STATA
TECHNICAL
BULLETIN

Sept 1992
STB-9

A publication to promote communication among Stata users

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

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

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

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

Contents of this issue

page

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.

an23 | CPS labor extracts available

Daniel Feenberg, National Bureau of Economic Research, feenberg@nber.harvard.edu.

The National Bureau of Economic Research has recently prepared a CD-ROM diskette including a series of extracts covering 13 years (1979 through 1991) from the Current Population Survey Outgoing Rotation Group Annual Merge Files. These are presented as Stata binary files. The annual files include interviews for everyone in a CPS outgoing rotation group during a single calendar year, or about 300,000 observations per year. To keep file sizes within reason for Stata users, each year of data is divided into three files. Only a few minutes are required to load each file, however. It is possible to leave a do-file running that will process all 13 years of data in a few hours, without operator intervention!

The extracts contain information for respondents who were 16 or older. The 50 or so variables selected for the extracts relate to employment: hours worked, earnings, industry, occupation, education, unionization. The extracts also contain many background variables: age, sex, race, ethnicity, geographic location, etc. Every effort has been made to keep the variables consistent over all the years. Users should note, however, that unionization variables are available for 1983 and after, student enrollment status is available for 1984 and after, metropolitan/central city variables undergo several changes in 1985 (e.g., SMSA status becomes metropolitan status) and unedited variables may contain spurious data for not-in-universe observations.

Also included on the CD-ROM is the complete 1991 Merged Outgoing Rotation Group file, as supplied by the BLS, but converted to ASCII and with DOS end-of-line characters after each record. This file is 292,590,662 bytes.

To read the extracts, a user will need a computer capable of reading ISO 9660 disks (any drive for IBM-PC, Mac or Unix workstation should be fine), 16 megabytes of memory and, of course, a 32-bit Stata. The diskette itself is available for $100 from:

Publications Department
National Bureau of Economic Research
1050 Mass Ave
Cambridge, MA 02138
Tel: 617-868-3900
Fax: 617-868-2754

Questions should be directed to feenberg@nber.harvard.edu. The 1991 extract is available for anonymous ftp from that address.

Variable list for 1991 extract:

Month in sample	Age
State	Marital status
Central city status	Race
MSA/PMSA FIPS code	Major activity last week
PMSA ranking	How many hours last week?
CMSA/MSA ranking	Reason <= 35 hours last week
MSA/CMSA size	Why absent from work last week?
CMSA code	3-digit industry code (1980)
Metropolitan status code	3-digit occupation code (1980)
Individual central city code	Class of worker 1
Household ID	Usual hours
Sex	Paid by the hour
Veteran	Union member
Highest grade attended	Ethnicity
Whether completed highest grade	Labor force status recode
What was doing most of last week	Full-time or part-time status
How many hours last all jobs	Detailed industry code
Usually works >= 35 hrs at this job	Detailed occupation code
Why not at least 35 hours last week	(Earnings) eligibility flag
Class of worker	Class of worker 2
Usual hours	Earnings per hour
Paid by the hour	Earnings per week
Earnings per hour	Final weight
Usual earnings per week	Earnings weight for all races
Union member	Usual hours (I25a) allocation flag
Covered by a union contract	Paid by hour (I25b) allocation flag
Enrolled student full/part time	Earnings/hr (I25c) allocation flag
Relationship to reference person	Usl Earn/hr (I25d) allocation flag

crc17 Bug fixes

For the most up-to-date list of new features and bug fixes, type '`help whatsnew`' after installing the CRC updates. Also remember, CRC updates are cumulative—if you have not installed past updates, it does not matter. The most recent updates include all past updates.

Fixed as of STB-7, but not reported due to publication deadlines, were the following:

1. (`cc`, `cs`, `mcc` in [5s] epitab.) Reported results were (obviously) incorrect when frequency weights were specified, some of the frequencies were 0, and the sum of the frequencies was 0 for one or more cells.
2. (`kwallis` in [5s] kwallis.) The `if` *exp* was ignored.

Fixed this time (or things to note) are

3. (`qreg` in [5s] qreg.) Despite the syntax diagram, `qreg` does *not* allow the `noconstant` option.
4. (`codebook` in *crc13* of STB-8.) A variable containing only zeros and missing values would cause `codebook` to loop endlessly (until *Break* was pressed).

crc18 Important difference between regular and Intercooled Stata

`codebook` (*crc13* in STB-8) emphasizes an important difference between the regular and Intercooled (which includes Unix) versions of Stata. The maximum number of variables that can be specified, either explicitly or implicitly, with `codebook` (and all other ado-commands) is 28 when running the regular version of Stata and 255 when running the Intercooled version. These maximums apply only to commands written as ado-files, not to Stata's built-in commands, and actually, both maximums are probably larger than 28 and 255. The difference between the two versions, however, is always present.

Let us explain: What is true is that the maximum length of a macro is 255 characters in regular Stata and 2,296 characters in Intercooled Stata. Among other things, ado-files use macros to store the names of the variables you specify or imply. The names are unabbreviated and a single blank is inserted between the names. The maximum length of a variable name in Stata is 8 characters, meaning that if all variable names were 8-characters long, a single macro could hold $\lfloor 255/9 \rfloor = 28$ names in regular Stata and $\lfloor 2296/9 \rfloor = 255$ names in Intercooled. ($\lfloor x \rfloor$ refers to the largest integer $k \leq x$ and is called the floor of x.) It is unlikely that all variables are exactly 8-characters long. If variable names averaged, say, 4 characters, a macro could hold $\lfloor 255/5 \rfloor = 51$ in regular Stata and $\lfloor 2296/5 \rfloor = 459$ in Intercooled.

Now consider what happens when you type '`codebook`' with no arguments, which is the same, as far as Stata is concerned, as typing '`codebook`' followed by every variable in the data. If you are using the regular version of Stata, all those variables names, with the intervening blank, must fit into a single macro—which is to say, 255 characters. If they do not, you will get a too-many-variables error. For Intercooled Stata users, the rule is less restrictive but still present—the variable names must fit into 2,296 characters.

In either case, if you get the too-many-variables error, you can avoid the problem by explicitly specifying a subset of the variables and issuing the command multiple times. In the case of `codebook`, however, if you do run into this difficulty, you must also not specify the `mv` option. The `mv` option searches the data for patterns of missing values and, in the process, must make a list of all the variable names it is to search over. Thus, typing '`codebook myvar, mv`' can also result in the too-many-variables error.

Regular's Stata 255-character maximum is arguably too small, but there is nothing we can do about it; regular Stata is designed to run on small computers with little memory. For Intercooled Stata users, we admit that even the 2,296-character maximum is unfortunately small; it will be increased in the next release.

crc19	Nonlinear regression command

Royston's nonlinear regression command `nl` (Royston 1992a, b) is now an official part of the CRC updates. This means (1) the command and help files will automatically be installed when you install these updates or any future updates; (2) `nl` will be a part of the next release of Stata with documentation printed in the *Reference Manual*; and (3) `nl` is now supported by us. In the meantime, for printed documentation on this command, see Royston (1992a).

References

Royston, P. 1992a. sg1.2: Nonlinear regression command. *Stata Technical Bulletin* 7: 11–18.

——. 1992b. sg1.3: Nonlinear regression command: bug fix. *Stata Technical Bulletin* 8: 12.

dm10	Infiling data: Automatic dictionary creation

William Gould, CRC, FAX 310-393-7551

Reading raw data into Stata (or any statistical package) is probably one of the most difficult tasks facing a researcher. On the STB-9 media, I provide a utility to examine formatted, raw data and automatically create a Stata `.dct` dictionary for reading it and, moreover, for reading it efficiently in that variables are given appropriate storage types.

This utility is in the form of an `.exe` file for DOS users but, for all users, source code—including a `makefile`—is provided. Unix users can easily create their own executable by typing `make`. Details of installation are explained at the end of this insert.

creatdct

`creatdct` is a command you issue from your operating system, not Stata. Typing `creatdct` without arguments presents a syntax diagram:

```
C:\XMPL> creatdct
creatdct: usage: creatdct [/b /d /t] infile > outfile
          Reads data with variable names on top and writes to standard out
          a Stata .dct dictionary that can read the data.
          Options:
            /b  treat blank lines as significant;
            /d  write only the dictionary, referring to rather than including
                the raw data in the output;
            /t  infile contains data only; no titles.
          Limits:
            Maximum width of input file:    1000
            Maximum number of variables:     254
          References:
            STB-9: dm10 (which includes source code)
```

(Under Unix, options are preceded by a '`-`' rather than '`/`' and, if run from Unix, the syntax diagram would reflect that fact.)

To understand what `creatdct` does, consider the following (small) data set:

```
---------------------------------- mydta.raw ----------------------------------
Make and Model          Price   MPG     Weight  Repair
AMC Concord              4099   2.2e+1  2930    average
"BMW 320"                9735   25      2650    good
Honda Civic              4499   28      1760    exc
Volvo 260               11995   17      3170    exc
--------------------------------- end of file ---------------------------------
```

In this data, each line represents an observation and a title line appears above the data. The file could also contain blank lines (although this one does not). To create a dictionary for this data, type

```
C:\XMPL> creatdct mydta.raw > mydta.dct
```

The new file `mydta.dct` contains

```
---------------------------------- mydta.dct ----------------------------------
dictionary {
*
* This file was created by creatdct from mydta.raw.
* Type "infile using <this file>" to read the data.
*
  _column(1)      str11  make_an  %11s  "Make and Model"
  _column(25)     int    price    %5f   "Price"
  _column(33)     byte   mpg      %6f   "MPG"
  _column(41)     int    weight   %4f   "Weight"
  _column(49)     str7   repair   %7s   "Repair"
}
AMC Concord              4099   2.2e+1  2930    average
"BMW 320"                9735   25      2650    good
Honda Civic              4499   28      1760    exc
Volvo 260               11995   17      3170    exc
--------------------------------- end of file ---------------------------------
```

Thus, the entire Stata session to read the data might be:

```
. !creatdct mydta.raw > mydta.dct
. infile using mydta.dct
```

And, to prove that the data read correctly:

```
. describe
Contains data
  Obs:     4 (max=  5119)
 Vars:     5 (max=    99)
Width:    23 (max=   200)
  1. make_an      str11  %11s             Make and Model
  2. price        int    %8.0g            Price
  3. mpg          byte   %8.0g            MPG
  4. weight       int    %8.0g            Weight
  5. repair       str7   %9s              Repair
Sorted by:
. list
         make_an     price       mpg    weight     repair
  1. AMC Concord      4099        22      2930    average
  2.     BMW 320      9735        25      2650       good
  3. Honda Civic      4499        28      1760        exc
  4.   Volvo 260     11995        17      3170        exc
```

Variations

By default, `creatdct` creates a new file containing the dictionary and the data, but you can prevent `creatdct` from making a second copy of the data by specifying the `/d` (`-d`, Unix) option:

```
C:\XMPL> creatdct /d mydta.raw > dctonly.dct
```

The new file `dctonly.dct` contains

```
--------------------------------- dctonly.dct ---------------------------------
dictionary using mydta.raw {
*
* This file was created by creatdct from mydta.raw.
* Type "infile using <this file> in 2/l" to read the data.
*
  _column(1)      str11  make_an  %11s  "Make and Model"
  _column(25)     int    price    %5f  "Price"
  _column(33)     byte   mpg      %6f  "MPG"
  _column(41)     int    weight   %4f  "Weight"
  _column(49)     str7   repair   %7s  "Repair"
}
--------------------------------- end of file ---------------------------------
```

As the comment at the top of the file instructs, typing '`infile dctonly in 2/l`' will read the data set. The `/d` option is most useful when dealing with large amounts of data. (Do not assume that you always type the modifier "`in 2/l`"; `creatdct` modifies the recommended command according to the actual line containing the first observation of the data. From inside Stata, you can use the `type` command to display the dictionary on your screen and so read the recommendation.)

