Introduction to STATA with Econometrics in Mind

John C. Frain

TEP Working Paper No. 0210

February 2010

Introduction to Stata with Econometrics in Mind

John C. Frain.

February 2010

Abstract

This paper is an introduction to Stata with econometrics in mind. One aim of the proposed methodology is the keeping of appropriate records so that results can be easily replicated. These records should meet the requirements of management and internal audit functions in policy making bodies and be sufficient for submission to journals that require such material.

The paper describes the Stata desktop, shows how to organise an analysis, how to read and transform data and covers the OLS regression command in detail. It includes details of various post-estimation commands, specification tests, model verification procedures, calculation of elasticities and other marginal effects, forecasting and the use of various statistics used by Stata during the estimation procedure. As all estimation commands in Stata share a common structure the detailed study of the OLS command will assist in the use of other commands.

Contents

1 Introduction

1.1 Before you begin

Stata is a computer package for the statistical analysis of data. It is an almost comprehensive collection of tools and is used for data analysis in many diverse fields as well as in econometrics. Stata can do much of the statistical/quantitative analysis required in econometrics – a lot with ease, some with a bit of work and some with considerable difficulty. It has always been particularly strong in microeconometrics and recent versions have extended facilities for time series analysis.

Stata is a very large package. The manuals are almost 9000 pages in length. Even the most experienced users have an in depth knowledge of only a small part of the package and need frequent access to on-line help files and manuals. Unless you know your way around the help files and manuals you will never master Stata. If you understand the general principles of how Stata works and can find your way around the help files and manuals you will find working with Stata much easier.

The aim of this note is to introduce new users to Stata. No prior knowledge of any econometric packages is presumed. You are expected to have some knowledge of Word processing, Spreadsheets and basic file handling on a PC with a Microsoft Windows operating system. You will learn how to use Stata to

- read and write data files, manipulate data and draw graphs,
- estimate an equation using OLS,
- complete various tests of hypotheses, specification tests, verification of models, etc.
- interpret the results of your estimated model (e.g. calculate elasticities, forecasts),

Other estimation commands in Stata may have their own special requirements but they do share common details in syntax and test methods with the OLS command. A good knowledge of the OLS command will provide a good indication of how one might proceed with the other estimation commands. For specific details, you will, of course, need to access the on-line help files and manuals. The contents list at the start of the note gives more details of what is covered here. If you are reading the pdf version of this note on a computer you will be able to use the the various hyperlinks to navigate between various sections of the note. There is a summary list of Stata estimation commands on pages 61 to 66.

Finally a warning. The proper use of Stata requires a good knowledge of economics econometric theory. A lack of this knowledge will generally lead to error. Proper empirical work needs a good knowledge of both the econometrics involved and the software used.

All practising econometricians should have an understanding of what might be called econometric philosophy or methodology. How does one include what Magnus and Morgan (1999) refer to as *tacit* knowledge? There are several approaches to these questions. How does one undertake a specification search? Why is general to specific modelling preferred? Depending on the approach and the *tacit* knowledge employed there may be several different valid results derived from the same data. These matters are not covered here. The interested party might consult Stigum (1990), Granger (1999), Magnus and Morgan (1999), Keuzenkamp (2000) or Hendry (2000).

This note is based on Stata version 11. If you are already familiar with Stata you will find descriptions of new Stata facilities of interest for econometrics in

- subsection 9.3 (page 29) on factor (categorical) variables,
- subsection 11.2 (page 36) on including factor variables in regressions
- subsection 12.6 (page 53) on the new `margins` command which replaces the old `mfx` command. The `mfx` command should not be used in Stata 11 (unless estimation commands are constrained under version control to be compatible with an earlier version of Stata).

Much of the note can be read fairly quickly, taking note of relevant points and marking others for later reference. There is a lot of computer output which makes the note appear longer.

The best way to use the examples is sitting beside a PC, reading the commentary, running Stata, entering the commands and checking that the output corresponds to the output given here.[2] The data sets used are available on my web page.[3].

For a beginner perhaps the most confusing thing about Stata is that there are many correct ways to complete one task. Some of these make Stata easier for the beginner while others are aimed at the more practised user.

[2]Some of the Stata output has been edited to make it more readable

[3]currently`www.tcd.ie/Economics/staff/frainj/home.htm`

Some of you may have already completed an introductory econometrics course and may have used some econometric package to do OLS regressions. It is likely that you were more interested in the final result than in using a method that ensured that your analysis could be easily replicated. You, probably, started the program and produced results using a series of mouse clicks selecting various choices from menus. This is a relatively easy way to do simple econometrics. It does add to the understanding of econometrics. However in real applications one may be faced with a series of data transformations, various options in a more complicated estimation routine and a complex series of model verification and forecasting procedures which you may need to replicate or change slightly at a later stage. In such cases one must keep a record of your original instructions. Stata has good facilities for this kind of work. Instructions can be entered as commands and the commands can be automatically recorded and reused. This is the basis of the methodology described in this note. Commands can also be given through the drop-down menus (or using dialogues in the help files. The methodology described here commands issued to Stata in this way are also recorded and may be reused in the same way as commands entered directly.

There are other reasons why it is important to record your work in this way.

1. If you are submitting an empirical article to a journal some journals insist that you supply the programs and data that you use. Otherwise they will not publish.
2. If you are working for a Financial Institution, Central Bank, Department of Finance or other body involved in policy analysis, management and the internal audit function will probably have agreed procedures that insist on a similar practise.
3. Even if there are no such rules this is good practise and will certainly make things easier in the long run.

1.2 Entering Stata Commands

There are at least four separate ways of entering commands in Stata

1. Enter the commands from the keyboard exactly as set out.
2. Almost all commands can be "entered" using the standard menus and the dialogues under these menus. This method is useful if you are not certain of the exact syntax of a Stata command
3. Almost all commands can be entered using the dialogues offered by the Stata help files. This method is similar to using the Stata menus
4. A list of commands may be entered on a text file and submitted to Stata. Stata can save, in a text file, the commands entered using any of the methods (1) to (3) for possible amendment and later use. This is the way we will use Stata. In this way we can build up the Stata program (do-file) which is the basic record of how we achieved our results[4].

Any Stata command, keyword or option can be abbreviated provided that abbreviation is unique. For example, the Stata command, `summarize`, can be abbreviated to `su`. While this facility may be useful ,on rare occasions, I do not recommend it, for general use, as it can make Stata programs and scripts very difficult to read.

[4]a do-file is a file containing a sequence of Stata instructions. The management of do-files is covered later in section 8 (page 23)

Many of the examples used in this note are used solely to illustrate the use of Stata commands or to give an example of Stata output. Econometrics is to a great extent the slave of economics and all econometric estimation and tests should have economic meaning. Many of my examples do not have economic content but are used to explain what could be concluded if they had.

1.3 Other Resources

Stata Press have recently published two books on using Stata for econometrics (Baum (2006)) and Cameron and Trivedi (2009)). Either is recommended to anyone who wishes to go beyond these notes. If you are interested in microeconometrics I would recommend Cameron and Trivedi (2009). For advanced microeconometrics the Stata (version 8) examples on the website[5] of the book Cameron and Trivedi (2005) are useful. Many of these examples are also given in Cameron and Trivedi (2009).

Adkins and Hill (2008) is a Stata guide to the undergraduate econometrics textbook Hill et al. (2008) which some may find useful. Stata programs and data for many of the examples in Wooldridge (2002) are available on that book's website.[6] Baltagi (2009a) contains Stata code for the examples in Baltagi (2009b) and data are available on the books web site. The web site `http://www.ats.ucla.edu/stat/Stata` also contains a lot of introductory material on Stata.

2 Organising your work

The recommended set up may appear a bit complicated at first sight. Investing time in organising your work properly may save you a lot of time later.

Set up each project in a separate directory[7] (e.g. `s:\housing` might have the files relevant to a project on housing). Store all data, program listings, output files and logs etc... relevant to each project in the separate directory that you have created for the project. If you are working on a public computer you will create the project directory on you networked `s:\directory`. It can then be accessed from an office PC or a public PC. Remember to maintain a backup of this directory.[8] The files you will keep in this directory will include.

Short cut to the Stata executable with its properties amended so that Stata starts, looks for input files and places output in this directory.

[5]`http://cameron.econ.ucdavis.edu/mmabook/mma.html`

[6]`https://www.msu.edu/~ec/faculty/wooldridge/book2.htm`

[7]The default set-up in MS Windows does not show extensions for file types that are known to Windows. If you know about this and are comfortable with it you may ignore this footnote. To me and to many other people this is a source of confusion. They may have several files say `housing.xls, housing.csv, housing.dta, housing.doc` etc... and they may all appear as `housing`. Worse still you may have a file `housing.do` that windows has saved as `housing.do.txt` and refuses to display the `.txt` extension. The system allows virus files to hide as innocent text files. To show the full name of these files extensions start Windows Explorer, select [Tools - Folder Options] and select the view tab. Make sure that the box "Hide file extensions for known file types" is not ticked. Click Apply and the click "Like current folder" to reset all folders. Windows will then show all file extensions in Windows Explorer and in the various file dialogues. This problem is a constant cause of confusion and lost files.

[8]Flash memory sticks are easy to use for backup. I use two backing up to alternate sticks every second day. Memory sticks are small and even if one gets lost at the wrong time you will only lose one day's work. Copying the material to your laptop or home desktop is an additional safeguard.

`profile.do` This file contains a sequence of Stata instructions which run automatically every time Stata is started from that directory. This is used to make Stata keep a record of the instructions we give and set up log files. There is a sample `profile.do` file on the class web-site. This sample file is described on page 7. If you use the methodology described here you must have a file similar to this in your working directory.

data files Keep a copy of your original data file(s) in original format. If you transform the data or add to the data you should save the extended data set on a new file. We will learn how to save the transformations in a file. In this way you will be able to easily reconstruct your analysis from the beginning if necessary.

`.do` files In Stata lists of commands are saved in `.do` files.

log files Log files record the input and output of a Stata session. I recommend the use of two kinds of log files

- A log of all work completed during the current session (commands and output). This file is called `stata.log` if you use the `profile.do` file recommended here.
- A separate log containing a cumulative list of all commands issued to Stata relevant to a particular project This file is called cmdlog.txt if you use my `profile.do`

reports Here (or perhaps in a subdirectory of your project directory) you might keep you project reports and other submissions.

If you follow these procedures the final `.do` and log files will provide part of your final paper or submission.

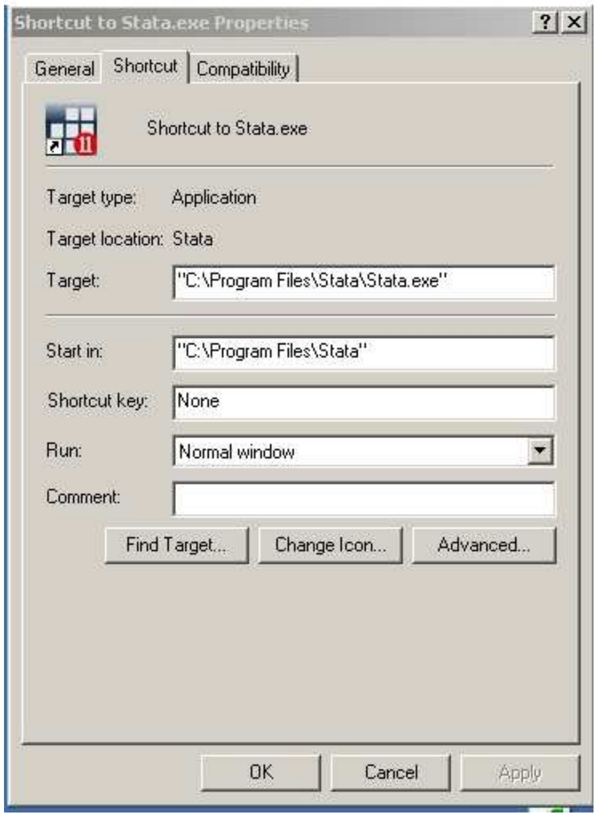
3 How to start Stata.

Stata can be started from a Start menu on your desktop (Details depend on the particular configuration on your PC or network). However it is better to start Stata from a properly configured short cut in your project directory. Then Stata will know where to find (read) and put (write) files. This procedure may look complicated but you will find that it makes work easier. Proceed as follows

Set up a short cut to Stata in the project directory If you have not already created a Stata short cut open Explorer and go to the directory containing the Stata .exe file. Use the right mouse button to drag it to your project directory and select "Create short cut here" from the context menu that appears.[9] If you have already created a Stata short cut in an earlier project directory copy that short cut to your new project directory.

Specify that Stata start in the project directory Within Windows Explorer right click on the short cut and select properties. A window similar to that in figure 1 should appear. Change the entry in the "Start in" box to refer to your new project directory (e.g. For this exercise my project directory is `C:\TCD\tutorials\session1`). When you start Stata it will then automatically look for and save data files etc... in this subdirectory. You can change this default subdirectory or to another directory from within Stata when Stata is running (see below)

[9] If there is already a short cut to the .exe file you may copy that short cut to your working directory


Shortcut to Stata.exe Properties

General | Shortcut | Compatibility

Shortcut to Stata.exe

Target type: Application

Target location: Stata

Target: "C:\Program Files\Stata\Stata.exe"

Start in: "C:\Program Files\Stata"

Shortcut key: None

Run: Normal window

Comment:

Figure 1: Stata Shortcut Properties Window

Extend data memory available to Stata Depending on the version of Stata that you are using Stata may allocate insufficient memory for your data. For example, the default on my laptop is 10 *MB*. This is not sufficient for many panel data sets. One way to allocate additional memory is to add `/m25` to the target entry (on my laptop `C:\Program Files\Stata10\wstata.exe /m25`) but the directory will be different on your PCs. When you start Stata from the desktop short cut 25 *MB* is allocated to data storage. If you add `/m64` it allocates 64 *MB*. You may also allocate the extra memory after starting Stata but before loading any data or in the `profile.do file` or expand memory availability within Stata before loading data.

Put a start up file in your project directory If Stata finds a file named profile.do in the start up directory it will run the Stata commands in that start up file every time Stata is started. My recommended start up file is as follows.

```
//profile.do
set obs 1
gen time="******* Session started: ‘c(current_date)’ ‘c(current_time)’"
outfile time using .\time.txt, noquote replace
clear
! copy /b .\cmdlog.txt + .\time.txt .\cmdlog.txt
```

```
cmdlog using .\cmdlog.txt, append
set logtype text
log using .\stata.log, replace
```

A copy of this file is on the web site with this document. The file is simpler than it looks. If you do not understand it now, don't worry as its contents will become clearer later. Simply put a copy in each project directory. When you know more about Stata you may wish to use an amended version. It contains two parts

1. The first seven lines deal with a file which is called `cmdlog.txt` When you start Stata using the method described here these lines enter the time and date of the start of your Stata session at the end of the file. As you issue instructions to Stata these are saved at the end of the file. Thus the file will contain a list of all instructions issued to Stata for the project in question. You can cut and past from this file to create a `.do` files. This file is **not** suitable for submission as a program listing.
2. The final two lines record all the input and output from the current session. With the appropriate `.do` file as input this file will produce the program output required in a submission. As set up here the contents of the old log will be deleted at the start of a session and a new log started. If you wish to retain the results of a session you must change the name of the old log file before starting the Stata session.

You may find it useful to add further instructions to this file.

4 Stata Workspace and Stata Widows

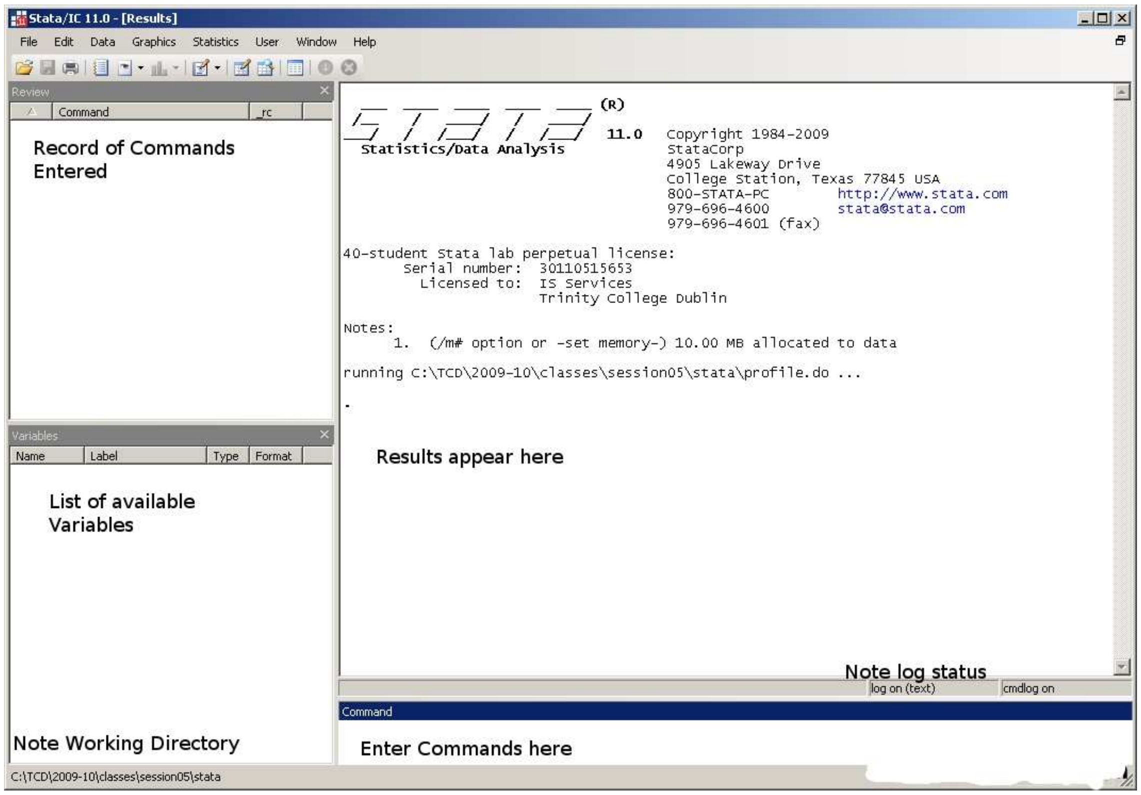
Figure 2 shows the basic Stata workspace when the program is started. First note that the current working directory is given at the bottom left hand corner of the work space.

