for skew populations), the bias depends on the sample size and, as far as I know, confidence intervals are not available, so their value seems to be limited. Arguably better indices of population shape are based on so-called L-moments (Hosking 1990, Royston 1991e), but even then confidence intervals are problematic.

A major problem with tests and estimates of non-normality underlies D'Agostino et al.'s (1991) comment "[skewness and kurtosis] can help us to judge if our later inferences will be affected by the nonnormality." I would ask, how? If one is trying to use the sample mean and standard deviation to estimate centiles (such as the 5th and 95th, a common clinical application) of a distribution believed to be approximately normal, even slight departures from normality may make the estimates unacceptably inaccurate. What values of $\sqrt{b_1}$ and b_2 (or any other statistic) indicate "slight" non-normality here? Similarly, one would like to know whether for example a given t-test or confidence interval for a mean is valid or is compromised by non-normality in the data. Until answers to specific questions of this sort are available, the inferential value of non-normality statistics is doubtful. Clearly, much research is needed.

We are left with a vague feeling that since many statistical tests and estimates assume normality, we ought to test for non-normality even if we can't really interpret the results. In this unsatisfactory situation, a choice between available tests, if it is to be made at all, should, I contend, be based mainly on power comparisons. Much has been written on this subject which I shall not try to summarize here. Generally speaking, the power of the K^2, W and Shapiro–Francia W' tests seems broadly comparable and considerably better than the older Kolmogorov and Pearson chi-square tests. K^2 seems weak against skewed, short-tailed distributions. W is weak against symmetric, rather long-tailed distributions. W' is weak against symmetric, short-tailed distributions. No test is perfect!

References

D'Agostino, R. B., A. Belanger and R. B. D'Agostino, Jr. 1990. A suggestion for using powerful and informative tests of normality. *American Statistician* 44(4): 316–321.

——. 1991. sg3.3: Comment on tests of normality. *Stata Technical Bulletin* 3: 20.

Hosking, J. R. M. 1990. L-moments: analysis and estimation of distributions using linear combinations of order statistics. *Journal of the Royal Statistical Society* B, 52: 105–124.

Royston, J. P. 1982. An extension of Shapiro and Wilk's W test for normality to large samples. *Applied Statistics* 31: 115–124.

——. 1991a. sg3.1: Tests for departure from normality. *Stata Technical Bulletin* 2: 16–17.

——. 1991b. sg3.5: Comment on sg3.4 and an improved D'Agostino test. *Stata Technical Bulletin* 3: 23–24.

——. 1991c. sg3.2: Shapiro–Wilk and Shapiro–Francia tests. *Stata Technical Bulletin* 3: 19–20.

——. 1991d. Estimating departure from normality. *Statistics in Medicine* 10: 1283–1293.

——. 1991e. Which measures of skewness and kurtosis are best? *Statistics in Medicine* 10, in press.

smv1	Single factor repeated measures ANOVA

Joseph Hilbe, Editor, STB, FAX 602-860-1446

The syntax for the `ranova` command is

`ranova` <*varlist*> [`if` *exp*] [`in` *range*]

`ranova` automatically checks for missing values across variables listed on the command line. When a missing value is found in any variable, it deletes the observation from active memory. However, the original data set is restored to memory upon completion of the analysis. The program provides information regarding excluded observations.

The statistical design permits analysis of repeated (treatment) measures on the same individuals. Each variable in the *varlist* corresponds to one treatment. `ranova` tests the hypothesis that all treatments have the same mean against the alternative that the treatment means are different from one another. This test is similar to a twoway analysis of variance in which the factors are treatment and subject, but the data are organized differently. Total model variability is divided into

1. SS Treatment: the variability resulting from the independent variable; that is, the levels or categories of response.
2. SS Within: the variability that cannot be accounted for by the independent variable.
    a. SS Subjects: the variability resulting from individual differences.
    b. SS Error: the variability resulting from random factors.

This test works for the simplest special case of the repeated measures design, but it does not handle any complications. Since each individual provides a value for each level or category of the independent variable, it is possible to measure the individual difference variability. This is not possible in randomized designs. However, there are several complications that may make this test invalid. In many instances when individuals are being measured over time, there may be a carry-over effect from early measurements to later ones. This will bias the test statistic. Moreover, when there are more than two measurements, the model has the assumption of homogeneity of covariance. This assumption is violated, for example, when one pair of levels is fairly close in time whereas another pair is more distant. Violations of this sort affect Type I error rate. This problem is ameliorated by using the Huynh–Feldt correction or by transforming the repetitions of dependent variables into separate dependent variables and analyzing the model by profile analysis or MANOVA.

If a significant difference between levels of the independent variable has been determined, Tukey HSD tests may be calculated to ascertain which level(s) are significantly different. The formula is

$HSD = q\sqrt{MS_{error}/N}$ where the appropriate `q` value is found on a Studentized Range Table.

Output of a `ranova` run with one missing value and four dropped observations appears as

```
. ranova Var1-Var4 in 5/20
                  Mean                  Standard Deviation
    Var1         101.2667                      14.9450
    Var2         103.4667                      15.1510
    Var3         104.4000                      17.0955
    Var4         107.3333                      15.9135
                Single Factor Repeated Measures ANOVA

       Number of obs in model = 15         Number of vars = 4
       Number of obs dropped  = 5
       Source |     SS          df       MS        F      Prob > F
-------------------------------------------------------------------
     Subjects | 13459.9300     14        -         -         -
   Treatments |  284.5800      3       94.8600     7.70    0.0003
        Error |  517.6700      42      12.3255     -         -
-------------------------------------------------------------------
        Total | 14262.1800     59
```

smv2 | Analyzing repeated measurements—some practical alternatives

William H. Rogers, CRC, FAX 310-393-7551

Longitudinal studies (also known as panels or cross-sectional time-series) are some of the most potentially informative yet complicated studies to analyze. A typical longitudinal study follows individuals over time to monitor the effects of some experimental treatment. For example, two baseline measures might be taken and then a drug administered to half the sample. The patients are assessed one, two, and three months later.

One technique for analyzing such data is repeated measures ANOVA, a powerful statistical technique with a well-deserved reputation for flexibility in addressing the complex relationships found in longitudinal studies. As a result, the technique is often recommended by well-intentioned theoreticians and authors of texts when, in practice, it is not usable in certain situations because the data are too messy. The most troublesome aspect of this messiness, from a repeated measures ANOVA standpoint, is missing data. One might set out to assess patients one, two, and three months later, but some patients may skip the first assessment, others the second, and so on.

Longitudinal studies *can* be successfully analyzed without resorting to repeated measures ANOVA and without discarding potentially informative incomplete observations. Moreover, the analytic alternatives throw the assumptions into sharper focus, are more descriptive, and offer more possibilities to connect intuition to analysis.

To demonstrate this, I will present some data that pose interesting substantive questions and reveal some of the complexities caused by missing data. We will then examine simple cross-sectional regressions and the behavior of changes over time and finally, we will compute individual "slopes" for each observation and analyze those slopes. When we are through, we will have a better understanding of this data than if we had been able to apply repeated measures ANOVA to this data. The message is that, even had the data been clean enough to apply repeated measures ANOVA, we might still wish to pursue these alternative, less exotic techniques.

The data are drawn from a real study (Tarlov et al. 1989). The underlying data consist of thousands of variables including variables recording marital status and gender, age, race (coded 1 for nonwhites and 0 for whites), and mental health measured at 5 points in time for patients suffering from chronic diseases (either mental or physical). The goal of our analysis is to model the relationship between the demographic information and mental health. The five time periods are unequally spaced. There are two measurements at the beginning, 3 months apart. The last three measurements follow 1, 2, and 4 years later. Our data contains

```
Contains data from mhifile.dta
  Obs:  3869 (max= 24749)                 MOS Patient Form Data
 Vars:     9 (max=   254)
  1. married      int    %8.0g
  2. iagecont     float  %10.0g           Patient age
  3. imale        int    %8.0g           Patient is male
  4. inonwht      int    %8.0g           Patient is nonwhite
  5. mhi0         float  %9.0g           Baseline Mental Health
  6. mhi3mo       float  %9.0g           Mental Health at 3 months
  7. mhi1yr       float  %9.0g           Mental Health at 1 year
  8. mhi2yr       float  %9.0g           Mental Health at 2 years
  9. mhi4yr       float  %9.0g           Mental Health at 4 years
```

As a way of getting to know this sample, I begin by presenting a series of marital status tabulations summarizing the mental health index:

```
. tab married, summ(mhi0)
           |  Summary of Baseline Mental Health
   married |        Mean   Std. Dev.       Freq.
-----------+------------------------------------
         0 |   65.238349   23.932533         751
         1 |   71.337259     21.6785        1104
-----------+------------------------------------
     Total |   68.868105   22.809219        1855

. tab married, summ(mhi3mo)
           | Summary of Mental Health at 3 months
   married |        Mean   Std. Dev.       Freq.
-----------+------------------------------------
         0 |   68.699844   22.002264        1376
         1 |   73.239149    20.31232        1968
-----------+------------------------------------
     Total |   71.371301   21.139313        3344

. tab married, summ(mhi1yr)
           |  Summary of Mental Health at 1 year
   married |        Mean   Std. Dev.       Freq.
-----------+------------------------------------
         0 |   69.756461   21.821531         761
         1 |   74.669736   18.927619        1086
-----------+------------------------------------
     Total |   72.645371   20.309153        1847

. tab married, summ(mhi2yr)
           | Summary of Mental Health at 2 years
   married |        Mean   Std. Dev.       Freq.
-----------+------------------------------------
         0 |   70.728295   21.383028         741
         1 |   74.994872   19.236051        1040
-----------+------------------------------------
     Total |   73.219727    20.26075        1781

. tab married, summ(mhi4yr)
           | Summary of Mental Health at 4 years
   married |        Mean   Std. Dev.       Freq.
-----------+------------------------------------
         0 |    72.62061    21.52095         579
         1 |    76.28414   18.028531         847
-----------+------------------------------------
     Total |   74.796634   19.597544        1426
```

The first measurement (`mhi0`) is available on a random half of the sample only, but `mhi3mo` is available on just about everybody. Between 3 months and one year, approximately half of the sample was discarded by the original investigators. There is additional attrition during the follow-up period.

