# Introduction to use of Stata in VHM 802/812

**Jenny Yu**
**January 12, 2021**

## 1. This session assumes you have the following knowledge of Stata:

**Understand and/or know how to:**

- Interact with Stata; organize your files
- Load non-Stata format data into Stata (mainly Excel files and csv files);
- Variable types, data types, variable naming conventions; missing values;
- Convert between numeric variables and string variables;
- Generate a categorical/an indicator variable from a continuous/string variable;
- Label a variable; define a value label and attach it to a variable;
- Explore/use Stata Menus, Toolbar and dialog boxes; create/edit graphs by menus/graph editor;
- Create clear, reproducible do-files and log files in your working folder;
- Use Stata functions

## 2. Steps in the data analysis:

- Planning the organization of your files and directories
  - systematically think ahead - **simple and consistent** principles
    - Directories on your hard drive – where to store your course and/or research materials
    - One (sub-)directory only for one project – so it's **easy to find** your project files
    - Names for projects, folders and files – all **mnemonic**
  - **Separate do-files for data management and statistical analysis**
    - Individual do-files: typically (but not always) start with the letters cr or an
    - Or create your own rules for your do-files' names
- Data management: well documented do-files
  - Verify your data are accurate and no duplicate observations; missing values are properly handled
  - Generate the variables that are required for the next step of statistical analysis; make sure all variables are **well named and properly labeled**
  - …
- **Running statistical analysis**
- Presenting results – interpret your analysis results
- Protecting files – **documentation** to keep track of what you have done and thought

## 3. Review Stata Language syntax:

- . h language
  **[prefix :] command [varlist] [=exp] [if] [in] [weight] [using filename] *[, options]***
  - Square brackets are optional (as opposed to the required), fill them in only if needed;
  - Each item should be typed exactly as they appear in the order of the diagram;
  - *Options,* are specified at the end of the command; a comma must precede the first option;
  - Click the links in the language help file to find out each of the items in the language syntax:
    - Prefix…
- . h more //or . h set
  **set more {on|off} [, permanently]**
  - Must select one item inside the curly brackets {}

- . use daisy2red, clear
  - . h codebook
  - . codebook, c

## 4. A sample do-file for statistical analysis: 1al.do

- Robust
  - Replace line 4 in 1al.do with your own working directory, then click 'Execute(do)' on the Toolbar
- Legible
  - Comments & blank lines

### 4.1 Robust – A few simple commands: usually at the beginning of a do-file

- Version control
  - . version 16.1
- Prevent –more- condition
  - . set more off
- Change working directory – always use it
  - . cd "r:\"
- Log the output of do-files*
  - . log using assignment1.smcl, replace
  - . log close // at the end of your do-file
- Include seeds for random numbers*
  - . help set seed
  - . set seed 20210112

### 4.2 Legible – Comments, blank lines and long lines in do-files

- Comments throughout the do-files:
  - . help comments
- Blank lines: freely include blank lines in your do-file to make your do-files more clear
- Split a long line into several lines to make long lines more readable
  - Use three forward slashes (///) where you want to split the line
  - Use /*
    */ in the two different lines

## 5. Debugging do-files: determining the source of an error

- Read the error messages or click error code:
  - If you understand the error message, there is no need to explore the error code
- Common errors – fix them
  - no; dataset in memory has changed since last saved
    - ask yourself whether you want to overwrite the current dataset in memory
  - log file already exists
    - ask yourself whether you want to overwrite the log file
  - incorrect command name:
    - . codbook // unrecognized command:
  - incorrect option of the command:
    - . codbook, b //option b not found
    - . twoway (scatter milk120 parity, mksymbol(circle_hollow)) //option mksymbol() not allowed
  - incorrect variable name:
    - . twoway (scatter milkl20 parity, msymbol(circle_hollow)) //variable milkl20 not found

- missing comma before option:
  - . use daisy2red clear //invalid 'clear'
  - . twoway (scatter milkl20 parity msymbol(circle_hollow)) //variable circle_hollow not found
  - . predict fit xb //too many variables specified
- A few more error messages:
  - not sorted; variable xxxx already defined; too many values, varlist required, = exp required, using required, by() option required …

- Tolerate errors - allowing the command/do-file to continue despite errors
  - . capture: suppressing all its output (including error messages, if any) and issues a return code of zero
    - . h error
  - . capture log close
  - . capture drop rawres stdres delres

    (note: Stata commands do either exactly what you say or nothing at all. If one of the three variables did not exist in the data, drop would then do nothing. It would not drop the other two. To achieve the desired result, we must give three commands:

    . capture drop rawres
    . capture drop stdres
    . capture drop delres
- Other methods to resolve errors
  - Drop all derived/generated variables, trying an alternative approach
  - Run the do-file in steps
  - Reload the data set
  - Restart Stata

## 6. The workflow of the do-file: 1al.do

- Understand the data
  - Understand the data structure
  - Assure the outcome variable and all independent variable(s) are numeric variables
- Describe the variables
  - Understand the distribution of each of the variables on its own
  - Understand the nature and strength of relationships among variables
  - Create summaries or distribution tables
    - Data > Describe data > Describe data contents (codebook)
    - Data > Describe data > Describe data in memory or in a file
    - Statistics > Summaries, tables, and tests > Summary and descriptive statistics
    - Statistics > Summaries, tables, and tests > Frequency tables
    - Statistics > Summaries, tables, and tests > Other tables
  - Create graphs
    - Graphics Menu
    - Statistics > Nonparametric analysis > Lowess smoothing
- Statistical model
  - Statistics Menu/commands
- Model evaluation/postestimation:
  - After the statistical model, click the menu Statistics > Postestimation
    - Acquire model predicted values
      - Generate variables/values required for model forecasting and/or graphs
      - Display/List values
      - Create graph(s)
    - Evaluate the model validity: compute model residuals

- Individual residual checking
  - List the extreme residuals: list command
- Overall residual checking
  - Can use summary statistics commands: such as
    - . list
    - . summary
    - . tabstat
  - Assessing normality and homoscedasticity of residuals
  - Other assessment: residuals versus fitted value – linearity
- Is the model valid?
  - Yes – interpretation and reporting
  - No – go back to modify the previous model

## 7. Do-files (recap): Robust & Legible

### 7.1 Robust means:

- A do-file produces exactly the same result when rerunning the do-file at a later time or on different computers (only need to change the working directory)
- A do-file is self-contained:
  - Should not rely on something left in memory by a prior do-file or commands run from the Command Window
  - Should not use a dataset unless it loads the dataset itself
  - Should not compute a test of coefficient unless it estimates those coefficients in the same do-file
- Explanation for the commands:
  - Use version control - Stata might change the command names or computational methods
  - Change directory – exclude directory information or keep minimum directory location in commands that read or write files
  - Include seeds if random numbers required
    - If try to replicate results that use random numbers, you need to use the same random numbers; otherwise you will obtain different results

### 7.2 Legible means:

- Internally documented and carefully formatted
  - Make the content/logic clear
  - Help you and your supervisor(s)/collaborator(s) easier to debug and understand what you did
- Several practical steps:
  - Use lots of comments
  - Use blank lines
  - Use alignment and indentation if you prefer
  - Split a long line into several lines