There are eight Stata windows of which four are visible at start up –

1. **COMMAND Window.** In this window one enters single commands and executes them.
2. **RESULTS Window.** The RESULTS window is immediately above the Command Window. As might be expected this window displays results. It can not be edited and is of limited size (early results in the session may be deleted as new results are generated. Cut and past or print your log file rather than the results window. Note that the status of the Stata logs is shown at the bottom right of this window[10]
3. **Command History Window.** This window shows a history of commands issued during the current session. These commands may be transferred from the **History** window to the **Command** window by a mouse click, edited if necessary and re-submitted to Stata.
4. **VARIABLES Window** List of variables in memory. Their names may be transferred to the **Command** window or to some dialogues by double clicking.

There are four other windows that will appear as necessary –

[10]If when you start Stata the wrong working directory is displayed in the bottom hand corner of the Stata workspace or the status of the log files is not " **log on (text) cmdlog on**" you have not set up Stata as required. Please read section 3 (6) again.


Stata/IC 11.0 - [Results]
File Edit Data Graphics Statistics User Window Help
Review
Command _rc
Record of Commands Entered
Variables
Name Label Type Format
List of available Variables
Note Working Directory
Results appear here
Note log status
log on (text) cmdlog on
Command
Enter Commands here
C:\TCD\2009-10\classes\session05\stata

```
                                          (R)
  Statistics/Data Analysis            11.0   Copyright 1984-2009
                                             StataCorp
                                             4905 Lakeway Drive
                                             College Station, Texas 77845 USA
                                             800-STATA-PC        http://www.stata.com
                                             979-696-4600        stata@stata.com
                                             979-696-4601 (fax)

40-student Stata lab perpetual license:
       Serial number:  30110515653
         Licensed to:  IS Services
                       Trinity College Dublin

Notes:
      1.  (/m# option or -set memory-) 10.00 MB allocated to data

running C:\TCD\2009-10\classes\session05\stata\profile.do ...

.
```

Figure 2: Stata Workplace and Windows at start up

1. **GRAPH Window** This window displays graphs.
2. **VIEWER** This Window shows help and other miscellaneous files.
3. **DATA EDITOR** Browse, display, edit or enter data.
4. **DO FILE EDITOR** Edit and submit lists of Stata commands to Stata

The buttons on the bar below the menu bar include access to these four windows which can also be accessed from the menu bar. Note that the Command window is not active when the data edit or browse windows are active. If you need to look at the data use the browse window rather than the edit window.

5 Log files in Stata

As you work, Stata can be instructed to maintain two types of log files. The first is a simple list of all Stata commands issued (the command log). The second is a complete record of all Stata input and output (the Stata log or simply the log). For each log there is provision to overwrite the old file or to add the log to the previous file. The set up in the proposed `profile.do` file is to start both log files at start up with the command log added to the old command file and the new complete input and output log overwriting the old file. If you wish to keep your old log file you can rename it before restarting Stata As set up both files are in plain text format and extracts can be cut and pasted from them into a word processor.[11] The files may also be saved in Stata's own smcl format which is suitable for viewing and printing in the Stata viewer window but is not suitable for transfer to a word processor.

If you look at the bar at the bottom of the RESULTS window and you see a notification `log on (text)` you know that Stata is recording a log of all input and output for you as you proceed. If you enter

```
log off
```

The entry will change to `log off (text)` and logging will be suspended. Now enter

```
log on
```

and you will revert to `log on (text)` and logging will be resumed. The `(text)` in the notification means that the log is being kept in plain text. It can thus be pasted and copied to a word processor. The default logs are kept in smcl (Stata Mark up and Control Language) which are not easily cut and pasted to a word processor and are best avoided. If you want to change log files you must first close the current log

```
log close
```

To reopen the log with a new file use the command

```
set logtype text
log using newlog.txt
```

Similarly the command `cmdlog using filename.txt` records all your input instructions to `filename.txt`.

Now return to the file `profile.do` and see if you understand better what it does.

6 Obtaining HELP in Stata

The printed manual for Stata 11 is made up of 19 volumes with about 9000 pages. These vary from the Users' Guide, the Base Command Reference Manual (3 volumes) a Time Series Reference Manual, a Longitudinal/Panel Data Reference Manual, Windows Getting Started Booklet and several specialist Manuals dealing with specific statistical problems. The manuals are now available in pdf format in each standard installation. They can be accessed from the help menu within Stata or from the `utilities` subdirectory of your Stata installation. Much of the material in the manuals is also available in the standard on-line documentation. The layout of this is similar to the the usual windows help system. The syntax of any command can be obtained through this on-line help system. You can also search the on-line help files for keywords. There are also hypertext links within the help files that allow you to move between one entry and another. In a introductory note such as this very little details of syntax are included and the use of the Stata requires frequent use of on-line help and/or manuals. The on-line help contains hyperlinks to the relevant manual sections. The command `help help_advice` or the corresponding menu item gives advice on how to make best use of the help system.

[11]Remember to change the font of tabular or similar material cut from a log file to a fixed font such as Courier. To accommodate page width you may also have to reduce the font size.

Stata is widely used and there is a lot of relevant material on the web. If you have a problem it is likely that someone else has had the same problem. You may find a solution in the FAQs or Resource links on the Stata web site (`http://www.Stata.com/`). You should have a good browse on this site and be familiar with what is available there.

As a newcomer to Stata you may be confused by the fact that there are often many ways to do the same thing. As you get more experience in the use of Stata you will find that very often this flexibility is useful.

7 Managing data in Stata.

7.1 Data types in Stata

Stata uses data in several forms. You can determine the form of data being used in each of your series with the `describe` command. Details are given in subsection 7.8 on page 20.

ordinary numbers This is the form of data that we expect to use in most of our calculations. Internally Stata stores data in several forms

byte for integers from -127 to $+100$ (1 Byte)

int for integers from -32767 to 32740 (2 Bytes)

long for integers from -2147483647 to 2147483620 (4 Bytes)

float For floating point numbers from $-1.70141173319 \times 10^{38}$ to $1.70141173319 \times 10^{38}$ (4 Bytes)

double For floating point numbers from $-8.9884656743 \times 10^{307}$ to $-8.9884656743 \times 10^{307}$ (8 Bytes)

If you are importing data into Stata it usually chooses the appropriate way to store the imported data. Occasionally you may wish to change these defaults or to compact your data to save computer memory. Unless your data set is very large this is usually not a problem.

String data A string data item consists of a “string” of printable characters. Strings are used for two purposes.

Identifying data This data item might contain reference data such as the name of the respondent to a survey or his address or other identifying material. It will usually not be used in calculations

Other String Data This may contain the answers in words to various questions.

Factors or Categorical data The respondents to a sample may belong to various categories. Examples are Male/Female or County of residence or Social Status. These variables are stored as non-negative integers in the range 0 to 32740. Often these values have labels attached. Thus for a variable `healthinsur` 1 might correspond to “VHI member”, 2 to “Quinn Healthcare” and 3 to “AVIVA Health Care”. If one browses the variable it will look like string data but it is not. In this case and in many other cases the actual numbers have no numerical meaning. Stata will calculate averages, standard deviations but these have no meaning. You should exercise care. The Male/Female variable might be stored as 0 for Males and 1 for Females with appropriate labels

Name	Sex	Age	Income
John	Male	21	405.25
Paul	Male	25	350.00
Ann	Female	28	450.14
Mary	Female	31	375.67
Sheila	Female	24	325.35
Dan	Male	21	250.13

Table 1: Simple Data Set

describing the variables. The Counties might be coded as integers 1 to 32 with appropriate labels. Such variables are known as factor[12] variables. The order in which a categorical variable is coded does not, in general, have any quantitative meaning. Such variables must, in general, be converted to dummy variables before being used in regression analyses. The Stata command `encode` translates string variables to factors. The command `decode` converts factor variables back to string. For more details see subsections 9.3 (page 29) and 11.2 (36)

It is possible that, in certain circumstances, numbers are stored as printable characters. In such circumstances the `describe` command will describe them as strings. The Stata command `destring` will convert such "numeric" strings back to an appropriate numeric form.

7.2 Stata Data Sets

A data set in Stata may be visualised as a rectangle of data or a data frame. Each column of the frame contains the data for one variable. Each row of the frame contains one observation for each variable. Table 1 is an example of a simple data set. There are 6 observations on 4 variables. The first variable is a string variable which acts as an index (reference) for the observations. the variable "Sex" is a string variable. The variables "Age" and "Income" are numeric variables. We would probably need to change "Sex" to a factor variable before analysis (assuming that it is not already a factor variable).

The Journals data set analysed in Stock and Watson (2007), page 288 is a further example of a Stata data set. This data set gives details of subscriptions to economics journals at US libraries, for the year 2000.[13] This data set contains 180 observations on 10 variables. The variables[14] in the data set are

code – Journal Code,

title – Journal title,

publisher – publisher name,

society – Is the journal published by a scholarly society?

[12]It should be noted that this is a different concept of factor to that used in, for example, Arbitrage Pricing Theory

[13]I have taken this data set from the R package AER (Kleiber and Zeileis (2008)). They note that the data as obtained from `http://wps.aw.com/aw_stock_ie_2` contained two journals with title "World Development". One of these (observation 80) appeared to be an error and was changed to "The World Economy"

[14]See also `http://www.econ.ucsb.edu/~tedb/Journals/jpricing.html` for general information on this topic as well as a more up-to-date version of the data set.

Table 2: Extract from Stock and Watson (2007) data set

Code	title	publisher	society	price	pages	charpp	citations
APEL	Asian-Pacific ...	Blackwell	no	123	440	3822	21
SAJoEH	South African ...	SAHA	no	20	309	1782	22
CE	Computational ...	Kluwer	no	443	567	2924	22
MEPiTE	MOCT-MOST ...	Kluwer	no	276	520	3234	22
JoSE	Journal of Soc...	Elsevier	no	295	791	3024	24
LabEc	Labour Economics	Elsevier	no	344	609	2967	24
EDE	Environment ...	CUP	no	90	602	3185	24
RoRPE	Review of ...	Elsevier	no	242	665	2688	27

price – Library subscription price,

pages – Number of pages,

charpp – Characters per page,

citations – Total number of citations,

foundingyear – Year journal was founded,

subs – Number of library subscriptions and

field – field of speciality of journal.

Table 2 contains the first few columns and rows of this data set. Stata stores missing data as "." (a full stop). (It may also store missing values as one of ".a", ".b", ..., ".z").

7.3 Reading data in native Stata form

Stata has its own native format with default extension `.dta`. Reading and saving data in this format is very easy. If your data is in the Stata file `s:\project1\housing.dta` you can read it into memory with the command

```
use housing
```

if the your working directory is `s:\project1`, or

```
use s:\project1\housing
```

otherwise.

You may enclose filename in double quotes and must do so if your filename contains blanks or other special characters.

Similarly if you wish to save a file in Stata format you may enter

```
save housing
```

to save `housing.dta` in your working directory or

```
save s:\project1\housing
```

This works if the file `housing.dta` does not exist. If the file already exists Stata will assume that you do not wish to overwrite an existing file and you must enter

```
save housing, replace
```

or

```
save s:\project1\housing, replace
```

and the existing file will be overwritten. Before reading data into Stata existing data must be cleared. This may be done by

```
clear
use housing
```

or

```
use housing,clear
```

7.4 Data in Spreadsheet Format

In all likelihood your data will have come from a program other than Stata. Presume that they look something like the data in Table 2 and is available in an excel file. Stata can not read Excel format directly and it should be transformed to a suitable text format. This can be done in many ways but I tend to work with comma separated value (csv) files. A csv file corresponding to the data displayed in the table would look like.

```
Code,title,publisher,society,price,pages,charpp,citations
APEL,"Asian-Pacific ... ",Blackwell,no,123,440,3822,21
SAJoEH,"South African ...",SAHA,no,20,309,1782,22
CE,"Computational ...",Kluwer,no,443,567,2924,22
MEPiTE,"MOCT-MOST ...",Kluwer,no,276,520,3234,22
JoSE,"Journal of Soc..." =,Elsevier,no,295,791,3024,24
LabEc,"Labour Economics",Elsevier,no,344,609,2967,24
EDE,"Environment ... ",CUP,no,90,602,3185,24
RoRPE,"Review of ... ",Elsevier,no,242,665,2688,27
```

Note that

1. Each row of the spreadsheet is on one line,
2. Columns are separated by commas,
3. Where there are text entries (also known as string variables) and they contain spaces they have been enclosed in quotation marks.[15]
4. This file has variable names separated by commas on the first line and values for each observation, separated by commas on the following line.
5. If the first line does not contain variable names they may be given in the Stata instruction to read the data or Stata will name the variables.
6. csv files saved from Excel are saved as displayed in Excel. If you wish to retain full accuracy you must change the display format so that it displays the required accuracy and then save as csv (or text).

The entire journals file can be read into Stata by

[15] single or double quotation marks may be used

```
insheet using journals
```

If the names on the first line were missing

```
insheet Code title publisher society price pages charpp citations foundingyear subs field
```

would produce the same results.

Once imported into Stata I would save the data-set in native Stata format. The instruction below saves current data in a file, in the current warning directory, called `journals.dta`. (Look at the bottom left hand corner of the Stata work space to ensure that the working directory is properly set. If you want to save the file elsewhere specify a full path and file name using quotes if the path or file name contain blanks,

```
save journals
```

Warning — Unlike some other econometric packages Stata is case sensitive; that is r and R are different variables. This applies to commands as well. The command to read a Stata data set is `use` — `USE` or `Use` will not work.

Always check that your data has been properly imported. Look at the list of series to check that the names are as expected. Browse the data to ensure that it is correct. It is often useful to graph the series. On a graph wrong entries often show up clearly. There are several problems that can occur in transferring data –

1. Entries in the original file may be entered in different units (e.g. some entries in grams and some in kilogrammes or indices to different bases in different parts of the series).

2. The source data may use a non-standard code for missing values. Ignoring this problem is likely to lead to disaster. If a missing value in a series is coded as NA the entire variable will be imported as a string variable and will not convert to numeric. This is inconvenient. Use your editor to replace NA with the Stata code for a missing value (a period "."). If the source may has a special code for missing values, you should either edit your file or use the special procedures in Stata for reading this data.

3. Sometimes the variable names or descriptions in the first row contain blanks or other characters that are not allowed in Stata variable names.[16] You can generally solve such problems by editing the original file in a text editor.

4. As already mentioned you may lose some level of significance when you transferred your data from Excel format to csv. The default display in EXCEL has only two places of decimals and it may export to csv data as displayed rather than as held in memory. If this is a problem use the Excel format menu to ensure that the required significance is displayed before exporting to csv.

5. Sometimes, no matter what you do, it appears that the data will not import into Stata. You might try to import it into excel and, after some possible editing, export it to csv format and then try to import the revised file into Stata. You should in any case have a look at your data in excel and/or a text editor before importing it to Stata. I have found the data import/export facilities in the OpenOffice spreadsheet program Calc more flexible than Excel for processing data in this way.

[16]Variable names may be 1 to 32 characters long and must start with a-z, A-Z, or _, and the remaining characters may be a-z, A-Z, _, or 0-9. I would not recommend starting variable names with _ as this is used by many internal variables.

6. There is a limit to the number of columns that excel can manage. Stata may be able to import such data and you may be able to examine it in a text editor. I have used Unix utilities (e.g. gawk) and the script program Python to extract required data from multi-gigabyte data sets. Presumably such problems could also be handled by a data base program. At this stage it is sufficient to know that such data sets can be managed.
7. Always read the documentation that came with your data. Check if there is other printed documentation available elsewhere. Can you easily replicate some other analysis of the data.
8. Stata may not know that it has not read your data properly and may issue no warnings. **It is up to you to ensure that your data has been read properly**.