Basic logic

The above two examples illustrate how `creatdct` is typically used. In particular, `creatdct` assumes:

1. Blank lines can occur any place in the file; their presence is irrelevant.
2. The first non-blank line is a title line; it is from that line that variable names and labels are to be derived.
3. After the title lines, remaining nonblank lines contain the data. Each line contains all the data for a single observation.

Assumptions 1 and 2 can be relaxed by specifying options, but assumption 3 must be true. The `/b` (`-b`, Unix) option relaxes the first assumption, but there is never any reason to specify it. The `/t` (`-t`, Unix) option relaxes the second assumption.

Data without titles

Small ASCII data sets often include a title line, but large data sets generally do not. Large data sets, however, at least tend to be formatted. `creatdct` can be used to automatically construct a dictionary for such data sets assuming (1) each observation occupies one line and (2) the values recorded in the data do not "run together"—there is white space separating the columns of the data. Consider `mydta.raw` without the title line:

```
---------------------------------- mydta2.raw ---------------------------------
AMC Concord             4099   2.2e+1  2930    average
"BMW 320"               9735   25      2650    good
Honda Civic             4499   28      1760    exc
Volvo 260              11995   17      3170    exc
--------------------------------- end of file ---------------------------------
```

Typing '`creatdct /t mydta2.raw > mydta2.dct`'—the `/t` indicates that the data does not include a title line—results in the file:

```
---------------------------------- mydta2.dct ---------------------------------
dictionary {
*
* This file was created by creatdct from mydta2.raw.
* Type "infile using <this file>" to read the data.
*
  _column(1)      str11  v1       %11s
  _column(25)     int    v2       %5f
  _column(33)     byte   v3       %6f
  _column(41)     int    v4       %4f
  _column(49)     str7   v5       %7s
}
AMC Concord             4099   2.2e+1  2930    average
"BMW 320"               9735   25      2650    good
Honda Civic             4499   28      1760    exc
Volvo 260              11995   17      3170    exc
--------------------------------- end of file ---------------------------------
```

When there is no title line, `creatdct` uses names like `v1`, `v2`, etc. The `/d` option can be combined with the `/t` option; typing '`creatdct /t /d mydta2.raw > mydta2.dct`' results in a file that references, rather than contains, the underlying data. This can be especially useful when the data is large because then you can edit the dictionary and change the names to more reasonable ones.

Data with "errors"

`creatdct` tries to deal with problems that may crop up in real applications. Consider:

```
---------------------------------- mydta3.raw ---------------------------------
Make and Model          Price   MPG     Weight  Repair     Repair
AMC Concord              4099   2.2e+1  2930    average    fair
BMW 320                  9735   25      2650    good       good
Honda Civic              4499   28      1760    exc        good
Volvo 260               11995   17      3170    exc        average
--------------------------------- end of file ---------------------------------
```

In this data, the title `Repair` occurs twice (the first time for repair record in 1978 and the second for 1977, although there is no way you could know that from the data). Typing '`creatdct /d mydta3.raw > mydta3.dct`' results in

```
---------------------------------- mydta3.dct ---------------------------------
dictionary using mydta3.raw {
*
* This file was created by creatdct from mydta3.raw.
* Type "infile using <this file> in 2/l" to read the data.
*
  _column(1)      str11  make_an  %11s  "Make and Model"
  _column(25)     int    price    %5f  "Price"
  _column(33)     byte   mpg      %6f  "MPG"
  _column(41)     int    weight   %4f  "Weight"
  _column(49)     str7   v5       %7s  "Repair"
  _column(60)     str7   repair   %7s  "Repair"
}
--------------------------------- end of file ---------------------------------
```

`creatdct` changed one of the duplicate names to `v5`. `creatdct` would also change variable names to names like `v5` if the column header was not a legal Stata variable name—for instance, if the column header were "`78 repair`".

One of the hardest problems for `creatdct` is to match the column header to the data—much of its code is dedicated to that problem and still, it is not always successful. `creatdct`, however, knows when it has problems and tries to behave reasonably if not as elegantly. Consider the raw data:

```
---------------------------------- mydta4.raw ---------------------------------
Make and Model   X      Price   MPG     Weight  Repair     Repair
AMC Concord              4099   2.2e+1  2930    average    fair
BMW 320                  9735   25      2650    good       good
Honda Civic              4499   28      1760    exc        good
Volvo 260               11995   17      3170    exc        average
--------------------------------- end of file ---------------------------------
```

Notice the extra "X" in the title line. Is it part of "Make and Model"; is it part of "Price"; or is it all by itself, representing a variable for which there is no data? Typing '`creatdct /d mydta4.raw > mydta4.dct`' creates

```
---------------------------------- mydta4.raw ---------------------------------
dictionary using mydta4.raw {
*
* This file was created by creatdct from mydta4.raw.
* Type "infile using <this file> in 2/l" to read the data.
*
* Note from creatdct:  I had trouble lining up the titles against the data.
* Due to blanks in the titles, it appeared that there were more columns than
* columns in the data.  I finally gave up and just used the parts of the
* headers that were directly above data.  Sorry.
*
  _column(1)      str11  make_an  %11s  "Make and Mo"
  _column(25)     int    price    %5f  "Price"
  _column(33)     byte   mpg      %6f  "MPG"
  _column(41)     int    weig     %4f  "Weig"
  _column(49)     str7   v5       %7s  "Repair"
  _column(60)     str7   repair   %7s  "Repair"
´}
--------------------------------- end of file ---------------------------------
```

All of `creatdct`'s complicated logic for extending the titles around the data columns failed it, so it resorted to a dumber rule—taking just the part of the titles that lie directly above the data. Notice that the weight variable is now simply called `weig`.

Installing creatdct

For DOS users, simply copy `creatdct.exe` to a directory that is in your DOS path such as `c:\dos` or `c:\stata`. Although `c:\stata` seems more appealing, `c:\dos` is probably a better idea. If you put it in `c:\stata`, the next time we update Stata, you will have to reinstall `creatdct.exe`.

For Unix users, note that the C source code is provided. Copy the source to any temporary directory and, with the temporary directory as your current directory, type '`make creatdct`'. This will produce the executable `creatdct` which you can copy to some place along your path.

Actually, the C source code is provided for all users, although it is setup to compile under Unix, which means only that the option switch character has been set to '`-`' rather than '`/`'. DOS users can edit the file `machdep.h` and reset the manifest `ALTSWITCHAR`—the change is obvious once you are looking at the file. The limits of a maximum file width of 1,000 characters and a maximum number of variables of 254 are also set in `machdep.h`. You can change these limits—much larger limits are no problem under Unix. DOS users are warned, however, that the size of buffers is determined by these limits and that buffers are allocated on the stack.

Request for comments

If you have a data set on which `creatdct` fails or, while in the spirit of the files processed by `creatdct`, does not exactly match the rules, please contact me. Our goal is to further refine `creatdct` through the STB and then, once it is well debugged, to include its logic in the Stata `infile` command, offering users a way of reading formatted data without the bothersome step of creating a dictionary.

Users interested in further automating the infile process might consider typing in the following ado-file and saving it in their ado directory (`c:\ado` under DOS or `~/ado` under Unix):

```
--------------------------------- autoin.ado --------------------------------
*! version 1.0.0
program define autoin /* input filename */
        if _N~=0 { error 18 }
        drop _all
        confirm file `1´
        if "`2´"~="" { error 198 }
        !creatdct `1´ > autoin.dct
        infile using autoin.dct
        erase autoin.dct
end
-------------------------------- end of file --------------------------------
```

Typing '`autoin mydta.raw`', for instance, would then read the data in the first example.

I do *not* include this ado-file among the `creatdct` materials because I cannot believe that `creatdct` is sufficiently robust that you should use it without at least looking at the dictionary it produces first, but you are welcome to use the `autoin` command if you wish. At some future date, `creatdct` should be sufficiently robust that a more elegant version of `autoin` will be justified.

gr11	Using CorelDraw as a Stata graphics editor

Marc Jacobs, Dept. of Sociology, Univ. of Utrecht, The Netherlands, FAX (011)-31-30 534405

After some work I have discovered a way to import Stata graphics into CorelDraw. It is still necessary to put some work into it.

1. A Stata graphic (boosted up by Stage or not) is translated into a Lotus `pic` file. For this, I use a DOS `BAT` file containing the single line:

   ```
   c:\stata\gphpen %1 /DC:\stata\pic.pen /oc:\data\%1.pic /n %2 %3 %4 %5
   ```

   Calling this file `pic.bat`, I can type '`pic` *filename*' to create the file *filename*`.pic` in my `\data` directory.

2. CorelDraw is started and the `pic` file is imported. `ALT-F(ile), I(mport)`, choose `LOTUS PIC` file format and pick the proper file. The `pic` file is imported, but anything, including text, is seen as several objects in one group. There can easily be 512 objects in one group.

3. Ungroup the objects first: `ALT-A(rrange), U(ngroup)`. In most cases the graphic is too small for comfort, or the conversion of the lines is not satisfying. Before editing, change the line size to 1 mm and black: Choose the `PENCIL` icon, then the `1 MM LINE` icon, and finally the `BLACK` icon, while the group object still is chosen.

Stata text is made of plotter symbols. It is better to remove the text and replace them by the fancier fonts of CorelDraw. The only way to do this is to delete the group of objects forming one word of sentence and to replace them by the CorelDraw text possibilities.

As general advise: Put guide lines (blue lines pulled out of the rulers) first, where the old Stata text is, before deleting it. Put `snap to guide lines` on by choosing `ALT-D(isplay)`, `G(uide lines)`.