It appears that married persons have "better" mental health (higher values), but that the gap may be narrowing over time. One purpose of longitudinal studies is to distinguish cohort effects—relationships that appear to be present but are really manifestations of group associations—from causal effects or trends. For example, it could be that marriage produces good mental health—a real effect—or it could be that people with good mental health tend to get married—a manifestation. If the latter is the case, scores would regress toward the mean over time.

A multivariate examination is now warranted. I will begin with a multivariate examination of the cross-sectional effects:

```
. corr married imale iagecont inonwht
(obs=3691)
        |  married    imale iagecont  inonwht
--------+------------------------------------
 married|   1.0000
   imale|   0.2456   1.0000
iagecont|   0.0663   0.0872   1.0000
 inonwht|  -0.0749  -0.0814  -0.0904   1.0000
```

Fortunately, there does not appear to be much intercorrelation between the demographic variables so we do not have to be overly concerned with which variables we include in the model. Even if we omitted a relevant variable, it would have little effect on the estimated coefficients of the others.

```
. reg mhi3mo imale iagecont married inonwht
(obs=3265)
  Source |     SS         df       MS          Number of obs =    3265
---------+------------------------------       F(  4,  3260) =  116.13
   Model |  182843.927     4  45710.9817        Prob > F      =  0.0000
Residual |  1283155.16  3260  393.605878        R-square      =  0.1247
---------+------------------------------       Adj R-square  =  0.1236
   Total |  1465999.09  3264  449.141878        Root MSE      =   19.84

Variable |  Coefficient    Std. Error       t    Prob > |t|        Mean
---------+--------------------------------------------------------------
  mhi3mo |                                                      71.36854
---------+--------------------------------------------------------------
   imale |     4.519749      .7416599     6.094     0.000      .3840735
iagecont |     .4287276      .0224545    19.093     0.000      53.66473
 married |     2.806007      .7322122     3.832     0.000      .5911179
 inonwht |      3.25354      .8643187     3.764     0.000      .2073507
   _cons |     44.29177      1.332665    33.235     0.000             1
---------+--------------------------------------------------------------
```

We see that the effect of being married is still significant, but not as large as we saw in the raw data. There are also positive effects for being male and nonwhite. Age is a strong positive predictor. The effect is about 17 points for 40 years of age, which is about 1 standard deviation for the mental health index. Since age has such a dramatic effect, it is important to measure its effect accurately and there is no reason to suspect that the effect is purely linear. One solution might be to include age squared, but there is also no reason to suspect a quadratic effect and, with this amount of data, it would be best to let the data "select" the functional form. One way is to decompose age into a set of splines:

```
. gen ages45 = max(0,iagecont-45)
. gen ages55 = max(0,iagecont-55)
. gen ages65 = max(0,iagecont-65)
```

These variables, along with `iagecont`, allow us to fit a connected set of line segments with hinges at 45, 55, and 65 years of age. If we include these four variables in a linear regression, the "age effect" is modeled as

$$E = \beta\texttt{iagecont} + \beta_{45}\texttt{ages45} + \beta_{55}\texttt{ages55} + \beta_{65}\texttt{ages65}$$

For persons less than age 45, the effect is simply $E = \beta\texttt{iagecont}$ because the other three variables are defined as zero, and the slope is β.

At age 45, `ages45` kicks in, taking on the value 0 (age 45), 1 (age 46), 2 (age 47), and so on. Thus, the age effect is $E = \beta\texttt{iagecont} + \beta_{45}\texttt{ages45}$. The line joins with the $\texttt{iagecont} < 45$ line at age 45 because `ages45` is zero there, but the slope (the change in E for a one-year change in age) is now $\beta + \beta_{45}$.

At age 55, the process repeats as `ages55` kicks in, taking on the value 0 (age 55), 1 (age 56), and so on. The age effect is $E = \beta\texttt{iagecont} + \beta_{45}\texttt{ages45} + \beta_{55}\texttt{ages55}$. Again the line joins at the age 55 hinge where `ages55` is zero, but the slope is now $\beta + \beta_{45} + \beta_{55}$.

The process repeats once more at age 65. We now estimate our regression, obtaining estimates for β, β_{45}, β_{55}, and β_{65}:

```
. reg mhi3mo imale iagecont married inonwht ages*
(obs=3265)
  Source |     SS         df       MS          Number of obs =    3265
---------+------------------------------       F(  7,  3257) =   67.63
   Model |  186044.231     7  26577.7473        Prob > F      =  0.0000
Residual |  1279954.86  3257  392.985833        R-square      =  0.1269
---------+------------------------------       Adj R-square  =  0.1250
   Total |  1465999.09  3264  449.141878        Root MSE      =  19.824

Variable |  Coefficient    Std. Error       t    Prob > |t|        Mean
---------+--------------------------------------------------------------
  mhi3mo |                                                      71.36854
---------+--------------------------------------------------------------
   imale |     4.555014      .7413871     6.144     0.000      .3840735
iagecont |     .3357814      .0832666     4.033     0.000      53.66473
 married |     2.698273      .7539907     3.579     0.000      .5911179
 inonwht |     3.342804      .8681187     3.851     0.000      .2073507
  ages45 |     .1939794      .2070005     0.937     0.349      11.87662
  ages55 |      .092175      .2676469     0.344     0.731      5.902229
  ages65 |    -.4430357      .2194174    -2.019     0.044      1.962236
   _cons |     47.33282      3.030649    15.618     0.000             1
---------+--------------------------------------------------------------
```

We can plot the function of age:

```
. gen ageeff = iagecont*_b[iagecont] + ages45*_b[ages45] +
                                       ages55*_b[ages55] + ages65*_b[ages65]
(7 missing values generated)
. graph ageeff iagecont, sy(.) connect(l) sort
```

We see (Figure 1) that the age effect gets a little stronger after 45 but flattens out around 65.

It is worth noting that a repeated measures ANOVA would not be very satisfying for determining cross-sectional effects. Although it would gain power from the longitudinal observations, it would also lose power because most observations are missing at least one time point and those observations would be unusable. In fact, we can find out how many observations would be lost:

```
. gen byte mhimiss = (mhi0==.) + (mhi3mo==.) + (mhi1yr==.) +
                     (mhi2yr==.) + (mhi4yr==.)
. tab mhimiss
    mhimiss|      Freq.     Percent        Cum.
-----------+-----------------------------------
         0 |        618       15.97       15.97
         1 |        909       23.49       39.47
         2 |        360        9.30       48.77
         3 |        854       22.07       70.85
         4 |        948       24.50       95.35
         5 |        180        4.65      100.00
-----------+-----------------------------------
     Total |       3869      100.00
```

There are only 618 observations (out of 3,869) with data defined for all time points. The repeated measures ANOVA approach would ignore 84% of our data! Nor would that be the end of our problems.

In this study, data are lost for three reasons: (1) one of the two baseline measures was not collected for a random half of the sample; (2) the original investigators threw out approximately half the sample between baseline and the first year follow-up; and (3) approximately 23% of the remaining sample was lost to follow-up. Repeated measures ANOVA cannot use observations with missing values. If we used that technique, we would have to be concerned with biases introduced by all three reasons for missing data and, in particular, by the observations lost to follow-up. (It might be more difficult to follow-up a patient with poor mental health.)

