Title

stata.com

concept: gph files — Using gph files

Description Remarks and examples Also see

Description

`.gph` files contain Stata graphs and, in fact, even include the original data from which the graph was drawn. Below we discuss how to replay graph files and to obtain the data inside them.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Background

`.gph` files are created either by including the `saving()` option when you draw a graph,

```
. graph ..., ... saving(myfile)
```

or by using the `graph save` command afterward:

```
. graph ...
. graph save myfile
```

Either way, file `myfile.gph` is created; for details see [G-3] ***saving_option*** and [G-2] **graph save**.

At some later time, in the same session or in a different session, you can redisplay what is in the `.gph` file by typing

```
. graph use myfile
```

See [G-2] **graph use** for details.

Gph files are machine/operating system independent

The `.gph` files created by `saving()` and `graph save` are binary files written in a machine-and-operating-system independent format. You may send `.gph` files to other users, and they will be able to read them, even if you use, say, a Mac and your colleague uses a Windows or Unix computer.

Gph files come in three forms

There are three forms of `graph` files:

1. an old-format Stata 7 or earlier `.gph` file
2. a modern-format graph in as-is format
3. a modern-format graph in live format

You can find out which type a .gph file is by typing

```
. graph describe filename
```

See [G-2] **graph describe**.

Live-format files contain the data and other information necessary to re-create the graph. As-is format files contain a recording of the picture. When you save a graph, unless you specify the asis option, it is saved in live format.

Advantages of live-format files

A live-format file can be edited later and can be displayed using different schemes; see [G-4] **schemes intro**. Also, the data used to create the graph can be retrieved from the .gph file.

Advantages of as-is format files

As-is format files are generally smaller than live-format files.

As-is format files cannot be modified; the rendition is fixed and will appear on anyone else's computer just as it last appeared on yours.

Retrieving data from live-format files

First, verify that you have a live-format file by typing

```
. graph describe filename.gph
```

Then type

```
. discard
```

This will close any open graphs and eliminate anything stored having to do with previously existing graphs. Now display the graph of interest,

```
. graph use filename
```

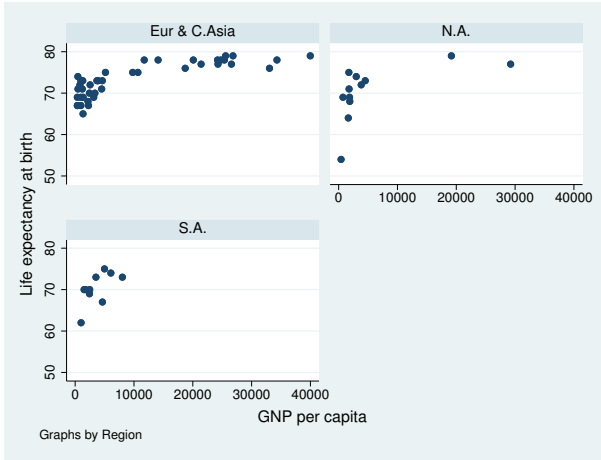
and then type

```
. serset dir
```

From this point on, you are going to have to do a little detective work, but usually it is not much. Sersets are how graph stores the data corresponding to each plot within the graph. You can see [P] **serset**, but unless you are a programmer curious about how things work, that will not be necessary. We will show you below how to load each of the sersets (often there is just one) and to treat it from then on just as if it came from a .dta file.

Let us set up an example. Pretend that previously we have drawn the graph and saved it by typing

```
. use http://www.stata-press.com/data/r15/lifeexp
(Life expectancy 1998)
. scatter lexp gnppc, by(region)
```




```
. graph save legraph
(file legraph.gph saved)
```

Following the instructions, we now type

```
. graph describe legraph.gph
legraph.gph stored on disk
        name:  legraph.gph
      format:  live
     created:  20 Jun 2016 13:04:30
      scheme:  s2gcolor
        size:  2.392 x 3.12
    dta file:  C:\Program Files\Stata15\ado\base\lifeexp.dta dated 26 Mar 2016 09:40
     command:  twoway scatter lexp gnppc, by(region)
. discard
. graph use legraph
. serset dir
  0.  44 observations on 2 variables
      lexp gnppc
  1.  14 observations on 2 variables
      lexp gnppc
  2.  10 observations on 2 variables
      lexp gnppc
```

We discover that our graph has three sersets. Looking at the graph, that should not surprise us. Although we might think of

```
. scatter lexp gnppc, by(region)
```

as being one plot, it is in fact three if we were to expand it:

```
. scatter lexp gnppc if region==1 ||
  scatter lexp gnppc if region==2 ||
  scatter lexp gnppc if region==3
```

The three sersets numbered 0, 1, and 2 correspond to three pieces of the graph. We can look at the individual sersets. To load a serset, you first set its number and then you type `serset use, clear`:

```
. serset 0
. serset use, clear
```

If we were now to type `describe`, we would discover that we have a 44-observation dataset containing two variables: `lexp` and `gnppc`. Here are a few of the data:

```
. list in 1/5
```

	lexp	gnppc
1.	72	810
2.	74	460
3.	79	26830
4.	71	480
5.	68	2180

These are the data that appeared in the first plot. We could similarly obtain the data for the second plot by typing

```
. serset 1
. serset use, clear
```

If we wanted to put these data back together into one file, we might type

```
. serset 0
. serset use, clear
. generate region=0
. save region0
. serset 1
. serset use, clear
. generate region=1
. save region1
. serset 2
. serset use, clear
. generate region=2
. save region2
. use region0
. append using region1
. append using region2
. erase region0.dta
. erase region1.dta
. erase region2.dta
```

In general, it will not be as much work to retrieve the data because in many graphs, you will find that there is only one serset. We chose a complicated `.gph` file for our demonstration.

Also see

[G-2] **graph display** — Display graph stored in memory

[G-2] **graph manipulation** — Graph manipulation commands

[G-2] **graph save** — Save graph to disk

[G-3] ***saving_option*** — Option for saving graph to disk

[P] **serset** — Create and manipulate sersets