Suggestions for improvement are welcome.

ip2 A keyboard shortcut

Peter A. Lachenbruch, Dept. of Biostatistics, UCLA

I am providing a neat trick that other Stata users may already know, but if not it's a stroke saver. Since the missing value code '`.`' is greater than any nonmissing value, we can use that to exclude cases in a simple manner. Suppose I want to compute the means for `X` for those cases in which `Y` is not missing. The usual way is to write

```
summ X if Y~=.
```

I find it easier to write

```
summ X if Y<.
```

The '`<`' key is easier for me to reach, and only takes one stroke.

sbe7 Hyperbolic regression analysis in biomedical applications

Paul Geiger, USC School of Medicine, pgeiger@vm.usc.edu

Hyperbolic regression fits a curve to experimentally obtained data points for many analyses used in biochemistry and molecular biology (Studnicka 1987, 1991). These areas include enzyme kinetics, agarose gel electrophoresis of DNA fragments, SDS-polyacrylamide gel electrophoresis of proteins, enzyme-linked immunosorbent assays (ELISA), radioimmunoassays (RIA), Bradford assays (protein), and the much used and cited Lowry protein assay.

Hyperbolic regression fits curves without the biases linear regression introduces to an equation transformed to double reciprocal coordinates (Lineweaver and Burk 1934). More complicated transforms like the logit-log or four-parameter methods (Rodbard et al. 1987; Geiger 1991, 1992) may also be unnecessary when a hyperbolic relationship between variables exists. Hyperbolic regression, like linear regression, has the additional advantage of yielding one simple equation with its estimated constants. This equation can model actual chemical and biological processes.

Fitting curves with the cubic spline or the model-free approach of Guardabasso et al. (1987) produces no single equation to describe the scientific phenomena meaningfully. Results can also differ from analysis to analysis.

Studnicka (1987) implemented an algorithm for hyperbolic regression on a Digital Equipment Corporation PDP-11 series computer and later (1991) designed a spreadsheet to carry out the process. An outline of the mathematics follows.

A general equation for a hyperbola can be written as

$$(Y - Y_o)(X - X_o) = C_o$$

where (X_o, Y_o) is the mathematical origin displaced from $(0, 0)$ and C_o is a constant. Rearranging the equation by multiplying and collecting the constants and defining a new constant, $W_o = C_o - X_oY_o$, gives:

$$XY = X_oY + XY_o + W_o$$

Following Bevington (1969) least squares was satisfied by minimizing:

$$\sum_{i=1}^{N}(X_iY_i - X_oY_i - X_iY_o - W_o)^2$$

This can be solved using Stata's linear regression where X_o, Y_o, and W_o are the coefficients.

The above method of fitting works only if curvature really exists. A perfectly linear data set is unacceptable as the determinants of the matrices all go to zero. Also Studnicka (1987) remarks that if each data point consists of a number of experimental estimates, the mean must be calculated and used in the equations to get the best fit. This characteristic is a result of the matrix algebra. I have not observed this behavior.

The above equations have been incorporated into a Stata program, `hbolic.ado`, that calculates the necessary X_o, Y_o and C_o and the fitted curve, `hat`. This program, together with a help file and an example data set, `dnafrag2.dta`, are provided on the STB diskette.

The syntax of `hbolic` is

`hbolic` *depvar indepvar*

It should be noted that `Xo`, `Yo`, and `Co` are calculated constants that can be renamed according to the application. For instance, name `Km` for `Xo`, `Vmax` for `Yo`, and `KmVmax` for `Co` if you are concerned with Michaelis-Menten kinetics. The variable `Yhat` now denotes the fitted regression line and `resids` the residuals. The program displays the regression curve fitted to the original data points. It also displays the residuals plotted against `Yhat`. These graphs are saved for printing as `gph1` and `gph2`. Finally, the contents of the completed file are shown on the screen with `describe` followed by `list Yhat depvar indepvar Xo Yo Co` and a message inviting the user to view the saved graphs.

Unlike Studnicka's program devised for the DEC PDP-11 (1987), I found `hbolic.ado` works with several Y values (replicates) for each X value (standard). A mean for the Ys need not be computed before running `hbolic`.

Three examples of the use of hyperbolic regression with `hbolic` follow.

Example 1. Michaelis-Menten kinetics model many enzymic and other biological processes. In studies of an enzyme, the two desired characteristic parameters are the maximum velocity, V_m, and the Michaelis constant, K_m. V_m reflects the maximum velocity of the reaction, usually in micromoles per minute, at substrate saturation. K_m is the binding constant for substrate with enzyme in units of micromoles. The Michaelis-Menten equation is usually written:

$$V_o = \frac{V_m[S]}{K_m + [S]}$$

Initial velocity is half-maximal when $[S] = K_m$. The Lineweaver-Burk (1934) rearrangement is the so-called double reciprocal form, and $1/V_o$ was plotted against $1/[S]$ in a graphical analysis approach necessary before the advent of computers. But the original equation is easily rearranged to make its hyperbolic form evident: $(V_o - V_m)([S] + K_m) = -K_m V_m$.

Studnicka (1987) used the values in the file `michment.dta` (taken from his paper and supplied on the disk) to show strengths and weaknesses in the hyperbolic solution versus the double reciprocal plot to which he applied linear regression. The file contains the model values from his demonstration.

Estimation errors are less likely to occur using the hyperbolic form. This form is most sensitive to changes at high $[S]$ values where measurement errors of initial velocity and $[S]$ occur less often. You can see how small changes to V_o or $[S]$ affect estimated V_m and K_m values (Y_o and X_o produced by `hbolic`). Simply run `hbolic` after changing one value of X or Y. If the highest value of V_o changes by $\pm 10\%$, K_m changes a whopping $\pm 50\%$ and the V_m by about $\pm 15\%$.

Example 2. Studnicka (1987) also applied hyperbolic regression to DNA fragment analysis. Data taken from Figure 2 in the 1987 paper referring to an agarose gel electrophoresis experiment are in the file `dnafrag.dta` supplied on the disk. `hbolic` generates the constants, X_o, Y_o, C_o, after entering the values for X, the known DNA fragment sizes in kilobasepairs (kB), and Y, the measured migration distances in mm.

Measuring the migration distances accurately in this kind of experiment is the critical aspect and most subject to error. Compute the sizes of your own unknowns in kB by giving the command '`generate sizes=(Co/(unk-Yo))+Xo`', solving the hyperbolic equation for X ("`unk`" not provided here). In this equation "`unk`" stands for the migration distances of unknowns, measured in mm. Alternatively, use `infile X Y unk` to import a complete ASCII data file containing X, Y and "`unk`". If you use `hbolic.ado` frequently for many such analyses, append the necessary command to solve for sizes of fragments.

Example 3. This sample radioimmunoassay (RIA) data comes from Sundqvist et al. (1989) and appears in Geiger (1991, 1992). Geiger (1991) demonstrates the overworked logit-log analysis and Geiger (1992) the four-parameter model solved with Danuso's nonlinear regression program (Danuso 1991), an iterative procedure.

Application of hyperbolic regression to radioimmunoassay data seems every bit as effective as these more complicated methods. Furthermore, the blank or nonspecific binding cpms need not be subtracted as when performing logit-log analysis. The included file, `hypria.dta`, reproduces the RIA data from the previous publications.

The original triplicate cpm data were averaged with the command '`egen Y=rmean(c1 c2 c3)`'. `hbolic` then generated the necessary constants. Applying the equation in Example 2 above provided the results (pg/ml) from the cpms of the unknowns after correcting for volume as in the previous reports.

With the file `hypria.dta` in Stata's memory, type '`describe`' to see the labeled variables. Enter '`list smpl ans _2ans`' to see Sundqvist's original sample identification numbers and the logit-log result compared to the hyperbolic regression result, `_2answer`. The comparison is

Sample ID	Result (pg/ml) logit-log calc.	Result (pg/ml) Hyperbolic regression
11772	1567.16	1557.26
11772	1544.25	1534.88
11772	1781.31	1788.14
11772	1767.13	1774.17
11773	1332.42	1344.80
11773	1300.03	1295.99
11774	1074.86	1075.22
11774	1167.37	1181.08
11774	1193.03	1206.56
11774	1032.11	1033.22
11775	1517.34	1508.58
11775	1557.94	1548.26
11775	1748.40	1755.71
11775	1805.22	1811.69
11776	1292.76	1288.87
11776	1343.17	1355.45
11777	1029.33	1030.49
11777	1209.31	1222.73
11778	1486.53	1497.30
11778	1512.91	1504.25
11779	1415.52	1409.01
11779	1360.10	1354.80
11779	1506.45	1516.98
11779	1526.62	1536.91
11780	927.03	929.89
11780	953.58	968.24
11780	986.39	1000.97

Of course, the hyperbolic regression technique still requires investigators to obtain good duplicate and preferably triplicate observations of their experimental data.

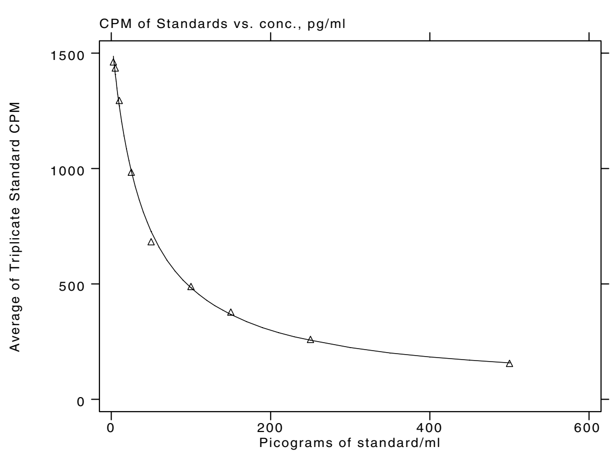


Figure 1

References

Bevington, P. R. 1969. *Data Reduction and Error Analysis for the Physical Sciences.* New York: McGraw–Hill, 92–163.

Danuso, F. 1991. sg1: Nonlinear regression command. *Stata Technical Bulletin* 1: 17–19.

Geiger, P. J. 1991. sbe3: Biomedical analysis with Stata: radioimmunoassay calculations. *Stata Technical Bulletin* 3: 12–15.

——. 1992. sbe4: Further aspects of RIA analysis. *Stata Technical Bulletin* 5: 7–10.

Guardabasso, V., D. Rodbard, and P. J. Munson. 1987. A model-free approach to estimation of relative potency in dose–response curve analysis. *Am. J. Physiol* 252(15): E357–E364.

Lineweaver, H. and D. Burk. 1934. The determination of enzyme dissociation constants. *J. Am. Chem. Soc.* 56: 658–666.

Rodbard, D., et al. 1987. Statistical Aspects of Radioimmunoassay. In *Radioimmunoassay in Basic and Clinical Pharmacology*, vol. 82 of *Handbook of Experimental Pharmacology*, ed. C. Patrono and B. A. Peskar, ch. 8. New York: Springer-Verlag.

