TOPIC 8

COMPUTER PRACTICAL

INTRODUCTION TO COMPUTING FOR STATISTICAL DATA ANALYSIS

We are now going to learn how to use the statistical computer package Stata to analyse data and carry out the various statistical tests of this course. Practical sessions from here onwards are almost entirely based on Stata. This double practical session provides an introduction to Stata.

Many students find Stata confusing, at least at first. Please allow time for your confidence to grow in using Stata. The only way to become proficient in Stata is to practice. The STEPH classes will introduce you to the basic commands, structure and syntax, but feel free to explore each dataset, and to try things in Stata – it is very hard to break Stata, so experiment with commands to see what happens! Each session will build on the previous one and we believe you will gain in confidence and understanding as you go through them.

In order to start this process, this first double session concentrates on using Stata and does not bring in any new statistical concepts or techniques. In most of the subsequent sessions you will use Stata to practice the statistical techniques that you learned in the previous lecture.

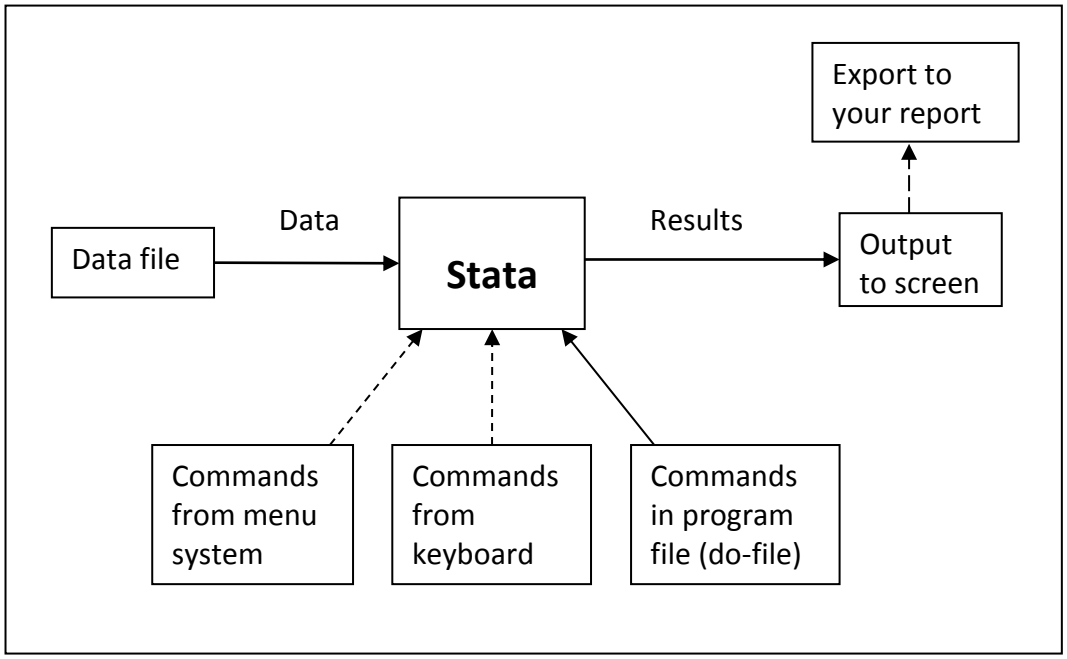
The notes here, and the commands we introduce throughout STEPH, are not a complete guide to Stata. We encourage you to experiment, and to investigate what Stata is capable of. The help files for each command are a good place to start (simply type the word help followed by the name of the command), and you'll also find lots of useful tips and tricks online – as always, Google is your friend if you get stuck.

How Stata works

The typical way we use Stata is as follows:

1. Load data
    a. Optionally change / manipulate the data
2. Run statistical commands / tests
3. Get output on screen: could be a single number, tables, graphs etc
    a. Optionally, export these results from Stata in a form to be used in (for example) a Word document

A schematic diagram of what happens is as follows:



Instructions can be given to Stata in three ways, as shown above:-

1) Via the command window
2) Using drop-down menus (point & click)
3) Via a do file – this is a text file containing multiple Stata commands

In this course, we will always use (3). Using a Do file allows you to save the code you write, and recreate your results at a later date. This is especially important when working on a project which won't be finished in one session.

