How can I generate automated filenames in Stata?

Authored by

stats writer

July 1, 2024

Automated filenames in Stata refer to the process of automatically naming files based on a pre-defined naming convention or criteria. This can be achieved by using Stata's programming language, which allows for the creation of customizable code to generate filenames. This feature is particularly useful for organizing and managing large datasets, as well as automating repetitive tasks. By using automated filenames, users can save time and effort in manually naming files, while also ensuring consistency and accuracy in file naming.

How can I generate automated filenames in Stata? | Stata FAQ

If you are creating multiple datasets in Stata, you may wish to name them in an automated manner. The code below demonstrates how to create a filename that is based on the date and timestamp. Such a file naming process will 1) prevent you from giving the same filename to two different datasets (assuming they are not created in the same minute) and 2) allow you to see when files were first created. The code below presents an example.

We first generate local macros for the current date and time.

local c_date = c(current_date)

local c_time = c(current_time)

In the first line of code below, we concatenate the two string variables.
To concatenate two strings variables we list the first string in quotation marks
(e.g. "`c_date'") followed by a plus sign (+) and the second string
in quotation marks. We want an underscore between the two string variables, so
we include that in the series of strings to be concatenated. We then use display
to look at the string we have so far. Notice that the string contains spaces and colons, which can
be problematic in file names.

local c_time_date = "`c_date'"+"_" +"`c_time'"
display "`c_time_date'"

3 Feb 2009_11:08:14

Below we use the function subinstr(...) to "clean up" our string.
subinstr(...) takes four arguments, the first is the string

we want to
modify (e.g. "`c_time_date'"), followed by the string we want to replace
(e.g. ":"), and what we want to replace it with (e.g. "_").
The final argument is the number of replacements we want to make, since we want
to replace every instance of a colon we use a period (.), but we could
also have used the number 2, since we know there are two colons in our string.
The second line of syntax below replaces the blank spaces (i.e. " ") with
underscores (_). The third line displays the result so we can see what our
string looks like now. This time our string is in a better form for a file name.

```
local time_string = subinstr("`c_time_date'", ":", "_", .)
local time_string = subinstr("`time_string'", " ", "_", .)
display "`time_string'"
```

3_Feb_2009_11_08_14

In the first line below we load a dataset so that we have

something to save.
Next, we save the existing dataset
using the macro variable "time_string" in the name of the saved dataset.

```
use https://stats.idre.ucla.edu/stat/stata/notes/hsb2, clear
save `time_string'.dta, replace

file 3_Feb_2009_11_08_14.dta saved
```

It is possible that we might want to prefix the file name with some other value.
In the syntax below we repeat the save command, this time prefixing it with the
file name with "myfile_" so that our filename will be myfile_ followed by
the date and time.

```
save myfile_`time_string'.dta, replace

file myfile_3_Feb_2009_11_08_14.dta saved
```