A Comprehensive Guide to Coding and Programming in Stata

Rafael Gafoor

1. Introduction

i. Why is programming for analysis of Electronic Health Records different from other analyses?

Using Stata for Electronic Health Records is more complicated than when you analyse other types of datasets. This is because you need to assemble the dataset you will be using for the analysis. In Stata this involves use of macros which can be difficult to understand. For this reason macros are introduced now at the very beginning. They can be challenging but time spent now understanding their complexity will pay off dividends in the long term.

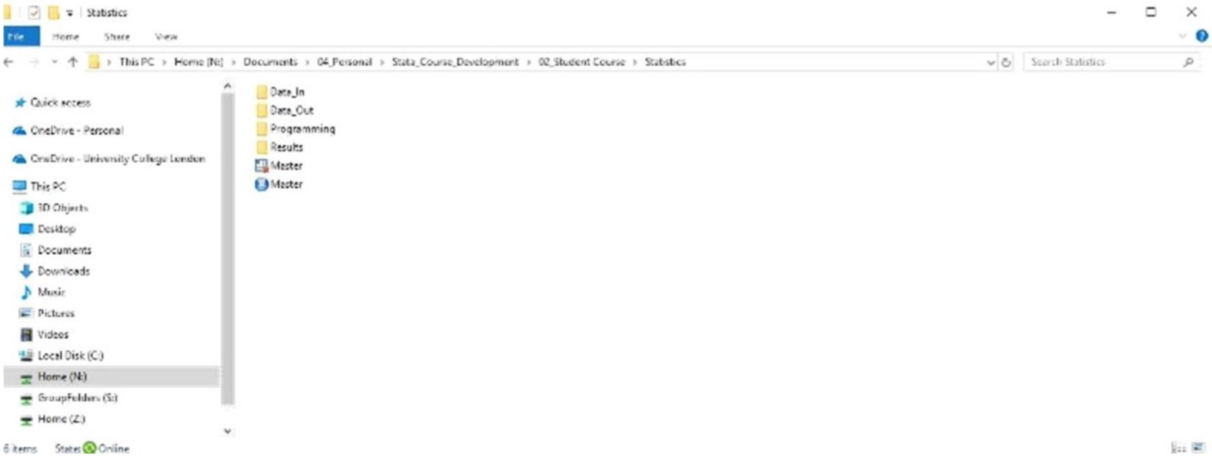
ii. The programming environment

It's important to keep a very clean programming environment to make sure you do not EVER overwrite your primary dataset. You should from the outset create a directory entitled something along the lines of "Stata EHR Course" and then place 4 folders within it

Data_In
Data_Out
Programming
Results

You can place additional folders within these folders, but this is the main structure

NOTE It is important to use the underscore and not to leave blank space when programming. While leaving blank space is not necessarily an issue for Stata users on Windows, it becomes a major issue for some R commands and in other programming environments.



iii. Data Download

You should now download your data from Moodle or the data stick and place the files into your file named “Data_In”. This file now should never be overwritten. Any temporary files or files which you make in the interim should now be placed into the folder named “Data_Out”.

iv. R and Stata (Filepaths)

Many analysts of Electronic Health Data program in R and Stata. It is possible to program entirely in R and in the future this may be the preferable programming language. However, if you are new to analysis and to programming you may find it easier to start in Stata. Many supervisors at UCL only program in Stata, so you may have little choice in the program you use to analyse the data. You may be working on a project that someone else has started in which case it wouldn’t make much sense to start all over again and reinvent the wheel.

However, the graphics in R using ggplot2 is superior to almost anything which you may find in Stata and many people use Stata to carry out the analysis and R to produce the graphics. This can lead to some entertaining difficulties. For the present, it would be important to program in such a manner that you can use your code efficiently in both environments in the event that you become bilingual in the future.

The most pressing issue at this stage is how to write filepaths so that they can be understood in both computing environments. "" This is a backslash; it is named because of the direction of the TOP of the hyphen with respect to the bottom. This "" is a forward slash.