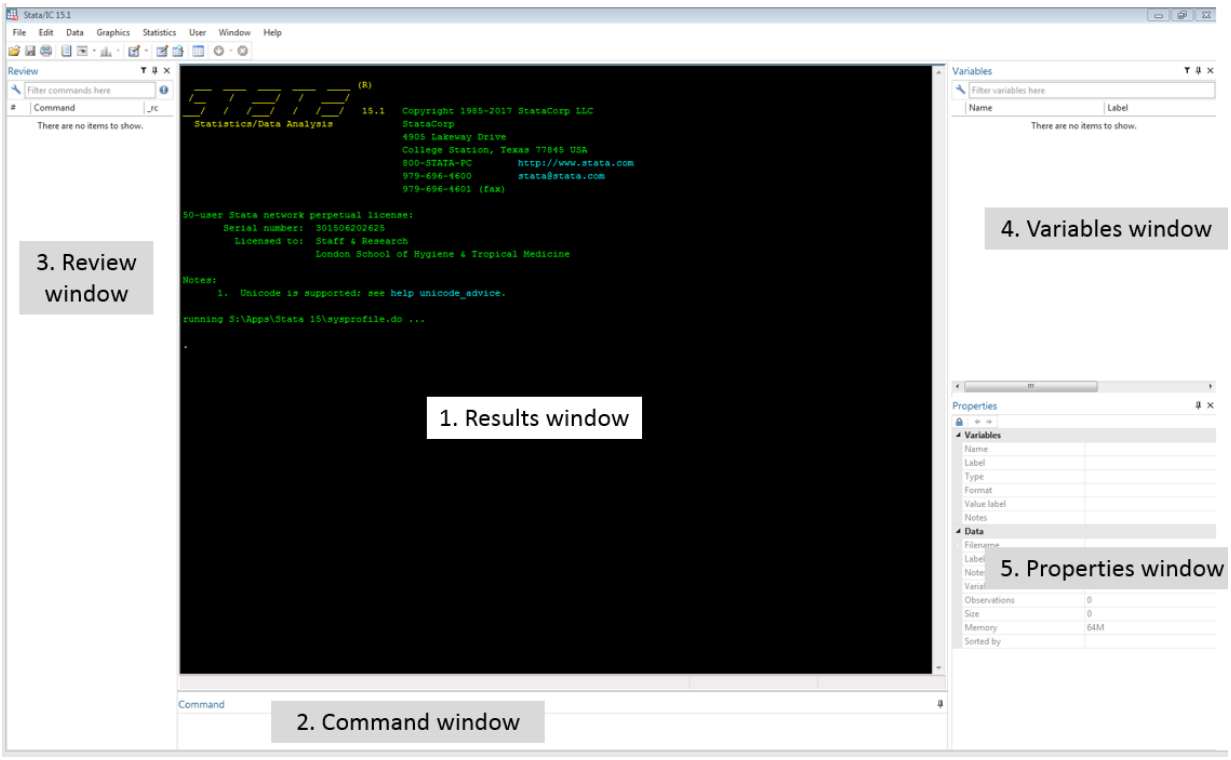
Although (2) can be an attractive approach, expressing a command in text helps us to understand the details of it. It also relies on you remembering which commands you clicked on. It can be a useful method of discovering new commands, which you can then add to your Do files.

You do not need to memorise the information in this guide! Rather, you should use it as a reference source you can come back to throughout the module (and in fact your entire Masters).