Studnicka, G. M. 1987. Hyperbolic regression analysis for kinetics, electrophoresis, ELISA, RIA, Bradford, Lowry, and other applications. CABIOS 3: 9–16.

——. 1991. ELISA assay optimization using hyperbolic regression. CABIOS 7: 217–224.

Sundqvist, C., et al. 1989. A radioimmunoassay program for Lotus 1-2-3. *Comput. Biol. Med.* 19: 145–150.

sbe8 | Left-censored survival data

William Rogers, CRC, FAX 310-393-7551

Left-censored survival data arise when a subject comes under observation some time after the event that starts the "clock." For example, suppose you are studying factors that predict whether a woman will give birth to a child. You sample women, give them a baseline survey, and then follow them annually for 6 years thereafter. At the start of the study, each woman has been at risk since her last birth, or since menarche.

Suppose that woman A in this sample had her last child 3.2 years ago, and she has a child 2.4 years after your baseline survey. You would probably want to analyze the birth interval of 5.6 years. The hitch is that you could not have observed an interval of 5.6 years for this woman unless she had gone at least 3.2 years from her last birth without having another child. In other words, you want to analyze the probability of surviving from 3.2 years to 5.6 years and then giving birth.

Actually, most analysts are not interested in the probability per se, but rather the relative risk of giving birth for a woman with one set of characteristics as opposed to another. For example, what is the relative risk of giving birth for an unmarried woman as opposed to a married one?

There are several ways to handle this kind of problem in Stata, of which I will discuss two here.

Method 1: Use exponential instead of Cox regression. In exponential regression, the likelihood is completely specified and the hazard does not depend on previous time at risk. There is, however, a price. It now becomes your responsibility to verify that the likelihood does not depend on time since last birth or to specify the dependence if there is one.

In this example, there is a dependence. It is almost impossible to have another child within 11 months, and the likelihood is low immediately after that. So you might break the time into intervals and introduce variables for 11–16 months, 17–24 months 25–36 months, etc., and replicate the observation on each woman up to the point she has another child or no longer appears in the data. This replication, resulting in a dataset where the same woman appears more than once, could potentially result in a dataset where the outcomes are dependent on each other. A solution to this problem is "Huber" exponential regression.

Method 2: Use time-varying Cox regression and introduce a variable that describes the left-censoring period.

The following example shows how to do the data preparation for each method. They are both tricky. Let me begin by concocting a data set:

```
. input id  age previous tbirth isbirth
       1   24    10      5        1
       2   19     2      2        1
       3   42     2      6        0
       4   36     6      6        0
       5   30     2      4        1
       6   32    18      6        1
       7   38    14      6        0
       8   28     6      3        1
       9   22     3      1        1
end
. save birth, replace
```

`previous` records the number of years at the time of the baseline survey since the previous birth (or menarche). `tbirth` is the time from the baseline survey to the next birth or end of the survey period. `isbirth` records 1 if the woman had another (or first) birth during the survey period and 0 otherwise.

The exponential-regression solution relies on the fact that the probability distribution of `tbirth` is the same as the distribution of `tbirth` + `previous` given `previous`:

```
. use birth, clear
. expand 2
. sort id
. qui by id: gen record = _n
. gen over5 = record==2
. drop if record==1 & previous>=5
. gen atrisk = min(tbirth,5-previous) if record==1
. replace isbirth=0 if record==1 & atrisk<tbirth
. drop if record==2 & isbirth[_n-1]==1 & id==id[_n-1]
. replace atrisk = tbirth-atrisk[_n-1] if record==2 & id==id[_n-1]
. replace atrisk = tbirth if record==2 & id~=id[_n-1]
. hereg atrisk age over5, dead(isbirth) group(id) hr
```

Each woman in the original data potentially becomes two observations, the first reflecting the first five years (from the last child or menarche) and the second the remaining period. Since the same woman may appear in the data more than once, we use `hereg` (see *sg8.1* in this issue) to correct for the violation of independent-observations assumption.

The second approach, using Cox regression, is easier. `tbirth` + `previous` is used as the time-to-birth variable, but we must create a censored segment for the times between 0 and `previous`:

```
. use birth, clear
. expand 2
. sort id
. qui by id: gen record = _n
. gen atrisk = tbirth+previous if record==2
. replace atrisk = previous if record==1
. gen left = (record==1)
. replace isbirth = 0 if record==1
. cox atrisk age left, dead(isbirth) tvid(id) hr
```

The output produced by the two estimation steps is

```
. hereg atrisk age over5, dead(isbirth) group(id) hr

Iteration 0: Log Likelihood = -12.037857
Iteration 1: Log Likelihood = -10.206155
Iteration 2: Log Likelihood = -8.9897404
Iteration 3: Log Likelihood = -8.8922672
Iteration 4: Log Likelihood = -8.8907967

Exponential regression (log relative hazard form)   Number of obs    =      11
Log Likelihood             =    -8.891              Pseudo R2        =  0.2614

Grouping variable: id
------------------------------------------------------------------------------
  atrisk |  Hz. Ratio   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     age |   .8403259   .0352805     -4.144   0.003       .7627832    .9257515
   over5 |   1.514651   1.063751      0.591   0.571       .2998902    7.650031
------------------------------------------------------------------------------

. cox atrisk age left, dead(isbirth) tvid(id) hr

Iteration 0:  Log Likelihood =-9.0484095
Iteration 1:  Log Likelihood =-2.8854591
Iteration 2:  Log Likelihood =-2.1052036
 (output omitted)
Iteration 44:  Log Likelihood =-1.7583729
Iteration 45:  Log Likelihood =-1.7583729
```

```
Cox regression                                        Number of obs =      18
                                                      chi2(2)       =   14.58
                                                      Prob > chi2   =  0.0007
Log Likelihood = -1.7583729                           Pseudo R2     =  0.8057
------------------------------------------------------------------------------
  atrisk |
 isbirth | Haz. Ratio   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
     age |   .7223358   .1759505     -1.335   0.199       .4320623    1.207624
    left |   3.37e-21          .          .       .              .           .
------------------------------------------------------------------------------
```

For the exponential regression, the nonsignificant coefficient for `over5` suggests that previous time exposed is not an important factor in the hazard rate.

For the Cox regression, the impact of the variable `left` is huge. Since this regression is presented in hazard form, the hazard ratio of 3.37e–21 reflects the fact that we observed no failures in the left-censoring period. The long iteration log is characteristic of a parameter that is being driven to infinity. Stata recognizes that something is wrong with this variable and prints a '`.`' for the standard error. In some cases you will get a zero t-statistic, which is also a sign of trouble.

Although both hazard ratios for age are well below 1, the two methods give slightly different hazard ratios. More importantly, the t-statistics are quite different. There are two reasons. First, the parametric information typically improves precision. Second, the `hereg` estimates are based on asymptotic results and overstate accuracy in very small samples.

sg8.1 Huber exponential regression

William Rogers, CRC, FAX 310-393-7551

After my insert on probability weighting (Rogers 1992), I have received a number of questions asking what to do about design problems (probability weighting and clustering) with survival analysis, most often in reference to Cox regression.

Unfortunately, the Huber method that worked for `regress` and `logit` (yielding the Stata `hreg` and `hlogit` commands) does not work well in the case of Cox regression because the conditional likelihood derivatives for each observation are a complicated function of the entire dataset. It is, however, easy to apply the Huber method to exponential regression, which is a close cousin of Cox proportional hazards model.

`hereg` has the syntax:

hereg [*depvar* [*varlist*] [*weight*] [**if** *exp*] [**in** *range*]] [**,** **hazard hr tr dead(***varname***)**
level(*#***) group(***varname***)** *maximize_options*]

See [5s] huber for a description of the treatment of the optional weight and `group()` option; see [5s] ereg for a description of the remaining options.

In survival analysis, violations of assumptions can arise both at the sampling stage and be purposefully induced by the researcher at the analysis stage. In *sbe8* of this issue, I demonstrate a purposeful induction to estimate a model with both left and right censoring. `hereg` handles the problems associated with the violations of assumptions.

References

Rogers, W. H. 1992. Probability weighting. *Stata Technical Bulletin* 8: 15–17.

sg9 Similarity coefficients for 2 x 2 binary data

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

Binary similarity measures estimate the proximity between two 1/0 binary variables. Three types of measures are provided: ones that can be thought of as similar to a correlation coefficient, others that can be interpreted as conditional probabilities, and lastly those that are predictability measures. I have provided a program called `similari` which displays twelve such similarity coefficients. The program is called `similari` and not `similar` because it is an immediate command (see [4] immediate); one need only input the tabulated summary data on the command line. The syntax of `similari` is

$$\texttt{similari}\ A_{0,0}\ B_{0,1}\ C_{1,0}\ D_{1,1}$$

Hence, you may directly type in the summary data from the Stata `tabulate` command.

The following statistics are provided:

1. Czekanowski (Dice): A matching coefficients measure in which double weights are given to matches (1,1).
2. Dispersion: A similarity measure that ranges from -1 to 1.
3. Jaccard: A similarity ratio in which 0,0 is excluded from the equation.
4. Match percent: The ratio of total matches to the total population.
5. Ochiai: A similarity measure in cosine form.
6. Phi 4-point: A binary form of the Pearson product correlation coefficient.
7. Russell & Rao: a binary dot product; 1,1 matches to total population.
8. Hamann: A conditional probability measure ranging in value from -1 to 1.
9. Anderberg's D: A predictability measure indicating the reduction in the probability of error when an item is used to predict another.
10. Goodman and Kruskal's Lambda: Indicates the proportional reduction in the probability of error when one item is used to predict another when the prediction directions are equal. The predictability of the value of one item given the value of another.
11. Yule's Q: A binary version of the Gamma test ranging from -1 to 1.
12. Yule's Y: A coefficient of colligation ranging from -1 to 1.

All coefficients range from 0 to 1 unless otherwise indicated. An example program run follows:

```
. use lbw
. tab smoke low
     smoked| birth weight<2500g
     during|
  pregnancy|         0          1 |     Total
-----------+----------------------+----------
         0 |        86         29 |       115
         1 |        44         30 |        74
-----------+----------------------+----------
      Total|       130         59 |       189

. similari 86 29 44 30
           Similarity coefficients for 2 X 2 binary data
                          Controls
        Cases  |        0           1           |     Total
        -------+--------------------------------+----------
          0    |        86          29          |      115
          1    |        44          30          |       74
        -------+--------------------------------+----------
        Total  |       130          59          |      189
    Proximity measures                 Conditional probability measure
Czekanowski   =    0.4511             Hamann               =    0.2275
Dispersion    =    0.0365
Jaccard       =    0.2913                 Predictability measures
Match %       =    0.6138             Anderberg´s D        =    0.0026
Ochiai        =    0.4540             G & K Lambda         =    0.0026
Phi 4-point   =    0.1614             Yules Q              =    0.3382
Russell & Rao =    0.1587             Yules Y (colligation)=    0.1742
```

Each listed statistic accords with that produced by the `proximity` command in SPSS for Windows.

