Using gretl for Principles of Econometrics, 3rd Edition Version 1.313^1

Lee C. Adkins Professor of Economics Oklahoma State University

May 8, 2010

¹Visit http://www.LearnEconometrics.com/gretl.html for the latest version of this book. Also, check the errata (page 286) for changes since the last update.

License

Using gretl for Principles of Econometrics, 3rd edition. Copyright © 2007 Lee C. Adkins. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation (see Appendix F for details).

Preface

This manual is about using the software package called **gret1** to do various econometric tasks required in a typical two course undergraduate or masters level econometrics sequence. It is written specifically to be used with *Principles of Econometrics, 3rd edition* by Hill, Griffiths, and Lim, although it could be used with many other introductory texts. The data for all of the examples used herein are available as a package from my website at http://www.learneconometrics.com/gret1.html. If you are unfamiliar with gret1 and are interested in using it in class, Mixon Jr. and Smith (2006) have written a brief review of gret1 and how it can be used in an undergraduate course that you may persuade you to give it a try.

The chapters are arranged in the order that they appear in *Principles of Econometrics*. Each chapter contains a brief description of the basic models to be estimated and then gives you the specific instructions or **gretl** code to reproduce all of the examples in the book. Where appropriate, I've added a bit of pedagogical material that complements what you'll find in the text. I've tried to keep this to a minimum since this is not supposed to serve as a substitute for your text book. The best part about this manual is that it, like **gretl**, is free. It is being distributed in Adobe's pdf format and I will make corrections to the text as I find errors.

To estimate a few of the models in POE I've had to resort to another free software called \mathbf{R} . As **gretl** develops I suspect that this small reliance on R will diminish. In any event, **gretl** contains a utility that makes using R quite easy. You'll find an appendix in this book that will get you started.

Gretl also gives users an ability to write his or her own functions, which greatly expands the usefulness of the application. In Chapters 14 and 16 functions are used to estimate a few of the models contained in *POE*. What's more, functions can be shared and imported easily through **gretl**, especially if you are connected to the internet. If **gretl** doesn't do what you want it to now, stay tuned. It soon may. If recent activity is any indication, I am confident that the the **gretl** team will continue to improve this already very useful application. I hope that this manual is similarly useful to those using *Principles of Econometrics*.

I want to thank the gretl team of Allin Cottrell and Riccardo "Jack" Lucchetti for putting so

much effort into **gretl**. It is a wonderful program for **teaching** and **doing** econometrics. It has many capabilities beyond the ones I discuss in this book and other functions are added regularly. Also, Jack has kindly provided me with suggestions and programs that have made this much better than it would have been otherwise. Any remaining errors are mine alone.

Finally, I want to thank my good friend and colleague Carter Hill for suggesting I write this and Oklahoma State University for continuing to pay me while I work on it.

Copyright © 2007, 2008, 2009 Lee C. Adkins.

Contents

1	Intr	duction	1
	1.1	What is Gretl ?	1
		I.1.1 Installing Gretl	2
		1.1.2 Gretl Basics	2
		1.1.3 Common Conventions	5
	1.2	mporting Data	5
	1.3	Using the gretl Language	8
		1.3.1 Console	8
		1.3.2 Scripts	0
		$1.3.3 \text{Sessions} \dots \dots \dots \dots \dots \dots \dots \dots \dots $	2
2	\mathbf{Sim}	le Linear Regression 1	5
	2.1	Simple Linear Regression Model	5
	2.2	Retrieve the Data	5
	2.3	Graph the Data	8
	2.4	Estimate the Food Expenditure Relationship	9

		2.4.1 Elasticity \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots	23
		2.4.2 Prediction	23
		2.4.3 Estimating Variance	25
	2.5	Repeated Sampling	26
	2.6	Script	29
3	Inte	erval Estimation and Hypothesis Testing	34
	3.1	Confidence Intervals	34
	3.2	Monte Carlo Experiment	37
	3.3	Hypothesis Tests	38
	3.4	Script for t-values and p-values	42
	3.5	Script	44
4	Pre	diction, Goodness-of-Fit, and Modeling Issues	46
4	Pre 4.1	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model	46
4	Pre 4.1 4.2	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model	46 46 48
4	Pre 4.1 4.2 4.3	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model	46 46 48 51
4	Pre 4.1 4.2 4.3 4.4	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model	46 46 48 51 53
4	 Pre 4.1 4.2 4.3 4.4 4.5 	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model Coefficient of Determination Reporting Results Functional Forms Testing for Normality	 46 48 51 53 55
4	Pre 4.1 4.2 4.3 4.4 4.5 4.6	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model Coefficient of Determination Reporting Results Functional Forms Testing for Normality Examples	 46 46 48 51 53 55 56
4	 Pre 4.1 4.2 4.3 4.4 4.5 4.6 	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model Coefficient of Determination Reporting Results Functional Forms Testing for Normality Examples 4.6.1	 46 48 51 53 55 56 56
4	Pre 4.1 4.2 4.3 4.4 4.5 4.6	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model Coefficient of Determination Reporting Results Functional Forms Testing for Normality Examples 4.6.1 Wheat Yield Example 4.6.2 Growth Model Example	 46 48 51 53 55 56 56 58
4	Pre 4.1 4.2 4.3 4.4 4.5 4.6	diction, Goodness-of-Fit, and Modeling Issues Prediction in the Food Expenditure Model Coefficient of Determination Reporting Results Functional Forms Testing for Normality Examples 4.6.1 Wheat Yield Example 4.6.3 Wage Equation	 46 46 48 51 53 55 56 56 58 58

		4.6.5	Generalized R^2	60
		4.6.6	Prediction Interval	61
	4.7	Script		61
5	Mu	ltiple I	Regression Model	64
	5.1	Linear	Regression	65
	5.2	Big Aı	ndy's Burger Barn	67
		5.2.1	SSE, R^2 and Other Statistics	68
		5.2.2	Covariance Matrix and Confidence Intervals	68
		5.2.3	t-Tests, Critical Values, and P-values	69
	5.3	Script		71
6	Fur	ther In	ference in the Multiple Regression Model	72
	6.1	F-test		72
	6.1 6.2	F-test Regres	sion Significance	72 78
	6.16.26.3	F-test Regres Extend	sion Significance	72 78 79
	6.16.26.3	F-test Regres Extend 6.3.1	sion Significance	72 78 79 79
	6.16.26.3	F-test Regres Extend 6.3.1 6.3.2	sion Significance	72 78 79 79 80
	6.16.26.36.4	F-test Regres Extend 6.3.1 6.3.2 Nonsa	sion Significance	 72 78 79 79 80 83
	 6.1 6.2 6.3 6.4 6.5 	F-test Regres Extend 6.3.1 6.3.2 Nonsa Model	sion Significance	72 78 79 79 80 83 84
	 6.1 6.2 6.3 6.4 6.5 6.6 	F-test Regres Extend 6.3.1 6.3.2 Nonsa Model RESE	sion Significance	 72 78 79 79 80 83 84 87
	 6.1 6.2 6.3 6.4 6.5 6.6 6.7 	F-test Regres Extend 6.3.1 6.3.2 Nonsa Model RESE Cars E	ssion Significance	 72 78 79 79 80 83 84 87 88
	 6.1 6.2 6.3 6.4 6.5 6.6 6.7 6.8 	F-test Regress Extend 6.3.1 6.3.2 Nonsat Model RESE Cars E Script	sion Significance	 72 78 79 79 80 83 84 87 88 89

7 Nonlinear Relationships

	7.1	Polynomials
	7.2	Interaction Terms
	7.3	Examples
		7.3.1 Housing Price Example
		7.3.2 CPS Example
		7.3.3 Chow Test
		7.3.4 Pizza Example
		7.3.5 Log-Linear Wages Example
	7.4	Script
8	Het	eroskedasticity 104
	8.1	Food Expenditure Example
	8.2	Weighted Least Squares
	8.3	Skedasticity Function
	8.4	Grouped Heteroskedasticity
		8.4.1 Wage Example
		8.4.2 Food Expenditure Example
	8.5	Other Tests for Heteroskedasticity
	8.6	Script
9	Dyn	namic Models and Autocorrelation 120
	9.1	Area Response Model for Sugar Cane
		9.1.1 Bandwidth and Kernel
		9.1.2 Dataset Structure

		9.1.3 HAC Standard Errors	24
	9.2	Nonlinear Least Squares	25
	9.3	Testing for Autocorrelation	27
	9.4	Autoregressive Models and Forecasting	31
		9.4.1 Using the Dialogs \ldots	32
		9.4.2 Using a Script	36
	9.5	Autoregressive Distributed Lag Model	37
	9.6	Script	38
10	Ran	dom Regressors and Moment Based Estimation 14	41
	10.1	Basic Model	41
	10.2	IV Estimation 1/	19
	10.2		±∠
	10.3	Specification Tests	45
		10.3.1 Hausman Test	45
		10.3.2 Testing for Weak Instruments	45
		10.3.3 Sargan Test $\ldots \ldots \ldots$	46
	10.4	Wages Example	48
	10.5	Script	52
11	Sim	ultaneous Equations Models 15	54
	11.1	Truffle Example	54
	11.2	The Reduced Form Equations	55
	11.3	The Structural Equations	55
	11.0		
	11.4	Fulton Fish Example	56

11.5 Script	
12 Analyzing Time Series Data and Cointe	egration 161
12.1 Series Plots	
12.2 Tests for Stationarity	
12.3 Spurious Regressions	
12.4 Cointegration	
12.5 The Analysis Using a Gretl Script $$	
12.6 Script	
13 Vector Error Correction and Vector A Macroeconometrics	utoregressive Models: Introduction to 178
13.1 Vector Error Correction	
13.1.1 Series Plots–constant and trends	
13.1.2 Selecting Lag Length	
13.1.3 Cointegration Test	
13.1.4 VECM	
13.2 Vector Autoregression	
13.3 Script	
14 Time-Varying Volatility and ARCH Me metrics	odels: Introduction to Financial Econo- 195
14.1 ARCH and GARCH	
14.2 Testing for ARCH	
14.3 Simple Graphs	
14.4 Threshold ARCH	

14.5	Garch-in-Mean	204
14.6	Script	208
15 Poo	ling Time-Series and Cross-Sectional Data	2 11
15.1	A Basic Model	212
15.2	Estimation	213
	15.2.1 Pooled Least Squares	214
	15.2.2 Fixed Effects	215
	15.2.3 Random Effects	216
	15.2.4 SUR	218
15.3	NLS Example	222
15.4	Script	226
16 Qua	litative and Limited Dependent Variable Models	227
16 Qua	litative and Limited Dependent Variable Models	227
16 Qua 16.1	Ilitative and Limited Dependent Variable Models Probit	227 227 230
16 Qua 16.1 16.2	Itative and Limited Dependent Variable Models Probit	 227 227 230 234
16 Qua 16.1 16.2	Itative and Limited Dependent Variable Models Probit	 227 227 230 234
 16 Qua 16.1 16.2 16.3 	Alitative and Limited Dependent Variable Models Probit	 227 227 230 234 236
 16 Qua 16.1 16.2 16.3 16.4 	Alitative and Limited Dependent Variable Models Probit	 227 227 230 234 236 236
 16 Qua 16.1 16.2 16.3 16.4 16.5 	Itative and Limited Dependent Variable Models Probit	 227 227 230 234 236 236 238
 16 Qua 16.1 16.2 16.3 16.4 16.5 16.6 	Itative and Limited Dependent Variable Models Probit	 227 227 230 234 236 236 238 240
 16 Qua 16.1 16.2 16.3 16.4 16.5 16.6 16.7 	Alitative and Limited Dependent Variable Models Probit	 227 227 230 234 236 236 238 240 241
 16 Qua 16.1 16.2 16.3 16.4 16.5 16.6 16.7 16.8 	Alitative and Limited Dependent Variable Models Probit	 227 227 230 234 236 236 238 240 241 243

	16.9.1 Multinomial Logit	249
	16.9.2 Conditional Logit	252
	16.9.3 Ordered Probit	254
	16.10Script	256
\mathbf{A}	gretl commands	261
	A.1 Estimation	261
	A.2 Tests	262
	A.3 Transformation	263
	A.4 Statistics	263
	A.5 Dataset	264
	A.6 Graphs	265
	A.7 Printing	265
	A.8 Programming	265
	A.9 Utilities	266
в	Some Basic Probability Concepts	267
\mathbf{C}	Some Statistical Concepts	273
	C.1 Summary Statistics	273
	C.2 Interval Estimation	275
	C.3 Hypothesis Tests	276
	C.4 Testing for Normality	277
D	Using R with gretl	279

	D.1 Packages	. 2	83
	D.2 Stata Datasets	. 2	84
	D.3 Final Thoughts	. 2	85
\mathbf{E}	Errata and Updates	28	36
F	GNU Free Documentation License	28	38
G	NU Free Documentation License	28	38
	1. APPLICABILITY AND DEFINITIONS	. 2	89
	2. VERBATIM COPYING	. 2	90
	3. COPYING IN QUANTITY	. 2	90
	4. MODIFICATIONS	. 2	91
	5. COMBINING DOCUMENTS	. 2	93
	6. COLLECTIONS OF DOCUMENTS	. 2	93
	7. AGGREGATION WITH INDEPENDENT WORKS	. 2	93
	8. TRANSLATION	. 2	94
	9. TERMINATION	. 2	94
	10. FUTURE REVISIONS OF THIS LICENSE	. 2	94

List of Figures

1.1	Opening the command line interface version of gretl using Start>Run
1.2	The command line version of gret1 3
1.3	The main window for gretl's GUI
1.4	Opening sample data files from gretl's main window
1.5	Data file window
1.6	Listing variables in your data set
1.7	The command reference window
1.8	The command reference window 10
1.9	Command script editor
1.10	The session window
1.11	Saving a session
2.1	Loading gretl data
2.2	Editing data attributes
2.3	Variable edit dialog box
2.4	Plotting dialog box

2.5	XY plot of the Food Expenditure data	19
2.6	Opening the OLS dialog box	20
2.7	OLS dialog box	21
2.8	Gretl console	21
2.9	Model Window: Least Squares Results	22
2.10	Obtaining Summary Statistics	24
2.11	Summary Statistics	24
2.12	Elasticity calculation	25
2.13	OLS covariance matrix	31
2.14	Monte Carlo experiments	31
2.15	Monte Carlo results	32
2.16	More Monte Carlo results	33
3.1	Critical values utility	35
3.2	Critical Values	35
3.3	Confidence intervals	37
3.4	Confidence intervals from the dialog	37
3.5	P-value utility	40
3.6	Results from the critical value utility	41
4.1	Selecting ANOVA	48
4.2	ANOVA table	48
4.3	Summary statistics: \bar{R}^2	49
4.4	Adding fitted values to the data	50

4.5	Highlight variables	50
4.6	Correlation matrix	51
4.7	Plotting predicted vs actual	52
4.8	LaTeX options	52
4.9	Adding new variables to the data	54
4.10	The summary statistics for the least squares residuals	55
4.11	Wheat yield XY plot	57
4.12	Wheat yield XY time series plot	58
4.13	Graph dialog	59
4.14	Wheat yield XY plot with cubic term	60
5.1	OLS dialog from the pull-down menu	65
5.2	OLS specify model dialog	66
5.3	The OLS shortcut	66
6.1	Least Squares model results	74
6.2	Tests pull-down menu	75
6.3	Omit variable dialog box	75
6.4	Results from omit variable dialog	76
6.5	Linear restriction dialog box	76
6.6	Restrict results	77
6.7	Overall F-statistic	79
6.8	Big Andy from the console	80
6.9	Does Advertising matter?	81

6.10	Using Restrict to test hypotheses
6.11	Adding logarithms of your variables
6.12	gretl output for the beer demand
6.13	Model table
7.1	Using genr and scalar
7.2	Data>Dataset Structure pull-down menu
7.3	Dataset Structure dialog box
8.1	Robust standard errors check box
8.2	Options dialog box
9.1	Dataset structure wizard
9.2	Nonlinear least squares results
9.3	Correlogram
9.4	Correlogram using the GUI
9.5	Correlogram lags dialog box
9.6	Correlogram produced by gnuplot
9.7	LM autocorrelation test results
9.8	Add lags to regressors
9.9	Lag order dialog box
9.10	Forecast model result
9.11	Add observations to your sample
9.12	Forecast dialog box
9.13	Forecast graph

9.14 ARDL(3,2) results
10.1 Two-Stage Least Squares estimator from the pull-down menus
10.2 Two-Stage Least Squares dialog box
10.3 Results from using the omit statement after least squares
12.1 Select all of the series
12.2 Add first differences to the data
12.3 Graphing multiple time series using the selection box
12.4 Multiple time series graphs. \ldots
12.5 Multiple time series graphs for Fed Funds rate and 3 year bonds
12.6 Choose the ADF test from the pull-down menu
12.7 The ADF test dialog box. \ldots
12.8 The ADF test results
12.9 Set sample box
12.10Sample information in the main window
12.11Two random walk series
12.12Scatter plot of two random walk series
12.13View the least squares results from a graph
12.14The dialog box for the cointegration test
12.15The pull-down menu for choosing whether to include constant or trends in the ADF regression
13.1 Plots of US and AU GDP and their differences
13.2 ADF levels results U.S. and AUS

13.3 Testing up in ADF regression
13.4 The VAR dialog box $\ldots \ldots 18$
13.5 VAR results
13.6 Impulse Response Functions
13.7 Graphing the Impulse Responses
13.8 Impulse Responses
13.9 Forecast Error Variance Decompositions
14.1 Choose GARCH from the main gretl window
14.2 Estimating ARCH from the dialog box
14.3 Test for ARCH using the pull-down menu
14.4 Testing ARCH box
14.5 ARCH test results
14.6 Histograms from the pull-down menu
14.7 Frequency plot setup box
14.8 Histogram with Normal curve
14.9 Plotting GARCH variances
14.10Plotting GARCH variances
14.11Threshold GARCH script
14.12TGARCH results
14.13MGARCH script
14.14MGARCH results
15.1 Database Server

15.2	Databases on the server
15.3	SUR output
16.1	Probit model dialog box
16.2	MNL estimates from Gretl
16.3	MNL estimates from Gretl
16.4	Ordered probit results from gretl
16.5	Heckit dialog box
16.6	Multinomial logit results from R
16.7	Conditional Logit from R
16.8	Ordered probit results from R
B.1	Obtaining summary statistics
B.2	Results for summary statistics
B.3	P-value finder dialog utility
B.4	P-value results
C.1	Critical values from the Console
D.1	The R console when called from Gretl
D.2	Gretl options
D.3	Least squares using R
D.4	ANOVA results from R

Chapter

Introduction

In this chapter you will be introduced to some of the basic features of **gretl**. You'll learn how to install it, how to get around the various windows in **gretl**, and how to import data. At the end of the chapter, you'll be introduced to **gretl**'s powerful language.

1.1 What is Gretl?

Gretl is an acronym for Gnu Regression, Econometrics and Time-series Library. It is a software package for doing econometrics that is easy to use and reasonably powerful. **Gretl** is distributed as free software that can be downloaded from http://gretl.sourceforge.net and installed on your personal computer. Unlike software sold by commercial vendors (SAS, Eviews, Shazam to name a few) you can redistribute and/or modify gretl under the terms of the GNU General Public License (GPL) as published by the Free Software Foundation.

Gretl comes with many sample data files and a database of US macroeconomic time series. From the **gretl** web site, you have access to more sample data sets from many of the leading textbooks in econometrics, including ours *Principles of Econometrics* by Hill et al. (2007). **Gretl** can be used to compute least-squares, weighted least squares, nonlinear least squares, instrumental variables least squares, logit, probit, tobit and a number of time series estimators. **Gretl** uses a separate Gnu program called *gnuplot* to generate graphs and is capable of generating output in LaTeX format. As of this writing **gretl** is under development so you can probably expect some bugs, but in my experience it is pretty stable to use with my Windows XP systems.

1.1.1 Installing Gretl

To install **gretl** on your system, you will need to download the appropriate executable file for the computer platform you are using. For Microsoft Windows users the appropriate site is http://gretl.sourceforge.net/win32/. One of the nice things about gretl is that Macintosh and Linux versions are also available. If you are using some other computer system, you can obtain the source code and compile it on whatever platform you'd like. This is not something you can do with any commercial software package that I've seen.

Gretl depends on some other (free) programs to perform some of its magic. If you install gretl on your Mac or Windows based machine using the appropriate executable file provided on gretl's download page then everything you need to make gretl work should be installed as part of the package. If, on the other hand, you are going to build your own gretl using the source files, you may need to install some of the supporting packages yourself. I assume that if you are savvy enough to compile your own version of gretl then you probably know what to do. For most, just install the self-extracting executable, gretl_install.exe, available at the download site. Gretl comes with an Adobe pdf manual that will guide you through installation and introduce you to the interface. I suggest that you start with it, paying particular attention to Chapters 1 and 2 which discuss installation in more detail and some basics on how to use the interface.

Since this manual is based on the examples from *Principles of Econometrics, 3rd edition (POE)* by Hill et al. (2007), you should also download and install the accompanying data files that go with this book. The file is available at

http://www.learneconometrics.com/gretl/poesetup.exe.

This is a self-extracting windows file that will install the *POE* data sets onto the c:\Program Files\gretl\data directory of your computer's harddrive. If you have installed gretl in any place other than c:\Program Files\gretl then you are given the opportunity to specify a new location in which to install the program during setup.

1.1.2 Gretl Basics

There are several different ways to work in **gretl**. Until you learn to use **gretl**'s rather simple and intuitive language syntax, the easiest way to use the program is through its built in graphical user interface (GUI). The graphical interface should be familiar to most of you. Basically, you use your computer's mouse to open dialog boxes. Fill in the desired options and execute the commands by clicking on the OK button. Those of you who grew up using MS Windows or the Macintosh will find this way of working quite easy. **Gretl** is using your input from the dialogs, delivered by mouse clicks and a few keystrokes, to generate computer code that is executed in the background.

Gretl offers a command line interface as well. In this mode you type in valid **gretl** commands either singly from the **console** or in batches using **scripts**. Once you learn the commands, this is

surely the easiest way to work. If you forget the commands, then return to the dialogs and let the graphical interface generate them for you.

There is a command line version of **gretl** that skips the dialogs altogether. The command line version is launched by executing **gretlcli** in a dos command window. In Windows choose **Start>Run** to open the dialog shown in figure 1.1. In the box, use **Browse** button to locate the

Figure 1 1. Dependent the command line intertage version of great using			
	STATT SKII	Startsk	{11r
i iguit 1.1. Optimig the command mit mornate version of great using	Duar triu	D 0 0 1 0 / 10	ιui

Run	? 🔀
-	Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:	"I:\Program Files\gretl\gretlcli.exe"
	OK Cancel Browse

directory in which gretl is installed. On my machine it is installed on the $I:\$ drive. Click OK and the command line version shown in figure 1.2 opens. There are a couple of messages that the





Windows registry couldn't be opened: this is a good thing so don't be alarmed. If you are in fact using the Windows operating system, then you probably won't be using **gretl** from the command line anyway. This version of the program is probably the most useful for Linux users wishing to run **gretl** from a terminal window. We won't be using it in this manual.

A better way to execute single **gretl** commands is through the **gretl console**. In normal practice, the console is a lot easier to use than the **gretlcli**. It offers some editing features and

immediate access to other ways of using **gretl** that aren't available in the straight command line version of the program. The console and its use is discussed in section 1.3.1.

If you want to execute a series of commands, you do this using **scripts**. One of the great things about **gretl** is that it accumulates commands executed singly from the console into a **command** log that can be run in its entirety at another time. This topic can be found in section 1.3.2. So, if you have completed an analysis that involves many sequential steps, the script can be open and run in one step to get the desired result.

You can use the script environment to conduct Monte Carlo studies in econometrics. Monte Carlo studies use computer simulation (sometimes referred to as experiments) to study the properties of a particular technique. This is especially useful when the mathematical properties of your technique are particularly difficult to ascertain. In the exercises below, you will learn a little about doing these kinds of experiments in econometrics.

In Figure 1.3 below is the main window in gretl.

	gre	etl											_ 🗆 🗙
	Eile	<u>T</u> ools	<u>D</u> ata	<u>V</u> iew	<u>A</u> dd	<u>S</u> ample	<u>V</u> ariable	<u>M</u> odel	<u>H</u> elp	-		Menu Bar	1
	andy.	gdt											
	ID #	Variable	e name	Descri	otive la	bel							
	0	const		auto-o	ienerat	ed consta	nt						
	1	S		Sales									
	2	Ρ		Price									
	3	A		Adver	tising								
								Undated	: Full ra	nge 1 - 75			
	iii	2 -		fx 👖	1	β β				— т	ool Bai		
Ŀ													

Figure 1.3: The main window for gretl's GUI

Across the top of the window you find the menu bar. From here you import and manipulate data, analyze data, and manage output. At the bottom of the window is the **gretl** toolbar. This contains a number of useful utilities that can be launched from within **gretl**. Among other things, you can get to the **gretl** web site from here, open the pdf version of the manual, or open the MS Windows calculator (very handy!). More will be said about these functions later.

1.1.3 Common Conventions

In the beginning, I will illustrate the examples using a number of figures (an excessive number to be sure). These figures are screen captures of **gretl**'s windows as they appear when summoned from the pull-down menus. As you become familiar with **gretl** the frequency of these figures will diminish and I will direct you to the proper commands that can be executed in the console or as a script using words only. More complex series of commands may require you to use the **gretl** script facilities which basically allow you to write simple programs in their entirety, store them in a file, and then execute all of the commands in a single batch. The convention used will be to refer to menu items as A>B>C which indicates that you are to click on option A on the menu bar, then select B from the pull-down menu and further select option C from B's pull-down menu. All of this is fairly standard practice, but if you don't know what this means, ask your instructor now.

1.2 Importing Data

Obtaining data in econometrics and getting it into a format that can be used by your software can be challenging. There are dozens of different pieces of software and many use proprietary data formats that make transferring data between applications difficult. You'll notice that the authors of your book have provided data in several formats for your convenience. In this chapter, we will explore some of the data handling features of **gretl** and show you (1) how to access the data sets that accompany your textbook (2) how to bring one of those data sets into **gretl** (3) how to list the variables in the data set and (4) how to modify and save your data. **Gretl** offers great functionality in this regard. Through **gretl** you have access to a very large number of high quality data sets from other textbooks as well as from sources in industry and government. Furthermore, once opened in **gretl** these data sets can be exported to a number of other software formats.

First, we will load the food expenditure data used in Chapter 2 of *POE*. The data set contains two variables named x and y. The variable y is weekly expenditures on food in a household and x is weekly income measured in \$100 increments.

Open the main **gretl** window and click on File>Open data>sample file as shown in Figure 1.4.

Alternately, you could click on the open dataset button on the toolbar. The button looks like a folder and is on the far right-hand side of the toolbar. This will open another window (Figure 1.5) that contains tabs for each of the data compilations that you have installed in the gretl/data directory of your program. If you installed the data sets that accompany this book using the self extracting windows program then a tab will appear like the one shown in Figure 1.5.

Click on the **POE** tab and scroll down to find the data set called 'food', highlight it using the cursor, and open it using the 'open' button at the top of the window. This will bring the variables of the food expenditure dataset into gretl. At this point, select Data on the menu bar



Figure 1.4: Opening sample data files from gretl's main window

Figure 1.5: This is **gretl**'s data files window. Notice that in addition to POE, data sets from Ramanathan (2002), Greene (2003), Stock and Watson (2006) are installed on my system.

📓 gretl: da	ta files	
Greti Green	e Pot Ramanathan Stock-Watson Stock-Watson 2e Wooldridge	
File	Summary	^
crime	Exercise 15.12	
csi	Exercise 12.9	
demand	Exercise 15.4	
edu_inc	Chapter 6	
euro	Chapter 14; Exercise 14.5	
exrate	Exercise 14.6	
fair	Exercise 2.14, 3.9, 4.11, 5.11, 6.16, 7.15	
FigureC-3	FigureC-3	
florida	Exercise 4.16	
food	Chapters 2,3,8; Exercises 8.10, 8.11	
fullmoon	Exercise 7.2	
fultonfish	Chapter 11; Exercises 11.11, 11.12, C.11	
gascar	Exercise 15.5	
gasga	Exercise 8.16	~

and then Display values as shown in Figure 1.6.

関 grett 📃 🗌 🗌 🗌									
Eile	Tools	<u>D</u> ata <u>Vi</u> ew <u>A</u> dd <u>S</u> ample ⊻ariable <u>M</u> od	el <u>H</u> elp						
andy.g	dt	Select all Ctrl+A							
ID#	Va	r <u>D</u> isplay values							
0	cc	n <u>E</u> dit values							
1	S	Add observations							
2		Remove extra observations							
3	A	Dataset info							
L		Observation markers							
L	⊻ariable labels…								
L	Dataset structure								
Compact data									
L	Expand data								
L	Iranspose data								
L	Sort data								
Set missing value code									
Count missing values									
C Eind variable Ctrl+F									
Refresh window									
L									
L									
		Unda	ted: Full range 1 - 75						
iii	🎟 🖻 🖿 f× 🔁 😫 🖉 ĝ 🗁								

Figure 1.6: Use the cursor to highlight all of the variables. Then click Data>Display values to list the data set.

From the this pull-down menu a lot can be accomplished. You can edit, add observations, and impose a **structure** of your dataset. The structure of your dataset is important. You can choose between time series, cross sections, or panel data structures. The options **Gretl** gives you depend on this structure. For instance, if your data are structured as a time series, **gretl** will allow you to take lags and differences of the variables. Certain procedures that can be used for time series analysis will only be available to you if your dataset has been structured structured for it. If a **gretl** command is not available from the defined dataset structure, then it will be greyed out in the pull-down menus.

Notice in Figure 1.4 that **gretl** gives you the opportunity to **import** data. Expanding this (File>Open data>Import) gives you access to several other formats, including ASCII, CSV, EX-CEL and others. Also, from the File pull-down menu you can export a data set to another format. If you click on File>Databases>On database server (Figure 1.4) you will be taken to a web site (provided your computer is connected to the internet) that contains a number of high quality data sets. You can pull any of these data sets into **gretl** in the same manner as that described above for the *POE* data sets. If you are required to write a term paper in one of your classes, these data sets may provide you with all the data that you need.

1.3 Using the gretl Language

The **gretl** GUI is certainly easy to use. However, you can get results even faster by using **gretl**'s language. The language can be used from the **console** or by collecting several lines of programming code into a file and executing them all at once in a **script**.

An important fact to keep in mind when using **gretl** is that its language is **case sensitive**. This means that lower case and capital letters have different meanings in **gretl**. The practical implication of this is that you need to be very careful when using the language. Since **gretl** considers x to be different from X, it is easy to make programming errors. If **gretl** gives you a programming error statement that you can't quite decipher, make sure that the variable or command you are using is in the proper case.

1.3.1 Console

Gretl's console provides you a way to execute programs interactively. A console window opens and from the prompt (?) you can execute gretl commands one line at a time. You can open the gretl console from the Tools pull-down menu or by a left mouse click on the "Gretl console" button is the toolbar. This button is the third one on the left side of the toolbar in Figure 1.3. From the console you execute commands, one by one by typing gretl code after the command prompt. Each command that you type in is held in memory so that you can accumulate what amounts to a "command history." To reuse a command, simply use the up arrow key to scroll through the commands you've typed in until you get to the one you want. You can edit the

From the command prompt, '?' you can type in commands from the **gretl** language. For instance, to estimate the food expenditure model in section 2.4 using least squares type

command to fix any syntax errors or to make any changes you desire before hitting the enter key

? ols y const x

to execute the statement.

The results will be output to the console window. You can use the window's scroll bar on the right hand side to scroll up if you need to.

Remember, (almost) anything that can be done with the pull-down menus can also be done through the console. Of course, using the console requires you to use the correct language syntax, which can be found in the **gretl** command reference. The command reference can be accessed from the toolbar by clicking the button that looks like a lifesaver. It's the fourth one from the right hand side of the toolbar:



. It is also accessible from the menu bar through Help. The option marked plain text F1 actually brings up all of the commands in a hypertext format. Clicking on anything in blue will take you to the desired information for that command. Obviously, the keyboard shortcut F1 will also bring up the command reference (Figure 1.7). Notice

Figure 1.7: The command reference can be accessed in a number of ways: The 'life-saver' icon on the toolbar, Help>Command reference>plain text from the pull-down menu, or keyboard shortcut F1.

Index Gretl Command Reference > Tests Statistics > Dataset add adf arch arima boxplot break catch chow coeffsum coint coint2 corr corrgm data dataset debug delete diff difftest discrete dummify duration elif else endif endiop equptint equation estimate fcast foreign freq function garch genr gmm gnuplot graphg hausman heckit help hsk hurst if include info intreg hsk hust if leverage logistic logit logs loop mahal meantest mle modeltab modprint modest mpols	🙀 gretl: command reference	e					_ 0 >
Index > Tests > Statistics > Dataset arbond arch arima boxplot break chow coeffsum coint coint2 corr corr corrgm difftest discrete dummify fcast foreign freq function garch genr printing gmm gnuplot graphy hak hurst if include info hsk hurst if include lags ldiff leverage logistic loggs <th>⊕, Q, 💥</th> <th></th> <th></th> <th></th> <th></th> <th></th> <th>🔍 this page 💌</th>	⊕, Q, 💥						🔍 this page 💌
negoin nis normtest hulldat ols omit open orthdev outfile panel pca pergm poisson print printf probit pvalue qlrtest qqplot quantreg quit rename reset restrict rmplot run runs scatters sdiff set setinfo setobs setmiss shell smpl spearman sprintf square sscanf store summary system tabprint textplot tobit tsls var varlist vartest vecm vif wls xcorrgm xtab	Index Tests Statistics Dataset Estimation Graphs Programming Transformations Printing Prediction Utilities	add arbond chow cusum difftest end fcast gmm hsk kalman leverage meantest negbin open poisson qqplot rmplot setinfo sprintf tabprint vartest	G adf arch coeffsum data discrete endif foreign gnuplot hurst kpss logistic mle nls orthdev print quantreg run setobs square textplot	anova arima coint dataset dummify endloop freq graphpg if labels logit modeltab normtest outfile printf quit runs setmiss sscanf tobit vif	mand Rei append boxplot coint2 debug duration eqnprint function hausman include lad logs modprint nulldata panel probit rename scatters shell store tsls wis	ar break corr delete elif equation garch heckit info lags loop modtest ols pca pvalue reset sdiff smp1 summary var xcorrgm	ar1 catch corrgm diff else estimate genr help intreg ldiff mahal mpols omit pergm qlrtest restrict set spearman system varlist xtab

that you can also search for commands by topic from the command syntax window. Select **Topics** and choose the desired category from the list. This can be helpful because it narrows the list to those things that you actually want (e.g., **Topics>Estimation>ols**).

The function reference is a relatively new addition to gret1 that will help you to locate the names gret1 uses to save results, transform variables, and that you will find helpful in writing your own programs. To access the function reference, click Help>Function reference from the pull-down menu as shown in Figure 1.8.

Figure 1.8: The function reference can be accessed by Help>Function reference from the pulldown menu.

gretl: function reference											
€. €. Ж					C	this page 💌					
Index Accessors	Gretl Function Reference										
 Accessors Mathematical Strings Numerical methods Fiters Statistical Probability Linear algebra Matrix shaping Transformations Data utilities 	<pre>\$ahat \$compan \$error \$hausman \$inl \$yvalue \$sigma \$trsq \$windows abs asinh bkfilt ceil cols cov diag dummify fdjac floor diag dummify fdjac floor diag dummify fdjac floor diag dummify fdjac floor diag dummify fdjac floor diag dummify fdjac floor diag dummify fdjac floor diag dummify fdjac floor diag dumnify fdjac floor diag dumnify fdjac floor ginv imhof issries ldiff log10 makemask mcov median missing monthlen mshape nelem obs pead qrdecomp rcows selifr sortby str y tr upper varnum</pre>	<pre>\$aic \$datatype \$ess \$hqc \$coeff \$tho \$stderr \$T \$uhat \$xlist acos atan boxcox cholesky corr critical diff fracdiff hafilt iminc invcdf kdensity isstring lincomb log2 max moory mexp misstero movag msortby nobs obslabel orthdev pmin psdroot quantile readfile seq sqrt strstr tan transp urcpval vec</pre>	Gretl Functi %bic %df %fstat %potential %nobs %rsq %stopwatch %t1 %unit %trinv acosh atanh cdemean cmult corrgm cum digamma epochday ffti gammafun I iminr invpd kfilter lags ljungbox logistic maxc mean min mlag mpols muniform normal obsnum pdf pnobs pvalue rank replace scin sccanf sun tanh trim values vech	on Refere \$chisq \$dwpval \$gmmcrit \$jbeta \$hvars \$sample \$sysA \$t2 \$vev \$yhat argname bessel cdf cnorm cos deseas dnorm exp filter getenv imaxc infnorm irr ksimul lastobs lngamma lower maxr meanc minc mormal mread mwrite pyr ok pergm polroots qform ranking resample sdiff sinh sst sumc toepsolv uniform var weekday	NCE \$coeff \$ec \$h \$ybeta \$pd \$sargan \$ysB \$test \$version asin BFGSmax cdiv colnames cosh det dsort fcstats firstobs gini imaxr int islist ksmooth ldet log lrvar mcorr meanr minr mols mreverse mxtab nullspace onenorm pmax princomp qnorm randgen round selifc sort strlen surt tolower unvech varame wmean surt strlen surt su						
	zeromiss	zeros									

1.3.2 Scripts

Gretl commands can be collected and put into a file that can be executed at once and saved to be used again. This is accomplished by opening a new **command script** from the file menu. The command File>Script files>New script from the pull-down menu opens the command script editor

shown in Figure 1.9. Type the commands you want to execute in the box using one line for each com-

Figure 1.9: The Command Script editor is used to collect a series of commands into what **gretl** calls a **script**. The script can be executed as a block, saved, and rerun at a later time.

🙀 grett: command script	_ 🗆 🗙
🖬 🗃 🖴 🔗 🖡 🖸 🤷 🎭 🎇 🖂 🧮 🗶	
<pre>open "I:\Program Files\gretl\data\poe\food.gdt" genr ly = log(y) genr lx = log(x) ols ly const lx</pre>	

mand. Notice that in the first line of the script, "I:\Program Files\gretl\data\poe\food.gdt", the complete file and path name are enclosed in double quotes, " ". This is necessary because the **Program Files** directory in the pathname includes a space. If you have **gretl** installed in a location that does not include a space, then these can be omitted.

If you have a very long command that exceeds one line, use the backslash (\backslash) as a **continuation command**. Then, to save the file, use the "save" button at the top of the box (first one from the left). If this is a new file, you'll be prompted to provide a name for it.

To run the program, click your mouse on the "gear" button. In the figure shown, the food.gdt gretl data file is opened. The genr commands are used to take the logarithm of y and x, and the ols command discussed in section 2.4 is used to estimate a simple linear regression model that has ln(y) as its dependent variable and ln(x) as the independent variable. Note, the model also includes constant.

A new script file can also be opened from the toolbar by mouse clicking on the "new script" button or by using the keyboard command, Ctrl+N.¹

One of the handy features of the command script window is how the help function operates.

At the top of the window there is an icon that looks like a lifesaver 0. Click on the help button and the cursor changes into a question mark. Move the question mark over the command you want help with and click. Voila! You either get an error message or you are taken to the topic from the command reference. Slick!

¹ "Ctrl+N" means press the "Ctrl" key and, while holding it down, press "N".

1.3.3Sessions

Gretl also has a "session" concept that allows you to save models, graphs, and data files into a common "iconic" space. The session window appears below in Figure 1.10. Objects are represented



Figure 1.10: The session window

as icons and these objects can be saved for later use. When you save your session, the objects you have added should be available again when you re-open the session. To add a model to your session, use the File>Save to session as icon option from the model's pull-down menu. To add a graph, right click on the graph and choose the option save to session as icon. If you want to save the results in your session, don't forget to do so; right click on the session window and choose Save session or from the main gretl window, select File>Session files>Save session as shown below in Figure 1.11.

Gretl also collects all of the commands you've executed via the GUI in the icon labeled 'session.' This makes it very easy to use the GUI to execute unfamiliar commands and then use the code generated by **gretl** to put into a script.

Once a model or graph is added, its icon will appear in the **session icon view** window. Doubleclicking on the icon displays the object, while right-clicking brings up a menu which lets you display or delete the object. You can browse the dataset, look at summary statistics and correlations, and save and revisit estimation results (Models) and graphs.

The model table is a way of combining several estimated models into a single table. This is very



Figure 1.11: Saving a session

useful for model comparison. From page 16 of the gretl manual ((Cottrell and Lucchetti, 2007)):

In econometric research it is common to estimate several models with a common dependent variable the models contain different independent variables or are estimated using different estimators. In this situation it is convenient to present the regression results in the form of a table, where each column contains the results (coefficient estimates and standard errors) for a given model, and each row contains the estimates for a given variable across the models.

In the Icon view window **gretl** provides a means of constructing such a table (and copying it in plain text, LaTeX or Rich Text Format). Here is how to do it:

- 1. Estimate a model which you wish to include in the table, and in the model display window, under the File menu, select Save to session as icon or Save as icon and close.
- 2. Repeat step 1 for the other models to be included in the table (up to a total of six models).
- 3. When you are done estimating the models, open the icon view of your **gretl** session, by selecting **Icon view** under the View menu in the main **gretl** window, or by clicking the **session icon view** icon on the **gretl** toolbar.
- 4. In the Icon view, there is an icon labeled Model table. Decide which model you wish to appear in the left-most column of the model table and add it to the table, either by dragging its icon onto the Model table icon, or by right-clicking on the model icon and selecting Add to model table from the pop-up menu.

- 5. Repeat step 4 for the other models you wish to include in the table. The second model selected will appear in the second column from the left, and so on.
- 6. When you are finished composing the model table, display it by double-clicking on its icon. Under the Edit menu in the window which appears, you have the option of copying the table to the clipboard in various formats.
- 7. If the ordering of the models in the table is not what you wanted, right-click on the model table icon and select Clear table. Then go back to step 4 above and try again.

In section 6.5 you'll find an example that uses the model table and a Figure (6.13) that illustrates the result.

Chapter

Simple Linear Regression

In this chapter you are introduced to the simple linear regression model, which is estimated using the principle of least squares.

2.1 Simple Linear Regression Model

The simple linear regression model is

$$y_t = \beta_1 + \beta_2 x_t + e_t \quad t = 1, 2, \dots, T \tag{2.1}$$

where y_t is your dependent variable, x_t is the independent variable, e_t is random error, and β_1 and β_2 are the parameters you want to estimate. The errors of the model, e_t , have an average value of zero for each value of x_t ; each has the same variance, σ^2 , and are uncorrelated with one another. The independent variable, x_t , has to take on at least two different values in your dataset. If not, you won't be able to estimate a slope! The error assumptions can be summarized as $e_t | x_t \text{ iid } N(0, \sigma^2)$. The expression *iid* stands for *independently and identically distributed* and means that the errors are statistically independent from one another (and therefor uncorrelated) and that each has the same probability distribution. Taking a random sample from a single population accomplishes this.

2.2 Retrieve the Data

The first step is to load the food expenditure and income data into **gretl**. The data file is included in your **gretl** sample files–provided that you have installed the *Principles of Econometrics* data supplement that is available from our website. See section 1.1.1 for details.

Figure 2.1: Food Expenditure data is loaded from food.gdt using File>Open data>sample file and choosing the food dataset from the sample files that accompany *POE*.

📓 gr	📓 greti 📃 🗖 🔤									
File	Tool	s Data	View	Add	Sample	Variable	Model	Help		
food.	food.adt									
ID # Variable name Descriptive label										
0	0 const auto-generated constant									
1	1 y Food Expenditure									
2	x		Weekly	Income	e (\$100)					
							Undated	I: Full range 1 - 40		
i		-	fx	A 🕅) 🗠 Â					
				_						

Load the data from the data file *food.gdt*. Recall, this is accomplished by the commands File>Open data>sample file from the menu bar.¹ Choose *food* from the list. When you bring the file containing the data into gretl your window will look like the one in Figure 2.1. Notice that in the Descriptive label column contains some information about the variables in the program's memory. For some of the datasets included with this book, it may be blank. These descriptions, when they exist, are used by the graphing program to label your output and to help you keep track of variables that are available for use. Before you graph your output or generate results for a report or paper you may want to label your variables to make the output easier to understand. This can be accomplished by editing the attributes of the variables.

To do this, first highlight the variable whose attributes you want to edit, right-click and the menu shown in (see Figure 2.2) appears. Select Edit attributes to open a dialog box (Figure 2.3) where you can change the variable's name, assign variable descriptions and display names. Describe and label the variable y as 'Food Expenditure' and x as 'Weekly Income (\$100).'

You can also bring up the edit attributes dialog from the main window by selecting Variable>Edit attributes. Finally, the setinfo command can be used to set the Description and the label used in graphs.

In the following example a script is generated that opens the *andy.gdt* dataset, and adds variable descriptions, and assigns a label to be used in subsequent graphs.

open "c:\Program Files\gretl\data\poe\andy.gdt" setinfo S -d "Monthly Sales revenue (\$1000)" \

¹Alternately, you could click on the open data button on the toolbar. It's the one that looks like a folder on the far right-hand side.
📓 gretl		_	
File Tools food.gdt ID # IVariabl 0 const 1 y 2 x	Data View Add Sample Variab e name Descriptive label auto-generated constant Food Expenditure Weekly Income (\$100)	e Model Help Display values Descriptive statistics Frequency plot Boxplot Edit attributes Edit values Copy to clipboard Delete Define new variable	
] 🕮 fx 🚺 🔀 💒 ĝ 📄	Undated: Full range 1 - 40	

Figure 2.2: Highlight the desired variable and right-click to bring up the pull-down menu shown here.

Figure 2.3: Variable edit dialog box

6

🛃 gretl: variable attributes							
Name of variable:	У						
Description:							
Weekly Food Expe	nditure						
Display name (show	n in graphs):	Food Expenditure					
		Cancel OK					

📓 gretl: define graph	
	XY scatterplot
У	X-axis variable
×	Choose-> x
	Y-axis variables
	У
	Add ->
	<- Remove
Help Clea	r <u>C</u> ancel <u>O</u> K

Figure 2.4: Use the dialog to plot of the Food Expenditure (y) against Weekly Income (x)

The -d flag is given followed by a string in double quotes. It is used to set the descriptive label. The -n flag is used similarly to set the variable's name in graphs. Notice that in the first and last uses of **setinfo** in the example that I have issued the continuation command (\) since these commands are too long to fit on a single line. If you issue the **labels** command, **gretl** will respond by printing the descriptions to the screen.

2.3 Graph the Data

To generate a graph of the Food Expenditure data that resembles the one in Figure 2.6 of POE, you can use the button on the **gretl** toolbar (third button from the right). Clicking this button brings up a dialog to plot the two variables against one another. Figure 2.4 shows this dialog where x is placed on the x-axis and y on the y-axis. The result appears in Figure 2.5. Notice that the labels applied above now appear on the axes of the graph.

Figure 2.5 plots Food Expenditures on the y axis and Weekly Income on the x. Gretl, by



Figure 2.5: XY plot of the Food Expenditure data

default, also plots the fitted regression line. The benefits of assigning labels to the variables becomes more obvious. Both X- and Y-axes are informatively labeled and the graph title is changed as well. More on this later.

2.4 Estimate the Food Expenditure Relationship

Now you are ready to use **gretl** to estimate the parameters of the Food Expenditure equation.

$$y_t = \beta_1 + \beta_2 x_t + e_t \tag{2.2}$$

From the menu bar, select Model>Ordinary Least Squares from the pull-down menu (see Figure 2.6) to open the dialog box shown in Figure 2.7. From this dialog you'll need to tell gretl which variable to use as the dependent variable and which is the independent variable. Notice that by default, gretl assumes that you want to estimate an intercept (β_1) and includes a constant as an independent variable by placing the variable const in the list by default. To include x as an independent variable, highlight it with the cursor and click the 'Add->' button.

At the question mark in the console simply type

Figure 2.6: From the menu bar, select Model>Ordinary Least Squares to open the least squares dialog box

🛃 gretl	🛛
<u>File Tools Data View Add Sample Variable</u>	Model Help
food.gdt	Ordinary Least Squares
ID # Variable name Descriptive label	Other linear models
0 const auto-generated constant	Nonlinear models
1 y Food Expenditure	Time series
2 × Weekly Income (\$100)	Panel
	Robust estimation
	Maximum likelihood
	Gmm
	2imutaneous equations
	Undated: Full range 1 - 40
μι 🖉 🗋 🔤 ΤΧ 🔁 🔛 🖉 β 📄	

ols y const x

to estimate your regression function. The syntax is very simple, **ols** tells **gret1** that you want to estimate a linear function using ordinary least squares. The first variable listed will be your dependent variable and any that follow, the independent variables. These names must match the ones used in your data set. Since ours in the food expenditure example are named, y and x, respectively, these are the names used here. Don't forget to estimate an intercept by adding a constant (const) to the list of regressors. Also, don't forget that **gret1** is case sensitive so that x and X are different entities.

This yields window shown in Figure 2.9 below. The results are summarized in Table 2.1.

An equivalent way to present results, especially in very small models like the simple linear regression, is to use equation form. In this format, the **gretl** results are:

$$\widehat{\mathbf{y}} = \underset{(1.922)}{83.4160} + \underset{(4.877)}{10.2096} \text{ x}$$

$$T = 40 \quad \overline{R}^2 = 0.3688 \quad F(1, 38) = 23.789 \quad \widehat{\sigma} = 89.517$$

$$(t\text{-statistics in parentheses})$$

Finally, notice in the main **gretl** window (Figure 2.1) that the first column has a heading called ID #. An ID # is assigned to each variable in memory and you can use these ID #s instead of variable names in your programs. For instance, the following two lines yield identical results:

ols y const x

📓 gretl: specify model									
	OLS								
const	Dependent variable								
У	Choose -> y								
×	Set as default								
	Independent variables								
	Add -> X <- Remove								
Robust standard errors configure									
<u>H</u> elp <u>Clear</u> <u>Cancel</u> <u>OK</u>									

Figure 2.7: The Specify Model dialog box opens when you select Model>Ordinary least squares

Figure 2.8: The Gretl console window. From this window you can type in **gretl** commands directly and perform analyses very quickly–if you know the proper **gretl** commands.



Figure 2.9: The model window appears with the regression results. From here you can conduct subsequent operations (graphs, tests, analysis, etc.) on the estimated model.

📓 gretl: model 1					_ 🗆 🔀							
<u> Eile E</u> dit <u>T</u> ests <u>S</u> ave <u>G</u>	įraphs <u>A</u> nalysis <u>L</u> aTeX											
Model 1: OLS estimates using the 40 observations 1-40 Dependent variable: y												
VARIABLE	VARIABLE COEFFICIENT STDERROR T STAT P-VALUE											
const	83.4160	43.4102	1.922	0.06218	*							
x	10.2096	2.09326	4.877	0.00002	* * *							
Mean of dependent Standard deviatio Sum of squared re Standard error of Unadjusted R-squar Adjusted R-squar Degrees of freed Log-likelihood = Akaike informatic Schwarz Bayesian Hannan-Quinn crit	: variable = 283.57 m of dep. var. = 1 ssiduals = 304505 f residuals = 89.51 ared = 0.385002 ed = 0.368818 m = 38 -235.509 m criterion (AIC) criterion (BIC) = terion (HQC) = 476.	4 12.675 7 = 475.018 478.395 239										

Table 2.1: OLS estimates using the 40 observations 1–40.

Dependent variable: y

Variable	Coefficient	Std. Error	$t ext{-st}$	atistic	p-value
const	83.4160	43.4102		1.9216	0.0622
х	10.2096	2.09326		4.8774	0.0000
	Mean of depend	ent variable	283.574		
	S.D. of dependent	nt variable	112.675		
	Sum of squared	residuals	304505.		
	Standard error o	of residuals $(\hat{\sigma})$	89.5170		
	Unadjusted R^2		0.3850	02	
	Adjusted \bar{R}^2		0.3688	18	
	Degrees of freed	om	38		
	Akaike informat	ion criterion	475.018		
	Schwarz Bayesia	n criterion	478.395		

ols 1 0 2

One (1) is the ID number for y and two (2) is the ID number of x. The constant has ID zero (0). If you tend to use long and descriptive variable names (recommended, by the way), using the ID number can save you a lot of typing (and some mistakes).