Getting started with Stata

To enter Stata double-click on the Stata 15 icon 

in the *Research* folder of the *Novell-delivered Applications window*. After a few seconds, you should get a screen labelled Stata/IC 15 that looks something like this:



Five windows are displayed. They are:

1) The Results window. This is where all of your results will appear! (Almost all – graphs will appear in a new window)
2) The Command window. Commands can be typed into this window, one at a time, followed by [Enter].
3) The Review window. This lists all of the commands that you have run. You can re-run a command by double-clicking it here (or single-click it to make it appear in the Command window)
4) The Variables window. When there is a dataset loaded into Stata, all of the variable names appear here. You can search them by typing in the box above this window.
5) The Properties window. This displays some useful information about your dataset: number of variables, number of observations etc.

Each of these windows can be resized and moved around. The fonts in a window can be changed by right-clicking within the window.

We have said that we will use a do file to enter commands to Stata. Why do we say that?

What is a Do file? And why should I use one?

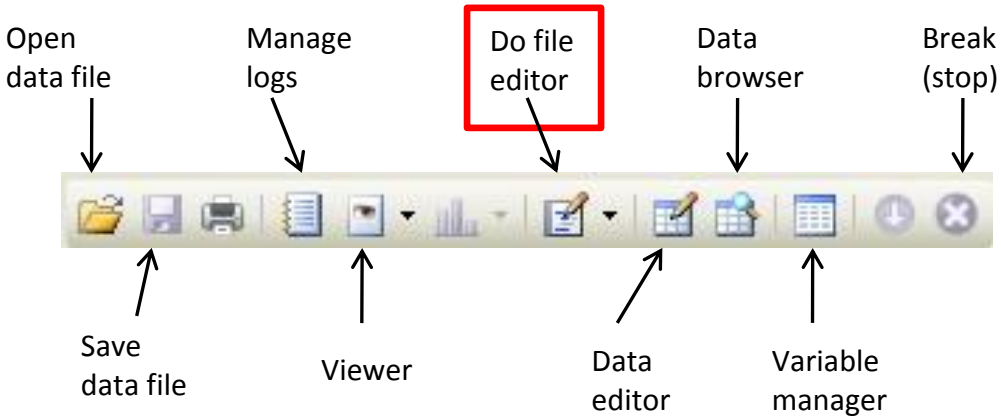
A do file is a simple text file which contains multiple lines of Stata commands. When you run a Do file, or selected lines from a Do file, you tell Stata to run all of the commands in the Do file in order.

There are multiple reasons why you may wish to use a do file instead of the command prompt:

1. You keep a record of all of your commands, which can be shared with collaborators/supervisors/your future self
2. You can run dozens - even hundreds - of commands with one click of a mouse
3. You can make comments and annotations within the Do file, to remind yourself what you were doing (especially useful if you work on a project over a period of time – like, for instance, your summer project…)
4. You can run more complex commands, such as loops (although we don't cover that in this course)

Using a Do file

The first thing we need to learn is how to open a Do file. There are two ways: using a "point & click" approach with the mouse, or, alternatively, type "doedit" into the command window and press [Enter].



You can write in the do file editor in the same manner as any text editor (like Word or Notepad). Type in the commands you will want to run in Stata, using one line per command. When you are ready to run the commands **highlight the ones you wish to run** and press "Ctrl + D" or click the "Execute (Do)" button, on the far right of the icons at the top of the screen:

If you do not highlight any lines in your do-file and then execute the do-file, Stata will try and run all the lines in your do-file, so be careful to highlight just the lines you'd like to run.

When you open the Do file editor you'll be given a new, blank, Do file to use. Alternatively, you can open a pre-written file. To open the Do file which accompanies this class, either press

"Ctrl+O", or use the "Open" icon on the toolbar. The Do file is called "Intro_to_Stata.do", and it contains many of the commands we explain in these notes.

Opening a dataset, setting the working directory

The first thing you'll want to do in Stata is to open a dataset. To do so, you'll first need to tell Stata where the data is. So, if you have made a folder called "STEPH" on your H drive, and inside that made a new folder called "Practical_8", you could run:

```
cd "H:\STEPH\Practical_8"
```

The working directory is shown at the bottom left of the Stata window (just above the Start button).