7.5 Entering Data by Hand

You may enter data into Stata using the Stata data editor. This is probably a last resort and is not recommended. It is probably better to enter data into a spreadsheet file and to transfer it to Stata using the methods described above. Data entered manually should be independently check-read by a second person. At one time this would have been standard practice for all persons making up tables. We all make mistakes. To access the data editor use the button on the icon bar or `[Window - Data Editor]` from the drop-down menus. If you just want to look at the data use the browse window – This will not change data in error.

7.6 Merging two data sets – append a data set to another

Two data sets may be joined in two ways.

append `append` is used to join a data set that has new or extra observations of the same variables as the original data set.

merge `merge` is use to add new variables to a data set.

The following example taken from Hamilton (2004) and involves three files. (Recall that the symbol "." represents a missing value in Stata.) The Stata command, `list`, displays or lists data in memory. The command, `sort`, sorts the data. As an exercise you should look at the help files for some Stata commands such as `list`, `sort`, `use`, `append` etc. There is no need to memorise these as they will always be there for consultation. Skim over the details but you should be able to use the help files to add to your understanding of what we are doing here.

```
. use newf1, clear
(Newfoundland 1985-89)

. list

     +---------------+
     | year      pop |
     |---------------|
  1. | 1985   580700 |
  2. | 1986   580200 |
  3. | 1987   568200 |
  4. | 1988   568000 |
  5. | 1989   570000 |
     +---------------+

. use newf2,clear
(Newfoundland 1990-95)

. list

     +-------------------------+
     | year      pop   jobless |
     |-------------------------|
  1. | 1990   573400     42000 |
  2. | 1991   573500     45000 |
  3. | 1992   575600     49000 |
  4. | 1993   584400     49000 |
  5. | 1994   582400     50000 |
     |-------------------------|
  6. | 1995   575449         . |
     +-------------------------+

. append using newf1

. list

     +-------------------------+
     | year      pop   jobless |
     |-------------------------|
  1. | 1990   573400     42000 |
  2. | 1991   573500     45000 |
  3. | 1992   575600     49000 |
  4. | 1993   584400     49000 |
  5. | 1994   582400     50000 |
     |-------------------------|
  6. | 1995   575449         . |
  7. | 1985   580700         . |
  8. | 1986   580200         . |
  9. | 1987   568200         . |
 10. | 1988   568000         . |
     |-------------------------|
 11. | 1989   570000         . |
     +-------------------------+

. sort year

. list

     +-------------------------+
     | year      pop   jobless |
     |-------------------------|
  1. | 1985   580700         . |
  2. | 1986   580200         . |
  3. | 1987   568200         . |
  4. | 1988   568000         . |
  5. | 1989   570000         . |
     |-------------------------|
  6. | 1990   573400     42000 |
  7. | 1991   573500     45000 |
  8. | 1992   575600     49000 |
  9. | 1993   584400     49000 |
 10. | 1994   582400     50000 |
     |-------------------------|
 11. | 1995   575449         . |
     +-------------------------+
```

Be very careful with the command append. List as necessary to ensure that you have done what you think you have done. Computers are very dumb and only do what you told them to do. Often this is not what you want them to do! We all have this problem from time to time.

```
. save newf3
file newf3.dta saved

. use newf4, clear
(Newfoundland 1980-94)

. list

     +--------------------------+
     | year   births   divorces |
     |--------------------------|
  1. | 1980    10332        555 |
  2. | 1981    11310        569 |
  3. | 1982     9173        625 |
  4. | 1983     9630        711 |
  5. | 1984     8560        590 |
     |--------------------------|
  6. | 1985     8080        561 |
  7. | 1986     8320        610 |
  8. | 1987     7656       1002 |
  9. | 1988     7396        884 |
 10. | 1989     7996        981 |
     |--------------------------|
 11. | 1990     7354        973 |
 12. | 1991     6929        912 |
 13. | 1992     6689        867 |
 14. | 1993     6360        930 |
 15. | 1994     6295        933 |
     +--------------------------+

. sort year

. merge year using newf3

. list

     +-----------------------------------------------------+
     | year   births   divorces      pop   jobless   _merge |
     |-----------------------------------------------------|
  1. | 1980    10332        555        .         .        1 |
  2. | 1981    11310        569        .         .        1 |
  3. | 1982     9173        625        .         .        1 |
  4. | 1983     9630        711        .         .        1 |
  5. | 1984     8560        590        .         .        1 |
     |-----------------------------------------------------|
  6. | 1985     8080        561   580700         .        3 |
  7. | 1986     8320        610   580200         .        3 |
  8. | 1987     7656       1002   568200         .        3 |
  9. | 1988     7396        884   568000         .        3 |
 10. | 1989     7996        981   570000         .        3 |
     |-----------------------------------------------------|
 11. | 1990     7354        973   573400     42000        3 |
 12. | 1991     6929        912   573500     45000        3 |
 13. | 1992     6689        867   575600     49000        3 |
 14. | 1993     6360        930   584400     49000        3 |
 15. | 1994     6295        933   582400     50000        3 |
     |----------------------------------------------------|
 16. | 1995        .          .   575449         .        2 |
     +----------------------------------------------------+

. exit, clear
```

`merge` has options `update` and `replace` which can be used to replace and update data sets. Note the new variable _merge in the merged data set. A value of 1 indicates that the observation is in the master data set only, 2 indicates observation in the using data set only and 3 the observation is in both data sets (using values ignored if different). (_merge has a different value if update option is being used). If you wish to do a second merge you need to delete the _merge variable. Note that data sets to be merged or appended must be in Stata formats

```
drop _merge
```

7.7 Memory settings

In the default set up Stata version 11 allocates I0 MB of memory to data.[17] The command

```
set memory #m
```

sets data memory to # MB. This command must be issued when there is no data in memory. You should save data and clear data memory before issuing this command. Thus ideally the command should be issued before you start your analysis. If your PC is short of memory and your data set is small use a smaller amount. The data memory allocation must be sufficient to cover the variables in memory along with all temporary variables generated by the Stata during calculations.

For large data set one may use up to 32766 variables. The default maximum is 5000. (in Intercooled Stata this is fixed at 2048) The maximum may be amended by the instruction

```
set maxvar #
```

To specify the maximum number of variables that may be included in a model (10 to 11000 with a default of 400) use the command

```
set matsize #
```

If you get a message that you have insufficient memory such as `No room to add more variables` or `No room to add more observations` you have the following options

- Use `compress` command. The `compress` command reduces stores the data in the most efficient format.
- Drop some variables – Use `drop command`
- Use `set memory` as above

The commands `query memory` or `memory` will give details of current memory and data settings.

[17]see page 6 for details of how to change default memory allocations from the Stata short cut. You may also include appropriate instructions in your `profile.do` file. The defaults and limits described in this section depend on the actual version of Stata used

7.8 Sorting and Describing data

`browse` displays data in the data editor. Data can not be changed with `browse` which is thus safe. `edit` allows the data to be changed and requires care. `edit` is not recommended if you only want to look at the data.

`describe` describes the data in memory. Loading the file `newf3.sta` that we saved in the previous example

```
. clear
. use newf3
(Newfoundland 1990-95)

. describe

Contains data from newf3.dta
  obs:            11                          Newfoundland 1990-95
 vars:             3                          13 Sep 2005 17:37
 size:           154 (99.9% of memory free)
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
year            int    %9.0g                  Year
pop             float  %9.0g                  Population
jobless         float  %9.0g                  Number of people unemployed
-------------------------------------------------------------------------------
Sorted by:  year
```

```
sort area
```

sorts observations in memory in order of the variable area. `gsort` is a more general sort command that allows sorting in ascending or descending order.

The command `summarise` gives number of observations (non-missing), mean, standard deviation, minimum and maximum for each variable in memory or in a list of variables if a list is given. `summarise var1 var2 var3`.

```
. use newf3
(Newfoundland 1990-95)

. summarise

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        year |        11        1990    3.316625       1985       1995
         pop |        11    575622.6    5692.075     568000     584400
     jobless |         5       47000    3391.165      42000      50000

. exit, clear
```

If the data have been sorted on var1

```
by var1 summarise var2 var3
```

gives the same statistics for var2 and var3 for the subgroups defined by var1. For example, if var1 is gender the Information and univariate statistics about variables

Probably the most useful instruction for constructing tables is the `table` command. I shall use the data set earn.asc from Baltagi (2002) to illustrate its use. This is a cross-section of 595 individuals. the variables included in the cross-section are

EXP = Years of full-time work experience.

EXP2 = Square of EXP.

WKS = Weeks worked.

OCC = (OCC=1, if the individual is in a blue-collar occupation).

IND = (IND=1, if the individual works in a manufacturing industry).

SOUTH = (SOUTH=1, if the individual resides in the South).

SMSA = (SMSA=1, if the individual resides in a standard metropolitan statistical area).

MS = (MS=1, if the individual is married).

FEM = (FEM=1, if the individual is female).

UNION = (UNION=1, if the individual's wage is set by a union contract).

ED = Years of education.

BLK = (BLK=1, if the individual is black).

LWAGE = Logarithm of wage.

M = (M=1, if the individual is male).

F_EDC = Years of education for females only.

(These are the variable names in the original data set and they probably date back to a time when there were restrictions on the length of variable names. To-day one is less constricted and one should choose mere appropriate names to make your programs/scripts more readable. For example I might have labelled the variable `OCC` as `blueCollar` or perhaps as `blue_collar`.

The commands `table var1, contents(...)` or `table var1 var2, contents(...)` can be used to produce frequency counts in cells of the tables or to produce summary statistics. Here we produce average earnings for males females and northern states and southern states.

```
. use earn

. describe

Contains data from earn.dta
  obs:           595
 vars:            15                          13 Sep 2005 22:32
 size:        13,685 (99.9% of memory free)
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
exp             byte   %8.0g                  EXP
exp2            int    %8.0g                  EXP2
wks             byte   %8.0g                  WKS
occ             byte   %8.0g                  OCC
ind             byte   %8.0g                  IND
south           byte   %8.0g                  SOUTH
smsa            byte   %8.0g                  SMSA
```

```
ms              byte   %8.0g                 MS
fem             byte   %8.0g                 FEM
union           byte   %8.0g                 UNION
ed              byte   %8.0g                 ED
blk             byte   %8.0g                 BLK
lwage           float  %9.0g                 LWAGE
m               byte   %8.0g                 M
f_edc           byte   %8.0g                 F_EDC
-------------------------------------------------------------------------------
Sorted by:

. summarise

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         exp |       595    22.85378    10.79018          7         51
        exp2 |       595    638.5277    552.4278         49       2601
         wks |       595     46.4521    5.185025          5         52
         occ |       595     .512605    .5002617          0          1
         ind |       595     .405042    .4913132          0          1
-------------+--------------------------------------------------------
       south |       595     .292437    .4552647          0          1
        smsa |       595    .6420168    .4798105          0          1
          ms |       595     .805042    .3965017          0          1
         fem |       595     .112605    .3163754          0          1
       union |       595    .3663866    .4822222          0          1
-------------+--------------------------------------------------------
          ed |       595    12.84538    2.790006          4         17
         blk |       595    .0722689    .2591505          0          1
       lwage |       595    6.950745    .4384033    5.67675      8.537
           m |       595     .887395    .3163754          0          1
       f_edc |       595    1.445378    4.146918          0         17

.  generate wage = exp(lwage)

. table m, contents(mean wage)

----------------------
        M | mean(wage)
----------+-----------
        0 |   735.8211
        1 |   1200.347
----------------------

. table m south, contents(mean wage)

------------------------------
          |       SOUTH
        M |        0         1
----------+-------------------
        0 | 770.1432  678.1601
        1 | 1250.834  1071.926
------------------------------

. table m south,contents(freq mean ed sd ed)

------------------------------
          |       SOUTH
        M |        0         1
----------+-------------------
        0 |       42        25
          |  12.8571      12.8
          | 2.425187  2.723356
          |
        1 |      379       149
          |  13.0633   12.2953
          | 2.667897   3.13115
------------------------------
. exit, clear
```

The options available in the table command are

freq for frequency

mean varname for mean of varname

sd varname for standard deviation of varname

sum varname for sum of varname

rawsum varname for sums ignoring optionally specified weight

count varname for count of non-missing observations of varname

n varname same as count

max varname for maximum of varname

min varname for minimum of varname

median varname for median

p1 varname for 1st percentile

p2 varname for 2nd percentile)

p50 varname for 50th percentile – same as median

p98 varname for 98th percentile

p99 varname for 99th percentile

iqr varname for interquartile range

`contents(freq)`is assumed if contents() is not specified.

8 Entering commands and do–files

8.1 Command Line, Drop Down Menus and `.do` Files

Up to this stage I have concentrated on entering Stata commands through the Command Window. Previously issued commands can be recalled from the Review window to the command window edited and sent again to Stata. Such commands are saved in the command log file. We may edit this command file and create a sequence of Stata commands or a `.do` file which may be used in Stata.

There is an alternative way of working. Instead of typing the command

```
use earn
```

we could use [File – Open] from the Drop Down Menus and negotiate through the directories to `earn.dta` and select open. This will generate a command similar to

```
use "D:\TCD\2005-6\Examples\baltagi2002\earn.dta", clear
```

This will be saved in the command log file in much the same way as the command entered through the command window. The `table` command [Statistics – Summaries tables and tests – Tables – Table of summary statistics (table)]. As you proceed you may find that the drop menus are a good way of initially entering commands. You will find it useful to recall these commands from the review window and fine tune them for further work.

The file

```
insheet using earn.csv, clear
describe
save earn
summarise
generate wage = exp(lwage)
table m, contents(mean wage)
table m south, contents(mean wage)
table m south,contents(freq mean ed sd ed)
exit, clear
```

is an example of a `.do` file that is an edited version of the Stata commands that reads data in delimited format, saves the data in Stata format and then produces the tables from the example above. Note that the `.do` file is not the same as the command log file. You must first edit[18] (as a text file) the command log and save the appropriate commands in the `.do` file.

You may also open a `.do` file editor window in Stata, edit files and submit parts of the file or the entire file to Stata. You are given two options "do" or "run". "do" echoes the commands and "run" is silent. Try it. If you have a problem with a do file this is where it should be tested and run. In any project or submission using Stata you will be required to produce a full and complete set of the `.do` files used to produce your results from the original data set(s). These `.do` files should enable the person assessing the project to duplicate all your results starting with the data provided. (You will also be required to submit the log files that arise from the application of these files in Stata)

8.2 Comments in do-files and programs

There are several reasons why you should make liberal use of comments in your do–files and programs. First unless you have an extremely good memory you will forget the names that you have put on various variables. The logic that came to you in a flash of inspiration may be gone the next day unless you make a note of it. Secondly you will be submitting these programs for assessment and the assessor will not understand them unless you explain what you are doing in comments throughout the program. You can **not** get marks for what your assessor can not understand. Also part of the marks for a project will be for good comments and layout.

There are several ways of adding comments in do-files and programs (ado-files).

1. begin the line with *.

[18]Edit your do-file with the Stata do-file editor or another text file editor (e.g. Scite, Context, Notepad). Do **not** use MS WORD. Even if you use MS WORD, and save as a text file, the default WORD grammar and spelling corrections may make your do–file unusable in Stata. There are ways around this problem but you must know how to disable the offending features in the version of WORD that you are using. It is easier to use a proper text editor.

2. place the comment between /* and */ delimiters.
3. begin the comment with //.
4. begin the comment with ///.

Note that the

- comment indicator * may only be used at the beginning of a line, but it does have the advantage that it can be used interactively. * indicates that the entire line is to be ignored by Stata.
- The /* and */ comment delimiter has the advantage that it may be used in the middle of a line, but it is more cumbersome to type than the other comment indicators. What appears inside the delimiters /* and */ is ignored.
- The // comment indicator may be used at the beginning of a line or at the end of a Stata instruction. Note, however, that if the // indicator is at the end of a Stata instruction, it must be preceded by one or more blanks. // indicates that the rest of the line is to be ignored.
- The /// comment indicator instructs Stata to view from /// to the end of a line as a comment, and to join the next line with the current line as one Stata instruction. By splitting long instructions over several lines the /// comment indicator is one way to make long lines more readable. Like the // comment indicator, the /// indicator must be preceded by one or more blanks. Using a series of /// instructions you may spread one Stata instruction over many lines.

To summarise comments should include the following

1. Your name and student number.
2. Date of last revision of do file.
3. A brief description of the program.
4. List of variables in your data set and their definitions.
5. Comments on what you are doing and why you are doing them.
6. Comments on your results.

9 Creating/Transforming variables

9.1 Using Functions

The command `generate` creates new variables using various functions and transformations of variables. Enter `help function` for a guide to functions. This will give you an introduction to functions and guide you to mere help (`help mathfun` gives you further help on mathematical functions. If `inc` is income you may create a new variable `linc`,[19] the log of income as follows[20]

[19]When I create a new variable which is the log of an existing variable I generally give it the name of the old variable preceded by an “l”. Feel free to adopt your own convention but but consistency makes your work more legible.