Even without repeated measures ANOVA, one can examine longitudinal effects by looking at specific changes over time. Below, I examine the change from baseline to 4 years out (`change0`) and the change from 3 months out to 2 years out (`change3`):

```
. gen change0 = mhi4 - mhi0
(3158 missing values generated)
. reg change0 imale iagecont married inonwht
(obs=688)
  Source |     SS         df       MS           Number of obs =     688
---------+------------------------------       F(  4,   683) =   14.35
   Model |  23724.4373     4  5931.10932        Prob > F      =  0.0000
Residual |  282236.805   683  413.231047        R-square      =  0.0775
---------+------------------------------       Adj R-square  =  0.0721
   Total |  305961.242   687  445.358431        Root MSE      =  20.328

Variable |  Coefficient    Std. Error       t    Prob > |t|        Mean
---------+------------------------------------------------------------
 change0 |                                                     4.854651
---------+------------------------------------------------------------
   imale |    -4.751704      1.666875    -2.851     0.004      .4171512
iagecont |    -.3240794      .0500617    -6.474     0.000      55.49571
 married |    -.9881201      1.653465    -0.598     0.550      .5930233
 inonwht |     -2.38699      2.129542    -1.121     0.263      .1613372
   _cons |     25.79293      3.035902     8.496     0.000             1
---------+------------------------------------------------------------

. gen change3 = mhi2 - mhi3mo
(2125 missing values generated)
. reg change3 imale iagecont married inonwht
(obs=1672)
  Source |     SS         df       MS           Number of obs =    1672
---------+------------------------------       F(  4,  1667) =   15.57
   Model |  16247.3659     4  4061.84149        Prob > F      =  0.0000
Residual |  434741.385  1667  260.792673        R-square      =  0.0360
---------+------------------------------       Adj R-square  =  0.0337
   Total |  450988.751  1671  269.891533        Root MSE      =  16.149
```

```
Variable |  Coefficient    Std. Error       t    Prob > |t|        Mean
---------+------------------------------------------------------------------
 change3 |                                                       2.003001
---------+------------------------------------------------------------------
   imale |    -1.339477      .8394873    -1.596     0.111      .4102871
iagecont |    -.1867027      .0251348    -7.428     0.000      55.79513
 married |    -.2844878      .8375503    -0.340     0.734      .5915072
 inonwht |    -1.863855      1.035882    -1.799     0.072      .1794258
   _cons |     13.47238      1.536624     8.768     0.000             1
---------+------------------------------------------------------------------
```

Repeated measures ANOVA in some sense is a summary of the changes between periods and how those changes differ with the independent variables. You can think of repeated measures ANOVA as taking all of the possible change regressions that one could run and weighting them somehow to produce a test statistic. Above are two of the regressions and they are similar (remember, the first regression is a change of four years while the second is only over two years, so effects in the second should be smaller). The similarity in the two regressions informs us that we can probably interpret any one of the regressions as reflecting overall trends. We find that the coefficients have reversed signs when compared to the cross-sectional regression. For instance, males start with higher mental health than females but, relative to females, their mental health declines over time.

One way to address the change over time is to compute the *slope* for each observation. That is, we have a mental health measurement at baseline, 3 months, and 1, 2, and 4 years. For each observation, we could estimate a regression fitting a straight line to the data. The regression itself does not interest us, but we could use the resulting estimate of the slope as a proxy for the change over time and we could then use the slope as the subject of our analysis. One major advantage is that we can compute slopes even in the presence of missing data.

There are pluses and minuses to using slopes. On the negative side, we must acknowledge that the proper relationship is not necessarily a constant slope over time. If this were intervention data, for example, we would anticipate a larger effect due to the intervention at earlier times and smaller ones later. If there is missing data, as there is in our case, we should weight the slopes in our subsequent analysis since they are not all computed using the same amount data. Slopes are also a somewhat inefficient estimate.

On the positive side, a slope is intuitive and, as pointed out, can be computed for any observation that has two or more data points. This gives us some options for dealing with missing data. I will discard observations which are not observed in at least one of the follow-up periods since any change score we could compute from them would depend only on baseline data. This said, I am going to compute the slopes. This turns out to be possible but tedious in Stata.

The formula for a slope is σ_{xy}/σ_x^2 where σ_{xy} is the covariance of y and x and σ_x^2 is the variance of x. In this case, y is the mental health measurement and x is the time at which the measurement was taken. We will calculate this ratio as $(\sum(x_i - \bar{x})y_i)/(\sum(x_i - \bar{x})^2)$. Remember that the previously calculated `mhimiss` is the number of missing mental health measurements:

```
. gen xbar = 0
. replace xbar = xbar + .25 if mhi3mo~=.        /* 3 months = .25 of a year */
(3415 changes made)
. replace xbar = xbar + 1   if mhi1yr~=.
(1891 changes made)
. replace xbar = xbar + 2   if mhi2yr~=.
(1822 changes made)
. replace xbar = xbar + 4   if mhi4yr~=.
(1455 changes made)
. replace xbar = xbar / (5-mhimiss)
(3869 changes made)

. gen x2 = 0
. replace x2 = x2 + (-xbar)^2 if mhi0~=.
(1879 changes made)
. replace x2 = x2 + (.25-xbar)^2 if mhi3mo~=.
(3415 changes made)
. replace x2 = x2 + (1-xbar)^2 if mhi1yr~=.
(1891 changes made)
. replace x2 = x2 + (2-xbar)^2 if mhi2yr~=.
(1822 changes made)
. replace x2 = x2 + (4-xbar)^2 if mhi4yr~=.
(1455 changes made)

. gen xy = 0
```

```
. replace xy = xy + (-xbar)*mhi0 if mhi0~=.
(1879 changes made)
. replace xy = xy + (.25-xbar)*mhi3mo if mhi3mo~=.
(3415 changes made)
. replace xy = xy + (1-xbar)*mhi1yr if mhi1yr~=.
(1891 changes made)
. replace xy = xy + (2-xbar)*mhi2yr if mhi2yr~=.
(1822 changes made)
. replace xy = xy + (4-xbar)*mhi4yr if mhi4yr~=.
(1455 changes made)
. gen slope = xy/x2 if mhimiss<4 & (mhi1yr~=. | mhi2yr~=. | mhi4yr~=.)
(1850 missing values generated)
```

We can now use `slope` as the subject of our analysis. The regression below is weighted by `x2` (which is proportional to the reciprocal of the variance of the slope) to account for the unequal sample sizes over which the slopes were calculated.

```
. reg slope married imale iagecont ages* inonwht =x2
(sum of wgt is   1.3459e+04)
(obs=1925)
  Source |     SS         df       MS           Number of obs =    1925
---------+------------------------------        F(  7,  1917) =   17.75
   Model |  3090.71465     7  441.530664        Prob > F      =  0.0000
Residual |  47690.8921  1917   24.877878        R-square      =  0.0609
---------+------------------------------        Adj R-square  =  0.0574
   Total |  50781.6067  1924  26.3937665        Root MSE      =  4.9878
Variable |  Coefficient    Std. Error       t    Prob > |t|        Mean
---------+-------------------------------------------------------------
   slope |                                                      .8629254
---------+-------------------------------------------------------------
 married |    -.3264564      .2480897    -1.316     0.188      .5914187
   imale |    -.6309712      .2438765    -2.587     0.010       .409048
iagecont |    -.0243468      .0301197    -0.808     0.419      55.76643
  ages45 |    -.0271835      .0727632    -0.374     0.709      13.50555
  ages55 |    -.1099776      .0904884    -1.215     0.224      7.085644
  ages65 |      .107903      .0701864     1.537     0.124       2.46074
 inonwht |    -.7529708      .3063881    -2.458     0.014      .1713754
   _cons |     3.681739      1.106394     3.328     0.001             1
---------+-------------------------------------------------------------

. test ages45 ages55 ages65
 ( 1)  ages45 = 0.0
 ( 2)  ages55 = 0.0
 ( 3)  ages65 = 0.0
       F(  3,  1917) =    2.36
            Prob > F =    0.0686
```

Since the F test is insignificant, I will dispense with the spline terms:

```
. reg slope married imale iagecont inonwht =x2
(sum of wgt is   1.3459e+04)
(obs=1925)
  Source |     SS         df       MS           Number of obs =    1925
---------+------------------------------        F(  4,  1920) =   29.23
   Model |  2914.63057     4  728.657642        Prob > F      =  0.0000
Residual |  47866.9762  1920  24.9307167        R-square      =  0.0574
---------+------------------------------        Adj R-square  =  0.0554
   Total |  50781.6067  1924  26.3937665        Root MSE      =  4.9931
Variable |  Coefficient    Std. Error       t    Prob > |t|        Mean
---------+-------------------------------------------------------------
   slope |                                                      .8629254
---------+-------------------------------------------------------------
 married |    -.2398817      .2423235    -0.990     0.322      .5914187
   imale |    -.6003657      .2437838    -2.463     0.014       .409048
iagecont |     -.073299      .0073438    -9.981     0.000      55.76643
 inonwht |     -.678705      .3049193    -2.226     0.026      .1713754
   _cons |     5.454313      .4483571    12.165     0.000             1
---------+-------------------------------------------------------------
```

As we found when we ran the specific change regressions, males relative to females have declining mental health over time. We started by wondering whether marriage might improve mental health over time (real effect) or if instead persons with

better mental health merely tend to be married (manifestation). Our results are consistent with the manifestation hypothesis (the estimated coefficient is negative) but they do not exclude a positive coefficient because of the insignificance of the measured effect.

