UNIVERSITY OF ESSEX
INSTITUTE FOR SOCIAL AND ECONOMIC RESEARCH
Professor Stephen P. Jenkins <stephenj@essex.ac.uk>

Essex Summer School course 'Survival Analysis'
and
EC968. Part II: Introduction to the analysis of spell duration data

Lesson 1. Preliminaries – Introduction to Lessons and Stata

Contents

1 Aims of the course

- To provide an introduction to the analysis of spell duration data ('survival analysis'); and
- To show how the methods can be implemented using Stata (http://www.stata.com), a program for statistics, graphics and data management.

The focus is on models for single-spell survival time data (with no left censoring or left truncation). The examples are based on Stata version 10 for Windows. (Most of the programs in the Exercises below can be run using version 6 onwards, but I cannot guarantee this.)

Steve Pudney's part of the course (EC968 Part I), on panel data analysis, also uses Stata.

2 Course resources

- The Lessons (ec968st1 to ec968st8), Stata data sets, and do files downloadable from http://www.iser.essex.ac.uk/teaching/degree/stephenj/ec968/index.php;
- Powerpoint presentations from the lecture course (the URL from which these are available will be given to you separately);
- The other materials, including web-based resources, cited below.

3 Lessons

1. Preliminaries – Introduction to Lessons and Stata (this document)
2. The shapes of hazard and survival functions
3. Preparing survival time data for analysis and estimation
4. Estimation of the empirical (KM) hazard and survivor functions
5. Estimation: (i) continuous time models – parametric and Cox
6. Estimation: (ii) discrete time models
7. Unobserved heterogeneity ('frailty')
8. Competing risks
9. Assorted other topics

4 How to use these resources

These materials are a do-it-yourself learning resource complementing the lectures. Work through the Lessons below, using Stata, in parallel with the lectures. There is material to read followed by exercises. Do files (names prefixed by 'ex') provide code to reproduce the material shown in the lessons and also to do the exercises. You are encouraged to run the do files yourself (type **do filename** where filename.do is the name of the do file) – preferably after attempting the exercises by yourself! Please bring any questions about the lessons to the lectures, where we shall make space to discuss them as required.

The rest of this Lesson introduces Stata. There are exercises on Stata basics at the end.

5 Acknowledgements

For obvious reasons I have drawn heavily on the Stata Reference Manuals written by the StataCorp staff. They are superb, and useful as a text not just a program manual. I have also drawn inspiration and, in some cases, directly used material, from other Stata users (many of whom are cited in the Stata Resources pages referred to below). In addition to the StataCorp staff, I would specifically like to cite the contributions of Jeroen Weesie (Utrecht University) and Nick Cox (Durham University). Feedback from cohorts of students has also improve the material.

These 'Survival analysis with Stata lessons' parallel my lecture course. A full set of acknowledgements appears in the Lecture Notes 'book' that accompanies this course.

The responsibility for the content of these Lessons is mine alone. Please email comments, questions, and suggestions to me (stephenj@essex.ac.uk).

6 Typographical conventions used to cite Stata commands

In the text below, words in the text written in Times Roman bold refer to Stata commands. For example, **help findit** means type the command "help findit" and then hit the Return key. Output from Stata echoed to the screen is written in `Courier New 8-point font`.

7 Stata: an overview

Stata is a sophisticated and comprehensive program for program for statistics, graphics and data management. 'Stata' rhymes with 'data' and it is spelt 'Stata' (not 'STATA' as it is a made-up word, not an acronym).

Stata for Windows is available in the University PC labs.

Stata is easy to use, but is also very powerful. It has a large range of built-in tools, but is also programmable. Stata programs and data sets are platform-independent. All the tools required for this course (and for much applied analysis of any kind) are provided in Stata. There is also a large user community which shares expertise and programs, and many extra resources are freely available via the web.

7.1 Overview of Stata's commands

This is taken from the Subject Table of Contents in the Stata version 8 *User's Guide*. The topics used most in this course are indicated by *.