[20]For a detailed list of mathematical functions enter `help math functions`

```
generate linc = ln(inc)
```

Mathematical functions you might use include

abs(x) Absolute value of x

exp(x) Exponential

log(x) Same as `ln(x)` – natural log of x

log10(x) log of x to base 10

sqrt(x) Square root

sin(x) Sin of x in radians

Stata also contains a large number or probability functions. Use `probfun` or the manuals. Typical functions include

normal(z) Cumulative standard normal distribution, mean 0, variance 1

normalden(z) Standard normal density, mean 0, standard deviation 1 1

normalden(z,s) Standard normal density, mean 0, standard deviation s

normalden(z,m,s) Standard normal density, mean m, standard deviation s

invnormal(p) Inverse cumulative standard normal distribution. if `normal(z)=p` then `invnormal(p)=z`

rnormal() Normal Random numbers with mean 0 and standard deviation 1,

rnormal(m) Normal Random numbers with mean m and standard deviation 1,

rnormal(m,s) Normal Random numbers with mean m and standard deviation s,

Similar probability functions and random number generators exist for several other distributions. For details enter `help density functions` or `help random_number_functions`. Further random number generating functions are available in the `rng` function available for download.

`replace` has a similar function to `generate`. It changes values in an existing variable. Thus if `pop` was population and you wished to replace it by population in thousands you could write

```
replace pop = pop/1000
```

`display` does a single calculation and prints the result in the Results window. Thus

```
.  display sqrt(5)
2.236068

. display 2*4 +6
14
```

Stata remembers the values of many statistics calculated when a command is executed. These values can be recalled and used in further calculations using `display, generate and replace`. This is illustrated in the following code segment.

```
. use earn.dta, clear

. summarize lwage

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
       lwage |       595    6.950908    .4385926       5.68       8.54

. // Display what is stored in memory
. return list

scalars:
                  r(N) =  595
              r(sum_w) =  595
               r(mean) =  6.950907568570947
                r(Var) =  .1923634982076572
                 r(sd) =  .4385926335537992
                r(min) =  5.679999828338623
                r(max) =  8.539999961853027
                r(sum) =  4135.790003299713

. // Print some of these items
. display r(N)
595

. display r(mean)
6.9509076

. display r(sd)
.43859263

. display "total is " r(mean)*r(N)
total is 4135.79

. // Use some in a calculation and look at the result
. generate lwage0 = (lwage - r(mean))/r(sd)

. summarize lwage lwage0

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
       lwage |       595    6.950908    .4385926       5.68       8.54
      lwage0 |       595   -1.37e-09           1  -2.897695   3.623163
```

The values stored in `r(N)` etc... are those of the last `summarise` command. When another `summarise` command is issued the previous values are overwritten. Details about the `r()` and other values available after similar instruction are given in the Stata manuals.

`egen` (extensions to generate) operates with extra functions which are not available under `generate`. These often involve creating new variables involving means, sums, order statistics and similar For details see the help files or manual.

Variables may be created or changed on the drop-down menus [Data – Create or Change Variable – Create New Variable] and filling in the following dialogues.

9.2 Limiting the extent of commands: `in` and `if` qualifiers

`in #/#` at the end of a command directs that the command be applied only to the range specified.

`list x in 10` lists the 10^{th} observation (other commands may be substituted for list)

list x in 10/15 lists observations 10 to 15

list x in f/10 lists the first 10 observations

list x in 20/l lists from observation 20 to the last observation. Note that l here is the letter l (for last) and not the number 1

list x in -10/l lists the last 10 observations

`list` applies to the data as sorted.

The `if` condition selects observations based on values of the variable. If we have our earnings data set loaded

```
. generate wage=exp(lwage)
. summarise wage
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        wage |       595    1148.039    531.0816   291.9989    5100.02
. summarise wage if fem == 1
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        wage |        67    735.8211    279.2956   313.9991   1450.001
. summarise wage if wage > 1000
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        wage |       358    1437.545    487.6382   1000.005    5100.02
. summarise wage if fem==1 & wage>1000
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        wage |        12    1202.919    132.7065   1000.005   1450.001

.
```

An `if` condition may use the following logical operators —

== equals (A common error is to use a single = which will not work as a logical operator).

! = is not equal to

> is greater than

< is less than

>= is greater than or equal to

<= is less than or equal to

Logical phrases may be combined with the following logical operators —

& and

| or

! negation

name	sex	country	age	income
John	Male	Ireland	21	201
Paul	Male	UK	13	235
Anne	Female	France	15	371
Mary	Female	Germany	26	175
Diarmaid	Male	Ireland 25	222	
Frank	Male	UK	24	199
Eabha	Female	France	34	208
Abaigh	Female	Germany	6	236

Table 3: Example of Factor variables

9.3 Factors (Categorical Variables)

The variable `Country` in Table 3 is an example of a factor or categorical variable. This factor has four levels "Ireland", "UK", "France" and "Germany".

If we import this data set to Stata from csv format and then summarise we get a not very satisfactory result. This is because the `Country` variable is a string or character variable. In Stata factor variables are stored as integers (possibly labelled). The Stata command `encode` transforms the `country` variable from string (character) to factor generating the factor variable `countryfac`. This factor variable takes one of the integer values 1, 2, 3 or 4 with labels France, Germany, Ireland and UK respectively. (See page 11 for the definition of factors.

```
. insheet using factor.csv
(5 vars, 8 obs)

. summarize country

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     country |         0

. encode country, generate(countryfac)

. summarize  countryfac

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
  countryfac |         8         2.5    1.195229          1          4

. exit, clear
```

Note that as the variable `countryfac` takes the values 1, 2, 3 and 4 the command `summarize countryfac` works and gives a mean of 2.5 and a Std. Dev of 1.195229. In this case and in many cases involving factors a mean and a standard deviation are totally meaningless.

The variable `sex` is also a string variable and may be converted to a dummy variable gender using the commands

```
generate gender = 0
recode gender = 1 if sex == "male"
```

`gender` is now the usual dummy variable that one might use in many regression equations. As it takes integer values it is also a factor with two levels. In this particular case `summarize male` will give the proportion of males in the sample.

To use `countryfac` in regressions we must first generate dummy variables. This can be done by using the `i.factor` notation described in subsection 11.2 on page 36.

If one wishes to generate permanent dummy variables from factors one could proceed as in the generation of the dummy variable for `gender` or use the `tabulate ..., generate()` sequence as below.

```
tabulate country, generate(co)
```

which generates four dummy variables `co1`, `co2`, `co3` and `co4` for Country 1, 2, 3 and 4 respectively.

10 Graphs

The Graphics manual for Stata 11 contains 649 pages describing its graphics facilities. These facilities are comprehensive and capable of producing very elaborate publication quality graphs. Stata instructions for graphics can be complicated. Any reasonable description of the facilities available and the syntax of such instructions would require a note longer than this introduction. However it is possible to draw graphs using the **Graphics** item on the menu bar. The two sample graphs below have been produced from the graphics menus in Stata, saved in the command log, extracted to a do file and then run again in Stata. The `graph export` command was generated from the **File|Save As** menu instruction in the graphics Window. This particular form of the `graph export` saves the graph in eps format.[21] The eps file was then imported into this document. A similar procedure will work in WORD.

The code generated by the menu system is automatically saved in the `cmdlog.txt` file if you use the `profile.do` file recommended here. The instructions were then copied from the command log to the do file. To make the do-file more readable I have inserted several /// instruction break comments.

```
. clear

. insheet using Journals.csv, comma
(11 vars, 180 obs)

. // Generate required variables for plots
. generate lsubs = ln(subs)

. generate citePrice =  price/citations

. generate lcitePrice =  ln( citePrice)

. //
. twoway (scatter lcitePrice lsubs), ytitle(log(price per citation)) xtitle(log(subscriptions)) ///
> title(Scatter Plot of log(subscriptions) and log(price per citation))

. graph export "C:\TCD\2009-10\document preparation\Introduction to stata\data_
```

[21]The Encapsulated Postscript format (eps or epsf) is derived from the Adobe Postscript format. It is suitable for high quality graphics. Most econometric, statistical, mathematical or graphics programs can produce this format. it can also be imported by most word processing and document production systems. Stata can export graphics files in a variety of formats. Details are in the manuals.

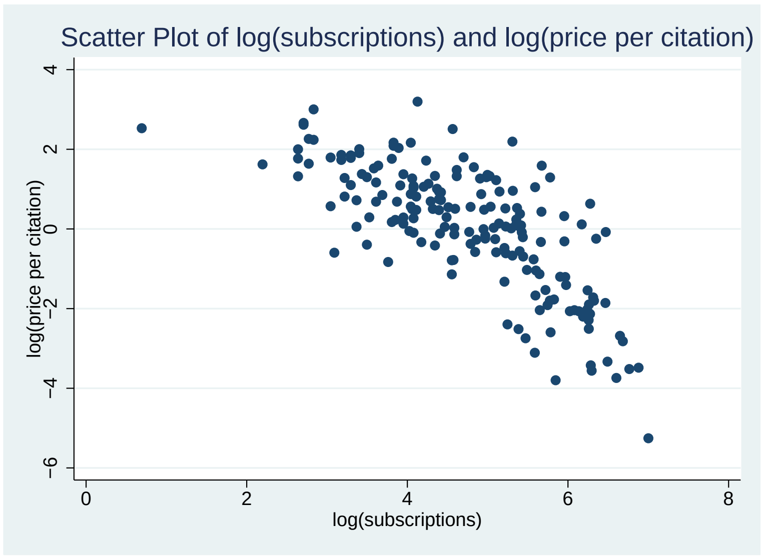


Figure 3: Sample Graph – Scatter Plot of Journals Data

```
> program_files\sw01.eps", ///
>         as(eps) preview(off) replace
(file C:\TCD\2009-10\document preparation\Introduction to stata\data_program_files
\sw01.eps written in EPS format)
```

The graph is reproduced as figure 3. The following code adds a least squares fit to the graph as shown in figure 4. It is relatively easy to amend the labels etc on the graphs by editing the do files.

```
. twoway (scatter lcitePrice lsubs) (lfit lcitePrice lsubs), ///
>                                        ytitle(log(price per citation)) ///
>         xtitle(log(subscriptions))  ///
>         title(OLS fit of log(subscriptions) and log(price per citation))
. graph export "C:\TCD\2009-10\document preparation\Introduction to stata\data_
> program_files\sw02.eps", ///
>         as(eps) preview(off) replace
(file C:\TCD\2009-10\document preparation\Introduction to stata\data_program_fi
> les\sw02.eps written in EPS format)

.
end of do-file

. exit, clear
```

11 Regression

11.1 Stata regress command

In Stata linear regression is done with the `regress` command. The general syntax of that command is

```
regress depvar [indepvar] [if] [in] [weight] [,options]
```

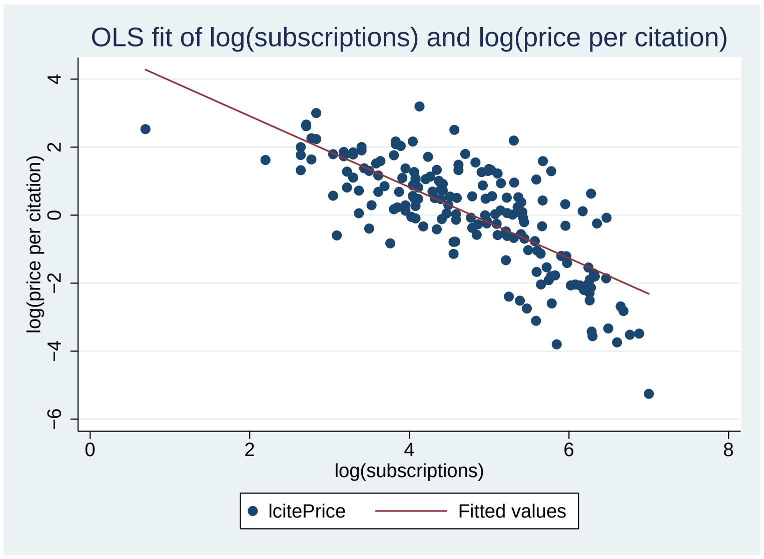


Figure 4: Sample Graph of Scatter Plot of Journals Data and Least Squares Fit

where

depvar dependent variable

indepvar list of independent variables

if and in limit the range in data set to which the regression is applied.

and the main options are

noconstant By default Stata adds a constant (`_cons`) to the model. This option estimates the model without adding the constant. Perhaps you have a set of mutually exclusive dummy variables that cover the entire data set (add to 1).

hasconstant On occasion you will have a predetermined constant in the model. If so it should be included last in the list of independent variables.

vce(vcetype) Allows the calculation of various Heteroskedastic consistent estimators of variances. The options available are specified by the following values of `vcetype`. (Many of these options may not apply to the analysis of survey samples with a complex design. In such cases check the manual before use.)

ols (default) Calculates the standard OLS estimate of the variance.

robust Amended[22] Huber-White Standard Errors, called HC_1 in Davidson and Mackinnon (2004) (page 200)

[22]The original Huber-White Standard Errors detailed in many texts (e.g. Verbeek (2008)), page 93-4 are not calculated, as far as I know, by Stata. The Amended versions are to be preferred. Davidson and Mackinnon (2004) contains a discussion about the most appropriate one to use. They conclude that HC_3 is possibly best although it can be outperformed by HC_2 on occasion. The best procedure to use probably depends on the particular $\boldsymbol{X}$ matrix. Use the vce() option rather than the `robust` option from earlier versions of Stata.

hc2 Amended Huber-White Standard Errors, called HC_2 in Davidson and Mackinnon (2004) (page 200)

hc3 Amended Huber-White Standard Errors, called HC_3 in Davidson and Mackinnon (2004) (page 200)

cluster Allows for intra-group correlation. This is the cross-section equivalent of Newey-West HAC estimation in time series. Suppose that we have data taken from a survey of individuals in towns in Ireland and we are estimating and OLS equation for some characteristic. For each town we might only have a single number for the distance from Dublin which is then the same for all persons in the sample. It is possible that in such circumstances that the disturbances within towns might be correlated and a `cluster` adjustment for heteroskedasticity be made.

bootstrap Allows for more complex situations where standard assumptions do not hold

jacknife Allows for more complex situations where standard assumptions do not hold

level(#) set confidence level – default is level(95)

other See manual or help files for additional options

To illustrate the regression facilities in Stata I will use a data set on cigarette smoking taken from Baltagi (2002). This contains five variables

1. `OBS` = observation number.
2. `STATE` = State abbreviation.
3. `LNC` = the logarithm of cigarette consumption (in packs) per person of smoking age (> 16 years) for 46 states in 1992.
4. `LNP` = the logarithm of real price of cigarette in each state.
5. `LNY` = the logarithm of real disposable income per-capita in each state.

An OLS regression of `LNC` on `LNP`, `LNY` and a constant may be carried out as follows.

```
. regress lnc lnp lny

      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  2,    43) =    9.38
       Model |  .500984744     2  .250492372           Prob > F      =  0.0004
    Residual |  1.14854473    43  .026710343           R-squared     =  0.3037
-------------+------------------------------           Adj R-squared =  0.2713
       Total |  1.64952947    45   .03665621           Root MSE      =  .16343

------------------------------------------------------------------------------
         lnc |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |  -1.338335   .3246015    -4.12   0.000    -1.992957    -.683714
         lny |   .1723862   .1967544     0.88   0.386    -.2244069    .5691793
       _cons |   4.299661   .9089257     4.73   0.000     2.466637    6.132684
------------------------------------------------------------------------------
```

The upper left hand portion of this output is the Analysis of Variance (ANOVA) table for the regression. If the regression model is (using the usual notation).

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$$

and

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

The ANOVA Table (assuming OLS and a constant in the model) is given by –

Source	SS	df	Mean Square
Model	$\hat{\beta}'X'y$-$n\bar{y}^2$	$k-1$	
Residual	$e'e$	$n-k$	s^2
Total	$y'y$-$n\bar{y}^2$	n-1	$S_{yy}/(n-1) = s_y^2$

For details see (Green, 2008, page 35).