And then to open a dataset (which you must have saved in the above folder first):

```
use BAB.dta , clear
```

The "clear" tells Stata to erase the data in memory, and replace it with this new dataset. In general, anything after the comma is referred to as an "option" of the command (in this case the command is "use").

Another way to get data into Stata is importing from an Excel spreadsheet; this is useful when collaborators send you raw data, which you want to use for statistical analysis.

```
import excel using "Masaka.xlsx", sheet("2012") firstrow clear
```

The option "firstrow" tells Stat that the first row of the data contains variable names. The option "sheet" tells Stata which worksheet to import – the default is to use the first worksheet.

Explore the data

There are many commands in Stata that you can use to explore your data. Some are given below. Most of them will allow you to add variable names after the command to restrict the output to just your variables of interest.

```
describe
codebook
summarize ht gestwks
browse
```

Getting Help

Stata has extensive Help facilities for both general queries and particular commands. If you know the name of the command for which you need help then you can type

```
help command_name
```

If the command doesn't exist Stata will search for you and present commands which may be of use. You can then click on the blue command names to get help for the specific command.

Help may seem incomprehensible at first but all help files have the same structure, so over time you should get used to how to read them. You may even get to like it, given time!

Exercise in entering data and producing simple summaries

The data that we will use for learning about Stata consists of information from 641 mothers of singleton births after in-vitro fertilisation. The file containing these data is called BAB.dta. Like all of the STEPH datasets, it is saved in U:\Download\Teach\STEPH. You should copy all of these onto your H drive in a suitably named folder (we recommend "H:\STEPH\Practical_8").

Table: Variables in the BAB dataset

Variable	Units or Coding	Type	Variable Name
Subject Number	-	-	id
Maternal Age	Years	Quantitative	matage
Hypertension	1=yes, 0=no	Binary	ht
Gestational age	Weeks	Quantitative	gestwks
Sex of infant	1=male, 2=female	Binary	sex
Birthweight	Grams	Quantitative	bweight

The data give, for each woman, her age, whether or not she had hypertension, the gestational age at birth, the baby's sex, and the birth weight. Recall from Practical 2 that categorical and binary variables record which category a woman falls into. The different categories are often coded using numbers, and they may sometimes have a logical order, as in the categories: poor, medium, good. When there are only two categories the categorical variable is a binary variable; there are two of these in this dataset. Quantitative variables record a measurement or count of some kind.

You can look at the just the first ten rows of data using browse or list

```
browse in 1/10
list in 1/10
```

The notation "1/10" means "from 1 to 10".

Let us now get Stata to produce some summaries of these data. We will start by making a one-way table to get the numbers and percentages of babies by sex. Type in the command window:

```
tab sex
```

In most datasets there will be some missing values. These are coded using a full stop symbol [.] in place of the value which is missing. The codebook command (shown above) is useful for seeing whether there are missing values. You can also add the "missing" option to the tab command

```
codebook sex
tab sex, missing
```

Hint

If you don't want to type the variable names, you can use the variable window: double-click on the name of the variable you want and it will appear in the command window.

How are the hypertensives divided by the sex of the child? You can tabulate these two variables against each other by typing this command:

```
tab sex ht
```

We can also look at averages of birthweight. Type the command:

```
summarize bweight
```

or, in an abbreviated form:

```
summ bweight
```

Note that in the results here you get more than you asked for. You also get the standard deviation and the range of birth weights. It is common for a command in a statistical package to give you more than you want; just take what you do want. You can get even more information by using the "detail" option.

You can also get the mean with this command:

```
mean bweight
```

Some general points about Stata commands

We will usually control Stata by typing a command at the prompt. For this, it is useful to learn a series of commands to perform specific tasks. The general form of a Stata command is

```
by varlist: command varname(s) if... in... using... , options
```

by repeats the command for each group of observations for which the values of the variables in varlist are the same; it is optional as part of the command. if and in lead to selecting a subset of records on which the command is executed; these are also options. using calls other data files (and is limited to only a few commands), and options are features specific to each command. You only sometimes need to use by, if, in or using, so the minimum form of a command is

```
command variable_name(s)
```

Stata always distinguishes between upper and lower case characters. Note that you must always use lower case when typing commands, and we recommend that, in general, you also use lower case for variable names.