References

Anderberg, M. R. 1973. *Cluster Analysis for Applications.* New York: Academic Press.

Romesburg, H. C. 1984. *Cluster Analysis for Researchers.* Belmont, CA: Lifetime Learning Publications.

SPSS. 1991. *SPSS Statistical Algorithms.* 2d ed. Chicago: SPSS.

sg10 Confidence limits in bivariate linear regression

Paul Geiger, USC School of Medicine, pgeiger@vm.usc.edu

[The method discussed in this insert is often described as the "calibration method."—Ed.]

Many analytical methods in chemistry and biology provide a linear relationship between the response variable, Y, and the amount of material determined, X. We construct a standard curve using known amounts of X and observe the response in Y. Samples containing unknown amounts of X are treated along with the known quantities. From the standard curve given by linear regression of Y on X, the unknown X's can be estimated. Examples that come readily to mind from my own work are the modified Fisk and Subbarow colorimetric assay for inorganic phosphate, spectrophotometric and fluorometric assays for various enzymes and their metabolic products (Lowry and Passonneau 1972; Bergmeyer 1983–1986) and scintillation counting of radioactive element-tagged compounds.

In planning further experiments or making decisions based on values of X computed from the standard curve, we may wish to know the confidence limits of X. Also, in developing a new method, we want to know if the analysis is trustworthy within reasonable limits.

I followed the equations given in Snedecor and Cochran (1989, 170–172) in writing the program `confx.ado`. The program is furnished along with a help file on the STB diskette. It gave correct answers for the problems listed in Snedecor and Cochran as well as for example 14.7 from Sokal and Rohlf (1981, 498). These exercises are also supplied on the disk.

References

Bergmeyer, H. U., editor in chief. 1983–1986. *Methods of Enzymatic Analysis.* 3d ed. Deerfield Beach, FL: Verlag Chemie.

Lowry, O. H. and J. V. Passonneau. 1972. *A Flexible System of Enzymatic Analysis.* New York: Academic Press.

Snedecor, G. W. and W. G. Cochran. 1989. *Statistical Methods.* 8th ed. Ames, IA: University of Iowa Press.

Sokal, R. R. and F. J. Rohlf. 1981. *Biometry: The Principles and Practice of Statistics in Biological Research.* 2d ed. San Francisco: Freeman.

sg11 Quantile regression standard errors

William Rogers, CRC, FAX 310-393-7551

Obtaining standard errors for the coefficients in quantile regression (`qreg`) is a difficult problem and one for which the literature provides only sketchy guidance. In the case of linear regression, one can prove that the standard errors of the coefficients are given by the diagonal of $s^2(\mathbf{X}'\mathbf{X})^{-1}$, at least if the residuals are i.i.d. (independently and identically distributed) $\mathrm{N}(0,\sigma^2)$ (s^2 is the estimate of σ^2 and formulas can similarly be derived). A considerable literature on "robustness" and accompanying algorithms is available when assumptions fail (e.g., `rreg`, `hreg`).

What is worth understanding is that there are no clear-cut answers for quantile regression *even when assumptions are met.* There are some existing proofs (Koenker and Bassett 1978, 1982) that suggest estimates for the asymptotic parameter variances and covariances. These estimates form the basis of the Stata calculations. Of course, all proofs depend on assumptions and asymptotic estimates are not guaranteed to apply in small samples. The question of how good these estimates really are in small samples is open.

A primary purpose of quantile regression is to escape assumptions. One particular noteworthy assumption in the Koenker and Bassett formulation is that the error distribution is *homoscedastic*, that is it does not depend on $\mathbf{x}$, the vector of independent variables. Indeed, a major use of quantile regression is to calculate quantiles in situations where the quantiles are not parallel.

Bootstrapping (Efron 1982) provides an alternative way of obtaining standard errors without assumptions, but at the cost of computer time. In bootstrapping, we replicate the sampling experiment that created the sample by drawing from the sample as if it were a population. The idea that the sample is a close approximation to the population is the fundamental idea behind sampling. (A companion insert (*sg11.1*) provides a `bsqreg` command for constructing such estimates.) However, the fact that bootstrapping involves resampling makes it nonverifiable because the answers depend on the vagaries of a random number generator. In order to minimize the randomness, many replications should be taken.

In this insert, I explore the quality of the standard errors produced by the `qreg` command and those produced by bootstrapping. I find that the reported standard errors are quite satisfactory except in the case of heteroscedastic errors, in which case they substantially understate the true standard error. Since quantile regression is often used to estimate with precisely this kind of data, in such cases, bootstrap standard errors are preferable.

Analytic standard errors for quantile regression

Before comparing the standard errors produced by `qreg` to bootstrap standard errors, let me quickly review how `qreg` calculates the standard errors it reports. The Stata manual is somewhat terse on the subject and, worse, there is an error. In [5s] qreg, the manual says the variance–covariance matrix is estimated by $\mathbf{R}_2^{-1}\mathbf{R}_1\mathbf{R}_2^{-1}$. What it does not say, but should, is that $\mathbf{R}_1$ is estimated as $\mathbf{X}'\mathbf{W}\mathbf{W}'\mathbf{X}$, where $\mathbf{W}$ is a $n \times n$ diagonal matrix with elements

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

and $\mathbf{R}_2$ is the design matrix $\mathbf{X}'\mathbf{X}$. This is derived from formula 3.11 in Koenker and Bassett (1982), although their notation is much different. $f_{\text{errors}}()$ refers to the density of the true residuals.

There are many things that Koenker and Bassett leave unspecified, including how one should obtain a density estimate for the errors in real data. For example, a side effect of quantile regression is that if there are k parameters, at least k residuals must be exactly zero. Also, while their formula recognizes that heteroscedasticity may alter the standard errors, the factor increase is not computable unless one goes back to the assumption that the errors are i.i.d.

Below, we will explore the quality of the standard errors produced by `qreg` and its boot-strapped variant `bsqreg` in:

1. A simple bivariate model $y = \alpha + \beta_1 x + u$.
2. A trivariate model $y = \alpha + \beta_1 x_1 + \beta_2 x_2 + u$.
3. Various models in which assumptions or tacitly assumed conditions fail to be true, including (a) heteroscedastic errors and (b) skewed covariates.

Monte Carlo Experiment 1: A simple bivariate model

We wish to consider estimation of various quantiles for models of the form $y = \beta_0 + \beta_1 x + u$. A typical experiment in this group worked like this:

```
. drop _all
. set obs 1000
. gen x = invnorm(uniform())
. gen y = invnorm(uniform())
. qreg y x, quant(0.4)
. display _b[x]/_se[x]
. display (_b[_cons]-invnorm(0.4))/_se[_cons]
```

If the standard errors produced by `qreg` are accurate, the results displayed by the last two `display` statements should each be normally distributed with mean 0 and variance 1. This example is actually more general than one might first think, since the quantile regression function is a linear function of the independent variables. The quantile-regression estimator is not a linear estimator, but it does have certain properties that are worth appreciating, and which we will call "transparency." Given the model $y = \mathbf{bx} + u$, where the vector $\mathbf{x}$ may include an element equal to 1 corresponding to the constant, the transparency properties are

1. $y \to cy \Rightarrow \mathbf{b} \to c\mathbf{b}, s_{\mathbf{b}} \to cs_{\mathbf{b}}$. Multiplying the dependent variable by any constant results in new coefficient estimates that are the constant multiplied by the original coefficients. Standard errors are similarly multiplied by the constant; t-statistics are unchanged. This statement holds for $c > 0$. If $c < 0$, it holds if the quantile fraction is reversed.
2. $\mathbf{x} \to \mathbf{Ax} \Rightarrow \mathbf{b} \to \mathbf{bA}^{-1}$. Transforming the independent variables in some way comes out entirely in the coefficient estimates and does not affect the errors.
3. $y \to y + \mathbf{gx} \Rightarrow \mathbf{b} \to \mathbf{b} + \mathbf{g}$. Adding a multiple of $\mathbf{x}$ to the dependent variable comes out entirely in the estimated coefficients and does not essentially change the solution.

You can add to the Stata experiment shown above to verify these properties:

```
. gen w = 2*y
. qreg w x, quant(.4)
. gen z = - x
. qreg y z, quant(.4)
. gen v = y + 10*x
. qreg v x, quant(.4)
```

As a result of these transparency properties, it is not necessary to consider a wide range of multivariate problems. Complex multivariate problems can be transformed to look like univariate ones. For example, I do not need to consider problems where the $\mathbf{x}$'s are highly correlated.

I do consider problems over a range of sample sizes. In order to gain a complete understanding of the most common situation (median regression), I ran this simulation 5,000 times with a variety of sample sizes ranging from 20 to 5,000 (this took 240 hours on an unloaded DECstation 3100).

Sample size does not appear to be a major consideration (there was no relationship of either the formula or the bootstrap performance based on sample size).

			var(t_x)		var(t_{cons})	
Quantile		Replications	Formula	Bootstrap	Formula	Bootstrap
Median		5000	1.02	1.13	0.98	1.11
.4		219	1.00	1.09	1.02	1.19
.3		1250	1.11	1.11	0.98	1.13
.2		1250	0.93	1.10	0.95	1.22
.1		200	0.97	1.29	1.09	1.33
.05	$n > 100$	236	1.40	0.91	1.10	1.40
.05	$n \leq 100$	1014	38.15	1.66	23.34	2.53

The desired answer for all of the entries on the right-hand side of the table is 1.00, meaning that both the formula answer and the bootstrap answer produce asymptotic normal t-statistics.

For the median, the variance of the bootstrap answer is slightly higher (e.g., 1.13 instead of 1.02) because the bootstrap standard error is computed from 20 replications. Since this estimated standard error becomes the denominator of a t-statistic, we must evaluate that t-statistic against a t-distribution with 20 degrees of freedom. The match is perfect; the probability of exceeding the 5% cutoff values for a t-distribution with 20 degrees of freedom is 5.1% for both the coefficient of `x` and `_cons`. The performance of the formula answers is consistent with the asymptotic theory for a z-statistic.

The performance of the formula holds up well down to about the 10th percentile and does not depend on sample size. Below that, small-sample phenomena begin to take over. At the 5th percentile, both the formula and the bootstrap break down, but for different reasons. The formula breaks down because the density estimate breaks down. In the group with sample size 100 and less, the 10th percentile was 3.8 observations from the bottom edge of the dataset on the average. The density estimate is based on the nearest $\sqrt{n}$ observations, which covers a thicker part of the distribution. The bootstrap is much better than the formula, but still underestimates the variance.