In the upper right hand column the F statistic is given by

$$F(k-1, n-k) = \frac{(\hat{\boldsymbol{\beta}}'\boldsymbol{X}'\boldsymbol{y} - \bar{y}^2)/(k-1)}{\boldsymbol{e}'\boldsymbol{e}/(n-k)}$$

$$\texttt{R-SQUARED} = 1 - \frac{\boldsymbol{e}'\boldsymbol{e}}{\boldsymbol{y}'\boldsymbol{y} - n\bar{y}^2}$$

$$\texttt{Adj R-SQUARED} = 1 - (1 - R^2)(n-1)/(n-k)$$

$$\texttt{Root MSE} = s = \sqrt{\boldsymbol{e}'\boldsymbol{e}}/(n-k)$$

The table of coefficients, Standard Errors, t-statistics Probabilities and confidence intervals is self explanatory.

$$\texttt{Coef.} = \hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$
$$\texttt{Std. Err.} = \sqrt{(s^2(\boldsymbol{X}'\boldsymbol{X})^{-1})_{ii}}$$

Variables can be recovered after a regression include. (See description above of `summarize` command)

e(N) number of observations

e(mss) model sum of squares

e(df_m) model degrees of freedom

e(rss) residual sum of squares

e(df_r) residual degrees of freedom

e(r2) R-squared

e(r2_a) adjusted R-squared

e(F) F statistic

e(rmse) root mean squared error

e(ll) log likelihood under additional assumption of i.i.d. normal errors

e(ll_0) log likelihood, constant-only model

e(N_clust) number of clusters

e(b) coefficient vector

e(V) variance-covariance matrix of the estimators

A more complete list of values is available in the manual

The following example is a continuation of the analysis in the previous regressions and illustrates illustrates the use of these variables to calculate NR^2.

```
. ereturn list

scalars:
                  e(N) =  46
               e(df_m) =  2
               e(df_r) =  43
                  e(F) =  9.378104068965701
                 e(r2) =  .3037137269833624
               e(rmse) =  .1634329909032695
                e(mss) =  .5009847436578099
                e(rss) =  1.148544728170291
               e(r2_a) =  .2713283189360769
                 e(ll) =  19.60217878514291
               e(ll_0) =  11.27630778276544

macros:
            e(cmdline) : "regress lnc lnp lny"
              e(title) : "Linear regression"
                e(vce) : "ols"
             e(depvar) : "lnc"
                e(cmd) : "regress"
         e(properties) : "b V"
            e(predict) : "regres_p"
              e(model) : "ols"
          e(estat_cmd) : "regress_estat"

matrices:
                  e(b) :  1 x 3
                  e(V) :  3 x 3
functions:
             e(sample)
```

We may use these items in many ways. A simple example follows.

```
. scalar nobs=e(N)

. scalar rsq = e(r2)

. display nobs * rsq
13.970831
```

We may also retrieve the estimated coefficients in another way which may be more convenient. The coefficient of `lnp` is available as `_b[lnp]` and its standard deviation as `_se[lnp]`. A similar result holds for the other coefficients (For the constant term use `_b[_cons]` or `_se[_cons]` as required.). Thus the fitted values of tobacco consumption from the regression above one could use the following expression.

```
. generate lnchat =  lnp*\_b[lnp]+lny*\_b[lny]+\_b[\_cons]
```

One could then calculate the residuals in the regression as `lnc - lnchat`. This approach can be use to calculate various functions or the coefficients and the data values. See, for example, subsection 11.2 on page 36.

11.2 Use of factor and dummy variables in estimation commands

Stata Version 11 contains enhanced features for using factor and dummy variables in estimation and post-estimation commands.[23]. These facilities are described in detail in Section 11.4.3 of the Stata User's Guide, Release 11. Examples of factor variables are given in subsection 9.3 on page 29 of this document . Standard dummy variables may be regarded as two level factors and there are advantages to treating them as two level factors in Stata 11.

In subsection 9.3 `countryfac` was a factor with 4 levels (France 1), (Germany 2), (Ireland 3) (UK 4). The inclusion of a variable `i.countryfac` in an estimation command creates 4 virtual dummy variables `1.countryfac` for France, `2.countryfac` for Germany, `3.countryfac` for Ireland and `4.countryfac` for the UK. These dummies are virtual because they only exist during the currency of the current estimation command. They are removed from memory when a new estimation command is issued. By default `1.countryfac` is the base variable and is omitted from the regression and the remaining three are included (to avoid multicollinearity). (It is possible to change the base variable and exclude one of the other dummies from the regression. The Stata manual describes how to do this). The `2.countryfac` dummy is defined as follows.

$$\texttt{2.countrfac} = \begin{cases} 1 & \text{if } \texttt{countryfac} = 2 \\ . & \text{if } \texttt{countryfac} = . \\ 0 & \text{otherwise} \end{cases}$$

with similar definitions for the other three dummies derived from the factor variable.

The small session below gives a simple example of the use of the `countryfac` factor variable in regression analysis. The first regression is an example of how **not** to use a factor variable.[24] The second is correct. Three dummy variable are included and `1.countryfac` is excluded. If we wish to exclude the dummy variable for Ireland for which `countryfac`=3 we would code the estimation command as

```
regress income ib3.countryfac
```

Using `ib3.` instead of `i.` specifies that `3.countryfac`, the third level of the factor is to be taken as the base and that it will be excluded from the estimation.

[23]In previous versions of Stata the prefix command `xi:` provided some of the same features. Where an estimation command does not permit the inclusion of factor variables one may continue to use this prefix command

[24]Any variable that takes a selection of integer values from 0 to 32740 can be interpreted by Stata as a factor. If this integer coding on the factor variable can be interpreted as a a set of values taken from a continuous or near continuous variable it may be possible to use the factor variable in this way. If you do be sure that the result has a sensible interpretation. Dummy variables, which are two level factor variables, are the exception to this rule and can be used in a regression.

```
. encode country, generate(countryfac)
// Do not use factor variables this way unless there is a
// good reason to do so.  Usually there is not
. regress income countryfac

      Source |       SS       df       MS              Number of obs =       8
-------------+------------------------------           F(  1,     6) =    1.29
       Model |   4473.225     1    4473.225           Prob > F      =  0.2997
    Residual |   20837.65     6  3472.94167           R-squared     =  0.1767
-------------+------------------------------           Adj R-squared =  0.0395
       Total |  25310.875     7  3615.83929           Root MSE      =  58.932

------------------------------------------------------------------------------
      income |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  countryfac |     -21.15   18.63583    -1.13   0.300    -66.75023    24.45023
       _cons |     283.75   51.03632     5.56   0.001     158.8686    408.6314
------------------------------------------------------------------------------

// Correct use of factor variable
. regress income i.countryfac

      Source |       SS       df       MS              Number of obs =       8
-------------+------------------------------           F(  3,     4) =    0.77
       Model |   9297.375     3    3099.125           Prob > F      =  0.5661
    Residual |    16013.5     4    4003.375           R-squared     =  0.3673
-------------+------------------------------           Adj R-squared = -0.1072
       Total |  25310.875     7  3615.83929           Root MSE      =  63.272

------------------------------------------------------------------------------
      income |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  countryfac |
          2  |        -84   63.27223    -1.33   0.255    -259.6719    91.67187
          3  |        -78   63.27223    -1.23   0.285    -253.6719    97.67187
          4  |      -72.5   63.27223    -1.15   0.316    -248.1719    103.1719
             |
       _cons |      289.5   44.74022     6.47   0.003     165.2812    413.7188
------------------------------------------------------------------------------

// Changing the base variable
. regress income ib3.countryfac

      Source |       SS       df       MS              Number of obs =       8
-------------+------------------------------           F(  3,     4) =    0.77
       Model |   9297.375     3    3099.125           Prob > F      =  0.5661
    Residual |    16013.5     4    4003.375           R-squared     =  0.3673
-------------+------------------------------           Adj R-squared = -0.1072
       Total |  25310.875     7  3615.83929           Root MSE      =  63.272

------------------------------------------------------------------------------
      income |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  countryfac |
          1  |         78   63.27223     1.23   0.285    -97.67187    253.6719
          2  |         -6   63.27223    -0.09   0.929    -181.6719    169.6719
          4  |        5.5   63.27223     0.09   0.935    -170.1719    181.1719
             |
       _cons |      211.5   44.74022     4.73   0.009     87.28123    335.7188
------------------------------------------------------------------------------
```

Suppose that we generate a dummy variable `male` from the data set in table 3. The

Male	Country
0	France
0	Germany
0	Ireland
0	UK
1	France
1	Germany
1	Ireland
1	UK

Table 4: Interactions of Factors Male and Country

following two regression equations will generate the same output

```
regress income male
regress income i.male
```

It is better to use the second form as Stata 11 then knows that the variable is a factor and later calculations will be completed on a correct basis. If you use the first form Stata 11 will treat the variable as a continuous variable.

Stata 11 also has facilities for dealing with

Factor Interactions Table 4 illustrates the idea of factor interactions. There are 8 possible combinations of factors Male and Country. One can create dummy variables for seven of these to include in a regression. The eighth must be excluded to avoid multicollinearity. We can accomplish this by using the # operator and specifying the regression as

```
regress income i.male\#i.country
```

We can also include levels of one or both of the factors in the regression

```
// levels of male and interactions of male and country
regress income i.male i.male\#i.country
// levels of country and interactions of male and country
regress income i.country i.male\#i.country
// levels of male country and interactions of male and country
regress income i.male i.country i.male\#i.country
// or equivalently using the \#\# operator
regress income i.male\#\#i.country
```

Stata will exclude sufficient variables from the regressions to stop multicollinearity

Slope Dummies There is also a `c.` operator which specifies that a variable is to be treated as continuous. We can use this to generate slope dummies. Consider the instruction

```
regress income i.male#c.age
```

`i.male#c.age` generates two virtual variables. The first is age when male=1 and zero elsewhere. The second is age when male=0 and zero elsewhere.

```
regress income age  i.male#c.age
```

will give the usual estimate of a slope dummy.

Powers of a continuous variable The following code shows how to use the `c.` to include the level, square and cube of a variable

```
regress income c.age c.age#c.age c.age#c.age#c.age
// or equivalently
regress income age c.age#c.age c.age#c.age#c.age
```

In earlier versions of Stata it is necessary to generate new variables for the square and cube of the variable.

The purpose of the new operators is summarised in table 5

operator	description
`i.`	unary operator generating indicators
`c.`	unary operator to treat as continuous
`#`	binary operator generating interactions
`##`	binary operator specifying indicators and interactions

Table 5: Factor Variable Operators

Details of the use of these operators is summarised in table 6. Here `country` is a 3 level factor (1 = US), (2 = UK) and (3 = IE). `sex` is a dummy variable with (0 = Female) and (1 = Male). `arm` is a third factor. The left hand column in the table shows an entry in a Stata estimation instruction. The right hand column is a brief explanation of the extra variables included in the estimation as a result of the list of variables containing this specification.

The dataset `DoctorContacts.csv` illustrate the use of these facilities.[25] The data set contains the following variables.

mdu number of outpatient visits to a medical doctor

lc log(coinsrate+1) where coinsurance rate is 0 to 100

idp individual deductible plan ?

lpi lpi log(annual participation incentive payment) or 0 if no payment

fmde log(max(medical deductible expenditure)) if $IDP = 1$ and $MDE > 1$ or 0 otherwise

physlim physical limitation ?

ndisease number of chronic diseases

health selfâĂŞrate health (excellent,good,fair,poor)

linc log of annual family income (in $)

lfam lfam log of family size

educdec years of schooling of household head

age exact age

[25] The data which were analysed in Cameron and Trivedi (1998) have been sourced from Croissant (2006). The regressions here simply illustrate the use of factors in Stata. The models estimated are probably not appropriate. Some changes have been made to the data set to facilitate its use in Stata.

factor	specification result
`i.country`	Indicators for levels of group
`i.country#i.sex`	indicators for each combination of levels of country and sex, a two-way interaction
`country#sex`	same as `i.country#i.sex`
`country#sex#arm`	indicators for each combination of levels of country, sex, and arm, a three-way interaction
`country##sex`	same as `i.country` `i.sex` `country#sex`
`country##sex##arm`	same as `i.country` `i.sex` `i.arm` `country#sex` `country#arm` `sex#arm` `country#sex#arm`
`sex#c.age`	two variables – `age` for males and 0 elsewhere, and `age` for females and 0 elsewhere; if `age` is also in the model, one of the two virtual variables will be treated as a base
`sex##c.age`	same as `i.sex` `age` `sex#c.age`
`c.age`	same as `age`
`c.age#c.age`	age squared
`c.age#c.age#c.age`	age cubed

Table 6: Examples of Factor Variable Operations

sex sex (male=1,female=0)

child age less than 18 ?

black is household head black

The following annotated Stata session illustrates some of these facilities.

```
. insheet using DoctorContacts.csv, clear
(16 vars, 20186 obs)
```

The variable `health` contains string (character) data and if we try to use it as a factor we get an error message.

```
. regress  mdu lc lpi i.health
health:  may not use factor variable operators on string variables
r(109);
```

To overcome this problem we must create an appropriate factor variable. the new variable healthstatus will be a factor variable having the four factor levels.

```
. label define healthstatus 1 "poor" 2 "fair" 3 "good" 4 "excellent"
. encode health, generate(healthstatus)
. drop health
```

Including `i.healthstatus` includes 3 dummy variables in the regression. Note the omission of the dummy for `healthstatus` 1.

```
. regress  mdu lc lpi i.healthstatus

      Source |       SS       df       MS              Number of obs =   20186
-------------+------------------------------           F(  5, 20180) =  100.19
       Model |  9921.79616     5  1984.35923           Prob > F      =  0.0000
    Residual |   399690.48 20180  19.8062676           R-squared     =  0.0242
-------------+------------------------------           Adj R-squared =  0.0240
```

```
       Total |  409612.276 20185  20.2929044           Root MSE      =  4.4504

------------------------------------------------------------------------------
         mdu |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          lc |  -.2864245    .017051   -16.80   0.000    -.3198459   -.2530031
         lpi |   .0708933   .0129004     5.50   0.000     .0456076    .0961791
             |
healthstatus |
          2  |  -2.040985   .2798119    -7.29   0.000    -2.589439   -1.492531
          3  |  -2.845158   .2613625   -10.89   0.000     -3.35745   -2.332866
          4  |  -3.116086   .2596252   -12.00   0.000    -3.624973     -2.6072
             |
       _cons |   6.097526   .2613922    23.33   0.000     5.585176    6.609876
------------------------------------------------------------------------------
```

Including `i.healthstatus#i.sex` includes dummies for sufficient combinations of health status and sex (interactions) in the regression. The interaction of `healthstatus` 1 and `sex` 0 is treated as base and excluded from the regression.

```
. regress  mdu lc lpi i.healthstatus#i.sex

      Source |       SS       df       MS              Number of obs =   20186
-------------+------------------------------           F(  9, 20176) =   78.92
       Model |  13929.1813     9  1547.68682           Prob > F      =  0.0000
    Residual |  395683.094 20176  19.6115729           R-squared     =  0.0340
-------------+------------------------------           Adj R-squared =  0.0336
       Total |  409612.276 20185  20.2929044           Root MSE      =  4.4285

------------------------------------------------------------------------------
         mdu |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          lc |  -.2879081   .0169689   -16.97   0.000    -.3211685   -.2546477
         lpi |   .0722814   .0128405     5.63   0.000     .0471129    .0974499
             |
healthstatus#|
         sex |
        1 1  |  -1.261347   .5402286    -2.33   0.020    -2.320239   -.2024544
        2 0  |  -1.622405   .3452315    -4.70   0.000    -2.299087   -.9457235
        2 1  |  -3.632816   .3573449   -10.17   0.000    -4.333241   -2.932391
        3 0  |  -2.819469   .3201336    -8.81   0.000    -3.446956   -2.191981
        3 1  |  -3.817742   .3218488   -11.86   0.000    -4.448592   -3.186892
        4 0  |  -3.302574   .3183127   -10.38   0.000    -3.926493   -2.678655
        4 1  |  -3.756018   .3179017   -11.82   0.000    -4.379132   -3.132905
             |
       _cons |   6.516455   .3162552    20.61   0.000     5.896569    7.136341
------------------------------------------------------------------------------
```

Including `i.healthstatus##i.sex` inserts, in the regression, dummies for levels of `healthstatus`, `sex` and (independent) combinations of `healthstatus` and `sex`. Again not the omission of base dummies.

```
. regress  mdu lc lpi i.healthstatus##i.sex

      Source |       SS       df       MS              Number of obs =   20186
-------------+------------------------------           F(  9, 20176) =   78.92
       Model |  13929.1813     9  1547.68682           Prob > F      =  0.0000
    Residual |  395683.094 20176  19.6115729           R-squared     =  0.0340
-------------+------------------------------           Adj R-squared =  0.0336
```

```
       Total |  409612.276 20185  20.2929044           Root MSE      =  4.4285

------------------------------------------------------------------------------
         mdu |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          lc |  -.2879081   .0169689   -16.97   0.000    -.3211685   -.2546477
         lpi |   .0722814   .0128405     5.63   0.000     .0471129    .0974499
             |
healthstatus |
          2  |  -1.622405   .3452315    -4.70   0.000    -2.299087   -.9457235
          3  |  -2.819469   .3201336    -8.81   0.000    -3.446956   -2.191981
          4  |  -3.302574   .3183127   -10.38   0.000    -3.926493   -2.678655
             |
       1.sex |  -1.261347   .5402286    -2.33   0.020    -2.320239   -.2024544
             |
healthstatus#|
         sex |
        2 1  |  -.7490643   .5860848    -1.28   0.201    -1.897838    .3997098
        3 1  |   .2630729   .5501945     0.48   0.633    -.8153533    1.341499
        4 1  |   .8079019   .5467734     1.48   0.140    -.2638186    1.879622
             |
       _cons |   6.516455   .3162552    20.61   0.000     5.896569    7.136341
------------------------------------------------------------------------------
```