Stata accepts abbreviations for commands and variable names providing they are not ambiguous. Thus, we have used tab for tabulate. The help command describes the syntax required for each command. The letters (or symbols) underlined are the minimum letters it is necessary to type for Stata to understand which command or option you are entering.

Getting results out of Stata

When using any statistical computer package it is important to know how to keep a record of any analysis performed. In Stata there are many ways to do this, but the simplest is by the use of a log file. A log file is simply a text file containing all of the output which is displayed in the Results window.

As a log file is just a text file it can be edited in a text editor, or copied into a word-processor e.g. Word. Within the word-processor the file can be opened, edited and then printed if required. When editing Stata output in, say, Word, you should change the font to Courier New (a fixed-width "typewriter" font), as the output is not very readable otherwise. The log file will contain all typed commands and the textual output from the analysis. It will not contain any graphics or listings from the help facility in Stata. It is possible to save graphic output from Stata, but not as a log file.

When starting a session in Stata you will need to open a log file, in which to save your output. This means that you instruct the computer to add all the text output from the screen to this file.

Let's say you wanted a record of all babies with birthweight less than 2kg, in descending order. You could use the following commands.

```
log using low_birthweight.txt , replace text
gsort bweight
list if bweight < 2000
log close
```

An existing log file can be added to, leaving all the old text as it is, so new text is added on the end, or overwritten, deleting all the old text and effectively starting a new file with the same name. To append, use the option "append". Use "help log" to see the other things you can do with log files.

Hint: Running single commands in the Command Window

You can type a single command directly into the Command Window (window 2, above). This allows you to quickly look at data or to try a new command. You will see that the commands you type are displayed in the Review Window (window 3) after you hit enter. It is possible to re-issue commands, and avoid retyping, by clicking on them in the Review window – when you do so they'll appear back in the Command Window. You can also copy commands from the Command Window (either individually or in groups) by highlighting them, right-clicking with your mouse, and selecting copy. You can then paste them into your Do file.

Alternatively, while your cursor is in the Command Window you can press [Page Up] or [Page Down] to recall and edit previous commands.

Both of these are tips that will save a lot of typing!

When starting to look at any new data the first step is to check that the values of the variables make sense and correspond to the codes defined in the coding schedule. For categorical variables one can look at one-way frequency tables and check that only the specified codes occur. For quantitative variables we need to look at ranges and histograms.

Questions

See if you can answer the following questions, using the commands you have learned so far:

1) *From the tables in the results window:*
   a) *How many male infants are there?*
   b) *How many mothers were hypertensive during pregnancy?*
2) *How many male infants were born to hypertensive mothers?*
3) *What was the prevalence of hypertension among mothers of boy and girl babies?*

The basic output from a cross tabulation reports frequencies only; to include row and/or column percentages and/or percentages based on the overall total, add the options `row`, `column`, `cell`, or any combination, as in

```
tab ht sex, column
```

This table gives the percentages to answer question 3.

4) *What was the percentage of babies who were male, among hypertensive mothers?*

The tabulate command can be used with any variable, provided the number of different values is not too large. However, the results are not very useful when there are lot of possible values. To demonstrate this, try

```
tab matage
tab gestwks
```

The first is useful but the second is almost the same as listing the values of gestwks.

Statistics on quantitative variables

The command `summarize` has already been used to give means and standard deviations for all variables together. It can be shortened to `summ`. To describe a particular variable such as bweight in more detail, type

```
summ bweight, detail
```

Using the `detail` option with the summarize command will give the mean, the standard deviation, the median, a selection of percentiles and the minimum and maximum values of bweight. It will also give statistical properties of the distribution of bweight called the skewness and kurtosis (beyond our scope here, and another example of the computer telling you more than you wanted to know).