Data manipulation and management
- basic data commands*
- functions and expressions*
- dates

- inputting and saving data*
- combining data
- reshaping datasets
- labeling, display formats, and notes
- changing and renaming variables
- examining data*
- miscellaneous data commands

Utilities
- basic utilities
- error messages
- saved results
- internet
- data types and memory
- advanced utilities

Graphics*

Statistics
- basic statistics*
- anova and ancova
- linear regressions and related maximum-likelihood regressions*
- logit and probit regression*
- pharmacokinetic statistics
- survival analysis*
- time series
- cross-sectional time series (panel data)
- auxiliary regression and related commands
- commands for epidemiologists
- analysis of survey data
- transforms and normality tests
- nonparametric statististics
- simulation/resampling
- cluster analysis
- factor analysis and principal components
- do-it-yourself maximum likelihood estimation
- quality control
- other statistics

Matrix commands
- basics
- programming
- other

Programming
- basics
- program control
- parsing and program arguments
- console output
- commonly used programming commands
- debugging
- advanced programming commands
- special interest programming commands
- file formats

7.2 Stata resources

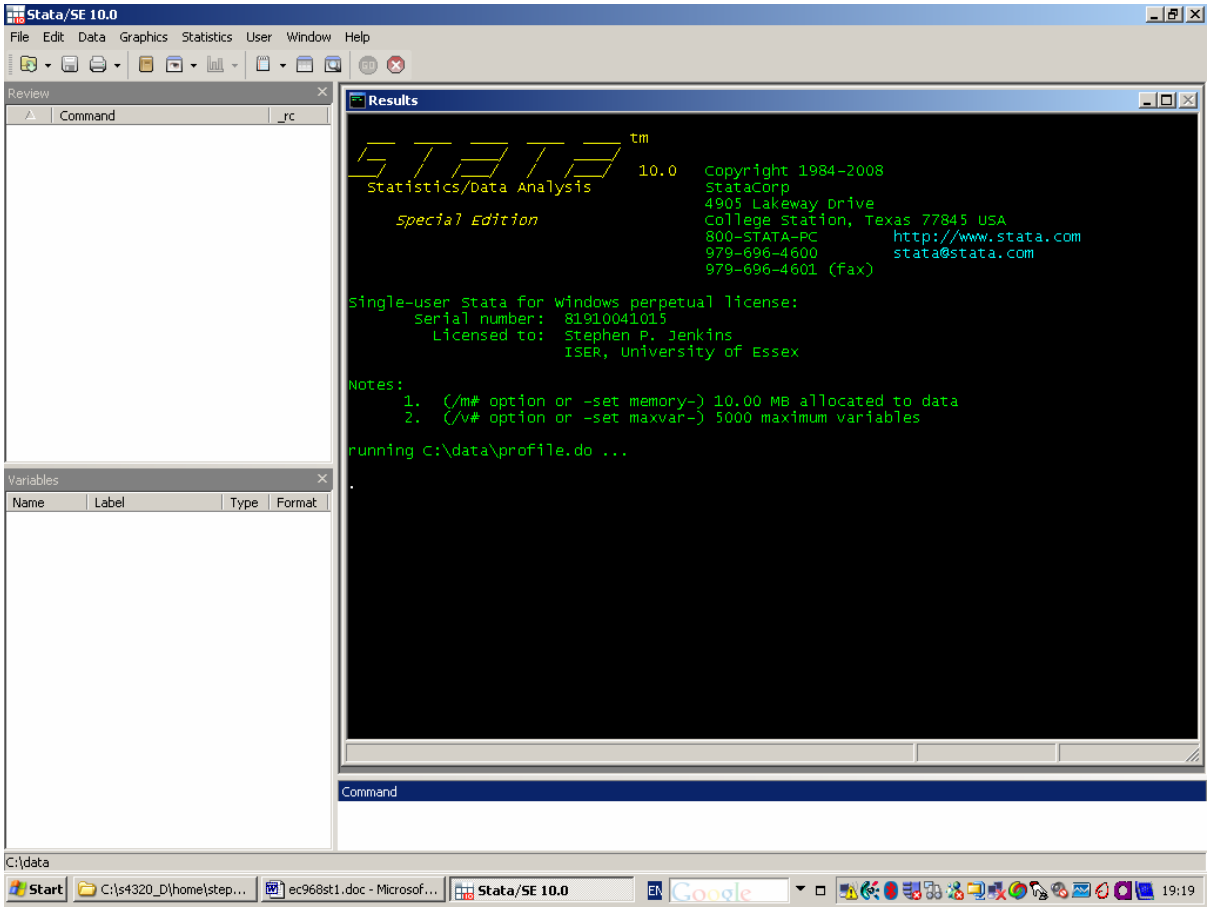
- Stata's on-line **help**, **search**, and **net search** commands, and especially **findit**. The **search** commands search Stata's keyword database, including manuals, FAQs and STB (see below); **findit** is a powerful web search engine for Stata materials. Start with this. E.g. type **findit survival**.
- Read the short Stata manual *Getting Started with Stata for Windows*. Move on to the *User's Guide*. The reference manual set is like an encyclopaedia: dip into it as you need to. Manual errata are published at http://www.stata.com/support/errata.
- Stata is 'web-aware'. You can access the StataCorp website and other Stata web-based resource from within Stata itself. Official updates to Stata, programs distributed with the *Stata Journal*, and other user-written programs are accessible via the drop-down Help menu or **net**. Do you have the latest version of official Stata? Updates to official Stata (executable and ado files) are downloadable for free: type **update query** and follow instructions.
- The Stata website, http://www.stata.com/links/resources.html contains links to many other resources, including websites providing tutorials on Stata.
- The UCLA Academic Technology Services site with everything from learning how to use Stata to worked exercises from leading texts: http://www.ats.ucla.edu/stat/stata/. This is a fantastic resource.
- Check StataCorp's own Frequently Asked Questions (FAQs) site at http://www.stata.com/support/faqs.
- Statalist, a valuable email listserv discussion group for all Stata users. New and experienced users, including StataCorp staff, contribute questions and answers; also a source for user-written programs. Past correspondence is available via searchable archives. See http://www.stata.com/support/statalist
- *The Stata Journal* is a quarterly refereed publication: see http://www.stata.com/support/faqs/res/sj.html or www.stata-journal.com. It contains refereed articles about statistical topics addressed using Stata. A high quality resource, accompanied by Stata programs written by users (freely downloadable). The *SJ* replaces the Stata Technical Bulletin (STB): see http://www.stata.com/support/stb/faq
- The SSC-IDEAS Software Archive, Boston College, found at http://ideas.uqam.ca/ideas/data/bocbocode.html contains an extensive library of programs written by Stata users, regularly updated, and searchable. The files can be freely downloaded from inside Stata using the **ssc** command. (Use **findit**)
- All the Stata data sets cited in the Stata Manuals, and in Cleves *et al.* (2002) are downloadable from http://www.stata-press.com/data/.
- Stat/Transfer (http://www.stattransfer.com) is a very useful file format conversion program (to and from, for example, Stata, SAS, SPSS, ascii). DBMS/Copy (http://www.dataflux.com/dbms/) is an alternative.
- Data can also be transferred to/from Stata to other formats using, inter alia, the commands **infile**, **infix** (ascii data), **insheet** (spreadsheets)**, fdause, fdasave** (SAS transport files).