The following extract shows how to add slope dummies for age to the regression equation. The coefficients of the two slope dummies may be regarded as the marginal effects of age for females and males respectively.

```
. regress  mdu lc lpi i.sex#c.age

      Source |       SS       df       MS              Number of obs =   20186
-------------+------------------------------           F(  4, 20181) =  184.42
       Model |   14444.674     4  3611.16851           Prob > F      =  0.0000
    Residual |  395167.602 20181  19.5811705           R-squared     =  0.0353
-------------+------------------------------           Adj R-squared =  0.0351
       Total |  409612.276 20185  20.2929044           Root MSE      =  4.4251

------------------------------------------------------------------------------
         mdu |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          lc |  -.2816667   .0169591   -16.61   0.000    -.3149079   -.2484254
         lpi |   .0624095    .012832     4.86   0.000     .0372576    .0875613
             |
   sex#c.age |
          0  |   .0393779   .0020611    19.10   0.000     .0353379    .0434179
          1  |   .0059746   .0021846     2.73   0.006     .0016927    .0102565
             |
       _cons |   2.625066   .0793758    33.07   0.000     2.469483    2.780649
------------------------------------------------------------------------------
```

The next estimate includes the square and cube of age.

```
. regress  mdu lc lpi i.sex c.age c.age#c.age c.age#c.age#c.age

      Source |       SS       df       MS              Number of obs =   20186
-------------+------------------------------           F(  6, 20179) =  167.30
       Model |   19410.932     6  3235.15533           Prob > F      =  0.0000
    Residual |  390201.344 20179   19.337001           R-squared     =  0.0474
-------------+------------------------------           Adj R-squared =  0.0471
```

```
      Total |  409612.276 20185  20.2929044           Root MSE      =  4.3974

------------------------------------------------------------------------------
         mdu |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          lc |   -.274803   .0168603   -16.30   0.000    -.3078505   -.2417555
         lpi |   .0545234   .0127586     4.27   0.000     .0295156    .0795312
       1.sex |  -.8102347   .0620382   -13.06   0.000    -.9318346   -.6886347
         age |  -.2402699   .0167146   -14.37   0.000     -.273032   -.2075079
             |
 c.age#c.age |   .0084894   .0006508    13.04   0.000     .0072138    .0097651
             |
 c.age#c.age#|
       c.age |  -.0000742   7.08e-06   -10.47   0.000     -.000088   -.0000603
             |
       _cons |   4.841772   .1317025    36.76   0.000     4.583624    5.099919
------------------------------------------------------------------------------

------------------------------------------------------------------------------

.
end of do-file
```

11.3 Constrained Estimation

Constrained estimation may be completed in two ways. Consider the following equation

$$y = \beta_1 X_1 + \beta_2 X_2 + \boldsymbol{\gamma}'\boldsymbol{z} + \varepsilon \qquad (1)$$

where, for example, y is expenditure on some item, X_1 and X_2 are income from sources 1 and 2, $\boldsymbol{z}$ is a vector of other exogenous variables and ε is a disturbance term. Economic theory might suggest that the coefficients on X_1 and X_2 are equal and we are required to estimate this single value. Put $\beta = \beta_1 = \beta_2$ and substitute into equation 1 to get

$$y = \beta(X_1 + X_2) + \boldsymbol{\gamma z} + \varepsilon$$

If all relevant conditions are satisfied then β can be estimated by creating a new variable $X = X_1 + X_2$ and estimating the regression

$$y = \beta X + \boldsymbol{\gamma z} + \varepsilon \qquad (2)$$

β may now be estimated by a Stata command such as

```
regress y X z
```

Many constrained parameters may be estimated in this way. The method has the advantage of allowing one to see clearly what one is doing. If one is dealing with multiple restrictions it can be a little long-winded. Stata does provide two commands to estimate such restricted coefficients. (It should be noted that when restrictions are placed on some coefficients, the estimates of the unrestricted coefficients may also change relative to their unrestricted estimates.)

The second method of estimating restricted coefficients in a linear regression involves two Stata commands.

1. `constraint` which allows us to specify the constraints, and

2. `cnsreg` which allows us to estimate the system subject to the constraints.

A constraint is defined using the following syntax.

```
constraint # constraint
```

where # is a number from 1 to 1999 and *constraint* is a constraint specified as the corresponding *test* in `test` command (See subsection 12.2 on page 46). In the example in this subsection the constraint would be specified as follows.

```
constraint 1 X1 = X2
```

The constrained equation would then be estimates with a command such as

```
cnsreg y X1 X2 z, constraints(1)
```

The constraints option is obligatory. If other constraints nave been defined the argument of the constraints option may include a list of the constraints to be active during the current estimation.

12 Post estimation commands after `regress`

12.1 Lists

Post estimation facilities have been greatly expanded in recent versions of Stata. Here I give a list of the commands currently available and their function. I then give some brief examples of the use of the commands which are of greater interest in econometrics. Statistical tests which are not available in the list below can usually be calculated from the e-variables which we have described in the previous section. I would recommend that you might profit by also repeating the calculation of some of the test results available from these commands using the e-variables. Such practise would add to your understanding of the tests.

The following post estimation commands are of special interest after regress:[26]

dfbeta DFBETA influence statistics.

estat hettest tests for heteroskedasticity.

estat imtest information matrix test.

estat ovtest Ramsey regression specification-error test for omitted variables or functional form.

estat szroeter Szroeter's rank test for heteroskedasticity.

estat vif variance inflation factors for the independent variables.

acprplot augmented component-plus-residual plot.

avplot added-variable plot.

avplots all added-variable plots in one image.

[26]These commands are not appropriate after the svy prefix see manual and help files.

cprplot component-plus-residual plot.

lvr2plot leverage-versus-squared-residual plot.

rvfplot residual-versus-fitted plot.

rvpplot residual-versus-predictor plot.

The following standard post-estimation apply to many estimation commands although their syntax, capabilities and defaults may vary from command to command (see help files and manual)..

adjust adjusted predictions of xb.

estat AIC, BIC, VCE, and estimation sample summary.

estat (svy) post-estimation statistics for survey data.

estimates cataloguing estimation results.

hausman Hausman's specification test.

lincom point estimates, standard errors, testing, and inference for linear combinations of coefficients.

linktest link test for model specification (not available with svy estimation results).

lrtest likelihood-ratio test (not available with svy estimation results).

mfx marginal effects or elasticities.

nlcom point estimates, standard errors, testing, and inference for non-linear combinations of coefficients.

predict predictions, residuals, influence statistics, and other diagnostic measures.

predictnl point estimates, standard errors, testing, and inference for generalized predictions.

suest seemingly unrelated estimation.

test Wald tests for simple and composite linear hypotheses.

testnl Wald tests of non-linear hypotheses.

When dealing with time series data you will also have the following post regression commands available:

estat archlm test for ARCH effects in the residuals

estat bgodfrey Breusch-Godfrey test for higher-order serial correlation

estat durbinalt Durbin's alternative test for serial correlation

estat dwatson Durbin-Watson dw statistic to test for first-order serial correlation

12.2 Wald tests – `test` and `testnl`

The output of the `regress` command includes a Wald F–test for the significance of the regression and t–statistics for the significance of the individual coefficients. The `test` command may be used to test joint hypotheses on the coefficients.

The examples below show how to test

1. The joint significance or the price and income coefficients in the regression estimated above
2. That the coefficients are equal
3. That the coefficients sum to 1
4. That the coefficients sum to -1

These tests are for illustration only to show how such a test would be implemented. They may not make economic sense in this particular case

```
. test lnp lny
 ( 1)  lnp = 0
 ( 2)  lny = 0
      F(  2,    43) =    9.38
           Prob > F =    0.0004

. test lnp=lny
 ( 1)  lnp - lny = 0
      F(  1,    43) =   11.03
           Prob > F =    0.0018

. test lnp+lny=1
 ( 1)  lnp + lny = 1
      F(  1,    43) =   57.78
           Prob > F =    0.0000

. test lnp+lny=-1
 ( 1)  lnp + lny = -1
      F(  1,    43) =    0.34
                  Prob > F =    0.5633
```

`testnl` will carry out tests of non-linear hypotheses as in the following example.[27]

```
. testnl _b[lnp]*_b[lny]/_b[_cons] = -.05

  (1)  _b[lnp]*_b[lny]/_b[_cons] = -.05

              F(1, 43) =       0.00
              Prob > F =       0.9636
```

12.3 Tests for heteroskedasticity

Three of the post regression commands are tests of heteroskedasticity – `esat hettest`, `estat imtest` and `estat szroeter`. The first two are illustrated in the extract below: For details of the theory underlying `estat szroeter` see Judge et al. (1985) and the Stata manuals.

[27]The Wald test may give different results if different but equivalent non-linear functional forms of a hypothesis are used.

```
  . use cigarett.dta, clear

. // Estimate basic regression
. regress  lnc lnp lny

      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  2,    43) =    9.38
       Model |  .500984744     2  .250492372           Prob > F      =  0.0004
    Residual |  1.14854473    43  .026710343           R-squared     =  0.3037
-------------+------------------------------           Adj R-squared =  0.2713
       Total |  1.64952947    45   .03665621           Root MSE      =  .16343

------------------------------------------------------------------------------
         lnc |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |  -1.338335   .3246015    -4.12   0.000    -1.992957    -.683714
         lny |   .1723862   .1967544     0.88   0.386    -.2244069    .5691793
       _cons |   4.299661   .9089257     4.73   0.000     2.466637    6.132684
------------------------------------------------------------------------------

. // Breusch-Godfrey
. estat hettest  lnp lny, mtest

Breusch-Pagan / Cook-Weisberg test for heteroskedasticity
         Ho: Constant variance

---------------------------------------
    Variable |      chi2   df      p
-------------+-------------------------
         lnp |      0.18    1   0.6696 #
         lny |      5.49    1   0.0192 #
-------------+-------------------------
simultaneous |      6.18    2   0.0455
---------------------------------------
                  # unadjusted p-values
```

By using the `mtest` option we have asked Stata to examine the effect of each regressor on heteroskedasticity. One can reject the hypothesis of no heteroskedasticity in the residuals. There are indications that the heteroskedasticity is related to `lny` and not to `lnp`.

```
. // White's Test
. estat imtest, preserve white

White's test for Ho: homoskedasticity
         against Ha: unrestricted heteroskedasticity

         chi2(5)      =     15.66
         Prob > chi2  =    0.0079

Cameron & Trivedi's decomposition of IM-test

---------------------------------------------------
              Source |       chi2     df      p
---------------------+-----------------------------
  Heteroskedasticity |      15.66      5    0.0079
            Skewness |       3.59      2    0.1663
            Kurtosis |       0.06      1    0.8029
---------------------+-----------------------------
               Total |      19.31      8    0.0133
---------------------------------------------------
```

White's test statistic is given in the first line of the table above as generated by Stata. The test statistic indicates that the hypothesis of no heteroskedasticity can be rejected. The next extract shows how Whites test statistic may be calculated directly from data stored during a regression.

```
. // Whites Test from first principles
. // generate required variables
. // (this can also be achieved using the predict command)
. generate lnchat =  lnp*_b[lnp] + lny*_b[lny] + _b[_cons]

. generate uhat = lnchat - lnc

. generate uhatsq =uhat * uhat

. generate lnysq = lny * lny

. generate lnpsq = lnp*lnp

. generate lnplny = lnp * lny

. // supplementary regression
. regress uhatsq lnp lny lnpsq lnysq lnplny

      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  5,    40) =    4.13
       Model |  .018301238     5  .003660248           Prob > F      =  0.0041
    Residual |  .035469298    40  .000886732           R-squared     =  0.3404
-------------+------------------------------           Adj R-squared =  0.2579
       Total |  .053770536    45  .001194901           Root MSE      =  .02978

------------------------------------------------------------------------------
      uhatsq |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |   9.505993   3.302553     2.88   0.006     2.831285     16.1807
         lny |  -7.893122   2.329365    -3.39   0.002    -12.60094     -3.1853
       lnpsq |   1.281125   .6562051     1.95   0.058    -.0451149    2.607365
       lnysq |   .8557196   .2530459     3.38   0.002     .3442948    1.367144
      lnplny |   -2.07862   .7275194    -2.86   0.007    -3.548992   -.6082484
       _cons |   18.22186    5.37401     3.39   0.002     7.360579    29.08314
------------------------------------------------------------------------------

. // Calculate test Statistic
. display e(N) * e(r2)
15.656473

. display chi2(e(N) * e(r2),5)
.00526644
```

12.4 The `estat ovtest` command

The `estat ovtest` command compiles Ramsey **R**egression **E**quation **S**pecification **E**rror **T**est (RESET) for omitted variables or functional form. It is a very general test and indicates that there is something wrong with the form of the equation or perhaps the underlying theory. It is **not** correct to use a failing RESET test to add variables to a model. Why was the variable not taken into account in the first place?

```
. use cigarett.dta, clear

. regress  lnc lnp lny
```

```
      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  2,    43) =    9.38
       Model |  .500984744     2  .250492372           Prob > F      =  0.0004
    Residual |  1.14854473    43  .026710343           R-squared     =  0.3037
-------------+------------------------------           Adj R-squared =  0.2713
       Total |  1.64952947    45   .03665621           Root MSE      =  .16343

------------------------------------------------------------------------------
         lnc |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |  -1.338335   .3246015    -4.12   0.000    -1.992957    -.683714
         lny |   .1723862   .1967544     0.88   0.386    -.2244069    .5691793
       _cons |   4.299661   .9089257     4.73   0.000     2.466637    6.132684
------------------------------------------------------------------------------

. // Standard RESET test as implemented in Stata -
. // uses 3 powers of fitted var
. estat ovtest

Ramsey RESET test using powers of the fitted values of lnc
       Ho:  model has no omitted variables
                  F(3, 40) =      3.11
                  Prob > F =      0.0369

. // Alternative using powers of rhs variables
. estat ovtest, rhs

Ramsey RESET test using powers of the independent variables
       Ho:  model has no omitted variables
                  F(6, 37) =      2.35
                  Prob > F =      0.0508
```

It is instructive to compute the test statistic directly. If the sample is small we might like to useless powers of the fitted variable in the test regression. Davidson and Mackinnon (2004) recommends that only one be used.

```
. // Store R-squared from regression for later work
. scalar r1sq = e(r2)

. // Replication
. // Try with 3 powers
. predict lnchat, xb

. generate lnchat2 = lnchat^2

. generate lnchat3 = lnchat2*lnchat

. generate lnchat4 = lnchat3*lnchat

. regress lnc lnp lny lnchat2 lnchat3 lnchat4

      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  4,    41) =    6.05
       Model |  .612011449     4  .153002862           Prob > F      =  0.0006
    Residual |  1.03751802    41  .025305318           R-squared     =  0.3710
-------------+------------------------------           Adj R-squared =  0.3097
       Total |  1.64952947    45   .03665621           Root MSE      =  .15908

------------------------------------------------------------------------------
```

```
         lnc |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |  (dropped)
         lny |  -.0244152   .1771587    -0.14   0.891    -.3821944     .333364
     lnchat2 |   78.13769   76.93193     1.02   0.316    -77.22949    233.5049
     lnchat3 |  -21.65652   20.97913    -1.03   0.308    -64.02472    20.71168
     lnchat4 |   1.689341   1.608436     1.05   0.300    -1.558961    4.937643
       _cons |  -297.1078   306.2742    -0.97   0.338     -915.641    321.4254
------------------------------------------------------------------------------

. // This is not working
. //
. summarize

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         obs |        46        23.5    13.42262          1         46
       state |         0
         lnc |        46    4.847844    .1914581    4.40859    5.37906
         lnp |        46    .2055087    .0862304     -.0326     .36399
         lny |        46    4.775455    .1422612    4.52938    5.10268
-------------+--------------------------------------------------------
      lnchat |        46    4.847844     .105513   4.654018   5.144778
     lnchat2 |        46    23.51248    1.025989   21.65988   26.46874
     lnchat3 |        46    114.0907    7.487367   100.8055   136.1758
     lnchat4 |        46    553.8665    48.60212   469.1505   700.5941

. //
. // There are numerical problems
```

We can solve these by rescaling the powers of the fitted variables. You should convince yourself that this has no effect on the significance tests.

```
. //
. summarize lnchat

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
      lnchat |        46    4.847844     .105513   4.654018   5.144778

. // Different variables are of different orders of magnitude
. replace lnchat = (lnchat-r(min))/(r(max)-r(min))
(46 real changes made)

. replace lnchat2 = lnchat^2
(46 real changes made)

. replace lnchat3 = lnchat2*lnchat
(46 real changes made)

. replace lnchat4 = lnchat3*lnchat
(46 real changes made)

. regress lnc lnp lny lnchat2 lnchat3 lnchat4

      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  5,    40) =    6.17
       Model |  .718299063     5  .143659813           Prob > F      =  0.0003
    Residual |  .931230408    40   .02328076           R-squared     =  0.4355
-------------+------------------------------           Adj R-squared =  0.3649
```

```
       Total |  1.64952947    45   .03665621           Root MSE      =  .15258

------------------------------------------------------------------------------
         lnc |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |  -9.554136   4.164306    -2.29   0.027    -17.97051    -1.13776
         lny |   1.116497   .5417511     2.06   0.046     .0215772    2.211417
     lnchat2 |  -15.26242   6.815871    -2.24   0.031    -29.03781   -1.487029
     lnchat3 |   24.33426   10.89703     2.23   0.031     2.310545    46.35798
     lnchat4 |  -11.79374   5.567685    -2.12   0.040    -23.04645   -.5410258
       _cons |   2.590192   1.371201     1.89   0.066    -.1811075    5.361492
------------------------------------------------------------------------------

. test lnchat2 lnchat3 lnchat4

 ( 1)  lnchat2 = 0
 ( 2)  lnchat3 = 0
 ( 3)  lnchat4 = 0

       F(  3,    40) =    3.11
            Prob > F =    0.0369
```

