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 17. (Most of the programs in the Exercises below can be run using version 6 onwards, but I cannot guarantee this.)

1. **Course resources**

* Slides
* STATA do files
* STATA data sets
* Lecture notes

All downloadable from <https://github.com/alessandrodinallo/EHA>

1. **Lessons**
2. Preliminaries – Introduction to Lessons and STATA (this document)
3. The shapes of hazard and survival functions
4. Preparing survival time data for analysis and estimation
5. Estimation of the empirical (KM) hazard and survivor functions
6. Estimation: (i) continuous time models – parametric
7. Estimation: (ii) discrete time models
8. Unobserved heterogeneity (‘frailty’)
9. Competing risks
10. **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!

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

1. **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).

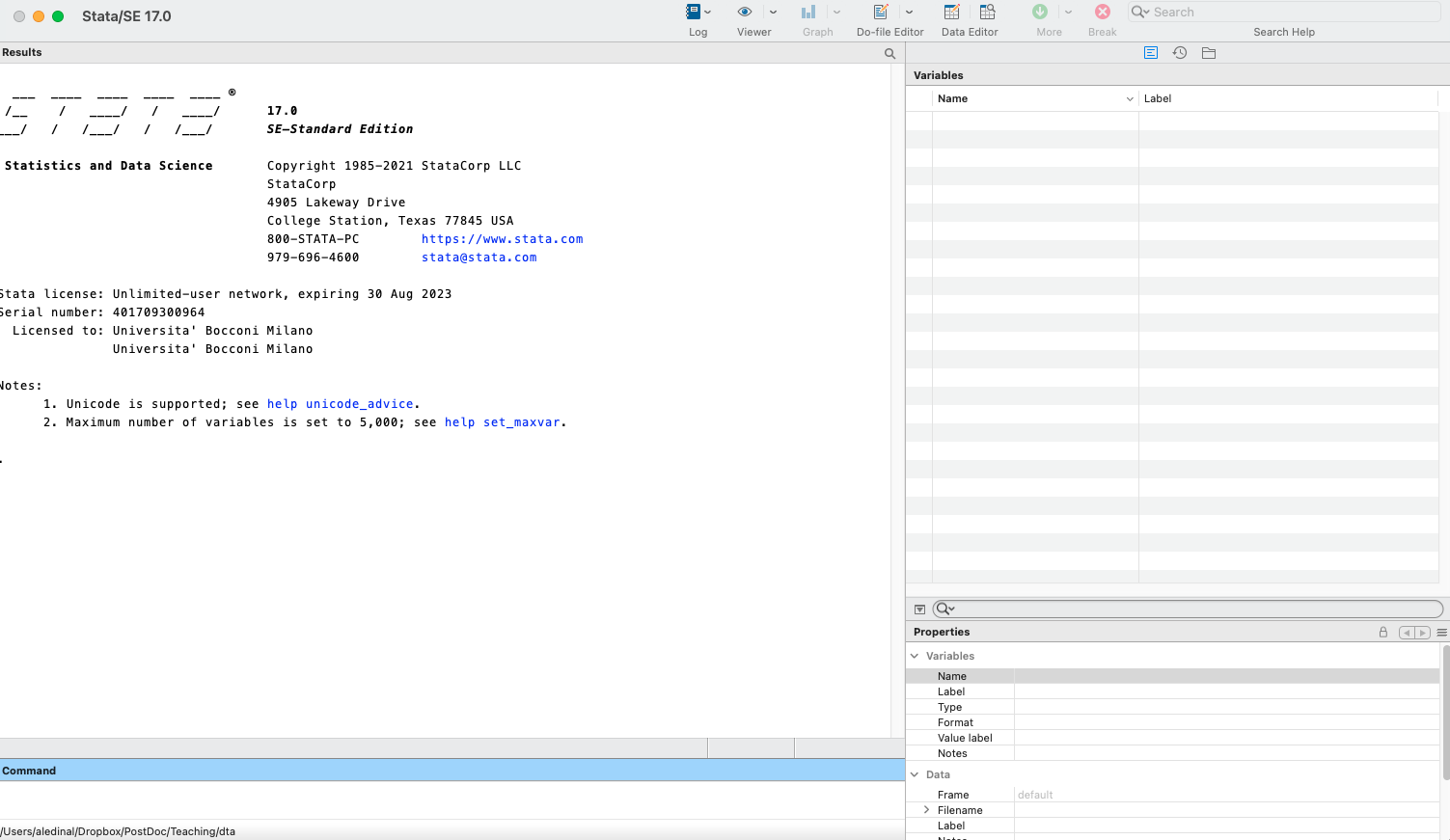
STATA is easy to use, but is also very powerful. It has a large range of built-in tools, but is also programmable. STATA programs and data sets are platform-independent. All the tools required for this course (and for much applied analysis of any kind) are provided in Stata. There is also a large user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

1. **STATA for Windows user interface**

The interface you will see is very similar to most 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 Windows 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’.

Figure 1. **Example of user-interface to STATA for Mac-OS**



1. **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.

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

1. **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 TEX. 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.

1. **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.

1. **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.do (Lesson 1) for an illustration.

1. **STATA datasets 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.

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. Contains data on strike durations. 62 observations, 2 variables:

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

**bc.dta** (Breast cancer data)

Test data used in STATA 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, 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, 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. 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

**sep.dta** (union dissolution dataset)

A sample of couples’ histories from BHPS and UKHLS (1991 – 2021), containing information on partners’ background (ethnicity, parents’ class etc.) and yearly-changing variables (education, job status etc…)