8 Stata for Windows user interface

Start Stata via the Windows Start button (Programs) or via a Stata shortcut if you have one. The interface you will see is very similar to most Windows programs, with drop-down menus, short-cut buttons, plus various windows—for inputting commands, seeing results (text and graphics), summaries of previous commands, variable lists, etc. Look at the example shown in Figure 1 below. (Stata on other platforms such as Mac or Unix will see a similar picture, and much of what is said below applies there too.)

Differences between what is on your screen and Figure 1 arise because the example is based on my own licensed version of Stata rather than a university network-licensed copy, and also because I have altered the screen appearance by changing the 'user preferences'. Also, I use Stata Special Edition rather than Stata Intercooled.

Figure 1. Example of user-interface to Stata for Windows



All the windows can be re-sized to match your preferences; so too can the font size and type in each window (click on the rectangular button to the left of each window title, and then on 'fonts'). Some windows (such as the graphics or log file windows) do not appear until output is sent to them.

Note 1 shown in the Results window says how much memory has been allocated to data. The Stata default is typically '1000k'. This may be sufficient for several of the exercises in this course, but not in normal use. You can increase the amount by issuing the command **set**

memory Xm, where X is typically a number slightly larger than your data set size but smaller than the RAM memory installed in your PC. You may need first to **clear** or **drop _all** (see **help memory**). (I set the amount of memory available at start-up using a profile.do file – this is run every time I start Stata: see Note 3. More on this command in the Manuals. See also http://www.stata.com/support/faqs/lang/profiledo.html)

At the bottom left of the screen it says "c:\data". This is my default 'current working directory' on this machine. With my personal license, I can set which directory Stata defaults to (e.g. by specifying it on the Properties tab on my Stata shortcut); you may not be able to if you are on a university network (instead Stata may default to your home directory).

I recommend is that you keep all the work for this course in a separate subdirectory. Outside of Stata, create a subdirectory of your home directory called, say, ec968. Then whenever you start Stata, begin the session with the command **cd ec968** (if the default directory is set as your home directory, else type the full pathname after 'cd ').

You leave Stata either by (i) clicking on the window close (X) button in the top-right-hand corner of the Stata main window (see Figure 1), or (ii) by issuing the command **exit** in the Command window. If the active data set has been modified but not saved beforehand, this will produce an error, in which case you need to save the data set, or **exit, clear**.

Stata can be used interactively or by processing collections of commands collected together in "do files". Do files are ascii files prepared using a editor such as Stata's own built-in one **doedit** (accessible from the command line or menu button), or your own favourite one. I recommend PFE (Programmer's File Editor), which is available on the university NT network or downloadable for free over the internet. Do *not* use a word processor (e.g. MS Word).