This is now working and gives the same result as the basic Stata routine. We give an alternative formulation of the test statistic which is more robust with respect to distributional assumptions.

```
. scalar r2sq = e(r2)

. scalar Fstat = ((r2sq-r1sq)/3)/((1-r2sq)/(e(N)-e(df_m)))

. display "F-statistic is " Fstat
F-statistic is 3.1892884

. display "Probably of a greater value is " Ftail(3,40,Fstat)
Probability of a greater value is .03382032

. // Similar result
```

One can not change the default 3 powers of the fitted variable used in the built in RESET test in Stata. If the sample is small you may wish to reduce the number of powers of the fitted variable used in the test. For example, to use only the square one must code directly as here and the t-statistic on the fit squared may be used as a test statistic.

```
. regress lnc lnp lny lnchat2

      Source |       SS       df       MS              Number of obs =      46
-------------+------------------------------           F(  3,    42) =    7.6
       Model |   .58210924     3  .194036413           Prob > F      =  0.0003
    Residual |  1.06742023    42  .025414767           R-squared     =  0.3529
-------------+------------------------------           Adj R-squared =  0.3067
       Total |  1.64952947    45   .03665621           Root MSE      =  .15942

------------------------------------------------------------------------------
         lnc |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         lnp |   .2613511   .9497055     0.28   0.785    -1.655232    2.177934
         lny |  -.0740625   .2363521    -0.31   0.756    -.5510404    .4029153
     lnchat2 |   .6844432   .3830932     1.79   0.081    -.0886702    1.457557
       _cons |   5.010102   .9716972     5.16   0.000     3.049138    6.971067
```

```
------------------------------------------------------------------------------
.
.
.
\\ The t-statistic is highly significant indicating an omitted variable or
\\ functional form problem.


. exit, clear
```

As one can see, there are a variety of RESET tests. On occasion these may give contradictory results. I am sometimes asked which result is correct. My usual reaction is that the test which is rejecting the null of correct functional form may be indicative of a problem that the other test is not detecting and that this should be taken into account in your conclusion. If the test requiring the testing of more coefficients accepts the null and the "smaller" test rejects and the sample is not very large then I might favour the "smaller" test.

12.5 The `predict` command

After a regression or other estimation command the `predict` command allows one to estimate various other related series. After a regression the series which may be estimated include:

xb linear prediction; the default.

residuals residuals.

score score; equivalent to residuals.

rstandard standardised residuals.

rstudent studentised (jackknifed) residuals.

cooksd Cook's distance.

leverage | hat leverage (diagonal elements of hat matrix).

pr(a,b) $\Pr(y|a < y < b)$.

e(a,b) $E(y|a < y < b)$.

ystar(a,b) $E(y*), y* = \max(a, \min(y, b)$.

* **dfbeta(varname)** DFBETA for varname.

stdp standard error of the linear prediction.

stdf standard error of the forecast.

stdr standard error of the residual.

* **covratio** COVRATIO.

* **dfits** DFITS.

* **welsch** Welsch distance.

Notes

- Unstarred statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.
- Starred statistics are calculated only for the estimation sample, even when if `(sample)` is not specified.
- `rstandard`, `rstudent`, `cooksd`, `leverage`, `dfbeta()`, `stdf`, `stdr`, `covratio`, `dfits`, and `welsch` are not available if any `vce()` other than `vce(ols)` was specified with `regress`.
- `xb`, `residuals`, `score`, and `stdp` are the only options allowed with `svy` estimation results.

where a and b may be numbers or variables; a missing ($a > .$) means minus infinity, and b missing ($b > .$) means plus infinity;

A simplified version of the syntax of the `predict` command is

```
 predict  newvarname ,what
```

where `what` is the item to be calculated taken from the above list and `newvarname`. The following code

```
predict lnchat, xb
predict uhat, residuals
```

calculates and stores the fitted values and residuals in variables `lnchat` and `uhat`.

12.6 Marginal Effects – The `contrasts`[28]command

The `contrasts` command is new in Stata version 11. It is a greatly extended version of the `mfx` command which was available in earlier versions of Stata. The Stata on-line help states

[28]If you are using Stata version 10 or earlier, you can use the `mfx` command to estimate marginal effects. This command does not calculate the average of the marginal effect over the sample. If you wish to calculate this you must use an analytic or numerical method similar to that described in the main text of this subsection. A simplified version of the syntax of the `mfx` command is

```
  mfx, predict() option
```

where the meaning of `predict()` and main options are

predict() Specifies what variable is to be predicted for the purpose of calculating derivatives and elasticities. This can be any variable that can be "predicted" by the `predict` command. If this is omitted the default is the default of the `predict` command of the current estimation command.

dydx Calculates the marginal effects ($\frac{\partial y}{\partial x}$) . In an OLS estimate these are the same as the estimated coefficients.

eyex Calculates elasticities in the form $\frac{\partial \log y}{\partial \log x}$.

eydx Calculates elasticities in the form $\frac{\partial \log y}{\partial x}$

dyex Calculates elasticities in the form $\frac{\partial y}{\partial \log x}$.

at() By default the elasticities are calculated at the mean of the independent variables. This allows them to be calculated at other values of these variables. See the help files and Manuals.

`mfx` has been superseded by `margins`. `margins` can do everything that `mfx` did and more. `margins` syntax differs from `mfx`; see `margins`. `mfx` continues to work but does not support factor variables and will often fail if you do not run your estimation command under version control, with the version set to less than 11. This help file remains to assist those who encounter an mfx command in old do-files and programs.

The Stata 11 manual does not contain any reference to the `mfx` command. We suspect that there are occasions that, unless unless version control, as specified, is in place `mfx` may produce wrong estimates of an elasticity and it will not issue any warning that something may be wrong. **Do not use `mfx` in Stata Version 11 unless you need to use a program written for an earlier version of Stata and you are using version control**

Not all the facilities provided by the `margins` command are of great use to econometricians. Here I will concentrate on those that are. If we write our model as

$$y = f(x, y, u)$$

where y and x are continuous variables. `margins` can estimate numerically the following marginal effects,semi-elasticities and elasticities of y of with respect to x.

$$\frac{\partial y}{\partial x}, \ \frac{\partial \log y}{\partial x}, \ \frac{\partial y}{\partial \log x}, \text{ and } \frac{\partial \log y}{\partial \log x}$$

These effects may be calculated for

- each observation and then averaged over the sample
- the average value of the right hand side variables
- specified values of the right hand side variables
- certain subsamples of the data

Students often ask which one of these is the best to use. To some extent each is answering a different question and the one to use is the one that answers your question. The first option is probably the most popular but is probably not always the best option.

If one uses the `c.` and `#` operators (see subsections 9.3 and 11.2 on pages 29 and 36 respectively) to insert polynomials in a continuous variable or interactions with a factor, `margins` will calculate the correct value of the marginal effect of the continuous variable .

If one uses the `i.` for a factor variable in an estimation command the correct discrete marginal effects are calculated. If a factor variable is included in an estimation and the `i.` notation is not used Stata will assume that the variable is continuous for the purpose of calculating these effects.

Continuing the example at the end of 11.2 we estimate the following equation

$$mdu = \beta_1 lc + \beta_2 lpi + \beta_3 sex + \beta_4 age + \beta_5 (age)^2 + \beta_6 (age)^3 I_{noChild} + \beta_7 (age)^3 I_{child} \quad (3)$$

where $I_{noChild}$ and I_{child} are dummies for families with no children and with children, respectively.

```
. regress  mdu lc lpi i.sex c.age c.age#c.age c.age#c.age#c#c.age

      Source |       SS       df       MS              Number of obs =   20186
```

```
-------------+------------------------------           F(  7, 20178) =  143.40
       Model |   19410.987     7  2772.99814           Prob > F      =  0.0000
    Residual |  390201.289 20178  19.3379566           R-squared     =  0.0474
-------------+------------------------------           Adj R-squared =  0.0471
       Total |  409612.276 20185  20.2929044           Root MSE      =  4.3975

------------------------------------------------------------------------------
         mdu |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          lc |  -.2747781   .0168671   -16.29   0.000     -.307839   -.2417172
         lpi |   .0544913   .0127731     4.27   0.000      .029455    .0795275
       1.sex |  -.8102682   .0620429   -13.06   0.000    -.9318774   -.6886589
         age |   -.240707   .0186155   -12.93   0.000    -.2771948   -.2042191
             |
 c.age#c.age |   .0085082     .00074    11.50   0.000     .0070578    .0099587
             |
 child#c.age#|
  c.age#c.age|
        0    |  -.0000744   8.00e-06    -9.29   0.000      -.00009   -.0000587
        1    |  -.0000728   .0000264    -2.76   0.006    -.0001245   -.0000211
             |
       _cons |   4.842552   .1325152    36.54   0.000     4.582811    5.102292
------------------------------------------------------------------------------
```

The `margins` command as used here has the following format. (Further options are described in the manual and help files).

```
margins, options
```

The most important options are

dydx(`varlist`) Calculates elasticities in the form $\frac{\partial y}{\partial x}$ with respect to the variables in `varlist`.

eydx(`varlist`) Calculates elasticities in the form $\frac{\partial \log y}{\partial x}$ with respect to the variables in `varlist`.

dyex(`varlist`) Calculates elasticities in the form $\frac{\partial y}{\partial \log x}$ with respect to the variables in `varlist`.

dydx(`varlist`) Calculates elasticities in the form $\frac{\partial \log y}{\partial \log x}$ with respect to the variables in `varlist`.

predictvalue where `predicted` is one of the options that can be predicted by the `predict` command. See section 12.5.

at() At specified values of the independent variables. The default is to calculate the marginal effect for each observation and to average over the sample.

atmeans At the mean value of the right hand side variables.

In equation 3 age enters in a non-linear form. The following command calculates the marginal effect of `age` on `mdu`.

```
. margins, dydx(age)

Average marginal effects                          Number of obs   =      20186
Model VCE    : OLS

Expression   : Linear prediction, predict()
```

```
dy/dx w.r.t. : age

------------------------------------------------------------------------------
             |            Delta-method
             |      dy/dx   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |  -.0131043   .0045265    -2.90   0.004    -.0219761   -.0042326
------------------------------------------------------------------------------
```

Analytically the marginal effect is

$$\frac{\partial mdu}{\partial age} = \beta_4 + 2\beta_5(age) + 3\beta_6(age)^2 I_{noChild} + 3\beta_7(age)^2 I_{child}.$$

We may use `generate` to estimate this quantity for each person in the sample and then `summarize` to calculate the average effect.

```
. //
. generate  m1 = _b[age] + 2 * _b[c.age#c.age] * age +  ///
>        3 * _b[0.child#c.age#c.age#c.age] * age^2 *0.child + ///
>        3 * _b[1.child#c.age#c.age#c.age] * age^2 *1.child

. summarize m1
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
          m1 |     20186   -.0131043    .0913563   -.240707   .0838344
```

Note that the answer is the same in both cases. Alternatively we may estimate the derivative numerically using the approximation

$$\lim_{h\to 0} \frac{f(x+h,z,u) - f(x,z,u)}{h} = \frac{\partial y}{\partial x} \tag{4}$$

We implement this numerical routine by choosing a small h, (`delta` in the program segment. Estimate the finite difference for each member in the sample.[29] Note that we get the same answer as above.

```
. scalar delta = .00001
. generate m2 = ( (_b[lc]*lc + _b[lpi]*lpi + _b[1.sex] * 1.sex + ///
>                _b[age]*(age+delta) + _b[c.age#c.age]*(age+delta)^2 + ///
>                _b[0.child#c.age#c.age#c.age]*0.child*(age+delta)^3 + ///
>                _b[1.child#c.age#c.age#c.age]*1.child*(age+delta)^3 ) - ///
>        (_b[lc]*lc + _b[lpi]*lpi + _b[1.sex] * 1.sex + ///
>                _b[age]*(age) + _b[c.age#c.age]*(age)^2 + ///
>                _b[0.child#c.age#c.age#c.age]*0.child*(age)^3 + ///
>                _b[1.child#c.age#c.age#c.age]*1.child*(age)^3 ) )/delta

. summarize m2

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
          m2 |     20186   -.0131043    .0913563  -.2407069   .0838344
```

[29]For serious numerical calculation of derivatives it is generally better to estimate the derivative using the formula

$$\lim_{h\to 0} \frac{f(x+h,z,u) - f(x-h,z,u)}{2h} = \frac{\partial y}{\partial x}$$

as this converges to the derivative much faster. In this case equation 4 is sufficient.

The analytic and numerical methods, used to check and illustrate the calculation of marginal effects, are useful in many circumstances. You may find that computer algebra systems such as

- Mathematica (`http://www.wolfram.com`),
- Maple (`http://www.maplesoft.com`), or
- Maxima (`http://maxima.sourceforge.net`)

helpful in deriving or checking required analytic expressions.

12.7 Processing estimation results

The results, including coefficients, variances, test statistics, e() variables etc. of a `regress` or any other estimation command are held in memory until another estimation command is run. When that other estimation command is run all of the results of the first estimation command are replaced, in active memory, by the results of the new command. The command `estimates store` *`model1`*, issued after the first command and before the second, stores the results of the first estimation in memory. If we then run another estimation command and save the results as *`model2`* we can use the command `estimates restore` *`model1`* to replace the current estimates (*`model2`* with the first set *`model1`*. This process facilitates many test processes including likelihood ratio tests, Hausman tests, and the preparation of comparisons of the results of competing models. The following is a summary of the main facilities offered by the `estimates` commands.

`estimates` provides the preferred method to store and restore estimation results. Results are identified by a name. In a namelist, you may use the * and ? wild cards. `_all` or * refers to all stored results. A period . refers to the most recent ("active") estimation results even if these have not (yet) been stored.

`estimates store` ***`name`*** saves the current (active) estimation results under the name *name*.

`estimates restore` ***`name`*** loads the results saved under *name* into the current (active) estimation results.

`estimates query` tells you whether the current (active) estimates have been stored and, if so, the name.

`estimates dir` displays a list of the stored estimates.

`estimates drop` ***`namelist`*** drops the specified stored estimation results.

`estimates clear` drops all stored estimation results.

`estimates clear`, `estimates drop _all`, and `estimates drop` * do the same thing. `estimates drop` and `estimates clear` do not eliminate the current (active) estimation results.

`estimates table [`*`namelist`*`] , [`*`options`*`]` draws compiles a table giving comparisons of the results of the models specified in *namelist* using the options set out under *options*.