2.4.1 Elasticity

Elasticity is an important concept in economics. It measures how responsiveness one variable is to changes in another. Mathematically, the concept of elasticity is fairly simple:

$$\varepsilon = \frac{\text{percentage change in } y}{\text{percentage change in } x} = \frac{\Delta y/y}{\Delta x/x}$$
(2.3)

In terms of the regression function, we are interested in the elasticity of average food expenditures with respect to changes in income:

$$\varepsilon = \frac{\Delta E(y)/E(y)}{\Delta x/x} = \beta_2 \frac{x}{E(y)}.$$
(2.4)

E(y) and x are usually replaced by their sample means and β_2 by its estimate. The mean of x and y can be obtained by using the cursor to highlight both variables, use the View>Summary statistics from the menu bar as shown in Figure 2.10, and the computation can be done by hand. However, you can make this even easier by using the gretl language to do all of the computations-no calculator needed! Simply open up a new script and type in:

ols y const x --quiet
genr elast=\$coeff(x)*mean(x)/mean(y)

This yields the output shown in the next figure 2.11:

Following a least squares regression, **Gretl** stores the least squares estimates of the constant and the slope in variables called coeff(const) and coeff(x), respectively. In addition, it uses mean(x) and mean(y) to compute the mean of the variables x and y. The -quiet option is convenient when you don't want or need the output from the regression printed to the screen. The result from this computation appears below in Figure 2.12.

2.4.2 Prediction

Similarly, **gretl** can be used to produce predictions. The predicted food expenditure of an average household having weekly income of \$2000 is:

$$\hat{y}_t = 83.42 + 10.21x_t = 83.42 + 10.21(20) = 287.61 \tag{2.5}$$

Figure 2.10: Using the pull-down menus to obtain summary statistics. Highlight the desired variables and use View>Summary statistics from the pull-down menu.

File Jools Qata Yew Add Sample Yariable Model Help D.# Variable namet Graph specified vars Multiple graphs Multiple graphs	ľ	gr 🛛	etl							
Undated: Full range 1 - 40		Eile food. ID # 0 1 2	Iools gdt Variable γ χ	Data name D	View Icon v Graph Multipl Correl Cross Princip Mahala Cross	Add iew specifi e grapi ation m I_abulai ation m I_abulai comreio	Sample ed vars is istics atrix cion ponents distances gram	Variable	Model	Heb
			2		fx	1) <u>_</u>		Undated	I: Full range 1 - 40

Figure 2.11: Summary statistics

💐 gretl: summa	ry statistics				
Z 4 9	TEX X				
	Summary Stat	istics, using the	observations 1	- 40	
Variable	MEAN	MEDIAN	MIN	MAX	
У ×	283.57 19.605	264.48 20.030	109.71 3.6900	587.66 33.400	
Variable	s.D.	c.v.	SKEW	EXCSKURT	
У Х	112.68 6.8478	0.39734 0.34929	0.49208 -0.62651	-0.14848 0.27973	

Figure 2.12: Results from the script to compute an elasticity based on a linear regression.

🛃 gretl console	
<pre>? ols y const xquiet ? genr elast=\$coeff(x)*mean(x)/mean(y) Replaced scalar elast = 0.70584 ?</pre>	

Remember, income is measured in \$100, so 20 in the above expression represents 20*\$100=\$2,000. The **gretl** script is:

genr yhat = \$coeff(const) + \$coeff(x)*20

which yields the desired result.

2.4.3 Estimating Variance

In section 2.7 of your textbook, you are given expressions for the variances of the least squares estimators of the intercept and slope as well as their covariance. These estimators require that you estimate the overall variance of the model's errors, σ^2 . Gret1 does not explicitly report the estimator, $\hat{\sigma}^2$, but rather, its square root, $\hat{\sigma}$. Gret1 calls this "Standard error of residuals" which you can see from the output is 89.517. Thus, $89.517^2 = 8013.29$. Gret1 also reports the sum of squared residuals, equal to 304505, from which you can calculate the estimate. Dividing the sum of squared residuals by the estimator's degrees of freedom yields $\hat{\sigma}^2 = 304505/38 = 8013.29$.

The estimated variances and covariance of the least squares estimator can be obtained once the model is estimated by least squares by selecting the Analysis>Coefficient covariance matrix command from the pull-down menu of the model window as shown in Figure 2.13. The result is:

Covariance matrix of regression coefficients

const x 1884.44 -85.9032 const 4.38175 x

So, estimated variances of the least squares estimator of the intercept and slope are 1884.44 and 4.38175, respectively. The least squares standard errors are simply the square roots of these numbers. The estimated covariance between the slope and intercept -85.9032.

You can also obtain the variance-covariance matrix by specifying the -vcv option when estimating a regression model. For the food expenditure example use:

ols y const x --vcv

to estimate the model using least squares and to print the variance covariance matrix to the results window.

2.5 Repeated Sampling

Perhaps the best way to illustrate the sampling properties of least squares is through an experiment. In section 2.4.3 of your book you are presented with results from 10 different regressions (POE Table 2.2). These were obtained using the dataset *table2-2.gdt* which is included in the **gret1** datasets that accompany this manual. To reproduce the results in this table estimate 10 separate regressions

```
ols y1 const x
ols y2 const x
.
.
ols y10 const x
```

You can also generate your own random samples and conduct a Monte Carlo experiment using **gretl**. In this exercise you will generate 100 samples of data from the food expenditure data, estimate the slope and intercept parameters with each data set, and then study how the least squares estimator performed over those 100 different samples. What will become clear is this, the outcome from any single sample is a poor indicator of the true value of the parameters. Keep this humbling thought in mind whenever you estimate a model with what is invariably only 1 sample or instance of the true (but always unknown) data generation process.

We start with the food expenditure model:

$$y_t = \beta_1 + \beta_2 x_t + e_t \tag{2.6}$$

where y_t is total food expenditure for the given time period and x_t is income. Suppose further that we know how much income each of 40 households earns in a week. Additionally, we know that on average a household spends at least \$80 on food whether it has income or not and that an average household will spend ten cents of each new dollar of income on additional food. In terms of the regression this translates into parameter values of $\beta_1 = 80$ and $\beta_2 = 10$.

Our knowledge of any particular household is considerably less. We don't know how much it actually spends on food in any given week and, other than differences based on income, we don't know how its food expenditures might otherwise differ. Food expenditures are sure to vary for reasons other than differences in family income. Some families are larger than others, tastes and preferences differ, and some may travel more often or farther making food consumption more costly. For whatever reasons, it is impossible for us to know beforehand exactly how much any household will spend on food, even if we know how much income it earns. All of this uncertainty is captured by the error term in the model. For the sake of experimentation, suppose we also know that $e_t \sim N(0, 88^2)$.

With this knowledge, we can study the properties of the least squares estimator by generating samples of size 40 using the known data generation mechanism. We generate 100 samples using the known parameter values, estimate the model for each using least squares, and then use summary statistics to determine whether least squares, on average anyway, is either very accurate or precise. So in this instance, we know how much each household earns, how much the **average** household spends on food that is not related to income ($\beta_1 = 80$), and how much that expenditure rises **on average** as income rises. What we do not know is how any **particular** household's expenditures responds to income or how much is autonomous.

A single sample can be generated in the following way. The systematic component of food expenditure for the t^{th} household is $80 + 10 * x_t$. This differs from its actual food expenditure by a random amount that varies according to a normal distribution having zero mean and standard deviation equal to 88. So, we use computer generated random numbers to generate a random error, u_t , from that particular distribution. We repeat this for the remaining 39 individuals. The generates one Monte Carlo sample and it is then used to estimate the parameters of the model. The results are saved and then another Monte Carlo sample is generated and used to estimate the model and so on.

In this way, we can generate as many different samples of size 40 as we desire. Furthermore, since we know what the underlying parameters are for these samples, we can later see how close our estimators get to revealing these true values.

Now, computer generated sequences of random numbers are not actually random in the true sense of the word; they can be replicated exactly if you know the mathematical formula used to generate them and the 'key' that initiates the sequence. In most cases, these numbers *behave as if* they randomly generated by a physical process.

To conduct an experiment using least squares in gretl use the script found in Figure 2.14.

Let's look at what each line accomplishes. The first line

open "c:\Program Files\gretl\data\poe\food.gdt"

opens the food expenditure data set that resides in the *poe* folder of the data directory. The next line, which is actually not necessary to do the experiments, estimates the model using the original data using **ols**. It is included here so that you can see how the results from the actual sample compare to those generated from the simulated samples. All of the remaining lines are used for the Monte Carlo.

In Monte Carlo experiments loops are used to estimate a model using many different samples that the experimenter generates and to collect the results. The loop construct in **gretl** begins with the command loop NMC --progressive and ends with endloop. NMC in this case is the number of Monte Carlo samples you want to use and the option --progressive is a command that suppresses the individual output at each iteration from being printed and allows you to store the results in a file. So that you can reproduce the results below, I have also initiated the sequence of random numbers using a key, called the *seed*, with the command set seed 3213798. Basically, this ensures that the stream of pseudo random numbers will start at the same place each time you run your program. Try changing the value of the seed (3213798) or the number of Monte Carlo iterations (100) to see how your results are affected.

Within this loop construct, you tell **gretl** how to generate each sample and state how you want that sample to be used. The data generation is accomplished here as

genr u = 88*normal()
genr y1 = 80 + 10*x + u

The genr command is used to generate new variables. In the first line u is generated by multiplying a normal random variable by the desired standard deviation. Recall, that for any constant, c and random variable, X, $Var(cX) = c^2 Var(X)$. The gret1 command normal() produces a computer generated standard normal random variable. The next line adds this random element to the systematic portion of the model to generate a new sample for food expenditures (using the known values of income in x).

Next, the model is estimated using least squares. Then, the coefficients are stored internally in variables you create b1 and b2 (I called them b1 and b2, but you can name them as you like). These are then stored to a data set coeff.gdt.

After executing the script, **gretl** prints out some summary statistics to the screen. These appear below in Figure 2.15. Note that the average value of the intercept is about 76.5950. This is getting close to the truth. The average value of the slope is 10.1474, also reasonably close to the true value. If you were to repeat the experiments with larger numbers of Monte Carlo iterations, you will find that these averages get closer to the values of the parameters used to generate the data. This is what it means to be unbiased. Unbiasedness only has meaning within the context of repeated sampling. In your experiments, you generated many samples and averaged results over those samples to get closer to the truth. In actual practice, you do not have this luxury; you have one sample and the proximity of your estimates to the true values of the parameters is always unknown.

After executing the script, open the coeff.gdt data file that gret1 has created and view the data. From the menu bar this is done using File>Open data>user file and selecting coeff.gdt from the list of available data sets. From the example this yields the output in Figure 2.16. Notice that even though the actual value of $\beta_1 = 80$ there is considerable variation in the estimates. In sample 21 it was estimated to be 170.8196 and in sample 19 it was 1.3003. Likewise, β_2 also varies around its true value of 10. Notice that for any given sample, an estimate is never equal to the

true value of the parameter!

2.6 Script

The script for Chapter 2 is found below. These scripts can also be found at my website http: //www.learneconometrics.com/gretl.

```
open "c:\Program Files\gretl\data\poe\food.gdt"
#Least squares
ols y const x --vcv
#Summary Statistics
summary y x
#Plot the Data
gnuplot y x
#List the Data
print y x --byobs
#Elasticity
genr elast=$coeff(x)*mean(x)/mean(y)
#Prediction
genr yhat = $coeff(const) + $coeff(x)*20
#Table 2.2
open "c:\Program Files\gretl\data\poe\table2-2.gdt"
ols y1 const x
ols y2 const x
ols y3 const x
ols y4 const x
ols y5 const x
ols y6 const x
ols y7 const x
ols y8 const x
ols y9 const x
ols y10 const x
#Monte Carlo
open "c:\Program Files\gretl\data\poe\food.gdt"
```

```
set seed 3213789
loop 100 --progressive --quiet
genr u = 88*normal()
genr y1 = 80 + 10*x + u
ols y1 const x
genr b1 = $coeff(const)
genr b2 = $coeff(x)
print b1 b2
store coeff.gdt b1 b2
endloop
```

Figure 2.13: Obtain the matrix that contains the least squares estimates of variance and covariance from the pull-down menu of your estimated model.

I	🚺 Mo	del 1															_ 0	X
Γ	Eile	<u>E</u> dit	<u>T</u> ests	<u>S</u> ave	<u>G</u> raphs	<u>A</u> nalysis	<u>L</u> aTeX											
						Display ac	tual, fitte	ed, res	idual									
	Mode	21 1	:			Eorecasts												
	OLS	est	imate	s usi	ng the	⊆onfidenc	e interval	ls for o	coeffici	ents								
	Depe	ende	nt va	riabl	e: y	Confidenc	e <u>e</u> llipse											
						Coefficier	it covariar	nce <u>m</u> a	atrix									
		V	ARIAB	LE	cc	ANOVA				N	1	Г	STAT	P	-VALUE			
						Bootstrap								_				
	co	onst				03.4100	,		43.	4102			1.922	0	.06218	*		
	×					10.2090	2		2.	09326	5		4.877	U	.00002	* * *		
	Ме	an	of de	pende	nt var	iable =	- 283.	574										
	St	and	ard d	eviat	ion of	dep. v	/ar. =	112	2.675	;								≣
	Sı	un o	f squ	ared	residu	als = 3	04505											
	St	and	ard e	rror	of res	iduals	= 89.	517										
	Uı	nadj	usted	R-sc	uared	= 0.385	5002											
	Ac	ijus	ted R	-squa	red =	0.3688:	٤8											
	De	egre	es of	free	dom =	38												
	Log-likelihood = -235.509																	
	Akaike information criterion (AIC) = 475.018																	
	Schwarz Bayesian criterion (BIC) = 478.395																	
		anna	n-Qui	nn cr	iterio	n (HQC)	= 47	6.23	99									
																		~

Figure 2.14: In the **gretl** script window you can type in the following commands to execute a Monte Carlo study of least squares. Then to execute the program, click on the small gear icon.

gretl: POEch2.inp	- 🗆 🛛
<pre>#Monte Carlo open "i:\Program Files\gretl\data\poe\food.gdt"</pre>	•
<pre>set seed 3213789 loop 100 progressive genr u = 88*normal() genr y1 = 80 + 10*x + u ols y1 const x rear b1 = \$coeff(const)</pre>	
<pre>genr b1 = \$coeff(x) print b1 b2 store coeff.gdt b1 b2 endloop</pre>	

Figure 2.15: The summary results from 100 random samples of the Monte Carlo experiment.



Figure 2.16: The results from the first 23 sets of estimates from the 100 random samples of the Monte Carlo experiment.

📓 gretl: disp	lay data	_	
1 L G	🔍 🟪 😹		
Obs	b1	b2	
1	70.1400	9.46787	
2	90.9485	10.14972	=
3	80.7360	10.32389	
4	112.1177	9.10936	
5	132.8950	7.81551	
6	53.6029	10.48355	
7	60.5105	11.49058	
8	84.3333	9.18269	
9	10.8011	13.92074	
10	89.0575	9.66496	
11	132.0336	6.77509	
12	126.4581	7.99526	
13	40.7848	11.25006	
14	106.0379	9.01178	
15	63.3129	10.05348	
16	31.9721	12.95446	
17	103.7510	8.31367	
18	58.8011	10.73272	
19	1.3003	13.27228	
20	44.3346	12.73427	
21	170.8196	6.70784	
22	20.4066	13.09706	_
23	91.6029	9.55563	~

Chapter 3

Interval Estimation and Hypothesis Testing

In this chapter, I will discuss how to generate confidence intervals and test hypotheses using **gretl**. **Gretl** includes several handy utilities that will help you obtain critical values and p-values from several important probability distributions. As usual, you can use the dialog boxes or **gretl**'s programming language to do this.

3.1 Confidence Intervals

It is important to know how precise your knowledge of the parameters is. One way of doing this is to look at the least squares parameter estimate along with a measure of its precision, i.e., its estimated standard error.

The confidence interval serves a similar purpose, though it is much more straightforward to interpret because it gives you upper and lower bounds between which the unknown parameter will lie with a given probability.¹

In gretl you can obtain confidence intervals either through a dialog or by manually building them using saved regression results. In the 'manual' method I will use the genr command to generate upper and lower bounds based on regression results that are saved in gretl's memory, letting gretl do the arithmetic. You can either look up the appropriate critical value from a table or use the gretl's critical function. I'll show you both.

¹This is probability in the frequency sense. Some authors fuss over the exact interpretation of a confidence interval (unnecessarily I think). You are often given stern warnings not to interpret a confidence interval as containing the unknown parameter with the given probability. However, the frequency definition of probability refers to the long run relative frequency with which some event occurs. If this is what probability is, then saying that a parameter falls within an interval with given probability means that intervals so constructed will contain the parameter that proportion of the time.

Here is how it works. Consider equation (3.5) from your text

$$P[b_2 - t_c se(b_2) \le \beta_2 \le b_2 + t_c se(b_2)] = 1 - \alpha$$
(3.1)

Recall that b_2 is the least squares estimator of β_2 , and that $se(b_2)$ is its estimated standard error. The constant t_c is the $\alpha/2$ critical value from the t-distribution and α is the total desired probability associated with the "rejection" area (the area outside of the confidence interval).

You'll need to know t_c , which can be obtained from a statistical table, the Tools>Statistical tables dialog contained in the program, or using the gretl command critical. First, try using the dialog box shown in Figure 3.1. Pick the tab for the t distribution and tell gretl how much weight to put into the right-tail of the probability distribution and how many degrees of freedom your t-statistic has, in our case, 38. Once you do, click on OK. You'll get the result shown in Figure 3.2. It shows that for the t(38) with right-tail probability of 0.025 and two-tailed probability of 0.05, the critical value is $2.02439.^2$ Then generate the lower and upper bounds (using the gretl

Figure 3.1: Obtaining critical values using the Tools>Statistical tables dialog box.

📓 gretl: critical values 🛛 🗖 🗖 🔀
normal t chi-square F binomial poisson DW
df 38 right-tail probability .025

Figure 3.2: The critical value obtained from Tools>Statistical tables dialog box.



console) with the commands:

open "c:\Program Files\gretl\data\poe\food.gdt"

²You can also get the α level critical values from the console by issuing the command genr c= critical(t,38, α). Here α is the desired area in the right-tail of the *t*-distribution.

```
ols y const x
genr lb = $coeff(x) - 2.024 * $stderr(x)
genr ub = $coeff(x) + 2.024 * $stderr(x)
print lb ub
```

The first line opens the data set. The second line (ols) minimizes the sum of squared errors in a linear model that has y as the dependent variable with a constant and x as independent variables. The next two lines generate the lower and upper bounds for the 95% confidence interval for the slope parameter β_2 . The last line prints the results of the computation.

The gretl language syntax needs a little explanation. When gretl makes a computation, it will save certain results like coefficient estimates, their standard errors, sum of squared errors and so on in memory. These results can then be used to compute other statistics, provided you know the variable name that gretl uses to store the computation. In the above example, gretl uses the least squares estimates and their estimated standard errors to compute confidence intervals. Following the ols command, least squares estimates are stored in coeff(variable name). So, since β_2 is estimated using the variable x, its coefficient estimate is saved in coeff(x). The corresponding standard error is saved in stderr(x). There are many other results saved and stored, each prefixed with the dollar sign coeff. Consult the gretl documentation for more examples and specific detail on what results can be saved and how to access them.

Equivalently, you could use **gretl**'s built in **critical** to obtain the desired critical value. The general syntax for the function depends on the desired probability distribution. This follows since different distributions contain different numbers of parameters (e.g., the t-distribution has a single degrees of freedom parameter while the standard normal has none!). This example uses the t-distribution and the script becomes:

```
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
print lb ub
```

The syntax for the t-distribution is critical(t,degrees of freedom, $\alpha/2$). The degrees of freedom from the preceding regression are saved as \$df and for a $1 - \alpha = 95\%$ confidence interval, $\alpha/2 = 0.025$.

The example found in section 3.1.3 of *POE* computes a 95% confidence interval for the income parameter in the food expenditure example. The **gretl** commands above were used to produce the output found in Figure 3.3.

To use the dialogs to get confidence intervals is easy as well. First estimate the model using least squares in the usual way. Choose Model>Ordinary least squares from the main pull-down menu, fill in the dependent and independent variables in the ols dialog box and click OK. The results

Figure 3.3: Obtaining 95% confidence interval for the food expenditure example.

📓 gretl: script output					<u>- 🗆 ×</u>	
	₽≩ 👆 ×					
Model 1: OLS esti Dependent variabl	mates using the 40 c le: y	bservations 1-	40		_	
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE		
const x	83.4160 10.2096	43.4102 2.09326	1.922 4.877	0.06218 * 0.00002 ***		
Mean of dependent variable = 283.574 Standard deviation of dep. var. = 112.675 Sum of squared residuals = 304505 Standard error of residuals = 89.517 Unadjusted R-squared = 0.385002 Adjusted R-squared = 0.368818 Degrees of freedom = 38 Log-likelihood = -235.509 Akaike information criterion (AIC) = 475.018 Schwarz Bayesian criterion (BIC) = 478.395 Hannan-Quinn criterion (HQC) = 476.239						
Generated scalar ? genr ub = \$coef Generated scalar	lb (ID 3) = 5.97288 f(x) + 2.024 * \$stde ub (ID 4) = 14.4464	err(x)				
? print 1b ub						
1b = ub =	5.97288 14.4464				•	
		Close				

appear in the model window. Now choose Analysis>Confidence intervals for coefficients from the model window's pull-down menu (seen in Figure 4.1). This generates the result shown in Figure 3.4.

Figure 3.4: Obtaining 95% confidence interval for the food expenditure example from the dialog.

l	🕷 gretl: coefficient confide	ence intervals		
	🎇 📇 🖬 🔍 🕸 🗙			
	t(38, .025) = 2.024			
	VARIABLE	COEFFICIENT	95% CONFIDENCE INTERVAL	
	const	83.4160	(-4.46328, 171.295)	
	х	10.2096	(5.97205, 14.4472)	

3.2 Monte Carlo Experiment

Once again, the consequences of repeated sampling can be explored using a simple Monte Carlo study. In this case, we will add the two statements that compute the lower and upper bounds

to our previous program listed in Figure 2.14. We've also added a parameter, sig2, which is the estimated variance of the model's errors.

The new script looks like this:

```
open "c:\Program Files\gretl\data\poe\food.gdt"
set seed 3213798
loop 100 --progressive --quiet
genr u = 88*normal()
genr y1 = 80 + 10*x + u
ols y1 const x
genr b1 = $coeff(const)
genr b2 = coeff(x)
genr s1 = $stderr(const)
genr s2 = $stderr(x)
# 2.024 is the .025 critical value from the t(38) distribution
genr c1L = b1 - critical(t,df,.025)*s1
genr c1R = b1 + critical(t, df, .025) * s1
genr c2L = b2 - critical(t,df,.025)*s2
genr c2R = b2 + critical(t,df,.025)*s2
# Compute the coverage probabilities of the Confidence Intervals
genr p1 = (80>c1L && 80<c1R)
genr p2 = (10>c2L && 10<c2R)
genr sigma = $sigma
genr sig2 = sigma*sigma
print b1 b2 p1 p2
store cicoeff.gdt b1 b2 s1 s2 sig2 c1L c1R c2L c2R
endloop
```

The results are stored in the gretl data set cicoeff.gdt. Opening this data set (open "c: $\ProgramFiles\gretl\user\cicoeff.gdt"$) and examining the data will reveal interval estimates that vary much like those in Tables 3.1 and 3.2 of your textbook. Also, notice that the estimated standard error of the residuals, $\hat{\sigma}$, is stored in \$sigma and used to estimate the overall model variance.

3.3 Hypothesis Tests

Hypothesis testing allows us to confront any prior notions we may have about the model with what we actually observe. Thus, if before drawing a sample, I believe that autonomous weekly food expenditure is no less than \$40, then once the sample is drawn I can determine via a hypothesis test whether experience is actually consistent with this belief.

In section 3.4 of your textbook the authors test several hypotheses about β_2 . In 3.4.1a the null hypothesis is that $\beta_2 = 0$ against the alternative that it is positive (i.e., $\beta_2 > 0$). The test statistic is:

$$t = (b_2 - 0)/se(b_2) \sim t_{38} \tag{3.2}$$

provided that $\beta_2 = 0$ (the null hypothesis is true). Select $\alpha = 0.05$ which makes the critical value for the one sided alternative ($\beta_2 > 0$) equal to 1.686. The decision rule is to reject Ho in favor of the alternative if the computed value of your t statistic falls within the rejection region of your test; that is if it is larger than 1.686.

The information you need to compute t is contained in the least squares estimation results produced by gretl:

Model 1:	OLS estimates using the 40 observations $1-40$
	Dependent variable: y

Variable	Coefficient	Std. Error	t-statistic	p-value
const	83.4160	43.4102	1.9216	0.0622
х	10.2096	2.09326	4.8774	0.0000

Mean of dependent variable	283.574
S.D. of dependent variable	112.675
Sum of squared residuals	304505.
Standard error of residuals $(\hat{\sigma})$	89.5170
Unadjusted R^2	0.385002
Adjusted \bar{R}^2	0.368818
Degrees of freedom	38
Akaike information criterion	475.018
Schwarz Bayesian criterion	478.395

The computations

 $t = (b_2 - 0)/se(b_2) = (10.21 - 0)/2.09 = 4.88$ (3.3)

Since this value is within the rejection region, then there is enough evidence at the 5% level of significance to convince us that the null hypothesis is incorrect; the null hypothesis rejected at this level of significance. We can use **gretl** to get the p-value for this test using the **Tools** pull-down menu. In this dialog, you have to fill in the degrees of freedom for your t-distribution (38), the value of b_2 (10.21), its value under the null hypothesis–something **gretl** refers to as 'mean' (0), and the estimated standard error from your printout (2.09). This will yield the information:

```
t(38): area to the right of 10.21 = 9.55024e-013
(two-tailed value = 1.91005e-012; complement = 1)
```

📓 gretl: p-value finder 💦 💷 🔀					
normal t chi-square F gamma binomial poisson					
df 38 value 10.21					
🗶 <u>C</u> lose 🥔 <u>O</u> K					

Figure 3.5: The dialog box for obtaining p-values using the built in statistical tables in gretl.

This indicates that the area in one tail is almost zero. The p-value is well below the usual level of significance, $\alpha = .05$, and the hypothesis is rejected.

Gretl also includes a programming command that will compute p-values from several distributions. The **pvalue** function works similarly to the **critical** function discussed in the preceding section. The syntax is:

genr p = pvalue(distribution,parameters,xval)

The pvalue function computes the area to the right of xval in the specified distribution. Choices include z for Gaussian, t for Student's t, X for chi-square, F for F, G for gamma, B for binomial or P for Poisson. The argument parameters refers to the distribution's known parameters, like its degrees of freedom. So, for this example try

```
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
genr t2 = ($coeff(x)-0)/$stderr(x)
genr p2 = pvalue(t,$df,t2)
```

In the next example, the authors of *POE* test the hypothesis that $\beta_2 = 5.5$ against the alternative that $\beta_2 > 5.5$. The computations

$$t = (b_2 - 5.5)/se(b_2) = (10.21 - 5.5)/2.09 = 2.25$$
(3.4)

The significance level in this case is chosen to be 0.01 and the corresponding critical value can be found using a tool found in **Gretl**. The Tools>Statistical tables pull-down menu bring up the dialog found in Figure 3.1.

This result is found in Figure 3.6. The .01 one-sided critical value is 2.429. Since 2.25 is less

Figure 3.6: The results from the dialog box for obtaining critical values using the built in statistical tables in **gretl**.

📓 gretl: critical values 🛛 🗐 🖂
t(38) right-tail probability = 0.01 complementary probability = 0.99 two-tailed probability = 0.02 Critical value = 2.42857

than this, we cannot reject the null hypothesis at the .01 level of significance.

In section 3.4.2 of *POE*, the authors conduct a one-sided test where the rejection region falls within the left tail of the t-distribution. The null hypothesis is $\beta_2 = 15$ and the alternative is $\beta_2 < 15$. The test statistic and distribution is

$$t = (b_2 - 15)/se(b_2) \sim t_{38} \tag{3.5}$$

provided that $\beta_2 = 15$. The computation is

$$t = (b_2 - 15)/se(b_2) = (10.21 - 15)/2.09 = -2.29$$
(3.6)

Based on the desired level of significance, $\alpha = .05$, we would reject the null in favor of the one-sided alternative if t < -1.686. It is and therefore we conclude that the coefficient is less than 15 at this level of significance.

In section 3.4.3 examples of two-tailed tests are found. In the first example the economic hypothesis that households will spend \$7.50 of each additional \$100 of income on food. So, H_0 : $\beta_2 = 7.50$ and the alternative is $H_1: \beta_2 \neq 7.50$. The statistic is $t = (b_2 - 7.5)/se(b_2) \sim t_{38}$ if H_0 is true which is computed $t = (b_2 - 7.5)/se(b_2) = (10.21 - 7.5)/2.09 = 1.29$. The two-sided, $\alpha = .05$ critical value is 2.024. This means that you reject H_0 if either t < -2.024 or if t > 2.024. The computed statistic is neither, and hence we do not reject the hypothesis that β_2 is \$7.50. There simply isn't enough information in the sample to convince us otherwise.

You can draw the same conclusions from using a confidence interval that you can from this two-sided t-test. The $100(1 - \alpha)\%$ confidence interval for β_2 is

$$b_2 - t_c se(b_2) \le \beta_2 \le b_2 + t_c se(b_2) \tag{3.7}$$

In terms of the example

$$10.21 - 2.024(2.09) \le \beta_2 \le 10.21 + 2.024(2.09) \tag{3.8}$$

which as we saw earlier in the manual was $5.97 \le \beta_2 \le 14.45$. Since 7.5 falls within this interval, you could not reject the hypothesis that β_2 is different from 7.5 at the .05% level of significance.

In the next example a test of the overall significance of β_2 is conducted. As a matter of routine, you always want to test to see if your slope parameter is different from zero. If not, then the variable associated with it may not belong in your model. So, $H_0: \beta_2 = 0$ and the alternative is $H_1: \beta_2 \neq 0$. The statistic is $t = (b_2 - 0)/se(b_2) \sim t_{38}$, if H_0 is true, and this is computed $t = (b_2 - 0)/se(b_2) = (10.21 - 0)/2.09 = 4.88$. Once again, the two-sided, $\alpha = .05$ critical value is 2.024 and 4.88 falls squarely within the 5% rejection region of this test. These numbers should look familiar since this is the test that is conducted by default whenever you run a regression in Gretl.

As we saw earlier, **gretl** also makes obtaining one- or two-sided p-values for the test statistics you compute very easy. Simply use p-value finder dialog box available from the Tools pull-down menu (see Figure 3.6) to obtain one or two sided p-values.

3.4 Script for t-values and p-values

One thing we've shown in this chapter is that many of the results obtained using the pull-down menus (often referred to as the GUI) in **gretl** can be obtained using **gretl**'s language from the console or in a script. In fact, the **gretl**'s GUI is merely a front-end to its programming language.³ In this chapter we used the **pvalue** and **critical** functions to get p-values or critical values of statistics. The following script accumulates what we've covered and completes the examples in the text.

```
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
genr tratio1 = ($coeff(x) - 0)/ $stderr(x)
#One sided test (Ha: b2 > zero)
genr c2 = critical(t,$df,.05)
genr p2 = pvalue(t,$df,tratio1)
#One sided test (Ha: b2>5.5)
genr tratio2 = ($coeff(x) - 5.5)/ $stderr(x)
genr c2 = critical(t,$df,.05)
genr p2 = pvalue(t,$df,tratio2)
#One sided test (Ha: b2<15)
genr tratio3 = ($coeff(x) - 15)/ $stderr(x)
genr c3 = -1*critical(t,$df,.05)
genr p3 = pvalue(t,$df,abs(tratio3))</pre>
```

³This is true in Stata as well.

```
#Two sided test (Ha: b2 not equal 7.5)
genr tratio4 = ($coeff(x) - 7.5)/ $stderr(x)
genr c4 = critical(t,$df,.025)
genr p4 = 2*pvalue(t,$df,tratio4)
#Confidence interval
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
#Two sided test (Ha: b2 not equal zero)
genr c1 = critical(t,$df,.025)
genr p1 = 2*pvalue(t,$df,tratio5)
```

The **pvalue** function in **gret** measures the area of the probability distribution that lies to the right of the computed statistic. If the computed t-ratio is positive and your alternative is two-sided, multiply the result by 2 to measure the area to the left of its negative.

If your t-ratio is negative, gretl won't compute the area (and you wouldn't want it to, anyway). This is what happened for tratio3 in the script and I used the absolute value function, abs(), to get its positive value. The area to the right of the positive value is equivalent to the area left of the negative value. Hence, the computation is correct.

Basically, proper use of the **pvalue** in tests of a single hypothesis requires a little thought. Too much thought, in my opinion. I would avoid it unless you are comfortable with its use. In other hypothesis testing contexts (e.g., χ^2 and F-tests) p-values are much easier to use correctly. I use them freely in those cases. With t-tests or z-tests (normal distribution), it is just easier conduct a test by comparing the computed value of your statistic to the correct critical value.

3.5 Script

```
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
genr tratio1 = (\text{scoeff}(x) - 0)/ \text{stderr}(x)
#One sided test (Ha: b2 > zero)
genr c2 = critical(t,$df,.05)
genr p2 = pvalue(t,$df,tratio1)
#One sided test (Ha: b2>5.5)
genr tratio2 = (\text{scoeff}(x) - 5.5)/ \text{stderr}(x)
genr c2 = critical(t,$df,.01)
genr p2 = pvalue(t,$df,tratio2)
#One sided test (Ha: b2<15)
genr tratio3 = (\text{scoeff}(x) - 15)/ \text{stderr}(x)
genr c3 = -1*critical(t,$df,.05)
genr p3 = pvalue(t,$df,abs(tratio3))
#Two sided test (Ha: b2 not equal 7.5)
genr tratio4 = (\text{scoeff}(x) - 7.5)/ \text{stderr}(x)
genr c4 = critical(t,df,.025)
genr p4 = 2*pvalue(t,$df,tratio4)
#Confidence interval
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
#Two sided test (Ha: b2 not equal zero)
genr c1 = critical(t, df, .025)
genr p1 = 2*pvalue(t,$df,tratio1)
```

And for the Monte Carlo experiment, the script is:

```
open "c:\Program Files\gretl\data\poe\food.gdt"
set seed 3213798
loop 100 --progressive --quiet
genr u = 88*normal()
genr y1 = 80 + 10*x + u
ols y1 const x
genr b1 = $coeff(const)
genr b2 = $coeff(x)
genr s1 = $stderr(const)
```

```
genr s2 = $stderr(x)
# POE uses 2.024 for the .025 critical value from the t(38) distribution
genr c1L = b1 - critical(t,$df,.025)*s1
genr c1R = b1 + critical(t,$df,.025)*s1
genr c2L = b2 - critical(t,$df,.025)*s2
genr c2R = b2 + critical(t,$df,.025)*s2
# Compute the coverage probabilities of the Confidence Intervals
genr p1 = (80>c1L && 80<c1R)
genr p2 = (10>c2L && 10<c2R)
genr sigma = $sigma
genr sig2 = sigma*sigma
print b1 b2 p1 p2
store cicoeff.gdt b1 b2 s1 s2 sig2 c1L c1R c2L c2R
endloop</pre>
```

Chapter

Prediction, Goodness-of-Fit, and Modeling Issues

Several extensions of the simple linear regression model are now considered. First, conditional predictions are generated using results saved by **gretl**. Then, a commonly used measure of the quality of the linear fit provided by the regression is discussed. We then take a brief detour to discuss how **gretl** can be used to provide professional looking output that can be used in your research.

The choice of functional form for a linear regression is important and the RESET test of the adequacy of your choice is examined. Finally, the residuals are tested for normality. Normality of the model's errors is a useful property in that, when it exists, it improves the the performance of least squares and the related tests and confidence intervals we've considered when sample sizes are small (finite).

4.1 Prediction in the Food Expenditure Model

Generating predicted values of food expenditure for a person with a given income is very simple in **gretl**. After estimating the model with least squares, you can use the **genr** to get predicted values. In the example, a household having $x_o = \$2000$ of weekly income is predicted to spend approximately \$287.61 on food. Recalling that income is measured in hundreds of dollars in the data, the **gretl** commands to compute this from the console are:

```
ols y const x
genr yhat0 = $coeff(const) + $coeff(x)*20
```

This yields $\hat{y}_0 = 287.609$.

Obtaining the 95% confidence interval is slightly harder in that there are no internal commands in **gretl** that will do this. The information needed is readily available, however. The formula is:

$$v\hat{a}r(f) = \hat{\sigma}^2 + \frac{\hat{\sigma}^2}{T} + (x_o - \bar{x})^2 v\hat{a}r(b_2)$$
(4.1)

In section 2.4 we estimated $\hat{\sigma}^2 = 8013.29$ and $v\hat{a}r(b_2) = 4.3818$. The mean value of income is found by highlighting the variable x in the main **gretl** window and the selecting View>Summary Statistics from the pull-down menu. This yields $\bar{x} = 19.6047$.¹ The t_{38} 5% critical value is 2.0244 and the computation²

$$v\hat{a}r(f) = 8013.2941 + \frac{8013.2941}{40} + (20 - 19.6047)^2 * 4.3818 = 8214.31$$
(4.2)

Then, the confidence interval is:

$$\hat{y}_0 \pm t_c se(f) = 287.6069 \pm 2.0244\sqrt{8214.31} = [104.132, 471.086]$$
(4.3)

The complete script to produce the computed results in **gretl** is:

```
ols y const x
genr yhat0 = $coeff(const) + $coeff(x)*20
genr f=8013.2941+(8013.2941/40)+4.3818*(20-19.6047)**2
genr ub=yhat0+2.0244*sqrt(f)
genr lb=yhat0-2.0244*sqrt(f)
```

At this point, you may be wondering if there is some way to use the internal functions of **gretl** to produce the same result? As we've seen, **gretl** saves many of the results we need internally and these can in turn be called into service in subsequent computations.

For instance, the sum of squared errors from the least squares regression is saved as **\$ess**. The degrees of freedom and number of observations are saved as **\$df** and **\$nobs**, respectively. Also, you can use an internal **gretl** function to compute \bar{x} , **mean(x)** and the **critical** function discussed in the preceding chapter to get the desired critical value. Hence, the prediction interval can be automated and made more precise by using the following script.

```
ols y const x
genr yhat0=$coeff(const)+20*$coeff(x)
genr sig2 = $ess/$df
genr f = sig2 + sig2/$nobs + ((20-mean(x))^2)*($stderr(x)^2)
genr lb = yhat0-critical(t,$df,0.025)*sqrt(f)
genr ub = yhat0+critical(t,$df,0.025)*sqrt(f)
```

¹Your result may vary a little depending on how many digits are carried out to the right of the decimal. ²You can compute this easily using the **gretl** console by typing in: **genr f=8013.2941+(8013.2941/40)+4.3818*(20-19.6047)**2**

4.2 Coefficient of Determination

One use of regression analysis is to "explain" variation in dependent variable as a function of the independent variable. A summary statistic that is used for this purpose is the coefficient of determination, also known as R^2 .

There are a number of different ways of obtaining R^2 in gret1. The simplest way to get R^2 is to read it directly off of gret1's regression output. This is shown in Figure 4.3. Another way, and probably most difficult, is to compute it manually using the **analysis of variance** (ANOVA) table. The ANOVA table can be produced after a regression by choosing Analysis>ANOVA from the **model** window's pull-down menu as shown in Figure 4.1. The result appears in Figure 4.2.

Figure 4.1: After estimating the regression, select Analysis>ANOVA from the model window's pull-down menu.

👰 gretl: model 4				<u>- 0 ×</u>			
<u>File Edit Tests Save Graphs</u>	<u>Analysis</u> LaTeX						
Model 4: OLS estimates Dependent variable: y VARIABLE CO const	Display actual, fitted, residual Eorecasts Confidence intervals for coefficients Confidence glipse Coefficient covariance matrix ANOVA Bootstrap	1-40 T STAT 1.922	P-VALUE 0.06218 *				
x	10.2096 2.05326	4.877	0.00002 1	* * *			
x 10.2096 2.0026 4.877 0.00002 *** Mean of dependent variable = 283.574 Standard deviation of dep. var. = 112.675 Sum of squared residuals = 304505 Standard error of residuals = 89.517 Unadjusted R-squared = 0.385002 Adjusted R-squared = 0.368018 Degrees of freedom = 38 Log-likelihood = -235.509 Akaike information criterion (AIC) = 475.018 Schwarz Bayesian criterion (BIC) = 476.239							
	Close						



🐙 greti: ANOVA				
🔏 🗳 🛱 🔍 🗙				
Analysis of Varia	nce:			
	Sum of squares	df	Mean square	
Regression	SSR 190627	1	190627	
Total	SSE 304505 SST 495132	38 39	13029.8	
$R^{2} = 190627 / -$	495132 = 0.385002			
r(1, 30) - 1900.	27 7 0013.29 - 23.7000	,		
		Close		
		CIOSE		

In Figure 4.2 the SSR, SSE, and SST are shown. **Gretl** also does the R^2 computation for you as shown at the bottom of the output. If you want to verify **gretl**'s computation, then

$$SST = SSR + SSE = 190627 + 304505 = 495132 \tag{4.4}$$

and

$$\frac{SSR}{SST} = 1 - \frac{SSE}{SST} = \frac{190627}{495132} = .385 \tag{4.5}$$

Figure 4.3	3: In ε	addition	to some	other	summary	statistics,	Gretl	$\operatorname{computes}$	the	unadjus	ted I	\mathbb{R}^2	from
the linear	$\operatorname{regr}\epsilon$	ession.											

📓 gretl: model 1					
<u>File E</u> dit <u>T</u> ests <u>S</u> ave <u>G</u> r	aphs <u>A</u> nalysis <u>L</u>	aTeX			
Model 1: OLS, using Dependent variable:	observation Y	s 1–40			
coeffi	cient std.	error	t-ratio	p-value	
const 83.4	160 43.	4102	1.922	0.0622	*
x 10.2	096 2.	09326	4.877	1.95e-05	* * *
Mean dependent var Sum squared resid R-squared F(1, 38) Log-likelihood Schwarz criterion	283.5735 304505.2 0.385002 23.78884 -235.5088 478.3954	S.D. de S.E. of Adjuste P-value Akaike Hannan-	pendent va regressid d R-squara (F) criterion Quinn	ar 112.6 on 89.51 ed 0.368 0.0000 475.0 476.23	752 700 318 019 176 389

Finally, you can think of R^2 is as the squared correlation between your observations on your dependent variable, y, and the predicted values of y based on your estimated model, \hat{y} . A greth script to compute this version of the statistic is found below in section 4.6.5.

To use the GUI you can follow the steps listed here. Estimate the model using least squares and add the predicted values from the estimated model, \hat{y} , to your data set. Then use the **gretl** correlation matrix to obtain the correlation between y and \hat{y} . Adding the fitted values to the data set from the pull-down menu in the model window is illustrated in Figure 4.4 below. Highlight the variables y, x, and yhat2 by holding the control key down and clicking on each variable in the main **gretl** window as seen in Figure 4.5 below. Then, **View>Correlation Matrix** will produce all the pairwise correlations between each variable you've chosen. These are arranged in a matrix as shown in Figure 4.6 Notice that the correlation between y and x is the same as that between yand \hat{y} (i.e., 0.6205). As shown in your text, this is no coincidence in the simple linear regression model. Also, squaring this number equals R^2 from your regression, $0.6205^2 = .385$.

In Figure 4.4 of POE the authors plot y against \hat{y} . A positive linear relationship between the two is expected since the correlation their correlation is 0.62. To produce this plot, estimate the regression to open the model window. Add the predicted values of from the regression to the dataset using **Save>Fitted values** from the model window's pull-down menu. Name the fitted value, yhat1 and click OK. Now, return to the main window, use the mouse to highlight the two

Figure 4.4: Using the pull-down menu in the Model window to add fitted values to your data set.

<u>File Edit T</u> ests	<u>Save</u> Graphs Analysis LaTeX									
	Eitted values									
Model 2: OLS	Residuals	hservations 1-40								
Dependent va	Squared residuals									
	Error sum of squares									
VARIABI	Standard error of residuals	STDERROR	T STAT	P-VALUE						
	– R-squared									
const	T*R-squared	43.4102	1.922	0.06218 *						
x	Akaike Information Criterion	2.09326	4.877	0.00002 ***						
	– Bayesian Information Criterion									
Mean of dep	– · Hannan-Quinn Information Criterion	4								
Standard de		12.675								
Sum of squa	Define new variable									
Standard en	cror of residuals = 89.51	7								
Unadjusted	R-squared = 0.385002									
Adjusted R-	-squared = 0.368818									
Degrees of	freedom = 38									
Log-likeli	nood = -235.509									
Akaike info	ormation criterion (AIC)	= 475.018								
Schwarz Bay	yesian criterion (BIC) =	478.395								
Hannan-Quii	nn criterion (HQC) = 476.	239								
I										
		Close								

Figure 4.5: Hold the control key and click on y, x, and $\hat{y} = yhat2$ from the food expenditure regression to select them.

1	gret	i i										_ 🗆 ×
E	ile	<u>T</u> ools	Data	<u>V</u> iew	Add	∑ample	⊻ariable	Model				Help
fo	od.g	dt *										
IC	#	Variable name Descriptive label										
	0	const		auto-	genera	ted const	ant					
				Food	Expen	diture						
				Week	ly Inco	me (\$100)						
	3	yhat2		fitted	value	from mode	el 2					
								Undakov	d. Eullyspan 1 - 4	10		
	iii	₽[-		7		β 🖻	Unualed	1: Fuilfange I - 4	10		

Figure 4.6: The correlation matrix for y, x, and $\hat{y} = yhat2$ is produced by selecting View>Correlation matrix from the pull-down menu.

🦉 gretl:	correlatio	n mat	rix						
	-	TEX	×						
	C	orre 5%	lati cri	ion Coeff itical va	icients, alue (two	using -tailed	the) =	: observations 1 - 4 : 0.3120 for n = 40	:0
		У		,	c	yhat2			
	1.000	00		0.6205	5	0.6205	У		
				1.0000)	1.0000	х		
						1.0000	yh	lat2	
						Close			

variables (y and yhat1),³ then select View>Graph specified vars>X-Y scatter from the pulldown menu. This opens the **define graph** dialog box. Choose yhat1 as the Y-axis variable and y as the X-axis variable and click **OK**. A graph appears that looks similar to the one in *POE*. This one actually has a fitted least squares line through the data scatter that, as expected, has a positive slope. In fact, the slope is estimated to be .385, which is the regression's R^2 !

A simpler approach is to open a console window and use the following commands:

ols y const x genr yhat1 = \$yhat gnuplot yhat1 y

The first line estimates the regression. The predicted values are saved by **gretl** in **\$yhat**. Use the **genr** command to create a new variable, **yhat1**, that uses these. Then, call **gnuplot** with the predicted values, **yhat1**, as the first variable and the actual values of food expenditure, **y**, as the second. The graph is shown below in Figure 4.7. Finally, if you execute these commands using a script, the graph is written to a file on your computer rather than opened in a window. For this reason, I recommend executing these commands from the console rather than from the script file that appears at the end of this chapter.

4.3 Reporting Results

In case you think **gretl** is just a toy, the program includes a very capable utility that enables it to produce professional looking output. LaTeX, usually pronounced "Lay-tek", is a typesetting program used by mathematicians and scientists to produce professional looking technical documents. It is widely used by econometricians to prepare manuscripts for wider distribution. In fact, this book is produced using LaTeX.

³Remember, press and hold Ctrl, then click on each variable



Figure 4.7: A plot of predicted vs. actual food expenditures produced using **gnuplot** .

Although LaTeX is free and produces very professional looking documents, it is not widely used by undergraduate and masters students because 1) most degree programs don't require you to write a lot of technical papers and 2) it's a computer language and therefore it takes some time to learn its intricacies and to appreciate its nuances. Heck, I've been using it for years and still scratch my head when I try to put tables and Figures in the places I'd like them to be!

In any event, **gretl** includes a facility for producing output that can be pasted directly into LaTeX documents. For users of LaTeX, this makes generating regression output in proper format a breeze. If you don't already use LaTeX, then this will not concern you. On the other hand, if you already use it, **gretl** can be very handy in this respect.

In Figure 4.3 you will notice that on the far right hand side of the menu bar <u>Eile Edit Iests Save Graphs Analysi</u> is a pull-down menu for LaTeX. From here, you click LaTeX on the menu bar and a number of options are revealed as shown in Figure 4.8. You can view, copy, or save the regression output in

🛐 gre	tl: moo	lel 4						
Eile	Edit	Tests	Save	Graphs	Analysis	LaTeX		
						⊻iew	•	
Mod	el 4:	OLS	esti:	mates	using t	⊆ору	+	ns 1-40
Dep	enden	t var	riabl	e: y		Save	•	
L						Equation option:	s 🕨	 Show standard errors
	VA	RIABI	LE	со	EFFICIE	Tabular options.		Show <u>t</u> -ratios

Figure 4.8: Several options for defining the output of LaTeX are available.
Variable	Coefficient	td. Error	t-s	tatistic	p-value
const	83.4160	43.4102		1.9216	0.0622
Х	10.2096	2.09326		4.8774	0.0000
	Mean of dependent	variable	283.574		
	S.D. of dependent v	ariable	112.675		
	Sum of squared resi	duals	304505.		
	Standard error of re	siduals $(\hat{\sigma})$	89.5170)	
	Unadjusted R^2		0.3850	002	
	Adjusted \bar{R}^2		0.3688	818	
	Degrees of freedom		38		
	Akaike information	criterion	475.018		
	Schwarz Bayesian c	riterion	478.395		

Table 4.1: This is an example of LaTeX output in tabular form.

Model 1: OLS estimates using the 40 observations 1–40 Dependent variable: y

Table 4.2: Example of LaTeX output in equation form

$$\begin{split} \widehat{\mathbf{y}} &= 83.4160 + 10.2096\,\mathbf{x} \\ {}_{(1.922)}^{(1.922)} &+ 10.2096\,\mathbf{x} \\ {}_{(4.877)}^{(2.877)} &= 10.3688 \quad F(1,38) = 23.789 \quad \widehat{\sigma} = 89.517 \\ {}_{(t\text{-statistics in parentheses})}^{(1.922)} &= 10.3688 \quad F(1,38) = 23.789 \quad \widehat{\sigma} = 89.517 \\ {}_{(t\text{-statistics in parentheses})}^{(2.922)} &= 10.3688 \quad F(1,38) = 23.789 \quad \widehat{\sigma} = 89.517 \\ {}_{(t\text{-statistics in parentheses})}^{(2.922)} &= 10.3688 \quad F(1,38) = 23.789 \quad \widehat{\sigma} = 89.517 \\ {}_{(t\text{-statistics in parentheses})}^{(2.922)} &= 10.3688 \quad F(1,38) = 23.789 \quad \widehat{\sigma} = 89.517 \\ {}_{(t\text{-statistics in parentheses})}^{(2.922)} &= 10.3688 \quad F(1,38) = 23.789 \quad \widehat{\sigma} = 89.517 \\ {}_{(t\text{-statistics in parentheses})}^{(2.922)} &= 10.3688 \quad F(1,38) = 10.3688 \quad \widehat{\sigma} = 10.3688 \\ {}_{(t\text{-statistics in parentheses})}^{(2.922)} &= 10.3688 \quad \widehat{\sigma} = 10.3688$$

either tabular form or in equation form. You can tell **gret** whether you want standard errors or t-ratios in parentheses below parameter estimates, and you can define the number of decimal places to be used of output. Nice indeed. Examples of tabular and equation forms of output are found below in Tables 4.1 and 4.2, respectively.

4.4 Functional Forms

Linear regression is considerably more flexible than its name implies. There is no reason to believe that the relationship between any two variables of interest is necessarily linear. In fact there are many relationships in economics that we know are not linear. The relationship between production inputs and output is governed in the short-run by the law of diminishing returns, suggesting that a convex curve is a more appropriate function to use. Fortunately, a simple transformation of the variables (x, y, or both) can yield a model that is linear in the parameters (but not necessarily in the variables).

Simple transformation of variables can yield regression functions that are quite flexible. The important point to remember is, the functional form that you choose should be consistent with how the data are actually being generated. If you choose an inappropriate form, then your estimated model may at best not be very useful and at worst be downright misleading.

In gretl you are given some very useful commands for transforming variables. From the main gretl window the Add pull-down menu gives you access to a number of transformations; selecting one of these here will automatically add the transformed variable to your data set as well as its description.

Figure 4.9 shows the available selections from this pull-down menu. In the upper part of the panel two options appear in black, the others are greyed out because they are only available is you



Figure 4.9: The pull-down menu for adding new variables to gretl

have defined the **dataset structure** to consist of time series observations. The available options can be used to add the natural logarithm or the squared values of any highlighted variable to your data set. If neither of these options suits you, then the next to last option **Define new variable** can be selected. This dialog uses the **genr** command and the large number of built in functions to transform variables in different ways. Just a few of the possibilities include square roots (sqrt), sine (sin), cosine (cos), absolute value (abs), exponential (exp), minimum (min), maximum (max), and so on. Later in the book, we'll discuss changing the dataset's structure to enable some of the other variable transformation options.

4.5Testing for Normality

Your book, Principles of Econometrics, discusses the Jarque-Bera test for normality which is computed using the skewness and kurtosis of the least squares residuals. To compute the Jarque-Bera statistic, you'll first need to estimate your model using least squares and then save the residuals to the data set.

