B Advanced Stata usage

Contents

B.1 Executing commands every time Stata is started

Stata looks for the file `profile.do` when it is invoked and, if it finds it, executes the commands in it. Stata looks for `profile.do` first in the directory where Stata is installed, then in the current directory, then along your path, and finally along the ado-path (see [P] **sysdir**). We recommend that you put `profile.do` in your bin directory `$HOME/bin`.

Say that every time you start Stata, you would like to start a dated log for the session. In `$HOME/bin`, create the file `profile.do` containing this rather odd-looking command (the command will be explained at the end of this session):

```
log using `: display %tCCCYY-NN-DD-HH-MM-SS ///
   Clock("`c(current_date)' `c(current_time)'","DMYhms")', ///
   name(default_log_file)
```

When you invoke Stata, the usual opening appears but with the following additional command, which will be executed:

```
running /home/mydir/bin/profile.do ...
```

How does the command work? Let's work from the inside out:

- `c(current_date)` and `c(current_time)` are local system macros containing the current date and current time. See [P] **creturn** for more information.
- The left (`) and right (') quotes around the local macros expand them. See [P] **macro** for a full explanation.
- The `Clock()` function uses the resulting date string and the date mask `"DMYhms"` to create a datetime number Stata understands. See [D] **datetime**.
- The format `%tCCCYY-NN-DD-HH-MM-SS` formats this number in year-month-day-hour-minute-second form because this will make the files sort nicely. See [D] **datetime display formats** for the details.
- The odd-looking `` `: display ...' `` allows the formatted date to be used directly in the command as the file name. This is the advanced concept of an in-line expansion of an extended macro function. You can see more in [P] **macro**.
- The `log using` command starts a log file, such as shown in [GSU] **16 Saving and printing results by using logs**.
- The `name` option gives the log file the internal name `default_log_file` so that it will not likely conflict with other log files. See [R] **log** for details.
- Finally, the `///` notations are continuation comments so that the three separate lines are interpreted as a single command. See [P] **comments** for more about comments.

There are many advanced Stata programming concepts in this one single command!

`profile.do` is treated just as any other do-file once it is executed; results are just the same as if you had started Stata and then typed `run profile.do`. The only special thing about `profile.do` is that Stata looks for it and runs it automatically.

System administrators might also find `sysprofile.do` useful. This file is handled in the same way as `profile.do`, except that Stata first looks for `sysprofile.do`. If that file is found, Stata will execute any commands it contains. After that, Stata will look for `profile.do` and, if that file is found, execute the commands in it.

One example of how `sysprofile.do` might be useful would be when system administrators want to change the path to one of Stata's system directories. Here `sysprofile.do` could be created to contain the command

```
sysdir set SITE "/opt/stata/ado"
```

See **[U] 16 Do-files** for an explanation of do-files. They are nothing more than ASCII text files containing sequences of commands for Stata to execute.

B.2 Advanced starting of Stata for Unix

The syntax of the command to start Stata(GUI) is

`xstata` [*-option* [*-option* [...]]] [*stata_command*]

The syntax of the command to start Stata(console) is

`stata` [*-option* [*-option* [...]]] [*stata_command*]

If you have Stata/SE, the commands are `xstata-se` and `stata-se`. If you have Stata/MP, the commands are `xstata-mp` and `stata-mp`.

The allowable options are

Option	Result
`-b`	set background (batch) mode and log in ASCII text (console only)
`-h`	display usage diagram
`-q`	suppress logo and initialization messages
`-s`	set background (batch) mode and log in SMCL (console only)

Typing `stata -h` does not start Stata, but just shows the syntax diagram for invoking Stata.

The `-q` option starts Stata, but it suppresses all the initialization messages, including the Stata logo.

The `-b` and `-s` options specify batch mode; see **[GSU] B.3 Stata batch mode**.

B.3 Stata batch mode

Suppose you had a do-file named `bigjob.do`. If you want to use Stata in batch mode, we suggest using Stata(console). Typing

```
% stata -b do bigjob
```

tells Stata to execute the commands in `bigjob.do`, suppress all screen output, and route the output to `bigjob.log` in the same directory.

```
% stata -s do bigjob
```

tells Stata to execute the commands in `bigjob.do`, suppress all screen output, and route the output to `bigjob.smcl` in the same directory.

You can also run the above examples in the background by typing

```
% stata -b do bigjob &
% stata -s do bigjob &
```

You may also use redirection, but this is not recommended:

```
% stata < bigjob.do > bigjob.log &
```

Warning: Redirection will not work if your do-file contains either the `#delimit` commands or comment delimiters (`/*` and `*/`, `//`, or `///`). It also cannot create SMCL output. Hence, we recommend using options directly: `stata -s do bigjob &` or `stata -b do bigjob &`.

Note: Stata runs `profile.do` before doing `bigjob.do`, just as it would if you were working interactively.

B.4 Using X Windows remotely

Suppose that you are sitting in front of a computer named `local` and that you wish to run Stata on a computer named `neighbor`.

1. Tell X Windows on `local` to allow `neighbor` to use its display by typing `xhost +neighbor`.
2. Be sure that you have set X Windows' `DISPLAY` environment variable on `neighbor` to contain `local:0.0`. X requires this. Important: this variable must be set on `neighbor`.

Having done this, Stata should work:

```
local% xhost +neighbor
local% ssh neighbor
neighbor% setenv DISPLAY local:0.0
neighbor% xstata
```

At this point, either Stata launches, or you see

```
Xlib:  connection to "local:0.0" refused by server
Xlib:  Client is not authorized to connect to Server
```

Here you will have to get help from your network administrator. Proper authorizations have not been given, and these problems have nothing to do with Stata.

To make this process simpler, you may also be able to use the `-X` flag when invoking `ssh`:

```
local% ssh -X neighbor
neighbor% xstata
```

Whether this works depends on your Unix installations. The first series of commands should always work.

B.5 Summary of environment variables

Environment variable	Description
HOME	User's home directory. Default is the directory specified in /etc/passwd.
PATH	Unix executable search path.
SHELL	What to execute when users try to shell out of Stata. Default is /bin/sh.
S_ADO	Sets Stata's ado-path.
STATATERM	Used only if you want to use a different termcap or terminfo entry from what is in your TERM environment variable.
STATATMP	Sets Stata's temporary directory. Default is /tmp.
STATA_PREF_DIR	Location Stata checks for preferences. If not set, Stata will use a subdirectory of $HOME/.stata13/. If you use Stata(console), preferences will be saved in $HOME/.stata13/console/. If you use Stata(GUI), preferences will stored in a subdirectory based on your DISPLAY environment variable, such as $HOME/.stata13/:0.0/.

B.6 Memory size considerations

Beginning with Stata 12, memory management in Stata became automatic—there is no longer any requirement to manually request memory for Stata when you know you will be using a large dataset. For details on efficiency tweaks needed by a very few Stata users, look at [D] **memory**.