One method to obtain the means and standard deviations of bweight separately by sex is to type

```
tab sex, summ(bweight)
```

This is an extension of the tab command in which the quantitative variable in the `summ()` is summarized for each category of the variable being tabulated.

5) *How does the mean birthweight for males compare to the mean birthweight for females?*

Restricting which subjects a command works on

As mentioned above when the command list was presented, Stata commands can be restricted in their range of operation to (for example) records 1, 2, ... , 10, by adding **in 1/10** to the command.

You can find out how many records there are by typing:

```
count
```

In the BAB dataset there are 641 records; this command lists the last 100

```
list in 541/641
```

Commands can also be restricted to operate only on records which satisfy given conditions. The conditions are added to the command using **if** followed by a logical expression. For example, to restrict the command list to records with birthweight less than or equal to 2000g, type

```
list if bweight <= 2000
```

Other useful logical expressions are **= =** for equal, and **!=** for not equal.

[Note: The double equals sign (= =) is **not** an error.]

Generating and recoding variables

Sometimes, you may need to create new variables. For example, low birthweight is defined as a birthweight of less than 2500 grams. To distinguish between normal and low birthweight babies, one needs a binary variable which could be coded 0 for normal birthweight babies and 1 for those with a low birthweight. This new variable can be "generated" from the actual birthweights. There are many different ways of generating new variables from existing variables, one method is described below.

New variables are generated using the command generate which adds a new variable (column) to the dataset. The contents of the new variable can then be changed (recoded) using `recode`. For example, suppose we wish to group the values of maternal age into four categories:

Values of matage	Value of new variable
20, 21, 22, 23, 24, 25, 26, 27, 28, 29	1
30, 31, 32, 33, 34	2
35, 36, 37, 38, 39	3
40, 41, 42, 43	4

We can use the following commands to first make a new variable called `matagegp`**:**

```
generate matagegp=matage
```

The contents of matagegp are the same as the contents of matage (you can browse to check this). Next, we change the contents of matagegp so that each observation contains the appropriate code (from 1 to 4 as in the table above):

```
recode matagegp min/29=1 30/34=2 35/39=3 40/max=4
```

Lastly, we give the variable an informative text label:

```
label variable matagegp "Categorical maternal age"
```

To make a new binary variable for groups of gestational age (below or above 37 weeks for gestational age) the commands are:

```
generate gestcat=gestwks
recode gestcat min/36.99=1 37/max=2
label variable gestcat "Pre-term baby"
```

By first creating a new variable, with values copied from the original variable, and then recoding this, the original variable is left unchanged. Note that gestwks, unlike matage, has non-integer values (there are numbers after the decimal point), so the categories must be specified more precisely. Again, the label command attaches some information to the new variable.

6) Create a binary variable for low birthweight (defined as above) and label the new variable

The name "lbw" is suitable, but you may prefer another.

The generate command can also be followed be used to "transform" a variable using an "expression" e.g. `generate bwtkgs=bweight/1000` generates a new variable bwtkgs (bweight in kilograms). The generate command can be abbreviated to `gen`.

It is sensible to browse, or to use the `list` command on, say, the first 10 records to check that the recoding or transforming has worked. In the case of matage and matagegp you could cross-tabulate the two to check that it looks correct – "`tab matage matagegp`".

The arithmetic and logical operators and their priority

Expressions follow the usual rules about order of calculation, parentheses etc. Expressions can include operators and functions. The standard operators are as follows:

	Arithmetic		Logical		Relational
+	Add	&	And	>	Greater than
-	Subtract	!	Not	<	Less than
*	Multiply	~	Not	>=	Greater than or equal to
/	Divide	|	Or (Shift and the	<=	Less than or equal to
^	Power		key to the left of z)	==	Equal
				!=	Not equal to

When to use = and when to use ==
You use = if you want to set the values of one variable equal to those of another.
You use == if you are comparing two sets of values; if you want to test whether one thing is equal to the other.
If you use = when you should have used == the error message is not particularly helpful and just says there is a syntax error.