To illustrate the `estimates` facility we will compile a table similar to table 8.2 on page 289 of Stock and Watson (2007). This is a study of the circulation of economic journals. The dataset has been described in Section 7 on page 11. First we estimate the four models, store the results of each model (`estimates store`) and then produce a table (`estimates table`) summarising the results of all four models.

```
. use Journals,clear

. set more off

. * data transformations
. generate lPriceCite = ln(price/citations)

. generate  lPriceCite2 =  lPriceCite * lPriceCite

. generate  lPriceCite3 =  lPriceCite2 * lPriceCite

. generate lAge = ln(2000 -  foundingyear)

. generate lAgelPcite =  lAge *  lPriceCite

. generate lChar = ln( pages* charpp/1000000)

. generate lsubs = ln( subs)

. * model for first column. One might add quietly here and later to suppress output
. regress  lsubs lPriceCite

      Source |       SS       df       MS              Number of obs =     180
-------------+------------------------------           F(  1,   178) =  224.04
       Model |  125.933995     1  125.933995           Prob > F      =  0.0000
    Residual |  100.056056   178  .562112673           R-squared     =  0.5573
-------------+------------------------------           Adj R-squared =  0.5548
       Total |  225.990051   179  1.26251425           Root MSE      =  .74974

------------------------------------------------------------------------------
       lsubs |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  lPriceCite |  -.5330535   .0356132   -14.97   0.000    -.6033319   -.4627751
       _cons |   4.766212   .0559091    85.25   0.000     4.655882    4.876542
------------------------------------------------------------------------------

. estimates store model1

. regress  lsubs lPriceCite lAge lChar

      Source |       SS       df       MS              Number of obs =     180
-------------+------------------------------           F(  3,   176) =   93.01
       Model |  138.579229     3  46.1930763           Prob > F      =  0.0000
    Residual |  87.4108216   176  .496652396           R-squared     =  0.6132
-------------+------------------------------           Adj R-squared =  0.6066
       Total |  225.990051   179  1.26251425           Root MSE      =  .70474

------------------------------------------------------------------------------
       lsubs |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  lPriceCite |  -.4077182   .0419604    -9.72   0.000    -.4905284    -.324908
        lAge |   .4236486   .0897157     4.72   0.000     .2465915    .6007056
       lChar |   .2056141    .107466     1.91   0.057    -.0064737     .417702
       _cons |   3.206648   .3141164    10.21   0.000     2.586728    3.826567
```

```
------------------------------------------------------------------------------

. estimates store model2

. regress  lsubs lPriceCite lPriceCite2 lPriceCite3 lAge lAgelPcite lChar

      Source |       SS       df       MS              Number of obs =     180
-------------+------------------------------           F(  6,   173) =   50.15
       Model |   143.49038     6  23.9150633           Prob > F      =  0.0000
    Residual |  82.4996706   173  .476876709           R-squared     =  0.6349
-------------+------------------------------           Adj R-squared =  0.6223
       Total |  225.990051   179  1.26251425           Root MSE      =  .69056

------------------------------------------------------------------------------
       lsubs |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  lPriceCite |  -.9609365   .1892198    -5.08   0.000    -1.334413   -.5874597
 lPriceCite2 |   .0165099   .0241346     0.68   0.495    -.0311263    .0641461
 lPriceCite3 |   .0036666   .0068621     0.53   0.594    -.0098776    .0172108
        lAge |   .3730539   .0893609     4.17   0.000     .1966759    .5494319
  lAgelPcite |   .1557773   .0550499     2.83   0.005     .0471214    .2644331
       lChar |   .2346177   .1061335     2.21   0.028     .0251346    .4441009
       _cons |   3.407596   .3184141    10.70   0.000     2.779119    4.036072
------------------------------------------------------------------------------

. estimates store model3

. regress  lsubs lPriceCite lAge  lAgelPcite lChar

      Source |       SS       df       MS              Number of obs =     180
-------------+------------------------------           F(  4,   175) =   75.75
       Model |  143.252483     4  35.8131208           Prob > F      =  0.0000
    Residual |  82.7375673   175  .472786099           R-squared     =  0.6339
-------------+------------------------------           Adj R-squared =  0.6255
       Total |  225.990051   179  1.26251425           Root MSE      =  .68759

------------------------------------------------------------------------------
       lsubs |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
  lPriceCite |  -.8989097   .1615082    -5.57   0.000    -1.217664   -.5801551
        lAge |   .3735148   .0889742     4.20   0.000     .1979143    .5491153
  lAgelPcite |   .1409591   .0448348     3.14   0.002     .0524725    .2294457
       lChar |    .229466   .1051262     2.18   0.030     .0219876    .4369445
       _cons |   3.433521    .314857    10.91   0.000     2.812115    4.054926
------------------------------------------------------------------------------

. estimates store model4

. estimates table model1 model2 model3 model4, b(%9.3f) se stats(N r2 F ll)

----------------------------------------------------------------
    Variable |  model1      model2      model3      model4
-------------+--------------------------------------------------
  lPriceCite |    -0.533      -0.408      -0.961      -0.899
             |     0.036       0.042       0.189       0.162
        lAge |                 0.424       0.373       0.374
             |                 0.090       0.089       0.089
       lChar |                 0.206       0.235       0.229
             |                 0.107       0.106       0.105
 lPriceCite2 |                             0.017
             |                             0.024
 lPriceCite3 |                             0.004
```

```
             |                               0.007
  lAgelPcite |                               0.156       0.141
             |                               0.055       0.045
       _cons |     4.766       3.207         3.408       3.434
             |     0.056       0.314         0.318       0.315
-------------+----------------------------------------------
           N |       180         180           180         180
          r2 |     0.557       0.613         0.635       0.634
           F |   224.037      93.009        50.149      75.749
          ll |  -202.559    -190.399      -185.194    -185.453
------------------------------------------------------------
                                                legend: b/se
```

It is possible to improve considerable on this by using the user written command `esttab`.[30]

```
. * use of use written command esttab
. * findit esttab
. esttab model1 model2 model3 model4, b(%9.3f) se scalars(N r2 F ll) ///
> mtitles("Model 1" "Model 2" "Model 3" "Model 4") star ///
> title("Estimates of the Demand for Economic Journals")

Estimates of the Demand for Economic Journals
----------------------------------------------------------------------------
                      (1)             (2)             (3)             (4)   
                  Model 1         Model 2         Model 3         Model 4   
----------------------------------------------------------------------------
lPriceCite         -0.533***       -0.408***       -0.961***       -0.899***
                  (0.036)         (0.042)         (0.189)         (0.162)   

lAge                                0.424***        0.373***        0.374***
                                  (0.090)         (0.089)         (0.089)   

lChar                               0.206           0.235*          0.229*  
                                  (0.107)         (0.106)         (0.105)   

lPriceCite2                                         0.017                   
                                                  (0.024)                   

lPriceCite3                                         0.004                   
                                                  (0.007)                   

lAgelPcite                                          0.156**         0.141** 
                                                  (0.055)         (0.045)   

_cons               4.766***        3.207***        3.408***        3.434***
                  (0.056)         (0.314)         (0.318)         (0.315)   
----------------------------------------------------------------------------
N                     180             180             180             180   
r2                  0.557           0.613           0.635           0.634   
F                 224.037          93.009          50.149          75.749   
ll               -202.559        -190.399        -185.194        -185.453   
----------------------------------------------------------------------------
Standard errors in parentheses
* p<0.05, ** p<0.01, *** p<0.001
```

It is also possible to output the table in a format suitable for use in producing your final report. Here we produce a file suitable for importing into LaTeX.

[30]A user written program is an addition to Stata which has been written by a user rather than StataCorp. Many such commands are available over the internet. Section 14 contains more information (see page 66 of this note).

```
. * Write results to a LaTeX file
. quietly esttab model1 model2 model3 model4 using journals.tex, replace ///
> b(%9.3f) se scalars(N r2 F ll) mtitles("Model 1" "Model 2" "Model 3" "Model 4") ///
> star title("Estimates of the Demand for Economic Journals")
```

The table produced by this final command is reproduced as Table 7. Similar results can be produced in Word using rtf format.

Table 7: Estimates of the Demand for Economic Journals

	(1)	(2)	(3)	(4)
	Model 1	Model 2	Model 3	Model 4
lPriceCite	-0.533***	-0.408***	-0.961***	-0.899***
	(0.036)	(0.042)	(0.189)	(0.162)
lAge		0.424***	0.373***	0.374***
		(0.090)	(0.089)	(0.089)
lChar		0.206	0.235*	0.229*
		(0.107)	(0.106)	(0.105)
lPriceCite2			0.017	
			(0.024)	
lPriceCite3			0.004	
			(0.007)	
lAgelPcite			0.156**	0.141**
			(0.055)	(0.045)
_cons	4.766***	3.207***	3.408***	3.434***
	(0.056)	(0.314)	(0.318)	(0.315)
N	180	180	180	180
r2	0.557	0.613	0.635	0.634
F	224.037	93.009	50.149	75.749
ll	-202.559	-190.399	-185.194	-185.453

Standard errors in parentheses
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

13 Other Stata estimation commands

A list of other estimation commands follows. Details of the use and syntax of these commands are in the help files and manuals. This list should be regarded as a quick outline of what can be done with Stata. You will find many items on the list that are of little interest to econometrics in general. The syntax of these estimation instructions have the same general structure as the `regress` command.

anova Analysis of variance and covariance

arch ARCH family of estimators

areg Linear regression with a large dummy-variable set

arima ARIMA, ARMAX, and other dynamic regression models

asclogit Alternative-specific conditional logit (McFadden's choice) model

asmprobit Alternative-specific multinomial probit regression

asroprobit Alternative-specific rank-ordered probit regression

binreg Generalized linear models: Extensions to the binomial family

biprobit Bivariate probit regression

blogit Logistic regression for grouped data

bootstrap Bootstrap sampling and estimation

boxcox Box-Cox regression

bprobit Probit regression for grouped data

bsqreg Quantile regression with bootstrap standard errors

ca Simple correspondence analysis

camat Simple correspondence analysis of a matrix

candisc Canonical linear discriminant analysis

canon Canonical correlations

clogit Conditional (fixed-effects) logistic regression

cloglog Complementary log-log regression

cnreg Censored-normal regression

cnsreg Constrained linear regression

discrim knn kth-nearest-neighbour discriminant analysis

discrim lda Linear discriminant analysis

discrim logistic Logistic discriminant analysis

discrim qda Quadratic discriminant analysis

dprobit Probit regression, reporting marginal effects

eivreg Errors-in-variables regression

exlogistic Exact logistic regression

expoisson Exact Poisson regression

factor Factor analysis

factormat Factor analysis of a correlation matrix

fracpoly Fractional polynomial regression

frontier Stochastic frontier models

glm Generalized linear models

glogit Logit and probit for grouped data

gnbreg Generalized negative binomial model

gprobit Weighted least-squares probit regression for grouped data

heckman Heckman selection model

heckprob Probit model with selection

hetprob Heteroskedastic probit model

intreg Interval regression

iqreg Inter-quantile range regression

ivprobit Probit model with endogenous regressors

ivregress Single-equation instrumental-variables estimation

ivtobit Tobit model with endogenous regressors

jackknife Jackknife estimation

logistic Logistic regression, reporting odds ratios

logit Logistic regression, reporting coefficients

manova Multivariate analysis of variance and covariance

mca Multiple and joint correspondence analysis

mds Multidimensional scaling for two-way data

mdsmat Multidimensional scaling of proximity data in a matrix

mdslong Multidimensional scaling of proximity data in long format

mean Estimate means

mfp Multi-variable fractional polynomial models

mlogit Multinomial (polytomous) logistic regression

mprobit Multinomial probit regression

mvreg Multivariate regression

nbreg Negative binomial regression

newey Regression with Newey-West standard errors

nl Non-linear least-squares estimation

nlogit Nested logit model (RUM-consistent and non-normalised)

nlsur System of non-linear equations

ologit Ordered logistic regression

oprobit Ordered probit regression

pca Principal component analysis

pcamat Principal component analysis of a correlation or covariance matrix

poisson Poisson regression

prais Prais-Winsten and Cochrane-Orcutt regression

probit Probit regression

procrustes Procrustes transformation

proportion Estimate proportions

_qreg Internal estimation command for quantile regression

qreg Quantile regression

ratio Estimate ratios

reg3 Three-stage estimation for systems of simultaneous equations

regress Linear regression

rocfit Fit ROC models

rologit Rank-ordered logistic regression

rreg Robust regression

scobit Skewed logistic regression

slogit Stereotype logistic regression

sqreg Simultaneous-quantile regression

stcox Fix Cox proportional hazards model

streg Fit parametric survival models

sureg Zellner's seemingly unrelated regression

svy: Estimation commands for survey data

svy: tabulate oneway One-way tables for survey data

svy: tabulate twoway Two-way tables for survey data

tobit Tobit regression

total Estimate totals

treatreg Treatment-effects model

truncreg Truncated regression

var Vector autoregressive models

var svar Structural vector autoregressive models

varbasic Fit a simple VAR and graph IRFs and FEVDs

vec Vector error-correction models

vwls Variance-weighted least squares

xtabond Arellano-Bond linear dynamic panel-data estimation

xtcloglog Random-effects and population-averaged cloglog models

xtdpdsys Arellano-Bond/Blundell-Bond estimation

xtdpd Linear dynamic panel-data estimation

xtfrontier Stochastic frontier models for panel data

xtgee Fit population-averaged panel-data models by using GEE

xtgls Fit panel-data models using GLS

xthtaylor Hausman-Taylor estimator for error-components models

xtintreg Random-effects interval data regression models

xtivreg Instrumental variables and two-stage least squares for panel-data models

xtlogit Fixed-effects, random-effects, and population-averaged logit models

xtmelogit Multilevel mixed-effects logistic regression

xtmepoisson Multilevel mixed-effects Poisson regression

xtmixed Multilevel mixed-effects linear regression

xtnbreg Fixed-effects, random-effects, and population-averaged negative binomial models

xtpcse OLS or Prais-Winsten models with panel-corrected standard errors

xtpoisson Fixed-effects, random-effects, and population-averaged Poisson models

xtprobit Random-effects and population-averaged probit models

xtrc Random-coefficients models

xtreg Fixed-, between-, and random-effects, and population-averaged linear models

xtregar Fixed- and random-effects linear models with an AR(1) disturbance

xttobit Random-effects tobit models

zinb Zero-inflated negative binomial regression

zip Zero-inflated Poisson regression

ztnb Zero-truncated negative binomial regression

ztp Zero-truncated Poisson regression

Two post regression commands are available after most estimation procedures `predict` and `test`. A set of stored values and versions of the `predict` and `test` commands are also available after the other estimation commands

14 Using user written programs

Stata consists of a fairly basic set of programming instructions. Many Stata commands are programs written using this basic set of instructions. These added commands also function in a similar way to the original instructions. They are included in the distribution as `.ado` files. These *.ado* files can be read on your system and some advanced users may look at these files where the manuals may not provide the information that they need.

Ordinary users can also produce `.ado files` and use the instructions generated by these new `.ado` files in their analysis. It is one of the great features of Stata that it can be expanded in this way. Many users have made their Stata programs available over the internet. If you need an implementation of some routines not contained in Stata it is often worthwhile to look on the internet to see if someone else has implemented what you are trying to do or,perhaps, something close to it.

If you can not find the function that you need in the list in Section 13 you might first try the Stata instruction `hsearch` *`keywords`*. This searches the help files installed on your Stata system, including any user instructions installed on your PC. If this does not yield any results you might try `findit` *`keywords`*. This searches both local files an the web for Stata material related to your `textitkeywords`

If `findit` locates what you need, installing the software may require only a simple click on a link to download and install the required files. If you have full access rights on your PC the files will be installed in the default directories and everything should work.

If you are working on a PARC the software will install. However, when the PC is shut down and restarted the default configuration is restored and the installed Stata files will be deleted. There are several ways around this problem. Which method you use depends on the amount of use that you are making of the facility.

1. By default the additional files are installed in `c:\\ado`. The system on a PACR PC will delete these files as soon as the PC is rebooted. As soon as you have downloaded such files copy this directory to your `s:\\`directory, to a flash drive or other USB device. Every time you want to use the command you simply copy the directory back to the `c:\\` drive.

2. The required `.ado` and help files can be put in your working directory. This is probably a better option but you will need duplicate copies for each project.

3. If you are looking for a more permanent solution you could add instructions to your `profile.do` file that change the default location of the relevant directories. For more details, the Stata command `help sysdir` will give more details.

References

Adkins, L. C. and R. C. Hill (2008). *Using Stata for Principles of Econometrics Third Edition.* Wiley. 5

Baltagi, B. H. (2002). *Econometrics* ($3^{rd} edition$ ed.). Springer. 20, 33

Baltagi, B. H. (2009a). *A Companion to Econometric Analysis of Panel Data.* Wiley. 5

Baltagi, B. H. (2009b). *Econometric Analysis of Panel Data* (Fourth ed.). Wiley. 5

Baum, C. F. (2006). *An Introduction to Modern Econometrics using Stata.* Stata Press. 5

Cameron, A. C. and P. K. Trivedi (1998). *Regression Analysis of Count Data.* Cambridge University Press. 39

Cameron, A. C. and P. K. Trivedi (2005). *Microeconometrics: Methods and Applications.* Cambridge University Press. 5

Cameron, A. C. and P. K. Trivedi (2009). *Microeconometrics using Stata.* Stata Press. 5

Croissant, Y. (2006). *Ecdat: Data sets for econometrics, Version 0.1-5.* http://www.r-project.org. 39

Davidson, R. and J. G. Mackinnon (2004). *Instructors Manual to accompany Econometric Theory and Methods.* Oxford University Press. 32, 33, 49

Granger, C. W. J. (1999). *Empirical Modeling in Economics Specification and Evaluation.* Cambridge University Press. 3

Green, W. H. (2008). *Econometric Analysis.* Pearson Prentice Hall. 34

Hamilton, L. C. (2004). *Statistics with STATA.* Brooks/Cole. 16

Hendry, D. F. (2000). *Econometrics Alchemy or Science* (new ed.). Oxford University Press. 3

Hill, R. C., W. E. Griffiths, and G. C. Lim (2008). *Principles of Econometrics* (Third ed.). Wiley. 5

Judge, G. G., W. E. Griffiths, R. C. Hill, H. Lutkepohl, and T.-C. Lee (1985). *The Theory and Practice of Econometrics* (second ed.). Wiley. 46

Keuzenkamp, H. A. (2000). *Probability, Econometrics and Truth.* Cambridge University press. 3

Kleiber, C. and A. Zeileis (2008). *Applied Econometrics with R.* New York: Springer-Verlag. ISBN 978-0-387-77316-2. 12

Magnus, J. R. and M. S. Morgan (1999). *Methodology and Tacit Knowledge.* Wiley. 3

Stigum, B. P. (1990). *Towards a Formal Science of Economics The Axiomatic method in Economics and Econometrics.* MIT Press. 3

Stock, J. H. and M. W. Watson (2007). *Introduction to Econometrics.* Pearson Addison Wesley. 12, 13, 58

Verbeek, M. (2008). *A guide to Modern Econometrics* (Third ed.). Wiley. 32

Wooldridge, J. M. (2002). *Econometric Analysis of Cross-Section and Panel Data.* MIT Press. 5