From the **gretl** console

ols y const x genr uhat1 = \$uhat summary uhat1

The first line is the regression. The next saves the least squares redsiduals, identified as \$uhat, into a variable I have called uhat1.⁴ You could also use the point-and-click method to add the residuals to the data set. This is accomplished from the regression's output window. Simply choose Save>Residuals from the model pull-down menu to add the estimated residuals to the dataset. The last line of the script produces the summary statistics for the residuals and yields the output in Figure 4.10. One thing to note, gretl reports excess kurtosis rather than kurtosis. The excess

Figure 4.10: The summary statistics for the least squares residuals.

? summary uhat1

Su	mmary Statistics, using the observations 1 - 40
f	or the variable 'uhat1' (40 valid observations)
Mean	0.00000
Median	-6.3245
Minimum	-223.03
Maximum	212.04
Standard devia	tion 88.362
C.V.	2.4147E+015
Skewness	-0.097319
Ex. kurtosis	-0.010966

kurtosis is measured relative to that of the normal distribution which has kurtosis of three. Hence, your computation is

$$JB = \frac{T}{6} \left(\text{Skewness}^2 + \frac{(\text{Excess Kurtosis})^2}{4} \right)$$
(4.6)

⁴You can't use uhat instead of uhat1 because that name is reserved by gretl.

Which is

$$JB = \frac{40}{6} \left(-0.097^2 + \frac{-0.011^2}{4} \right) = .063 \tag{4.7}$$

Gretl also includes a built in test for normality proposed by Doornik and Hansen (1994). Computationally, it is much more complex than the Jarque-Bera test. The Doornik-Hansen test also has a χ^2 distribution if the null hypothesis of normality is true. It can be produced from the **gretl** console after running a regression using the command **testuhat**.⁵

4.6 Examples

4.6.1 Wheat Yield Example

The results from the example in section 4.3 of your textbook is easily produced in **gret1**. Start by loading the data and estimating the effect of time, *time* on yield *green* using least squares. The following script will load the data file, estimate the model using least square, and generate a graph of the actual and fitted values of yield (green) from the model.

```
open "c:\Program Files\gretl\data\poe\wa-wheat.gdt"
ols green const time
gnuplot green time
```

The resulting plot appears below in Figure 4.11. The simple **gnuplot** command works well enough. However, you can take advantage of having declared the dataset structure to be time series to improve the look. In this example we'll reproduce Figure 4.8 of *POE* using two options for **gnuplot**. Figure 4.8 of *POE* plots the residuals, the actual yield, and predicted yield from the regression against time. Estimate the model using least squares and save the predicted values (\$yhat) and residuals (\$uhat) to new variables using the **genr** command. We'll call these yhat1 and uhat1, respectively. Then use

```
gnuplot green yhat1 uhat1 --with-lines --time-series
```

There are two options listed after the plot. The first (--with-lines) tells gnuplot to connect the points using lines. The second option (--time-series) tells gnuplot that the graph is of time series. In this case, the dataset's defined time variable will be used to locate each point's position on the X-axis. The graph in Figure 4.10 can be produced similarly. The complete script for Figure 4.8 of *POE* is:

⁵The R software also has a built-in function for performing the Jarque-Bera test. To use it, you have to download and install the *tseries* package from CRAN. Once this is done, estimate your model using least squares as discussed in appendix D and execute jarque.bera.test(fitols\$residual).



Figure 4.11: The plot of the actual yield and predicted yield from your estimated model

open "c:\Program Files\gretl\data\poe\wa-wheat.gdt"
ols green const time
genr yhat1 = \$yhat
genr uhat1 = \$uhat
gnuplot green yhat1 uhat1 --with-lines --time-series

The comparable graph in **gretl** is found in Figure 4.12. Actually, this graph has had a bit of editing done via **gretl**'s graph editing dialog shown in Figure 4.13. From Figure 4.13 we have selected the **lines** tab and changed a few of the defaults. The **legend** for each series is changed from the variable's name to something more descriptive (e.g., *uhat1* is changed to *Residual*). The line styles were also changed. Steps were used for the residuals to mimic the output in Figure 4.9 of *POE* that shows a bar graph of the least squares residuals. From the stepped line, it becomes more obvious that yield is probably not linear in time. The X-axis and Main tabs were also used to change the name of the X-axis from time to Year and to add a title for the graph.

To explore the behavior of yield further, create a new variable using the genr command from $t3 = time^3/1,000,000$ as shown below. The new plot appears in Figure 4.14.

```
genr t3=time^3/1000000
ols green const t3
genr yhat2 = $yhat
genr uhat2 = $uhat
gnuplot green yhat2 uhat2 --with-lines --time-series
```





4.6.2 Growth Model Example

Below you will find a script that reproduces the results from the growth model example in section 4.4.1 of your textbook.

```
open "c:\Program Files\gretl\data\poe\wa-wheat.gdt"
genr lyield = log(green)
ols lyield const time
```

4.6.3 Wage Equation

Below you will find a script that reproduces the results from the wage equation example in section 4.4.2 of your textbook.

```
open "c:\Program Files\gretl\data\poe\cps1.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr lb = $coeff(educ) - 1.96 * $stderr(educ)
genr ub = $coeff(educ) + 1.96 * $stderr(educ)
print lb ub
```

Figure 4.13: The graph dialog box can be used to change characteristics of your graphs. Use the Main tab to give the graph a new name and colors; use the X- and Y-axes tabs to refine the behavior of the axes and to provide better descriptions of the variables graphed.

1 20	
grett plot controls	`
Main X-axis Y-axis Lines	Labels Output to file
line 1; legen	d Actual
typ	e linespoints 🔹
scal	e 1.0
y ax	s left
line widt	h 1
line 2: legen	d Predicted
typ	e lines 💌
scal	e 1.0
y ax	s left
line widt	h 1 🥏
line 3: legen	d Residual
typ	e steps
scal	e 1.0
y ax	s left
line widt	h 1
🔯 Help 🗸	Apply



Figure 4.14: The plot of the actual yield and predicted yield from the model estimated with the cubic term

4.6.4 Predictions in the Log-linear Model

In this example, you use your regression to make predictions about the log wage and the level of the wage for a person having 12 years of schooling.

```
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr lyhat_12 = $coeff(const) + $coeff(educ)*12
genr yhat_12 = exp(lyhat_12)
genr corr_yhat_12 = yhat_12*exp($ess/(2*$df))
```

4.6.5 Generalized R^2

A generalized version of the goodness-of-fit statistic R^2 can be obtained by taking the squared correlation between the actual values of the dependent variable and those predicted by the regression. The following script reproduces the results from section 4.4.4 of your textbook.

```
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr l_yhat = $yhat
genr y = exp(l_yhat)
genr corr1 = corr(y, wage)
```

genr Rsquare = corr1²

4.6.6 Prediction Interval

In this script the 95% prediction interval for someone having 12 years of education is estimated.

```
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr lyhat_12 = $coeff(const) + $coeff(educ)*12
genr sig2 = $ess/$df
genr f = sig2 + sig2/$nobs + ((12-mean(educ))^2)*($stderr(educ)^2)
genr sef = sqrt(f)
genr lb = exp(lyhat_12-1.96*sef)
genr ub = exp(lyhat_12+1.96*sef)
print lyhat_12 sig2 f sef lb ub
```

4.7 Script

```
open "c:\Program Files\gretl\data\poe\food.gdt"
```

```
ols y const x
genr yhat0 = $coeff(const) + $coeff(x)*20
genr f=8013.2941+(8013.2941/40)+4.3818*(20-19.6047)**2
genr ub=yhat0+2.0244*sqrt(f)
genr lb=yhat0-2.0244*sqrt(f)
#Prediction Intervals
ols y const x
genr yhat0=$coeff(const)+20*$coeff(x)
genr sig2 = $ess/$df
genr f = sig2 + sig2/$nobs + ((20-mean(x))^2)*($stderr(x)^2)
genr lb = yhat0-critical(t,$df,0.025)*sqrt(f)
genr ub = yhat0+critical(t,$df,0.025)*sqrt(f)
```

```
#Plot predictions vs actual food exp
#note: the plot will be written to a file.
```

```
#To see the plot, open a console window and execute the commands
ols y const x
genr yhatime = $yhat
gnuplot yhat1 y
```

```
#Testing normality of errors
ols y const x
genr uhat1 = $uhat
summary uhat1
#Wheat yield example
open "c:\Program Files\gretl\data\poe\wa-wheat.gdt"
ols green const time
gnuplot green time
ols green const time
genr yhat1 = $yhat
genr uhat1 = $uhat
gnuplot green yhat1 uhat1 --with-lines --time-series
genr t3=time^3/1000000
ols green const t3
genr yhat2 = $yhat
genr uhat2 = $uhat
gnuplot green yhat2 uhat2 --with-lines --time-series
#Growth model example
open "c:\Program Files\gretl\data\poe\wa-wheat.gdt"
genr lyield = log(green)
ols lyield const time
#Wage Equation
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr lb = $coeff(educ) - 1.96 * $stderr(educ)
genr ub = $coeff(educ) + 1.96 * $stderr(educ)
print lb ub
#Predictions in the Log-linear model
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr lyhat_12 = $coeff(const) + $coeff(educ)*12
genr yhat_12 = exp(lyhat_12)
genr corr_yhat_12 = yhat_12*exp($ess/(2*$df))
#Generalized R-Square
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
```

```
genr l_wage = log(wage)
ols l_wage const educ
genr l_yhat = $yhat
genr y = exp(l_yhat)
genr corr1 = corr(y, wage)
genr Rsquare = corr1^2
#Prediction interval
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ
genr lyhat_12 = $coeff(const) + $coeff(educ)*12
genr sig2 = $ess/$df
genr f = sig2 + sig2/$nobs + ((12-mean(educ))^2)*($stderr(educ)^2)
genr sef = sqrt(f)
genr lb = exp(lyhat_12-1.96*sef)
genr ub = exp(lyhat_12+1.96*sef)
print lyhat_12 sig2 f sef lb ub
```

Chapter

Multiple Regression Model

The multiple regression model is an extension of the simple model discussed in Chapter 2. The main difference is that the multiple linear regression model contains more than one explanatory variable. This changes the interpretation of the coefficients slightly and requires another assumption. The general form of the model is shown in equation (5.1) below.

$$y_i = \beta_1 + \beta_2 x_{i2} + \ldots + \beta_K x_{iK} + e_i \quad i = 1, 2, \ldots, N$$
(5.1)

where y_i is your dependent variable, x_{ik} is the i^{th} observation on the k^{th} independent variable, $k = 2, 3, \ldots, K$, e_i is random error, and $\beta_1, \beta_2, \ldots, \beta_K$ are the parameters you want to estimate. Just as in the simple linear regression model, each error, e_i , has an average value of zero for each value of the independent variables; each has the same variance, σ^2 , and are uncorrelated with any of the other errors. In order to be able to estimate each of the βs , none of the independent variables can be an exact linear combination of the others. This serves the same purpose as the assumption that each independent variable of the simple linear regression take on at least two different values in your dataset. The error assumptions can be summarized as $e_i|x_{i2}, x_{i3}, \ldots, x_{iK}$ iid $(0, \sigma^2)$. Recall from Chapter 2 that expression iid means that the errors are statistically independent from one another (and therefor uncorrelated) and each has the same probability distribution. Taking a random sample from a single population accomplishes this.

The parameters $\beta_2, \beta_3, \ldots, \beta_K$ are referred to as *slopes* and each slope measures the effect of a 1 unit change in x_{ik} on the average value of y_i , holding all other variables in the equation constant. The conditional interpretation of the coefficient is important to remember when using multiple linear regression.

The example used in this chapter models the sales for Big Andy's Burger Barn. The model includes two explanatory variables and a constant.

$$S_i = \beta_1 + \beta_2 P_i + \beta_3 A_i + e_i \quad i = 1, 2, \dots, N$$
(5.2)

where S_i is monthly sales in a given city and is measured in \$1,000 increments, P_i is price of a

hamburger measured in dollars, and A_i is the advertising expenditure also measured in thousands of dollars.

5.1 Linear Regression

The parameters of the model are estimated using least squares which can be done using the pull-down menus and dialog boxes (GUI) or by using **gretl** language itself. Both of these will be demonstrated below. The GUI makes it easy to estimate this model using least squares. There are actually two ways to open the dialog box. The first is to use the pull-down menu. Select Model>Ordinary Least Squares from the main gretl window as shown below in Figure 5.1. This

Figure 5.1: Using the pull-down menu to open the ordinary least squares dialog box.

📓 gr	etl										×
Eile	Tools	Data	⊻iew	<u>A</u> dd	<u>S</u> ample	<u>V</u> ariable	Model	Help			
andy.	.gdt						Or	dinary Least Squares			
ID#	◀ Varia	ble nam	ne 🖣 🛛 D	escripti	ive label		Ins	strumental variables	+	•	
0	cons	t	a	uto-ge	nerated o	onstant	Ot	her linear models	+		
1	S		P	1onthly	Sales rev	enue (\$100	No	nlinear models	+		-
2	Р		\$	Price				ne series	×.		
3	A		P	1onthy	Advertisin	ig Expendit	<u>P</u> a	nel	×.		
							<u>R</u> o	bust estimation	•		
							Ma	ximum likelihood			
							GM	1M			
							Sin	nultaneous equations			
											_
							Undated	d: Full range 1 - 75			
			fx	1 13	0	3 🖻					
	- L										

brings up the dialog box shown in Figure 5.2. As in Chapter 2 you need to put the dependent variable (S) and the independent variables (const, P, and A) in the appropriate boxes. Click OK and the model is estimated.

There is a shortcut to get to the specify model dialog box. On the toolbar located at the bottom of the main **gretl** window is a button labeled $\hat{\beta}$. Clicking on this button as shown in Figure 5.3 will open the OLS specify model dialog box in Figure 5.2.

1 7	0	v				
gretl: specify model						
	OLS					
const	Depende	ent variable				
S	Choose ->	i				
A	🔲 Set as default					
	Independe	ent variables				
	Add ->	st				
	<- Remove					
Robust standard errors configure						
🞯 Help 🛛 📎	ے (Clear کی Cance)	н 🖉 ок				

Figure 5.2: The specify model dialog box for ordinary least squares (OLS)

Figure 5.3: The OLS shortcut

gret	1					
Eile	<u>T</u> ools <u>D</u> ata	<u>V</u> iew <u>A</u> dd Sample <u>V</u> ariable <u>M</u> odel	Help			
andy.g	gdt					
ID #	Variable name	Descriptive label				
0	const	auto-generated constant				
1	S	Sales				
2	Р	Price				
3	А	Advertising				
4	ЬL	\$coeff(P) - critical(t,\$df,0.025) * \$stderr(P)				
5	Ьυ	\$coeff(P) + critical(t,\$df,0.025) * \$stderr(P)				
6	t1	(\$coeff(P)-0)/\$stderr(P) 人				
7	t2	(\$coeff(A)-1)/\$stderr(A)				
8	с	critical(t,\$df,.025)				
	Click here to open the OLS dialog box					
		Undated: Full range 1 - 75				

Variable	Coefficient	Std. Error	t-statistic	p-value
const	118.914	6.35164	18.7217	0.0000
Р	-7.9078	1.09599	-7.2152	0.0000
А	1.86258	0.683195	2.7263	0.0080
	Mean of depe	ndent variable	77.3747	
	S.D. of depen	dent variable	6.48854	
	Sum of square	ed residuals	1718.94	
	Standard erro	or of residuals $(\hat{\sigma})$	4.88612	
	Unadjusted R	22	0.448258	
	Adjusted \bar{R}^2		0.432932	
	F(2, 72)		29.2479	
	Log-likelihood	1	-223.87	
	Akaike inform	nation criterion	453.739	
	Schwarz Baye	sian criterion	460.691	
	Hannan–Quir	in criterion	456.515	

Table 5.1: The regression results from Big Andy's Burger Barn

Model 1: OLS estimates using the 75 observations 1–75 Dependent variable: S

5.2 Big Andy's Burger Barn

To estimate the model for Big Andy's, we'll use a script file. The following two lines are typed into a script file which is executed by clicking your mouse on the "gear" button of the script window.

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
ols S const P A
```

This assumes that the gretl data set andy.gdt is installed at c:\userdata\gretl\data\poe\. The results, in tabular form, are in Table 5.1 and match those presented in the textbook.

In addition to providing information about how sales change when price or advertising change, the estimated equation can be used for prediction. To predict sales revenue for a price of \$5.50 and an advertising expenditure of \$1,200 we can use the **genr** to do the computations. From the console,

```
? genr S_hat = coeff(const) + coeff(P)*5.5 + coeff(A)*1.2
Generated scalar S_hat (ID 4) = 77.6555
```

which matches the result in your text.

5.2.1 SSE, R^2 and Other Statistics

Other important output is included in Table 5.1. For instance, you'll find the sum of squared errors (SSE) which **gretl** refers to as "sum of squared residuals." In this model SSE = 1718.94. To obtain the estimated variance, $\hat{\sigma}^2$, divide SSE by the available degrees of freedom to obtain

$$\hat{\sigma}^2 = \frac{SSE}{N-K} = \frac{1718.94}{75-3} = 23.874 \tag{5.3}$$

The square root of this number is referred to by **gretl** as the "Standard error of residuals, $\hat{\sigma}$ " and is reported to be 4.88612. **Gretl** also reports R^2 in this table. If you want to compute your own versions of these statistics using the total sum of squares from the model, you'll have to use **Analysis>ANOVA** from the model's pull-down menu to generate the ANOVA table. Refer to section 4.2 for details.

To compute your own from the standard **gretl** output recall that

$$\hat{\sigma}_y = \sqrt{\frac{SST}{N-1}} \tag{5.4}$$

The statistic $\hat{\sigma}_y$ is printed by **gretl** and referred to as "S.D. of dependent variable" which is reported to be 6.48854. A little algebra reveals

$$SST = (N-1)\hat{\sigma}_y^2 = 74 * 6.48854 = 3115.485$$
(5.5)

Then,

$$R^2 = 1 - \frac{SSE}{SST} = 1 - \frac{1718.94}{3115.485} = 0.448$$
(5.6)

Otherwise, the goodness-of-fit statistics printed in the **gretl** regression output or the ANOVA table are perfectly acceptable.

Gretl also reports the **adjusted** R^2 in the standard regression output. The adjusted R^2 imposes a small penalty to the usual R^2 when a variable is added to the model. Adding a variable with any correlation to y always reduces SSE and increases the size of the usual R^2 . With the adjusted version, the improvement in fit may be outweighed by the penalty and it could become smaller as variables are added. The formula is:

$$\bar{R}^2 = 1 - \frac{SSE/(N-K)}{SST/(N-1)}$$
(5.7)

This sometimes referred to as "R-bar squared," (i.e., \overline{R}^2) although in **gretl** it is called "adjusted R-squared." For Big Andy's Burger Barn the adjusted R-squared is equal to 0.4329.

5.2.2 Covariance Matrix and Confidence Intervals

Gretl can be used to print the variance-covariance matrix by using the pull-down menu as shown in Figure 2.13. Or, the --vcv option can be used with the ols command to obtain this result from the console or using a script. The example code is:

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
ols S const P A --vcv
```

Confidence intervals are obtained using the **genr** command in the same way as in Chapter 3. The **gretl** commands

genr bL = \$coeff(P) - critical(t,\$df,0.025) * \$stderr(P)
genr bU = \$coeff(P) + critical(t,\$df,0.025) * \$stderr(P)

Remember, you can also summon the 95% confidence intervals from the model window using the pull-down menu by choosing Analysis>Confidence intervals for coefficients.

5.2.3 t-Tests, Critical Values, and P-values

In Section 3.3 we used the GUI to obtain test statustics, critical values and p-values. However, it is much easier to use the the **genr** command from either the console or as a script to compute these. For t-ratios and one- and two-sided hypothesis tests the appropriate commands are:

genr t1 = (\$coeff(P)-0)/\$stderr(P)
genr t2 = (\$coeff(A)-1)/\$stderr(A)

The critical values for the t_{72} and the p-values for the two statistics can be easily obtained using the command

genr c=critical(t,\$df,0.025)
pvalue t \$df t1
pvalue t \$df t2

These last three commands produce the output shown below:

```
? genr c=critical(t,$df,.025)
Generated scalar c (ID 8) = 1.99346
? pvalue t $df t1
t(72): area to the right of -7.21524 =~ 1
(to the left: 2.212e-010)
(two-tailed value = 4.424e-010; complement = 1)
? pvalue t $df t2
t(72): area to the right of 1.26257 = 0.105408
(two-tailed value = 0.210817; complement = 0.789183)
```

It is interesting to note that when a negative t-ratio is used in the **pvalue** function, **gretl** returns both the area to its right, the area to its left and the sum of the two areas. So, for the alternative hypothesis that the coefficient on P is less than zero (against the null that it is zero), the p-value is the area to the left of the computed statistic is the desired one.

5.3 Script

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
#Change the descriptive labels and graph labels
setinfo S -d "Monthly Sales revenue ($1000)" -n "Monthly Sales ($1000)"
setinfo P -d "$ Price" -n "Price"
setinfo A -d "Monthy Advertising Expenditure ($1000)" -n \
    "Monthly Advertising ($1000)
#Print the new labels to the screen
labels
#Summary Statistics
summary S P A
#Regression with covariance matrix printed
ols S const P A --vcv
#Prediction
genr S_hat = $coeff(const) + $coeff(P)*5.5 + $coeff(A)*1.2
#Confidence Intervals
  #Price
  genr bL = $coeff(P) - critical(t,$df,0.025) * $stderr(P)
  genr bU = $coeff(P) + critical(t,$df,0.025) * $stderr(P)
  #Advertising
  genr bL = $coeff(A) - critical(t,$df,0.025) * $stderr(A)
  genr bU = $coeff(A) + critical(t,$df,0.025) * $stderr(A)
#t-ratios
  #Two tail tests
  genr t1 = ($coeff(P)-0)/$stderr(P)
  genr t2 = (\text{scoeff}(A)-0)/\text{stderr}(A)
  #One tail test
  genr t3 = (\text{scoeff}(A)-1)/\text{stderr}(A)
#Ctitical value and p-values
genr c=critical(t,$df,.025)
pvalue t $df t1 #used for both 1 and 2 tail tests
pvalue t $df t2
pvalue t $df t3
```

Chapter 6

Further Inference in the Multiple Regression Model

In this chapter several extensions of the multiple linear regression model are considered. First, we test joint hypotheses about parameters in a model and then learn how to impose linear restrictions on the parameters. A condition called *collinearity* is also explored.

6.1 F-test

An F-statistic can be used to test multiple hypotheses in a linear regression model. In linear regression there are several different ways to derive and compute this statistic, but each yields the same result. The one used here compares the sum of squared errors (SSE) in a regression model estimated under the null hypothesis (H_0) to the SSE of a model under the alternative (H_1) . If the sum of squared errors from the two models are similar, then there is not enough evidence to reject the restrictions. On the other hand, if imposing restrictions implied by H_0 alter SSE substantially, then the restrictions it implies don't fit the data and we reject them.

In the Big Andy's Burger Barn example we estimated the model

$$S_i = \beta_1 + \beta_2 P_i + \beta_3 A_i + e_i \tag{6.1}$$

Suppose we wish to test the hypothesis that price, P_i , has no effect on sales against the alternative that it does. Thus, $H_0: \beta_2 = 0$ and $H_1: \beta_2 \neq 0$. Another way to express this is in terms of the models each hypothesis implies.

$$H_0: \qquad \beta_1 + \beta_3 A_i + e_i$$

$$H_1: \quad \beta_1 + \beta_2 P_i + \beta_3 A_i + e_i$$

The model under H_0 is **restricted** compared to the model under H_1 since in it $\beta_2 = 0$. The F-statistic used to test H_0 versus H_1 estimates each model by least squares and compares their respective sum of squared errors using the statistic:

$$F = \frac{(SSE_r - SSE_u)/J}{SSE_u/(N-K)} \sim F_{J,N-K} \quad \text{if } H_0 \text{ is true}$$
(6.2)

The sum of squared errors from the unrestricted model (H_1) is denoted SSE_u and that of the restricted model (H_0) is SSE_r . The numerator is divided by the number of hypotheses being tested, J. In this case that is 1 since there is only a single restriction implied by H_0 . The denominator is divided by the total number of degrees of freedom in the unrestricted regression, N - K. N is the sample size and K is the number of parameters in the unrestricted regression. When the errors of your model are (1) independently and identically distributed (*iid*) normals with zero mean and constant variance ($e_t \ iid \ N(0, \sigma^2)$) and (2) H_0 is true, then this statistic has an F distribution with J numerator and N - K denominator degrees of freedom. Choose a significance level and compute this statistic. Then compare its value to the appropriate critical value from the F table or compare its p-value to the chosen significance level.

The script to estimate the models under H_0 and H_1 and to compute the test statistic is given below.

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
ols S const P A
genr sseu = $ess
genr unrest_df = $df
ols S const A
genr sser = $ess
genr Fstat=((sser-sseu)/1)/(sseu/(unrest_df))
pvalue F 1 unrest_df Fstat
```

Gretl refers to the sum of squared residuals (SSE) as the "error sum of squares" and it is retrieved from the regression results using the syntax genr sseu = \$ess. In this case, \$ess points to the error sum of squares computed in the regression that precedes it. You'll also want to save the degrees of freedom in the unrestricted model so that you can use it in the computation of the p-value for the F-statistic. In this case, the F-statistic has 2 known parameters (J=1 and N - K=unrest_df) that are used as arguments in the pvalue function.

There are a number of other ways within **gretl** to do this test. These are available through scripts, but it may be useful to demonstrate how to access them through the GUI. First, you'll want to estimate the model using least squares. From the pull-down menu (see Figure 5.1) select Model>Ordinary Least Squares, specify the unrestricted model (Figure 5.2), and run the regression. This yields the result shown in Figure 6.1.

You'll notice that along the menu bar at the top of this window there are a number of options

	gretl: m	odel 4											_ 🗆 🗵
Ē	ile <u>E</u> dit	Tests	Save	Graphs	Analysis	LaTeX							
M	odel 4 epende	: OLS	esti: riabl	mates e: S	using t	the 75	observ	ations	1-75				
	V	ARIAB:	LE	cc	EFFICIE	ENT	ຮາ	DERROR		T STAT	P-VALUE		
	const	;		1	18.914		6	.35164		18.722	<0.00001	***	
	Р				-7.9078	35	1	.09599		-7.215	<0.00001	* * *	
	A				1.8625	58	C	.683195	5	2.726	0.00804	***	
Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48654 Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515													
							Clos	•					

Figure 6.1: The model results from least squares regression using the pull-down menu

that are available to you. Choose **Tests** and the pull-down menu shown in Figure 6.2 will be revealed. The first four options in 6.2 are highlighted and these are the ones that are most pertinent to the discussion here. This menu provides you an easy way to omit variables in the null, add variables to the alternative, test a sum of your coefficients, or to test arbitrary linear restrictions on the parameters of your model.

Since this test involves imposing a zero restriction on the coefficient of the variable P, we can use the omit option. This brings up the dialog box shown in Figure 6.3. Notice the two radio buttons at the bottom of the window. The first is labeled *Estimate reduced model* and this is the one you want to use to compute equation 6.2. If you select the other, no harm is done. It is computed in a different way, but produces the same answer in a linear model. The only advantage of the Wald test (second option) is that the restricted model does not have to be estimated in order to perform the test. Given **gretl**'s speed, there is not much to be gained here from using the Wald form of the test, other than it generates less output to view! Select the variable P and click OK to reveal the result shown in Figure 6.4. The interesting thing about this option is that it mimics your manual calculation of the F statistic from the script. It computes the sum of squared errors in the unrestricted and restricted models and computes equation (6.2) based on those regressions. Most pieces of software choose the alternative method (Wald) to compute the test, but you get the same result.

You can also use the linear restrictions option from the pull-down menu shown in Figure 6.2. This produces a large dialog box that requires a bit of explanation. The box appears in Figure 6.5. The restrictions you want to impose (or test) are entered here. Each restriction in the set should be expressed as an equation, with a linear combination of parameters on the left and a numeric value to the right of the equals sign. Parameters are referenced in the form b[variable number], where variable number represents the position of the regressor in the question, which starts with 1. This means that β_2 is equivalent to b[2]. Restricting $\beta_2 = 0$ is done by issuing b[2]=0 in

Figure 6.2: Choosing **Tests** from the pull-down menu of the model window reveals several testing options

🔣 gretl: m	odel 1	
<u>F</u> ile <u>E</u> dit	<u>T</u> ests <u>S</u> ave <u>G</u> raphs <u>A</u> nalysis	LaTeX
Model 1 Depende	<u>O</u> mit variables <u>A</u> dd variables Sum of coefficients	ns 1-75
 const P A	Linear restrictions Non-linearity (squares) Non-linearity (logs) <u>R</u> amsey's RESET	. error t-ratio p-value
Mean de <u>Mormality of residual</u> Sum squ <u>Influential observations</u> F (2, 72 <u>Collinearity</u> Loge 1 k		S.D. dependent var 6.488537 S.E. of regression 4.886124 Adjusted R-squared 0.432932 P-value(F) 5.04e-10 Akaike criterion 453.7390
Schwarz	Autocorrelation Durbin-Watson p-value ABCH QLR test CUSUM test CUSUM5Q test Cosum5Q test	Hannan-Quinn 456.5151
	Panel diagnostics]

Figure 6.3: The Omit variables dialog box available from the Tests pull-down menu in the model window.

🔣 gretl: model tests	
Select variables to omit	
Available vars	Selected vars
Const	
A Add ->	
<- Remove	
Estimate reduced model	
 Wald test, based on covariance matrix 	
 Sequential elimination of variables using two-sided p-value: 	0.10
Help Clear Cancel	

Figure 6.4: The results using the Omit variables dialog box to test zero restrictions on the parameters of a linear model.

👰 gretl: model 2						<u> </u>
Eile Edit Tests Save Gr	aphs <u>A</u> nalysis LaTeX					
Model 2: OLS estimat Dependent variable:	tes using the 75 o S	observations 1-7	'5			<u> </u>
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE		
const A	74.1797 1.73262	1.79898 0.890324	41.234 1.946	<0.00001 0.05550	* * * *	
Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 2961.83 Standard error of residuals = 6.36969 Unadjusted R-squared = 0.0493197 Adjusted R-squared = 0.0362967 Degrees of freedom = 73 Log-likelihood = -244.273 Akaike information criterion (AIC) = 492.546 Schwarz Bayesian criterion (BIC) = 497.181 Hannan-Quinn criterion (HQC) = 494.397						
Null hypothesis: the regression parameters are zero for the variables						
P Test statistic: F Of the 3 model se:	(1, 72) = 52.0597, lection statistics	, with p-value = s, O have improv	4.424e-0 red.	10		_
		Close				

Figure 6.5: The linear restriction dialog box obtained using the Linear restrictions option in the Tests pull-down menu.

📓 gretl: linear restrictions	×
Specify restrictions: (Please refer to Help for guidance)	
b[2]=0 ← List the restrictions here. Put each restriction on a separate line	
Use bootstrap	
Help QK	

this dialog. Sometimes you'll want to use a restriction that involves a multiple of a parameter e.g., $3\beta_3 = 2$. The basic principle is to place the multiplier first, then the parameter, using * to multiply. So, in this case the restriction in gret1 becomes 3*b[3] = 2.

When you use the console or a script instead of the pull-down menu to impose restrictions, you'll have to tell **gretl** where the restrictions start and end. The restriction(s) starts with a **restrict** statement and ends with **end restrict**. The statement will look like this:

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
ols S const P A
restrict
    b[2] = 0
end restrict
```

When you have more than one restriction to impose or test, put each restriction on its own line. Here is an example of a set of restrictions from a **gretl** script:

```
restrict
    b[1] = 0
    b[2] - b[3] = 0
    b[4] + 2*b[5] = 1
end restrict
```

Of course, if you use the pull-down menu to impose these you can omit the restrict and end restrict statements. The results you get from using the restrict statements appear in Figure 6.6. The test statistic and its p-value are highlighted in red.

Figure 6.6: The results obtained from using the restrict dialog box.

🎆 gretl: linear	restrictions						
1	. *						
Restriction: b[P] = 0							
Test statistic: $F(1, 72) = 52.0597$, with p-value = 4.424e-010							
Restricted	estimates:						
	coefficient	std. error	t-ratio	p-value		≡	
const	74.1797	1.79898	41.23	2.56e-052	***		
P	0.000000	0.000000	NA	NA			
A	1.73262	0.890324	1.946	0.0555	*		
Standard	error of the re	gression = 6.	.36969			~	

6.2 Regression Significance

To statistically determine whether the regression is actually a model of the average behavior of your dependent variable, you can use the F-statistic. In this case, H_0 is the proposition that y does not depend on your independent variables, and H_1 is that it does.

$$H_o: \quad \beta_1 + e_i H_1: \quad \beta_1 + \beta_2 x_{i2} + \ldots + \beta_k x_{ik} + e_i$$

The null hypothesis can alternately be expressed as $\beta_2, \beta_3, \ldots, \beta_K = 0$, a set of K - 1 linear restrictions. In Big Andy's Burger Barn the script is

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
ols S const P A
genr sseu = $ess
genr unrest_df = $df
ols S const
genr sser = $ess
genr rest_df = $df
genr J = rest_df - unrest_df
genr Fstat=((sser-sseu)/J)/(sseu/(unrest_df))
pvalue F J unrest_df Fstat
```

The only difference is that you now have two hypotheses to test jointly and the numerator degrees of freedom for the F-statistic is J = K - 1 = 2. The saved residual degrees of freedom from the restricted model can be used to obtain the number of restrictions imposed. Each unique restriction in a linear model reduces the number of parameters in the model by one. So, imposing one restriction on a three parameter unrestricted model (e.g., Big Andy's), reduces the number of parameters in the restricted model to two. Let K_r be the number of regressors in the restricted model and K_u the number in the unrestricted model. Subtracting the degrees of freedom in the unrestricted model $(N - K_u)$ from those of the restricted model $(N - K_r)$ will yield the number of restrictions you've imposed, i.e., $(N - K_r) - (N - K_u) = (K_u - K_r) = J$.

The test of regression significance is important enough that it appears on the default output of every linear regression estimated using **gretl**. The statistic and its p-value are highlighted in Figure 6.7. The F-statistic for this test and its p-value are highlighted. Since the p-value is less than = .05, we reject the null hypothesis that the model is insignificant at the five percent level.

Figure 6.7: The overall F-statistic of regression significance is produced by default when you estimate a linear model using least squares.

File Edit Tests Save Graphs Analysis LaTeX Model 2: OLS estimates using the 75 observations 1-75 Dependent variable: S VARIABLE COEFFICIENT STDERROR T STAT P-VALUE const 118.914 6.35164 18.722 <0.00001 **** P -7.90785 1.09599 -7.215 <0.00001 **** A 1.86258 0.683195 2.726 0.00804 **** Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 Overall F-test of regression F-statistic (2, 72) = 29.2479 (p-value < 0.00001) significance Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515 - -	💽 gre	etl: mo	del 2																<u> </u>
<pre>Model 2: OLS estimates using the 75 observations 1-75 Dependent variable: S VARIABLE COEFFICIENT STDERROR T STAT P-VALUE const 118.914 6.35164 18.722 <0.00001 **** P -7.90785 1.09599 -7.215 <0.00001 **** A 1.86258 0.683195 2.726 0.00804 **** Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 Overall F-test of regression F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515</pre>	Eile	⊑dit	Tests	Save	Grap	hs <u>A</u> n	alysis	LaT	еX										
Model 2: OLS estimates using the 75 observations 1-75 Dependent variable: S VARIABLE COEFFICIENT STDERROR T STAT P-VALUE const 118.914 6.35164 18.722 <0.00001 ****								_	_				_	_	_				
VARIABLE COEFFICIENT STDERROR T STAT P-VALUE const 118.914 6.35164 18.722 <0.00001	Model 2: OLS estimates using the 75 observations 1-75																		
VARIABLE COEFFICIENT STDERROR T STAT P-VALUE const 118.914 6.35164 18.722 <0.00001	1.00			10001															
const 118.914 6.35164 18.722 <0.00001 ***		V.	ARIABI	LE		COEFI	FICI	ENT		2	STDER	ROR		тз	TAT	P-	VALUE		
P -7.90785 1.09599 -7.215 <0.00001 ****	c	onst				118.	.914				6.35	164		18.	722	<0.	00001	* * *	
<pre>A 1.86258 0.683195 2.726 0.00804 *** Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515 </pre>	P					-7.	.9078	85			1.09	599		-7.3	215	<0.	00001	* * *	
<pre>Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515 </pre>	A					1.	.862	58			0.68	8195		2.	726	ο.	00804	***	
Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001)	"		af da.			eriek	-1		,	147									
Sum of squared residuals = 1718.94 Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515	n S	tand:	ard de	eviat	ion	aria. of de	ore · en. v	var.	57	6.488	354								
Standard error of residuals = 4.88612 Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515	ŝ	um o:	f sau	ared	resi	duals	з = :	1718	.94	1 1									
Unadjusted R-squared = 0.448258 Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515	S	tand	ard e	rror	of r	esidu	uals	= 4	1.88	8612									
Adjusted R-squared = 0.432932 F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515	U.	nadj	usted	R-sq	luare	d = 0	0.448	8258	}			Overa	ILE.	tost	ofr	ograe	sion		
F-statistic (2, 72) = 29.2479 (p-value < 0.00001) Significance Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515	A	djus [.]	ted R	-squa	red	= 0.4	4329:	32				Overa			011	egres	sion		
Log-likelihood = -223.87 Akaike information criterion (AIC) = 453.739 Schwarz Bayesian criterion (BIC) = 460.691 Hannan-Quinn criterion (HQC) = 456.515	F-statistic (2, 72) = 29.2479 (p-value < 0.00001) significance																		
Schwarz Bayesian criterion (BIC) = 456.515 Hannan-Quinn criterion (HQC) = 456.515	Log-11Kellhood = -223.87																		
Hannan-Quinn criterion (HQC) = 456.515	Schwarz Bayesian criterion (BTC) = 460 691																		
	Hannan-Ouinn criterion (HOC) = 456.515																		
								-											
Close																			

6.3 Extended Model

In the extended model, we add the squared level of advertising to the model, A^2 , which permits the possibility of diminishing returns to advertising. The model to be estimated is

$$S_i = \beta_1 + \beta_2 P_i + \beta_3 A_i + \beta_4 A_i^2 + e_i \tag{6.3}$$

This time, open a console window from the toolbar by clicking on the **open gretl console** button; then, generate a new variable, A^2 using genr A2 = A*A. Then estimate (6.3) using the command: ols S const P A A2. This yields the output in Figure 6.8:

6.3.1 Is Advertising Significant?

The marginal effect of another unit of advertising on average sales is

$$\frac{\partial E[Sales_i]}{\partial A_i} = \beta_3 + 2\beta_4 A_i \tag{6.4}$$

This means that the effect of another unit of advertising depends on the current level of advertising, A_i . To test for the significance of all levels of advertising requires you to test the joint hypothesis $H_0: \beta_3 = \beta_4 = 0$ against the alternative $H_a: \beta_3 \neq 0$ or $\beta_4 \neq 0$. From the console, following estimation of the full model, type omit A A2 and gretl will execute the omit variables test discussed in the preceding section. The console window is shown in Figure 6.9 below and the outcome from the omit test is highlighted.

Figure 6.8: The results of the extended model of Big Andy's Burger Barn obtained from the **gretl** console.

🙀 gretl console						
🔏 📇 🛱 🔍 🗙						
<pre>gret1 console: type 'help' for a list of commands ? genr &2 = A*A Generated vector &2 (ID 9) ? ols S const P & &2 Model 3: OLS estimates using the 75 observations 1-75</pre>						
Dependent variable	: 5					
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE		
const P A A2	109.719 -7.64000 12.1512 -2.76796	6.79905 1.04594 3.55616 0.940624	16.137 -7.304 3.417 -2.943	<0.00001 <0.00001 0.00105 0.00439	* * * * * * * * *	
<pre>Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1532.08 Standard error of residuals = 4.64528 Unadjusted R-squared = 0.508235 Adjusted R-squared = 0.487456 F-statistic (3, 71) = 24.4593 (p-value < 0.00001) Log-likelihood = -219.554 Akaike information criterion (AIC) = 447.108 Schwarz Bayesian criterion (BIC) = 456.378 Hannan-Quinn criterion (HQC) = 450.809</pre>						
		Close				

6.3.2 Optimal Level of Advertising

The optimal level of advertising is that amount where the last dollar spent on advertising results in only 1 dollar of additional sales (we are assuming here that the marginal cost of producing and selling another burger is zero!). Find the level of level of advertising, A_o , that solves:

$$\frac{\partial E[Sales_i]}{\partial A_i} = \beta_3 + 2\beta_4 A_o = \$1 \tag{6.5}$$

Plugging in the least squares estimates from the model and solving for A_o can be done in **gretl**. A little algebra yields

$$A_o = \frac{\$1 - \beta_3}{2\beta_4} \tag{6.6}$$

The script in **gretl** to compute this follows.

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
genr A2 = A*A
ols S const P A A2
genr Ao =(1-$coeff(A))/(2*$coeff(A2))
```

which generates the result:

Figure 6.9: Testing the significance of Advertising using the omit statement from the console.

👷 greti console					
? omit & &2	<u> </u>				
Model 4: OLS estimates using the 75 observations 1-75 Dependent variable: S					
VARIABLE COEFFICIENT STDERROR T STAT P-VAL	LUE				
const 121.900 6.52629 18.678 <0.000 P -7.82907 1.14286 -6.850 <0.000	001 *** 001 ***				
Mean of dependent variable = 77.3747 Standard deviation of dep. var. = 6.48854 Sum of squared residuals = 1896.39 Standard error of residuals = 5.09686 Unadjusted R-squared = 0.382963 Degrees of freedom = 73 Log-likelihood = -227.554 Akaike information criterion (AIC) = 459.107 Schwarz Bayesian criterion (BIC) = 463.742 Hannan-Quinn criterion (HQC) = 460.958 Comparison of Model 3 and Model 4: Null hypothesis: the regression parameters are zero for the variables A A2					
Test statistic: $F(2, 71) = 8.44136$, with p-value = 0.000514159 Of the 3 model selection statistics, 0 have improved.					
?	•				
Close					

? genr Ao =(1-\$coeff(A))/(2*\$coeff(A2))
Generated scalar Ao (ID 7) = 2.01434

This implies that the optimal level of advertising is estimated to be approximately \$2014.

To test the hypothesis that \$1900 is optimal (remember, A is measured in \$1000)

```
H_o: \quad \beta_3 + 2\beta_4 1.9 = 1 
H_1: \quad \beta_3 + 2\beta_4 1.9 \neq 1
```

you can use a t-test or an F-test. Following the regression, use

```
restrict
  b[3] + 3.8*b[4]=1
end restrict
```

Remember that b[3] refers to the coefficient of the third variable in the regression (A) and b[4] to the fourth. The output from the script is shown in Figure 6.10.

Figure 6.10: Testing whether \$1900 in advertising is optimal using the restrict statement.

```
🦉 gretl: script output
                                                                             <u>- 🗆 ×</u>
 🔏 占 🛅 🖻 🖉 🥱 🗙
   SCHWARZ BAYESIAN CRICERION
                             (BIC)
                                                                                 Hannan-Quinn criterion (HQC) = 450.809
 ? genr Ao =(1-$coeff(A))/(2*$coeff(A2))
 Generated scalar Ao (ID 5) = 2.01434
 ? restrict
 ? b3 + 3.8*b4=1
 ? end restrict
 Restriction:
 b[A] + 3.8*b[A2] = 1
 Test statistic: F(1, 71) = 0.936195, with p-value = 0.336543
 Restricted estimates:
       VARIABLE
                     COEFFICIENT
                                        STDERROR
                                                      T STAT P-VALUE
                      110.359
                                         6.76380
                                                      16.316 <0.00001 ***
   const
                                                      -7.277 <0.00001 ***
   Ρ
                       -7.60310
                                         1.04478
                       11.9308
                                         3.54728
                                                               0.00124 ***
   A
                                                       3.363
   A2
                        -2.87652
                                         0.933496
                                                      -3.081
                                                               0.00292 ***
   Standard error of residuals = 4.64322
                                                                                 -
                                       Close
```

6.4 Nonsample Information

In this section we'll estimate the beer demand model. The data are in beer.gdt and are in level form. The model to be estimated is

$$\ln(Q_i) = \beta_1 + \beta_2 ln(PB_i) + \beta_3 ln(PL_i) + \beta_4 ln(PR_i) + \beta_5 ln(M_i) + e_i$$
(6.7)

The first thing to do is to convert each of the variables into natural logs. **Gretl** has a built in function for this that is very slick. From the main window, highlight the variables you want to transform with the cursor. Then go to Add>Logs of selected variables from the pull-down menu as shown in Figure 6.11. This can also be done is a script or from the console using the

- D X 📓 greti <u>File T</u>ools Data ⊻iew Add Sample Variable Model Help beer.gdt * Logs of selected variables k Squares of selected variables ID# Variable name Descr Lags of selected variables 0 const auto First differences of selected variables Inca М Log differences of selected variables 2 PB Price Seasonal differences of selected variables 3 PL Price 4 PR Price Index variable 5 Q Quan Time trend Random variable Periodic dummies Unit dummies Time dummies Dummies for selected discrete variables Define new variable... ◀ Þ Undated: Full range 1 - 30 🖩 🕑 🗖 🎟 🐻 🔁 Â

Figure 6.11: Use the pull-down menu to add the natural logs of each variable

command logs Q PB PL PR M. The natural log of each of the variables is obtained and the result stored in a new variable with the prefix l_{-} ("el" underscore).

No money illusion can be parameterized in this model as $\beta_2 + \beta_3 + \beta_4 + \beta_5 = 0$. This restriction is easily estimated within **gretl** using the restrict dialog or a script as shown below.

open "c:\Program Files\gretl\data\poe\beer.gdt"

Figure 6.12: **gretl** output for the beer demand

```
? restrict
? b2+b3+b4+b5=0
? end restrict
Restriction:
b[1_PB] + b[1_PL] + b[1_PR] + b[1_M] = 0
Test statistic: F(1, 25) = 2.49693, with p-value = 0.126639
Restricted estimates:
      VARIABLE COEFFICIENT
                                 STDERROR
                                                T STAT
                                                         P-VALUE
  const
                   -4.79780
                                  3.71390
                                                -1.292
                                                         0.20778
  1_PB
                  -1.29939
                                  0.165738
                                                -7.840
                                                        <0.00001 ***
  1_PL
                   0.186816
                                  0.284383
                                                 0.657
                                                         0.51701
  1_PR
                   0.166742
                                  0.0770752
                                                 2.163
                                                         0.03989 **
  l_M
                   0.945829
                                  0.427047
                                                 2.215
                                                         0.03574 **
  Standard error of residuals = 0.0616756
logs Q PB PL PR M
ols l_Q const l_PB l_PL l_PR l_M
restrict
```

b2+b3+b4+b5=0 end restrict

The syntax for the restrictions is new. Instead of b[2]+b[3]+b[4]+b[5]=0 a simpler form is used. This is undocumented in the version I am using (1.6.5) and I am uncertain of whether this will continue to work. It does for now and I've shown it here. Apparently **gretl** is able to correctly parse the variable number from the variable name without relying on the brackets. The output from the **gretl** script output window appears in Figure 6.12.

6.5 Model Specification

There are several issues of model specification explored here. First, it is possible to omit relevant independent variables from your model. A **relevant independent variable** is one that affects the mean of the dependent variable. When you omit a relevant variable that happens to be correlated with any of the other included regressors, least squares suffers from omitted variable bias.

The other possibility is to include **irrelevant variables** in the model. In this case, you include extra regressors that either don't affect y or, if they do, they are not correlated with any of the other regressors. Including irrelevant variables in the model makes least squares less precise than it otherwise would be—this increases standard errors, reduces the power of your hypothesis tests, and increases the size of your confidence intervals.

The example used in the text uses the dataset $edu_inc.gdt$. The first regression

$$faminc_i = \beta_1 + \beta_2 * he_i + \beta_3 we_i + \beta_4 k l 6_i + \beta_5 x_{i5} + \beta_6 x_{i6} + e_i$$
(6.8)

where *faminc* is family income, he is husband's years of schooling, we is woman's years of schooling, and kl6 are the number of children in the household under age 6. Several variations of this model are estimated. The first includes only he, another only he and we, and one includes the two irrelevant variables, x_5 and x_6 . The **gretl** script to estimate these models and test the implied hypothesis restrictions follows. If you type this in yourself, omit the line numbers.

#line	code
01	<pre>open "c:\Program Files\gretl\data\poe\edu_inc.gdt"</pre>
02	ols faminc const he we kl6 x5 x6
03	modeltab add
04	omit x5 x6
05	modeltab add
06	omit kl6
07	modeltab add
08	omit we
09	modeltab add
10	modeltab show

The models can be estimated and saved as icons (File>Save to session as icon) within gretl. Once they've all been estimated and saved as icons, open a session window (Figure 1.10) and drag each model onto the *model table* icon. Click on the model table icon to reveal the output shown in Figure 6.13.

In the above script, we've used the modeltab function after each estimated model to add it to the model table. The final line tells gretl to display (show) the resulting model table.

One word of caution is in order about the given script and its interpretation. The **omit** statement tests the implied restriction (the coefficient on the omitted variable is zero) versus the estimated model that **immediately precedes it**. Thus, when we test that the coefficient on k16 is zero in line 06, the alternative model is the restricted model from line 04, which already excludes x5, and x6. Thus, only one restriction is being tested. If your intention is to test all of the restrictions (omit x5, x6 and k16) versus the the completely unrestricted model in line 02 that includes all of the variables, you'll need to modify your code. I'll leave this an an exercise.

Figure 6.13: Save each model as an icon. Open the session window and drag each model to the model table icon. Click on the model table icon to reveal this output.

👰 gretl: model	l table				
🛛 🕹 🖬	C TEX X				
OLS estima Dependent	ates variable: famin	nc			
	Model 1	Model 2	Model 3	Model 4	
const	-7559 (1.120e+04)	-7755 (1.116e+04)	-5534 (1.123e+04)	2.619e+04** (8541)	
he	3340** (1250)	3212** (796.7)	3132** (802.9)	5155** (658.5)	
we	5869** (2278)	4777** (1061)	4523** (1066)		
k16	-1.420e+04** (5044)	-1.431e+04** (5004)			
x5	888.8 (2242)				
x6	-1067 (1982)				
n Adj. R**2 lnL	428 0.1681 -5142.20	428 0.1714 -5142.37	428 0.1574 -5146.45	428 0.1237 -5155.33	
Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level					
			Close		

6.6 RESET

The **RESET** test is used to assess the adequacy of your functional form. The null hypothesis is that your functional form is adequate. The alternative is that it is not. The test involves running a couple of regressions and computing an F-statistic.

Consider the model

$$y_i = \beta_1 + \beta_2 x_{i2} + \beta_3 x_{i3} + e_i \tag{6.9}$$

and the hypothesis

$$H_0: \quad E[y|x_{i2}, x_{i3}] = \beta_1 + \beta_2 x_{i2} + \beta_3 x_{i3}$$

$$H_1: \quad \text{not } H_0$$

Rejection of H_0 implies that the functional form is not supported by the data. To test this, first estimate (6.9) using least squares and save the predicted values, \hat{y}_i . Then square and cube \hat{y} and add them back to the model as shown below:

$$y_{i} = \beta_{1} + \beta_{2}x_{i2} + \beta_{3}x_{i3} + \gamma_{1}\hat{y}_{i}^{2} + e_{i}$$

$$y_{i} = \beta_{1} + \beta_{2}x_{i2} + \beta_{3}x_{i3} + \gamma_{1}\hat{y}_{i}^{2} + \gamma_{2}\hat{y}_{i}^{3} + e_{i}$$

The null hypotheses to test (against alternative, 'not H_0 ') are:

$$H_0: \quad \gamma_1 = 0$$
$$H_0: \quad \gamma_1 = \gamma_2 = 0$$

Estimate the auxiliary models using least squares and test the significance of the parameters of the \hat{y}^s . This is accomplished through the following script. Note, the **reset** command issued after the first regression computes the test associated with $H_0: \gamma_1 = \gamma_2 = 0$. It is included here so that you can compare the 'canned' result with the one you compute using the two step procedure suggested above. The two results should match.