The age effect, however, is more difficult to disentangle because the patients age over time. Note that the average `slope` is positive, which means that most people in the sample have increasing mental health over time or, said differently, as they age. This supports a causal hypothesis over a cohort difference hypothesis. The age coefficient, on the other hand, is negative, so the impact of increasing mental health with age must dampen with age. That was consistent with the cross-sectional results, even though they were not significant!

There is an important point here. Let's pretend that our data had been "clean" in that there were no missing values and repeated measures ANOVA had been a real alternative. One should feel uncomfortable performing a repeated measures ANOVA analysis at the outset. For instance, there was no guarantee that the age effects we found would be consistent and, had they not been, repeated measures ANOVA would have oversimplified the problem. If one has complete data, repeated measures ANOVA can well be a convenient summary device after one has verified the assumptions using the techniques of the sort outlined above. And if one does not have complete data, repeated measures ANOVA is an even less attractive alternative.

The techniques outlined above are hardly a definitive treatise on the analysis of this sort of data. I invite readers to comment on how they would analyze such data and we will follow up in a later issue.

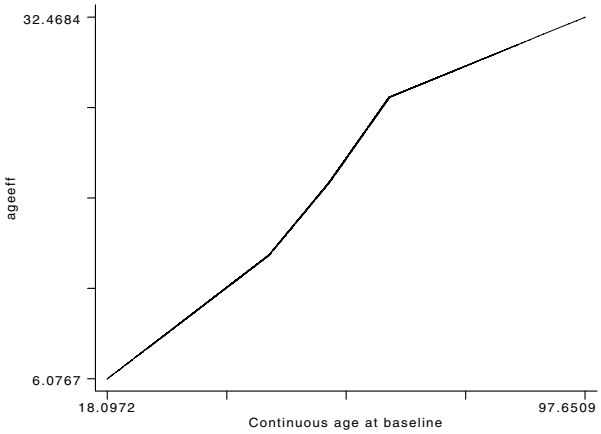

Figure 1

References

Tarlov, A. R., J. E. Ware, S. Greenfield, E. C. Nelson, E. Perrin, and M. Zubkoff. 1989. The medical outcome study. *Journal of the American Medical Association* 262: 925–930.

sqv1.3 An enhanced Stata logistic regression program

Joseph Hilbe, Editor, STB, FAX 602-860-1446

`logiodd2` has been corrected to provide m-asymptotic influence statistics as described by Hosmer and Lemeshow (1989). The new `e` option has been modified to provide only the basic goodness-of-fit statistics, Wald statistics, and partial correlations, and the new `i` option provides the influence and residual statistics. Unfortunately, in the current version, `i` can only be specified when there are 10 or fewer independent variables.

The important difference is that `logiodd2` now adjusts for the number of covariate patterns of the independent variables. For example, the data set

```
y     x1   x2   x3
1     1     0    1
1     1     1    0
0     0     1    1
0     1     0    1
1     1     1    0
```

consists of five observations but only three covariate patterns. The residual and influence statistics are a function of the number of such patterns in the data set, the number of observations sharing the same covariate pattern, and the number of positive responses within each pattern.

Statistics calculated by the `i` option are stored in variables named `presid`, `hat`, `stpresid`, etc. Here is a listing of the additional diagnostic variables created.

```
logindex =  Logit; Index value
sepred   =  Standard error of index
pred     =  Probability of success (1)
mpred    =  Prob of covariate pattern success
presid   =  Pearson Residual
stpresid =  Standardized Pearson Residual
hat      =  Hat matrix diagonal
dev      =  Deviance
cook     =  Cook´s distance
deltad   =  Change in Deviance
deltax   =  Change in Pearson chi-square
deltab   =  Difference in coefficient due to
            deletion of observation and others
```

The formulas for each are given below, although the interested reader is directed to Hosmer and Lemeshow (1989) for a detailed discussion. Also see Hamilton (1992).

Stored in `presid` is $r_j = (y_j - m_j pred_j)/\sqrt{m_j pred_j(1-pred_j)}$ where j represents the observation number, m_j the number of observations sharing j's covariate pattern, and y_j the number of positive responses within the covariate pattern of which j is a member. $pred_j$ is the predicted probability of a positive outcome. Note that this residual is the same for all observations sharing the same covariate pattern.

`mpred` contains $m_j pred_j$, which is the expected number of positive responses for observations sharing j's covariate pattern.

`hat` contains $h_j = (sepred_j^2)(m_j pred_j)(1-pred_j)$, where `sepred` represents the standard error of index.

`stpresid` contains $r_{sj} = r_j/\sqrt{1-h_j}$.

`dev` contains

$$d_j = \pm\sqrt{2\left(y_j \ln\left(\frac{y_j}{m_j pred_j}\right) + (m_j - y_j)\ln\left(\frac{(m_j - y_j)}{m_j(1-pred_j)}\right)\right)}$$

The sign of the deviance residual is identical to the sign of $(y_j - m_j pred_j)$. For observations having only a single covariate pattern $d_j = \sqrt{-2\ln pred_j}$ for observed positive responses and $d_j = -\sqrt{-2\ln(1-pred_j)}$ for observed negative responses.

`cook` contains $r_j^2 h_j/(1-h_j)$.

`deltax` contains $r_j^2/(q-h_j) = r_{sj}^2$.

`deltad` contains $d_j^2 + ((r_j^2 h_j)/(1-h_j))$.

`deltab` contains $r_j^2 h_j/(1-h_j)^2$.

The Pearson χ^2 statistic is $\sum r_j^2$ and the deviance statistic is $\sum d_j^2$.

Hosmer and Lemeshow suggest the following four diagnostic plots:

```
. gr deltax pred, xlab ylab yline(4)
. gr deltad pred, xlab ylab yline(4)
. gr deltab pred, xlab ylab yline(1)
. gr deltax pred=deltab, xlab ylab yline(4)
```

Observations or covariate patterns whose values exceed `yline` are considered significantly influential.

An Example

Hosmer and Lemeshow present a full model example based on a study of low birth-weight babies. The following is part of the output.

```
. logiodd2 low age race2 race3 smoke ht ui lwd ptd inter1 inter2, i

Number of Predictors         =        10
Number of Non-Missing Obs    =       189
Number of Covariate Patterns =       128
Pearson X2 Statistic         =       137.7503
  P>chi2(117)                =         0.0923
Deviance                     =       147.3371
  P>chi2(117)                =         0.0303
```

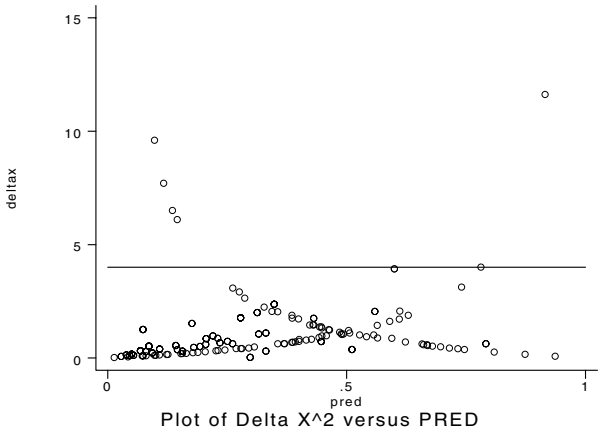

Figure 1

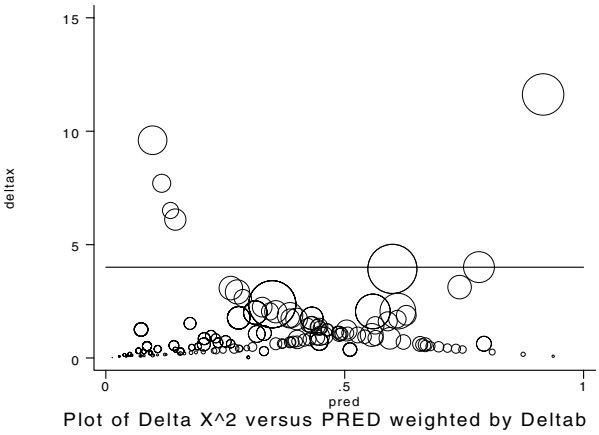

Figure 2

References

Hamilton, L. C. 1992. *Regression with Graphics.* Pacific Grove, CA: Brooks/Cole Publishing Company.

Hosmer, D. W. and S. Lemeshow. 1989. *Applied Logistic Regression.* New York: John Wiley & Sons.

ssi2	Bootstrap programming

Lawrence C. Hamilton, Dept. of Sociology, University of New Hampshire

Bootstrapping refers to a process of repeatedly sampling (with replacement) from the data at hand. Instead of trusting theory to tell us about the sampling distribution of an estimator b, we approximate that distribution empirically. Drawing B bootstrap samples of size n (from an original sample also size n) obtains B new estimates, each denoted b^*. The bootstrap distribution of b^* forms a basis for standard errors or confidence intervals (Efron and Tibshirani, 1986; for an introduction see Stine in Fox and Long, 1990). This empirical approach seems most attractive in situations where the estimator is theoretically intractable, or where the usual theory rests on untenable assumptions.