Interactive use is great for short exploratory sessions. However for proper analysis, use of do files is strongly recommended in order to ensure that your work is reproducible. More about this and audit trails generally, using 'log files' below.

9 Using Stata interactively

Enter a command in the Command window using the keyboard, hit the Return key, and output then appears in the Results window. Type **help** and see what happens (and read what it says!). For some tasks you might alternatively use the menu buttons (e.g. File to **use** a data set), or the Function keys at the top of the keyboard (experiment!), or by clicking on commands issued earlier which are shown in the Review window (one click on the relevant one to put it in the command window; double-click to re-issue the command). If you really do prefer to issue commands via menus (an SPSS for Windows type approach), then you can use the Stata dialog menus. I rarely use them, except for occasional exploratory analysis, especially that using the Stata graphics commands.

While you work interactively, it is desirable to keep a record of your session. See the next section.

The command line in the Command window can be edited using standard Windows methods (cursor, backspace, delete, escape keys, cut and paste etc.).

Various keyboard combinations have special meaning to Stata. Particular useful is Ctrl-Break (hold down the Ctrl key and hit Break at the same time), which tells Stata to stop what it is doing and return control to the keyboard (alternatively, hit the menu button with a white X on red background). Page-Up and Page-Down can also be used to bring previously issued commands back to the Command window. See **help keyboard**.

Try some other commands, e.g. **help st**, **help net**, **findit survival**, **findit pgmhaz8**, **findit cox**

Browse Stata's Help more generally by following the hyperlinks from the Help pull-down menu. Remember: use Ctrl-Break if you want to stop Stata and get control again.

There are several file management commands available from within Stata. Check out the help on commands such as: **cd**, **dir** (synonym **ls**), **erase**, **mkdir**, **copy**, and **shell** (synonym **!**).

Now do the Stata built-in tutorials (they don't require much time). Start by typing **help tutorial** and follow instructions. You could use the 'auto.dta' data set, cited in the tutorials, to experiment with Stata commands.

Many Stata commands (and variable names) can be abbreviated to just a few letters, typically as few as is required to ensure that they are unique. E.g. **summarize** can be shortened to **su**, **list** to **l**, **tabulate** to **ta**, **describe** to **d**, etc.

Stata is case-sensitive. **help language** provides information about the standard syntax of Stata commands. The following is an extract (the material in the square brackets is not mandatory):

```
Language syntax
---------------

With few exceptions, the basic language syntax is

        [prefix :] command [varlist] [=exp] [if] [in] [weight] [using filename] [, options]


    see               language element      description
    ------------------------------------------------------------------------------------
    help prefix       prefix :              prefix command
    help command      command               Stata command
    help varlist      varlist               variable list
    help exp          =exp                  expression
    help if           if                    if exp qualifier
    help in           in                    in range qualifier
    help weight       weight                weight
    help using        using filename        using filename modifier
    help options      options               options
    ------------------------------------------------------------------------------------

In a syntax diagram, square brackets distinguish optional from required options.  Items
presented like this should be typed exactly as they appear in the diagram.  Underlining is
used to indicate the shortest abbreviations where abbreviations are allowed, so that an item
presented like this indicates  that this may be abbreviated to th.  Items presented like this
represent arguments for which you are to substitute variable names, observation numbers, and
the like.

Some options take numeric lists as arguments. See help numlist for details on
various ways of specifying these numeric lists.

Some commands also have an immediate form (allow you to enter numbers directly
instead of entering variable names). See help immed for details.

Programmers interested in incorporating Stata's language features into their
Stata programs should see help syntax for the syntax command.
```

For general information and rules about other Stata syntax issues such as filenames and variable names, see the *User's Guide*.

10 Audit trails: keeping records of your Stata sessions using log files

Even when working interactively, it is important to keep a copy of your session in some form. In particular, you may want to repeat part of a session again in future (to cut out material not required or incorrect). And if you are doing analysis for a paper rather than simply exploration, you need a permanent copy of your results and how you derived them. (Good scientific practice also requires you to be able to reproduce your results – a practice facilitated by using do files in conjunction with log files. See more on this below.)

Stata uses 'log files' for recording session output. See **help log**. There are two types of log file, one for results created using the **log** command, and the other for storing commands that you have typed, and is created using the **cmdlog** command. Either or both or neither may be used in a session. Command logs are plain ascii text files. For results logs, there are two formats that you may choose from: (a) plain ascii text, with filename suffix 'log', and (b) Stata Markup Control Language format (SMCL, pronounced 'smickel'), with filename suffix 'smcl'. SMCL files preserve fonts and colour. You can always switch between formats using the **translate** command.