NOTE: The direction of the hyphens. Stata doesn't really care which ones you use - either backslash or forward slash but R is very rigid and will only accept forward slash. The windows operating system produces file paths using backslashes. However, if you are going to use R and Stata simultaneously it's better to change them as you go along to forward slashes. You can program R to accept backslashes but it's easier at the beginning to just change the slashes to the correct direction.

v. Changing working directory

The command **pwd** will tell you where your current working directory is located.

```
pwd

.pwd
c:\EHR_Course
```

You now need to move this working directory to the folder you created entitled **"Stata EHR Course"** or equivalent.

```
cd  "C:/EHR_Course"

. cd "C:/EHR_Course"
C:\EHR_Course
```

Everything now is coded in relation to this home directory. It means that if you move the folder or send it to a colleague for further work, they only ever need to change one filepath and the file will work. This is called Hard Coding. This is a very important concept that you should **NEVER** hard code except at the beginning of your code and make it explicitly obvious so that it's easy for others to easily identify where code has to be amended for it to work. This hardcoding is usually placed at the top of the master file. One obvious hard code is your working directory.

vi. Working from your home directory

This home directory will become your base and you very rarely if ever move outside it!!

If for example, you wish to create files and/or folders within your home directory, you will sometimes need to tell Stata where the home directory is. This location is stored in a macro within Stata and you can access it this way.

For example, to find a list of the files and folders within the Data_In folder you can issue the following command:

```
dir "`c(pwd)'/Data_In/"

<dir> 4/30/19 15:15 .
<dir> 4/30/19 15:15 ..
<dir> 4/30/19 15:15 British_Election_Survey
<dir> 1/27/19 20:07 Regional_Data
<dir> 4/30/19 15:15 Stata_Data
<dir> 4/05/19 10:07 Station_Data
<dir> 4/30/19 15:15 Tim´s datasets
```

The macro c(pwd)' is where the contents of your working directory are stored and you can use this as a short cut in filepaths so that you do not hard code. Once you set the working directory,c(pwd)’ will always point to the correct location.

vii. Folder structure

Your coding for all projects encompasses several steps in data processing, production of interim datasets, graphs, tables etc. If you code all of your program for an assignment in one file it can become very long and you can’t easily distinguish the stages in your analysis.

It is essential that you create a master file that you can use to call the sub programmes from. The master file sits in the top level of the folder structure and the sub directories and files for your programming steps sit in the folder named “Programming”. You may wish to additional folders in programming entitled **“Data_Input”**, **“Data_Processing”**, **“Tables”** etc.

viii. Creating a “Master File”

At the root level of the analysis folder, place a Stata .do file entitled Master. This file will call all your subprograms and contains three additional crucial pieces of information.

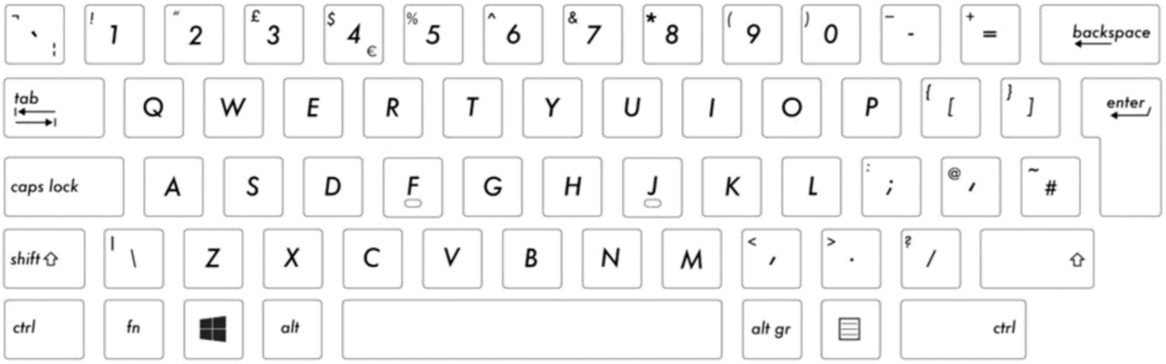
At the top of the Master file you should include some preliminary information about the date the file was created, the name of the programmer, the purpose of the file, the version of Stata under which it was made, the organization to which the programmer is associated & the name of the programmer.

The next step is to create a section where you place all the hardcoding in your analysis. This should be the only place where hard coding is present. This allows the analysis to very easily ported across computing environments.

The second step is to place all of your global macros in a section clearly defined for this purpose. You will learn more about global macros later and the reason why this step is so important.

ix. Special characters

Be sure you can identify the backtick, apostrophe, backslash, forwardslash and curly bracket characters. These will be used extensively in the course.


¬ ` ¦ ! 1 " 2 £ 3 $ 4 € % 5 ^ 6 & 7 * 8 (9) 0 _ - + = backspace
tab Q W E R T Y U I O P { [}] enter
caps lock A S D F G H J K L : ; @ ' ~ #
shift | \ Z X C V B N M < , > . ? /
ctrl fn alt alt gr ctrl