```
open "c:\Program Files\gretl\data\poe\cars.gdt"
ols mpg const cyl eng wgt
genr y = $yhat
genr y2 = y*y
genr y3 = y2*y
ols mpg const cyl eng wgt y2
omit y2
ols mpg const cyl eng wgt y2 y3
```

6.7 Cars Example

The data set cars.gdt is included in package of datasets that are distributed with this manual. The script to reproduce the results from your text is

open "c:\Program Files\gretl\data\poe\cars.gdt"

```
ols mpg const cyl eng wgt
vif
omit cyl eng
```

The test of the individual significance of cyl and eng can be read from the table of regression results. Neither are significant at the 5% level. The joint test of their significance is performed using the omit statement. The F-statistic is 4.298 and has a p-value of 0.0142. The null hypothesis is rejected in favor of their joint significance.

The new statement that requires explanation is **vif**. **vif** stands for variance inflation factor and it is used as a collinearity diagnostic by many programs, including **gret**l. The vif is closely related to the statistic suggested by Hill et al. (2007) who suggest using the R^2 from auxiliary regressions to determine the extent to which each explanatory variable can be explained as linear functions of the others. They suggest regressing x_j on all of the other independent variables and comparing the R_j^2 from this auxiliary regression to 10. If the R_j^2 exceeds 10, then there is evidence of a collinearity problem.

The vif_j actually reports the same information, but in a less straightforward way. The vif associated with the j^{th} regressor is computed

$$vif_{j} = \frac{1}{1 - R_{j}^{2}} \tag{6.10}$$

which is, as you can see, simply a function of the R_j^2 from the j^{th} regressor. Notice that when $R_j^2 > .80$, the $vif_j > 10$. Thus, the rule of thumb for the two rules is actually the same. A vif_j greater than 10 is equivalent to an R^2 greater than .8 from the auxiliary regression.

The output from **gretl** is shown below:

Variance Inflation Factors

Minimum possible value = 1.0 Values > 10.0 may indicate a collinearity problem

2)	cyl	10.516
3)	eng	15.786
5)	wgt	7.789
$VIF(j) = 1/(1 - R(j)^2)$, where R(j) is the multiple correlation coefficient between variable j and the other independent variables

Once again, the **gretl** output is very informative. It gives you the threshold for high collinearity $(vif_j) > 10$) and the relationship between vif_j and R_j^2 . Clearly, these data are highly collinear. Two variance inflation factors above the threshold and the one associated with wgt is fairly large as well.

The variance inflation factors can be produced from the dialogs as well. Estimate your model then, in the model window, select Tests>Collinearity and the results will appear in gretl's output.

6.8 Script

```
open "c:\Program Files\gretl\data\poe\andy.gdt"
ols S const P A
genr sseu = $ess
genr unrest_df = $df
ols S const A
genr sser = $ess
genr Fstat=((sser-sseu)/1)/(sseu/(unrest_df))
pvalue F 1 unrest_df Fstat
ols S const
genr sser = $ess
genr rest_df = $df
genr J = rest_df-unrest_df
genr Fstat=((sser-sseu)/J)/(sseu/(unrest_df))
pvalue F J unrest_df Fstat
genr A2 = A*A
ols S const P A A2
genr Ao =(1-$coeff(A))/(2*$coeff(A2))
restrict
 b3 + 3.8*b4=1
end restrict
```

```
open "c:\Program Files\gretl\data\poe\beer.gdt"
logs Q PB PL PR M
ols 1_Q const 1_PB 1_PL 1_PR 1_M
restrict
  b2+b3+b4+b5=0
end restrict
open "c:\Program Files\gretl\data\poe\edu_inc.gdt"
ols faminc const he we kl6 x5 x6
modeltab add
omit x5 x6
modeltab add
omit kl6
modeltab add
omit we
modeltab add
modeltab show
open "c:\Program Files\gretl\data\poe\cars.gdt"
ols mpg const cyl eng wgt
reset
ols mpg const cyl eng wgt
genr y = $yhat
genr y2 = y*y
genr y3 = y2*y
ols mpg const cyl eng wgt y2
omit y2
ols mpg const cyl eng wgt y2 y3
omit y2 y3
ols mpg const cyl eng wgt
vif
omit cyl eng
```

l Chapter

Nonlinear Relationships

In Chapter 7 of *Principles of Econometrics*, the authors consider several methods for modeling nonlinear relationships between economic variables. As they point out, if the slope (the effect of one variable on another) changes for any reason, then the relationship is nonlinear. Specifically, we examine the use of polynomials, dummy variables, and interaction effects to make the basic linear regression model much more flexible.

7.1 Polynomials

The first model considered is a basic wage equation, where the worker's wage depends of his level of education and experience. We suspect that there are diminishing returns to experience and hence that the wage benefit of another year of experience will decline as a work gains experience.

$$wage_t = \beta_1 + \beta_2 educ_t + \beta_3 exper_t + \beta_4 exper_t^2 + e_t$$

$$(7.1)$$

The marginal effect of another year of experience on the average wage is

$$\frac{\partial E(wage_t)}{\partial exper_t} = \beta_3 + 2\beta_4 exper_t \tag{7.2}$$

Diminishing returns implies that $\beta_3 > 0$ and $\beta_4 < 0$. The maximum is attained when the slope of the function is zero so setting equation (7.2) equal to zero and solving for *exper* defines the level of experience that we expect will maximize wages.

$$\beta_3 + 2\beta_4 exper_t = 0 \quad \text{and} \quad exper_t = -\beta_3/2\beta_4. \tag{7.3}$$

Using **gretl** and the 1000 observations from the 1997 CPS stored in the **gretl** dataset *cps_small.gdt* we use the following script:

open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr exper2 = exper^2
ols wage const educ exper exper2

which yields the result

$$\widehat{\text{wage}} = -\begin{array}{c} 9.818 \\ (-9.306) \end{array} + \begin{array}{c} 1.210 \\ (17.228) \end{array} \underbrace{\text{educ}}_{(6.629)} + \begin{array}{c} 0.341 \\ (-4.252) \end{array} \underbrace{\text{exper2}}_{(-4.252)} \\ T = 1000 \quad \overline{R}^2 = 0.2687 \quad F(3,996) = 123.38 \quad \widehat{\sigma} = 5.3417 \\ (t\text{-statistics in parentheses}) \end{array}$$

The marginal effect for someone with 18 years of experience is obtained by using the statement scalar me18=\$coeff(exper)+2*\$coeff(exper2)*18 which yields the desired result. Similarly, the turning point can be computed using the command scalar turnpt=-\$coeff(exper)/(2*\$coeff(exper2)). Notice, the scalar command is used instead of genr here because the result is a single number rather than a data series. You could use genr and obtain exactly the same result. There is little reason to prefer scalar over genr for the generation of scalars in gretl, but I've developed the habit of referring to a single number generated by other software as a scalar and I try to follow this convention in gretl. Actually, for scalars in gretl you could omit the prefix altogether. In recent versions of gretl it is unnecessary as you can see from Figure 7.1. In this figure, the same computation is made using genr, scalar, and without either!





As noted earlier **gretl** includes a data utility that makes it very easy to add the square of experience to your data file. First, select the variable you want to transform by highlighting it with the cursor. Then from the main **gretl** window use Add>Squares of selected variables to add them to the data set.

There are a number of other transformations you can make in this way, including add a time trend, logs, lags, differences and dummy variables for units or for panels. The pull-down list is illustrated in Figure 4.9.

As mentioned earlier some of the choices in Figure 4.9 are greyed out, meaning that they can not be selected at this time. This is because they are time series or panel specific functions and can only be used if you have first designated your data as such. To set your data up as time series use the Data>Dataset structure pull-down menu which is obtained as shown in Figure 7.2 below. Clicking on Dataset structure reveals the dialog box shown in Figure 7.3. If you select



Figure 7.2: Data>Dataset Structure pull-down menu

time series you will be taken to additional boxes that allow you to define its periodicity (yearly, quarterly, monthly, etc.) and the dates the time series covers. This is a very nice utility and I have used it to convert many of the *POE* datasets to time series for you. We will return to this topic in later chapters.

7.2 Interaction Terms

Another tool for capturing some types of nonlinearity is the creation of interaction terms. An **interaction term** is a variable that is created by multiplying two or more variables together. As discussed in *POE*, interaction terms are useful in allowing the marginal effect of a change in an independent variable on the average value of your dependent variable to be different for different observations in your sample. For instance, you may decide that the average return to another year of schooling is higher the younger a person is, other things being equal. To capture this effect in a model of wages you could create an interaction between years of schooling (S_i) and a person's age (A_i) by generating a new variable $SA_i = S_i * A_i$ and including it as a regressor in your model. This is the overall gist of the pizza example from your textbook, where a person's age and income

Figure 7.3: Dataset Structure dialog box

📓 Data structure wiza 💶 🗙		
Structure of dataset		
Cross-sectional		
O Time series		
O Panel		
X Cancel		

are interacted and included in a basic model of pizza demand.

7.3 Examples

7.3.1 Housing Price Example

The model to be estimated is

$$price_{t} = \beta_{1} + \delta_{1}utown_{t} + \beta_{2}sqft_{t} + \gamma sqft_{t} * utown_{t} + \beta_{3}age_{t} + \delta_{2}pool_{t} + \delta_{3}fplace_{t} + e_{t}$$

$$(7.4)$$

The script to estimate this model is

open "c:\Program Files\gretl\data\poe\utown.gdt"
genr p = price/1000
genr sqft_ut = sqft*utown
ols price const utown sqft sqft_ut age pool fplace

Notice that the dependent variable, *price*, has been rescaled to be measured in \$1,000 increments. This basically reduces the sizes of the estimated coefficients and standard errors by a factor of 1,000. It has no effect on t-ratios or their p-values. The results appear below.

$$\begin{split} \widehat{\mathbf{p}} &= \underbrace{24.5}_{(3.957)} + \underbrace{27.453}_{(3.259)} \operatorname{utown} + \underbrace{0.07612}_{(31.048)} \operatorname{sqft} + \underbrace{0.01299}_{(3.913)} \operatorname{sqft_ut} \\ &- \underbrace{0.190}_{(-3.712)} \operatorname{age} + \underbrace{4.377}_{(3.658)} \operatorname{pool} + \underbrace{1.649}_{(1.697)} \operatorname{fplace} \\ T &= 1000 \quad \bar{R}^2 = 0.8698 \quad F(6, 993) = 1113.2 \quad \hat{\sigma} = 15.225 \\ &\quad (t\text{-statistics in parentheses}) \end{split}$$

7.3.2 CPS Example

In this example, the *cps_small.dat* data are used to estimate wage equations. The basic equation is

$$wage_i = \beta_1 + \beta_2 educ_i + \delta_1 black_i + \delta_2 female_i + \gamma black_i * female_i + e_i$$
(7.5)

In this specification white-males are the **reference group**. The parameter δ_1 measures the effect of being black, the parameter δ_2 measures the effect of being female, and the parameter γ measures the effect of being black and female, all measured relative to the white-male reference group.

The first part of the script generates the interaction between females and blacks and then uses least squares to estimate the coefficients. The next line uses the omit statement to omit the three dummy variables (black, female, b_female) from the model to estimate a restricted version. Further, it performs the F-test of the joint null hypothesis that the three coefficients $(\delta_1, \delta_2, \gamma)$ are zero against the alternative that at least one of them is not.

The next model adds the three regional dummies and tests the null hypothesis that they are jointly zero. Then additional interactions are created between **south** and the other variables. Finally, **gretl**'s **wls** command is used to estimate separate regressions for southerners and non southerners. Here is the script file to compute all of the results for the the CPS examples.

```
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
```

The results are collected in model table 7.1 using the modeltab function.

Table 7.1: CPS results

OLS estimates Dependent variable: wage

	Model 1	Model 2	Model 3	Model 4	Model 5
const	-3.230^{**} (0.9675)	-4.912^{**} (0.9668)	-2.456^{**} (1.051)	-3.230^{**} (0.9675)	-3.578^{**} (1.151)
educ	1.117^{**} (0.06971)	1.139^{**} (0.07155)	1.102^{**} (0.06999)	1.117^{**} (0.06971)	1.166^{**} (0.08241)
black	-1.831^{**} (0.8957)		-1.608^{*} (0.9034)	-1.831^{**} (0.8957)	-0.4312 (1.348)
female	-2.552^{**} (0.3597)		-2.501^{**} (0.3600)	-2.552^{**} (0.3597)	-2.754^{**} (0.4257)
b_female	0.5879 (1.217)		$0.6465 \\ (1.215)$	0.5879 (1.217)	0.06732 (1.906)
south			-1.244^{**} (0.4794)		1.302 (2.115)
midwest			-0.4996 (0.5056)		
west			-0.5462 (0.5154)		
ed_{-south}					-0.1917 (0.1542)
b_south					-1.744 (1.827)
f_south					0.9119 (0.7960)
b_f_sth					0.5428 (2.511)
$egin{array}{c} n \ ar{R}^2 \ \ell \end{array}$	$1000 \\ 0.2451 \\ -3107.86$	$1000 \\ 0.2016 \\ -3137.43$	$1000 \\ 0.2482 \\ -3104.33$	$1000 \\ 0.2451 \\ -3107.86$	$1000 \\ 0.2490 \\ -3102.81$

Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level

7.3.3 Chow Test

The Chow test is an easy way to test the equivalency of two regressions estimated using different subsets of the sample. In this section I'll show you a trick that you can use for estimating subset regressions and then how to perform the Chow test.

Suppose you wanted to estimate separate wage equations based on the model in equation (7.5): one regression for southerners and another for everyone else. **Gretl** can accomplish this using the weighted least squares, wls, estimator. The weighted least squares estimator takes the model

$$y_i = \beta_1 + \beta_2 x_{i2} + \beta_3 x_{i3} + \ldots + \beta_k x_{ik} + e_i \tag{7.6}$$

and reweighs it using weights, w_i according to

$$w_i * y_i = \beta_1 w_i + \beta_2 w_i * x_{i2} + \beta_3 w_i * x_{i3} + \ldots + \beta_k w_i * x_{ik} + w_i * e_i$$
(7.7)

and estimates the coefficients using least squares. This estimator is used later in the book for different purposes, but here it can be used to omit desired observations from your model. Basically, what you want to do is to let $w_i = 1$ for all observations you want to include and $w_i = 0$ for those you want to exclude.

The syntax for the wls command is simple.

wls w y const x2 x3 x4

First call for the weighted least squares estimator with wls; next specify the weights to be used (w); then, state the regression to be estimated y const x2 x3 x4.

In the context of equation (7.5) generate a new dummy variable that takes the value 1 for nonsoutherners and zero for southerners; then, use weighted least squares. The following script uses this approach to estimate the two sample subsets. The sum of squared errors are saved for later use.

wls nonsouth wage const educ black female b_female scalar sse_ns = \$ess wls south wage const educ black female b_female scalar sse_s = \$ess

If the coefficients for southerners are equal to those for nonsoutherners, then you would **pool** the two subsamples together and estimate the model using the command **ols wage const educ black female b_female**. Otherwise, separate regressions are required. The **Chow test** is used to determine whether the subsamples are really necessary in light of the data you have. To determine whether the regressions were actually equal to one another compute

$$Chow = \frac{SSE_{full} - (SSE_{south} + SSE_{nonsouth})/5}{(SSE_{south} + SSE_{nonsouth})/(n-10)} \sim F_{5,n-10}$$
(7.8)

if the two subset regressions are equivalent. You will reject the null hypothesis that the coefficients of the two subsamples are equal if the p-value is less than the desired significance level of the test, α .

The script to compute the Chow test is:

```
ols wage const educ black female b_female
scalar sse_r = $ess
scalar sse_u = sse_ns+sse_s
scalar chowtest = ((sse_r-sse_u)/5)/(sse_u/($nobs-10)
pvalue F 5 $nobs-10 chowtest
```

As you can see, this is just an application of the F-statistic of equation (6.2) discussed in Chapter 6. The unrestricted sum of squares is obtained by adding the sum of squared errors of the two subset regressions. The restricted sum of square errors is from the pooled regression.

7.3.4 Pizza Example

ols piz const age y

The pizza examples considers the model

$$piz_i = \beta_1 + \beta_2 age_i + \beta_3 y_i + \beta_4 age_i * y_i + e_i \tag{7.9}$$

where i = 1, 2, ..., T. The marginal effects of age on pizza demand are computed for families having \$25,000 and \$90,000 in income. The **gretl** code to estimate this model using least squares and to obtain the marginal effects is:

open "c:\Program Files\gretl\data\poe\pizza.gdt"

```
genr age_inc = age*y
ols piz const age y age_inc
scalar p25 = $coeff(age)+$coeff(age_inc)*25000
scalar p90 = $coeff(age)+$coeff(age_inc)*90000
```

The estimates from the first equation are:

$$\widehat{\text{piz}} = 342.885 + 0.00238222 \text{ y} - 7.57556 \text{ age} (4.740) + 0.00238222 \text{ y} - 7.57556 \text{ age} (-3.270) T = 40 \quad \overline{R}^2 = 0.2930 \quad F(2,37) = 9.0811 \quad \widehat{\sigma} = 131.07 (t-\text{statistics in parentheses})$$

Table 7.3: Regression results for the Chow test.

Dependent variable: wage

	Model 7	Model 8	Model 9
	Non-South	South	All
const	-3.578^{**}	-2.275	-3.230^{**}
	(1.211)	(1.555)	(0.9675)
educ	1.166^{**}	0.9741^{**}	1.117^{**}
	(0.08665)	(0.1143)	(0.06971)
black	-0.4312	-2.176^{**}	-1.831^{**}
	(1.418)	(1.080)	(0.8957)
female	-2.754^{**}	-1.842^{**}	-2.552^{**}
	(0.4476)	(0.5896)	(0.3597)
b_female	0.06732	0.6102	0.5879
	(2.004)	(1.433)	(1.217)
n	685	315	1000
$ar{R}^2$	0.2486	0.2143	0.2451
SSE	22031.3	6981.39	29307.7

Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level

The results of the test are:

? scalar chowtest = ((sse_r-sse_u)/5)/(sse_u/(\$nobs-10)
Generated scalar chowtest (ID 20) = 2.01321
? pvalue F 5 \$nobs-10 chowtest

F(5, 990): area to the right of 2.01321 = 0.074379(to the left: 0.925621) and those from the second:

$$\vec{piz} = 342.885 + 0.00238222 \text{ y} - 7.57556 \text{ age}$$

$$(4.740) \quad T = 40 \quad \bar{R}^2 = 0.2930 \quad F(2,37) = 9.0811 \quad \hat{\sigma} = 131.07$$

$$(t\text{-statistics in parentheses})$$

and the computed predictions are:

p25 = -6.98270p90 = -17.3964

7.3.5 Log-Linear Wages Example

In the final example a model of log-wages is estimated and the **genr** command is used to compute the percentage difference between male and female wages and the marginal effect of another year of experience on the log-wage.

```
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ female
scalar pdiff = exp($coeff(female))-1
genr expersq = exper*exper
genr educ_exp = educ*exper
ols l_wage const educ exper educ_exper
scalar me = 100*($coeff(exper)+$coeff(educ_exp)*16)
ols l_wage const educ exper expersq educ_exper
```

The results from the three regressions appear in Table 7.5.

7.4 Script

```
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr exper2 = exper^2
ols wage const educ exper exper2
scalar me18 = $coeff(exper)+2*$coeff(exper2)*18
scalar turnpt = -($coeff(exper))/(2*$coeff(exper2))
```

Table 7.5: The regression results from the log-linear wages example.

OLS estimates Dependent variable: ln(wage)

	Model 1	Model 2	Model 3
const	0.9290^{**} (0.08375)	0.1528 (0.1722)	-0.2646 (0.1808)
educ	0.1026^{**} (0.006075)	0.1341^{**} (0.01271)	0.1506^{**} (0.01272)
female	-0.2526^{**} (0.02998)		
exper		0.02492^{**} (0.007075)	0.06706^{**} (0.009533)
educ_exp		-0.0009624^{*} (0.0005404)	-0.002019^{**} (0.0005545)
expersq			-0.0006962^{**} (0.0001081)
$n \over \bar{R}^2$	$1000 \\ 0.2654 \\ 0.2654$	1000 0.2785	1000 0.3067
ℓ	-670.50	-660.97	-640.54

Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level

```
? scalar pdiff = exp($coeff(female))-1
Generated scalar pdiff (ID 11) = -0.223224
? scalar me = 100*($coeff(exper)+$coeff(educ_exp)*16)
Generated scalar me (ID 14) = 0.951838
```

```
open "c:\Program Files\gretl\data\poe\utown.gdt"
genr p = price/1000
genr sqft_ut = sqft*utown
ols price const utown sqft sqft_ut age pool fplace
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr b_female = black*female
ols wage const educ black female b_female
omit black female b_female
ols wage const educ black female b_female south midwest west
omit south midwest west
genr ed_south = educ*south
genr b_south = black*south
genr f_south = female*south
genr b_f_sth = black*female*south
#Use omit statement to test joint hypothesis
ols wage const educ black female b_female south \
          ed_south b_south f_south b_f_sth
omit south ed_south b_south f_south b_f_sth
#Using wls to omit observations
genr nonsouth = 1-south
wls nonsouth wage const educ black female b_female
scalar sse_ns = $ess
wls south wage const educ black female b_female
scalar sse_s = $ess
#Chow test
#Pooled regression (restricted)
ols wage const educ black female b_female
scalar sse_r = $ess
scalar sse_u = sse_ns+sse_s
scalar chowtest = ((sse_r-sse_u)/5)/(sse_u/($nobs-10)
pvalue F 5 $nobs-10 chowtest
#Pizza Example
open "c:\Program Files\gretl\data\poe\pizza.gdt"
ols piz const age y
genr age_inc = age*y
ols piz const age y age_inc
```

```
scalar p25 = $coeff(age)+$coeff(age_inc)*25000
scalar p90 = $coeff(age)+$coeff(age_inc)*90000
#Log wages example
open "c:\Program Files\gretl\data\poe\cps_small.gdt"
genr l_wage = log(wage)
ols l_wage const educ female
scalar pdiff = exp($coeff(female))-1
genr expersq = exper*exper
genr educ_exp = educ*exper
ols l_wage const educ exper educ_exper
scalar me = 100*($coeff(exper)+$coeff(educ_exp)*16)
```

Chapter 8

Heteroskedasticity

The simple linear regression models of Chapter 2 and the multiple regression model in Chapter 5 can be generalized in other ways. For instance, there is no guarantee that the random variables of these models (either the y_i or the e_i) have the same inherent variability. That is to say, some observations may have a larger or smaller variance than others. This describes the condition known as **heteroskedasticity**. The general linear regression model is shown in equation (8.1) below.

$$y_i = \beta_1 + \beta_2 x_{i2} + \ldots + \beta_k x_{iK} + e_i \quad i = 1, 2, \ldots, T$$
(8.1)

where y_i is the dependent variable, x_{ik} is the i^{th} observation on the k^{th} independent variable, $k = 2, 3, \ldots, K$, e_i is random error, and $\beta_1, \beta_2, \ldots, \beta_K$ are the parameters you want to estimate. Just as in the simple linear regression model, e_i , have an average value of zero for each value of the independent variables and are uncorrelated with one another. The difference in this model is that the variance of e_i now depends on i, i.e., the observation to which it belongs. Indexing the variance with the i subscript is just a way of indicating that observations may have different amounts of variability associated with them. The error assumptions can be summarized as $e_i|x_{i2}, x_{i3}, \ldots x_{iK} \text{ iid } N(0, \sigma_i^2)$.

The intercept and slopes, β_1 and $\beta_2 \dots \beta_K$, are consistently estimated by least squares even if the data are heteroskedastic. Unfortunately, the usual estimators of the least squares standard errors and tests based on them are inconsistent and invalid. In this chapter, several ways to detect heteroskedasticity are considered. Also, statistically valid ways of estimating the parameters of 8.1 and testing hypotheses about the β s when the data are heteroskedastic are explored.

8.1 Food Expenditure Example

First, a simple model of food expenditures is estimated using least squares. The model is

$$y_i = \beta_1 + \beta_2 x_i + e_i \quad i = 1, 2, \dots, N.$$
 (8.2)

where y_i is food expenditure and x_i is income of the i^{th} individual. When the errors of the model are heteroskedastic, then the least squares estimator of the coefficients is consistent. That means that the least squares *point estimates* of the intercept and slope are useful. However, when the errors are heteroskedastic the usual least squares **standard errors** are **inconsistent** and therefor should not be used to form confidence intervals or to test hypotheses.

To use least squares estimates with heteroskedastic data, at a very minimum, you'll need a consistent estimator of their standard errors in order to construct valid tests and intervals. A simple computation proposed by White accomplishes this. Standard errors computed using White's technique are loosely referred to as **robust**, though one has to be careful when using this term; the standard errors are robust to the *presence of heteroskedasticity* in the errors of model (but not necessarily other forms of model misspecification).

Open the *food.gdt* data in **gretl** and estimate the model using least squares.

```
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
```

This yields the usual least squares estimates of the parameters, but the wrong standard errors when the data are heteroskedastic. To obtain the robust standard errors, simply add the --robust option to the regression as shown in the following **gretl** script. After issuing the --robust option, the standard errors stored in **\$stderr(x)** are the robust ones.

ols y const x --robust

```
# confidence intervals (Robust)
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
print lb ub
```

In the script, we've used the critical(t,\$df,0.025) function to get the desired critical value from the t-distribution. Remember, the degrees of freedom from the preceding regression are stored in \$df. The first argument in the function indicates the desired distribution, and the last is the desired right-tail probability ($\alpha/2$ in this case).

This can also be done from the pull-down menus. Select Model>Ordinary Least Squares (see Figure 2.6) to generate the dialog to specify the model shown in Figure 8.1 below. Note, the check box to generate 'robust standard errors' is highlighted in yellow. You will also notice that there is a button labeled 'configure' just to the right of the check box. Clicking this button reveals a dialog from which several options can be selected. In this case, we can select the particular method that will be used to compute the robust standard errors and even set robust standard errors to be the default computation for least squares. This dialog box is shown in Figure 8.2 below. To reproduce the results in Hill et al. (2007), you'll want to select HC1 from the pull-down list. As you can see, other gret1 options can be selected here that affect the default behavior of the program.

gretl: specify model	
	OLS
const Y	Dependent variable
x	Set as default
	Independent variables
	Add ->
	<- Remove
Robust standard errors	configure
🔯 Help 🛛 🗞 🖸	lear 🎗 Cancel 🖉 OK

Figure 8.1: Check the box for heterosked asticity robust standard errors.

📓 gretl: options	$\overline{\mathbf{X}}$
General Databases Programs HCCME Manuals	
For cross-sectional data	HC1
For time-series data	HAC 💌
For panel data	Arellano 🔽
For GARCH estimation	QML 🔽
Use robust covariance matrix by default	
	Help
<u>Apply</u> <u>Cancel</u>	<u>o</u> k

Figure 8.2: Set the method for computing robust standard errors.

The model results for the food expenditure example appears in the table below. After estimating the model using the dialog, you can use Analysis>Confidence intervals for coefficients to generate 95% confidence intervals. Since you used the robust option in the dialog, these will be based on the variant of White's standard errors chosen using the 'configure' button. The result is:

t(38, .025) = 2.024

VARIABLE	COEFFICIENT	95% CONFIDENCE INTERVAL
const	83.4160	(27.8186, 139.013)
x	10.2096	(6.54736, 13.8719)

8.2 Weighted Least Squares

If you know something about the structure of the heteroskedasticity, you may be able to get more precise estimates using a generalization of least squares. In heteroskedastic models, observations that are observed with high variance don't contain as much information about the location of the regression line as those observations having low variance. The basic idea of generalized least squares in this context is to reweigh the data so that all the observations contain the same level of information (i.e., same variance) about the location of the regression line. So, observations that contain more **noise** are given small weights and those containing more **signal** a higher weight. Reweighing the data in this way is known in some statistical disciplines as *weighted least squares*. Table 8.1: Least squares estimates with the usual and robust standard errors.

	Usual Std errors	Robust Std errors
const	83.42*	83.42**
	(43.41)	(27.46)
x	10.21^{**}	10.21^{**}
	(2.093)	(1.809)
n	40	40
R^2	0.3850	0.3850
ℓ	-235.51	-235.51

OLS estimates Dependent variable: y

Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level

This descriptive term is the one used by **gretl** as well.

Suppose that the errors vary proportionally with x_i according to

$$Var(e_i) = \sigma^2 x_i \tag{8.3}$$

The errors are heteroskedastic since each error will have a different variance, the value of which depends on the level of x_i . Weighted least squares reweighs the observations in the model so that each transformed observation has the same variance as all the others. Simple algebra reveals that

$$\frac{1}{\sqrt{x_i}} Var(e_i) = \sigma^2 \tag{8.4}$$

So, multiply equation (8.1) by $1/\sqrt{x_i}$ to complete the transformation. The transformed model is homoskedastic and least squares and the least squares standard errors are statistically valid and efficient.

Gretl makes this easy since it contains a function to reweigh all the observations according to a weight you specify. The command is wls, which naturally stands for weighted least squares! The only thing you need to be careful of is how gretl handles the weights. Gretl takes the square root of the value you provide. That is, to reweigh the variables using $1/\sqrt{x_i}$ you need to use its square $1/x_i$ as the weight. Gretl takes the square root of w for you. To me, this is a bit confusing, so you may want to verify what gretl is doing by manually transforming y, x, and the constant and running the regression. The script file shown below does this.

In the example, you first have to create the weight, then call the function wls. The script appears below.

```
open "c:\Program Files\gretl\data\poe\food.gdt"
#GLS using built in function
genr w = 1/x
wls w y const x
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
print lb ub
#GLS using OLS on transformed data
genr wi = 1/sqrt(x)
genr ys = wi*y
genr xs = wi*x
genr cs = wi
ols ys cs xs
```

The first argument after wls is the name of the weight variable. Then, specify the regression to which it is applied. Gretl multiplies each variable (including the constant) by the *square root* of the given weight and estimates the regression using least squares.

In the next block of the program, $w_i = 1/\sqrt{x_i}$ is created and used to transform the dependent variable, x and the constant. Least squares regression using this manually weighted data yields the same results as you get with **gretl**'s **wls** command. In either case, you interpret the output of weighted least squares in the usual way.

The weighted least squares estimation yields:

and the 95% confidence interval for the slope β_2 is (7.64542, 13.2566).

8.3 Skedasticity Function

A commonly used model for the error variance is the **multipicative heteroskedasticity** model. It appears below in equation 8.5.

$$\sigma_i^2 = \exp\left(\alpha_1 + \alpha_2 z_i\right) \tag{8.5}$$

The variable z_i is an independent explanatory variable that determines how the error variance changes with each observation. You can add additional z_s if you believe that the variance is related to them (e.g., $\sigma_i^2 = \exp(\alpha_1 + \alpha_2 z_{i2} + \alpha_3 z_{i3})$). It's best to keep the number of z_s relatively small. The idea is to estimate the parameters of (8.5) using least squares and then use predictions as weights to transform the data.

In terms of the food expenditure model, let $z_i = ln(x_i)$. Then, taking the natural logarithms of both sides of (8.5) and adding a random error term, v_i , yields

$$\ln\left(\sigma_i^2\right) = \alpha_1 + \alpha_2 z_i + v_i \tag{8.6}$$

To estimate the α s, first estimate the linear regression (8.2) (or more generally, 8.1) using least squares and save the residuals. Square the residuals, then take the natural log; this forms an estimate of $\ln(\sigma_i^2)$ to use as the dependent variable in a regression. Now, add a constant and the zs to the right-hand side of the model and estimate the α s using least squares.

The regression model to estimate is

$$\ln\left(\hat{e}_{i}^{2}\right) = \alpha_{1} + \alpha_{2}z_{i} + v_{i} \tag{8.7}$$

where \hat{e}_i^2 are the least squares residuals from the estimation of equation (8.1). The predictions from this regression can then be transformed using the exponential function to provide weights for weighted least squares.

For the food expenditure example, the **gretl** code appears below.

```
ols y const x
genr lnsighat = log($uhat*$uhat)
genr z = log(x)
ols lnsighat const z
genr predsighat = exp($yhat)
genr w = 1/predsighat
wls w y const x
```

The first line estimates the linear regression using least squares. Next, a new variable is generated (lnsighat) that is the natural log of the squared residuals from the preceding regression. Then, generate z as the natural log of x. Estimate the skedasticity function using least squares, take the predicted values (yhat) and use these in the exponential function (i.e., $\exp(\hat{y}_i)$). The reciprocal of these serve as weights for generalized least squares. Remember, gretl automatically takes the square roots of w for you in the wls function.

This results in:

$$\hat{\mathbf{y}} = \begin{array}{c} 76.0538 + 10.6335 \,\mathrm{x} \\ _{(9.7135)}^{(9.7135)} & \begin{array}{c} (0.97151) \end{array} \end{array}$$
$$T = 40 \quad \bar{R}^2 = 0.7529 \quad F(1, 38) = 119.8 \quad \hat{\sigma} = 1.5467 \\ (\text{standard errors in parentheses}) \end{array}$$

8.4 Grouped Heteroskedasticity

Using examples from Hill et al. (2007) a model of grouped heteroskedasticity is estimated and a Goldfeld-Quandt test is performed to determine whether the two sample subsets have the same error variance.

8.4.1 Wage Example

Below, I have written a **gretl** program to reproduce the wage example from Hill et al. (2007) that appears in Chapter 8. The example is relatively straightforward and I'll not explain the script in much detail. It is annotated to help you decipher what each section of the program does.

The example consists of estimating wages as a function of education and experience. In addition, a dummy variable is included that is equal to one if a person lives in a metropolitan area. This is an "intercept" dummy which means that folks living in the metro areas are expected to respond similarly to changes in education and experience (same slopes), but that they earn a premium relative to those in rural areas (different intercept).

Each subset (metro and rural) is estimated separately using least squares and the standard error of the regression is saved for each (\$sigma). To create weights for weighted least squares, the full sample is restored and the metro and rural dummy variables are each multiplied times their respective regression's standard error. The two variables are added together to form a single variable that is equal to the metro regression standard error for each observation located in a metro area and equal to the rural regression standard error for each observation located in a rural area. The weight is created by taking the reciprocal and squaring. Recall, gret1 needs the variance rather than the standard error of each observation to perform weighted least squares.

```
open "c:\Program Files\gretl\data\poe\cps2.gdt"
ols wage const educ exper metro
# Use only metro observations
smpl metro --dummy
ols wage const educ exper
scalar stdm = $sigma
#Restore the full sample
smpl full
#Create a dummy variable for rural
genr rural = 1-metro
#Restrict sample to rural observations
smpl rural --dummy
```

```
ols wage const educ exper
scalar stdr = $sigma
#Restore the full sample
smpl full
#Generate standard deviations for each metro and rural obs
genr wm = metro*stdm
genr wr = rural*stdr
#Make the weights (reciprocal)
#Remember, Gretl's wls needs these to be variances
#so you'll need to square them
genr w = 1/(wm + wr)^2
#Weighted least squares
```

```
wls w wage const educ exper metro
```

Weighted least squares estimation yields:

$$\begin{split} \widehat{\text{wage}} &= -9.39836 + 1.19572 \text{ educ} + 0.132209 \text{ exper} + 1.53880 \text{ metro} \\ (1.0197) & (0.068508) & (0.014549) & (0.34629) \\ T &= 1000 \quad \bar{R}^2 = 0.2693 \quad F(3,996) = 123.75 \quad \hat{\sigma} = 1.0012 \\ & (\text{standard errors in parentheses}) \end{split}$$

The Goldfeld-Quandt statistic is the formed as the ratio of the two variances:

$$F = \frac{\hat{\sigma}_M^2}{\hat{\sigma}_R^2} \sim F_{N_M - K_M, N_R - K_R} \tag{8.8}$$

if the null hypothesis of homoskedasticity is true. Rejection of the Ho means that the subsets have different variances.

#Goldfeld Quandt statistic ?scalar fstatistic = stdm²/stdr² Generated scalar fstatistic (ID 17) = 2.08776

You could simplify the script a bit by using the regression trick explored in Chapter 7. Create a dummy variable that takes the value of 1 for the desired observations and 0 for the ones you want to drop. Then, use weighted least squares on the entire sample using the dummy variable as your weight. This effectively drops all observations in the sample for which the dummy variable is zero. This trick is useful since it keeps you from having to keep explicit track of which sample is actually in memory at any point in time. Thus,

smpl metro --dummy
ols wage const educ exper

could be replaced by

wls metro wage const educ exper

Then there is no need to restore the full sample in the next block of code!

8.4.2 Food Expenditure Example

In this example, the Goldfeld-Quandt test is applied in a more traditional way. Here, you suspect that the variance depends on a specific variable. You sort the data based on this variable and then compare the subset variances to one another using the Goldfeld-Quandt test statistic.

For the food expenditure example, the script follows. Essentially, you want to run two regressions using subsets of the sample. One subset contains observations with low variance, the other observations with high variance. In most cases this means that you'll have to sort your data based on its variability. In our example, one would sort the data based on x_i . Gretl gives us another option and that is to create the subsamples based on some criterion. In this case, we want observations separated based on high and low values of x so we can use the median function. To pick all observations for which x is above the median, use smpl x > median(x) --restrict. Recall that the smpl command allows us to manipulate the sample in memory. In this case we use the logical statement that we want observations where x is greater than the median of x, followed by the --restrict option. This should give us half the observations.

```
open "c:\Program Files\gretl\data\poe\food.gdt"
#Take subsample where x > median(x)
smpl x > median(x) --restrict
ols y const x
scalar stdL = $sigma
scalar df_L = $df
#Restore the full sample
smpl full
#Take subsample where x < median(x)
smpl x < median(x) --restrict
ols y const x
scalar stdS = $sigma
scalar df_S = $df</pre>
```

#Goldfeld Quandt statistic scalar fstatistic = stdL^2/stdS^2 pvalue F df_L df_S fstatistic

The full sample is restored and the variance for the lower half is saved. Then the test statistic is computed and can be compared to the appropriate critical value. The last statement computes the p-value from the F-distribution. Recall that the degrees of freedom were saved from each subset and they can be used here as the arguments for the numerator and denominator degrees of freedom for F.

The test statistic and p-value are:

```
? scalar fstatistic = stdL^2/stdS^2
Generated scalar fstatistic (ID 7) = 3.61476
? pvalue F df_L df_S fstatistic
```

F(18, 18): area to the right of 3.61476 = 0.00459643(to the left: 0.995404)

8.5 Other Tests for Heteroskedasticity

The Goldfeld-Quandt test of the null hypothesis of homoskedasticity is only useful when the data can be neatly partitioned into subsamples having different variances. In many circumstances this will not be the case and other tests of the homoskedasticity null hypothesis are more useful. Each of these tests share the same null hypothesis as the Goldfeld-Quandt test: homoskedasticity. They differ in the specification of the *alternative hypothesis*.

The first test considered is based on the estimated multiplicative heteroskedasticity function of section 8.3. The null and alternative hypotheses are

$$H_o: \quad \sigma_i^2 = \sigma^2 \tag{8.9}$$

$$H_1: \quad \sigma_i^2 = \exp\left(\alpha_1 + \alpha_2 z_i\right) \tag{8.10}$$

The homoskedastic null hypothesis is tested against a specific functional relationship. In this case, we know the function (exponential) as well as the variable(s) that causes the variance to vary (z_i) . Basically, we want to test whether $\alpha_2 = 0$. If it is, then the errors of the regression model are homoskedastic.

The test of this hypothesis is based on your regression in equation (8.7). The t-ratio on α_2 is approximately normally distributed under Ho so you could use the t-test to test this proposition. If you have multiple zs, use the F-test.

Other equivalent ways of testing this hypothesis are available. As Hill et al. (2007) point out, it is common to test the same hypothesis based on a linear regression

$$\hat{e}_i^2 = \alpha_1 + \alpha_2 z_i + v_i \tag{8.11}$$

Breusch and Pagan have proposed a couple of tests of the homoskedasticity hypothesis (8.9) against the alternative

$$H_1: \sigma_i^2 = h(\alpha_1 + \alpha_2 z_i) \tag{8.12}$$

where h() is some arbitrary function (e.g., linear or exponential). These tests are carried out based on equation (8.11). The alternative hypothesis in (8.12) is more general than that in (8.10) and includes it as a special case. There are two versions of the Breusch-Pagan test. One is used when the errors of the regression are normally distributed and the other when they are not. I suggest using the latter since it is seldom if ever known what the error distribution is. I'll tell you how to do the preferred version in **gret**.

Basically, estimate (8.11) using least squares and take NR^2 from this regression, where N is your sample size. Under the null hypothesis it has a χ^2_{S-1} distribution, where S is the total number of parameters (the α s) in the estimated equation.

The alternative hypothesis of White's test is even more general than the Breusch-Pagan. The alternative hypothesis is

$$H_1: \sigma_i^2 \neq \sigma^2 \tag{8.13}$$

Thus the alternative is completely general. The test is similar to the Breusch-Pagan test in that you'll run a regression with \hat{e}^2 as a dependent variable and zs as the independent variables. In White's test you will include each z, its square, and their (unique) cross products as regressors. In the food expenditure example that amounts to

$$y_i = \alpha_1 + \alpha_2 z_i + \alpha_3 z_i^2 + v_i \tag{8.14}$$

You can do this in one of two ways. You can run the original regression, save the residuals and square them. Then square z_i to use as an independent variable. Run the regression. For the food expenditure example:

```
open "c:\Program Files\gretl\data\poe\food.gdt"
```

ols y const x #Save the residuals

```
genr ehat = $uhat
```

#Square the residuals
genr ehat2 = ehat*ehat

```
#White's test
#Generate squares, cross products (if needed)
```

genr x2 = x*x
#Test regression
ols ehat2 const x x2
scalar teststat = \$trsq
pvalue X 2 teststat

Gretl computes NR^2 in every regression and saves it in **\$trsq**. The statistic NR^2 is distributed as a χ^2_{S-1} under the null hypothesis of homoskedasticity and we can use the **pvalue** function to obtain the p-value for the computed statistic. The syntax is **pvalue X** df statistic, with X indicating the χ^2 , df the degrees of freedom, and statistic the computed value of NR^2 . The script yields a computed test statistic of 7.555, and the p-value of 0.0228789. Homoskedasticity is rejected.

Or, you can use the **gretl** function modtest! In this case, run the original regression and follow it with modtest --white as shown in the script.

```
open "c:\Program Files\gretl\data\poe\food.gdt"
```

```
#White's test --built-in
ols y const x
modtest --white
```

This yields the output:

```
White's test for heteroskedasticity
OLS estimates using the 40 observations 1-40
Dependent variable: uhat<sup>2</sup>
```

VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE
const	-2908.78	8100.11	-0.359	0.72156
x	291.746	915.846	0.319	0.75186
sq_x	11.1653	25.3095	0.441	0.66167

Unadjusted R-squared = 0.188877

```
Test statistic: TR<sup>2</sup> = 7.555079,
with p-value = P(Chi-square(2) > 7.555079) = 0.022879
```

As you can see, the results from modtest --white and your (labor intensive) script are the same!

8.6 Script

```
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
ols y const x --robust
# confidence intervals (Robust)
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
print lb ub
#GLS using built in function
genr w = 1/x
wls w y const x
genr lb = $coeff(x) - critical(t,$df,0.025) * $stderr(x)
genr ub = $coeff(x) + critical(t,$df,0.025) * $stderr(x)
print lb ub
#GLS using OLS on transformed data
genr wi = 1/sqrt(x)
genr ys = wi*y
genr xs = wi*x
genr cs = wi
ols ys cs xs
#Estimating the skedasticity function and GLS
ols y const x
genr lnsighat = log($uhat*$uhat)
genr z = log(x)
ols lnsighat const z
genr predsighat = exp($yhat)
genr w = 1/predsighat
wls w y const x
#-----
#Wage Example
open "c:\Program Files\gretl\data\poe\cps2.gdt"
ols wage const educ exper metro
# Use only metro observations
smpl metro --dummy
ols wage const educ exper
```

```
scalar stdm = $sigma
#Restore the full sample
smpl full
#Create a dummy variable for rural
genr rural = 1-metro
#Restrict sample to rural observations
smpl rural --dummy
ols wage const educ exper
scalar stdr = $sigma
#Restore the full sample
smpl full
#Generate standard deviations for each metro and rural obs
genr wm = metro*stdm
genr wr = rural*stdr
#Make the weights (reciprocal)
#Remember, Gretl's wls needs these to be variances
#so you'll need to square them
genr w = 1/(wm + wr)^2
#Weighted least squares
wls w wage const educ exper metro
#Goldfeld Quandt statistic
scalar fstatistic = stdm<sup>2</sup>/stdr<sup>2</sup>
#-----
#Food Expenditure Example
open "c:\Program Files\gretl\data\poe\food.gdt"
#Take subsample where x > median(x)
smpl x > median(x) --restrict
ols y const x
scalar stdL = $sigma
scalar df_L = $df
#Restore the full sample
smpl full
```

```
#Take subsample where x < median(x)</pre>
smpl x < median(x) --restrict</pre>
ols y const x
scalar stdS = $sigma
scalar df_S = df
#Goldfeld Quandt statistic
scalar fstatistic = stdL^2/stdS^2
pvalue F df_L df_S fstatistic
#-----
#LM Test
open "c:\Program Files\gretl\data\poe\food.gdt"
ols y const x
#Save the residuals
genr ehat = $uhat
#Square the residuals
genr ehat2 = ehat*ehat
#White's test
#Generate squares, cross products (if needed)
genr x^2 = x x
#Test regression
ols ehat2 const x x2
scalar teststat = $trsq
pvalue X 2 teststat
#-----
#White's test
open "c:\Program Files\gretl\data\poe\food.gdt"
#White's test --built-in
ols y const x
```

```
modtest --white
```

Chapter 9

Dynamic Models and Autocorrelation

The multiple linear regression model of equation (5.1) assumes that the observations are not correlated with one another. While this is certainly believable if one has drawn a random sample, it's less likely if one has drawn observations sequentially in time. Time series observations, which are drawn at regular intervals, usually embody a structure where time is an important component. If you are unable to completely model this structure in the regression function itself, then the remainder spills over into the unobserved component of the statistical model (its error) and this causes the errors be correlated with one another.

One way to think about it is that the errors will be **serially correlated** when omitted effects last more than one time period. This means that when the effects of an economic 'shock' last more than a single time period, the unmodelled components (errors) will be correlated with one another. A natural consequence of this is that the more frequently a process is sampled (other things being equal), the more likely it is to be autocorrelated. From a practical standpoint, monthly observations are more likely to be autocorrelated than quarterly observations, and quarterly more likely than yearly ones. Once again, ignoring this correlation makes least squares inefficient at best and the usual measures of precision (standard errors) inconsistent.

In this chapter, several ways to detect autocorrelation in the model's errors are considered. Also, statistically valid ways of estimating the parameters of 8.1 and testing hypotheses about the β s in autocorrelated models are explored.

9.1 Area Response Model for Sugar Cane

Hill et al. (2007) considers a simple model of the area devoted to sugar cane production in

Bangladesh. The equation to be estimated is

$$\ln(A_t) = \beta_1 + \beta_2 \ln(P_t) + e_t \qquad t = 1, 2, \dots, N$$
(9.1)

The data consist of 34 annual observations on area (A) and price (P). The error term contains all of the economic factors other than price that affect the area of production. If changes in any of these other factors (shocks) affect area for more than one year, then the errors of the model will not be mutually independent of one another. The errors are said to be *serially correlated* or **autocorrelated**. Least square estimates of the β s are consistent, but the usual computation for the standard errors is not.

If the shock persists for two periods, and the shock is stable in the sense that its influence on the future diminishes as time passes, then we could use a model such as

$$e_t = \rho e_{t-1} + v_t \tag{9.2}$$

where ρ is a parameter and v_t is random error. This says that today's shock is in part due to the shock that happened in the previous period. Thus, there is some *persistence* in the area under tillage that is unrelated to price. The model referred to in equation 9.2 is called **first order** autocorrelation and is abbreviated **AR(1)**.

Stability means that the parameter ρ must lie in the (-1,1) interval (not including the endpoints). If $|\rho|$ is one or greater than the errors are not stable and a shock will send your model spiraling out of control!

As is the case with heteroskedastic errors, there is a way to salvage least squares when your data are autocorrelated. In this case you can use an estimator of standard errors that is robust to both heteroskedasticity and autocorrelation proposed by Newey and West. This estimator is sometimes called **HAC**, which stands for **heteroskedasticity autocorrelated consistent**. This and some issues that surround its use are discussed in the next few sections.

9.1.1 Bandwidth and Kernel

HAC is not quite as automatic as the heteroskedasticity consistent (HC) estimator in Chapter 8. To be robust with respect to autocorrelation you have to specify how far away in time the autocorrelation is likely to be significant. Essentially, the autocorrelated errors over the chosen time window are averaged in the computation of the HAC standard errors; you have to specify how many periods over which to average and how much weight to assign each residual in that average. The language of time series analysis can often be opaque. This is the case here. The weighted average is called a **kernel** and the number of errors to average in this respect is called **bandwidth**. Just think of the kernel as another name for weighted average and bandwidth as the term for number of terms to average.

Now, what this has to do with **gretl** is fairly simple. You get to pick a method of averaging (Bartlett kernel or Parzen kernel) and a bandwidth (nw1, nw2 or some integer). **Gretl** defaults to

the Bartlett kernel and the bandwidth $nw1 = 0.75xN^{1/3}$. As you can see, the bandwidth nw1 is computed based on the sample size, N. The nw2 bandwidth is $nw2 = 4 \times (N/100)^{2/9}$. This one appears to be the default in other programs like EViews.

Implicitly there is a trade-off to consider. Larger bandwidths reduce bias (good) as well as precision (bad). Smaller bandwidths exclude more relevant autocorrelations (and hence have more bias), but use more observations to increase precision (smaller variance). The general principle is to choose a bandwidth that is large enough to contain the largest autocorrelations. The choice will ultimately depend on the frequency of observation and the length of time it takes for your system to adjust to shocks.

The bandwidth or kernel can be changed using the **set** command from the console or in a script. The **set** command is used to change various defaults in **gretl** and the relevant switches for our use are **hac_lag** and **hac_kernel**. The use of these is demonstrated below. The following script could be used to change the kernel to **bartlett** and the bandwidth to **nw2**:

```
open "c:\Program Files\gretl\data\poe\bangla.gdt"
set hac_kernel bartlett
set hac_lag nw2
```

9.1.2 Dataset Structure

The other key to using HAC is that your data must be structured as a time series. This can be done through the dialogs or very simply using the console. First let's look at the **Dataset wizard** provided in the system of menus.

Open the bangla.gdt data and choose Data>Data set structure from the main pull-down menu (refer to Figure 7.2). This brings up the Data structure wizard dialog box (Figure 7.3). Choose Time series and click Forward. Although the data were collected annually, no actual dates (years) are provided. So in the next box (top of Figure 9.1) choose other and click Forward. This leads to the second box in the figure. Start the observations at 1, click Forward again, and a window confirming your choices (shown at the bottom) will open. If satisfied, click OK to close the wizard or Back to make changes. If you had chosen annual, quarterly, monthly or other actual time frame in the first of the wizard's boxes, then you would be given the opportunity to select actual dates in the second box. Again, your choices are confirmed in the final box generated by the wizard.

Gretl includes the **setobs** command that will do the same thing. For *bangla.gdt* dataset the command is

```
setobs 1 1 --time-series
```

The first number identifies the periodicity (1=year, 4=quarter, 12=month, and so on). The second

Figure 9.1: Choose Data>Dataset structure from the main window. This starts the Dataset wizard, a series of dialogs that allow you to specify the periodicity and dates associated with your data.

🎉 Data structur	e wizard	
Т	ime series frequenc	у
C Annual C Quarterly C Monthly C Weekly C Daily C Hourly C Hourly	Actual not pro these da	dates were ovided with ta so choose Other'
Other 1 ★ Cancel	Back	Forward
Starting obse	e wizard ervation:	
X Cancel	🗢 Back	Porward Eorward
	ll,	7
Data structur Co tir	e wizard nfirm dataset structo ne-series data, 1 to	×
X Cancel	🖨 Back	₽₽QK

number sets the starting date. Since there is no date for this data we start the time counter at 1. Finally, the **--time-series** option is used to declare the data to be time-series. Here are a few other examples:

setobs 4 1978:3 --time-series setobs 12 1950:01 --time-series setobs 1 1949 --time-series

The first statement starts a quarterly series in the third quarter of 1978, the second a monthly series beginning in January 1950, and the last a yearly series beginning in 1949. See the help on **setobs** to declare daily or hourly series, or to setup your data as a cross-section or panel.

9.1.3 HAC Standard Errors

Once **gretl** recognizes that your data are time series, then the robust command will automatically apply the HAC estimator of standard errors with the default values of the kernel and bandwidth (or the ones you have set with the **set** command). Thus, to obtain the HAC standard errors simply requires

open "c:\Program Files\gretl\data\poe\bangla.gdt"
logs p a
ols l_p const l_a --robust

The statement $\log p$ a creates the natural logarithms of the variables p and a and puts them into the dataset as l_p and l_a . These are used in the regression with the --robust option to produce least squares estimates with HAC standard errors.

The results appear below:

OLS estimates Dependent variable: l_a

	OLS (wrong SE)	OLS with HAC
const	3.893**	3.893**
	(0.06058)	(0.06135)
l_p	0.7761^{**}	0.7761^{**}
	(0.3669)	(0.2775)
n	34	34
\mathbb{R}^2	0.1965	0.1965
ℓ	-7.15	-7.15
Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level

Notice that the standard errors computed using HAC are a little different from those in Hill et al. (2007). No worries, though. They are statistically valid and suggest that EViews and **gretl** are doing the computations a bit differently.

9.2 Nonlinear Least Squares

Perhaps the best way to estimate a linear model that is autocorrelated is using nonlinear least squares. As it turns out, the nonlinear least squares estimator only requires that the errors be stable (not necessarily stationary). The other methods commonly used make stronger demands on the data, namely that the errors be covariance stationary. Furthermore, the nonlinear least squares estimator gives you an unconditional estimate of the autocorrelation parameter, ρ , and yields a simple t-test of the hypothesis of no serial correlation. Monte Carlo studies show that it performs well in small samples as well. So with all this going for it, why not use it?

The biggest reason is that nonlinear least squares requires more computational power than linear estimation, though this is not much of a constraint these days. Also, in **gretl** it requires an extra step on your part. You have to type in an equation for **gretl** to estimate. This is the way one works in EViews and other software by default, so the burden here is relatively low.

Nonlinear least squares (and other nonlinear estimators) use numerical methods rather than analytical ones to find the minimum of your sum of squared errors objective function. The routines that do this are iterative. You give the program a good first guess as to the value of the parameters and it evaluates the sum of squares function at this guess. The program looks at the slope of your sum of squares function at the guess and computes a *step* in the parameter space that takes you closer to a minimum (further down the hill). If an improvement in the sum of squared errors function is found, the new parameter values are used as the basis for another step. Iterations continue until no further significant reduction in the sum of squared errors function can be found.

In the context of the area response equation the AR(1) model is

$$\ln(A_t) = \beta_1(1-\rho) + \beta_2(\ln(P_t) - \rho\ln(P_{t-1})) + \rho\ln(A_{t-1}) + v_t$$
(9.3)

The errors, v_t , are random and the goal is to find β_1 , β_2 , and ρ that minimize $\sum v_t^2$. Ordinary least squares is a good place to start in this case. The OLS estimates are consistent so we'll start our numerical routine there, setting ρ equal to zero. The **gretl** script to do this follows:

```
open "c:\Program Files\gretl\data\poe\bangla.gdt"
logs p a
ols l_a const l_p --robust
```

```
genr beta1 = $coeff(const)
genr beta2 = $coeff(l_p)
genr rho = 0
nls l_a = beta1*(1-rho) + rho*l_a(-1) + beta2*(l_p-rho*l_p(-1))
end nls
```

Magically, this yields the same result from your text!

The nls command is initiated with nls followed by the equation representing the systematic portion of your model. The command is closed by the statement end nls. In the script, I used gretl's built in functions to take lags. Hence, l_a(-1) is the variable l_a lagged by one period (-1). In this way you can create lags or leads of various lengths in your gretl programs without explicitly having to create new variables via the generate command. The results of nonlinear least squares appear below in Figure 9.2.

Figure 9.2: Nonlinear least squares results for the AR(1) regression model.

🦉 gretl: script output						
🛛 🗛 🛍 🖻 🕹						
Generated scalar rl ? nls l_a = betal* ? end nls Using numerical den Convergence achieve Tolerance = 1.81899 Model 3: NLS estime Dependent variable	no (ID 7) = 0 (1-rho) + rho*1_a(rivatives ed after 37 iterat 9e-012 ates using the 33 : 1_a	<pre>-1) + beta2*(1_p ions observations 2-3</pre>	rho*l_p(-1))	1	•
PARAMETER	ESTIMATE	STDERROR	T STAT	P-VALUE		
beta1 rho beta2	3.89877 0.422139 0.888371	0.0921658 0.166047 0.259298	42.302 2.542 3.426	<0.00001 0.01641 0.00180	*** **	
Mean of dependent variable = 3.99931 Standard deviation of dep. var. = 0.325164 Sum of squared residuals = 2.44358 Standard error of residuals = 0.285399 Unadjusted R-squared = 0.277777 Akaike information criterion (AIC) = 13.7495 Schwarz Bayesian criterion (BIC) = 18.239 Hannan-Quinn criterion (HQC) = 15.26						▼
		Close				

Equation 9.3 can be expanded and rewritten in the following way:

$$\ln(A_t) = \beta_1(1-\rho) + \beta_2 \ln(P_t) - \beta_2 \rho \ln(P_{t-1}) + \rho \ln(A_{t-1}) + v_t$$
(9.4)

$$\ln(A_t) = \delta + \delta_0 \ln(P_t) - \delta_1 \ln(P_{t-1}) + \theta_1 \ln(A_{t-1}) + v_t$$
(9.5)

Both equations contain the same variables, but Equation (9.3) contains only 3 parameters while (9.5) has 4. This means that (9.3) is *nested* within (9.5) and a formal hypothesis test can be

performed to determine whether the implied restriction holds. The restriction is $\delta_1 = -\theta_1 \delta_0$.¹ To test this hypothesis using **gretl** you can use a variant of the statistic (6.2) discussed in section 6.1. You'll need the restricted and unrestricted sum of squared errors from the models. The statistic is

$$J \times F = \frac{(SSE_r - SSE_u)}{SSE_u/(N - K)} \dot{\sim} \chi_J^2 \qquad \text{if } H_0: \ \delta_1 = -\theta_1 \delta_0 \text{ is true}$$
(9.6)

Since J = 1 this statistic has an approximate χ_1^2 distribution and it is equivalent to an F test. Note, you will get a slightly different answer that the one listed in your text. However, rest assured that the statistic is asymptotically valid.

For the example, we've generated the output:

```
Replaced scalar fstat (ID 12) = 1.10547
? pvalue X 1 fstat
Chi-square(1): area to the right of 1.10547 = 0.293069
(to the left: 0.706931)
? pvalue F 1 $df fstat
F(1, 29): area to the right of 1.10547 = 0.301752
(to the left: 0.698248)
```

Because the sample is so small (only 29 degrees of freedom) the p-values from the F(1,29) and the χ_1^2 are a bit different. Still, neither is significant at the 5% level.

9.3 Testing for Autocorrelation

Two methods are used to determine the presence or extent of autocorrelation. The first is to take a look at the **residual correlogram**. A correlogram is a graph that plots series of correlations between \hat{x}_t and \hat{x}_{t-j} against the time interval between the observations, $j = 1, 2, \ldots, m$. A residual correlogram uses residuals from an estimated model as the time series, x_t . So, the first thing to do is to estimate the model using least squares and then save the residuals, \hat{e}_t . Once you have the residuals, then use the **corrgm** command to get the correlogram. The syntax follows:

open "c:\Program Files\gretl\data\poe\bangla.gdt" logs p a ols l_a const l_p --robust genr ehat = \$uhat corrgm ehat 12 $\overline{^{1}\delta = \beta_1(1-\rho), \delta_0 = \beta_2, \delta_1 = -\rho\beta_2, \theta_1 = \rho}$ The output is found in Figure 9.3 below. Essentially, the 12 autocorrelations plotted are simple



Figure 9.3: Correlogram of the least squares residuals

correlations between \hat{e}_t and \hat{e}_{t-m} for m = 1, 2, ..., 12. Statistical significance at the 5% level is denoted with two asterisks (**) and at the 10% level with one (*). The correlogram is just a way of visualizing this, as it plots each of the autocorrelations against its lag number.

The dialogs yields a much prettier and marginally more informative result. Estimate the model using Model>Ordinary Least Squares as shown in Figures 5.1 and 5.2. Click OK to run the regression and the results appear in a model window. Then select Graphs>Residual plot>Correlogram from the pull-down menus as shown in Figure 9.4. Select the number of lags to include using the dialog box (Figure 9.5). Click OK and gretl opens two windows containing results. The first contains the table shown at the bottom half of Figure 9.3, which shows the computed sample autocorrelations (ACF) and partial autocorrelations (PACF). The other is a graph of these along with 95% confidence bands. This graph is depicted in Figure 9.6 below. You can see that the first and fifth autocorrelations lie outside of the confidence band, indicating that they are individually

Figure 9.4: From the model window you can obtain the correlogram of the least squares residuals with Graph>Residual plot>Correlogram.

🖉 gretl: model 6					
<u>File Edit T</u> ests Save <mark>Graphs</mark> <u>A</u> nalysis LaTeX					
Residual plot Correlog Model 6: OLS esti Eitted, actual plot Spectrum Dependent variable: 1_a Against VARIABLE COEFFICIENT Against const 3.89326 State	ne a p RROR T STAT P-VALUE 0.0613451 63.465 <0.00001 ***				
const 3.89326 0.0613451 63.465 <0.00001 ***					
	ose				

Figure 9.5: Choose the desired number of lags using the dialog box.

🦉 gretl: corre		
Maximum lag:	12	
🔯 Help	🗶 🤇 ancel	₽дк



Figure 9.6: This version of the correlogram is much prettier and includes confidence bands for the autocorrelations.

significant at the 5% level.

The other way to determine whether or not your residuals are autocorrelated is to use an LM (Lagrange multiplier) test. For autocorrelation, this test is based on an auxiliary regression where lagged least squares residuals are added to the original regression equation. If the coefficient on the lagged residual is significant then you conclude that the model is autocorrelated. So, for a regression model $y_t = \beta_1 + \beta_2 x_t + e_t$ the first step is to estimate the parameters using least squares and save the residuals, \hat{e}_t . An auxiliary regression model is formed using \hat{e}_t as the dependent variable and original regressors and the lagged value \hat{e}_{t-1} as an independent variables. The resulting auxiliary regression is

$$\hat{e}_t = \beta_1 + \beta_2 x_t + \rho \hat{e}_{t-1} + v_t \tag{9.7}$$

Now, test the hypothesis $\rho = 0$ against the alternative that $\rho \neq 0$ and you are done. The test statistic is NR^2 from this regression which will have a χ_1^2 if Ho: is true. The script to accomplish this is:

ols ehat const l_p ehat(-1) scalar NR2 = \$trsq pvalue X 1 NR2

If you prefer to use the dialogs, then estimate the model using least square in the usual way (Model>Ordinary least squares) and select Tests>Autocorrelation from the resulting model

window (i.e., the one in Figure 9.4). Choose the number of lagged values of \hat{e}_t you want to include in (9.7) (in our case only 1) and click OK. This will give you the same result as the script. The result appears in Figure 9.7. Note, the first statistic reported is simply the joint test that all the lagged values of \hat{e} you included in (9.7) are jointly zeros. The second one is the NR^2 version of the test done in the script. **Gretl** also computes a Ljung-Box Q statistic whose null hypothesis is no autocorrelation. It is also insignificant at the 5% level.

Figure 9.7: Using Test>Autocorrelation from the model pull-down menu will generate the following output.

```
🖉 gretl: LM test (autocorrelation)
                                                                               - 🗆 ×
 🔏 📇 🖻 🖄 🗙
 Breusch-Godfrey test for first-order autocorrelation
 OLS estimates using the 33 observations 2-34
 Dependent variable: uhat
      VARIABLE
                      COEFFICIENT
                                         STDERROR
                                                        T STAT
                                                                 P-VALUE
                         0.0132921
                                          0.0608353
   const
                                                        0.218
                                                                 0.82852
   1_p
                        -0.0151610
                                          0.280749
                                                        -0.054
                                                                 0.95729
                         0.398553
                                          0.167309
                                                                0.02375 **
   uhat 1
                                                        2.382
   Unadjusted R-squared = 0.163295
 Test statistic: LMF = 5.854922,
 with p-value = P(F(1, 30) > 5.85492) = 0.0218
 Alternative statistic: TR^2 = 5.388728.
 with p-value = P(Chi-square(1) > 5.38873) = 0.0203
 Ljung-Box O' = 5.78134 with p-value = P(Chi-square(1) > 5.78134) = 0.0162
                                        Close
```

9.4 Autoregressive Models and Forecasting

A autoregressive model will include one or more lags of your dependent variable on the righthand-side of your regression equation. An AR(p) includes p lags of y_t as shown below in equation (9.8).

$$y_t = \delta + \theta_1 y_{t-1} + \theta_2 y_{t-2} + \dots + \theta_p y_{t-p} + v_t \tag{9.8}$$

In general, p should be large enough so that v_t is white noise.

The dataset *inflation.gdt* includes 270 monthly observations on the CPI from which an inflation variable is computed. To estimate an AR(3) model of inflation, simply use the script

```
open "c:\Program Files\gretl\data\poe\inflation.gdt"
ols infln const infln(-1 to -3)
```

In this case a bit of shorthand is used to generate the lagged values of inflation to include as regressors. The syntax infln(-1 to -3) tells gretl to include a range of the variable inflation from lags from 1 to 3. The minus signs indicate lags. This is equivalent to using a list of variables as in

```
ols infln const infln(-1) infln(-2) infln(-3)
```

Obviously, if p were large then using the range version would save a lot of typing.

Using this model to forecast in **gretl** is very simple. The main decision you have to make at this point is how many periods into the future you want to forecast. In **gretl** you have to extend the sample to include future periods under study.

9.4.1 Using the Dialogs

Return to the main **gretl** window and choose Model>Ordinary least squares. This will bring up the 'specify model' dialog box. Choose infln as the dependent variable as shown.

Since your data are defined as time series (recall, you did this through Data>Dataset structure) an extra button, labeled 'lags...', appears at the bottom of the dialog as highlighted in Figure 9.8. Click the 'lags...' button in the specify model dialog box and the 'lag order' dialog box shown in Figure 9.9 opens. Click OK and the the 3 lagged values of inflation are added to the model. Now, click OK in the specify model dialog as in Figure 9.8. The model is estimated and the model window shown in Figure 9.10 opens.

Now, we'll use the dialogs to extend the sample and generate the forecasts. From the model window choose Analysis>Forecasts. This opens the Add observations dialog box shown in Figure 9.11. To add three observations change the number in the box to 3. Click OK to open the forecast dialog box shown below in Figure 9.12.

By choosing to add 3 observations to the sample, the forecast range is automatically set to 2006:06 to 2006:08. Notice that we've chosen 'automatic forecast (dynamic out of sample).' Click OK and the forecast results appear:

```
For 95% confidence intervals, t(262, .025) = 1.969
    Obs infln Forecast SE 95% C.I.
1998:02 0.000000 0.23350
    .
.
```

Figure 9.8: From the main window select Model>Ordinary least squares. This brings up the specify model dialog box that includes a button for adding lags of the variables to your model

📓 gretl: specify model		<u>- 0 ×</u>
	OLS	
const	Dependent variable	
month	Choose -> infln	
pcwage	Set as default	
cpi		
Infin	Independent variables	
	const	
	Add ->	
	C. Bemoue	
Robust standard errors	configure	
	ear 🛛 💥 Cancel 🛛 🎸	7_ок

Figure 9.9: Check the box labeled 'Lags of dependent variable' and change the second counter to '3' as shown.

🌉 lag order			
Variable	lags	or	specific lags
🔽 Lags of d	lependent variable		
infln 1	🖨 to 🕻		
			·
🔯 Help	× 1	Cancel	⊘₽ок

Figure 9.10: Choose Analysis>Forecasts from the estimated forecast model to open the forecast dialog box.

👰 gretl: model 1							_ 🗆 ×
Eile Edit Tests Save Grap	ns <u>A</u> nalysis <u>L</u> aTeX						
	Display actual, fitted, res	idual					
Model 1: OLS estimate	Eorecasts		ş 198	34:04-200	6:05		
Dependent variable: i	nf ⊆onfidence intervals for (coefficients 🏻	0				
	Confidence ellipse						
VARIABLE	CO Coefficient covariance m	atrix		T STAT	P-VALUE		
gonat	ANOVA		05	7 447	<0.00001	* * *	
infln 1	Bootstrap		03 08	6 072	<0.00001	***	
infln 2	-0 217919	0.00140	25	-3 380	0.00001	* * *	
infln 3	0.101254	0.06126	80	1.653	0.09960	*	
<pre>infln_2 -0.217919 0.0644725 -3.380 0.00084 *** infln_3 0.101254 0.0612680 1.653 0.09960 * Mean of dependent variable = 0.253389 Standard deviation of dep. var. = 0.210185 Sum of squared residuals = 10.1934 Standard error of residuals = 0.197247 Unadjusted R-squared = 0.129295 Adjusted R-squared = 0.119325 F-statistic (3, 262) = 12.9685 (p-value < 0.00001) Durbin-Watson statistic = 2.00025 First-order autocorrelation coeff. = -0.00147561 Log-likelihood = 56.3753 Akaike information criterion (AIC) = -104.751 Schwarz Bayesian criterion (HQC) = -98.992</pre>							
		Close					

Figure 9.11: Using Data>Add observations from the main gretl pull-down menu will extend the sample period. This is necessary to generate forecasts.



Figure 9.12: Forecast dialog box



2006:03	0.350966	0.05914		
2006:04	0.598804	0.37476		
2006:05	0.446762	0.34046		
2006:06		0.26015	0.19724	-0.12823 - 0.64854
2006:07		0.24872	0.21054	-0.16584 - 0.66329
2006:08		0.26972	0.21111	-0.14596 - 0.68541

Miraculously, these match those in *POE*! Gretl also uses gnuplot to plot the time series and the forecasts (with intervals) as shown in Figure 9.13. The last three observations are forecasts (in





blue) and include the 95% confidence intervals shown in green. Actual inflation appears in red.

9.4.2 Using a Script

Doing all of this using a script is easy as well. Simply estimate the model using ols infln const infln(-1 to -3), use the addobs 3 command to add 3 observations to the end of the sample, and forecast 3 periods using fcasterr 2006:06 2006:08. The --plot option ensures that the graph will be produced. The script is:

```
open "c:\Program Files\gretl\data\poe\inflation.gdt"
ols infln const infln(-1 to -3)
```

addobs 3 fcasterr 2006:06 2006:08 --plot

To estimate the distributed lag model of inflation

$$infl_t = \alpha + \beta_0 x_t + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \beta_3 x_{t-3} + e_t \tag{9.9}$$

where x_t is the percentage change in wages at time t. The script is:

```
open "c:\Program Files\gretl\data\poe\inflation.gdt"
```

```
ols infln const pcwage(0 to -3)
scalar in1 = $coeff(pcwage)+$coeff(pcwage_1)
scalar in2 = in1 + $coeff(pcwage_2)
scalar in3 = in2 + $coeff(pcwage_3)
```

Here, the independent variable is pcwage, which is already in the dataset. To add the contemporaneous (lag=0) and 3 lagged values to the list of independent variables, simply add pcwage(0 to -3) as shown. The **delay multipliers** are just the coefficients of the corresponding lagged variables. The **interim multiplier** is obtained by cumulatively adding the coefficients together. For instance the interim multiplier at lag 1 is equal to the sum of the delay multipliers (e.g., interim multiplier at lag 1 is $(\beta_0 + \beta_1)$. When using the range version (e.g., pcwage(0 to -3)) of the language to generate lags, gretl appends an underline _ and the corresponding lag number to the variable. So, $pcwage_{t-1}$ is referred to as pcwage_1.

9.5 Autoregressive Distributed Lag Model

This model is just a generalization of the ones previously discussed. In this model you include lags of the dependent variable (autoregressive) **and** the contemporaneous and lagged values of independent variables as regressors (distributed lags). The shorthand notation is ARDL(p,q) where p is the maximum distributed lag and q is the maximum autoregressive lag. The model is

$$y_t = \delta + \delta_0 x_t + \delta_1 x_{t-1} + \dots + \delta_q x_{t-q} + \theta_1 y_{t-1} + \dots + \theta_p y_{t-p} + v_t$$
(9.10)

The ARDL(3,2) model of inflation includes the contemporaneous and first 3 lagged values of *pcwage* and the first 2 lags of *infl* as independent variables.²

The script is

```
open "c:\Program Files\gretl\data\poe\inflation.gdt"
ols infln const pcwage(0 to -3) infln(-1 to -2)
```

 $^{^{2}}$ Technically, lagged values of inflation are *predetermined* not independent, but we'll leave this discussion for others. Their treatment in a regression is the same, though.

The result appears in Figure 9.14.

Figure 9.14: Results of the autoregressive distributed lag model produced by the script.

greti: script output						
2 4 6 8 9	> 🥎 🗙					
gretl version 1.6.	2					
Current session: 2	007/05/08 11:45					
? ols infln const	pewage(0 to -3) in:	fln(-1 to -2)				
Model 4: OLS estim	ates using the 266	observations 19	84:04-200	6:05		
ependent variable)	: infin					
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE		
const	0.0988766	0.0468069	2.112	0.03561	**	
pcwage	0.114903	0.0833901	1.378	0.16942		
pcwage 1	0.0377337	0.0812455	0.464	0.64272		
pcwage_2	0.0592746	0.0811736	0.730	0.46592		
pcwage_3	0.236130	0.0829441	2.847	0.00477	* * *	
infln_1	0.353640	0.0604112	5.854	<0.00001	* * *	
infln_2	-0.197561	0.0604212	-3.270	0.00122	* * *	
Mean of dependen	t variable = 0.253:	389				
Standard deviati	on of dep. var. =)	0.210185				
Sum of squared r	esiduals = 9.74826					
Standard error o	f residuals = 0.19	1005				
Unadjusted R-squ	ared = 0.167322					
Adjusted R-squar	ed = 0.148032					
F-statistic (6,	259) = 8.67409 (p-	value < 0.00001)				
Durbin-Watson st	atistic = 1.93031					
First-order auto	correlation coeff.	= 0.0343753				
Durbin's h stat.	3.08607					
(Using variable 6 for h stat, with T' = 265)						
Log-likelihood =	62.3145					
Akaike informati	on criterion (AIC)	= -110.629				
Schwarz Bayesian	criterion (BIC) =	-85.5446				
Hannan-Quinn cri	terion (HQC) = -10	0.552				
xcluding the cons	tant, p-value was :	highest for vari	able 9 (p	cwage_1)		
		Close				

9.6 Script

```
open "c:\Program Files\gretl\data\poe\bangla.gdt"
#declare the data time-series
setobs 1 1 --time-series
#Least squares with wrong std errors
logs p a
ols l_a const l_p
#Least squares with HAC standard errors
#choose lag
set hac_lag nw2
#choose weights
set hac_kernel bartlett
#run regression with robust std errors
ols l_a const l_p --robust
#Nonlinear least squares
```

```
#step 1: set the starting values
genr beta1 = $coeff(const)
genr beta2 = $coeff(1_p)
genr rho = 0
#step 2: type in the model
nls l_a = beta1*(1-rho) + rho*l_a(-1) + beta2*(l_p-rho*l_p(-1))
end nls
#save restricted sum of squared errors for the hypothesis test
scalar sser=$ess
#get the unrestricted sse
ols l_a const l_p l_p(-1) l_a(-1)
scalar sseu=$ess
scalar fstat = (sser-sseu)/(sseu/$df)
pvalue X 1 fstat
pvalue F 1 $df fstat
#Correlogram
ols l_a const l_p --robust
genr ehat = $uhat
corrgm ehat 12
#LM test
ols l_a const l_p
genr ehat = $uhat
ols ehat const l_p ehat(-1)
scalar NR2 = $trsq
#Dynamic forecasting in an autoregressive model
open "c:\Program Files\gretl\data\poe\inflation.gdt"
ols infln const infln(-1 to -3)
addobs 3
fcasterr 2006:06 2006:08--plot
#Distributed Lag model and interim multipliers
ols infln const pcwage(0 to -3)
scalar in1 = $coeff(pcwage)+$coeff(pcwage_1)
scalar in2 = in1 + $coeff(pcwage_2)
scalar in3 = in2 + $coeff(pcwage_3)
#ARDL(3,2)
ols infln const pcwage(0 to -3) infln(-1 to -2)
```

```
#First 5 lag weights for infinite distributed lag
scalar b0 = $coeff(pcwage)
scalar b1 = $coeff(infln_1)*b0+$coeff(pcwage_1)
scalar b2 = $coeff(infln_1)*b1+$coeff(infln_2)*b0+$coeff(pcwage_2)
scalar b3 = $coeff(infln_1)*b2+$coeff(infln_2)*b1+$coeff(pcwage_3)
scalar b4 = $coeff(infln_1)*b3+$coeff(infln_2)*b2
```

Chapter 10

Random Regressors and Moment Based Estimation

In this chapter you will learn to use instrumental variables to obtain consistent estimates of a model's parameters when its independent variables are correlated with the model's errors.

10.1 Basic Model

Consider the linear regression model

$$y_i = \beta_1 + \beta_2 x_i + e_i \quad i = 1, 2, \dots, N \tag{10.1}$$

Equation (10.1) suffers from a significant violation of the usual model assumptions when its explanatory variable is contemporaneously correlated with the random error, i.e., $Cov(e_i, x_i) = E(e_i x_i) \neq 0$. In this instance, least squares is known to be both biased and inconsistent.

An **instrument** is a variable, z, that is correlated with x but not with the error, e. In addition, the instrument does not directly affect y and thus does not belong in the actual model. It is common to have more than one instrument for x. All that is required is that these instruments, z_1, z_2, \ldots, z_s , be correlated with x, but not with e. Consistent estimation of (10.1) is possible if one uses the **instrumental variables** or **two-stage least squares estimator**, rather than the usual OLS estimator.

10.2 IV Estimation

Gretl handles this estimation problem with ease using what is commonly referred to as *two-stage least squares*. In econometrics, the terms two-stage least squares (TSLS) and instrumental variables (IV) estimation are often used interchangeably. The 'two-stage' terminology is a legacy of the time when the easiest way to estimate the model was to actually use two separate least squares regressions. With better software, the computation is done in a single step to ensure the other model statistics are computed correctly. Since the software you use invariably expects you to specify 'instruments,' it is probably better to think about this estimator in those terms from the beginning. Keep in mind though that **gretl** uses the old-style term *two-stage least squares* (tsls) as it asks you to specify instruments in it dialog boxes and scripts.

To perform TSLS or IV estimation you need instruments that are correlated with your independent variables, but not correlated with the errors of your model. First, load the *ch10.gdt* data into gret1. Then, to open the basic gret1 dialog box that computes the IV estimator choose Model>Instrumental Variables>Two-Stage Least Squares from the pull-down menu as shown below in Figure 10.1. This opens the dialog box shown in Figure 10.2.

🚺 gre	🕅 gretl 📃 🗆 💟								
Eile	Tools	<u>D</u> ata <u>V</u> ie	w <u>A</u> dd	<u>S</u> ample	<u>V</u> ariable	Model	Help		
ch10.	gdt						dinary Least Squares		L
ID#	◀ Variat	ole name 🖪	Descrip	tive label		Ins	strumental variables	•	Two-Stage Least Squares
0	const	:	auto-ge	enerated o	onstant	0	her linear models	•	LIML
1	x					No	nlinear models	•	<u>G</u> MM
2	У					Iin	ne series		
3	z1					Ea Pa	nel	1	
4	z2					<u>R</u> o	bust estimation		
5	z3					<u>M</u> a	aximum likelihood		
							1191		
						20	nuicaneous equations		1
						Undated	: Full range 1 - 100		
	2	fx	1	3 🗠 /	ŝ 📄				

Figure 10.1: Two-Stage Least Squares estimator from the pull-down menus

In this example we choose y as the dependent variable, put all of the desired instruments into the *Instruments* box, and put all of the independent variables, including the one(s) measured with error, into the *Independent Variables* box. If some of the right-hand side variables for the model are exogenous, they should be referenced in both lists. That's why the const variable (for the constant) appears in both places. Press the OK button and the results are found in Table 10.1. Notice that gretl ignores the sound advice offered by the authors of your textbook and computes

📓 gretl: specify model					
Тwo	-stage least squares				
const x y z1 z2	Dependent variable Choose -> y Set as default				
z3	Add -> <- Remove				
	Instruments Const z1 <- Remove				
Robust standard errors configure					
Help Clear Cancel OK					

Figure 10.2: Two-Stage Least Squares dialog box

Table 10.1: Model 1: TSLS, using observations 1–100 Dependent variable: y Instrumented: x Instruments: const z1 Coefficient Std. Error p-value z-stat 1.10110 0.10912810.0900 0.0000 const 1.192450.1945186.13020.0000 х Mean dependent var 1.386287S.D. dependent var 1.838819 Sum squared resid S.E. of regression 95.49855 0.987155 \mathbb{R}^2 Adjusted R^2 0.7853850.783195F(1, 98)P-value(F)37.57988 1.84e-08Log-likelihood Akaike criterion -507.21241018.425 Schwarz criterion 1023.635 Hannan–Quinn 1020.534 Hausman test -Null hypothesis: OLS estimates are consistent Asymptotic test statistic: $\chi^2(1) = 15.0454$ with p-value = 0.000104958Weak instrument test –

First-stage F(1, 98) = 38.9197

an R^2 . Keep in mind, though, gretl computes this as the squared correlation between observed and fitted values of the dependent variable, and you should resist the temptation to interpret this in the usual manner.

If you prefer to use a script, the syntax is very simple. The script for the example above is

open "c:\Program Files\gretl\data\poe\ch10.gdt"
tsls y const x;const z1

The gretl command tsls calls for the IV estimator to be used and it is followed by the linear model you wish to estimate. List the dependent variable (y) first, followed by the independent variables (const x). A semicolon separates the model to be estimated from the list of instruments (const z1). Notice that the constant is listed again as an instrument; once again, this is because it is exogenous with respect to the errors of the model and all exogenous variables should be listed in both places.

10.3 Specification Tests

There are three specification tests you will find useful with instrumental variables estimation. By default, **Gretl** computes each of these whenever you estimate a model using two-stage least squares. Below I'll walk you through doing it manually and we'll compare the manual results to the automatically generated ones.

10.3.1 Hausman Test

The first test is to determine whether the independent variable(s) in your model is (are) in fact uncorrelated with the model's errors. If so, then least squares is more efficient than the IV estimator. If not, least squares is inconsistent and you should use the less efficient, but consistent, instrumental variable estimator. The null and alternative hypotheses are H_o : $Cov(x_i, e_i) = 0$ against H_a : $Cov(x_i, e_i) \neq 0$. The first step is to use least squares to estimate

$$x_i = \gamma_1 + \theta_1 z_{i1} + \theta_2 z_{i2} + \nu_i \tag{10.2}$$

and to save the residuals, $\hat{\nu}_i$. Then, add the residuals to the original model

$$y_i = \beta_1 + \beta_2 x_i + \delta \hat{\nu}_i + e_i \tag{10.3}$$

Estimate this equation using least squares and use the t-ratio on the coefficient δ to test the hypothesis. If it is significantly different from zero then the regressor, x_i is not exogenous or predetermined with respect to e_i and you should use the IV estimator (TSLS) to estimate β_1 and β_2 . If it is not significant, then use the more efficient estimator, OLS.

The **gretl** script for the Hausman test is:

```
open "c:\Program Files\gretl\data\poe\ch10.gdt"
ols x const z1 z2
genr uhat1 = $uhat
ols y const x uhat1
```

You may have noticed that whenever you use two-stage least squares in **gretl** that the program automatically produces the test statistic for the Hausman test. There are several different ways of computing this statistic so don't be surprised if it differs from the one you compute manually using the above script.

10.3.2 Testing for Weak Instruments

To test for weak instruments, regress each independent variable suspected of being contemporaneously correlated with the error (x) onto all of the instruments (z's). If the overall F statistic in this regression¹ is less than 10, then you conclude that the instruments are weak. If it is greater than 10, you conclude that the instruments are strong enough. The following script uses least squares to perform three such tests. The first regression assumes there is only one instrument, z_1 ; the second that the single instrument is z_2 ; the third assumes both are instruments.

```
open "c:\Program Files\gretl\data\poe\ch10.gdt"
ols x const z1
omit z1 --quiet
ols x const z2 --quiet
omit z2 --quiet
ols x const z1 z2 --quiet
omit z1 z2 --quiet
```

When omit follows an OLS regression (e.g., ols x const z1 z2), gretl estimates a restricted model where the variables listed after it are omitted from the model above. It then performs a joint hypothesis test that the coefficients of the omitted variables are zero against the alternative that one or more are not zero. The --quiet option reduces the amount of output you have to wade through by suppressing the regressions; only the test results are printed. The output from gretl appears in Figure 10.3 below: Notice that the t-ratio on z1 is equal to 0.571088/0.0915416 = 6.23856 and the F(1,98) statistic associated with the same null hypothesis (i.e., that the coefficient on z1 is zero) is 38.9197. In fact there is an exact relationship between these numbers since $t_{n-k}^2 = F_{1,n-k}$. This is easily verified here by computing $6.239^2 \doteq 38.9197$.² Since the F value is well beyond 10, we can reject the hypothesis that the instrument z1 is weak in favor of the alternative that it is strong enough to be useful.

The second pair of statements in the script assume that z^2 is the single available instrument and the omit statement is again used to elicit the F statistic.

In the last regression, we use both instruments and the **omit** statement in **gretl** to perform the joint test that the instruments are jointly weak.

Gretl proves its worth here. Whenever you estimate a model using two stage least squares, gretl will compute the test statistic for the weak instruments test.

10.3.3 Sargan Test

The final test is the Sargan test of the overidentifying restrictions implied by an overidentified model. Recall that to be overidentified just means that you have more instruments than you have endogenous regressors. In our example we have a single endogenous regressor (x) and three instruments (z1, z2 and z3). The first step is to estimate the model using TSLS using all the instruments. Save the residuals and then regress these on the instruments alone. NR^2 from this

¹Recall that the null hypothesis for the overall F statistic is that all slopes are zero.

²The small discrepancy you will find if you try the calculation occurs because of rounding.



Figure 10.3: Results from using the omit statement after least squares

regression is approximately χ^2 with the number of surplus instruments as your degrees of freedom. **Gretl** does this easily since it saves TR^2 as a part of the usual regression output, where T is the sample size (which we are calling N). The script for the Sargan test follows:

```
open "c:\Program Files\gretl\data\poe\ch10.gdt"
list inst = const z1 z2 z3
tsls y const x; inst
genr uhat2 = $uhat
ols uhat2 inst
genr test = $trsq
pvalue X 3 test
```

This script uses a convenient way to accumulate variables into a set using the list command. The command list inst = const z1 z2 z3 puts the variables contained in const, z1, z2, and z3 into a set called inst. Once defined, the set of variables can be referred to as inst rather than listing them individually as we've done up to this point. In the script above, inst is used in the third line to list the instruments for tsls and again in the fifth line to include these variables in the ols regression.

Rejection of the null hypothesis implies that one or more of the overidentifying restrictions are not valid; you've chosen an inappropriate instrument. If the test statistic is insignificant, then your set of instruments passes muster. Whenever you have extra instruments (the model is overidentified), **gretl** will compute and print out the results from the Sargan test automatically. Unlike the Hausman test, these results should match those you compute manually using the script.

```
Sargan over-identification test -
Null hypothesis: all instruments are valid
Test statistic: LM = 13.1107
with p-value = P(Chi-Square(2) > 13.1107) = 0.00142246
```

10.4 Wages Example

The following script uses the results above to quickly reproduce the results from the wages example in your text. Open the data, restrict the sample to those working (wage > 0), and generate logarithm of wages and square experience.

```
open "c:\Program Files\gretl\data\poe\mroz.gdt"
```

restrict your sample to include only positive values for wage smpl wage > 0 --restrict

```
# generate ln(wage) and experience squared
genr lwage = log(wage)
genr expersq = exper*exper
```

The next thing we'll do is to create lists that contain regressors and instruments. These will simplify the program and help us to avoid having to use the continuation command for long lines of code.

```
# create lists of variables to include in each regression
# Regressors in x
list x = const educ exper expersq
# Instrument sets in z1 and z2
list z1 = const exper expersq mothereduc
list z2 = const exper expersq mothereduc fathereduc
```

The first list command puts the regressors const, educ, exper, and expersq into a set called x. The first set of instruments includes all of the exogenous variables in the list of regressors and adds mothereduc; it is called z1. The second set, called z2, adds fathereduc to the list of instruments.

Now, estimate the model using least squares. Notice that the list of regressors has been replaced by the list we created above. If education is endogenous in this regression, then least squares is inconsistent and should not be used.

least squares regression of wage equation
ols lwage x

This produces:

$$\begin{aligned} \hat{\text{lwage}} &= -\begin{array}{c} 0.522 \\ (0.19863) \end{array} + \begin{array}{c} 0.107 \\ (0.0141) \end{array} \text{educ} + \begin{array}{c} 0.0416 \\ (0.0132) \end{array} \text{exper} - \begin{array}{c} 0.000811 \\ (0.00039) \end{array} \text{experson} \\ T &= 428 \quad \bar{R}^2 = 0.1509 \quad F(3, 424) = 26.286 \quad \hat{\sigma} = 0.66642 \\ \text{(standard errors in parentheses)} \end{aligned}$$

Estimate the reduced form equation that uses mother's education as the sole instrument along with the other exogenous variables in the model; all of these were collected into z1.

least squares regression of the reduced form
ols educ z1

This produces:

educ = 9.775 + 0.04886 exper - 0.001281 expersq + 0.268 mothereduc
(0.424) (0.0417) (0.00124) (0.00124) (0.031)

$$T = 428 \quad \bar{R}^2 = 0.1467 \quad F(3, 424) = 25.47 \quad \hat{\sigma} = 2.1111$$
(standard errors in parentheses)

Estimate the model using the instrumental variable estimator. The instrumental variable estimators will be consistent if education is endogenous or not. It is not efficient. In the first instance below, only mother's education is used as an instrument and in the second both mother's and father's education are used.

tsls regression using 1 instrument (mother's education)
tsls lwage x ; z1
tsls using 2 instruments (mother's and father's education)
tsls lwage x ; z2

The TSLS results for the regression with one instrument is:

 $\widehat{\text{lwage}} = \underbrace{0.1982}_{(0.473)} + \underbrace{0.04926}_{(0.0374)} \text{educ} + \underbrace{0.04486}_{(0.0136)} \text{exper} - \underbrace{0.0009221}_{(0.000406)} \text{expersq}$ $T = 428 \quad \overline{R}^2 = 0.1293 \quad F(3, 424) = 22.137 \quad \widehat{\sigma} = 0.6796$ (standard errors in parentheses)

and that for the model with two instruments is:

Model 3: TSLS, using observations 1–428 Dependent variable: lwage Instrumented: educ Instruments: const exper experse mothereduc fathereduc

	Coefficient	Std. Error	z-stat	p-value
const	0.0481003	0.400328	0.1202	0.9044
educ	0.0613966	0.0314367	1.9530	0.0508
exper	0.0441704	0.0134325	3.2883	0.0010
expersq	-0.000898970	0.000401686	-2.2380	0.0252

Mean dependent var	1.190173	S.D. dependent var	0.723198
Sum squared resid	193.0200	S.E. of regression	0.674712
R^2	0.145660	Adjusted R^2	0.139615
F(3, 424)	8.140709	$\operatorname{P-value}(F)$	0.000028

Hausman test -

Null hypothesis: OLS estimates are consistent Asymptotic test statistic: $\chi^2(1) = 2.8256$ with p-value = 0.0927721

Sargan over-identification test –

Null hypothesis: all instruments are valid Test statistic: LM = 0.378071 with p-value = $P(\chi^2(1) > 0.378071) = 0.538637$ Weak instrument test – First-stage F(2, 423) = 55.4003

A Hausman test statistic is manually computed to test the validity of the instruments. The least squares residuals from the reduced form equation are regressed on all exogenous and instrumental variables. The residuals are saved and added to the original structural equation. Test the significance of the residuals coefficient using a t-test, or as we've done here, the equivalent F(1,N-K) test using the omit statement.

```
# Hausman test (check the t-ratio on ehat for significance)
ols educ z2 --quiet
genr ehat=$uhat
ols lwage x ehat --quiet
omit ehat --quiet
```

This produces:

Test statistic: F(1, 423) = 2.7926, with p-value = 0.09544

which is very close the automatic result produced by **gretl** as part of the tsls output.

To test the strength of the instruments we estimate the reduced form equation for education and conduct a joint significance test of the two instruments (mothereduc and fathereduc). Once again, the --quiet option is used to suppress unnecessary output.

```
# test for strength of instruments (coeffs on instruments
# are jointly zero)
ols educ z2 --quiet
omit mothereduc fathereduc --quiet
```

Finally the test of overidentification is done. This requires residuals from the instrumental variable estimator, TSLS. Estimate the model using TSLS and save the residuals. In the second regression, which is estimated using least squares, these residuals are regressed on all exogenous and instrumental variables. NR^2 from this regression is compared to the $\chi^2(2)$ distribution. If the p-value is smaller than the desired α then at least one of the instruments is not appropriate. You'll need to either drop the offending ones or find others to use.

```
# (Sargan's test)
# requires residuals from tsls to use in this test
```

```
tsls lwage x; z2
genr vhat=$uhat
ols vhat z2
genr lmstat = $trsq
pvalue X 2 lmstat
```

The script will produce the same results you get from gretl's tsls command.

10.5 Script

```
open "c:\Program Files\gretl\data\poe\ch10.gdt"
tsls y const x; const z1
#Hausman test
ols x const z1 z2
genr uhat1 = $uhat
ols y const x uhat1
#Testing for weak instruments
open "c:\Program Files\gretl\data\poe\ch10.gdt"
ols x const z1
omit z1 --quiet
ols x const z2 --quiet
omit z2 --quiet
ols x const z1 z2 --quiet
omit z1 z2 --quiet
#Sargan Test
open "c:\Program Files\gretl\data\poe\ch10.gdt"
list inst = const z1 z2 z3
tsls y const x; inst
genr uhat2 = $uhat
ols uhat2 inst
genr test = $trsq
pvalue X 3 test
#Wages Example
open "c:\Program Files\gretl\data\poe\mroz.gdt"
# restrict your sample to include only positive values for wage
smpl wage > 0 --restrict
```

generate ln(wage) and experience squared genr lwage = log(wage) genr expersq = exper*exper # create lists of variables to include in each regression # Regressors in x list x = const educ exper expersq # Instrument sets in z1 and z2 list z1 = const exper expersq mothereduc list z2 = const exper experse mothereduc fathereduc # least squares regression of wage equation ols lwage x # least squares regression of the reduced form ols educ z1 # tsls regression using 1 instrument (mother's education) tsls lwage x; z1 # tsls using 2 instruments (mother's and father's education) tsls lwage x ; z2 genr vhat = \$uhat # Hausman test (check the t-ratio on ehat for significance) ols educ z2 --quiet genr ehat=\$uhat ols lwage x ehat --quiet omit ehat --quiet # test for strength of instruments (coeffs on instruments # are jointly zero) ols educ z2 --quiet omit mothereduc fathereduc --quiet #Repeat using HCCME ols educ z2 --robust --quiet omit mothereduc fathereduc --quiet # test for validity of instruments using residuals from tsls # (Sargan's test) ols vhat z2 --quiet genr lmstat = \$trsq pvalue X 2 lmstat

Chapter 11

Simultaneous Equations Models

In Chapter 11 of *POE* the authors present a model of supply and demand. The econometric model contains two equations and two dependent variables. The distinguishing factor for models of this type is that the values of two (or more) of the variables are jointly determined. This means that a change in one of the variables causes the other to change and vice versa. The model is demonstrated using the truffle example which is explained below.

11.1 Truffle Example

Consider a supply and demand model for truffles:

$$Q_i = \alpha_1 + \alpha_2 P_i + \alpha_3 P S_i + \alpha_4 D I_i + e_i^d \tag{11.1}$$

$$Q_i = \beta_1 + \beta_2 P_i + \beta_3 P F_i + e_i^s \tag{11.2}$$

The first equation (11.1) is demand and Q us the quantity of truffles traded in a particular French market, P is the market price of truffles, PS is the market price of a substitute good, and DI is per capita disposable income of the local residents. The supply equation (11.2) contains the variable PF, which is the price of a factor of production. Each observation is indexed by i, i = 1, 2, ... N. As explained in the text, prices and quantities in a market are jointly determined; hence, in this econometric model P and Q are both endogenous to the system.

11.2 The Reduced Form Equations

The reduced form equations express each endogenous variable as a linear function of every exogenous variable in the entire system. So, for our example

$$Q_i = \pi_{11} + \pi_{21} P S_i + \pi_{31} D I_i + \pi_{41} P F_i + \nu_{i1}$$
(11.3)

$$P_i = \pi_{12} + \pi_{22} P S_i + \pi_{32} D I_i + \pi_{42} P F_i + \nu_{i2}$$
(11.4)

Since each of the independent variables is exogenous with respect to Q and P, the reduced form equations (11.3) and (11.4) can be estimated using least squares. In **gret1** the script is

```
open "c:\Program Files\gretl\data\POE\truffles.gdt"
ols q const ps di pf
ols p const ps di pf
```

The gretl results appear in Table 11.1

Table 11.1: The least squares estimates of the reduced form equations.

 $\begin{aligned} \widehat{\mathbf{q}} &= 7.89510 + 0.656402 \,\mathrm{ps} + 2.16716 \,\mathrm{di} - 0.506982 \,\mathrm{pf} \\ (2.434) & (4.605) \,\mathrm{ps} + 2.16716 \,\mathrm{di} - 0.506982 \,\mathrm{pf} \\ T &= 30 \quad \overline{R}^2 = 0.6625 \quad F(3,26) = 19.973 \quad \widehat{\sigma} = 2.6801 \\ (t\text{-statistics in parentheses}) \end{aligned}$

$$\widehat{\mathbf{p}} = -32.5124 + 1.70815 \,\mathrm{ps} + 7.60249 \,\mathrm{di} + 1.35391 \,\mathrm{pf} \\ (-4.072) \quad (4.868) \quad F(3,26) = 69.189 \quad \widehat{\sigma} = 6.5975 \\ (t\text{-statistics in parentheses})$$

11.3 The Structural Equations

The structural equations are estimated using two-stage least squares. The basic **gretl** commands for this estimator are discussed in Chapter 10. The instruments consist of **all** exogenous variables, i.e., the same variables you use to estimate the reduced form equations (11.3) and (11.4).

The **gretl** commands to open the truffle data and estimate the structural equations using twostage least squares are: open "c:\Program Files\gretl\data\poe\truffles.gdt"
tsls q const p ps di; const ps di pf
tsls q const p pf; const ps di pf

The second line of the script estimates the demand equation. The **gretl** command **tsls** calls for the two-stage least squares estimator and it is followed by the structural equation you wish to estimate. List the dependent variable (q) first, followed by the regressors variables (const p ps di). A semicolon separates the model to be estimated from the list of instruments (const ps di pf). Don't forget to list the constant again as an instrument. The third line uses the same format to estimate the parameters of the supply equation. Refer to section 10.2, and Figures 10.1 and 10.2 specifically, about using the GUI to estimate the model.

The results from two-stage least squares appear below in Table 11.2

Table 11.2: Two-stage least square estimates of the demand and supply of truffles.

Demand

 $\begin{aligned} \widehat{\mathbf{q}} &= -4.27947 - 0.374459 \,\mathrm{p} + 1.29603 \,\mathrm{ps} + 5.01398 \,\mathrm{di} \\ (-0.772) & (-2.273) \,\mathrm{p} + 1.29603 \,\mathrm{ps} + 5.01398 \,\mathrm{di} \\ T &= 30 \quad \bar{R}^2 = 0.1376 \quad F(3,26) = 2.5422 \quad \hat{\sigma} = 4.93 \\ & (t\text{-statistics in parentheses}) \end{aligned}$

Supply

 $\widehat{\mathbf{q}} = \underbrace{20.0328}_{(16.379)} + \underbrace{0.337982}_{(13.563)} \mathrm{p} - \underbrace{1.00091}_{(-12.128)} \mathrm{pf}$ $T = 30 \quad \overline{R}^2 = 0.8946 \quad F(2, 27) = 124.08 \quad \widehat{\sigma} = 1.4976$ (t-statistics in parentheses)

11.4 Fulton Fish Example

The following script estimates the reduced form equations using least squares and the demand equation using two-stage least squares for Graddy's Fulton Fish example.

In the example, ln(quantity) and ln(price) are endogenously determined. There are several potential instruments that are available. The variable *stormy* may be useful in identifying the demand equation. In order for the demand equation to be identified, there must be at least one variable available that effectively influences the supply of fish without affecting its demand. Presumably, stormy weather affects the fishermen's catch without affecting people's appetite for fish! Logically, stormy may be a good instrument.

The model of demand includes a set of dummy variables for day of the week. Friday is omitted to avoid the dummy variable trap. These day of week variables are not expected to affect supply; fishermen catch the same amount on average on any working day. They may affect demand though, since people in some cultures buy more fish on some days than others.

In both demand and supply equations, ln(price) is the right-hand side endogenous variable. Identification of the demand equation requires *stormy* to be significantly correlated with *lprice*. This can be determined by looking at the t-ratio in the *lprice* reduced form equation.

For supply to be identified, at least one of the day of the week dummy variables (mon tue wed thu), which are excluded from the supply equation, has to be significantly correlated with *lprice* in the reduced form. If not, the supply equation cannot be estimated; it is not identified.

Proceeding with the analysis, open the data and estimate the reduced form equations for *lquan* and *lprice*. Go ahead and conduct the joint test of the day of the week variables using the --quiet option. The results of this test can help determine whether the supply equation is identified.