You can open a results log file with **log using filename [, append replace]**. The default format if no file extension is used is SMCL; use extension 'log' if you wish to have ascii

format (e.g. **log using filename.log, replace**) You can also use the fourth menu button from left – see Figure 1. The [.] in the command refer to command options (the '[' and ']' are not typed!): 'replace' means that file filename is over-written if it already exists, 'append' means append the output to an existing file. **log close** closes the file. **log off** and **log on** temporarily switch recording off and on. If you have a log file open, you can view the contents as you go along in the Log window (via Window menu button). Command logs have a similar format: **cmdlog using filename [, append replace]**.

When I am working interactively, I typically issue the command **log using junk.log, replace** at the very start of a session. ('junk' since I then know it can be over-written in future.) This is insurance – I've learnt the hard way that I sometimes needed a session record but then didn't have one! Of course, you could also scroll back up the Results window to look at earlier output, but the buffer contents of the window are limited. A log file is not limited in the same way.

It is useful to be able to annotate output as you go along, for future reference. Stata automatically puts a date- and time-stamp in the log file. You can insert comments in several ways. Lines beginning '*' are simply echoed to the output device. Material after a '//' anywhere in a command line is ignored. All material entered between '/*' and '*/' is ignored: This is particularly useful in do files for (a) entering long multi-line comments. '///' at the end of a command is a line-continuation marker. Stata jumps from there to the next line and so this provides an easy way of breaking long commands over several lines. Examples appear in do files with the exercises. Breaking commands over multiple lines is not essential, but is useful since PCs typically display things best when restricted to width of 80 columns. For an alternative way of breaking command lines up in do files, see **help delimit**.

11 Printing and other processing of Stata output (text and graphics)

If you have created a log file, then you can simply print it! You might want to edit it first, in which case take the ascii version and simply read it into your favourite editor (e.g. PFE or Notepad) and print it from there. (You could also use Word, but ensure you use a fixed pitch font such as Courier New – not a proportional font such as Arial or Times Roman – or else table columns will not be properly aligned.) Alternatively, with a log file open, you can print the log directly from within Stata via the File menu.

Sometimes you may want to include selected parts of your output directly into another document for further processing, e.g. a table of summary statistics or regression output. There are several ways of expediting this:

(1) Check out the Stata Copy Text and Copy Table commands, found under the Edit menu button. Copy Table, for example, can be used to copy the output from the Review window into an Excel spreadsheet, process it further there (e.g. rounding numbers, deleting extraneous columns), and then copy and paste into Word. Alternatively, you can copy and paste straight into a Word Table and then edit that.
(2) The latter step can, of course, also be done directly from your log file.
(3) You can use the **estimates table** command to format regression-like output in a powerful way.
(4) Several users have written Stata programs to run directly after estimation commands in order to convert the Stata formatted output into tabular formats corresponding to those in

academic papers and reports. The best of these, in my opinion, are the **estout** command by Ben Jann for formatting estimation results and **tabout** by Ian Watson for formatting crosstabulations. The latest version of each of them is available from the SSC-IDEAS software archive at Boston College (op. cit.): type **ssc install estout** and **ssc install tabout**. Both programs can produce either (i) tab-delimited output for pasting directly into Word or Excel, (ii) html output for web pages, or (iii) text output marked up in T_EX. See also **fsum** by Fred Wolfe for summary statistics.

All the discussion above refers to alphanumeric text output, but not graphics. If you are working interactively and have the relevant graph in an open Graphics window, then you can simply print it via the File menu. (You may wish to first change the graph preferences, e.g. increase line thicknesses, or remove Stata logo: do this via the Preferences submenu under the Edit menu.) Alternatively, you might want to copy and paste the graph directly into a Word document, in which case use the Edit menu (Copy graph) to copy, and then paste as usual. Ensure that you have set all the graph preferences as you would like them, under all three of the Graph Window, Printer, Clipboard tabs in the Edit/Preferences menu. See also **help printing,** and **translate**.

If the graph is created within a do file, then the 'saving(filename)' option on the **graph** command is required in order to produce a permanent copy in a file on disk. Then to replay the graph in a subsequent session, type **graph use filename**. Then proceed as above. See **help graph** for further information about the graph options and the graph command itself.

12 Using do files

Stata command **do filename** makes Stata execute a sequence of commands stored in a file called filename.do as though you had entered the commands in the file sequentially from the keyboard. If no file extension is specified, Stata assumes that it is '.do'. See **help do**. **do filename** echoes the commands to the screen (and log file if open) as it executes them. **run filename** runs the file, but without echoing output. do files are an essential part of Stata use.

do files are plain text (ascii) files and are most easily created using either your favourite editor or Stata's built-in editor **doedit**. (Word might also be used, but as it is a word-processor rather than a specialist editor, you always have to ensure you save the file as an ascii file, and note too the earlier comments about fonts.)