Monte Carlo Experiment 2: A trivariate model

The second experiment is a trivariate regression. For the reasons of the previous discussion (which I confirmed in simulations not reported), there is nothing to be learned from an experiment with two normally distributed x's. So I simulated one x (say `x2`) that distributed with one observation at $+1$, one at -1, and 198 at zero.

I was interested in the effect that `x2` might have on the other x, which was normally distributed. But this `x2` pointed out an interesting dilemma faced by quantile regression theory. It is easy to construct examples where there are two quite different answers with exactly the same minimum weighted sum of deviations, and this particular design happens to be one such instance. Regardless of the quantile, one point seems to be ignored completely, while the coefficient of `x2` fits the other point exactly.

The answer to our original question—which standard error is better—now becomes clear. Although the formula is stressed, it is in the ballpark. The bootstrap frequently misses both of the high-leverage points and comes up with a much different solution. For the normally distributed x, the answers are pretty good for both methods, but for `x2` and `_cons`, the bootstrap is not a competitor.

Monte Carlo Experiment 3: Other not-so-nice problems

The following kinds of problems might be suspected to cause some problems for the theory:

a. Problems where there are heteroscedastic errors.

   I simulated a case similar to the one-variable simulation above, but made $\text{sd}(y) = \exp(x)$. This implies that the $\text{sd}(y)$ is about 50 times higher on one end of the x-distribution compared with the other one. This would be noticeable in a graph!

b. Problems with skewed covariates.

These simulations were all done with median regression with 1,250 replications of a sample size of 200:

	var(t_x)		var(t_{cons})	
Problem	Formula	Bootstrap	Formula	Bootstrap
Heteroscedastic Errors	2.89	0.88	3.93	1.13
Skewed Covariates	1.23	1.18	0.92	1.07

Heteroscedastic errors turn out to be a major problem for the standard error formula, consistent with the theory suggested by Koenker and Bassett (1982). They have roughly the same kind of impact as they would have in regular regression. The actual standard deviations of the t-statistics are well above 1. Standard errors of coefficients are underestimated, and so results are non-conservative.

The bootstrap is better, but slightly conservative for the coefficient of `x`. With the 20 degree-of-freedom approximation, we would like a variance of 1.11 in this column.

For a skewed covariate (implying a few points with very high leverage), the bootstrap performed well, but the formula was slightly high for the variance of `x` and low for `_cons`.

Conclusions

Quantile regression is becoming a favorite technique for identifying and exploiting naturally heteroscedastic phenomena, such as income inequality. These results suggest that standard errors produced in this kind of problem should be looked at suspiciously. We suggest that the bootstrap standard errors be used in cases where heteroscedasticity is suspected.

Quantile regression has also been suggested as a form of robust regression—a method that can be trusted without looking into the impact of outliers in `y` or `x`. These results suggest that such complacency is premature.

Finally, do not assume that either answers are good for extreme quantiles; for example, if $n < 5/q$ or $n < 5/(1-q)$.

References

Efron, B. 1982. *The Jackknife, the Bootstrap and Other Resampling Plans.* Philadelphia: Society for Industrial and Applied Mathematics.

Koenker, R., and G. Bassett. 1978. Asymptotic theory of least absolute error regression, *Journal of the American Statistical Association* 73: 618–622.

——. 1982. Robust tests for heteroscedasticity based on regression quantiles. *Econometrica* 50: 43–61.

Ruppert, D., and R. J. Carroll. 1980. Robust regression by trimmed least squares. *Journal of the American Statistical Association* 75: 828–838.

sg11.1 | Quantile regression with bootstrapped standard errors

William Gould, CRC, FAX 310-393-7551

Rogers in *sg11* argues that the formula-based Koenker and Bassett standard errors used by `qreg` are not satisfactory when heteroscedasticity of the residuals is suspected and suggests the substitution of bootstrap standard errors. Bootstrapping allows one to obtain standard errors for any statistic even when an analytical formula is not available and a large literature on this subject is available (see, for example, Efron 1982 and Wu 1986). The method is procedurally simple but computationally expensive. One has a data set containing N observations and an estimator which, when applied to the data, produces certain statistics (such as the coefficients produced by `qreg`). One draws, with replacement, N observations from the N observation data set. In this random drawing, some of the original observations will appear once, some more than once, and some not at all. Using that data set, one applies the estimator and estimates the statistics. One then does it again, drawing a new random sample and reestimating, and again, and keeps track of the estimated statistics at each step of the way (called a replication).

Thus, one builds a data set of the estimated statistics. From this data, one calculates the standard deviation of the statistic using the standard formula ($\sqrt{\sum[b_i - \bar{b}]^2/[K-1]}$, where K is the number of replications). That number is your estimate of the standard error of the statistic. Note that, while the average value of the observed statistic ($\bar{b}$) is used in the calculation of the standard deviation, it is not used as the estimated value of the statistic itself. The statistic is obtained in the normal way using the original N observations. (Many researchers new to bootstrapping think that $\bar{b}$ is somehow a better estimate of the statistic than the statistic obtained in normal ways. That is not quite true. What is true is, that if the statistic is biased in some way, $\bar{b}$ exaggerates the bias. Denoting b as the statistic calculated in the normal way, the amount of the bias can be estimated as $\bar{b} - b$ and, in fact, an unbiased statistic would be $b - (\bar{b} - b) = 2b - \bar{b}$. This adjustment, however, should only be applied if there are strong theoretical reasons to believe the statistic is biased.)

The syntax of `bsqreg` is

bsqreg [*depvar* [*indepvars*] [**if** *exp*] [**in** *range*]] [, **level(**#**)**
quantile(#**)** **wlsiter(**#**)** **reps(**#**)**]

That is, the syntax is identical to **qreg** (see [5s] qreg) except (1) weights are not allowed and (2) the option **reps()** is added. The **reps(***#***)** option controls the number of bootstrap replications used for calculating the standard errors and defaults to 20 if not specified.

In [5s] qreg, the following model was reported as an example:

```
. qreg price weight length foreign
Iteration  1:  WLS sum of weighted deviations =   114043.7

Iteration  1: sum of abs. weighted deviations =  114998.67
Iteration  2: sum of abs. weighted deviations =   111786.7
  (output omitted)
Iteration  9: sum of abs. weighted deviations =  108822.59

Median Regression                                    Number of obs =        74
  Raw sum of deviations   142205 (about 4934)
  Min sum of deviations 108822.6                     Pseudo R2     =    0.2347

------------------------------------------------------------------------------
   price |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  weight |   3.933588   .8602185      4.573   0.000       2.217936     5.64924
  length |  -41.25191   28.86931     -1.429   0.157      -98.82993     16.3261
 foreign |   3377.771   577.3392      5.850   0.000       2226.304    4529.238
   _cons |   344.6494   3260.245      0.106   0.916      -6157.704    6847.002
------------------------------------------------------------------------------
```

This model is an excellent candidate for bootstrapping as one might suspect that a model of price (as opposed to, say, log of price) would suffer from heteroscedasticity. Estimating this model with **bsqreg**:

```
. bsqreg price weight length foreign
(estimating base model)
(bootstrapping ....................)

Median Regression, bootstrap(20) SEs                 Number of obs =        74
  Raw sum of deviations   142205 (about 4934)
  Min sum of deviations 108822.6                     Pseudo R2     =    0.2347

------------------------------------------------------------------------------
   price |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  weight |   3.933588   2.837216      1.386   0.170      -1.725061    9.592236
  length |  -41.25191   73.45879     -0.562   0.576      -187.7608     105.257
 foreign |   3377.771   1164.462      2.901   0.005       1055.325    5700.217
   _cons |   344.6494   6587.404      0.052   0.958      -12793.51    13482.81
------------------------------------------------------------------------------
```

As we would expect based on Rogers' findings, the standard errors are larger (the ratios vary from 2.02 to 3.3).

The standard errors produced by the bootstrap technique are only approximations. Estimating the same model again produces different estimates:

```
. bsqreg price weight length foreign
(estimating base model)
(bootstrapping ....................)

Median Regression, bootstrap(20) SEs                 Number of obs =        74
  Raw sum of deviations   142205 (about 4934)
  Min sum of deviations 108822.6                     Pseudo R2     =    0.2347

------------------------------------------------------------------------------
   price |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  weight |   3.933588   3.042472      1.293   0.200      -2.134431    10.00161
  length |  -41.25191   84.80841     -0.486   0.628       -210.397    127.8931
 foreign |   3377.771   974.9878      3.464   0.001       1433.219    5322.323
   _cons |   344.6494   7482.968      0.046   0.963      -14579.66    15268.96
------------------------------------------------------------------------------
```

The accuracy of the approximation increases with the number of replications, but it is worth noting that, even at a moderate 20 replications, results are not substantively different.

Anytime one works with a computer, it is important to be able to reproduce results. Although the standard errors produced by **bsqreg** have a random component, they can be mechanically reproduced by resetting the random number seed (see [5d] generate) assuming one knows the seed prior to estimation. For instance, the following two commands will always produce the same results:

```
. set seed 573998311
. bsqreg price weight length foreign
  (output omitted)
```

One could produce more accurate estimates of the standard errors by including the `rep()` option: '`bsqreg price weight length foreign, rep(50)`'.

`bsqreg` is not as well integrated into Stata as the other estimation commands. While `_b[]` does contain the parameter estimates, and `predict` can be used to obtain predicted values and residuals, `_se[]`, `test`, and `correlate` cannot be used after estimation.

References

Efron, B. 1982. *The Jackknife, the Bootstrap and Other Resampling Plans.* Philadelphia: Society for Industrial and Applied Mathematics.

Wu, C. F. J. 1986. Jackknife, bootstrap and other resampling methods in regression analysis. *Annals of Statistics* 14: 1261–1350 (including comments and reply).

snp4 Non-parametric test for trend across ordered groups

K. A. Stepniewska and D. G. Altman, Imperial Cancer Research Fund, London, EMAIL k_stpniewska@icrf.ac.uk

The syntax of the `nptrend` command is

nptrend *varlist* [**if** *exp*] [**in** *range*], **by(***groupvar***)** [**trend(***scorevar***)**]

`nptrend` performs a non-parametric test for trend across ordered groups. This test, developed by Cuzick (1985), is an extension of the Wilcoxon rank-sum test and is a useful adjunct to the Kruskal–Wallis test (`kwallis`). Formula for the test statistic is given by Cuzick (1985) and Altman (1991). Correction for ties is incorporated into the formula.

groupvar is a grouping variable, and *scorevar* defines scores for groups. When `trend()` is not specified, the values of *groupvar* are used as the scores.