Bootstrapping requires fewer assumptions but more computing than classical methods. The January 1991 *Stata News* (p.6–7) described two general bootstrap programs, `bootsamp.ado` and `boot.ado`. Help files document these programs, which provide a relatively easy way to start bootstrapping. Even with these ready-made programs, however, users must do some programming themselves and know exactly what they want. This can be tricky: bootstrapping is fraught with nonobvious choices and with "obvious" solutions that don't work. Researchers have the best chance of successful bootstrapping when they can write programs to fit specific analytical needs. Towards this goal I reinvent the wheel below, showing the construction of several simple bootstrap programs. Rather than being general-purpose routines like `boot.ado` or `bootsamp.ado`, these four examples are problem-specific but illustrate a general, readily modified approach.

The first three examples expect to find raw data in a file named `source.dta`, with variables called X and Y. For illustration I employ data from Zupan, 1973, on the population density (X) and air pollution levels (Y) in 21 New York counties:[1]

```
          county        X        Y
 1.     New York    61703.7     .388
 2.        Kings   38260.87     .213
 3.        Bronx   33690.48     .295
 4.       Queens   17110.09     .307
 5.       Hudson   13377.78     .209
 6.        Essex   7382.813     .142
 7.      Passaic   2284.946     .054
 8.        Union   5116.505     .161
 9.       Nassau   4560.403      .15
10.  Westchester   1921.839     .072
11.     Richmond   4034.483     .059
12.       Bergen   3648.069     .112
13.    Middlesex   1597.444     .076
14.    Fairfield   1583.113     .065
15.    New Haven   1149.425     .053
16.      Suffolk   1329.114     .072
17.     Rockland    905.028     .052
18.     Monmouth   781.9706    .0325
19.     Somerset   527.6873     .029
```

```
 20.       Morris   683.7607      .0316
 21.       Mercer    1228.07       .049
```

A Simple Bootstrap

`example1.ado` performs *data resampling*, the simplest kind of bootstrap. From an original sample with n cases, we draw bootstrap samples (also size n) by random sampling with replacement. This is accomplished by letting Stata's random-number function `uniform()` choose the observation numbers (explicit subscripts) of cases included in each bootstrap sample. As written, `example1.ado` executes B=1,000 iterations—adequate for standard-error estimation but probably too few for confidence intervals. Number of iterations, variable names, and other features can easily be changed or generalized. Comment lines (beginning with *) briefly explain what the program is doing.

```
program define example1
*   The first line tells Stata we are going to define a program
*   named "example1".  This program bootstraps the mean of a
*   variable named "X", from a dataset called "source.dta".
*   To apply example1.ado to your own data:
*
*           . use <yourfile.dta>
*           . rename <yourvar> X
*           . keep if X~=.
*           . save source, replace
*
set more 1
*   Tells Stata to wait only 1 second before scrolling a full
*   screen.  Default:  waits for keyboard input before scrolling.
drop _all
capture erase bootdat1.log
set maxobs 2000
*   For confidence intervals or other applications using
*   bootstrap-distribution tail percentiles, at least B=2,000
*   bootstrap iterations are needed.  Simpler purposes, including
*   standard error estimation, require substantially fewer
*   iterations.
*   If source.dta contains > 2,000 cases, set maxobs higher.
log using bootdat1.log
log off
*   Log file bootdat1.log will record bootstrap results.
set seed 1111
*   Sets the random-generator seed.  We can repeat the random
*   sequence later by using the same seed, or avoid repeating it
*   by choosing a different seed (any large odd number).
macro define _bsample 1
*   _bsample counts the number of bootstrap samples.
*   _bsample is the name of this macro; %_bsample refers to
*   the macro´s current contents:
while %_bsample<1001 {
     quietly use source.dta, clear
     quietly drop if X==.
     quietly generate XX=X[int(_N*uniform())+1]
     *   Variable XX holds randomly resampled X values.  The
     *   expression int(_N*uniform())+1 generates random integers
     *   from 1 through  _N (sample size).
     quietly summarize XX
     log on
     display %_bsample
     display _result(3)
     display
     log off
     *   For each bootstrap sample, the log file contains the
     *   sample number and mean of XX.
     macro define _bsample=%_bsample+1
     }
     *   Curly brackets enclose "while %_bsample<1001" loop.
log close
drop _all
infile bsample bmean using bootdat1.log
label variable bsample "bootstrap sample number"
label variable bmean "sample mean of X"
label data "bootstrap mean"
```

```
save boot1.dta, replace
*   Final steps read the log file "bootdat1.log", label
*   variables, and save dataset boot1.dta containing means from
*   1,000 bootstrap resamplings of the original source.dta data.
end
```

When debugging or modifying ado-files, type `program drop _all` between run attempts, so that Stata will forget any previous buggy versions.

Applied to New York population density, `example1.ado` generates the bootstrap distribution graphed in Figure 1. When the data contain outliers, bootstrapping often produces odd-looking sampling distributions—a thought-provoking antidote to routine normality assumptions. Bootstrapping here also yields a somewhat lower standard error estimate, but no evidence of bias:

```
                              mean            standard error
original sample               9661                 3474
bootstrap--data resampling    9662                 3395
```

What else can we conclude from the Figure 1 results? Several authors have recommended using bootstrap percentiles directly as confidence-interval bounds. For example, one might form a "90% confidence" interval from the bootstrap 5th and 95th percentiles. Unfortunately, this often works poorly.

Peter Hall (1988) observes that if the sampling distribution is asymmetrical (like Figure 1), using 5th and 95th percentiles as low and high confidence-interval endpoints is "backwards." For example, 90% of sample b values fall between the 5th ($b_{.05}$) and 95th ($b_{.95}$) percentiles of b's sampling distribution:

$$b_{.05} < b < b_{.95} \qquad [1a]$$

Writing [1a] as a distance above and below the true parameter β:

$$\beta + (b_{.05} - \beta) < b < \beta + (b_{.95} - \beta) \qquad [1b]$$

Confidence intervals rearrange this inequality to isolate β:

$$b - (b_{.95} - \beta) < \beta < b - (b_{.05} - \beta) \qquad [2]$$

This suggests a better ("hybrid") bootstrap-percentile confidence interval formula:

$$b - (b^*_{.95} - b) < \beta < b - (b^*_{.05} - b) \qquad [3]$$

where b is the original sample statistic, and $b^*_{.95}$ and $b^*_{.05}$ represent bootstrap 95th and 5th percentiles.

Monte Carlo research finds that with or without this asymmetry correction, bootstrap-percentile confidence intervals often achieve less than nominal coverage. Strategies for improvement include *accelerated bias correction* (BC_a) and *percentile-t* methods. The simpler of the two, percentile-t, first obtains studentized values:

$$t^* = (b^* - b)/SE^*_b \qquad [4]$$

then uses bootstrap percentiles of t^* to form confidence intervals, for example:

$$b - t^*_{.95}SE_b < \beta < b - t^*_{.05}SE_b \qquad [5]$$

The standard error of b—SE_b—might be estimated from either the original sample or (better) from the bootstrap standard deviation.

Bootstrapping Regression

In many instances, bootstrapping a mean (as in `example1.ado`) has no advantage over inferences based on the Central Limit Theorem. Bootstrapping helps more with multivariable methods like regression, where the classic inferential procedures depend on a longer list of often-false assumptions. Some bootstrapping methods implicitly make similar assumptions, while others abandon them—obtaining quite different results.

The bootstrapping method of `example1.ado`, data resampling, generalizes to resampling entire cases. In two-variable regression, this means we resample (X,Y) pairs as in `example2.ado`.[2]

```
program define example2
*   data-resampling regression bootstrap
*   assumes variables "Y" and "X" in "source.dta"
*
set more 1
drop _all
set maxobs 2000
*   If source.dta contains > 2,000 cases, set maxobs higher.
quietly use source.dta
quietly drop if Y==. | X==.
save, replace
quietly regress Y X
macro define _coefX=_b[X]
*   _coefX equals the original-sample regression coefficient on X
capture erase bootdat2.log
log using bootdat2.log
log off
set seed 1111
macro define _bsample 1
while %_bsample<1001 {
     *   For confidence intervals or tests, we need 2000 or more
     *   bootstrap samples.
     quietly use source.dta, clear
     generate randnum=int(_N*uniform())+1
     quietly generate YY=Y[randnum]
     quietly generate XX=X[randnum]
     quietly regress YY XX
     *   The last three commands randomly resample (X,Y) pairs
     *   from the data.
     macro define _bSE=_b[XX]/sqrt(_result(6))
     log on
     display %_bsample
     display _b[_cons]
     display _b[XX]
     display %_bSE
     display (_b[XX]-%_coefX)/%_bSE
     *   Calculated either way, this command obtains a
     *   studentized coefficient:
     *   (bootstrap coef. - original coef.)/SE of bootstrap coef.
     display
     log off
     macro define _bsample=%_bsample+1
     }
log close
drop _all
infile bsample bcons bcoefX bSE stucoefX using bootdat2.log
label variable bsample "bootstrap sample number"
label variable bcons "sample Y-intercept, b0"
label variable bcoefX "sample coefficient on X, b1"
label variable bSE "sample standard error of b1"
label variable stucoefX "studentized coefficient on X"
label data "regression boot/data resampling"
save boot2.dta, replace
end
```

Figure 2 shows two distributions obtained by bootstrapping the regression of New York air pollution on population density. Data resampling (at top in Figure 2) does not make the usual regression assumptions of fixed X and independent, identically distributed (i.i.d.) errors. Consequently it often yields larger standard error estimates and skewed, multimodal sampling distributions. If the usual assumptions are false, we are right to abandon them, and bootstrapping may provide better guidance. If the assumptions are true, on the other hand, data resampling is too pessimistic.