My personal recommendation for an editor is PFE (freeware), cited above. (Notepad is another alternative, amongst many.) A notable feature of Stata's built-in editor **doedit** is that it is easy to run and re-run a selection of commands (rather than the complete do file): see the Tools menu. For this reason we use it in this course. You can open the Stata editor by clicking on the menu button fifth from right in the main Stata window (see Figure 1).

My do files have several common elements which always appear in them:
(1) commands opening a new log file at the start, and closing it at end;
(2) a version statement (because some things are version-specific. Stata is backward compatible, if told to be using **version**);
(3) comment lines giving a brief indication of what the function of the do file is.

I usually also

(4) **set more off** to stop Stata pausing at the end of each 'page' and asking for 'more' (see **help more**), so that the job runs until completion without further intervention (if error-free!); and
(5) add a **clear** command at the top of the do file.

Exercise 1.1 in Lesson 1 provides an illustrative example.

My mode of working on a project typically involves the following steps:

(1) Interactive 'explorations' on a data set in Stata, using a temporary log file (as above) and maybe capturing commands using a command log.
(2) Creation of a first analysis do file using PFE or Stata's **doedit** (perhaps via editing of junk.log) – call it an1.do – which in turn creates a log file called an1.log, say. (an1.do also contains the other command elements (i)-(iv) cited above.) The do and log files are both saved to the current working directory.
(3) Return to Stata (leaving the PFE window open), ensuring Stata is in the current working directory (otherwise use the **cd** command first to change directories).
(4) Type **do an1**
(5) Inspect an1.log using your editor.
(6) Modify an1.do, correcting errors and adding commands, as required; return to step 4.
(7) If the job now works to my satisfaction, stop. If not, return to step 6.

You may of course find a different way of working which suits you better. The particular method (and editor used) does not matter. What does matter is the audit trail – the ability to reproduce results, to remember why you did them (hence comments), and when (date and time stamping). This becomes particularly important once you begin to accumulate many do files (easy to do in the course of a project). Judicious use of separate working (sub)directories for different projects is also advisable.

13 Basic Stata tasks

The Lessons do not provide an introduction to learning Stata (most learning comes from doing!) Instead use the Stata tutorials (**help tutorial**) or the tutorials available on the web that were cited earlier. Lesson 1 provides some exercises too.

The main tasks with which you will need to become familiar with for this course are listed below, together with (some of the) Stata commands relevant to these tasks. Stata **help** is, of course, available on each of the commands.

Searching for help and programs (official and user-written)
help, **findit, ssc, search**
Utilities
log, **cd**, **dir**, **erase**, **mkdir**, **copy**, **display**, **delimit**
Reading data in, and saving it in Stata format
use, **insheet**, **infile**, **compress**, **save**
Setting memory size

memory, **clear**, **set memory**, **set matsize**

Inspecting, summarising and describing data

describe, **list**, **inspect**, **summarize**, **tabulate**, **correlate**, **sort** and **bysort**

Creating new variables and re-organising data

generate, **replace**, **egen**, **recode**, **label**, **keep**, **drop**, **rename**, **expand**, **by**

[see also **help functions**, **help exp**, **help operators** about functions and expressions]

Statistical analysis, estimation and graphics

regress, **logit**, **probit**, **tobit**, **cnreg**

predict, **test**, **testnl**, **lrtest**, **lincom** [also see **help est** about retrieving estimates]

ltable, **st**

stset, **stdes**, **streg**, **stcox**, **sts list**, **sts generate**, **sts graph**, **sts sts**, **stsplit**, **stgen**

xtlogit, xtclog

Graphics

graph

[NB standard graph options (for labelling etc) are typically applicable to graphs produced by other commands]

NB By default, Stata stores missing values on a variable as infinitely large values and refers to them with the symbol '.' (There are also additional optional missing value codes .a, .b, .c, …, that are each treated as being larger in magnitude than '.') You need to be careful in the treatment of missing values when using logical expressions to avoid unintended effects. E.g. **replace y = x if z > 1 & z < .**, or **replace y = x if z > 1 & !missing(z)** leaves y unchanged if z is missing. (**!missing(z)** refers to the missing(*varlist*) logical function which evaluates to 1 if any of *varlist* is missing and 0 otherwise; the '!' is the logical 'not' operator.) Compare these statements with **replace y = x if z > 1** in which case y is replaced with x if z is missing (probably an unintended result). See ex1.2.do (Lesson 1) for an illustration.