Internally, Stata stores missing values (which you see as .) to have a value larger than all other possible values. This can lead to problems when there are missing values in the data. For example, `list if bweight>=2000` would display records with missing `bweight` if there were missing values in `bweight`. In this case, one should type:

```
list if bweight>=2000 & bweight<.
```

To exclude any records with missing `bweight`. This will not affect any mathematical calculation in Stata since subjects with missing value(s) are automatically excluded from such calculations, but can cause problems if you are generating categorical variables from continuous ones, so be careful!

See `help operators` and `help functions` for more information.

Saving data

In order to keep the dataset with these new variables for use in future sessions, click on *File* in the menu bar and select ***Save As***. Replace the highlighted "*.dta" in the 'File name:' box by typing *babnew* over it and click on Save. Since a file of this name already exists in the directory (a ready-made version of this file was copied over when you copied files from the U: drive at the beginning of this session), a Windows dialog box will appear asking whether you want to replace the existing file. Click on Yes to replace it.

Alternatively, you may type

```
save babnew, replace
```

This will save your data into your working directory with the name babnew. If a file with the same name already exists in the directory, you have to add the option replace, after a comma. The replace option tells Stata to overwrite the existing data file. Be careful when you use this option because once a file has been overwritten the original cannot be retrieved.

The data currently in memory are now saved in the file called **babnew.dta**. This contains all the data from the BAB dataset plus the new variables you have generated (matagepg, gestcat and lbw).

Sorting

The records in a dataset can be sorted according to the values of one or more variables. The `babnew` dataset is currently sorted by `id` but for some purposes it might be better to have it sorted by `bweight`. Try:

```
sort bweight
list id bweight in 1/30
```

The records are now in order of bweight and the id numbers and all other variables have also been sorted in this order.

Some Stata commands which use the option `by()` require the data to be first sorted by the variable in the `by()` option and the sort is not done automatically. For many commands you can use the `bysort()` option instead which will sort the data for you. Graph commands only allow the `by()` option, but these commands don't mind how the data are sorted.

Graphical displays

The graph command has many options. Bar charts are used to display the distributions of categorical variables, while histograms and box plots are used to display the distributions of quantitative variables.

To obtain a histogram of bweight, type

```
histogram bweight, frequency
hist bwe, freq // This is the same, it just saves typing!
```

The keyword frequency ensures that the number of women is marked on the vertical axis; an alternative is `percent`. You can vary the number of bars in the histogram (called bins) by adding `bin(8)` or `bin(10)`, or whatever. To superimpose the histogram with a normal curve which has the same mean and standard deviation as the data, add the option `normal`. Try, for example,

```
hist bweight, bin(10) percent normal
```

7) What do you conclude about the distribution of birthweight?

An alternative to the histogram is the box plot, which allows you to compare the distribution of one variable by groups of another variable. The following command draws two box plots for the bweight variable, one for babies born before 37 weeks and one for babies born later:

```
graph box bweight, by(gestcat)
```

8) What do the box-plots suggest about the relationship between birthweight and gestational age?

Scatter plots can be used to evaluate the association between `bweight` and the quantitative variables `matage` and `gestwks` by typing

```
scatter bweight matage
scatter bweight gestwks
```

9) What do these plots suggest about:

a) the relationship between maternal age and birthweight?
b) the relationship between gestational age and birthweight?

To combine more than one scatter plot in the same graph it is easiest to use the **`by()`** option. For example, to show a scatter plot of `bweight` against `gestwks` separately for each value of `sex`, type:

```
scatter bweight gestwks, by(sex)
```

(Be careful not to insert a space between by and (sex); the error message is not helpful if you do this).

Sometimes, if you are not sure how to generate a graph correctly, using the drop down menus can be a useful way to figure out how to do it. When you run from the drop down menu and get the graph that you want, you can copy the command and paste it into your do-file.

Finishing up

You should now have a new dataset called babnew.dta and a Do file. In general, it is often more important to save your Do file than your dataset, as you can always re-create the data by re-running the Do file. Therefore, remember to save your Do file, give it a sensible name, and include helpful comments throughout!