Since it scrambles the case sequence, data resampling is also inappropriate with time or spatial series. We could get bootstrap time series in which 1969 appears three times, and 1976 not at all, for instance.

Residual resampling, an alternative regression bootstrap approach, retains the fixed-X and i.i.d.-errors assumptions. Residuals from the original-sample regression, divided by $\sqrt{1-K/\_N}$, are resampled and added to original-sample $\hat{Y}$ values to generate bootstrap Y^* values, which then are regressed on original-sample X values. `example3.ado` illustrates, using the same two-variable model as `example2.ado`. Results appear at bottom in Figure 2. Comments explain features new since `example2.ado`.

```
program define example3
*   residual resampling regression bootstrap
*   assumes variables "Y" and "X" in "source.dta"
*
set more 1
drop _all
set maxobs 2000
*   If source.dta contains > 2,000 cases, set maxobs higher.
quietly use source.dta
quietly drop if Y==. | X==.
quietly regress Y X
capture predict Yhat
capture predict e, resid
quietly replace e=e/sqrt(1-((_result(3)+1)/_result(1)))
*   Previous two commands obtain full-sample regression
*   residuals, and "fatten" them, dividing by:
*         sqrt(1 - K/_N)
*   where K is # of model parameters and _N is sample size.
macro define _coefX=_b[X]
quietly save, replace
capture erase bootdat3.log
log using bootdat3.log
log off
set seed 1111
macro define _bsample 1
while %_bsample<1001 {
     quietly use source.dta, clear
     quietly generate ee=e[int(_N*uniform())+1]
     quietly generate YY=Yhat+ee
     quietly regress YY X
     *   We resample residuals only, then generate bootstrap
     *   Y values (called YY) by adding bootstrap residuals (ee)
     *   to predicted values from the original-sample
     *   regression (Yhat).  Finally, regress these bootstrap
     *   YY values on original-sample X.
     macro define _bSE=_b[X]/sqrt(_result(6))
     log on
     display %_bsample
     display _b[_cons]
     display _b[X]
     display %_bSE
     display (_b[X]-%_coefX)/%_bSE
     display
     log off
     macro define _bsample=%_bsample+1
     }
log close
drop _all
infile bsample bcons bcoefX bSE stucoefX using bootdat3.log
label variable bsample "bootstrap sample number"
label variable bcons "sample Y-intercept, b0"
label variable bcoefX "sample coefficient on X, b1"
label variable bSE "sample standard error of b1"
label variable stucoefX "studentized coefficient on X"
label data "regression boot/residual resampling"
save boot3.dta, replace
end
```

To summarize our results in the regression of New York air pollution (Y) on population density (X):

	slope	standard error
original sample	$5.67 \cdot 10^{-6}$	$7.13 \cdot 10^{-7}$
bootstrap—data resampling	$6.24 \cdot 10^{-6}$	$21.0 \cdot 10^{-7}$
bootstrap—residual resampling	$5.66 \cdot 10^{-6}$	$7.89 \cdot 10^{-7}$

Since they both assume fixed X and i.i.d. errors, results from residual resampling resemble results from the original-sample regression (but with about 10% higher standard error). In contrast, data resampling obtains a standard error almost three times the original-sample estimate, and a radically nonnormal distribution (skewness=3.6, kurtosis=18.3) centered right of the original-sample regression slope. The differences in sampling distributions seen in Figure 2 dramatize how crucial the fixed-X and i.i.d. errors assumptions are.

But Does It Work?

The bootstrap's growing popularity derives partly from hope; its actual performance sometimes disappoints. Monte Carlo simulation provides one way to evaluate bootstrapping objectively. The simulation generates samples according to a known (user-designed) model; we then apply bootstrapping to discover (for example) how often bootstrap-based confidence intervals actually contain the model parameters. `example4.ado` does this, embedding data resampling within a Monte Carlo simulation.

At the heart of `example4.ado` is a misspecified regression model. The usual standard errors and tests assume:

$$Y = \beta_0 + \beta_1 X + \epsilon \qquad [6]$$

with X fixed in repeated samples, and errors (ϵ) normally, independently, and identically distributed (normal i.i.d.). But this Monte Carlo simulation generates data according to the model:

$$Y = 0 + 3X + X\epsilon \qquad [7]$$

with ϵ distributed as $\chi^2(1) - 1$. (Note that this has a mean of 0 and a variance of 2.) X values, drawn from a $\chi^2(1)$ distribution, vary randomly. In Figure 3, 5,000 data points illustrate the problematic nature of model [7]: it challenges analysis with leverage, outliers, skewed errors and heteroscedasticity. A Monte Carlo experiment drawing 10,000 random n=80 samples according to [7], and analyzing them by ordinary least squares (OLS) reveals a nasty-looking sampling distribution (Figure 4). As expected, OLS estimates are unbiased: the mean slope over 10,000 random samples ($\bar{b}$ = 2.99988) is indistinguishable from β=3. Otherwise, model [7] demolishes the usual OLS assumptions, and also those of residual resampling. Can data resampling still produce valid inferences?

`example4.ado` explores this question. As listed here it calls for 100, n=80 Monte Carlo samples, with B=2,000 bootstrap iterations per sample. (Results reported later represent 400 Monte Carlo samples, however.) For each Monte Carlo sample, it obtains "90% confidence" intervals based on standard t-table procedures and three bootstrap methods: using 5th and 95th percentiles; Hall's "hybrid" percentile-reversal method (equation [3]); and the studentized or percentile-t method (equation [5]). Finally, it calculates the width of each interval and checks whether the interval actually contains the parameter $\beta = 3$.

```
program define example4
*   Monte Carlo simulation of bootstrap confidence intervals for a misspecified
*   (heteroscedastic, nonnormal errors) regression model.  Generates 100 Monte
*   Carlo samples, and resamples each of them 2,000 times.
drop _all
set more 1
set maxobs 2100
set seed 33333
capture erase example4.log
macro define _mcit=1
while %_mcit<101 {
      quietly drop _all
      quietly set obs 80
      quietly generate X=(invnorm(uniform())) ^ 2
      quietly generate Y=3*X+X*((invnorm(uniform()) ^ 2)-1)
      *   Previous two lines define the true model.
      quietly regress Y X
      macro define _orb=_b[X]
      macro define _orSE=%_orb/sqrt(_result(6))
      *   Perform the original-sample regression, storing slope
      *   as _orb and standard error _orSE.
      quietly generate XX=.
      quietly generate YY=.
      quietly generate randnum=.
      macro define _bsample=1
      capture erase bstemp.log
      log using bstemp.log
      log off
      while %_bsample<2001 {
            *   Begin bootstrap iterations, indexed by _bsample.
            quietly replace randnum=int(_N*uniform())+1
            quietly replace XX=X[randnum]
            quietly replace YY=Y[randnum]
            quietly regress YY XX
            *   Data resampling, not assuming i.i.d. errors.
            log on
            display %_orb
            display %_orSE
            display _b[XX]
```

```
            display (_b[XX]-%_orb)/(_b[XX]/sqrt(_result(6)))
            *   Previous four lines display the original-sample regression slope
            *   (_orb), original-sample standard error (_orSE), bootstrap
            *   regression slope, and bootstrap studentized slope.
            log off
            macro define _bsample=%_bsample+1
            }
      log close
      drop _all
      infile orb orSE bootb stub using bstemp.log
      save mcit%_mcit, replace
      *   Each Monte Carlo iteration saves a dataset containing
      *   results from 2,000 resamplings.
      macro define _mcit=%_mcit+1
      }
macro define _mcit=1
capture erase mctemp.log
log using mctemp.log
log off
while %_mcit<101 {
      quietly use mcit%_mcit, clear
      log on
      display %_mcit
      quietly summ orb
      display _result(3)
      quietly summ orSE
      display _result(3)
      quietly summ bootb, detail
      display _result(3)
      display sqrt(_result(4))
      display _result(7)
      display _result(13)
      quietly summ stub, detail
      display _result(7)
      display _result(13)
      display
      log off
      *   Preceding lines display the Monte Carlo sample number, original-sample
      *   slope and standard error, bootstrap mean and standard deviation of
      *   regression slope; and bootstrap percentiles and studentized percentiles.
      macro define _mcit=%_mcit+1
      }
log close
drop _all
infile iterate orb orSE bootb bootSE p05 p95 t05 t95 /*
       */ using mctemp.log;
label variable orb "original sample b coefficient"
label variable orSE "original sample SE of b"
label variable bootb "mean bootstrap b coefficient"
label variable bootSE "SD of bootstrap b coefficient"
label define correct 0 "wrong" 1 "correct"
*
generate stanlo=orb-1.665*orSE
generate stanhi=orb+1.665*orSE
generate stanwide=stanhi-stanlo
label variable stanwide "width standard t interval"
generate stancor=0
replace stancor=1 if stanlo<3 & 3<stanhi
label variable stancor "standard 90% c.i. correct?"
label values stancor correct
*
generate perwide=p95-p05
label variable perwide "width 5th-95th percentile"
generate percor=0
replace percor=1 if p05<3 & 3<p95
label variable percor "percentile 90% c.i. correct?"
label values percor correct
*
generate hyblo=orb-(p95-orb)
generate hybhi=orb-(p05-orb)
generate hybwide=hybhi-hyblo
label variable hybwide "width hybrid percentile interval"
generate hybcor=0
replace hybcor=1 if hyblo<3 & 3<hybhi
```

```
label variable hybcor "hybrid 90% c.i. correct?"
label values hybcor correct
*
generate stulo=orb-t95*bootSE
generate stuhi=orb-t05*bootSE
generate stuwide=stuhi-stulo
label variable stuwide "width percentile-t interval"
generate stucor=0
replace stucor=1 if stulo<3 & 3<stuhi
label variable stucor "student 90% c.i. correct?"
label values stucor correct
*
label data "n=80 bootstrap/Monte Carlo"
label variable orb "Y=3X + Xe,e~chi2-1,X~chi2"
save boot4.dta, replace
end
```