14 Stata data sets for this course

All the data sets have been chosen to be small, so that results can be derived very quickly and to minimise potential hardware constraints (memory etc). Also, there are no missing values in the data. And the data already contain variables summarising survival times and censoring status, and thus do not have to be created. Be aware that much time spent in real-life research is spent sorting out the data rather than just estimation!

We assume throughout the course that our data sets contain a random sample of spells, with one spell per subject (and there is no left-censoring). The modifications to estimation methods which are required when this is not the case, e.g. because of left truncation, are not considered here. (There is some discussion in the course lectures.)

The data sets cited below that are provided along with Stata itself (auto.dta, cancer.dta, kva.dta) can be found in your Stata system directory. To find out what this is use **sysdir**: the files are in the directory labelled 'STATA:'. You can simply **use** the data from there using the full path name or, even simpler, use the **sysuse** command (in which case you don't have to know the host directory). Alternatively, simply download copies from the course website along with the other files. You will find it useful to place copies of all the files in your current working directory (called m:\ec968 or something similar).

To read Stata data sets into memory, you use the **use** command (**help use**). See the tutorials for more about this. If you have data which is not in Stata format there are a range of commands which may help you. E.g. **help infile** and **help infix**. Commands such as **insheet** allow you to read in data created in a spreadsheet, and **fdause**, **fdasave** read and write SAS Transport file format data. For data which is held in another file format, you can either use these programs to write files which can then be read into Stata using **infile**, or else you can use a general file conversion program such as Stat/Transfer (http://www.stattransfer.com) which automates these tasks.

auto.dta(1978 Automobile data)

Test data supplied with Stata, and used extensively in their tutorials. Originally came from Consumer Reports, April 1979, and from the United States Government EPA statistics on fuel consumption. It was compiled and published in Graphical Methods for Data Analysis, The Wadsworth Statistics/Probability Series, 1983. 74 observations, 12 variables.

kva.dta (Generator failure time data)

Test data supplied with Stata (versions before version 9). Contains failure time data for a fictional experiment with generators. 12 observations, 3 variables.

cancer.dta (Cancer data)

Test data supplied with Stata, which we use extensively in this course. These contain fictional data on a drug trial. 48 observations, 4 variables. The four variables are:

studytim: survival time in months
died: censoring status (=1 if died, 0 if censored)
age: patient's age in years
drug: = 2 or 3 if received a test drug, = 1 if received the placebo.

kennan.dta (Strike data)

Test data reported in the LIMDEP manual, originally derived from work by John Kennan. Contains data on strike durations. 62 observations, 2 variables:

time: strike length (survival time)
status: censoring status: status (=1 if event, 0 if censored)

duration.dta (Marriage duration data)

Test data reported in the LIMDEP manual. Contains fictional data on partnership durations. 22 observations, 4 variables:

time: partnership length (survival time)
status: censoring status: status (=1 if event, 0 if censored)
sex: sex of person (1 = male, 0 = female)
married: legal status of partnership (1 = cohabiting union, 2 = legally married)

bc.dta (Breast cancer data)

Test data used in Stata 7 Reference Manual (Volume 3 Q-St, p. 361), supplied courtesy of R. Guitierrez of StataCorp. Fictional data on 80 women with breast cancer, 5 variables.

t: survival time in study
dead: censoring status: status (=1 if event, 0 if censored)
age: age in years
smoking: 1 if smoker, 0 otherwise
dietfat: average weekly calorific intake (x 1000) in patient's diet over course of study

dropout.dta (College dropout data)

Test data about time-to-dropout for a sample of 254 college entrants, copied from Yamaguchi (1991, Table 6.6), with 9 variables.

obs: Id # for observation
dur: Time until school dropout (#months since entry)
evt: event/censoring status (1=drop-out,0=censored)
sex: sex (0=male,1=female)
grd: High-school grades (self reported)
prt: 1=part-time student,0=full-time
lag: time lag between high school graduation and college entry (months)
mrg: time of marriage (#months since jan1980, except that 99=never married)
stm: time of college entry (#months since jan1980)

hmohiv.dta (HIV survival data)

Data from a hypothetical HMO-HIV+ study, and shown in Table 1.1 of Hosmer and Lemeshow (1998), with 100 study participants, 9 variables.

id: subject ID code
entdate: entry date (ddmmyr, string variable)
enddate: end date (ddmmyr, string variable)
time: survival time: enddate-entdate (months)
age: age (years)
drug: whether has history of IV drug use (1 = yes, 0 = no)
censor: follow-up status (0 = alive at study end or lost to follow-up, 1 = death due to AIDS or AIDS-related factor).

unemp.dta (Unemployment duration data)

