# The Stata Journal



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

The *Stata Journal* is indexed and abstracted by *CompuMath Citation Index*, *Current Contents/Social and Behavioral Sciences*, *RePEc: Research Papers in Economics*, *Science Citation Index Expanded* (also known as *SciSearch*, *Scopus*, and *Social Sciences Citation Index*.

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

http://www.stata-journal.com

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

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

**Subscription rates** listed below include both a printed and an electronic copy unless otherwise mentioned.

| U.S. and Canada | | Elsewhere | |
|---|---|---|---|
| 1-year subscription | $ 79 | 1-year subscription | $115 |
| 2-year subscription | $155 | 2-year subscription | $225 |
| 3-year subscription | $225 | 3-year subscription | $329 |
| 3-year subscription (electronic only) | $210 | 3-year subscription (electronic only) | $210 |
| 1-year student subscription | $ 48 | 1-year student subscription | $ 79 |
| 1-year university library subscription | $ 99 | 1-year university library subscription | $135 |
| 2-year university library subscription | $195 | 2-year university library subscription | $265 |
| 3-year university library subscription | $289 | 3-year university library subscription | $395 |
| 1-year institutional subscription | $225 | 1-year institutional subscription | $259 |
| 2-year institutional subscription | $445 | 2-year institutional subscription | $510 |
| 3-year institutional subscription | $650 | 3-year institutional subscription | $750 |

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

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

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

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

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

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





The *Stata Journal* (ISSN 1536-867X) is a publication of Stata Press. Stata, Stata, Stata Press, Mata, Mata, and NetCourse are registered trademarks of StataCorp LP.

The Stata Journal (2012)
**12**, Number 4, pp. 761–764

# Stata tip 113: Changing a variable's format: What it does and does not mean

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

## 1 Introduction

Stata variables are all associated with a display format. Users can assign such a display format or work with a display format assigned by default. Thus suppose that you create new variables, for example,

```
. set obs 1
obs was 0, now 1
. generate mynum = 42
. generate mystr = "42"
```

Now type

```
. describe
Contains data
  obs:             1
 vars:             2
 size:             6
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
mynum           float  %9.0g
mystr           str2   %9s
-------------------------------------------------------------------------------
Sorted by:
     Note:  dataset has changed since last saved
```

You can see that `mynum` has been created as a `float` variable with display format `%9.0g` and that `mystr` has been created as a `str2` variable with display format `%9s`. If you do not like either of those formats, you can change them using the `format` command (see [D] **format**).

The problem addressed in this tip is that users are often puzzled over exactly what is meant by changing formats. Part of the problem behind such puzzlement may be linguistic. Sometimes, the term "format" is used vaguely or loosely, such as when "formatting" implies something like initial preparation or transformation of the data. At other times, the term may be used precisely but not in the sense of Stata's `format` command. Thus you may read of long or wide format in the sense of dataset shape or

 

structure. Part of the solution to such puzzlement is thus also linguistic, to remember that "format" here means "display format".

## 2 Applying format

Consider `auto.dta`, in particular the `gear_ratio` variable.

```
. sysuse auto, clear
(1978 Automobile Data)
. codebook gear_ratio
```

```
gear_ratio                                                    Gear Ratio
-------------------------------------------------------------------------

                  type:  numeric (float)
                 range:  [2.19,3.89]                  units:  .01
         unique values:  36                       missing .:  0/74
                  mean:   3.01486
             std. dev:   .456287
           percentiles:        10%       25%       50%       75%       90%
                              2.43      2.73     2.955      3.37      3.72
```

`codebook` usefully reports that this variable has units (some say "resolution": for example, Murphy [1997]) of 0.01, meaning that values for this variable are given to 2 decimal places. Regardless of that, from the results of

```
. summarize gear_ratio
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
  gear_ratio |         74    3.014865    .4562871       2.19       3.89
```

we can see that the default display of `summarize` shows many decimal places. The mean is reported to 6 decimal places and the standard deviation to 7, which is many more than are present in the original data. It can be asserted confidently that `summarize` shows far more minute detail than anyone can use or interpret. However, typing

```
. describe gear_ratio
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------
gear_ratio      float  %6.2f                  Gear Ratio
```

shows that `gear_ratio` has a display format of `%6.2f`. If we wish `summarize` to honor that display format, we must specify the `format` option:

```
. summarize gear_ratio, format
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
  gear_ratio |         74        3.01        0.46       2.19       3.89
```

Now the results may be a little too Spartan for some tastes. One common suggestion is that a standard deviation can be reported a little more precisely than the original data. To get a more precise display, we can change the format of `gear_ratio`:

```
. format gear_ratio %6.3f
```

We see the result when we reissue the `summarize` command:

```
. summarize gear_ratio, format
```

| Variable | Obs | Mean | Std. Dev. | Min | Max |
|---|---|---|---|---|---|
| gear_ratio | 74 | 3.015 | 0.456 | 2.190 | 3.890 |

This example is telling. The first thing to do is look for an option that changes the format of results. If there is not one, or it does not do what you want, then use `format` directly and reissue the command.

Moreover, it should be clear that `format` cannot introduce a third decimal place to the `gear_ratio` data that was never typed in originally, that is, at the time `auto.dta` was compiled. All `format` does is change how data and results are displayed. Even if the format is changed to one coarser than was entered, a detailed check will confirm that the data themselves are unchanged. One way to see this, left as an exercise for you, is to vary the format of some key variable and then check that the results from some interesting command remain completely identical. Use `return list` or `ereturn list` to get a high-resolution display.

# 3 Date formats

Dates are common in many problems but often do not arrive in exactly the right form for analysis. Hence users often want to convert dates as received to some other kind of date.

A common misconception is that changing the date format is the way to change one kind of date to another kind. That is wrong. Suppose we wish to change daily dates to monthly dates. As an experiment, set up a daily date variable:

```
. clear
. set obs 1
obs was 0, now 1
. generate mydate = d(8oct2012)
. format mydate %td
. list
```

| | mydate |
|---|---|
| 1. | 08oct2012 |

Now change the format to monthly:

```
. format mydate %tm
. list
```

| | mydate |
|---|---|
| 1. | 3566m3 |

You may think that the data have changed but to something absurd, so how did Stata mess up? In fact, the data have not changed. The data are still an integer value of 19274, which is October 8, 2012, when you count days from a zero date of January 1, 1960. The integer value 19274 is also March 3566 when you count months from a zero date of January 1960. The result is not what we may have wanted or expected, but Stata's point of view is that it gave us what we asked for, the same data value but interpreted according to a different format.

To convert a date, we need to use an appropriate conversion function. Then, and only then, will `format` work to produce nicer displays:

```
. generate mymonth = mofd(mydate)
. format mymonth %tm
. format mydate %td
. list
```

| | mydate | mymonth |
|---|---|---|
| 1. | 08oct2012 | 2012m10 |

## Reference

Murphy, E. A. 1997. *The Logic of Medicine*. 2nd ed. Baltimore: Johns Hopkins University Press.