Exit from Stata

When you have finished using Stata, you can close it in one of three ways:

- Click the close button, X, at the top right hand corner of the Stata window
- Select the File menu from the menu bar and select Exit
- Type "`exit, clear`" in the Stata command window and press [Enter]

If any do files or datasets have not been saved, Stata will prompt you to save them now.

Further Exercises

If you still have some time attempt the following. You may choose to save some of the results in a log file.

10. Read in the BAB data from the file babnew.dta which contains the two new variables you created during this session.

11. List the variables bweight and ht for records 20-25 inclusive.

12. Obtain the frequency distribution of matagegp. What does it show?

13. Use count to find how many hypertensive women have babies with birthweight less than 2000g. (hint: try help count)

*14. Create a new variable called **gestcat4**, by grouping the values of gestwks into min to 34.99, 35 to 36.99, 37 to 38.99, 39 to max, and recoding them as 1, 2, 3, 4 respectively. Use the list command to check that the recoding has worked.*

15. Obtain the one-way table of frequencies for the new categorical variable gestcat4. What does it show?

16. Fill in Tabl on the next page by analysing the BAB data.

17. Find the mean values and standard deviations (SDs) of bweight within each category of the other variables and fill in Table. What do you conclude from the results in tables 3 and 4?

Table: Distribution of birthweight, gestational age, and maternal age, by sex.

Variable	Result	Sex	
		Male	Female
Birthweight (grams)	Number		
	Mean		
	SD		
Gestational Age (weeks)	Number		
	Mean		
	SD		
Maternal age (years)	Number		
	Mean		
	SD		

Table: Mean and SD of birthweight by gestational age, maternal age, sex, and hypertension.

Variable	Categories	Number	Mean bweight	SD
Gestational age (weeks)	<37			
	≥37			
Maternal age (years)	<30			
	30-34			
	35-39			
	40+			
Hypertension	No			
	Yes			

Appendix: Tips for writing do files

1. Write lots and lots and lots of comments. Comments are notes written to the user of the do file and are ignored by Stata. A comment will automatically display in a different colour than your commands. There are two ways to incorporate comments into your do file.

a. For a short comment on a single line: Use "*" to start the comment.

b. For a long comment which may be several lines in length: Use "/*" to start your comment and "*/" to end the comment.

2. When you have written a new command into your do file, run the do file from the start (CTRL-D) to make sure all your commands work together. The do file will stop running if there is an error.

3. When you write a command that becomes too long to read easily, use "///" at the end of the line to tell Stata that you are splitting the command across the present and next line.

4. Be careful about using abbreviations (such as "tab" for "tabulate") for commands and variable names. Your collaborators - and particularly non-Stata users - may have difficulty deciphering abbreviations.

5. To log your output, it is recommended that you precede the command "log using" with the command "capture log close". When you first run your do file, this first command will be ignored. When you re-run your do file, your log file will close before re-opening.

6. Give your log file the same name as the do file that generated it, eg "myanalysis.log" and "myanalysis.do" so it is easy to see these two files are related.

7. Indentations and lines with no text are both ignored by Stata. You can use these to make your do file more structured and readable.

8. When writing a do file, save it early and save it often. Stata does not automatically save your work.

Advanced tips

9. If you precede a command with "capture", you are telling Stata to ignore an error associated with the command and to continue with the do file. In addition to the "capture log close" command (see above), "capture cd" is useful for using the same do file across different computers, whereby you can name multiple folder locations, and only the relevant folder option will run.

10. Write one do file to complete all data management steps on your original dataset (eg data cleaning, variable creation, labelling, and recoding). Write another do file for all of the statistical analysis steps. Include a command “do” to run your data management do file at the start of your analysis do file.

11. Write lots and lots and lots of comments! If you share your do file with a collaborator, or need to return to your analysis far into the future, you will really appreciate all of the comments.

For additional advice on using Stata, see Long, J. Scott “The workflow of data analysis using Stata.” (2009) Stata Press, College Station, TX, USA.