```
open "c:\Program Files\gretl\data\poe\fultonfish.gdt"
```

```
#Estimate the reduced form equations
ols lquan const stormy mon tue wed thu
ols lprice const stormy mon tue wed thu
omit mon tue wed thu --quiet
```

The reduced form results for *lquan* appear below:

Variable	Coefficient	Std. Error	t-statistic	p-value
const	8.810	0.147	59.922	0.000
stormy	-0.388	0.144	-2.698	0.008
mon	0.101	0.207	0.489	0.626
tue	-0.485	0.201	-2.410	0.018
wed	-0.553	0.206	-2.688	0.008
thu	0.054	0.201	0.267	0.790
Standard error of residuals $(\hat{\sigma})$			0.68	1790

Model 1: OLS estimates using the 111 observations 1–111 Dependent variable: lquan

Standard error of residuals $(\hat{\sigma})$	0.681790
Unadjusted R^2	0.193372
F(5, 105)	5.03429
p-value for $F()$	0.000356107

and the results for *lprice*

Variable	e Coefficient	Std. Error	t-statistic	p-value
const	-0.272	0.076	-3.557	0.001
stormy	0.346	0.075	4.639	0.000
mon	-0.113	0.107	-1.052	0.295
tue	-0.041	0.105	-0.394	0.695
wed	-0.012	0.107	-0.111	0.912
thu	0.050	0.104	0.475	0.636
	Unadjusted R^2		0.178889	
F(5, 105)		4.57511		
p-value for $F()$		0.000815589		

Model 2: OLS estimates using the 111 observations 1–111 Dependent variable: lprice

In this equation, stormy is highly significant with a t-ratio of 4.639, but the daily dummy variables are not. A joint test of their significance reveals that they are not jointly significant, either; the F-statistic has a p-value of only .65. Supply is not identified and can't be estimated without better instruments.

The two-stage least squares estimates of the demand equation are obtained using:

to produce the result:

Model 3: TSLS estimates using the 111 observations 1–111 Dependent variable: lquan Instruments: stormy

Variable	Coefficient	Std. Error	t-statistic	p-value
const	8.506	0.166	51.189	0.000
mon	-0.025	0.215	-0.118	0.906
tue	-0.531	0.208	-2.552	0.011
wed	-0.566	0.213	-2.662	0.008
$_{ m thu}$	0.109	0.209	0.523	0.601
lprice	-1.119	0.429	-2.612	0.009

Mean of dependent variable	8.52343
S.D. of dependent variable	0.741672
Sum of squared residuals	52.0903
Standard error of residuals $(\hat{\sigma})$	0.704342
F(5, 105)	5.13561
p-value for $F()$	0.000296831

Hausman test -

Null hypothesis: OLS estimates are consistent Asymptotic test statistic: $\chi_1^2 = 2.4261$ with p-value = 0.119329

First-stage F(1, 105) = 21.5174

11.5 Script

open "c:\Program Files\gretl\data\PoE\truffles.gdt" # Least Squares ols q const ps di pf ols p const ps di pf

#Two Stage Least Squares
open "c:\Program Files\gretl\data\PoE\truffles.gdt"
tsls q const p ps di;const ps di pf
tsls q const p pf;const ps di pf

Fulton Fish example
open "c:\Program Files\gretl\data\PoE\fultonfish.gdt"

#Estimate the reduced form equations ols lquan const stormy mon tue wed thu ols lprice const stormy mon tue wed thu omit mon tue wed thu --quiet
Chapter 12

Analyzing Time Series Data and Cointegration

The main purpose this chapter is to use **gret1** to explore the time series properties of your data. One of the basic points we make in econometrics is that the properties of the estimators and their usefulness for point estimation and hypothesis testing depend on how the data behave. For instance, in a linear regression model where errors are correlated with regressors, least squares won't be consistent and consequently it should not be used for either estimation or subsequent testing.

In time series regressions the data need to be **stationary**. Basically this requires that the means, variances and covariances of the data series cannot depend on the time period in which they are observed. For instance, the mean and variance of the probability distribution that generated GDP in the third quarter of 1973 cannot be different from the one that generated the 4th quarter GDP of 2006. Observations on stationary time series can be correlated with one another, but the nature of that correlation can't change over time. U.S. GDP is growing over time (not mean stationary) and may have become less volatile (not variance stationary). Changes in information technology and institutions may have shortened the persistence of shocks in the economy (not covariance stationary). Nonstationary time series have to be used with care in regression analysis. Methods to effectively deal with this problem have provided a rich field of research for econometricians in recent years.

12.1 Series Plots

The first thing to do when working with time series is to take a look at the data graphically. A time series plot will reveal potential problems with your data and suggest ways to proceed statistically. In **gretl** time series plots are simple to generate since there is a built in function that performs this task. Open the data file usa.gdt.

open "c:\Program Files\gretl\data\poe\usa.gdt"

Fist, note that I have renamed the variables for this book to be a little more descriptive than in POE. I assure you that the variables are the same, only the names changed (to protect the innocent!). Then, use your mouse to select all of the series as shown in Figure 12.1 below. Then,



Figure 12.1: Select all of the series.

select Add>First differences of selected variables from the pull-down menu as shown in Figure 12.2. The first differences of your time series are added to the data set and each of the

📓 gretl		
<u>File T</u> ools <u>D</u> ata <u>y</u>	ljew <u>A</u>dd Sample ⊻ariable <u>M</u> odel	Help
usa.gdt ID # Variable name D 0 const a 1 gdp r 2 inflation II 3 FedFunds F 4 Bonds 3	Logs of selected variables Logs of selected variables bescrip Squares of selected variables autor Einst differences of selected variables cell U: Log differences of selected variables cell U: Seasonal differences of selected variables seasonal differences of selected variables Seasonal differences of selected variables seasonal differences of selected variables Seasonal differences of selected variables seasonal differences of selected variables Seasonal differences of selected variables	
	Bandom variable Periodic dummies Unit dummies Jime dummies Dummies for selected discrete variables Define new variable Define matrix	
🗐 🎽 🎦 📖	fx 🔁 🔯 🖉 â 📄	

Figure 12.2: Add the first differences of the selected series from the pull-down menu.

differenced series is prefixed with 'd_', e.g., $gdp_t - gdp_{t-1} = d_{-g}dp$.

Plotting the series can be done in any number of ways. The easiest is to use view>multiple graphs>Time series from the pull-down menu. This will allow you to graph the eight series in two batches. Two batches are required since the maximum number of series that can be graphed simultaneously is currently limited to six.

Select gdp, inflation, d_gdp, and d_inflation as shown in Figure 12.3. The result appears

📓 gretl: define graph		
Available vars		Selected vars
gdp inflation FedFunds Bonds d_gdp d_inflation d_FedFunds d_Bonds	Select ->	gdp inflation d_gdp d_inflation
	<- Remove	
🔀 Help 🔒 🤇	_lear 🛛 🔏 🤇	ancel

Figure 12.3: Graphing multiple time series using the selection box.

in Figure 12.4. Repeat this exercise for the remaining series to get the result shown in Figure 12.5.

12.2 Tests for Stationarity

The (augmented) Dickey-Fuller test can be used to test for the stationarity of your data. To perform this test, a few decisions have to be made regarding the time series. The decisions are usually made based on visual inspection of the time series plots. By looking at the plots you can determine whether the time series have a linear or quadratic trend. If the trend in the series is quadratic then the differenced version of the series will have a linear trend in them. In Figure 12.5 you can see that the Fed Funds rate appears to be trending downward and its difference appears to wander around some constant amount. Ditto for bonds. This suggests that the Augmented Dickey Fuller test regressions for each of the series should contain a constant, but not a time trend.

The GDP series in the upper left side of Figure 12.4 appears to be slightly quadratic in time. The differenced version of the series that appears below it has a slight upward drift and hence I would choose an ADF test that included a constant and a time trend. As you may have guessed, analyzing time series in this way is a bit like reading entrails and there is something of an art to it. Our goal is to reduce some of the uncertainty using formal tests whenever we can, but realize that choosing the appropriate test specification requires some judgement by the econometrician.



Figure 12.4: Multiple time series graphs.

Figure 12.5: Multiple time series graphs for Fed Funds rate and 3 year bonds.



The next decision is to pick the number of lagged terms to include in the ADF regressions. Again, this is a judgement call, but the residuals from the ADF regression should be void of any autocorrelation. **Gretl** is helpful in this respect since it reports the outcome of an autocorrelation test whenever the built-in ADF routines are used. Below is the example from your text, where the stationarity of the Fed Funds rate and the three year bond series are explored.

To perform the ADF test on the Fed Funds rate, use the cursor to highlight the series and click Variable>Augmented Dickey Fuller test from the pull-down menu as shown in Figure 12.6 below. This brings up the dialog box shown in the next Figure, 12.7. Notice that here is where

	iguio 12		ene mer eese n	0111 0	ne puil down menu.
gre	etl				2
Eile	<u>T</u> ools <u>D</u> ata	<u>V</u> iew <u>A</u> dd <u>S</u> ample	<u>V</u> ariable <u>M</u> odel		Help
usa.g	dt *		🔍 Find	Ctrl+F	
ID#	Variable name	Descriptive label	Display values		
0	const	auto-generated consta	Summary statistics		
1	qdp	real US Gross Domestic	Erequency distribution		
2	inflation	Inflation Rate	Frequency plot		
3	FedFunds	Federal Funds Rate	Estimated density plot		
4	Bonds	3 year bond rate	<u>G</u> ini coefficient		
5	d_gdp	= first difference of gd	<u>R</u> ange-mean graph		
6	d_inflation	= first difference of inf	Time series plot		
7	d_FedFunds	= first difference of Fe	Correlogram		
8	d_Bonds	= first difference of Bo	Spectrum	•	
			Augmented Dickey-Fuller test		
			ADF-GLS test		
			KPSS test		
			Eilter	•	
			X-12-ARIMA analysis		
1			<u>T</u> RAMO analysis		
) 📝 🛅 🔛	🛯 fx 🚺 🔛 🚈	Hurst exponent		

Figure 12.6: Choose the ADF test from the pull-down menu

you inform **gretl** whether you want to include a constant, trend, trend squared, seasonal dummies, etc. We have chosen to use only 1 lag, and to include a constant in the ADF regression. Also, we've checked the box to have **gretl** report the results from the regression itself in order to make the results a bit more transparent.

At the bottom of the dialog you must choose whether you want to use the level or the difference of the variable. Choosing the level, as shown in the box, puts the difference on the left-hand side of the regression. This can be a bit confusing, but in reality it should not be. Remember, you are testing to see whether the levels values of the series are stationary. Choosing this box is telling **gret1** that you want to first test levels.

If you want to check to see whether the differences are nonstationary, then click the radio button below the one indicated. Click OK and the results appear as in Figure 12.8.

The test results are quite informative. First it tells you that you are performing a test based on a regression with a constant. It provides you with an estimate of γ , which it refers to as a-1, the t-ratio for γ , and the correct p-value for the statistic as computed by Davidson and MacKinnon. It also reports an estimated autocorrelation coefficient for the errors (0.061) which should be small if you have chosen the correct number of lags in the ADF regression.



📓 gretl: ADF test	×				
Lag order for ADF test:					
test without constant					
✓ with constant					
with constant and trend					
with constant, trend and trend squared					
🔲 include seasonal dummies	include seasonal dummies				
show regression results					
Lest down from maximum lag order					
 use level of variable 					
O use first difference of variable					
Help 💥 Cancel 🖉 QK					

Figure 12.8: The ADF test results.

```
Augmented Dickey-Fuller tests, order 1, for FedFunds

sample size 79

unit-root null hypothesis: a = 1

test with constant

model: (1 - L)y = b0 + (a-1)*y(-1) + ... + e

1st-order autocorrelation coeff. for e: 0.061

estimated value of (a - 1): -0.0370668

test statistic: tau_c(1) = -2.0903

asymptotic p-value 0.2487

Augmented Dickey-Fuller regression

OLS estimates using the 79 observations 1985:3-2005:1

Dependent variable: d_FedFunds

VARIABLE COEFFICIENT STDERROR T STAT P-VALUE
```

		DID DIG.OI.		
const	0.177862	0.100751	1.765	
FedFunds_1	-0.0370668	0.0177327	-2.090	0.24875
d_FedFunds_1	0.672478	0.0853664	7.878	

The null hypothesis of the ADF test is that the time series has a unit root and is not stationary. *If you reject this hypothesis then you conclude that the series is stationary.* To not reject the null means that the level is *not* stationary. Here, the test statistic for the stationarity of the Fed Funds rate is -2.090 which has a p-value of 0.24875. Nonstationarity of the Fed Funds rate can not be rejected in this case at the usual 5 or 10% levels of significance.

One more thing should be said about the ADF test results. **Gretl** expresses the model in a slightly different way than your textbook. The model is

$$(1-L)y_t = \beta_0 + (\alpha - 1)y_{t-1} + \alpha_1 \Delta y_{t-1} + e_t \tag{12.1}$$

The coefficient β_0 is included because you believe the series has a trend, $(\alpha - 1) = \gamma$ is the coefficient of interest in the Dickey-Fuller regression, and α_1 is the term that 'augments' the Dickey-Fuller regression. It is included to eliminate autocorrelation in the model's errors, e_t , and more lags can be included if needed to accomplish this. The notation on the left side of the equation $(1-L)y_t$ makes use of the lag operator, L. The lag operator performs the magic $Ly_t = y_{t-1}$. Thus, $(1-L)y_t = y_t - Ly_t = y_t - y_{t-1} = \Delta y_t$!

The next thing to do is to create a set of summary statistics. In this case, the textbook has you produce summary statistics for subsamples of the data. The first subsample consists of the 40 observations from 1985:1 to 1994:4. The second also contains 40 observations (a decade!) and continues from 1995:1 to 2004:4. The summary command is used to obtain the summary statistics on the desired subsample. In the script, open the data file *usa.gdt* and change the sample to 1985:1-1994:4 using the command smpl 1985:1 1994:4. Issue the summary command to print the summary statistics of all variables in memory to the screen. Finally, restore the sample to the full range using smpl full.

Gretl's smpl functions are cumulative. This means that whatever modifications you make to the sample are made based on the sample that is already in memory. So, to get summary statistics on the second subsample (which is not in memory) you have to restore the full sample first using smpl full. It is a little clumsy, but it makes sense once you know how it works.

open "c:\Program Files\gretl\data\poe\usa.gdt"

smpl 1985:1 1994:4
summary
smpl full
smpl 1995:1 20054:4
summary
smpl full

The sample can be manipulated through the dialogs as well. Open the dataset and select Sample>Set range from the pull-down menu to reveal the dialog in Figure 12.9. Use the scroll buttons to change the ending date to 1994:4. The observation counter will change and show that the selected sample

Figure 12.9: Choose Sample>Set range to reveal the Set sample dialog box. Use the scroll buttons to set the desired sample range.



has 40 observations. Click OK and you are returned to the main **gretl** window. This is shown in the next Figure, 12.10.

Figure 12.10: Any changes to the sample should be visible in the main window. Here you can see that the data in memory consist of the 1985:1-1994:4 subsample.

1	gre	etl						
	Eile	<u>T</u> ools <u>D</u> ata	⊻iew <u>A</u> dd	<u>S</u> ample	<u>V</u> ariable	<u>M</u> odel		<u>H</u> elp
	usa.g	dt *						
	ID #	Variable name	Descriptive la	abel				
	0	const	auto-genera	ted consta	nt			
	1	gdp	real US Gros	s Domestic	Product			
	2	inflation	Inflation Rat	е				
	3	FedFunds	Federal Fund	is Rate				
	4	Bonds	3 year bond	rate				
	5	d_gdp	= first differ	ence of gd	р			
	6	d_inflation	= first differ	ence of inf	lation			
	- 7	d_FedFunds	= first differ	ence of Fe	dFunds			
	8	d_Bonds	= first differ	ence of Bo	nds		The full range and available subsample are given here	
	Quarterly: Full range 1985:1 - 2005:1; sample 1985:1 - 1994:4							

Now, select all of the levels variables either by holding down the **Crtl** key and clicking on each of the levels variables or by clicking on the first variable (gdp), holding down the **Shift** key, and clicking on the last desired variable in the list (Bonds). This is a Microsoft Windows convention and may not work the same on other systems.

Once the desired variables are selected, and hence highlighted, choose View>Summary statistics will reveal the desired information, which is shown below:

Variable	Mean	Median	Minimum	Maximum
gdp	5587.70	5650.35	4119.50	7232.20
inflation	3.55601	3.50904	1.30831	6.03757
FedFunds	6.28808	6.65667	2.99000	9.72667
Bonds	7.22700	7.49000	4.32000	10.6767
Variable	Std. Dev.	C.V.	Skewness	Ex. kurtosis
Variable gdp	Std. Dev. 922.950	C.V. 0.165175	Skewness 0.0439792	Ex. kurtosis -1.1688
Variable gdp inflation	Std. Dev. 922.950 1.09067	C.V. 0.165175 0.306713	Skewness 0.0439792 0.0926796	Ex. kurtosis -1.1688 -0.455736
Variable gdp inflation FedFunds	Std. Dev. 922.950 1.09067 2.08741	C.V. 0.165175 0.306713 0.331963	Skewness 0.0439792 0.0926796 -0.301024	Ex. kurtosis -1.1688 -0.455736 -1.1781

Summary Statistics, using the observations 1985:1–1994:4

Now restore the full sample using Sample>Restore full range from the pull-down menu and repeat, changing the sample range to 1995:1 - 2004:4 using the set sample dialog. The results are

Variable	Mean	Median	Minimum	Maximum
gdp	7584.25	7298.30	4119.50	12198.8
inflation	2.99005	2.90131	1.24379	6.03757
FedFunds	5.16955	5.50667	0.996667	9.72667
Bonds	5.94741	6.00000	1.77333	10.6767
Variable	Std. Dev.	C.V.	Skewness	Ex. kurtosis
gdp	2312.62	0.304924	0.266928	-1.1168
inflation	1.05382	0.352444	0.571135	0.0136503
FedFunds	2.29634	0.444206	-0.199304	-0.760287
Bonds	2.03711	0.342521	-0.109496	-0.553230

Summary Statistics, using the observations 1985:1–2005:1

12.3 Spurious Regressions

It is possible to estimate a regression and find a statistically significant relationship even if none exists. In time series analysis this is actually a common occurrence when data are not stationary. This example uses two data series, rw1 and rw2, that were generated as independent random walks.

$$rw_1: \qquad y_t = y_{t-1} + v_{1t} \tag{12.2}$$

$$rw_2: \qquad x_t = x_{t-1} + v_{2t} \tag{12.3}$$

The errors are independent standard normal random deviates generated using a pseudo-random number generator. As you can see, x_t and y_t are not related in any way. To explore the empirical relationship between these unrelated series, load the *spurious.dta* data, create a time variable, and declare the data to be time series.

open "c:\Program Files\gretl\data\poe\spurious.gdt"

The sample information at the bottom of the main **gretl** window indicates that the data have already been declared as time series and that the full range (1-700) is in memory. The first thing to do is to plot the data using a time series plot. To place both series in the same time series graph, select View>Graph specified vars.>Time series plots from the pull-down menu. This will reveal the 'define graph' dialog box. Place both series into the 'Selected vars' box and click OK. The result appears in Figure 12.11 below. A scatter plot is revealing as well. Select View>Graph

Figure 12.11: These random walk series appear to be correlated but they are not. It is not uncommon to observe spurious relationships between nonstationary series.



specified vars.>X-Y scatters and place rw2 on the X-axis, rw1 on the Y-axis to produce the next graph (Figure 12.12. The linear regression confirms this. Left click on the graph to reveal the pop-up menu shown in Figure 12.13. Select the OLS estimates option to reveal the regression results in Table 12.1.

The coefficient on rw2 is positive (.842) and significant (t = 40.84 > 1.96). However, these variables are not related! The observed relationship is purely spurious. The cause of the spurious result is the nonstationarity of the two series. This is why you must check your data for stationarity whenever you use time series in a regression.

The script to produce these graphs is very simple. Use

```
open "c:\Program Files\gretl\data\poe\spurious.gdt"
```

```
gnuplot rw1 rw2 --with-lines --time-series
gnuplot rw1 rw2
```

Figure 12.12: The scatter plot of the random walk series makes them appear to be related, but they are not. They are nonstationary and the relationship is spurious.



Figure 12.13: Left-click on the graph to reveal this menu. Choose OLS estimates to reveal the underlying least squares results that produce the regression line in the graph.

Save as Windows metafile (EMF)	•
Save as PNG	
Save as postscript (EPS)	►
Save as PDF	
Copy to clipboard	►
Save to session as icon	
Zoom	
Print	►
Display PDF	
OLS estimates	
Edit	
Close	

Table 12.1: OLS estimates of a spurious relationship using the 700 observations of the *spurious.gdt* dataset.

		Dependent varia	able: rw1		
Variable	Coefficient	Std. E	rror	t-statistic	p-value
const rw2	$17.8180 \\ 0.842041$	0.6 0.0	520478 0206196	$28.7167 \\ 40.8368$	0.0000 0.0000
	Un Ad	adjusted R^2 ljusted \bar{R}^2	$\begin{array}{c} 0.704943 \\ 0.704521 \end{array}$		

ols rw1 rw2 const

The first plot applies lines and uses the time-series option to use time as the X-axis measurement. The second plot is a simple scatter with the first variable on the Y-axis and the second on the X-. The final statement estimates the regression.

12.4 Cointegration

Two nonstationary series are cointegrated if they tend to move together through time. For instance, we have established that the levels of the Fed Funds rate and the 3-year bond are non-stationary, whereas their differences are stationary. In the opaque terminology used in time series literature, each series is said to be "integrated of order 1" or I(1). If the two nonstationary series move together through time then we say they are "cointegrated." Economic theory would suggest that they should be tied together via arbitrage, but that is no guarantee. In this context, testing for cointegration amounts to a test of the substitutability of these assets.

The basic test is very simple. Regress one I(1) variable on another using least squares. If the series are cointegrated, the residuals from this regression will be stationary. This is verified using augmented Dickey-Fuller test.

The null hypothesis is that the residuals are nonstationary, which implies that the series are not cointegrated. Rejection of this leads to the conclusion that the series are cointegrated. The **coint** function in **gretl** carries out each of the three steps in this test. First, it carries out a Dickey-Fuller test of the null hypothesis that each of the variables listed has a unit root. Then it estimates the cointegrating regression using least squares. Finally, it runs a Dickey Fuller test on the residuals from the cointegrating regression. This procedure, referred to as the Engle-Granger cointegration test and discussed in chapter 12 of Hill et al. (2007), is the one done in **gretl** by default. **Gretl** can also perform cointegration tests based on maximum likelihood estimation of the cointegrating relationships proposed by Johansen and summarized in (?, Chapter 20). The Johansen tests use

the coint2 command, which is explained in gretl's documentation.

Figure 12.14 shows the dialog box used to test cointegration in this way. To obtain it use Model>Time series>Cointegration test>Engle-Granger from the main gretl window. In the dialog box you have to indicate how many lags you want in the initial Dickey-Fuller regressions on the the variables, which variables you want to include in the cointegrating relationship, and whether you want a constant, trend, or quadratic trend in the regressions.

To select these additional modeling options you'll have to click on the down arrow button indicated in Figure 12.14. This will reveal the four choices shown in the next figure (Figure 12.15).

📓 gretl: cointegration t	est		
	Cointegration		
gdp inflation FedFunds Bonds d_gdp d_inflation d_FedFunds d_Bonds	lag order: Va Add ->	ariables to test Bonds d_FedFunds	
	<- Remove		
Test down from maximum Skip initial DF tests test without constant	Click additi	this to i onal mo choices	reveal deling s
🔀 Help 🔒 🖆	ear 🛛 🔏 🤇	ancel	<u> «Ро</u> к

Figure 12.14: The dialog box for the cointegration test.

Figure 12.15: The pull-down menu for choosing whether to include constant or trends in the ADF regression.

	Test down from maximum lag order	Use the	cursor to
	Skip initial DF tests	highight t test with	his option: constant
	test without constant		
_	test with constant		
	with constant and trend	Marrie 1	
	with constant and quadratic trend	M Cancel	<u>0k</u>

12.5 The Analysis Using a Gretl Script

Below, you will find a summary of the **gretl** commands used to produce the results for the usa.gdt data from Chapter 12.

```
open "c:\Program Files\gretl\data\poe\usa.gdt"
# Difference each variable
diff gdp inflation FedFunds Bonds
# Augmented Dickey Fuller regressions
# This is the manaul way of doing this regression
ols d_FedFunds const FedFunds(-1) d_FedFunds(-1)
ols d_Bonds const Bonds(-1) d_Bonds(-1)
# Augmented Dickey Fuller regressions using the built-in function
# Note: 1 lag is called for and a constant is included (--c)
adf 1 FedFunds --c --verbose
adf 1 Bonds --c --verbose
# Dickey-Fuller regressions for first differences
# Note: adf 0 indicates no lags for the difference version
adf 0 FedFunds --nc --verbose --difference
adf 0 Bonds --nc --verbose --difference
# Engle-Granger test of cointegration
# Note: one lag is used in the adf portion of the test
coint 1 Bonds FedFunds
```

The diff function takes the first difference of each series. The adf function conducts the augmented Dickey-Fuller test. The number 1 that follows the adf command is the number of lags to use in the augmented version, in this case only one. Then, list the series name and any options you wish to invoke. Here, the --c option is used, indicating that we want a constant term included in the Dickey-Fuller regression. The --verbose statement is included so that gret1 will print the results from the Dickey-Fuller regression itself. I think this makes interpreting the result much easier, so I always include it.

Other options in the example include -nc which directs the Dickey-Fuller regression to omit the constant altogether. The --difference option tells gretl to run the augmented Dickey-Fuller regressions under the assumption that the first difference of the series is nonstationary.

Finally, the **coint** command conducts the Engle-Granger test for the cointegration of the two series that follow. Again, the number 1 that follows **coint** is actually for the first step of the procedure, which tells **gretl** how many lags to include in the initial augmented Dickey-Fuller regressions.

The output generated from the simple command coint 1 Bonds FedFunds is shown below.

```
# Engle-Granger test of cointegration
? coint 1 Bonds FedFunds
Step 1: testing for a unit root in Bonds
Augmented Dickey-Fuller test, order 1, for Bonds
sample size 79
unit-root null hypothesis: a = 1
  test with constant
   estimated value of (a - 1): -0.0562195
  test statistic: tau_c(1) = -1.97643
   asymptotic p-value 0.2975
Step 2: testing for a unit root in FedFunds
Augmented Dickey-Fuller test, order 1, for FedFunds
sample size 79
unit-root null hypothesis: a = 1
  test with constant
   estimated value of (a - 1): -0.0370668
  test statistic: tau_c(1) = -2.0903
   asymptotic p-value 0.2487
Step 3: cointegrating regression
```

Cointegrating regression -OLS estimates using the 81 observations 1985:1-2005:1 Dependent variable: Bonds VARIABLE COEFFICIENT STDERROR T STAT P-VALUE 0.19482 const 1.64373 8.437 <0.00001 *** FedFunds 0.832505 0.03448 24.147 <0.00001 *** Unadjusted R-squared = 0.880682 Adjusted R-squared = 0.879172 Durbin-Watson statistic = 0.413856 First-order autocorrelation coeff. = 0.743828 Akaike information criterion (AIC) = 175.927 Schwarz Bayesian criterion (BIC) = 180.716 Hannan-Quinn criterion (HQC) = 177.848 Step 4: Dickey-Fuller test on residuals lag order 1 sample size 79 unit-root null hypothesis: a = 1 estimated value of (a - 1): -0.31432 test statistic: $tau_c(2) = -4.54282$ asymptotic p-value 0.0009968 P-values based on MacKinnon (JAE, 1996) There is evidence for a cointegrating relationship if: (a) The unit-root hypothesis is not rejected for the individual variables. (b) The unit-root hypothesis is rejected for the residuals (uhat) from the cointegrating regression.

Notice that at the bottom of the output **gretl** gives you some useful advice on interpreting the outcome of the test. Cointegration requires both series to be I(1)-not rejecting nonstationarity in the initial Dickey-Fuller regressions and then rejecting nonstationarity in the Dickey-Fuller regression using the residuals. Nice!

12.6 Script

```
open "c:\Program Files\gretl\data\poe\usa.gdt"
```

```
# Difference each variable
diff gdp inflation FedFunds Bonds
# Augmented Dickey Fuller regressions
ols d_FedFunds const FedFunds(-1) d_FedFunds(-1)
ols d_Bonds const Bonds(-1) d_Bonds(-1)
# Augmented Dickey Fuller regressions using built in functions
adf 1 FedFunds --c --verbose
adf 1 Bonds --c --verbose
# Dickey-Fuller regressions for first differences
adf 0 FedFunds --nc --verbose --difference
adf 0 Bonds --nc --verbose --difference
# Summary Statistics
smpl 1985:1 1994:4
summary
smpl full
smpl 1995:1 2004:4
summary
smpl full
#Spurious Regressions
open "c:\Program Files\gretl\data\poe\spurious.gdt"
gnuplot rw1 rw2 --with-lines --time-series
gnuplot rw1 rw2
ols rw1 rw2 const
# Engle-Granger test of cointegration
open "c:\Program Files\gretl\data\poe\usa.gdt"
coint 1 Bonds FedFunds
```

Chapter 13

Vector Error Correction and Vector Autoregressive Models: Introduction to Macroeconometrics

The vector autoregression model is a general framework used to describe the dynamic interrelationship between stationary variables. So, the first step in your analysis should be to determine whether the levels of your data are stationary. If not, take the first differences of your data and try again. Usually, if the levels (or log-levels) of your time series are not stationary, the first differences will be.

13.1 Vector Error Correction

If the time series are not stationary then we need to modify the vector autoregressive (VAR) framework to allow consistent estimation of the relationships between the series. The vector error correction model (VECM) is just a special case of the VAR for variables that are stationary in their differences (i.e., I(1)) and cointegrated.

In the first example, we use quarterly data on the Gross Domestic Product of Australia and the U.S. to estimate a VEC model. We decide to use the vector error correction model because (1) the time series are not stationary in their levels but are in their differences (2) the variables are cointegrated.

In an effort to keep the discussion moving, the authors of POE opted to avoid discussing how they actually determined the series were nonstationary in levels, but stationary in differences. This is an important step and I will take some time here to explain how one could approach this. There are several ways to do this and I'll show you two ways to do it in gretl.

13.1.1 Series Plots–constant and trends

Our initial impressions of the data are gained from looking at plots of the two series. The data plots are obtained in the usual way after importing the dataset. The data on U.S. and Australian GDP are found in the gdp.gdt file and were collected from 1970:1 - 2004:4.¹ Open the data and set the data structure to quarterly time-series using the setobs 4 command, start the series at 1970:1, and use the --time-series option.

```
open "c:\Program Files\gretl\data\poe\gdp.gdt"
setobs 4 1970:1 --time-series
```

One purpose of the plots is to help you determine whether the Dickey-Fuller regressions should contain constants, trends or squared trends. The simplest way to do this is from the console using the scatters command.

```
scatters usa diff(usa) aus diff(aus)
```

The scatters command produces multiple graphs, each containing one of the listed series. The diff() function is used to take the differences of usa and aus, which appear in the graphs featured in Figure 13.1 below.

This takes two steps from the pull-down menu. First, use the mouse to highlight the two series and then create the differences using Add>First differences of selected variables. Then, select View>Multiple graphs>Time series. Add the variables to the selected list box to produce Figure 13.1.

From the time series plots it appears that the levels are mildly parabolic in time. The differences have a small trend. This means that the augmented Dickey-Fuller (ADF) regressions need to contain these elements.

13.1.2 Selecting Lag Length

The second consideration is the specification of lags for the ADF regressions. There are several ways to select lags and **gretl** automates one of these. The basic concept is to include enough lags in the ADF regressions to make the residuals white noise. These will be discussed presently.

¹POE refers to these variables as U and A, respectively.





Testing Down

The first strategy is to include just enough lags so that the last one is statistically significant. **Gretl** automates this using the **--test-down** option for the augmented Dickey-Fuller regressions. Start the ADF regressions with a generous number of lags and **gretl** automatically reduces that number until the t-ratio on the longest remaining lag is significant at the 10 percent level. For the levels series we start with a maximum lag of 6, include a constant, trend, and trend squared (**--ctt option**), and use the **--test-down** option.

adf 6 usa --ctt --test-down adf 6 aus --ctt --test-down

The result is shown in Figure 13.2. The --test-down option selected two lags for the usa series and three for aus. Both ADF statistics are insignificant at the 5% or 10% level, indicating they are nonstationary. This is repeated for the differenced series using the commands:

adf 6 diff(usa) --ct --test-down adf 6 diff(aus) --ct --test-down

Figure 13.2: Based on ADF tests, the levels of Australian and U.S. GDP are nonstationary.



The selected lags for the U.S. and Australia are one and three, respectively. Both ADF statistics are significant at the 5% level and we conclude that the differences are stationary.

Testing Up

The other strategy is to test the residuals from the augmented Dickey-Fuller regressions for autocorrelation. In this strategy you can start with a small model, and test the residuals of the Dickey-Fuller regression for autocorrelation using an LM test. If the residuals are autocorrelated, add another lagged difference of the series to the ADF regression and test the residuals again. Once the LM statistic is insignificant, you quit you are done. This is referred to as **testing-up**.

To employ this strategy in **gretl**, you'll have to estimate the ADF equations manually using the **ols** command. Since the data series has a constant and quadratic trend, you have to define a time trend (**genr time**) and trend squared (**genr t2 = time*time**) to include in the regressions. You will also need to generate the differences to use in a new function called **lags**. The script to do this follows:

```
genr time
genr t2 = time*time
genr d_usa = diff(usa)
```

Now, estimate a series of augmented Dickey-Fuller regressions using ols. Follow each regression with the LM test for autocorrelation of the residuals discussed in Chapter 9.

```
ols diff(usa) usa(-1) lags(1,d_usa) const time t2 --quiet
modtest 1 --autocorr
ols diff(usa) usa(-1) lags(2,d_usa) const time t2 --quiet
modtest 1 --autocorr
```

The first ols regression is the ADF(1). It includes 1 lagged value of the d_usa as a regressor in addition to the lagged value of usa, a constant, a trend, and a squared trend. Gretl's lags(q,variable) function creates a series of lags from 1 through q of variable. So in the first regression, lags(1,d_usa) creates a single lagged value of d_usa. After the regression, use the modtest 1 --autocorr to conduct the LM test of first order autocorrelation discussed in Chapter 9. If the p-value is greater than .10 then this is your model. If not, add another lag of d_usa using lags(2,d_usa) and repeat the test. In this example, the ADF(2) produces residuals that are not autocorrelated and 'wins' the derby.

In this code example we chose to suppress the results from the first regression so that the output from the tests would fit on one page (Figure 13.3). In practice, you could skip this option and read the Dickey-Fuller t-ratio directly from the output. The only disadvantage of this is that the proper p-value for it is not computed using the manual approach.

If you repeat this exercise for **aus** (as we have done in the script at the end of the chapter) you will find that testing up determines *zero* lags of **d_aus** are required in the Dickey-Fuller regression; testing down revealed *three* lags were needed. The incongruency occurs because we did a poor job of testing up, failing to include enough autocorrelation terms in in the LM test. This illustrates a danger of testing up. When we conducted the LM test using only a single autocorrelation term, we had not searched far enough in the past to detect significant autocorrelations that lie further back in time. Adding terms to the autocorrelation test using modtest 3 --autocorr resolves this.

So which is better, testing down or testing up? I think the econometric consensus is that testing down is safer. We'll leave it for future study!

13.1.3 Cointegration Test

Given that the two series are stationary in their differences (i.e., both are I(1)), the next step is to test whether they are cointegrated. In the discussion that follows, we return to reproducing results from *POE*. To do this, use least squares to estimate the following regression.

$$aus_t = \beta usa_t + e_t \tag{13.1}$$

obtain the residuals, \hat{e}_t , and then estimate

$$\Delta \hat{e}_t = \gamma \hat{e}_{t-1} + u_t \tag{13.2}$$

Figure 13.3: Manually estimate the ADF regressions and use LM tests for autocorrelation to determine the proper lag length.

gretl: script output					
🔏 🗛 🛱 🛱 😫 🖄	🥱 ×				
<pre>? ols diff(usa) usa(? lmtest 1autocor</pre>	-1) lags(1,d_usa) r	const time t2 -	-quiet		-
Breusch-Godfrey test OLS estimates using Dependent variable:	for first-order a the 121 observatio uhat	utocorrelation ns 1970:4-2000:	4		
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE	
const usa_1 d_usa_1 time t2 uhat_1 Unadjusted R-squar	-2.09676 0.0450922 f 0.858795 -0.00659378 -0.000150292 -0.964336 ed = 0.0574421	1.41084 0.0342477 0.336339 0.00835085 9.38685E-05 0.364285	-1.486 1.317 2.553 -0.790 -1.601 -2.647	0.13997 0.19057 0.01198 ** 0.43139 0.11210 0.00925 ***	
Test statistic: LMF with p-value = P(F(1	= 7.008417, ,115) > 7.00842) =	0.00925	Sig	nificant	
Alternative statisti with p-value = P(Chi	c: TR^2 = 6.950492 -square(1) > 6.950	, 49) = 0.00838	autoc	orrelation	
Ljung-Box Q' = 0.452	324 with p-value =	P(Chi-square(1) > 0.452	2324) = 0.501	
? ols diff(usa) usa(? lmtest 1autocor	-1) lags(2,d_usa) r	const time t2 -	-quiet		
Breusch-Godfrey test for first-order autocorrelation OLS estimates using the 120 observations 1971:1-2000:4 Dependent variable: uhat					
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE	
const usa_1 2 lags o d_usa_2 time t2 uhat_1	-0.457842 0.0121178 0.212033 -0.0665249 -0.00436385 -1.94450E-05 -0.239387	1.64093 0.0403440 0.351573 0.134242 0.00949874 0.000103370 0.388727	-0.279 0.300 0.603 -0.496 -0.459 -0.188 -0.616	0.78074 0.76445 0.54765 0.62117 0.64682 0.85113 0.53925	
Unadjusted R-squared = 0.00553086					
Test statistic: LMF = 0.628464, Not significant: with p-value = P(F(1,113) > 0.628464) = 0.43					
Alternative statistic: $TR^2 = 0.663704$, with p-value = P(Chi-square(1) > 0.663704) = 0.415					
Ljung-Box Q' = 0.0278827 with p-value = P(Chi-square(1) > 0.0278827) = 0.867					
Close					

This is the "case 1 test" from Chapter 12 of Hill et al. (2007) and the 5% critical value for the t-ratio is -2.76. The following script estimates the model cointegrating regression, saves the residuals, and estimates the regression required for the unit root test.

ols aus usa
genr uhat = \$uhat
ols diff(uhat) uhat(-1)

The result is:

$$\Delta \hat{e}_t = -\underbrace{0.127937\hat{e}_{t-1}}_{(0.044279)} \tag{13.3}$$

 $T = 123 \quad \bar{R}^2 = 0.0640 \quad F(1, 122) = 8.3482 \quad \hat{\sigma} = 0.5985$ (standard errors in parentheses)

The t-ratio is -0.1279/.0443 = -2.889 which lies in the rejection region for this test. Therefore, you reject the null hypothesis of no cointegration.

13.1.4 VECM

You have two difference stationary series that are cointegrated. Consequently, an error correction model of the short-run dynamics can be estimated using least squares. The error correction model is:

$$\Delta aus_t = \beta_{11} + \beta_{12}\hat{e}_{t-1} + v_{1t}$$
(13.4)

$$\Delta usa_t = \beta_{21} + \beta_{22}\hat{e}_{t-1} + v_{2t} \tag{13.5}$$

and the estimates

$$\begin{aligned} \Delta \widehat{aus}_t &= 0.491706 + -0.0987029 \hat{e}_{t-1} \\ (8.491) &(-2.077) \\ \Delta \widehat{usa}_t &= 0.509884 + +0.0302501 \hat{e}_{t-1} \\ (10.924) &(0.790) \end{aligned}$$

(t-statistics in parentheses)

which is produced using

ols diff(aus) const uhat(-1)
ols diff(usa) const uhat(-1)

13.2 Vector Autoregression

The vector autoregression model (VAR) is actually a little simpler to estimate than the VEC model. It is used when there is no cointegration among the variables and it is estimated using time series that have been transformed to their stationary values.

In the example from POE, we have macroeconomic data on GDP and the CPI for the United States. The data are found in the *growth.gdt* dataset and have already been transformed into their natural logarithms. In the dataset, $\ln(GDP)$ is referred to as G and $\ln(CPI)$ as P. As in the previous example, the step is to determine whether the variables are stationary. If they are not, then you transform them into stationary time series and test for cointegration.

The data need to be analyzed in the same way as the GDP example. Examine the plots to determine possible trends and use the ADF tests to determine which form of the data are stationary. These data are nonstationary in levels, but stationary in differences. Then, estimate the cointegrating vector and test the stationarity of its residuals. If stationary, the series are cointegrated and you estimate a VECM. If not, then a VAR treatment is sufficient.

Open the data and take a look at the time series plots.

open "c:\Program Files\gretl\data\poe\growth.gdt"
scatters G diff(G) P diff(P)

Next, estimate

$$ln(GDP)_t = \beta_1 + \beta_2 ln(CPI) + u_t \tag{13.6}$$

using least squares and obtain the residuals, \hat{u}_t . Then, difference the least squares residuals and estimate

$$\Delta \hat{u}_t = \alpha_1 + \alpha_2 \hat{u}_{t-1} + residual, \tag{13.7}$$

again, using least squares. The t-ratio on α_2 is computed and compared to the 5% critical value tabled in *POE* (Table 12.3), which is -3.37. The computed value of the statistic is -.97, which is not in the rejection region of the test; we conclude that the residuals are stationary which means that G and P are not cointegrated. The script that accomplishes this is

ols G const P
series uhat = \$uhat
ols diff(uhat) uhat(-1)

You could use the built-in command for the augmented Dickey-Fuller regressions adf to obtain the t-ratio on the lagged residual. Unfortunately, the critical values produced by the automatic routine does not take into account that the regressors are estimated residuals and they are not valid for the Engle-Granger test of cointegration. If you choose to use the adf command, be sure to use the the --nc no constant option in this case.

adf 0 uhat --nc --verbose

The --verbose option tells gretl to print the actual regression results from the Dickey-Fuller test. The regression results will match those you obtained using the manual method above. Ignore the p-value for the Dickey-Fuller since the regressors are residuals. Since G and P are not cointegrated, a vector autoregression model can be estimated using the differences.

The script to estimate a first order VAR appears below:

var 1 diff(P) diff(G)

The diff() function is used to take the first differences of the time series and can be used in the var command. The command var 1 tells gretl to estimate a VAR of order 1, which means lag the right-hand-side variable one period. Then list the variables to include on the right-hand side.

In practice, you might want to explore whether the order of the VAR (number of lags on the right-hand side) are sufficient. This can be done using the --lagselect option in the var statement. You start the VAR with a relatively long lag length and gretl estimates each successively smaller version, computing various goodness-of-fit measures. Gretl then tells you which is the optimal lag length based on each criterion. For instance, starting the VAR at 8 lags and using --lagselect is accomplished by:

var 8 diff(G) diff(P) --lagselect

You can also get gret1 to generate this command through the dialogs. Select Model>Time series>VAR lag selection from the pull-down menu. This reveals the VAR lag selection dialog box. You can choose the maximum lag to consider, the variables to include in the model, and whether the model should contain constant, trend, or seasonal dummies. The output is:

? var 8 diff(G) diff(P) --lagselect VAR system, maximum lag order 8

The asterisks below indicate the best (that is, minimized) values of the respective information criteria, AIC = Akaike criterion, BIC = Schwartz Bayesian criterion and HQC = Hannan-Quinn criterion.

lags	loglik	p(LR)	AIC	BIC	HQC
1	1273.78393		-14.827882	-14.717648	-14.783154
2	1277.61696	0.10461	-14.825929	-14.642206	-14.751382
3	1300.25039	0.00000	-15.043864*	-14.786652*	-14.939498*
4	1303.30981	0.19045	-15.032863	-14.702162	-14.898679
5	1306.32104	0.19748	-15.021299	-14.617108	-14.857295
6	1311.53702	0.03375	-15.035521	-14.557841	-14.841699
7	1313.40649	0.44249	-15.010602	-14.459433	-14.786961
8	1315.11728	0.48990	-14.983828	-14.359170	-14.730368

The AIC, BIC, and HQC criteria each select a VAR with 3 lags.

Obtaining the impulse responses is easy in **gret1**. The first step is to estimate the VAR. From the main **gret1** window choose Model>Time series>Vector Autoregression. This brings up the dialog, shown in Figure 13.4. Set the lag order to 1, and add the differenced variables to the box labeled Endogenous Variables. Make sure the 'Include a constant' box is checked and click OK.

The results are shown in Figure 13.5. You can generate impulse responses by selecting Analysis>Impulse responses from the results window. This will produce the results shown in Figure 13.6.

These can be graphed for easier interpretation from the results window by selecting Graphs>Impulse responses (combined) from the pull-down menu. The graph is shown in Figure 13.7. This yields the graph shown in Figure 13.8.

The forecast error variance decompositions (FEVD) are obtained similarly. Select Analysis>Forecast variance decomposition from the vector autoregression model window to obtain the result shown in Figure 13.9.



Figure 13.4: From the main gretl window, choose Model>Time series>Vector Autogregression to bring up this VAR dialog box.

Figure 13.5: VAR results

VAR system, lag order 1 OLS estimates, observations 1960:3–2004:4 $\left(T=178\right)$

Equation 1: d_-P

Variable	Coefficient	Std. Error	<i>t</i> -statistic	p-value
$const d_P_{t-1}$	$0.00143280 \\ 0.826816$	0.000710432 0.0447068	$2.0168 \\ 18.4942$	$0.0452 \\ 0.0000$
dG_{t-1}	0.0464420	0.0398581	1.1652	0.2455
	Sum of squared resi Standard error of re	iduals 0.0 esiduals $(\hat{\sigma})$ 0.0	00347709 00445747	
	Unadjusted R^2	0.6	667250	
	Adjusted \bar{R}^2	0.6	663447	
	F(2, 175)	175.4	60	
	E	Equation 2: d_G		
Variable	Coefficient	Std. Error	t-statistic	p-value
const	0.00981441	0.00125091	7.8458	0.0000
d_P_{t-1}	-0.326952	0.0787188	-4.1534	0.0001
dG_{t-1}	0.228505	0.0701813	3.2559	0.0014
	Sum of squared resi	iduals 0.0	0107802	
	Standard error of residuals $(\hat{\sigma})$ Unadjusted R^2		00784863	
			.68769	
	Adjusted \bar{R}^2	0.1	59269	
	F(2, 175)	17.7	7656	

Responses to a one-standard error shock in d.	_G
---	----

period	$d_{-}G$	d_P
1	0.00778221	0.000358053
2	0.00166121	0.000657465
3	0.000164635	0.000620753
4	-0.000165336	0.000520894
5	-0.000208088	0.000423005
6	-0.000185852	0.000340084
7	-0.000153659	0.000272555
8	-0.000124224	0.000218217
9	-9.97324e-005	0.000174656
10	-7.98936e-005	0.000139777
11	-6.39564 e-005	0.000111859
12	-5.11870e-005	8.95168e-005

Responses to a one-standard error shock in d_P

period	d_G	d_P
1	0.000000	0.00440523
2	-0.00144030	0.00364231
3	-0.00151998	0.00294463
4	-0.00131008	0.00236408
5	-0.00107230	0.00189382
6	-0.000864213	0.00151604
7	-0.000693149	0.00121335
8	-0.000555095	0.000971026
9	-0.000444321	0.000777080
10	-0.000355598	0.000621867
11	-0.000284577	0.000497655
12	-0.000227737	0.000398253



Figure 13.7: Select Graphs>Impulse responses (combined) from the VAR results window.

13.3 Script

```
#VECM example
open "c:\Program Files\gretl\data\poe\gdp.gdt"
#Declare the data to be time series
setobs 4 1970:1 --time-series
#Analyze the plots for constants and trends
scatters usa diff(usa) aus diff(aus)
#Testing down with ADF
adf 6 usa --ctt --test-down
adf 6 aus --ctt --test-down
adf 6 usa --difference --ct --test-down
adf 6 aus --difference --ct --test-down
#Testing up (manually for usa)
genr time
genr t2 = time*time
genr d_usa = diff(usa)
genr d_aus = diff(aus)
ols diff(usa) usa(-1) const time t2
```



Figure 13.8: U.S. $\ln(\text{GDP})$ and $\ln(\text{CPI})$ impulse responses

```
modtest 1 --autocorr
ols diff(usa) usa(-1) lags(1,d_usa) const time t2 --quiet
modtest 1 --autocorr
ols diff(usa) usa(-1) lags(2,d_usa) const time t2 --quiet
modtest 1 --autocorr
#This test can be misleading: not enough AR terms in LM test
ols diff(aus) aus(-1) const time t2
modtest 1 --autocorr
#Be sure to test for enough AR terms in the LM test!
ols diff(aus) aus(-1) const time t2
modtest 3 --autocorr
ols diff(aus) aus(-1) lags(1,d_aus) const time t2 --quiet
modtest 3 --autocorr
ols diff(aus) aus(-1) lags(2,d_aus) const time t2 --quiet
modtest 3 --autocorr
ols diff(aus) aus(-1) lags(3,d_aus) const time t2 --quiet
modtest 3 --autocorr
#Cointegration test
ols aus usa
genr uhat = $uhat
ols diff(uhat) uhat(-1)
adf 0 uhat --nc
```

Decomposition of variance for d_G

period	std. error	d_G	d_P
1	0.00778221	100.0000	0.0000
2	0.00808683	96.8279	3.1721
3	0.00823008	93.5265	6.4735
4	0.00833534	91.2187	8.7813
5	0.0084066	89.7399	10.2601
6	0.00845295	88.8068	11.1932
7	0.00848272	88.2175	11.7825
8	0.00850177	87.8440	12.1560
9	0.00851395	87.6064	12.3936
10	0.00852175	87.4550	12.5450
11	0.00852674	87.3582	12.6418
12	0.00852994	87.2964	12.7036

Decomposition of variance for d_P

period	std. error	d_G	d_P
1	0.00441975	0.6563	99.3437
2	0.0057648	1.6865	98.3135
3	0.00650301	2.2365	97.7635
4	0.00693897	2.5278	97.4722
5	0.00720519	2.6891	97.3109
6	0.00737081	2.7825	97.2175
7	0.00747498	2.8385	97.1615
8	0.00754094	2.8728	97.1272
9	0.00758289	2.8941	97.1059
10	0.00760963	2.9076	97.0924
11	0.0076267	2.9161	97.0839
12	0.00763762	2.9215	97.0785

```
#VECM
ols diff(aus) const uhat(-1)
ols diff(usa) const uhat(-1)
#Growth example
open "c:\Program Files\gretl\data\poe\growth.gdt"
#Analyze the plots
scatters G diff(G) P diff(P)
#Cointegration test
ols G const P
series uhat = $uhat
ols diff(uhat) uhat(-1)
adf 3 uhat --nc --test-down --verbose
adf 0 uhat --nc --verbose
#VAR
var 1 diff(P) diff(G)
#Using lagselect
var 8 diff(G) diff(P) --lagselect
#Estimate the VAR with IRFs and FEVDs
var 1 diff(P) diff(G) --impulse-responses --variance-decomp
```

Chapter 14

Time-Varying Volatility and ARCH Models: Introduction to Financial Econometrics

In this chapter we'll estimate several models in which the variance of the dependent variable changes over time. These are broadly referred to as ARCH (autoregressive conditional heteroskedas-ticity) models and there are many variations upon the theme.

14.1 ARCH and GARCH

The basic ARCH(1) model can be expressed as:

$$y_t = \beta + e_t \tag{14.1}$$

$$e_t | I_{t-1} \sim N(0, h_t)$$
 (14.2)

$$h_t = \alpha_0 + \alpha_1 e_{t-1}^2 \qquad \alpha_0 > 0, \qquad 0 \le \alpha_1 < 1 \tag{14.3}$$

The first equation describes the behavior of the mean of your time series. In this case, equation (14.1) indicates that we expect the time series to vary randomly about its mean, β . If the mean of your time series drifts over time or is explained by other variables, you'd add them to this equation just as you would a regular regression model. The second equation indicates that the error of the regression, e_t , are normally distributed and heteroskedastic. The variance of the current period's error depends on information that is revealed in the preceding period, i.e., I_{t-1} . The variance of e_t is given the symbol h_t . The final equation describes how the variance behaves. Notice that h_t depends on the error in the preceding time period. The parameters in this equation have to be positive to ensure that the variance, h_t , is positive.

The ARCH(1) model can be extended to include more lags of the errors, e_{t-q} . In this case, q refers to the order of the ARCH model. For example, ARCH(2) replaces (14.3) with $h_t =$

 $\alpha_0 + \alpha_1 e_{t-1}^2 + \alpha_2 e_{t-2}^2$. When estimating regression models that have ARCH errors in **gretl**, you'll have to specify this order.

ARCH is treated as a special case of a more general model in **gretl** called GARCH. GARCH stands for generalized autoregressive conditional heteroskedasticity and it adds lagged values of the variance itself, h_{t-p} , to (14.3). The GARCH(1,1) model is:

$$y_t = \beta + e_t \tag{14.4}$$

$$e_t | I_{t-1} \sim N(0, h_t)$$
 (14.5)

$$h_t = \delta + \alpha_1 e_{t-1}^2 + \beta_1 h_{t-1} \tag{14.6}$$

The difference between ARCH (14.3) and its generalization (14.6) is a term $\beta_1 h_{t-1}$, a function of the lagged variance. In higher order GARCH(p,q) model's, q refers to the number of lags of e_t and p refers to the number of lags of h_t to include in the model of the regression's variance.

To open the dialog for estimating ARCH and GARCH in **gretl** choose Model>Time series>GARCH from the main **gretl** window as shown in Figure 14.1 below.¹

·			0
🦉 greti			
<u>File T</u> ools <u>D</u> ata	<u>Y</u> iew <u>A</u> dd <u>S</u> ample <u>V</u> ariable	Model	Help
BYD.gdt		Ordinary Least Squares	
ID # Variable name	Descriptive label	Other linear models	
0 const 1 r	auto-generated constant returns to shares in BrightenYourD	Time series Panel Nonlinear models Bobust estimation Maximum likelihood GMM Simultaneous equations	Cochrane-Orcutt Hildreth-Lu Prais-Winsten Autoregressive estimation ARIMA GARCH Vector Autoregression VAR lag selection VECM Cointegration test
	Time series: F	Full range 1 - 500	

Figure 14.1: Choose Model>Time series>GARCH from the main gretl window.

To estimate the ARCH(1) model, you'll place the time series \mathbf{r} into the dependent variable box

¹In a later version of **gretl**, an ARCH option has been added. You can use this as well, but the answer you get will be slightly different due to differences in the method used to estimate the model.
and set q=1 and p=0 as shown in Figure (14.2) This yields the results:

rigure 14.2. Estimating M	ton using the dialog box in green
gretl: specify model	
	GARCH
const	Dependent variable
r	Choose -> r
г	Set as default
	Independent variables
	Add ->
	<- Remove
ARCH p: 🏚 🚔 ARCH o	q: 1
Robust standard errors con	ifigure
Show details of iterations	
lags	
🔯 <u>H</u> elp 🛛 💩 ⊆lear	Cancel

Figure 14.2: Estimating ARCH using the dialog box in **gret**

$$\begin{split} \hat{r} &= \underset{(26.886)}{1.06394} \\ \hat{h}_t &= 0.642139 + 0.569347 \, e_{t-1}^2 \\ T &= 500 \quad \ln L = -740.7932 \quad \hat{\sigma} = 1.2211 \\ & (t\text{-statistics in parentheses}) \end{split}$$

To estimate the GARCH(1,1) model, set p=1 and q=1 to obtain:

$$\hat{\mathbf{r}} = \underbrace{1.04987}_{(0.040465)}$$

$$\hat{\sigma}_{t}^{2} = \underbrace{0.40105}_{(0.089941)} + \underbrace{0.491028}_{(0.10157)} \varepsilon_{t-1}^{2} + \underbrace{0.237999}_{(0.1115)} \sigma_{t-1}^{2}$$

$$T = 500 \quad \ln L = -736.0281 \quad \hat{\sigma} = 1.2166$$
(standard errors in parentheses)

You will notice that the coefficient estimates and standard errors for the ARCH(1) and GARCH(1,1) models are quite close to those in Chapter 14 of your textbook. To obtain these, you will have to

change the default variance-covariance computation using set garch_vcv op before running the script. Although this gets you close the the results in *POE*, using the garch_vcv op is not usually recommended; just use the gretl default, set garch_vcv unset.

The standard errors and t-ratios often vary a bit, depending on which software and numerical techniques are used. This is the nature of maximum likelihood estimation of the model's parameters. With maximum likelihood estimation the model's parameters are estimated using numerical optimization techniques. All of the techniques usually get you to the same parameter estimates, i.e., those that maximize the likelihood function; but, they do so in different ways. Each numerical algorithm arrives at the solution iteratively based on reasonable starting values and the method used to measure the curvature of the likelihood function at each round of estimates. Once the algorithm finds the maximum of the function, the curvature measure is reused as an estimate of the variance covariance matrix. Since curvature can be measured in slightly different ways, the routine will produce slightly different estimates of standard errors.

Gretl gives you a way to choose which method you like use for estimating the variancecovariance matrix. And, as expected, this choice will produce different standard errors and t-ratios. The set garch_vcv command allows you to choose among five alternatives: unset-which restores the default, hessian, im (information matrix), op (outer product matrix), qml (QML estimator), or bw (Bollerslev-Wooldridge). If unset is given the default is restored, which in this case is the Hessian; if the "robust" option is given for the garch command, QML is used.

14.2 Testing for ARCH

Testing for the presence of ARCH in the errors of your model is straightforward. In fact, there are at least two ways to proceed. The first is to estimate the regression portion of your model using least squares. Then choose the Tests>ARCH from the model's pull-down menu. This is illustrated in Figure 14.3 below.

This brings up the box where you tell **gretl** what order of ARCH(q) you want as your alternative hypothesis. In the example, q = 1 which leads to the result obtained in the text. The output is shown below in Figure 14.5. **Gretl** produces the LM statistic discussed in your text; the relevant part is highlighted in red.

The other way to conduct this test is manually. The first step is to estimate the regression (14.1) using **gretl**. Save the squared residuals and then regress these on their lagged value. Take TR^2 from this regression as your test statistic. The script for this appears below:

open "c:\Program Files\gretl\data\poe\BYD.gdt"