A sample of Spanish men aged 18-54 years who started a spell of Unemployment Insurance in February 1987. They were followed until they exited from UI or exhausted their entitlement. (This sample is drawn from real-life administrative data: for further details, see Jenkins and Garcia-Serrano (2004). 1507 observations on 12 variables:

age: age, in years
conmths: Spell length, in months (survival time)
famresp: Whether has family responsibilities or not (0 = no, 1 = yes)
groupreg: Region of residence (1 = North, 2 = Centre, 3 = North-East, 4 = South, 5 = Islands)
potmths: Max # UI months eligible for, in months

rn1: net replacement rate months 1-6 of spell
rn2: net replacement rate months 7-12 of spell
rn3: net replacement rate months 13-24 of spell, where the net replacement rate is the ratio of income received if not working to the income received if working (net of tax).
tyentry: Type contract in job prior to UI spell (0 = permanent, 1 = temporary)
exit: UI spell ended? (Censoring status: 0 = censored, 1 = exit)
status: general exit status variable (0 = exhausted UI-no UA, 1 = exhausted UI-then UA, 2 = exit UI to a job, 3 = exit UI to other states). NB variable exit = 1 if status = 2 or 3.
newid: Case identifier

NB rn1, rn2, rn3 can be used to create a time-varying covariate summarising the net replacement rate in each spell month at risk of exit.

15 Learning Stata basics

The tasks and exercises are to help you become familiar with what Stata is and what it can do, and to provide practice with some of the basic tasks of data and file management.

1. Read the material about Stata resources (in particular where to get help!), the Stata interface, and using Stata;
2. Browse the UCLA Academic Technology Services site http://www.ats.ucla.edu/stat/stata/. It has everything from learning how to use Stata to worked exercises from leading texts:
3. Download all the files from the course materials website into a subdirectory of your home directory. (I recommend calling the subdirectory ec968).
4. Do Exercises 1.1 and 1.2 on Stata basics.

16 Exercise 1.1

Try the following commands (excluding comments) interactively and then package them into a do file and run them (including the comment lines). You could also **do ex1_1**.

```
version 10
capture log close
        /* This closes any open log files.
           These can be left open if a do file stops with an error before
           its completion. Prefacing a command with 'capture' means
           that an open log file is closed; if there were no log file open, the
           usual error message gets swallowed. See -help capture- */
clear // clears all data, matrices etc from memory
set more off    see -help more-
log using ex1_1.log, replace
* this is a test job
cd  //-cd- by itself returns the name of the current working directory
use cancer
describe
summarize
* add any other commands you like below here
log close
```

17 Exercise 1.2

Use the marriage duration data set, duration.dta, to do the following tasks. You might first experiment interactively, and then write your own do file. Later you could **do ex1_2** to see how I answered the exercise.

1. What are the variable names, and how many observations are there?
2. What are the value labels for the variables? [Hint: **help label**, look at 'label list']
3. How many men are there in the data? (# and % of all persons)
4. How many women are married? (# and % of all women)
5. What does the observed distribution of marriage durations look like? e.g. min, max, median, mean; general shape etc
6. How does mean observed marriage duration differ between persons with censored and uncensored spells?

Let's create a new (fictional) variable 'age'. For each person, we'll set it equal to their observation number (for observations sorted in ascending order by 'time') plus 30. Also, for the sake of practice, we'll assume that any ages greater than 50 are 'missing'.

```
sort time
ge age = _n + 30
replace age = . if age > 50
```

7. How many cases are missing a value for age?
8. What is the mean, median, minimum and maximum of age?
9. Create a new age variable with values equal to the square of each person's age.
10. Create a new age variable with values equal to the natural logarithm of each person's age.
11. Create a new categorical variable summarising age in 5-year bands (30–34, 35–39, 40–44, 45–49, 50+). Give the new variable a useful variable name and label its values. Hint: **help label**. NB there are several ways of creating the variable. Hint: methods include using **generate** and **replace** commands; **generate** command with its **recode** function; **recode** command. Beware of how you treat the missing values – ensure that the categorical age variable is missing whenever age is missing.
12. Create a set of dummy variables based on the categorical variable derived in Q11. NB there are several ways of doing this. You need to decide how to treat the missing values. Should they equal zero for each of the dummies? Or be equal to missing for all the dummies? Use the second convention in this exercise. Hint: use **generate** and logical expressions, or the **tabulate** command (use **help tabulate** and check out the generate(.) option).
13. **describe** your data set, then **compress** it, and finally **describe** it again. Look at what happened to the size of the file in Kb, and the storage types of variables.
14. Now drop the log(age) variable from Q11 from the data set, and save the data in a file called 'mydurat.dta'.