`example4.ado` creates a dataset, `boot4.dta`, containing information on the width and inclusion rates of four types of "90% confidence" intervals: standard t-table, bootstrap percentile, hybrid bootstrap percentile (equation [3]), and bootstrap percentile-t intervals (equation [5]). Here are results based on 400 Monte Carlo samples:

```
Variable |   Obs        Mean    Std. Dev.       Min        Max
---------+--------------------------------------------------------
 iterate |   400       200.5    115.6143         1        400
     orb |   400    2.999104    .5820095   2.119488   5.551427
    orSE |   400      .17754    .0804133   .0574202   .4905333
   bootb |   400    3.003376    .5399735   2.153254    5.26923
  bootSE |   400    .4232904    .2956025   .0889773   1.708986
     p05 |   400    2.402226    .2288134   2.019951   3.410539
     p95 |   400    3.748615    1.005218   2.370434   7.698514
     t05 |   400   -6.176597    4.773216  -33.64367  -1.241619
     t95 |   400    3.456492    1.069656   1.383403   7.744884
  stanlo |   400      2.7035    .4828733   1.942544   4.943048
  stanhi |   400    3.294708    .6929325   2.242965   6.159807
stanwide |   400    .5912081    .2677761   .1912093   1.633476
 stancor |   400       .3475    .4767725          0          1
 perwide |   400    1.346389    .9071145   .2817495   5.096023
  percor |   400       .7625    .4260841          0          1
   hyblo |   400    2.249593    .4123866   .7103348   3.764181
   hybhi |   400    3.595982    1.019064    2.19773   8.523996
 hybwide |   400    1.346389    .9071145   .2817495   5.096023
  hybcor |   400        .615    .4872047          0          1
   stulo |   400    1.359049    1.242871  -5.784867   3.046353
   stuhi |   400    6.822549    7.106885   2.264905   53.62768
 stuwide |   400      5.4635    8.198419   .3112769   57.97469
  stucor |   400       .9025    .2970089          0          1
```

Means of `stancor`, `percor`, `hybcor`, and `stucor` indicate the proportion of "90% confidence" intervals that actually contained $\beta = 3$. Of course the standard t-table interval fails completely: only about 35% of these "90%" intervals contain the true parameter. The narrow intervals dictated by this method drastically understate actual sampling variation (Figure 5). Neither bootstrap percentile approach succeeds either, obtaining about 76% and 61% coverage. (Theoretically the hybrid-percentile method should work better than percentiles, but in experiments it often seems not to.) But the studentized or percentile-t method seemingly works: 90% of its "90% confidence" intervals contain 3.

The percentile-t method succeeds by constructing much wider confidence intervals, which more accurately reflect true sampling variation. The median width of percentile-t intervals is 2.66, compared with only .59 for standard t-table intervals. The mean percentile-t interval width (5.46) reflects the pull of occasional extremely wide intervals, as seen in Figure 6.

In Hamilton (1992), I report on another OLS experiment using a somewhat less pathological regression model. There too, bootstrap percentile-t methods achieved nominal coverage rates (over 1,000 Monte Carlo samples) when other methods did not. That discussion includes a closer look at how studentization behaves in the presence of outliers. Bootstrap confidence intervals based on robust estimators and standard errors (for example, see Hamilton 1991b) might achieve equally good coverage with narrower intervals—one of many bootstrap/Monte Carlo experiments worth trying.

Warning: even with just 100 Monte Carlo samples and 2,000 bootstrap resamplings of each, `example4.ado` requires hours of computing time and over three megabytes of disk space. Scaled-down experiments can convey a feel for such work, and explore promising new ideas. For example, change the two "`while %_mcit<101 {`" statements to "`while %_mcit<51 {`" and change "`while %_bsample<2001 {`" to "`while %_bsample<101 {`". Full-scale efforts might easily require four million iterations per model/sample size (2,000 bootstrap resamplings for each of 2,000 Monte Carlo samples), tying up a desktop

computer for weeks. The value of such work lies in the possibility of finding applications where bootstrapping solves (or less gratifyingly, fails to solve) otherwise intractable problems.

Notes

1. X represents population density in people per square mile. Y represents metric tons of NOx emissions per square mile. The more crowded (also poorer) areas are, of course, more polluted.
2. The mean bootstrap slope estimates are given.
3. Bootstrap standard errors are standard deviations of the bootstrap distributions; original-sample SE is that printed by `regress`.

References

Efron, B. and R. Tibshirani. 1986. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science* 1(1): 54–77.

Hall, P. 1988. Theoretical comparison of bootstrap confidence intervals. *The Annals of Statistics* 16(3): 927–953.

Hamilton, L. C. 1991a. ssi1: Monte carlo simulation. *Stata Technical Bulletin* 1: 25–28.

——. 1991b. srd1: How robust is robust regression? *Stata Technical Bulletin* 2: 21–26.

——. 1992. *Regression with Graphics: A Second Course in Applied Statistics.* Pacific Grove, CA: Brooks/Cole.

Stine, R. 1990. An introduction to bootstrap methods: examples and ideas. In *Modern Methods of Data Analysis*, ed. J. Fox and J. S. Long, 353–373. Newbury Park, CA: Sage Publications.

Zupan, J. M. 1973. *The Distribution of Air Quality in the New York Region.* Baltimore: Johns Hopkins University Press.

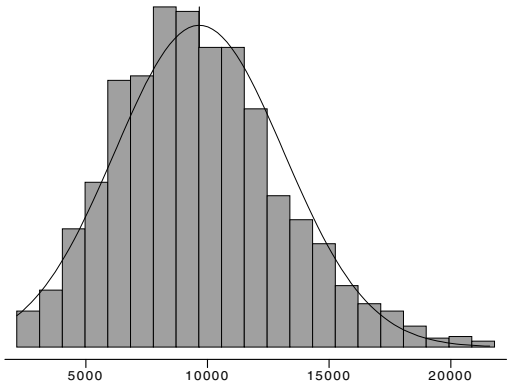

Figure 1: Means from 1,000 bootstrap samples

Figure 1

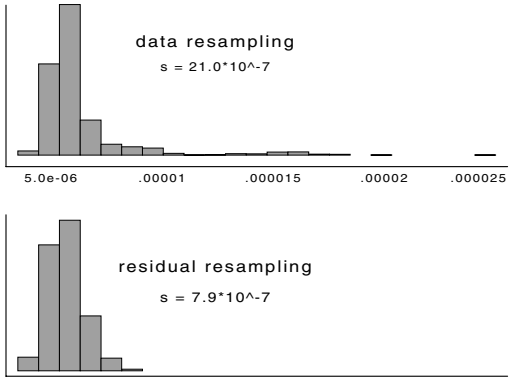

Figure 2: Regression slopes from 1,000 bootstrap samples

Figure 2

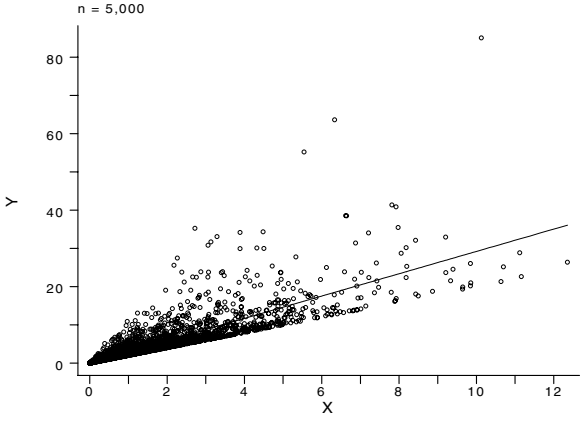

Figure 3: A problematic regression--model [7]