Example

Consider the following data (Altman 1991):

Group	Transmission of visible light	Ocular exposure to ultraviolet radiation
1	$< 25\%$	1.4 1.4 1.4 1.6 2.3 2.3
2	25 to 35%	0.9 1.0 1.1 1.1 1.2 1.2 1.5 1.9 2.2 2.6 2.6 2.6 2.8 2.8 3.2 3.5 4.3 5.1
3	$> 35\%$	0.8 1.7 1.7 1.7 3.4 7.1 8.9 13.5

We can use `nptrend` to test for a trend of increasing exposure across the three groups. When we do not specify scores for groups, they are defined by the grouping variable `groupvar`:

```
. nptrend exp, by(group)
Test: Trend across groups
       gr     _Score        _Obs   _RankSum
        1          1           6      76.00
        2          2          18     290.00
        3          3           8     162.00
          z =     1.519
probability =     0.1288
```

When the groups are given any equally spaced scores—such as -1, 0, 1—we obtain the same answer as above. To illustrate the effect of changing scores, an analysis of these data with scores 1, 2, 5 (admittedly not very sensible in this case) gives

```
. nptrend exp, by(group) tr(score)
Test: Trend across groups
    group     _Score        _Obs   _RankSum
        1          1           6      76.00
        2          2          18     290.00
        3          5           8     162.00
          z =     1.464
probability =     0.1432
```

This example suggests that the analysis is not all that sensitive to the scores chosen.

References

Altman, D. G. 1991. *Practical Statistics for Medical Research.* London: Chapman and Hall, 215–217.

Cuzick, J. 1985. A Wilcoxon-type test for trend. *Statistics in Medicine* 4: 87–90.

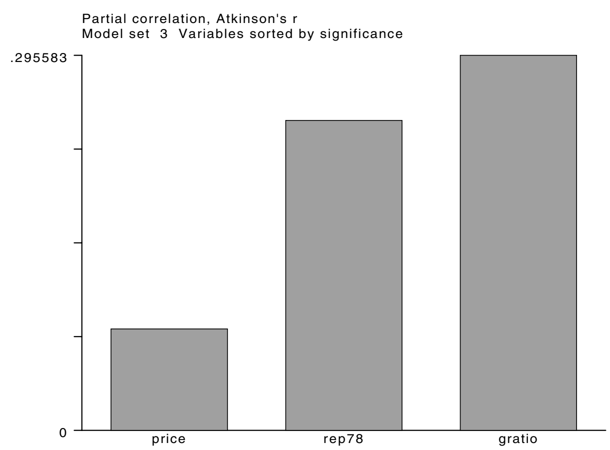
sqv3.1 Graphical display of Atkinson's R values

Marc Jacobs, Dept. of Sociology, Univ. of Utrecht, The Netherlands, FAX (011)-31-30-53 4405

Building on the `lwald` command supplied in STB-7 (Hilbe 1992), I have written a program that graphically displays Atkinson's R values in ascending order. Moreover, the program saves the graph for future printing. It can be located in the `c:\ado` directory as `pc_`*n*`.gph`, where *n* is the number of variables in the model. Histograms are not provided for R values of 0.

The program, called `atrgph`, is used like `logit`. An example partial output is shown below.

```
. atrgph foreign price rep78 gratio
/* Normal logistic regression output not shown  */
Wald Statistics and Partial Correlations (Atkinson´s R)
No.  Var       Wald     Prob(Chi)    Partial Cor
----------------------------------------------------
1    price       2.575     0.109         0.080
2    rep78       7.376     0.007         0.244
3    gratio      9.869     0.002         0.296
----------------------------------------------------
```



References

Hilbe, J. 1992. sqv3: Wald and Atkinson's R extension to logistic. *Stata Technical Bulletin* 7: 18.

sqv4.1 Correction to ldev command output

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

Users of the `ldev` extension to the `logistic` command may have noticed that in some cases the screen display indicating the number of observations in the data set is incorrect. The deviance and χ^2 values are not affected. A fix has been made to `ldev`; simply replace the old program with the new one found on the STB-9 diskette.

sqv5 Univariate log-likelihood tests for model identification

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

Comparing the log-likelihood of a logistic regression model containing only the intercept with that of a model having a single predictor provides *prima facie* evidence of whether the predictor in fact contributes to the model. The comparison statistic is provided by the likelihood-ratio test. At the early model building stage, a p value of .25 or less can be considered adequate for inclusion of a variable as a main effects predictor. However, this does not mean that transformation or collapsing value levels may not prove to later enhance the contribution a variable may make to the full model. What we are really looking for at this stage are high p values. If we find one, at say .80, we can probably exclude it from subsequent analyses. But a caveat—if the variable is likely to serve as a factor in a significant interaction—we may need to retain it regardless of its p value.

I provide a program called `unilogit`. It calculates, for each variable listed after the response variable, the coefficient, standard error, log-likelihood, chi-square (LL ratio statistic), and significance. The intercept-only model log-likelihood is also provided for comparison. The default is one degree of freedom.

The intercept-only log-likelihood value can be calculated directly from the distribution of the response variable. Let n_0 and n_1 represent the respective number of observations having 0's or 1's and let N be the total number of nonmissing observations. The log-likelihood can be determined by

$$LL = n_0 ln(n_0) + n_1 ln(n_1) - N ln(N)$$

After the `logit` command, one can also obtain the LL statistic directly from Stata by

```
LLo = -(_result(6)+(-2*_result(2)))/2
```

where `_result(6)` is the χ^2 and `_result(2)` is the log-likelihood of the model with predictor(s).

The likelihood-ratio test evaluates the hypothesis that the slope coefficient is zero. Given LL_1 as the log-likelihood of the model with the predictor, and LL_0 as the intercept log-likelihood, the ratio is determined by $\chi^2 = 2(LL_1 - LL_0)$.

Example

```
. use lbw

. describe

Contains data from lbw.dta
  Obs:   189 (max= 166927)
 Vars:    12 (max=    99)
Width:    14 (max=   200)
  1. id          int    %8.0g            identification code
  2. low         byte   %8.0g            birth weight<2500g
  3. age         byte   %8.0g            age of mother
  4. lwt         int    %8.0g            weight at last menstrual period
  5. smoke       byte   %8.0g            smoked during pregnancy
  6. ptl         byte   %8.0g            premature labor history (count)
  7. ht          byte   %8.0g            has history of hypertension
  8. ui          byte   %8.0g            presence, uterine irritability
  9. race1       byte   %8.0g            race==white
 10. race2       byte   %8.0g            race==black
 11. race3       byte   %8.0g            race==other
 12. ftv         byte   %8.0g            1st trimester M.D. visits
Sorted by:

. unilogit low age lwt smoke ptl ht ui ftv

                   Univariate Logistic Regression Models

                           1 Degrees of Freedom

Intercept LL = -117.3360

Variable     Coeff       St Error          LL            Chi2         Prob

---------------------------------------------------------------------------
age        -0.0512         0.0315      -115.9560        2.7600      0.0966
lwt        -0.0141         0.0062      -114.3453        5.9813      0.0145
smoke       0.7041         0.3196      -114.9023        4.8674      0.0274
ptl         0.8018         0.3172      -113.9463        6.7794      0.0092
ht          1.2135         0.6083      -115.3249        4.0221      0.0449
ui          0.9469         0.4168      -114.7979        5.0761      0.0243
ftv        -0.1351         0.1567      -116.9494        0.7731      0.3792
```

References

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

srd13 Maximum R-squared and pure error lack-of-fit test

Richard Goldstein, Qualitas, Brighton, MA, EMAIL goldst@harvarda.bitnet

The syntax of `maxr2` is

`maxr2`

This program can only be used after the `fit` estimation command. No options are allowed; the routine will automatically ensure that the same cases that are in your `fit` are used here.

How does one determine the "goodness-of-fit" (GOF) of an ordinary least squares regression? Although many people use R-squared as a summary GOF measure, it is not a good measure for a number of reasons; thus, most users of regression supplement R-squared with a number of more specific, and limited, measures and graphs. Many of these are included in Stata.

There is a situation in which (1) it is possible to obtain a simple summary GOF measure, and (2) in which R-squared is a particularly bad GOF measure. This occurs when one's data set includes "replicates": cases that are tied on every independent variable but that may differ in their values on the dependent variable. In particular, if the replicates do differ in their Y values, it is impossible to obtain an R-squared of 1.0 making the usual use of R-squared questionable (at best).

For instance, we want to predict a person's weight based on their height and have the following data:

Weight	Height
130	66
135	67
140	68
145	69
150	70
155	71
160	72
165	73
170	74
175	75
180	76
185	77

In the above data, we have a perfect relation: for every additional inch of height we have 5 more pounds of weight. Following is the regression:

```
. fit weight height
  Source |       SS       df       MS                  Number of obs =      12
---------+------------------------------               F(  1,    10) =       .
   Model |    3575.00      1    3575.00                Prob > F      =       .
Residual |       0.00     10       0.00                R-square      =  1.0000
---------+------------------------------               Adj R-square  =  1.0000
   Total |    3575.00     11     325.00                Root MSE      =    0.00

------------------------------------------------------------------------------
  weight |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  height |          5          0          .       .               .           .
   _cons |       -200          0          .       .               .           .
------------------------------------------------------------------------------
```

R-squared is 1.0 and everything else is missing since all measures of variance are equal to 0. Now let's add a case with height equal to 70 but weight equal to 145; we also show the results of the `maxr2` routine:

```
. fit weight height
  Source |       SS       df       MS                  Number of obs =      13
---------+------------------------------               F(  1,    11) = 1787.58
   Model | 3696.48422      1 3696.48422                Prob > F      =  0.0000
Residual | 22.7465536     11  2.0678685                R-square      =  0.9939
---------+------------------------------               Adj R-square  =  0.9933
   Total | 3719.23077     12  309.935897               Root MSE      =   1.438

------------------------------------------------------------------------------
  weight |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  height |    5.04772   .1193884     42.280   0.000       4.784948    5.310492
   _cons |  -203.7911   8.531825    -23.886   0.000      -222.5695   -185.0127
------------------------------------------------------------------------------
```

```
. maxr2
max R-square      = 0.9966
relative R-square = 0.9972
Rel. Adj. R-square= 0.9970

SSLF (df) = 10.246554 (10) MSLF = 1.0246554
SSPE (df) = 12.5 (1) MSPE = 12.5

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    0.0820 (10,1)
                                     prob > F =    0.9942

number of covariate patterns = 12
  as ratio of observations: 0.923
```

Note that not only is R-squared no longer equal to 1.0 (and variances are now positive so we have p-values), but that the maximum possible R-squared is less than 1.0 (an R-squared of 1.0 implies that the regression line goes through every point; this is not possible if two points have the same X value(s) but different Y values). The "relative R-square" is the R-square reported by Stata (here, .9939) divided by the maximum possible R-square (.9939/.9966) and similarly for "Rel. Adj. R-square" (except for possible rounding errors). Thus, the "relative" values are again proportions of 1.0.