```
ols r const
series ehat = $uhat
series ehat2 = ehat*ehat
```

📓 gretl: ma	del 4					
Eile Edit	Tests Save Graphs	Analysis LaTe	x			
Model 4 Depender	Omit variables Add variables Sum of coefficients	ing the S	500 observations 1-	500		-
v.	Von-linearity (squares)	FICIENT	STDERROR	T STAT	P-VALUE	
const	Non-linearity (logs) <u>R</u> amsey's RESET	.07829	0.0529959	20.347	<0.00001	***
Mean (Stands Sum o: Stands Unadju Adjust	Heteroskedasticity Mormality of residual Influential observations Collinearity Autocorrelation	ble = 1.0 lep. var. s = 700.7 luals = 1. 0	07829 = 1.18502 737 18502			
Degree Durbin First- Log-1: Akaika Schwan Hannan	ABCH Chow test Course of the st Cursum Sector and the st Cursum Sector	9 = 1.9189 tion coef 51 erion (A) ion (BIC) (HQC) = 1	97 Ef. = 0.0377057 IC) = 1589.7 I = 1593.92 IS91.35			_
			Close			

Figure 14.3: Choose Tests>ARCH from the model's pull-down menu.

Figure 14.4: Testing ARCH dialog box



Figure 14.5: ARCH test results

🦉 gretl: ARCH test					
🔏 占 🛱 🔍 🗙					
Test for ARCH of OLS estimates usi Dependent variabl	order 1 ng the 499 observat: e: ut^2	ions 2-500			
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE	
const ut ² (-1)	0.908262 0.353071 ľ	0.124401 0.0419848	7.301 8.410	<0.00001 <0.00001	***
No of obs. = 499, LM test statistic Area to the right ARCH effect is si	unadjusted R ² = 0. (62.159501) is dist of LM = 0.000000 gnificant at the 10	.124568 tributed as Chi-: percent level.	square (1)	
WLS (ARCH) estima Dependent variabl Variable used as	tes using the 499 ok e: r_byd weight: 1/sigma	oservations 2-50	D		
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE	
const	1.03259	0.0466978	22.112	<0.00001	***
		Close			

ols ehat2 const ehat2(-1) scalar tr2 = \$trsq

The first line estimates the regression

$$r_t = \beta + e_t \tag{14.7}$$

The residuals are saved in ehat and then squared as ehat2. The next line estimates the regression

$$\hat{e}_t = \alpha_1 + \alpha_2 \hat{e}_{t-1} + u_t \tag{14.8}$$

The notation ehat2(-1) takes the variable ehat2 and offsets it in the dataset by the amount in parentheses. In this case, ehat2(-1) puts a minus one period lag of ehat2 into your regression. The final line computes TR^2 from the regression.

14.3 Simple Graphs

There are several figures that you can produce with **gret1** and **gnuplot**. One useful graph is a histogram of the time series you are studying. The easiest way to get this is through the pull-down menus. In Figure 14.6 you'll find a histogram of the Brighten Your Day returns. A normal density is superimposed on the series. Selecting Variable>Frequency plot>against Normal from the pull-down menu opens a small dialog box that allows you to control how the histogram looks. You can choose the number of bins, which in this case has been set to 23 (Figure 14.7). Click OK and the result appears in Figure 14.8.

Once you've estimated your ARCH or GARCH model, you can graph the behavior of the variance as done in the textbook. After estimating ARCH or GARCH, you can save the predicted variances using the command genr ht = \$h. Then plot them using gnuplot ht time. The result is shown in Figure 14.9. A prettier plot can be obtained using the pull-down menus or by editing the plot yourself. To modify the graph, right click on the graph and choose edit. From here you can add labels, titles or replace the crosses with lines. That's what I have done to produce the result in Figure 14.10.

14.4 Threshold ARCH

Threshold ARCH (TARCH) can also be estimated in **gretl**, though it requires a little programming; there aren't any pull-down menus for this estimator. Instead, we'll introduce **gretl**'s powerful **mle** command that allows user defined (log) likelihood functions to be maximized.

The threshold ARCH model replaces the variance equation (14.3) with

$$h_t = \delta + \alpha_1 e_{t-1}^2 + \gamma d_{t-1} e_{t-1}^2 + \beta_1 h_{t-1}$$
(14.9)



Figure 14.6: Highlight the desired series using your cursor, then choose Variable>Frequency plot>against Normal from the pull-down menu

Figure 14.7: Choosing Variable>Frequency plot>against Normal from the pull-down menu reveals this dialog box.





Figure 14.8: The histogram produced using the dialogs from the pull-down menu in gretl.

Figure 14.9: Plot of the variances after estimating the GARCH(1,1) using the BrightenYourDay returns. Right click on the graph to bring up the menu shown. Then choose edit to modify your graph.





Figure 14.10: Plot of the variances after estimating the GARCH(1,1) using Brighten Your Day's returns

$$d_t = \begin{cases} 1 & if \ e_t < 0 \\ 0 & otherwise \end{cases}$$
(14.10)

The model's parameters are estimated by finding the values that maximize its likelihood. Maximum likelihood estimators are discussed in appendix C of Hill et al. (2007).

Gretl provides a fairly easy way to estimate via maximum likelihood that can be used for a wide range of estimation problems (see Chapter 16 for other examples). To use gretl's mle command, you must specify the log-likelihood function that is to be maximized. Any parameters contained in the function must be given reasonable starting values for the routine to work properly. Parameters can be declared and given starting values (using the genr command).

Numerical optimization routines use the partial derivatives of the objective function to iteratively find the minimum or maximum of the function. If you want, you can specify the analytical derivatives of the log-likelihood function with respect to each of the parameters in **gretl**; if analytical derivatives are not supplied, **gretl** tries to compute a numerical approximation. The actual results you obtain will depend on many things, including whether analytical derivatives are used and the starting values.

For the threshold GARCH model, open a new script file and type in the program that appears in Figure 14.11.

Figure 14.11: Threshold GARCH script

```
open "c:\Program Files\gretl\data\poe\BYD.gdt"
scalar mu = 0.5
scalar omega = .5
scalar alpha = 0.4
scalar delta = 0.1
scalar beta = 0
mle ll = -0.5*(log(h) + (e^2)/h)
series h = var(r)
series e = r - mu
series e2 = e^2
series e2m = e2 * (e<0)
series h = omega + alpha*e2(-1) + delta*e2m(-1) + beta*h(-1)
params mu omega alpha delta beta
end mle</pre>
```

The first few lines of the script gives starting values for the parameters. The second part of the script contains the the algebraic expression of the likelihood function. The first line $11 = -0.5*(\log(h) + (e^2)/h)$ is what is called the *kernel* of the normal probability density function. Recall that the errors of the ARCH model are assumed to be normally distributed and this is reflected in the kernel. Next, we have to specify an initial guess for the variances of the model, and these are set using var(r). Then, the errors are generated, squared, and the threshold term is created using series $e^{2m} = e^2 * (e^{<0})$; the expression (e<0) takes the value of 1 for negative errors, e, and is zero otherwise. Then, the heteroskedastic function h_t is specified. The parameters of the model are given at the end, which also tells gret1 to print the estimates out once it has finished the numerical optimization. The mle loop is ended with end mle. The output appears in Figure 14.12. The coefficient estimates are very close to those printed in your text, but the standard errors and corresponding t-ratios are quite a bit different. This is not that unusual since different pieces of software that no doubt use different algorithms were used to numerically maximize the log-likelihood function.

14.5 Garch-in-Mean

The Garch-in-mean (MGARCH) model adds the equation's variance to the regression function. This allows the average value of the dependent variable to depend on volatility of the underlying asset. In this way, more risk (volatility) can lead to higher average return. The equations are listed

Figure 14.12: Threshold ARCH results

📓 gretl: script output					_		
	🥎 🗙						
<pre>? params mu omega alpha delta beta ? end mle Using numerical derivatives Tolerance = 1.81899e-012 Function evaluations: 62 Evaluations of gradient: 19 Model 1: ML estimates using the 500 observations 1-500 11 = -0.5*(log(h) + (e²2)/h) Standard errors based on Outer Products matrix</pre>							
PARAMETER	ESTIMATE	STDERROR	T STAT	P-VALUE			
mu	0.995256	0.0429408	23.177	<0.00001	* * *		
omega	0.356063	0.0900963	3.952	0.00008	***		
alpha	0.263282	0.0805990	3.267	0.00109	* * *		
delta	0.490534	0.204458	2.399	0.01643	* *		
beta	0.286870	0.115478	2.484	0.01298	* *		
Log-likelihood = -271.116 Akaike information criterion (AIC) = 552.232 Schwarz Bayesian criterion (BIC) = 573.305 Hannan-Quinn criterion (HQC) = 560.501							
1							

below:

$$y_t = \beta_0 + \theta h_t + e_t \tag{14.11}$$

$$h_t = \delta + \alpha_1 e_{t-1}^2 + \gamma d_{t-1} e_{t-1}^2 + \beta_1 h_{t-1}$$
(14.12)

Notice that in this formulation we left the threshold term in the model. The errors are normally distributed with zero mean and variance h_t .

The parameters of this model can be estimated using **gretl**, though the recursive nature of the likelihood function makes it a bit more difficult. In the script below (Figure 14.13) you will notice that we've defined a function to compute the log-likelihood.² The function is called **gim_filter** and it contains eight arguments. The first argument is the time series, y. Then, each of the parameters is listed (mu, theta, delta, alpha, gam, and beta) as scalars. The final argument is a placeholder for the variance, h, that is computed within the function.

Once the function is named and its arguments defined, you need to initiate series for the variances and the errors; these have been called 1h and 1e, respectively. The log-likelihood function is computed using a loop that runs from the second observation through the last. The length of the series can be obtained using the saved result **\$nobs**, which is assigned to the variable **T**.

Gretl's loop syntax is fairly simple, though there are several variations. In this example the loop is controlled using the special index variable, i. In this case you specify starting and ending values for i, which is incremented by one each time round the loop. In the TGARCH example the loop syntax looks like this:

²Actually, **gretl** genius Professor 'Jack' Lucchetti wrote the function and I'm very grateful!

```
loop for i=2..T --quiet
.
.
.
end loop
```

The first line start the loop using an index variable named i. The first value of i is set to 2. The index i will increment by 1 until it reaches T, which has already been defined as being equal to **\$nobs**. The **end loop** statement tells **gret1** the point at which to return to the top of the loop and advance the increment i. The **--quiet** option just reduces the amount of output that is written to the screen.

Within the loop itself, you'll want to lag the index and create an indicator variable that will take the value of 1 when the news is bad $(e_{t-1} < 0)$. The next line squares the residual. lh[i] uses the loop index to place the variance computation from the iteration into the i^{th} row of lh. The line that begins le[i] = works similarly for the errors of the mean equation.

The variances are collected in **h** and the residuals in **le**, the latter of which is returned when the function is called. The function is closed using end function.

If this looks too complicated, you can simply highlight the code with your cursor, copy it using Ctrl-C, and paste it into a **gretl** script file (or use the scripts provided with this book).

Once the function is defined, you need to initialize each parameter just as you did in TGARCH. The series that will eventually hold the variances also must be initialized. The latter is done using **series** h = NA, which creates the series h and fills it with missing values (NA). The missing values for observations 2 through T are replaced as the function loops.

Next, the built-in mle command is issued and the normal density kernel is specified just as it was in the TGARCH example. Then, use the predefined e=gim_filter() function, putting in the variable r for the time series, the initialized parameters, and &h as a pointer to the variances that will be computed within the function. Issue the params statement to identify the parameters and have them print to the screen. Close the loop and run the script. The results appear in Figure 14.14 below. This is a difficult likelihood to maximize and gret1 may take some time to compute the estimates. Still, it is quite remarkable that we get so close using a free piece of software and the numerical derivatives that it computes for us. I'm impressed!

Figure 14.13: The MGARCH script includes a function to compute the log-likelihood.

```
function gim_filter(series y, \
        scalar mu, scalar theta, scalar delta, scalar alpha, \setminus
        scalar gam, scalar beta, series *h)
    series lh = var(y)
    series le = y - mu
    scalar T = nobs
    loop for i=2..T --quiet
        scalar ilag = $i - 1
        scalar d = (le[ilag]<0)</pre>
        scalar e2lag = le[ilag]^2
        lh[i] = delta + alpha*e2lag + gam*e2lag*d + beta*lh[ilag]
        le[i] = le[i] - theta*lh[i]
    end loop
    series h = lh
    return series le
end function
open "c:\Program Files\gretl\data\poe\BYD.gdt"
scalar mu = 0.8
scalar gam = .1
scalar alpha = 0.4
scalar beta = 0
scalar delta = .5
scalar theta = 0.1
series h = NA
mle ll = -0.5*(\log(2*pi) + \log(h) + (e^2)/h)
   e = gim_filter(r, mu, theta, delta, alpha, gam, beta, &h)
   params mu theta delta alpha gam beta
end mle --robust
```

Figure 14.14: Garch-in-mean results

📓 gretl: script output					_	
84669						
? end mlerobust						- ^
Using numerical der	ivatives					
Tolerance = 1.81899	e-012					
Function evaluation	s: 104					
Evaluations of grad	ient: 24					
Model 1: ML estimat	es using the 5	UU observations 1-5	500			
II = -U.S*(log(2*p1) + log(n) + (e''2)/n)				
QML Scandard errors						
PARAMETER	ESTIMATE	STDERROR	T STAT	P-VALUE		
mu	0.814457	0.0677442	12.023	<0.00001	* * *	
theta	0.200801	0.0610628	3.288	0.00101	* * *	
delta	0.370791	0.0658594	5.630	<0.00001	* * *	
alpha	0.296684	0.0735616	4.033	0.00006	* * *	
gam	0.313666	0.128515	2.441	0.01466	* *	
beta	0.279000	0.0544586	5.123	<0.00001	* * *	=
Log-likelihood =	-724.461	7.00 4.4.50 00				
Akaike informatio	n criterion (A	(10) = 1460.92				
Schwarz Bayeslan	criterion (BIC) = 1486.21 1470 84				
hannan-Quinn erit	erion (NQC) =	14/0.04				
<		1111				>

14.6 Script

open "c:\Program Files\gretl\data\poe\BYD.gdt"

```
# ARCH(1) Using built in command for ARCH
arch 1 r const
# GARCH(0,1)=ARCH(1)
garch 0 1 ; r const
# GARCH(1,1)
garch 1 1 ; r const
#LM test for ARCH
ols r const
modtest 1 --arch
#LM test manually
ols r const
series ehat = $uhat
series ehat2 = ehat*ehat
ols ehat2 const ehat2(-1)
scalar tr2 = $trsq
#Plotting
```

```
garch 1 1 ; r const
genr ht = 
gnuplot ht time
#Threshold Garch
open "c:\Program Files\gretl\data\poe\BYD.gdt"
scalar mu = 0.5
scalar omega = .5
scalar alpha = 0.4
scalar delta = 0.1
scalar beta = 0
mle ll = -0.5*(\log(h) + (e^2)/h)
   series h = var(r)
   series e = r - mu
   series e^2 = e^2
   series e^2m = e^2 * (e^{0})
   series h = \text{omega} + \text{alpha}*e2(-1) + \text{delta}*e2m(-1) + \text{beta}*h(-1)
   params mu omega alpha delta beta
end mle
#Garch in Mean
function gim_filter(series y, \
        scalar mu, scalar theta, scalar delta, scalar alpha, \setminus
        scalar gam, scalar beta, series *h)
    series lh = var(y)
    series le = y - mu
    scalar T = snobs
    loop for i=2..T --quiet
        scalar ilag = $i - 1
        scalar d = (le[ilag]<0)</pre>
        scalar e2lag = le[ilag]^2
        lh[i] = delta + alpha*e2lag + gam*e2lag*d + beta*lh[ilag]
        le[i] = le[i] - theta*lh[i]
    end loop
    series h = lh
    return series le
end function
open "c:\Program Files\gretl\data\poe\BYD.gdt"
```

```
scalar mu = 0.8
scalar gam = .1
scalar alpha = 0.4
scalar beta = 0
scalar delta = .5
scalar theta = 0.1
series h = NA
mle ll = -0.5*(log(2*pi) + log(h) + (e^2)/h)
        e = gim_filter(r, mu, theta, delta, alpha, gam, beta, &h)
        params mu theta delta alpha gam beta
end mle --robust
```

Chapter 15

Pooling Time-Series and Cross-Sectional Data

A panel of data consists of a group of cross-sectional units (people, firms, states or countries) that are observed over time. Following Hill et al. (2007) we will denote the number of cross-sectional units by N and the number of time periods we observe them as T.

Gretl gives you easy access to a number of useful panel data sets via its database server.¹ These include the Penn World Table and the Barro and Lee (1996) data on international educational attainment. These data can be installed using File>Databases>On database server from the menu bar as shown in Figure 15.1 below. From here, select a database you want. In Figure 15.2

Eile	<u>T</u> ools <u>D</u> ata	<u>V</u> iew <u>A</u> dd	Sample Variable Model	1
	<u>O</u> pen data	•		
	Append data		label	
	<u>S</u> ave data	Ctrl+S		
	Save data <u>a</u> s			
	Export data	÷.		
	Send To			
	<u>N</u> ew data set	Ctrl+N		
8	Clear data set			
	Seript files	ŀ		
	Databases	•	Cret pative	
	_ Eunction files	•		
4	Evit	Ctrl+X	EAIS 4	
	E <u>A</u> R	Carra	CGive	
			On database server	
		L		
·				

Figure 15.1: Accessing data from the database server via the pull-down menus

¹Your computer must have access to the internet to use this.

the entry for the Penn World Table is highlighted. To its right, you are given information about whether that dataset is installed on your computer. Double click on pwtna and a listing of the series in this database will appear in a new window. From that window you can search for a particular series, display observations, graph a series, or import it. This is a VERY useful utility, both for teaching and research and I encourage you to explore what is available on the **gretl** server.

🔛 gi	📓 gretl: databases on server 📃 🗌 🗋						
R		×					
_							
Data	abase	Source	Local status				
Ь	arro_lee	Barro - Lee panel of 138 countries	Not installed				
ы	Ь	UK National Statistics (Blue Book)	Not installed				
Ь	can	Bank of Canada (money, credit)	Not installed				
Ь	cih	Dept of Commerce (Business Cycle Indicators 1945-1995)	Not installed				
Ð		Banco de Espana					
Ь	eana	Bureau of Economic Analysis (US national accounts)	Not installed				
Ь	eapira	Bureau of Economic Analysis (Income and Population Data)	Not installed				
e	t	UK National Statistics (Economics Trends)	Not installed				
e	tas	UK National Statistics (Economic Trends Annual Supplement)	Not installed				
fe	edbog	Federal Reserve Board (interest rates)	Not installed				
fe	edstl	St Louis Fed (various series, large)	Not up to date				
ft ft	nfb	Federal Housing Finance Board (mortgages)	Not installed				
9	dpo	UK National Statistics (GDP statistics, output)	Not installed				
ja	apan	Bank of Japan (macro, monetary data)	Not installed				
k k	s13	eh.net 19th century labor survey	Not installed				
k k	s14	eh.net 19th century labor survey	Not installed				
Ð		NBER macro historical data					
р	rof	UK National Statistics Corporate Profits	Not installed				
р P	wt61	Penn World Table (version 6.1)	Not installed				
P	wtna	Penn World Table (version 6.1 National Accounts)	Not installed				
st	p	Standard and Poors (US stock price indices)	Not installed				
) Notwo	ork status	. OK					
Netwo	JIN SCOLUS						

Figure 15.2: Installing a data from the database server via the pull-down menus

15.1 A Basic Model

The most general expression of linear regression models that have both time and unit dimensions is seen in equation 15.1 below.

$$y_{it} = \beta_{1it} + \beta_{2it} x_{2it} + \beta_{3it} x_{3it} + e_{it} \tag{15.1}$$

where i = 1, 2, ..., N and t = 1, 2, ..., T. If we have a full set of time observations for every individual then there will be NT total observations in the sample. The panel is said to be **balanced** in this case. It is not unusual to have some missing time observations for one or more individuals. When this happens, the total number of observation is less than NT and the panel is **unbalanced**.

The biggest problem with equation (15.1) is that even if the panel is complete (balanced), the model contains 3 times as many parameters as observations (NT)! To be able to estimate the model, some assumptions have to be made in order to reduce the number of parameters. One of the most

common assumptions is that the slopes are constant for each individual and every time period; also, the intercepts vary only by individual. This model is shown in equation (15.2).

$$y_{it} = \beta_{1i} + \beta_2 x_{2it} + \beta_3 x_{3it} + e_{it} \tag{15.2}$$

This specification, which includes N + 2 parameters, includes dummy variables that allow the intercept to shift for each individual. By using such a model you are saying that over short time periods there are no substantive changes in the regression function. Obviously, the longer your time dimension, the more likely this assumption will be false.

In equation (15.2) the parameters that vary by individual are called **individual fixed effects** and the model is referred to as **one-way fixed effects**. The model is suitable when the individuals in the sample differ from one another in a way that does not vary over time. It is a useful way to avoid unobserved differences among the individuals in your sample that would otherwise have to be omitted from consideration. Remember, omitting relevant variables may cause least squares to be biased and inconsistent; a one-way fixed effects model, which requires the use of panel data, can be very useful in mitigating the bias associated with time invariant, unobservable effects.

If you have a longer panel and are concerned that the regression function is shifting over time, you can add T-1 time dummy variables to the model. The model becomes

$$y_{it} = \beta_{1i} + \beta_{1t} + \beta_2 x_{2it} + \beta_3 x_{3it} + e_{it} \tag{15.3}$$

where either β_{1i} or β_{1t} have to be omitted in order to avoid perfect collinearity. This model contains N + (T - 1) + 2 parameters which is generally fewer than the NT observations in the sample. Equation (15.3) is called the **two-way fixed effects** model because it contains parameters that will be estimated for each individual and each time period.

15.2 Estimation

Estimating models using panel data is straightforward in **gretl**. There are several built in functions to estimate fixed effects, random effects, and seemingly related regression models. In this section the **gretl** commands for each will be discussed using the examples in Hill et al. (2007).

In order to use the predefined procedures for estimating models using panel data in **gretl** you have to first make sure that your data have been properly structured in the program. The dialog boxes for assigning dataset structure in **gretl** are shown in Figures 7.2 and 7.3. The data have to include variables that identify each individual and time period. Select the **Panel** option using the radio button and **gretl** will then be able to work behind the scenes to accurately account for the time and individual dimensions. The datasets that come with this manual have already been setup this way, but if you are using your own data you'll want to to assign the proper dataset structure to it so that all of the appropriate panel data procedures are available for use.

Now consider the investment model suggested by Grunfeld (1958). Considering investment

decisions of only two firms, General Electric (GE) and Westinghouse (W), we have

$$INV_{GE,t} = \beta_{1,GE} + \beta_{2,GE}V_{GE,t} + \beta_{3,GE}K_{GE,t} + e_{GE,t}$$
(15.4)

$$INV_{W,t} = \beta_{1,W} + \beta_{2,W}V_{W,t} + \beta_{3,W}K_{W,t} + e_{W,t}$$
(15.5)

where t = 1, 2, ..., 20.

How one proceeds at this point depends on the nature of the two firms and the behavior of all the omitted factors affecting investment. There are a number of modeling options and POE suggests several tests to explore whether the modeling decision we make is an appropriate one. These are considered in the following sections.

15.2.1 Pooled Least Squares

Suppose that that the two firms behave identically and that the other factors influencing investment also have similar effects. In this case, you could simply pool the observations together and estimate a single equation via least squares. This simple model implies that the intercepts and each of the slopes for the two equations are the same and that the omitted factors are not correlated with one another and that they have the same variability. In other words, there is no autocorrelation and the variances are homoscedastic; when the data are actually generated in this way, least squares is efficient.

In terms of the parameters of the model, $\beta_{i,GE} = \beta_{i,W}$ for i = 1, 2, 3; $E[e_{GE,t}] = E[e_{W,t}] = 0$; $Var[e_{GE,t}] = Var[e_{W,t}] = \sigma^2$; $Cov(e_{GE,t}, e_{W,t}) = 0$ for all time periods; and $Cov(e_{i,t}, e_{i,s}) = 0$ for $t \neq s$ for each firm, i = GE, W. It should be clear that in this case,

$$INV_{i,t} = \beta_1 + \beta_1 + \beta_2 V_{i,t} + \beta_3 K_{i,t} + e_{i,t}$$
(15.6)

for observations i = GE, W and t = 1, 2, ..., 10. The **gretl** script for estimating this model using *grunfeld.gdt* is

```
open "c:\Program Files\gretl\data\poe\grunfeld.gdt"
smpl firm = 3 || firm = 8 --restrict
ols Inv const V K
modtest --panel
```

The sample is restricted to firms 3 and 8 in the first line. Note the double vertical lines (||) is the new symbol used to designate 'and'. The results are

Inv =
$$17.87 + 0.015_{(7.02)}$$
 V + 0.144 K
(0.0062) V + (0.0186)
 $T = 40 \quad \bar{R}^2 = 0.7995 \quad F(2,37) = 78.752 \quad \hat{\sigma} = 21.158$
(standard errors in parentheses)

Using the robust option would yield consistent standard errors even if the two firms have different variances. The final line in the script performs a test of the equal variance null hypothesis against the alternative that the variances of the two firms differ $\sigma_{GE,t}^2 = \sigma_{GE}^2 \neq \sigma_W^2 = \sigma_{W,t}^2$. Note, in this test the errors within each group are homoscedastic. The output from this test is

```
Likelihood ratio test for groupwise heteroskedasticity -
Null hypothesis: the units have a common error variance
Test statistic: Chi-square(1) = 13.1346
with p-value = 0.000289899
```

which allows us to reject homoscedasticity in favor of groupwise heteroskedasticity at any reasonable level of significance.

The scenario that leads us to use this model seems unlikely, though. At a minimum the variances of the two conglomerate firms will differ due to their differences in size or diversity. Further, since the two firms share output in at least one industry, omitted factors like macroeconomic or market conditions, might reasonably affect the firms similarly. Finally, there is no reason to believe that the coefficients of the two firms will be similar.

15.2.2 Fixed Effects

In the fixed effects model, the intercepts for each firm (or individual) are allowed to vary, but the slopes for each firm are equal. It is particularly useful when each individual has unique characteristics that are both unmeasurable and constant over time (also known by the fancy sounding phrase, 'unobserved time-invariant heterogeneity'). The general form of this model is found in equation (15.2). The **gretl** command to estimate this model is extremely simple. Once your data set is structured within **gretl** as a panel, the fixed effect model is estimated using the **panel** command as shown below in the script.

```
open "c:\Program Files\gretl\data\poe\grunfeld.gdt"
smpl full
panel Inv const V K
```

The results are:

Model 2:	Fixed-effects estimates using 200 observations
	Dependent variable: Inv

Variable	Coefficient	Std. Error	<i>t</i> -statistic	p-value
V	0.109771	0.0118549	9.2596	0.0000
Κ	0.310644	0.0173704	17.8835	0.0000

Sum of squared residuals	522855.
Standard error of residuals $(\hat{\sigma})$	52.7366
Unadjusted R^2	0.944144
Adjusted \bar{R}^2	0.940876
F(11, 188)	288.893
Durbin–Watson statistic	0.667695
Log-likelihood	-1070.6

Test for differing group intercepts –

Null hypothesis: The groups have a common intercept Test statistic: F(9, 188) = 48.9915with p-value = P(F(9, 188) > 48.9915) = 1.11131e-044

By default, **gretl** will test the hypothesis that the fixed effects are the same for each individual. If you do not reject this hypothesis, then you can estimate the model using pooled least squares as discussed in the previous section. The test statistic has an F(9,188) sampling distribution if the pooled least squares model is the correct one. The computed value is 48.99 and the p-value is less than 5%, therefore we would reject the pooled least squares formulation in favor of the fixed effect model in this example.

In this formulation you are assuming that the errors of your model are homoscedastic within each firm and across firms, and that there is no contemporaneous correlation across firms. Gretl allows you to compute standard errors that are robust to the homoscedasticity assumption. Simply use the --robust option in the panel regression. i.e., panel Inv const V K --robust. This option computes the *cluster* standard errors that are discussed in Chapter 15 of Hill et al. (2007).

15.2.3 Random Effects

Gretl also estimates random effects models using the **panel** command. In the random effects model, the individual firm differences are thought to represent random variation about some average intercept for the individual in the sample. Rather than estimate a separate fixed effect for each firm, you estimate an overall intercept that represents this average. Implicitly, the regression function for the sample firms vary randomly around this average. The variability of the individual effects is captured by a new parameter, σ_u^2 . The larger this parameter is, the more variation you find in the implicit regression functions for the firms.

Once again, the model is based on equation (15.2). The difference is that $\beta_{1i} = \overline{\beta}_1 + u_i$ where u_i represents random variation. The model becomes:

$$y_{it} = \beta_1 + u_i + \beta_2 x_{2it} + \beta_3 x_{3it} + e_{it} \tag{15.7}$$

The new parameter, σ_u^2 , is just the variance of the random effect, u_i . If $\sigma_u^2 = 0$ then the effects are "fixed" and you can use the fixed effects estimator if the effects are indeed different across firms or the pooled estimator if they are not.

To estimate the model, using the Grunfeld data use the script

open "c:\Program Files\gretl\data\poe\grunfeld.gdt" smpl full panel Inv const V K --random-effects

This yields

with p-value = 0.330301

	De	ependent variable: In	nv	
Variable	Coefficient	Std. Error	t-statistic	p-value
const	-57.872	28.8747	-2.0043	0.0464
V	0.109478	0.0104895	5 10.4369	0.0000
Κ	0.308694	0.0171938	8 17.9538	0.0000
	Mean of dependen	t variable	145.907	
	S.D. of dependent	variable	216.886	
	Sum of squared re	siduals	4.28309e + 08	
	Standard error of	residuals $(\hat{\sigma})$	1470.77	
	$\hat{\sigma}_arepsilon^2$		2781.14	
	$\hat{\sigma}_u^2$		7218.23	
	heta		0.861203	
	Akaike information	n criterion	3488.98	
	Schwarz Bayesian	criterion	3498.88	
	Hannan–Quinn cr	iterion	3492.99	
Breusch-Pagar Null hypoth	n test – esis: Variance of the unit	s-specific error $= 0$		
Asymptotic with p-value	test statistic: $\chi_1^2 = 797.7$ e = 1.63899e-175	781		
Hausman test	_			
Null hypoth	esis: GLS estimates are o	consistent		
Asymptotic	test statistic: $\chi_2^2 = 2.215$	55		

Model 3: Random-effects (GLS) estimates using 200 observations Dependent variable: Inv

Gretl tests the null hypothesis $\sigma_u^2 = 0$ against the alternative $\sigma_u^2 > 0$ by default and is referred to as the **Breusch-Pagan test**.

The **Hausman test** is a test of the null hypothesis that the random effects are indeed random. If they are random then they should not be correlated with any of your other regressors. If they are correlated with other regressors, then you should use the fixed effects estimator to obtain consistent parameter estimates of your slopes.

In the Grunfeld data, a p-value less than 5% indicates that the Breusch-Pagan test rejects the hypothesis that the effects are not random (in other words, the effects **are** random). For the Hausman test, the p-value is greater than 5%. The random effects do not appear to be correlated with the regressors and random effects can be used.

15.2.4 SUR

The acronym SUR stands for **seemingly unrelated regression equations**. SUR is another way of estimating panel data models that are long (large T) but not wide (small N). More generally though, it is used to estimate systems of equations that do not necessarily have any parameters in common and are hence unrelated. In the SUR framework, each firm in your sample is parametrically different; each firm has its own regression function, i.e., different intercept and slopes. Firms are not totally unrelated, however. In this model the firms are linked by what is **not** included in the regression rather than by what is. The firms are thus related by unobserved factors and SUR requires us to specify how these omitted factors are linked in the system's error structure.

In the basic SUR model, the errors are assumed to be homoscedastic and linearly independent within each equation, or in our case, each firm. The error of each equation may have its own variance. Most importantly, each equation (firm) is correlated with the others in the same time period. The latter assumption is called **contemporaneous correlation**, and it is this property that sets SUR apart from other models.

In the context of the two firm Grunfeld model in (15.4) this would mean that $Var[e_{GE,t}] = \sigma_{GE}^2$; $Var[e_{W,t}] = \sigma_W^2$; $Cov(e_{GE,t}, e_{W,t}) = \sigma_{GE,W}$ for all time periods; and $Cov(e_{i,t}, e_{i,s}) = 0$ for $t \neq s$ for each firm, i = GE, W. So in the SUR model you essentially have to estimate a variance for each individual and a covariance between each pair of individuals. These are then used to construct a generalized least squares estimator of the equations parameters.

Even though SUR requires a T and an N dimension, it is not specifically a panel technique. This is because the equations in an SUR system may be modeling different behaviors for a single individual rather than the same behavior for several individuals. As mentioned before, it is best used when panels are long and narrow since this gives you more observations to estimate the equations variances and the cross equation covariances. More time observations reduces the sampling variation associated with these estimates, which in turn improves the performance of the feasible generalized least squares estimator. If your panel dataset has a very large number of individuals and only a few years, then FGLS may not perform very well in a statistical sense. In the two firm Grunfeld example, N=2 and T=20 so we needn't worry about this warning too much, although the asymptotic inferences are based on T (and not N) being infinite.

When estimating an SUR model, the data have to be arranged in a slightly different way than in the preceding panel examples. Basically, they need to be arranged as a time series (not a panel) with different firms variables listed separately. Hill et al. (2007) have done this for us in the grunfeld2.gdt data set. The gretl script to estimate the two firm SUR model using this data is

```
open "c:\Program Files\gretl\data\poe\grunfeld2.gdt"
```

system name="Grunfeld"
equation inv_ge const v_ge k_ge
equation inv_we const v_we k_we
end system

estimate "Grunfeld" method=sur

Since SUR is a method of estimating a system of equations (just like you did in chapter 11), the same syntax is used here. It consists of a block of code that starts with the system name="Grunfeld" line. One advantage naming your system is that results are attached to it and you can perform subsequent computations based on them. For instance, with a saved set of equations you can impose restrictions on a single equation in the model or impose restrictions across equations. This is accomplished using the restrict statement.

Following the system name, each equation is put on a separate line. Notice that each equation is identified using equation which is followed by the dependent variable and then the independent variables which include a constant. Close the system block using the end system command. The system is then estimated using the line estimate "Grunfeld" method=sur. Executing this script yields Figure 15.3 below.

The test to determine whether there is sufficient contemporaneous correlation is simple to do from the standard output. Recall from POE that the test is based on the squared correlation

$$r_{GE,W}^2 = \frac{\sigma^2_{GE,W}}{\hat{\sigma}^2_{GE}\hat{\sigma}^2_W} \tag{15.8}$$

A little caution is required here. The squared correlations are supposed to be computed based on the residuals from the *least squares estimator*, not SUR. The "Cross-equation VCV for residuals" in the output in Figure 15.3 is computed based on SUR residuals. So, you'll need to change the estimation method to **ols** and rerun the script to get the right inputs for this statistic. The new script is:

open "c:\Program Files\gretl\data\poe\grunfeld2.gdt"

system name="Grunfeld"
equation inv_ge const v_ge k_ge
equation inv_we const v_we k_we
end system
estimate "Grunfeld" method=ols

Figure 15.3: The results from the two firm model estimated as seemingly unrelated regression equations

🦉 gretl: script output					
🕜 🛆 🖬 🖻 🛛	★ ★				
gretl version 1.6. Current session: 2 ? open c:\userdata	0 007/01/25 16:02 \gret1\data\PoE\gru	nfeld2.gdt			
Read datafile c:\u periodicity: 1, ma observations range	serdata\gretl\data\] xobs: 20, : 1935-1954	PoE\grunfeld2.q	gdt		
Listing 7 variable O) const 1) 5) v_we 6)	s: inv_ge 2) v_ge k_we	3) k_ge	4) inv_0	re	
<pre>? system method=su ? equation inv_ge ? equation inv_we ? end system</pre>	r const v_ge k_ge const v_we k_we				
Equation system, S	eemingly Unrelated D	Regressions			
Equation 1: SUR es Dependent variable	timates using the 20 : inv_ge) observations	1935-1954		
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE	
const v_ge k_ge	-27.7193 0.0383102 0.139036	27.0328 0.0132901 0.0230356	-1.025 2.883 6.036	0.31955 0.01034 ** 0.00001 **	*
Mean of dependen Standard deviati Sum of squared r Standard error o	t variable = 102.29 on of dep. var. = 46 esiduals = 13788.4 f residuals = 26.25	3.5845			
Equation 2: SUR es Dependent variable	timates using the 20 : inv_we) observations	1935-1954		
VARIABLE	COEFFICIENT	STDERROR	T STAT	P-VALUE	
const v_we k_we	-1.25199 0.0576298 0.0639781	6.95635 0.0134110 0.0489010	-0.180 4.297 1.308	0.85930 0.00049 ** 0.20818	*
Mean of dependen Standard deviati Sum of squared r Standard error o	t variable = 42.8913 on of dep. var. = 19 esiduals = 1801.3 f residuals = 9.4903	5 9.1102 26			
Cross-equation VCV (correlations abov	for residuals e the diagonal)				
689.42 190.64	(0.765) 90.065				
log determinant =	10.1562				•
		Close			

and the resulting cross-equation variance covariance for the residuals is

Cross-equation VCV for residuals (correlations above the diagonal)

777.45	(0.729)
207.59	104.31

Then you compute

$$r_{GE,W}^2 = \frac{207.59^2}{(777.45)(104.31)} = 0.729 \tag{15.9}$$

Notice that **gretl** produces this number for you in the upper diagonal of the matrix and places it in parentheses. Using the given computation the test statistic is

$$LM = Tr_{GE,W}^2 \underline{d} \chi_{(1)}^2 \tag{15.10}$$

provided the null hypothesis of no correlation is true. The arithmetic is (20 * 0.729) = 14.58

The restrict command can be used to impose the cross-equation restrictions on a system of equations that has been previously defined and named. The set of restrictions is started with the keyword restrict and terminated with end restrict. Some additional details and examples of how to use the restrict command are given in section 6.1. Each restriction in the set is expressed as an equation. Put the linear combination of parameters to be tested on the left-hand-side of the equality and a numeric value on the right. Parameters are referenced using b[i,j] where *i* refers to the equation number in the system, and *j* the parameter number. So, to equate the intercepts in equations one and two use the statement

$$b[1,1] - b[2,1] = 0 \tag{15.11}$$

The full syntax for testing the full set of cross-equation restrictions

$$\beta_{1,GE} = \beta_{1,W}, \quad \beta_{2,GE} = \beta_{2,W}, \quad \beta_{3,GE} = \beta_{3,W} \tag{15.12}$$

on equation 15.4 is shown in Table 15.1: **Gretl** estimates the two equation SUR subject to the restrictions. Then it computes an F-statistic of the null hypothesis that the restrictions are true versus the alternative that at least one of them is not true. It returns the computed F-statistic and its p-value. A p-value less than the desired level of significance leads to a rejection of the hypothesis.

The **gretl** output from this test procedure is

```
F test for the specified restrictions:
F(3,34) = 2.92224 with p-value 0.0478934
```

which matches the results in the text. At the 5% level of significance, the equality of the two equations is rejected.

Table 15.1: Script for imposing cross-equation restrictions in an SUR model

```
system name="Grunfeld"
equation inv_ge const v_ge k_ge
equation inv_we const v_we k_we
end system
restrict "Grunfeld"
b[1,1]-b[2,1]=0
b[1,2]-b[2,2]=0
b[1,3]-b[2,3]=0
end restrict
```

estimate "Grunfeld" method=sur --geomean

15.3 NLS Example

Hill et al. (2007) provides a subset of National Longitudinal Survey which is conducted by the US Department of Labor. The database includes observations on women, who in 1968, were between the ages of 14 and 24. It then follows them through time, recording various aspects of their lives annually until 1973 and bi-annually afterwards. Our sample consists of 716 women observed in 5 years (1982, 1983, 1985, 1987 and 1988). The panel is balanced and there are 3580 total observations.

Two models are considered in equations (15.13) and (15.14) below.

$$ln(WAGE)_{it} = \beta_{1i} + \beta_2 exper_{it} + \beta_3 exper_{it}^2 + \beta_4 tenure_{it} + \beta_5 tenure_{it}^2 + \beta_6 south_{it} + \beta_7 union_{it} + e_{it} \quad (15.13)$$

$$ln(WAGE)_{it} = \beta_{1i} + \beta_2 exper_{it} + \beta_3 exper_{it}^2 + \beta_4 tenure_{it} + \beta_5 tenure_{it}^2 + \beta_6 south_{it} + \beta_7 union_{it} + \beta_8 black_{it} + \beta_9 educ_{it} + e_{it} \quad (15.14)$$

The first model (15.13) is estimated using fixed effects. Race (black) and education (educ) are added to form the model in (15.14). Since these variables do not change for individuals in the sample, their influences cannot be estimated using fixed effects. So, this equation is estimated using random effects using the script below:

open "c:\Program Files\gretl\data\poe\nels_panel.gdt"
panel lwage const exper exper2 tenure tenure2 south union

panel lwage const exper exper2 tenure tenure2 south union \
 black educ --random-effects

Notice that in the random effects line a backslash follows the variable union. This is the continuation command, which tells **gret1** that the command continues on the next line. The results, in tabular form, are in Table 15.2 below. Wisely, **gret1** has omitted the R^2 for the random effects model. Recall that R^2 is only suitable for linear models estimated using OLS, which is the case for one-way fixed effects.

The complete set of results of random effects estimation is shown in the table 15.3 below. The estimate of $\hat{\sigma}_{\varepsilon} = \sqrt{0.0380681} = 0.1951$. Also, the result of the LM test for the randomness of the individual effects ($\sigma_u^2 > 0$) and the Hausman test of the independence of the random effects from the regressors matches that of your text.

The conclusion from these tests is that even though there is evidence of random effects (LM rejects), the random effects are not independent of the regressors; GLS estimator will be inconsistent and you'll have to use the fixed effects estimator of the smaller model. As a result, you will be unable to determine the effects of education and race on wages.

There is one difference between the **gretl** results and those from *POE*, namely $\hat{\sigma}_u = \sqrt{0.115887} = 0.3404$ from **gretl** is slightly larger than that obtained by Hill et al. (2007) using Stata. This is not too surprising since there are several ways to compute σ_u^2 . The difference apparently has little effect on the computation of the coefficients and standard errors since these are fairly close matches to those in the text.

Table 15.2: Fixed Effects and Random Effects estimates for equations (15.13) and (15.14), respectively.

Model Estimates Dependent variable: lwage

	Fixed Effects	Random Effects
exper	0.04108^{**} (0.006620)	0.04362^{**} (0.006358)
exper2	-0.0004091 (0.0002733)	-0.0005610^{**} (0.0002626)
tenure	0.01391^{**} (0.003278)	0.01415^{**} (0.003167)
tenure2	-0.0008962^{**} (0.0002059)	-0.0007553^{**} (0.0001947)
south	-0.01632 (0.03615)	-0.08181^{**} (0.02241)
union	0.06370^{**} (0.01425)	0.08024^{**} (0.01321)
const		0.5339^{**} (0.07988)
black		-0.1167^{**} (0.03021)
educ		(0.07325^{**}) (0.005331)
$n \over ar{R}^2$	$3580 \\ 0.8236$	3580
ℓ	1173.78	-6999.08

Standard errors in parentheses * indicates significance at the 10 percent level ** indicates significance at the 5 percent level

Table 15.3: Random-effects (GLS) estimates using 3580 observations

Dependent variable: lwage

Variable	Coefficient	Std. Error	t-statistic	p-value
const	0.533929	0.0798828	6.6839	0.0000
exper	0.0436170	0.00635758	6.8606	0.0000
exper2	-0.000560959	0.000262607	-2.1361	0.0327
tenure	0.0141541	0.00316656	4.4699	0.0000
tenure2	-0.000755342	0.000194726	-3.8790	0.0001
south	-0.0818117	0.0224109	-3.6505	0.0003
union	0.0802353	0.0132132	6.0724	0.0000
black	-0.116737	0.0302087	-3.8643	0.0001
educ	0.0732536	0.00533076	13.7417	0.0000
	Mean of dependent variab	ble 1.918	324	
	S.D. of dependent variabl	e 0.464	4607	
	Sum of squared residuals	10460.3		
	Standard error of residual	ls $(\hat{\sigma})$ 1.711	126	
	$\hat{\sigma}_{arepsilon}^2$	0.038	80681	
	$\hat{\sigma}_u^2$	0.115	5887	
	heta	0.743	3683	
	Akaike information criter	ion 14016.2		
	Schwarz Bayesian criterio	n 14071.8		
	Hannan–Quinn criterion	14036.0		
Breusch-Pagar Null hypoth Asymptotic with p-value	test – esis: Variance of the unit-specific e test statistic: $\chi_1^2 = 3859.28$ $\kappa = 0$	rror = 0		
Hausman test Null hypoth Asymptotic with p-value	- esis: GLS estimates are consistent test statistic: $\chi_6^2 = 20.7252$ $\epsilon = 0.00205521$			

15.4 Script

```
open "c:\Program Files\gretl\data\poe\grunfeld.gdt"
smpl firm = 3 || firm = 8 --restrict
ols Inv const V K
modtest --panel
open "c:\Program Files\gretl\data\poe\grunfeld.gdt"
smpl full
panel Inv const V K
open "c:\Program Files\gretl\data\poe\grunfeld.gdt"
smpl full
panel Inv const V K --random-effects
open "c:\Program Files\gretl\data\poe\grunfeld2.gdt"
system name="Grunfeld"
equation inv_ge const v_ge k_ge
equation inv_we const v_we k_we
end system
estimate "Grunfeld" method=sur --geomean
restrict "Grunfeld"
b[1,1]-b[2,1]=0
b[1,2]-b[2,2]=0
b[1,3]-b[2,3]=0
end restrict
estimate "Grunfeld" method=sur --geomean
system name="Grunfeld"
equation inv_ge const v_ge k_ge
equation inv_we const v_we k_we
end system
estimate "Grunfeld" method=ols --geomean
```

Chapter 16

Qualitative and Limited Dependent Variable Models

16.1 Probit

There are many things in economics that cannot be meaningfully quantified. How you vote in an election, whether you go to graduate school, whether you work for pay, or what major you choose has no natural way of being quantified. Each of these expresses a *quality* or *condition* you possess. Models of how these decisions are determined by other variables are called **qualitative choice** or **qualitative variable** models.

In a **binary choice** model, the decision you wish to model has only two possible outcomes. You assign artificial numbers to each outcome so that you can do further analysis. In a binary choice model it is conventional to assign '1' to the variable if it possesses a particular quality or if a condition exists and '0' otherwise. Thus, your dependent variable is

 $y_t = \begin{cases} 1 & \text{if individual t has the quality} \\ 0 & \text{if not.} \end{cases}$

The probit statistical model expresses the probability p that your dependent variable takes the value 1 as a function of your independent variables.

$$P[(y_t|x_t) = 1] = \Phi(\beta_1 + \beta_2 x_t) \tag{16.1}$$

where Φ is the cumulative normal probability distribution (cdf). Estimating this model using maximum likelihood is very simple since the MLE of the probit model is already programmed into **gretl**. The syntax for a script is the same as for linear regression except you use the **probit** command in place of **ols**. The following script estimates how the difference in travel time between bus and auto affects the probability of driving a car. The dependent variable (*auto*) is equal to 1 if travel is by car, and *dtime* is (bus time - auto time).

open "c:\Program Files\gretl\data\poe\transport.gdt"

```
probit auto const dtime
genr p1 = $coeff(const)+$coeff(dtime)*20
genr dt = dnorm(p1)*$coeff(dtime)
genr p2 = cnorm($coeff(const)+$coeff(dtime)*30)
```

The second line computes the predicted value of the index $(\beta_1 + \beta_2 dtime)$ when dtime = 20 using the estimates from the probit MLE. The next line computes the marginal affect on the probability of driving if you increase the difference in travel time by one minute when dtime = 20, i.e., $\phi(\beta_1 + \beta_2 dtime)\beta_2$. The **dnorm** function in **gretl** computes $\phi()$, the normal pdf evaluated at the argument in parentheses. The last line computes the estimated probability of driving, given that it takes 30 minutes longer to ride the bus. This computation requires **cnorm**, which computes the cumulative normal cdf, $\Phi()$.

The results are:

p1 = 0.535545 dt = 0.0103690 p2 = 0.798292

Of course, you can also access the probit estimator from the pull-down menus using Model>Nonlinear models>Probit>Binary. The dialog box (Figure 16.1 looks very similar to the one for linear regression, except it gives you a new option to view the details of the iterations.

Whether you use the script or the dialog box, you will get the following results:

	D	epen	dent va	riable	e: auto			
	Coefficie	ent	Std. Ei	ror	z-stat	Slop	pe*	
const	-0.0644	342	0.39924	14	-0.1614			
dtime	0.0299	989	0.01028	867	2.9163	0.011	9068	
Mean depend	dent var	0.	476190	S.D	. depender	nt var	0.39690	7
McFadden R	2^2	0.	575761	Adj	usted \mathbb{R}^2		0.43813	6
Log-likelihoo	d	-6.	165158	Aka	aike criteri	on	16.3303	2
Schwarz crite	erion	18	3.41936	Har	nan-Quin	n	16.7836	9

Model 1: Probit estimates using the 21 observations 1–21

*Evaluated at the mean

Number of cases 'correctly predicted' = 19 (90.5 percent)

📓 gretl: specify model	_ 🗆 🔀
	Probit
const car_time bus_time dtime auto	Dependent variable Choose -> auto Set as default Independent variables Add -> Add -> Const dtime
 Robust standard errors Show details of iterations Show slopes at mean Show p-values 	
Help Clear	<u>Cancel</u> <u>O</u> K

Figure 16.1: Use Model>Nonlinear models>Probit to open the Probit model's dialog box.

Likelihood ratio test: $\chi^2(1) = 16.734 \ [0.0000]$

Several other statistics are computed. They include a measure of fit (McFadden's pseudo- R^2), the value of $f(\beta' x)$ at mean of independent variables, and a test statistic for the null hypothesis that the coefficients on the independent variables (but not the constant) are jointly zero; this corresponds to the overall F-statistic of regression significance in Chapter 6.

16.2 Multinomial Logit

Starting with version 1.8.1, **Gretl** includes a routing to estimate multinomial logit (MNL) using maximum likelihood. In versions before 1.8.1 the alternatives were either (1) use **gretl**'s maximum likelihood module to estimate your own or (2) use another piece of software! In this section we'll estimate the multinomial logit model using the native **gretl** function and I'll relegate the other methods to a separate (optional) section 16.2.1. The other methods serve as good examples of how to use **gretl**'s scripting language and to use it in conjunction with R.

The first step is to open the nels_small.gdt data

```
open "c:\Program Files\gretl\data\poe\nels_small.gdt"
```

Next consider the model. The dependent variable represents choice of school. We have 1000 observations on students who choose, upon graduating from high school, either no college psechoice=1, a 2-year college psechoice=2, or a 4-year college psechoice=3. The explanatory variable is grades, which is an index ranging from 1.0 (highest level, A+ grade) to 13.0 (lowest level, F grade) and represents combined performance in English, Math and Social Studies. For this example, the choices are treated as being unordered.

To estimate the model of school choice as a function of grades and a constant open the **gretl** console and type:

logit psechoice const grades --multinomial

This yields the output:

Model 1: Multinomial Logit estimates using the 1000 observations 1–1000 Dependent variable: psechoice Standard errors based on Hessian

	Coefficient	Std. E	rror	z-stat	p-va	alue
const grades const grades	$2.50642 \\ -0.308789 \\ 5.76988 \\ -0.706197$	$\begin{array}{c} 0.4183 \\ 0.0522 \\ 0.4043 \\ 0.0529 \end{array}$	85 849 23 246	5.9907 -5.9059 14.2705 -13.3435	0.00 0.00 0.00 0.00	000 000 000 000
Mean depende Log-likelihood Schwarz criter	ent var 2 -8 rion 1	.305000 75.3131 778.257	S.D. Aka Han	. dependent ike criterion nan–Quinn	var	0.810328 1758.626 1766.087

Number of cases 'correctly predicted' = 552 (55.2 percent) Likelihood ratio test: $\chi^2(2) = 286.689 [0.0000]$

It is a little confusing because the sets of coefficients are not labeled. However, the first set are the coefficients that go with psechoice=2 and the second set go with psechoice=3; psechoice=1 is used at the base.

To obtain the probabilities and marginal effects, a little work is required. Fortunately, **gretl**'s matrix and scripting abilities will save you from doing a lot of calculations by hand. The first thing to do is to place the coefficients into a matrix, which I will call **theta**. Then each of the elements of **theta** has to be assigned to the desired coefficient. I refer to β_{12} as **b12** and so on. Then, equations (16.9) in *POE* are used to compute the estimated probabilities for the 50th and 5th percentiles of the data.