Figure 3

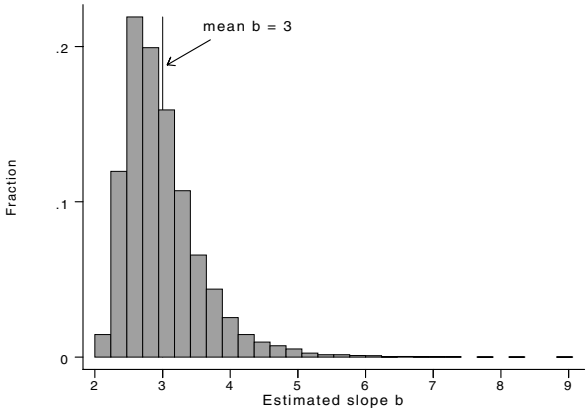

Figure 4: 10,000 n=80 Monte Carlo samples, model [7]

Figure 4

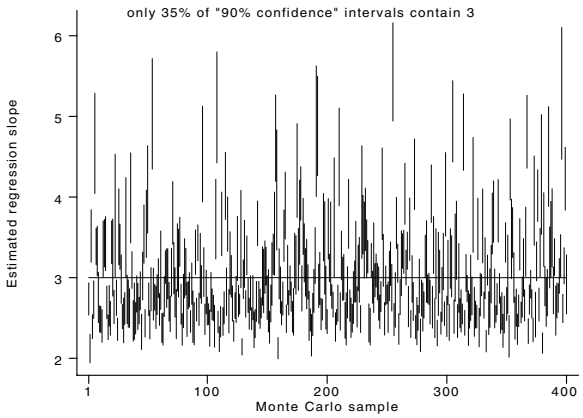

Figure 5: Standard t-table 90% confidence intervals

Figure 5

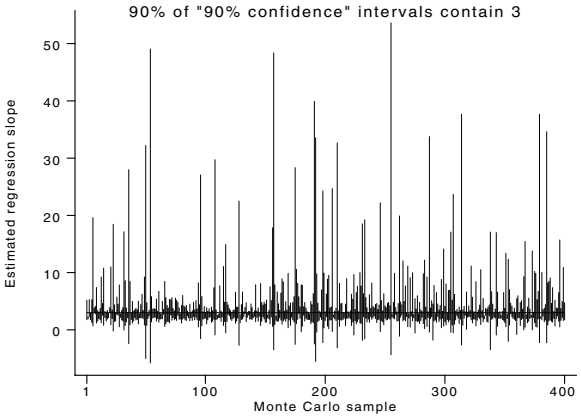

Figure 6: Bootstrap percentile-t 90% intervals

Figure 6

tt1	Teaching beginning students with Stata

Ted Anagnoson, California State University, LA & Richard DeLeon, San Francisco State University

Problem and Solution

Many instructors face a situation where students have to do basic data analysis, but their keyboarding skills may be rudimentary, their interest in learning how to change directories or `set maxobs` may be limited, etc. Here is a system we found in the Sociology/Political Science computer lab at the University of California, Santa Barbara, which was used in a first year class of 300 students who came in on their own time to a 40-station microcomputer lab to run Stata and do frequency distributions, and two- and three-way cross-tabulations. Our thanks to Joan Murdoch, Associate Director of the lab, for passing along this idea.

Every microcomputer had a batch file which automatically changed to the subdirectory for the class with the class's data file(s) in it and ran Stata with a `profile.do` file implied. An example of `pols480.bat`:

```
cd d:\pols480
d:\stata\stata run profile
```

A `profile.do` file is placed in the `\POLS480` (class) subdirectory which `sets maxobs` for the particular dataset, uses it, and sets up a log file. At UCSB each microcomputer has its own printer, and `log using prn:` is used to print results as each student goes along. `profile.do`:

```
set maxobs 1500
use campfin     [or whatever file is being used]
log using prn:
```

Thus each student turns on the computer, types `pols480` and finds that Stata is loaded, the data file is already in memory, no memory reconfiguration is necessary, and a log file is going to print all results on the printer.

Stata's do-files can be used to abbreviate the `tabulate` command so that students need not enter options, etc. Some examples are

```
Filename          Contents
freq.do           tab %_1
cross2.do         tab %_1 %_2, row col chi2
cross3.do         sort %_3
                  by %_3: tab %_1 %_2, row col chi2
```

Thus a typical student session might consist of only three or four commands:

```
C:\> pols480              [from the DOS prompt]
   . des                  [in Stata]
   . do freq varname
   . do cross2 var1 var2
   . do cross3 var1 var2 var3
   . exit
```

Comments and a Policy Problem

What little machine manipulation is necessary with Stata can be insulated from beginning students with a system like this. But do we really want to do this? And under what circumstances? Clearly, at one end of a continuum, we have classes that are large, where the central theme of the class is not data analysis, in disciplines where the students' quantitative and computer skills are not well developed. Here a system like the one above can save instructors time and energy and under some circumstances save the instructor from complete insanity.

On the other hand, where the class is small, where the central theme of the class is data analysis, where discipline or major requires data analytic skills as prerequisite to the class or uses them extensively, what is the justification for a system which keeps students from becoming familiar with the kinds of problems and situations they are likely to see if they attempt to practice data analysis for their employer using the employer's microcomputer and a purchased copy of Stata?

Anagnoson and DeLeon differ on this issue. DeLeon feels there is little justification for insulating students from the machine, especially with microcomputers and Stata. Anagnoson definitely sees a need for insulation of students in the first situation above, but feels that in the second situation, insulation is inappropriate.

Our system above is relatively primitive and easy to implement. One can go further and buy menu driven packages which insulate students from the need to type commands. Two such packages are MicroCase and MIDAS. Since the latter uses Stata as its statistical/data analysis "engine", we have asked Professor Michael Macy of the Department of Sociology at Brandeis University, the author of MIDAS and the author of several papers on the need for a new system of teaching statistics, for his comments. They are to be found in *tt2*.

Other comments and suggestions are welcome. Send them to Joe Hilbe, STB Editor, and he will pass them on to us. Other do- or ado-files that make Stata easier to use for students are welcome.

tt2	Using "front ends" for Stata

Michael Macy, Dept. of Sociology, Brandeis University

There is no general pedagogic principal that governs the "insulation" of students from statistical software and command syntax. The use of an interface between the student and the command line depends entirely on the objectives of the instructor. If the goal is to train students to use sophisticated stand-alone statistical software, to learn the mechanics of data management and the mathematics behind computer routines, "friendly" programs like MIDAS, CHIP, or MicroCase may be inappropriate. Indeed, when I teach introductory statistics, I have my students start out with MIDAS but I expect them to quickly move on to Stata.

However, many instructors, particularly those in the social sciences, tend to have different priorities. Their goal may be to incorporate a laboratory component into a course that addresses issues for which data analysis may be a useful complement to readings and lectures. Where the objective is to introduce liberal arts students to quantitative reasoning in an applied setting, the use of stand-alone statistical packages may be counterproductive. Students are likely to be frustrated by the usual pitfalls that await the novice researcher: arcane computer syntax, didactic and mathematical vocabulary, the disappearance of all their cases through the conjunction of skip patterns, meaningless correlations between nominal variables, cross-tabulation of continuous measures, etc. Rather than providing a highly motivating reinforcement for an introduction to quantitative reasoning, "hands-on research" can become a burdensome and unsatisfying experience.

Faced with certain disaster, the instructor then has little choice but to develop lab exercises that simply walk the student through the demonstration of the intended result—exercises that read rather like cookbook recipes, with no opportunity for genuine inquiry. The problem is that students quickly grow tired of rote exercises and lockstep instructions. Once the novelty wears off, most lab exercises tend to read like those dreadful instruction sheets included with children's toys that carry the warning label "Some assembly required."

Dissatisfied with current lab curricula, I developed MIDAS as a "front end" for Stata. MIDAS consists of a command file (written in PASCAL) and a Stata "do-file" that handshake with one another. The idea is essentially the same as Anagnoson and DeLeon's housekeeping programs, but I have simply pushed the principal a bit further. I wanted to not only simplify the commands but to provide a structured research environment in which students could chart their own course without falling off the edge of the earth. MIDAS does this by altering its menu-choices on the fly, depending on the user's cumulative decisions. As students gain confidence and savvy, MIDAS lets them "go behind" the interface and analyze their data directly with Stata, using the command line. Indeed, that is my "hidden agenda!"

I suspect that readers who experiment with "housekeeping" do-files will end up doing the same thing I did as MIDAS evolved...adding new features and routines. For those seeking a shortcut, I am happy to send copies of the MIDAS do-file to readers who want to create their own "front ends" for Stata. Better yet, I will send you, on spec, the entire MIDAS program to try.