Following the R-squared information is a pure error lack-of-fit F test, if the p-value is "small," then the fit is not good. This test examines the sums of squares within "replicates." If this test is not significant, and thus the fit is "good," you must still examine your regression for failures of assumptions, influential points, etc.

The final block of information just gives a count of the number of covariate patterns and the ratio of the number of covariate patterns to N, the number of observations.

Note that if we change the 13th data point so that the weight becomes more discrepant, the R-squared information changes, sometimes drastically, but the GOF test does not change. The following two examples demonstrate this. In the first, the weight for the 13th data point is changed to 130 (from 145) and the height is left at 70; in the second, the weight is changed to 110 and the height is again unchanged.

```
. fit weight height

  Source |       SS       df       MS                  Number of obs =      13
---------+------------------------------               F(  1,    11) =  118.15
   Model |  3909.13207     1  3909.13207               Prob > F      =  0.0000
Residual |  363.944857    11  33.0858961               R-square      =  0.9148
---------+------------------------------               Adj R-square  =  0.9071
   Total |  4273.07692    12  356.089744               Root MSE      =   5.752

------------------------------------------------------------------------------
  weight |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  height |    5.19088   .4775538     10.870   0.000       4.139791    6.241969
   _cons |  -215.1644    34.1273     -6.305   0.000       -290.278   -140.0507
------------------------------------------------------------------------------

. maxr2
max R-square      = 0.9532
relative R-square = 0.9597
Rel. Adj. R-square= 0.9561

SSLF (df) = 163.94486 (10) MSLF = 16.394486
SSPE (df) = 200 (1) MSPE = 200

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    0.0820 (10,1)
                                     prob > F =    0.9942

number of covariate patterns = 12
  as ratio of observations: 0.923

. fit weight height

  Source |       SS       df       MS                  Number of obs =      13
---------+------------------------------               F(  1,    11) =   31.75
   Model |  4201.91288     1  4201.91288               Prob > F      =  0.0002
Residual |  1455.77943    11  132.343584               R-square      =  0.7427
---------+------------------------------               Adj R-square  =  0.7193
   Total |  5657.69231    12  471.474359               Root MSE      =  11.504

------------------------------------------------------------------------------
  weight |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
  height |    5.38176   .9551076      5.635   0.000       3.279583    7.483938
   _cons |  -230.3287    68.2546     -3.375   0.006      -380.5561   -80.10138
------------------------------------------------------------------------------
```

```
. maxr2
max R-square      = 0.8586
relative R-square = 0.8650
Rel. Adj. R-square= 0.8527

SSLF (df) = 655.77943 (10) MSLF = 65.577943
SSPE (df) = 800 (1) MSPE = 800

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    0.0820 (10,1)
                                     prob > F =    0.9942

number of covariate patterns = 12
  as ratio of observations: 0.923
```

This comparison helps give a better idea of what is going on: the GOF test is not affected by how much different the one case is, but the maximum R-square is. The test and measure are discussed in Draper and Smith (1981, 33–42); the test is also discussed in Neter, Wasserman and Kutner (1989, 131–140) and Weisberg (1985, 89–95). Each book gives a bivariate regression example; the data for these three are on the disk and the examples follow:

```
. use ds38
(Draper & Smith example, p. 38)
. fit y x
  Source |       SS       df       MS              Number of obs =      24
---------+------------------------------           F(  1,    22) =    6.57
   Model |  6.32466658     1  6.32466658           Prob > F      =  0.0178
Residual |  21.1936683    22  .963348559           R-square      =  0.2298
---------+------------------------------           Adj R-square  =  0.1948
   Total |  27.5183349    23  1.19644934           Root MSE      =   .9815

------------------------------------------------------------------------------
       y |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
       x |   .3378862   .1318692      2.562  0.018       .0644062    .6113662
   _cons |   1.436396   .5900072      2.435  0.023       .2127955    2.659996
------------------------------------------------------------------------------

. maxr2
max R-square      = 0.5468
relative R-square = 0.4203
Rel. Adj. R-square= 0.3939

SSLF (df) = 8.7236679 (11) MSLF = .79306072
SSPE (df) = 12.47 (11) MSPE = 1.1336364

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    0.6996 (11,11)
                                     prob > F =    0.7183

number of covariate patterns = 13
  as ratio of observations: 0.542

. use neter132
(Neter, Wasserman & Kutner example, p. 132)

. fit numnew minimum
  Source |       SS       df       MS              Number of obs =      11
---------+------------------------------           F(  1,     9) =    3.14
   Model |  5141.33841     1  5141.33841           Prob > F      =  0.1102
Residual |  14741.5707     9   1637.9523           R-square      =  0.2586
---------+------------------------------           Adj R-square  =  0.1762
   Total |  19882.9091    10  1988.29091           Root MSE      =  40.472

------------------------------------------------------------------------------
  numnew |      Coef.   Std. Err.       t     P>|t|       [95% Conf. Interval]
---------+--------------------------------------------------------------------
 minimum |   .4867016   .2747105      1.772  0.110      -.1347368     1.10814
   _cons |   50.72251   39.39791      1.287  0.230      -38.40176    139.8468
------------------------------------------------------------------------------

. maxr2
max R-square      = 0.9423
relative R-square = 0.2744
Rel. Adj. R-square= 0.1938

SSLF (df) = 13593.571 (4) MSLF = 3398.3927
SSPE (df) = 1148 (5) MSPE = 229.6

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =   14.8014 (4,5)
                                     prob > F =    0.0056

number of covariate patterns = 6
  as ratio of observations: 0.545
```

```
. use wberg90
(Weisberg example, p. 90)

. fit y x

  Source |       SS       df       MS                  Number of obs =      10
---------+------------------------------               F(  1,     8) =    8.67
   Model |  4.56925017     1  4.56925017               Prob > F      =  0.0186
Residual |  4.21663901     8  .527079876               R-square      =  0.5201
---------+------------------------------               Adj R-square  =  0.4601
   Total |  8.78588918     9  .976209909               Root MSE      =    .726

------------------------------------------------------------------------------
       y |      Coef.   Std. Err.       t     P>|t|      [95% Conf. Interval]
---------+--------------------------------------------------------------------
       x |   .5327329   .1809361      2.944   0.019       .1154935    .9499724
   _cons |   2.090621   .5397843      3.873   0.005       .8458762    3.335366
------------------------------------------------------------------------------

. maxr2
max R-square      = 0.7316
relative R-square = 0.7109
Rel. Adj. R-square= 0.6748

SSLF (df) = 1.8582478 (2) MSLF = .92912389
SSPE (df) = 2.3583912 (6) MSPE = .3930652

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    2.3638 (2,6)
                                     prob > F =    0.1750

number of covariate patterns = 4
 as ratio of observations: 0.400
```

The final two examples use the Stata `auto.dta` for multiple regression—first with three variables and then with two variables and an `if` expression.

```
. use auto
(1978 Automobile Data)

. fit mpg weight w2 foreign

  Source |       SS       df       MS                  Number of obs =      74
---------+------------------------------               F(  3,    70) =   52.25
   Model |  1689.15372     3   563.05124               Prob > F      =  0.0000
Residual |   754.30574    70  10.7757963               R-square      =  0.6913
---------+------------------------------               Adj R-square  =  0.6781
   Total |  2443.45946    73  33.4720474               Root MSE      =  3.2827

------------------------------------------------------------------------------
     mpg |      Coef.   Std. Err.       t     P>|t|      [95% Conf. Interval]
---------+--------------------------------------------------------------------
  weight |  -.0165729   .0039692     -4.175   0.000      -.0244892   -.0086567
      w2 |   1.59e-06   6.25e-07      2.546   0.013       3.45e-07    2.84e-06
 foreign |    -2.2035   1.059246     -2.080   0.041        -4.3161   -.0909003
   _cons |   56.53884   6.197383      9.123   0.000       44.17855    68.89913
------------------------------------------------------------------------------

. maxr2
max R-square      = 0.9808
relative R-square = 0.7049
Rel. Adj. R-square= 0.6922

SSLF (df) = 707.30574 (65) MSLF = 10.881627
SSPE (df) = 47 (5) MSPE = 9.4

F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    1.1576 (65,5)
                                     prob > F =    0.4897

number of covariate patterns = 69
  as ratio of observations: 0.932
```

```
. fit mpg weight w2 if foreign==0
  Source |       SS       df       MS                  Number of obs =      52
---------+------------------------------               F(  2,    49) =   91.64
   Model |  905.395466     2  452.697733               Prob > F      =  0.0000
Residual |  242.046842    49  4.93973146               R-square      =  0.7891
---------+------------------------------               Adj R-square  =  0.7804
   Total |  1147.44231    51  22.4988688               Root MSE      =  2.2226
------------------------------------------------------------------------------
     mpg |      Coef.   Std. Err.       t     P>|t|      [95% Conf. Interval]
---------+--------------------------------------------------------------------
  weight |  -.0131718   .0032307     -4.077   0.000      -.0196642   -.0066794
      w2 |   1.11e-06   4.95e-07      2.249   0.029       1.19e-07    2.11e-06
   _cons |   50.74551   5.162014      9.831   0.000       40.37205    61.11896
------------------------------------------------------------------------------
. maxr2
max R-square      = 0.9590
relative R-square = 0.8228
Rel. Adj. R-square= 0.8155
SSLF (df) = 195.04684 (44) MSLF = 4.4328828
SSPE (df) = 47 (5) MSPE = 9.4
F (dfn, dfd) for lack-of-fit test (MSLF/MSPE) =    0.4716 (44,5)
                                     prob > F =    0.9193
number of covariate patterns = 47
  as ratio of observations: 0.904
```

I know of no other software that provides the maximum R-square measure for multiple regression. MINITAB does provide the pure error GOF test, and I used MINITAB for the two regressions using the auto data—both matched to the precision shown by the two packages. MINITAB also provides, and Draper and Smith and others (see, e.g., Christensen, 1989), a GOF test based on "near-replicates," but I have not found this very useful due to problems in defining what is "near" and have not implemented it.

Finally, weighting is not allowed in the regression, although such an adjustment could probably be added fairly easily (at least for frequency weighting).

References

Christensen, R. 1989. Lack-of-fit tests based on near or exact replicates. *The Annals of Statistics* 17: 673–83.

Draper, N. and H. Smith. 1981. *Applied Regression Analysis*. 2d ed. New York: John Wiley & Sons.

Neter, J., W. Wasserman, and M. H. Kutner. 1989. *Applied Linear Regression Models*. 2d ed. Homewood, IL: Irwin.

Weisberg, S. 1985. *Applied Linear Regression*. 2d ed. New York: John Wiley & Sons.