```
matrix theta = $coeff
# Assign elements of theta to coefficient names
scalar b12 = theta[1]
scalar b22 = theta[2]
scalar b13 = theta[3]
scalar b23 = theta[4]
#Use the Quantile function to get the 5% and 50% quantiles
scalar q50 = quantile(grades,.5)
scalar q5 = quantile(grades,.05)
# Note: gret1 uses a different method to get quantiles than poe so
# I reassigned the 5th quantile to match that in POE.
scalar q5 = 2.635
#No College probabilities
```

The estimated marginal effects from POE can also be easily reproduced using the following script.

```
#Marginal effects, No College
scalar pa_50 = 1/(1+exp(b12+b22*(q50-.5))+exp(b13+b23*(q50-.5)))
scalar pa_5 = 1/(1+exp(b12+b22*(q5-.5))+exp(b13+b23*(q5-.5)))
scalar pb_50 = 1/(1+exp(b12+b22*(q50+.5))+exp(b13+b23*(q50+.5)))
scalar pb_5 = 1/(1+exp(b12+b22*(q5+.5))+exp(b13+b23*(q5+.5)))
scalar m1=pb_50-pa_50
scalar m2=pb_5-pa_5
#Marginal effects, 2 Year College
scalar pa_50 = exp(b12+b22*(q50-.5))/(1+exp(b12+b22*(q50-.5)) \
                               + exp(b13+b23*(q50-.5)))
scalar pa_5 = exp(b12+b22*(q5-.5))/(1+exp(b12+b22*(q5-.5)))
                               + exp(b13+b23*(q5-.5)))
scalar pb_{50} = exp(b12+b22*(q50+.5))/(1+exp(b12+b22*(q50+.5)))
                               + exp(b13+b23*(q50+.5)))
scalar pb_5 = \exp(b12+b22*(q5+.5))/(1+\exp(b12+b22*(q5+.5)))
                               + exp(b13+b23*(q5+.5)))
scalar m3=pb_50-pa_50
scalar m4=pb_5-pa_5
#Marginal effects, 4 Year college
scalar pa_50 = exp(b13+b23*(q50-.5))/(1+exp(b12+b22*(q50-.5)))
                               + exp(b13+b23*(q50-.5)))
scalar pa_5 = exp(b13+b23*(q5-.5))/(1+exp(b12+b22*(q5-.5)))
```
This script uses a common trick. The quantiles are evaluated at $\pm .5$ on either side of each quantile; then the discrete difference is taken. The results match those in *POE* as well as those produced using the slick mfx command in Stata.

The option --multinomial is used when the choices are unordered. For ordered logit, this option is omitted. Gretl takes a look at the dependent variable, in this case psechoice, to make sure that it is actually discrete. Ours takes on three possible values (1,2, or 3) and the logit function in gretl will handle this automatically.

The output appears in Figure 16.2. As you can see, these results match those in POE almost

Figure 16.2: These results are from the native gretl routine to estimate unordered choice models.

🎇 gretl: model 3	3					
<u>File E</u> dit <u>T</u> ests	<u>S</u> ave <u>G</u> raphs <u>A</u> na	ilysis <u>L</u> aTeX				
Model 3: Mul Dependent va Standard err	tinomial Logit riable: psecho ors based on H	estimates u ice essian	sing the 10	00 observat	tions 1-	1000
	coefficient	std. error	t-ratio	p-value		
psechoice	= 2					
const	2.50642	0.418385	5.991	2.09e-09	* * *	
grades	-0.308789	0.0522849	-5.906	3.51e-09	* * *	
psechoice	= 3					
const	5.76988	0.404323	14.27	3.34e-046	* * *	
grades	-0.706197	0.0529246	-13.34	1.29e-040	***	
Mean depende Log-likeliho Schwarz crit	nt var 2.305 od -875.3 erion 1778.	000 S.D. d 131 Akaike 257 Hannan	ependent va criterion -Quinn	r 0.81032 1758.62 1766.08	28 26 37	
Number of ca Likelihood r	ses 'correctly atio test: Chi	predicted' -square(2) =	= 552 (55.2 286.689 [O	%) .0000]		

exactly.

16.2.1 Using a script for MNL

In this section I'll give you an idea of how to estimate this model using **gretl** script and in section 16.9 I'll show you how to estimate the model in another free software called R.

Although versions of **Gretl** prior to 1.8.1 did not include a specific function for estimating MNL, it can estimated with a little effort. **Gretl** contains two things that make this reasonably easy to do. First, it includes a module for maximizing likelihood functions (see Chapter 14 for other examples). To use the mle function, the user has to write a program using gretl's language to compute a model's log-likelihood given the data. The parameters of the log-likelihood must be declared and given starting values (using the genr command). If you want, you can specify the derivatives of the log-likelihood function with respect to each of the parameters; if analytical derivatives are not supplied, a numerical approximation is computed. In many instances, the numerical approximations work quite well. In the event that the computations based on numerical derivatives fail, you may have to specify analytical ones to make the program work.

Gretl also includes a way for users to define new functions. These are placed in a script that can be run from the script editor. Once a function is written, it can often be reused with ease. Functions can also be published and shared via **gretl**'s database server. The **Gretl** Users Guide will have the most up-to-date information on the use of functions and I suggest you look there for further information. What appears below is taken from the **gretl** Users Guide. The example for MNL for *POE* requires only a slight modification in order for the program to run with our dataset.

Functions must be defined before they are called (used). The syntax for defining a function looks like this

```
function name(inputs)
    function body
end function
```

You select a name to give your function. Keep it under 32 characters and start the name with a character. The inputs usually include the data and any parameters included in the log-likelihood. The parameters can be in matrix or scalar form.

The multinomial logit function, which can be found in the **Gretl** User's Guide, is defined

```
function mlogitlogprobs(series y, matrix X, matrix theta)
scalar n = max(y)
scalar k = cols(X)
matrix b = mshape(theta,k,n)
matrix tmp = X*b
series ret = -ln(1 + sumr(exp(tmp)))
loop for i=1..n --quiet
series x = tmp[,i]
```

ret += (y=\$i) ? x : 0
end loop
return series ret
end function

The function is named mlogitlogprobs and has three arguments. The first is the dependent variable, series y, the second is set of independent variables contained in matrix X, and the last is the matrix of parameters, called theta. Scalars in the function are defined for sample size, number of regressors, and the coefficients are placed in an nxk array in order to match the dimensionality of the data. The index tmp=X*b is created and ret returns the log-likelihood function. Don't worry if you can't make sense of this because you should not have to change any of this to estimate MNL with another dataset. That is one of the beauties of defining and using a function.

To use the mlogitlogprobs function, you need to know a little about how it works. You will have to get your data into the right form in order for the function to work properly. After loading the data, make sure that the dependent choice variable is in the correct format for the function. The function requires the choices to start at 0. If you list the data, you'll find that psechoice is coded 1, 2, 3 instead of the required 0, 1, 2. So the next step is to subtract 1 from psechoice.

Create the matrix of regressors, define the number of regressors and use these to initialize the matrix of coefficients, theta. Then list the dependent variable, matrix of independent variables, and the initialized parameter matrix in the function. Click the run button and wait for the result.

```
open "c:\Program Files\gretl\data\poe\nels_small.gdt"
# dep. var. must be 0-based
psechoice = psechoice-1
#put regressors into a matrix called X
smpl full matrix X = { grades const }
scalar k = cols(X)
matrix theta = zeros(2*k, 1)
mle loglik = mlogitlogprobs(psechoice,X,theta)
params theta
end mle --verbose --hessian
```

The only changes I had to make to the original example in the **Gretl** User Guide are (1) change the dataset (2) change the dependent variable to **psechoice** (3) put the desired regressors into X and (4) make sure the function contains the desired variables.

The results from the program appear below in Figure 16.3. Wow! They match those in POE and are dirt simple to obtain!¹ Finally, if you want to produce the probabilities and marginal

¹Thanks to Jack Lucchetti for pointing this out to me.

Figure 16.3: These results are from a gretl function taken from the Gretl Users Guide.

```
🦉 gretl: script output
                                                                             <u>- 🗆 ×</u>
                                                                      2
🔏 占 🛅 🖻 😰 🥱 🗴
 Gradients: -6.3063e-005 9.3064e-006 7.7355e-006-1.4433e-005
                                                                                 -
  -- FINAL VALUES:
 Iteration 17: Log-likelihood = -875.313086514 (steplength = 1.28e-005)
                                                       5.7699
 Parameters:
                -0.30879
                             2.5064 -0.70620
 Gradients: -6.3063e-005 9.3064e-006 7.7355e-006-1.4433e-005
 Tolerance = 1.81899e-012
 Function evaluations: 54
 Evaluations of gradient: 17
 Model 1: ML estimates using the 1000 observations 1-1000
 loglik = mlogitlogprobs(psechoice, X, theta)
 Standard errors based on Hessian
       PARAMETER
                      ESTIMATE
                                        STDERROR
                                                       T STAT
                                                               P-VALUE
                       -0.308789
   theta[1]
                                         0.0522846
                                                      -5.906 <0.00001 ***
                                                       5.991 <0.00001 ***
   theta[2]
                        2.50642
                                         0.418383
                        -0.706197
                                                     -13.343 <0.00001 ***
   theta[3]
                                         0.0529248
                                                      14.270 <0.00001 ***
                                         0.404324
                        5.76988
   theta[4]
   Log-likelihood = -875.313
   Akaike information criterion (AIC) = 1758.63
   Schwarz Bayesian criterion (BIC) = 1778.26
   Hannan-Quinn criterion (HQC) = 1766.09
                                       Close
```

effects, you can use the estimates **gretl** has stored in the 4x1 vector called **theta**. This was the approach taken in the preceding section and I won't repeat the details here.

16.3 Conditional Logit

Gretl doesn't include a routine to estimate conditional logit yet (as of version 1.8.1), so you'll want to use R to estimate this model. See sections 16.9 and 16.9.2 for details.

16.4 Ordered Probit

In this example, the probability of attending no college, a 2 year college, and a 4 year college are modeled as a function of a student's grades. In principle, we would expect that those with higher grades to be more likely to attend a 4 year college and less likely to skip college altogether. In the dataset, grades are measured on a scale of 1 to 13, with 1 being the highest. That means that if higher grades increase the probability of going to a 4 year college, the coefficient on grades will be *negative*. The probabilities are modeled using the normal distribution in this model where the outcomes represent increasing levels of difficulty.

We can use **gretl** to estimate the ordered probit model because its **probit** command actually handles multinomial ordered choices as well as binomial choice. Open the **nels_small.gdt** data

```
open "c:\Program Files\gretl\data\poe\nels_small.gdt"
```

probit psechoice const grades

The results in Figure 16.4 are very much like the ones in *POE* and produced by MCMCpack below.

	Depend	iciii variai	sie. pseene	nee	
	Coefficien	t Std. E	rror z-s	stat p-v	value
grades	-0.306624	4 0.0191	735 - 15	.9921 0.0	0000
$\operatorname{cut1}$	-2.94559	0.14682	28 -20	.0615 0.0	0000
$\mathrm{cut}2$	-2.08999	0.1357	68 - 15	.3938 0.0	0000
Mean depende	ent var f	2.305000	S.D. depe	endent var	0.810328
Log-likelihood	l —8	875.8217	Akaike cr	iterion	1757.643
Schwarz criter	rion	1772.367	Hannan–	Quinn	1763.239

Model 3: Ordered Probit estimates using the 1000 observations 1–1000 Dependent variable: psechoice

Number of cases 'correctly predicted' = 545 (54.5 percent) Likelihood ratio test: $\chi^2(1) = 285.672 [0.0000]$

From the pull-down menus simply click on Model>Nonlinear model>Probit>Ordered and fill in the now familiar dialog box. To get marginal effects is easy using some of the built-in functions in gret1. The algebraic results we use are:

$$\begin{aligned} \frac{\partial P(y=1)}{\partial grades} &= -\phi(\mu_1 - \beta grades)\beta\\ \frac{\partial P(y=2)}{\partial grades} &= [\phi(\mu_1 - \beta grades) - \phi(\mu_2 - \beta grades)]\beta\\ \frac{\partial P(y=3)}{\partial grades} &= \phi(\mu_2 - \beta grades)\beta \end{aligned}$$

where ϕ is the probability density function of a standard normal distribution. The parameters μ_1 and μ_2 are the thresholds and β is the coefficient on **grades**. So, for example if you want to calculate the marginal effect on the probability of attending a 4-year college (y = 3) for a student having grades at the median (6.64) and 5th percentile (2.635) use:

probit psechoice grades

Figure 16.4: Ordered probit results from the gretl's probit command called from the Console

```
📓 gretl console
                                                                                                   _ 🗆 🗙
 🔏 占 🕞 🔍 🗙
 ? probit psechoice const grades
                                                                                                           ^
 Function evaluations: 63
 Evaluations of gradient: 12
 Model 2: Ordered Probit estimates using the 1000 observations 1-1000
 Dependent variable: psechoice
                  coefficient std. error t-ratio
                                                                   p-value
   grades
                   -0.306624 0.0191735 -15.99 1.45e-057 ***
                   -2.94559 0.146828
                                                  -20.06
                                                                  1.60e-089 ***
   cut1
                                                                  1.80e-053 ***
   cut2
                   -2.08999
                                  0.135768
                                                   -15.39

        Mean dependent var
        2.305000
        S.D. dependent var
        0.810328

        Log-likelihood
        -875.8217
        Akaike criterion
        1757.643

        Schwarz criterion
        1772.367
        Hannan-Quinn
        1763.239

 Number of cases 'correctly predicted' = 545 (54.5%)
 Likelihood ratio test: Chi-square(1) = 285.672 [0.0000]
```

```
k = $ncoeff
matrix b = $coeff[1:k-2]
mu1 = $coeff[k-1]
mu2 = $coeff[k]
matrix X = {6.64}
scalar Xb = X*b
P3a = pdf(N,mu2-Xb)*b
matrix X = 2.625
```

matrix X = 2.635
scalar Xb = X*b
P3b = pdf(N,mu2-Xb)*b

printf "\nFor the median grade of 6.64, the marginal effect is $%.4f\n$ ", P3a printf "\nFor the 5th percentile grade of 2.635, the marginal effect is $%.4f\n$ ", P3b

16.5 Poisson Regression

When the dependent variable in a regression model is a count of the number of occurrences of an event you'll want to use the poisson regression model. In these models, the dependent variable is a nonnegative integer, (i.e., y = 0, 1, ...), which represent the number of occurrences of a particular event. The probability of a given number of occurrences is modeled as a function of independent variables.

$$P(Y = y|x) = \frac{e^{-\lambda}\lambda^y}{y!} \ y = 0, 1, 2, \dots$$
 (16.2)

where $\lambda = \beta_1 + \beta_2 x$ is the regression function.

Estimating this model using maximum likelihood is very simple since the MLE of the poisson regression model is already programmed into **gret1**. The syntax for a script is the same as for linear regression except you use the **possion** command in place of **ols**. This is shown in the following script which replicates the example from your textbook.

A country's total number of medals (medaltot) in the 1988 olympics is modeled as a function of ln(gdp) and ln(pop). Of course, you can also access the poisson regression estimator from the pull-down menus using Model>Nonlinear models>Possion. To replicate the example in *POE* be sure to restrict the sample to 1988 before estimating the model.

```
open "c:\Program Files\gretl\data\poe\olympics.gdt"
```

The results for poisson model are:

Model 2:	Poisson est	imates us	sing the	151 o	bservations	29 - 179
	Depe	endent va	riable: r	nedal	tot	

	Coefficient	Std.	Error	z-stat	p-val	ue
const	-15.8875	0.51	1805	-31.0420	0.000	00
lgdp	0.576603	3 0.02	47217	23.3238	0.000	00
lpop	0.180038	8 0.03	22801	5.5773	0.000	00
Mean depende	ent var	4.88741	7 S.D.	. dependent	var	16.62670
Sum squared	resid	25165.5	8 S.E.	of regressic	n	13.03985
McFadden \mathbb{R}^2		0.54465	8 Adj	usted R^2		0.542766
Log-likelihood	l –	722.336	5 Aka	ike criterion	L	1450.673
Schwarz criter	rion	1459.72	5 Han	nan–Quinn		1454.350

16.6 Tobit

The tobit model is essentially just a linear regression where some of the observations on your dependent variable have been censored. A **censored** variable is one that, once it reaches a limit, it is recorded at that limit no matter what it's actual value might be. For instance, anyone earning \$1 million or more per year might be recorded in your dataset at the upper limit of \$1 million. That means that Bill Gates and the authors of your textbook earn the same amount in the eyes of your dataset (just kidding, fellas). Least squares can be seriously biased in this case and it is wise to use a censored regression model (tobit) to estimate the parameters of the regression when a portion of your sample is censored.

Hill et al. (2007) use tobit to estimate a model of hours worked shown in equation (16.3).

$$hours_i = \beta_1 + \beta_2 * educ_i + \beta_3 exper_i + \beta_4 * age_i + \beta_5 * kidsl_{i} + e_i$$
(16.3)

using the *mroz.gdt* data. A number of individuals in the sample do not work and report zero hours worked. Estimation of this model in **gretl** is shown in the following script which replicates the example from *POE*. The script estimates a tobit model of hours worked and generates the marginal effect of another year of schooling on the average hours worked.

```
open "c:\Program Files\gretl\data\poe\mroz.gdt"
tobit hours const educ exper age kidsl6
```

The results from the basic tobit estimation of the hours worked equation are:

		L				
	Coeffici	ent Std.	Error	z-stat	p-valı	ıe
const	1349.88	3 382.	729	3.5270	0.0004	4
educ	73.29	910 20.	7496	3.5322	0.0004	4
exper	80.53	353 6.	58247	12.2348	0.000	0
age	-60.76	678 7.	27480	-8.3532	0.000	0
kidsl6	-918.92	18 113.	036	-8.1294	0.000	0
Mean depende	ent var	740.576	4 S.I). dependent	var	871.3142
Censored obs		32	$5 ext{ sig}$	ma		1133.697
Log-likelihood		-3827.14	3 Ak	aike criterio	n	7666.287
Schwarz criter	ion	7694.03	1 Ha	nnan–Quinn	L	7676.975

Model 1:	Tobit	estimates	using	the '	753	observ	vations	1 - 753
		Depender	nt vari	able:	ho	urs		

Test for normality of residual –

Null hypothesis: error is normally distributed

Test statistic: $\chi^2(2) = 6.31677$

with p-value = 0.0424944

The marginal effect of another year of schooling on hours worked is

$$\frac{\partial E(Hours_i)}{\partial Educ_i} = \Phi(\widehat{Hours_i})\hat{\beta}_2, \tag{16.4}$$

where $Hours_i$ is the estimated regression function evaluated at the mean levels of education, experience, and age for a person with one child under the age of six. Then, the **cnorm** function is used to compute the normal CDF, $\Phi(Hours_i)$, evaluated at the prediction.

Note, the backward slashes (\backslash) used at the end of the first two lines in the generation of H_hat are continuation commands. The backslash at the end of a line tells **gret1** that the next line is a continuation of the current line. This helps keep your programs looking good (and in this case, fitting within the margins of the page!).

Finally, estimates of the restricted sample using least squares and the full sample that includes the zeros for hours worked follow.

smpl hours > 0 --restrict
ols hours const educ exper age kidsl6
smpl --full
ols hours const educ exper age kidsl6

Notice that the sample is restricted to the positive observations using the smpl hours > 0 --restrict statement. To estimate the model using the entire sample the full range is restored using smpl full.

16.7 Simulation

You can use **gretl** to show that least squares is biased when the sample is censored using a Monte Carlo simulation. The simulated data are generated

$$y_i^* = -9 + 1x_i + e_i \tag{16.5}$$

where $e_i \sim N(0, 16)$. Then,

$$y_i = \begin{cases} y_i^* & \text{if } y_i^* > 0\\ 0 & \text{if } y_i^* \le 0 \end{cases}$$

The $x_i \sim U(0, 20)$, which are held constant in the simulated samples.

The following script demonstrates that least squares is indeed biased when all observations, including the zero ones, are included in the sample. The line genr yi = y > 0 is a logical statement that generates '1' or '0' depending on whether the statement to the right of the equal sign is true. Thus, a new variable, yi, is created that takes the value 1 if y > 0 and is zero if not. When multiplied by y in the next statement, the result is a sample, yc, censored from below at zero.

```
open "c:\Program Files\gretl\data\poe\tobit.gdt"
smpl 1 200
genr xs = 20*uniform()
loop 1000 --progressive
  genr y = -9 + 1*xs + 4*normal()
  genr yi = y > 0
  genr yc = y*yi
  ols yc const xs
  genr b1s = $coeff(const)
  genr b2s = $coeff(xs)
  store coeffs.gdt b1s b2s
endloop
```

To repeat the exercise using least squares on only the positive observations use

```
open "c:\Program Files\gretl\data\poe\tobit.gdt"
genr xs = 20*uniform()
genr idx = 1
matrix A = zeros(1000,3)
loop 1000
   smpl --full
   genr y = -9 + 1 \times s + 4 \times normal()
   smpl y > 0 --restrict
   ols y const xs --quiet
   genr b1s = $coeff(const)
   genr b2s = $coeff(xs)
   matrix A[idx,1]=idx
   matrix A[idx,2]=b1s
   matrix A[idx,3]=b2s
   genr idx = idx + 1
endloop
A matrix bb = meanc(A) bb
```

In this case, we are not able to use the --progressive loop construct in gretl. Without it, gretl generates a lot of output to the screen, but it can't be avoided in this case. Using the regular loop

function, store each round's estimates in a matrix called A. Then, after the loop is finished, matrix bb = meanc(A) returns the column means of your matrix. These are the average values of the parameters in the Monte Carlo.

16.8 Selection Bias

Selection bias occurs when your sample is truncated and the cause of that truncation is correlated with your dependent variable. Ignoring the correlation, the model could be estimated using least squares or truncated least squares. In either case, obtaining consistent estimates of the regression parameters is not possible. In this section the basic features of the this model will be presented.

Consider a model consisting of two equations. The first is the **selection equation**, defined

$$z_i^* = \gamma_1 + \gamma_2 w_i + u_i, \quad i = 1, \dots, N$$
(16.6)

where z_i^* is a latent variable, γ_1 and γ_2 are parameters, w_i is an explanatory variable, and u_i is a random disturbance. The latent variable is unobservable, but we do observe the dichotomous variable

$$z_i = \begin{cases} 1 & z_i^* > 0\\ 0 & \text{otherwise} \end{cases}$$
(16.7)

The second equation, called the **regression equation**, is the linear model of interest. It is

$$y_i = \beta_1 + \beta_2 x_i + e_i, \quad i = 1, \dots, n, \quad N > n$$
 (16.8)

where y_i is an observable random variable, β_1 and β_2 are parameters, x_i is an exogenous variable, and e_i is a random disturbance. It is assumed that the random disturbances of the two equations are distributed as

$$\begin{bmatrix} u_i \\ e_i \end{bmatrix} \sim N \begin{bmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & \sigma_e^2 \end{pmatrix} \end{bmatrix}$$
(16.9)

A selectivity problem arises when y_i is observed only when $z_i = 1$ and $\rho \neq 0$. In this case the ordinary least squares estimator of β in (16.8) is biased and inconsistent. A consistent estimator has been suggested by Heckman (1979) and is commonly referred to as Heckman's two-step estimator, or more simply, *Heckit*. Because the errors are normally distributed, there is also a maximum likelihood estimator of the parameters. **Gret1** includes routines for both.

The two-step (Heckit) estimator is based on conditional mean of y_i given that it is observed

$$E[y_i|z_i > 0] = \beta_1 + \beta_2 x_i + \beta_\lambda \lambda_i \tag{16.10}$$

where

$$\lambda_i = \frac{\phi(\gamma_1 + \gamma_2 w_i)}{\Phi(\gamma_1 + \gamma_2 w_i)} \tag{16.11}$$

is the *inverse Mill's ratio*; $\phi(\cdot)$ is the standard normal probability density function evaluated at the argument, and $\Phi(\cdot)$ is the cumulative density function of the standard normal random variable evaluated at the argument $(\gamma_1 + \gamma_2 w_i)$. The argument $(\gamma_1 + \gamma_2 w_i)$ is commonly referred to as the **index function**. Adding a random disturbance yields:

$$y_i = \beta_1 + \beta_2 x_i + \beta_\lambda \lambda_i + v_i \tag{16.12}$$

It can be shown that (16.12) is heteroskedastic and if λ_i were known (and nonstochastic), then the selectivity corrected model (16.12) could be estimated by generalized least squares. Alternately, the heteroskedastic model (16.12) could be estimated by ordinary least squares, using White's heteroskedasticity consistent covariance estimator (HCCME) for hypothesis testing and the construction of confidence intervals. Unfortunately, λ_i is not known and must be estimated using the sample. The stochastic nature of λ_i in (16.12) makes the automatic use of HCCME in this context inappropriate.

The two-steps of the Heckit estimator consist of

- 1. estimate the selection equation to obtain $\hat{\gamma}_1$ and $\hat{\gamma}_2$. Use these in equation (16.11) to estimate the inverse Mill's ratio, $\hat{\lambda}_i$.
- 2. Add $\hat{\lambda}_i$ to the regression model as in equation (16.12) and estimate it using least squares.

The example from *POE* uses the *mroz.gdt* data. The first thing we'll do is to estimate the model ignoring selection bias using least squares on the nonzero observations. Load the data and generate the natural logarithm of wages. Since wages are zero for a portion of the sample, **gretl** will generate an error when you take the natural logs. You can safely ignore it as **gretl** will simply create missing values for the variables that cannot be transformed. Then use the ols command to estimate a linear regression on the truncated subset.

open "c:\Program Files\gretl\data\poe\mroz.gdt"
logs wage
ols l_wage const educ exper

The results are:

Model 1:	OLS estimates using the 428 observations	1 - 428
	Dependent variable: l_wage	

	Coefficient	Std. Error	t-ratio	p-value
const	-0.400174	0.190368	-2.1021	0.0361
educ	0.109489	0.0141672	7.7283	0.0000
exper	0.0156736	0.00401907	3.8998	0.0001

Mean dependent var	1.190173	S.D. dependent var	0.723198
Sum squared resid	190.1950	S.E. of regression	0.668968
R^2	0.148358	Adjusted R^2	0.144350
F(2, 425)	37.01805	P-value (F)	$1.51e{-}15$
Log-likelihood	-433.7360	Akaike criterion	873.4720
Schwarz criterion	885.6493	Hannan–Quinn	878.2814

Notice that the sample has been truncated to include only 428 observations for which hour worked are actually observed. The estimated return to education is about 11%, and the estimated coefficients of both education and experience are statistically significant.

The Heckit procedure takes into account that the decision to work for pay may be correlated with the wage a person earns. It starts by modeling the decision to work and estimating the resulting selection equation using a probit model. The model can contain more than one explanatory variable, w_i , and in this example we have four: a womans age, her years of education, a dummy variable for whether she has children and the marginal tax rate that she would pay upon earnings if employed. Generate a new variable **kids** which is a dummy variable indicating the presence of any kids in the household.

genr kids = (kids16+kids618>0)

Estimate the probit model, generate the index function, and use it to compute the inverse Mill's ratio variable. Finally, estimate the regression including the IMR as an explanatory variable.

This script uses a convenient way to accumulate variables into a set using the list command first encountered in section 10.3.3. The command list X = const educ exper puts the variables contained in const, educ, and exper into a set called X. Once defined, the set of variables can be referred to as X rather than listing them individually as we've done up to this point. Similarly, we've put the variables from the selection equation into a set called W. The dnorm and cnorm functions return the normal density and normal cumulative density evaluated at the argument, respectively. The results are:

Model 2: OLS estimates using the 428 observations 1–428 Dependent variable: l_wage

С	oefficient	Std. E	Error	t-ratio	p-va	lue
const	0.810542	0.4944	72	1.6392	0.10	19
educ	0.0584579	0.0238	495	2.4511	0.01	46
exper	0.0163202	0.0039	9836	4.0817	0.00	01
lambda –	0.866439	0.3269	86	-2.6498	0.00	84
Mean dependent	var 1.19	0173	S.D. de	ependent v	ar	0.723198
Sum squared resid	d 187.	0967	S.E. of	regression		0.664278
R^2	0.162	2231	Adjust	ed R^2		0.156304
F(3, 424)	27.3	6878	P-value	e(F)		$3.38e{-16}$
Log-likelihood	-430.2	2212	Akaike	criterion		868.4424
Schwarz criterion	884.	6789	Hannaı	n–Quinn		874.8550

Notice that the estimated coefficient of the inverse Mill's ratio is statistically significant, implying that there is a selection bias in the least squares estimator. Also, the estimated return to education has fallen from approximately 11% (which is inconsistently estimated) to approximately 6%. Unfortunately, the usual standard errors do not account for the fact that the inverse Mills ratio is itself an estimated value and so they are not technically correct. To obtain the correct standard errors, you will use **gretl**'s built-in heckit command.

The heckit command syntax is

heckit y const x2 x3 ... xk; z const w2 w3 ... ws --options

where const $x2 \ldots xk$ are the k independent variables for the regression and const $w2 \ldots ws$ are the s independent variables for the selection equation. In this example, we've used the twostep option which mimics the manual procedure employed above, but returns the correct standard errors. If you don't specify the option, gretl will estimate the model using maximum likelihood. For the Mroz data the gretl command is

heckit lwage X ; lfp W --two-step

Again, we've used the results from the list function, which put the independent variables for the regression into X and the variables for the selection equation into W.

The results appear below:

Model 3: Two-step Heckit estimates using the 428 observations 1–428 Dependent variable: l_wage Selection variable: lfp

	Coefficient	Std. Error	z-stat	p-value
const	0.810542	0.610798	1.3270	0.1845
educ	0.0584579	0.0296354	1.9726	0.0485
exper	0.0163202	0.00420215	3.8838	0.0001
lambda	-0.866439	0.399284	-2.1700	0.0300
	Selec	ction equation	n	
const	1.19230	0.720544	1.6547	0.0980
mtr	-1.39385	0.616575	-2.2606	0.0238
age	-0.0206155	0.00704470	-2.9264	0.0034
kids	-0.313885	0.123711	-2.5372	0.0112
educ	0.0837753	0.0232050	3.6102	0.0003
Mean depende $\hat{\sigma}$	ent var 1.190 0.932	173 S.D. de 559 $\hat{\rho}$	pendent va	r 0.723198 -0.929098

Total observations: 753 Censored observations: 325 (43.2%)

To use the pull-down menus, select Model>Nonlinear models>Heckit from gretl's main window. This will reveal the dialog shown in figure 16.5. Enter lwage as the dependent variable and the 0/1 variable lfp as the selection variable. Then enter the desired independent variables for the regression and selections equations. Finally, select the 2-step estimation button at the bottom of the dialog box and click OK.

You will notice that the coefficient estimates are identical to the ones produced manually above. However, the standard errors, which are now consistently estimated, have changed. The t-ratio of the coefficient on the inverse Mills ratio, λ , has fallen to -2.17, but it is still significant at the 5% level. Gretl also produces the estimates of the selection equation, which appear directly below those for the regression.

16.9 Using R for Qualitative Choice Models

R is a programming language that can be very useful for estimating sophisticated econometric models. In fact, many statistical procedures have been written for R. Although gretl is reasonably powerful, there are still many things that it won't do. The ability to export gret data into Rmakes it possible to do some fancy econometrics with relative ease.

To do some of these, you'll need a copy of R and access to its packages. A package is just a collection of programs written in R that make it easier to use for specific tasks. Below, we use a package to read in data saved in Stata's format and another to estimate qualitative choice models.

Figure 16.5: Choose Model>Nonlinear models>Heckit from gretl's main window to reveal the dialog box for Heckit.

📓 gretl: specify model 🛛 📃 🗖 🔀					
	Heckit				
hmothereduc 🗠	Dependent variable				
siblings	Choose -> Iwage				
lfp	Set as default				
hours					
kidsl6	Selection variable				
kids618	Choose-> Ifp				
age					
educ	Independent variables				
wage	const				
Wage/6	educ				
hade	exper				
l beduc					
hodde					
faminc	<- Remove				
mtr					
mothereduc	·				
fathereduc	Selection equation regressors				
unemployment	const				
largecity	Add -> educ				
exper	kids				
kids	mtr				
lwage	age				
l ind	<- Remove				
l imr 🔽 🗸					
Show details of iterations					
O Maximum likelihood estimat	ion				
• 2-step estimation					
	ear 🦧 Cancel 🖉 🖓 🖓 K				

The R software package that is used to estimate qualitative choice models is called MCMCpack. MCMCpack stands for Markov Chain Monte Carlo package and it can be used to estimate every qualitative choice model in this chapter. We will just use it to estimate multinomial logit, conditional logit, and ordered probit. So, let's take a quick look at MCMCpack and what it does.

The Markov chain Monte Carlo (MCMC) methods are basic numerical tools that are often used to compute Bayesian estimators. In Bayesian analysis one combines what one already knows (called the *prior*) with what is observed through the sample (the likelihood function) to estimate the parameters of a model. The information available from the sample information is contained in the likelihood function; this is the same likelihood function discussed in your book. If we tell the Bayesian estimator that everything we know is contained in the sample, then the two estimators are essentially the same. That is what happens with MCMCpack under its defaults. The biggest difference is in how the two estimators are computed. The MLE is computed using numerical optimization of the likelihood function, whereas MCMCpack uses simulation to accomplish virtually the same thing. See Lancaster (2004) or Koop (2003) for an introduction to Bayesian methods and its relationship to maximum likelihood.

The MCMC creates a series of estimates–called a (Markov) chain–and that series of estimates has a probability distribution. Under the proper circumstances the probability distribution of the chain will mimic that of the MLE. Various features of the chain can be used as estimates. For instance, the sample mean is used by MCMCpack to estimate the parameters of the multinomial logit model. MCMCpack uses variation within the chain to compute the MLE variance covariance matrix, which is produced using the summary command.

One piece of information that you must give to MCMCpack is the desired length of your Markov chain. In the examples here, I chose 20,000, which is the number used in the sample programs included in MCMCpack. Longer chains tend to be more accurate, but take longer to compute. This number gets us pretty close to the MLEs produced by Stata.

16.9.1 Multinomial Logit

The program code to estimate the multinomial logit example is shown below:

First, read the Stata dataset $nels_small.dta^2$ into an object we will call nels. This requires you to first load the *foreign* library in R using the command library(foreign). The read.dta() command reads data in Stata's format; its argument points to the location on your computer where the Stata dataset resides. Refer to sections D.1 and D.2 for a brief introduction to packages and reading Stata datasets in R.

Then load MCMCpack library into R. The next line calls the multinomial logit estimator (MCMCmn1). The first argument of MCMCmn1 is the dependent variable nels\$psechoice, followed by a \sim , and then the independent variable nels\$grades. The last argument tells R how many simulated values to compute, in this case 20,000. The results of the simulation are stored in the object called posterior. Posterior is the name given in the Bayesian literature to the probability distribution of the estimates. The mean or median of this distribution is used as a point estimate (vis-a-vis the MLE). The last line of the program requests the summary statistics from the Markov chain. The results appear in Figure 16.6 In the MNL model, the estimates from MCMCpack are a little

Figure 16.6: Multinomial logit results from the MCMCmnl estimator in R

🥂 R Console							
> summary(post	erior)						1
Iterations = 1	1001:21000	1					
Thinning inter	val = 1						
Number of cha:	ins = 1						
Sample size pe	er chain =	20000					
1 Empirical r	een end e	tenderc	ldevieti	on for	each vari	eble	
plus stands	ard error	of the	mean:	on tor	Cach Vall	abie,	
		~					
17	nean	50	Naive S	E lime-	Series SE		
(Intercept).2	Z.5264 U	.42003	0.002970	4	0.0027186		
(Intercept).3	5.8023 0	.40622	0.002872	4	0.0027793		
nels&grades.2	-0.3111 0	.05232	0.000370	4	0.0003449		
neisygrades.5	-0.7102 0	.03322	0.000376	-	0.0003627		_
2. Quantiles f	for each v	ariable	::				
	2.5%	25%	50%	754	97.5%		
(Intercept).2	1.7145	2.2451	2.5221	2.8053	3.3548		
(Intercept).3	5.0279	5.5260	5.7901	6.0681	6.6434		-
nels\$grades.2	-0.4151 -	0.3463	-0.3108	-0.276:	-0.2092		
nels\$grades.3	-0.8196 -	0.7445	-0.7088	-0.6743	-0.6080		
>							-
							1
<							>

different from those produced by Stata, but they are reasonably close.

To compute predicted probabilities and marginal effects, you can use the following script for inspiration:

²This should be available from the POE website.

```
summary(nels$grades)
q5 <- quantile(nels$grades,.05)
q50 <- quantile(nels$grades,.5)
b12 <- mean(posterior[,1])</pre>
b13 <- mean(posterior[,2])</pre>
b22 <- mean(posterior[,3])</pre>
b23 <- mean(posterior[,4])</pre>
"No College probabilities"
p1_50 <- 1/(1+exp(b12+b22*q50)+exp(b13+b23*q50))
p1_5 <- 1/(1+exp(b12+b22*q5)+exp(b13+b23*q5))
p1_50
p1_5
"Marginal effects, No College"
p2_50 <- 1/(1+exp(b12+b22*(q50+1))+exp(b13+b23*(q50+1)))
p2_5 <- 1/(1+exp(b12+b22*(q5+1))+exp(b13+b23*(q5+1)))
p2_50-p1_50
p2_5-p1_5
"2 Year college probabilities"
p1_50 <- exp(b12+b22*q50)/(1+exp(b12+b22*q50)+exp(b13+b23*q50))
p1_5 <- exp(b12+b22*q5)/(1+exp(b12+b22*q5)+exp(b13+b23*q5))
p1_50
p1_5
"Marginal effects, 2 Year College"
p2_50 <- exp(b12+b22*(q50+1))/
            (1+\exp(b12+b22*(q50+1))+\exp(b13+b23*(q50+1)))
p2_5 <- exp(b12+b22*(q5+1))/
            (1+exp(b12+b22*(q5+1))+exp(b13+b23*(q5+1)))
p2_50-p1_50
p2_5-p1_5
"4 Year college probabilities"
p1_50 <- exp(b13+b23*q50)/(1+exp(b12+b22*q50)+exp(b13+b23*q50))
p1_5 <- exp(b13+b23*q5)/(1+exp(b12+b22*q5)+exp(b13+b23*q5))
p1_50
p1_5
"Marginal effects, 4 Year college"
p2_50 <- exp(b13+b23*(q50+1))/
            (1+exp(b12+b22*(q50+1))+exp(b13+b23*(q50+1)))
```

16.9.2 Conditional Logit

In this example I'll show you how to use MCMCpack in R to estimate the conditional logit model.

The first order of business is to get the data into a format that suits *R*. This part is not too pretty, but it works. The data are read in from a Stata dataset using the read.dta function that is included in the *foreign* library. The data are assigned (<-) to the object cola. The attach(cola) statement is not necessary, but including it will enable you to call each of the variables in the object cola by name. For example, cola\$price refers to the variable named price in the object named cola. Once the data object cola is attached, you can simply use price to refer to the variable without prefixing it with the object to which it belongs (i.e., cola\$).

The data in the original Stata dataset are arranged

> c	ola	1:12	2,]			
obs	id	cho	ice prio	ce fe	eature	display
1	1	0	1.79	0	0	
2	1	0	1.79	0	0	
3	1	1	1.79	0	0	
4	2	0	1.79	0	0	
5	2	0	1.79	0	0	
6	2	1	0.89	1	1	
7	3	0	1.41	0	0	
8	3	0	0.84	0	1	
9	3	1	0.89	1	0	
10	4	0	1.79	0	0	

The MCMCpack routine in R wants to see it as

id	bev.choice	pepsi.price	<pre>sevenup.price</pre>	coke.price
1	3	1.79	1.79	1.79
2	3	1.79	1.79	0.89
3	3	1.41	0.84	0.89
4	3	1.79	1.79	1.33
4	3	1.79	1.79	1.33

where each line represents an individual, recording his choice of beverage and each of the three prices he faces. The goal then is to reorganize the original dataset so that the relevant information for each individual, which is contained in 3 lines, is condensed into a single row. To simplify the example, I dropped the variables not being used.

Most of the program below is devoted to getting the data into the proper format. The line

```
pepsi.price <- cola$price[seq(1,nrow(cola),by=3)]</pre>
```

creates an object called pepsi.price. The new object consists of every third observation in cola\$price, starting with observation 1. The square brackets [] are used to take advantage of R's powerful indexing ability. The function seq(1,nrow(cola),by=3) creates a sequence of numbers that start at 1, increment by 3, and extends until the last row of cola i.e., [1 3 6 9 ... 5466]. When used inside the square brackets, these numbers constitute an index of the object's elements that you want to grab. In this case the object is colaprice. The sevenup.price and coke.price lines do the same thing, except their sequences start at 2 and 3, respectively.

The next task is to recode the alternatives to a single variable that takes the value of 1, 2 or 3 depending on a person's choice. For this I used the same technique.

```
pepsi <- cola$choice[seq(1,nrow(cola),by=3)]
sevenup <- 2*cola$choice[seq(2,nrow(cola),by=3)]
coke <- 3*cola$choice[seq(3,nrow(cola),by=3)]</pre>
```

The first variable, pepsi, takes every third observation of cola\$choice starting at the first row. The variable will contain a one if the person chooses Pepsi and a zero otherwise since this is how the variable choice is coded in the Stata file. The next variable for Sevenup starts at 2 and the sequence again increments by 3. Since Seven-up codes as a 2 the ones and zeros generated by the sequence get multiplied by 2 (to become 2 or 0). Coke is coded as a 3 and its sequence of ones and zeros is multiplied by 3. The three variables are combined into a new one called bev.choice that takes the value of 1,2, or 3 depending on a person's choice of Pepsi, Seven-up, or Coke.

Once the data are arranged, load the MCMCpack library and use MCMCmnl to estimate the model. In the conditional logit model uses choice specific variables. For MCMCmnl choice-specific covariates have to be entered using a special syntax: choicevar(cvar,"var","choice") where cvar is the name of a variable in data, var is the name of the new variable to be created, and choice is the level of bev.choice that cvar corresponds to.

```
library(foreign)
cola <- read.dta("c:/Data/Stata/cola.dta")
attach(cola) # optional
pepsi.price <- cola$price[seq(1,nrow(cola),by=3)]
sevenup.price <- cola$price[seq(2,nrow(cola),by=3)]</pre>
```

In this example, we specified that we want to normalize the conditional logit on the coke choice; this is done using the **baseline="3"** option in MCMCmnl.

The results appear in Figure 16.7.



Figure 16.7: Conditional logit results from the MCMCoprobit estimator in R

16.9.3 Ordered Probit

MCMCpack can also be used to estimate the ordered probit model. It is very easy and the results you get using the Markov chain Monte Carlo simulation method are **very** similar to those from maximizing the likelihood. In principle the maximum likelihood and the simulation estimator used by MCMCpack are asymptotically equivalent.³ The difference between MCMCpack and Stata's

³Of course, if you decide to use more information in your prior then they can be substantially different.

MLE results occurs because the sample sizes for the datasets used is small.

```
library(foreign) nels <- read.dta("C:/Program Files/nels_small.dta")
attach(nels)
library(MCMCpack)
posterior <- MCMCoprobit(psechoice ~ grades, mcmc=20000)
summary(posterior)</pre>
```

The first line loads the **foreign** package into into your R library. This package allows you to read in Stata's datasets. The second line creates the data object called **nels**. The attach(nels) statement allows you to refer to the variables in nels directly by their names.

The next line loads MCMCpack into R. Then the ordered probit estimator (MCMCoprobit) is called. The first argument of MCMCoprobit is the dependent variable psechoice, followed by a \sim , and then the independent variable grades. The last argument tells R how many simulated values to compute, in this case 20,000. The results of the simulation are stored in the object called posterior. The mean or median of this distribution is used as your point estimate (vis-a-vis the MLE). The last line of the program requests the summary statistics from the simulated values of the parameters. The results appear in Figure 16.8, where the MLEs are highlighted in red. One important difference between MCMCpack and the MLE is in how the results are reported.

Figure 16.8: Ordered probit results from the MCMC probit estimator in R

<u>R</u> R Console		_		×
The Metropo. 0000000000000 > summary(po	lis acceptance rate for beta was 0.86010 990008000000000000000000000000000000			^
Iterations :	= 1001:21000			
Thinning int	cerval = 1			
Number of cl	nains = 1			
Sample size	per chain = 20000			
1. Empirica. plus star	l mean and standard deviation for each variable, ndard error of the mean:			
	Mean SD Naive SE Time-series SE			
(Intercept)	2.9491 0.14497 0.0010251 0.0029746			
grades	-0.3072 0.01896 0.0001341 0.0002616			
gamma2	0.8541 0.04736 0.0003349 0.0022787			
2. Quantile:	s for each variable:			
	2.5% 25% 50% 75% 97.5%			
(Intercept)	2.6682 2.8497 2.9477 3.0464 3.2385			
grades	-0.3447 -0.3199 -0.3070 -0.2944 -0.2699			
gamma2	0.7654 0.8219 0.8537 0.8854 0.9492			
21				~
<			>	

The model as specified in your textbook contains no intercept and 2 thresholds. To include a separate intercept would cause the model to be perfectly collinear. In MCMCpack, the default model includes an intercept and hence can contain only one threshold.

The 'slope' coefficient β , which is highlighted in Figure 16.8, is virtually the same as that reported in Hill et al. (2007). The other results are also similar and are interpreted like the ones produced in **gret**. The intercept in MCMCpack is equal to $-\mu_1$. The second cut-off in *POE*'s no-intercept model is $\mu_2 = -(Intercept - \gamma_2)$, where γ_2 is the single threshold in the MCMCpack specification. The standard errors are comparable and you can see that they are equivalent to 3 or 4 decimal places to those from the MLE.

16.10 Script

```
open "c:\Program Files\gretl\data\poe\transport.gdt"
#Probit
probit auto const dtime
genr p1 = $coeff(const)+$coeff(dtime)*20
genr dt = dnorm(p1)*$coeff(dtime)
genr p2 = cnorm($coeff(const)+$coeff(dtime)*30)
# Multinomial Logit
open "c:\Program Files\gretl\data\poe\nels_small.gdt"
logit psechoice const grades --multinomial
#Ordered Probit
open "c:\Program Files\gretl\data\poe\nels_small.gdt"
probit psechoice const grades
# Marginal effects on probability of going to 4 year college
k = sncoeff
matrix b = $coeff[1:k-2]
mu1 = $coeff[k-1]
mu2 = \text{$coeff[k]}
matrix X = \{6.64\}
scalar Xb = X*b
P3a = pdf(N,mu2-Xb)*b
matrix X = 2.635
scalar Xb = X*b
P3b = pdf(N,mu2-Xb)*b
printf "\nFor the median grade of 6.64, the marginal effect is %.4f\n", P3a
printf "\nFor the 5th percentile grade of 2.635, the marginal effect is %.4f\n", P3b
# Poisson Regression
```

```
open "c:\Program Files\gretl\data\poe\olympics.gdt"
smpl year = 88 --restrict
genr lpop = log(pop)
genr lgdp = log(gdp)
poisson medaltot const lpop lgdp
genr mft = exp($coeff(const)+$coeff(lpop)*median(lpop) \
                   +$coeff(lgdp)*median(lgdp))*$coeff(lgdp)
#Tobit
open "c:\Program Files\gretl\data\poe\mroz.gdt"
tobit hours const educ exper age kidsl6
genr H_hat = $coeff(const)+$coeff(educ)*mean(educ) \
                          +$coeff(exper) *mean(exper) \
                          +$coeff(age)*mean(age)+$coeff(kids16)*1
genr z = cnorm(H_hat/$sigma)
genr pred = z*$coeff(educ)
smpl hours > 0 --restrict
ols hours const educ exper age kidsl6
smpl --full
ols hours const educ exper age kidsl6
#Heckit
open "c:\Program Files\gretl\data\poe\mroz.gdt"
genr kids = (kidsl6+kids618>0)
logs wage
list X = const educ exper
list W = const mtr age kids educ
probit lfp W
genr ind = $coeff(const) + $coeff(age)*age + \
           $coeff(educ)*educ + $coeff(kids)*kids + $coeff(mtr)*mtr
genr lambda = dnorm(ind)/cnorm(ind)
ols lwage X lambda
heckit lwage X ; lfp W --two-step
#Monte Carlo
open "c:\Program Files\gretl\data\poe\tobit.gdt"
smpl 1 200
genr xs = 20*uniform()
loop 1000 --progressive
```

```
genr y = -9 + 1 + xs + 4 + normal()
  genr yi = y > 0
  genr yc = y*yi
  ols yc const xs --quiet
  genr b1s = $coeff(const)
  genr b2s = $coeff(xs)
  store coeffs.gdt b1s b2s
endloop
open "c:\Program Files\gretl\data\poe\tobit.gdt"
genr xs = 20*uniform()
genr idx = 1
matrix A = zeros(1000,3)
loop 1000 --quiet
   smpl --full
  genr y = -9 + 1 + xs + 4 + normal()
  smpl y > 0 --restrict
  ols y const xs --quiet
  genr b1s = $coeff(const)
  genr b2s = coeff(xs)
  matrix A[idx,1]=idx
  matrix A[idx,2]=b1s
  matrix A[idx,3]=b2s
  genr idx = idx + 1
endloop
# The matrix A contains all 1000 sets of coefficients
# bb finds the column mean of A
matrix bb = meanc(A)
bb
```

And the MNL.inp script for multinomial logit.

```
open "c:\Program Files\gretl\data\poe\nels_small.gdt"
logit psechoice const grades --multinomial
matrix theta = $coeff
#To get predictions
scalar b12 = theta[1]
scalar b22 = theta[2]
scalar b13 = theta[3]
scalar b23 = theta[4]
```

```
#Use the Quantile function to get the 5% and 50% quantiles
scalar q50 = quantile(grades,.5)
scalar q5 = quantile(grades,.05)
scalar q5 = 2.635
#No College probabilities
scalar p1_50 = 1/(1+exp(b12+b22*q50)+exp(b13+b23*q50))
scalar p1_5 = 1/(1+exp(b12+b22*q5)+exp(b13+b23*q5))
#2 Year college probabilities
scalar p2_50 = exp(b12+b22*q50)/(1+exp(b12+b22*q50)+exp(b13+b23*q50))
scalar p2_5 = exp(b12+b22*q5)/(1+exp(b12+b22*q5)+exp(b13+b23*q5))
#4 Year college probabilities
scalar p3_50 = exp(b13+b23*q50)/(1+exp(b12+b22*q50)+exp(b13+b23*q50))
scalar p3_5 = exp(b13+b23*q5)/(1+exp(b12+b22*q5)+exp(b13+b23*q5))
print "Predicted Probabilities for 50th and 5th quantiles
print p1_50 p2_50 p3_50 p1_5 p2_5 p3_5
#Marginal effects, No College
scalar pa_{50} = 1/(1+exp(b12+b22*(q50-.5))+exp(b13+b23*(q50-.5)))
scalar pa_5 = 1/(1+exp(b12+b22*(q5-.5))+exp(b13+b23*(q5-.5)))
scalar pb_50 = 1/(1+exp(b12+b22*(q50+.5))+exp(b13+b23*(q50+.5)))
scalar pb_5 = 1/(1+exp(b12+b22*(q5+.5))+exp(b13+b23*(q5+.5)))
scalar m1=pb_50-pa_50
scalar m2=pb_5-pa_5
#Marginal effects, 2 Year College
scalar pa_50 = exp(b12+b22*(q50-.5))/(1+exp(b12+b22*(q50-.5)) \
               + exp(b13+b23*(q50-.5)))
scalar pa_5 = \exp(b12+b22*(q5-.5))/(1+\exp(b12+b22*(q5-.5)))
               + exp(b13+b23*(q5-.5)))
scalar pb_{50} = exp(b12+b22*(q50+.5))/(1+exp(b12+b22*(q50+.5)))
               + exp(b13+b23*(q50+.5)))
scalar pb_5 = exp(b12+b22*(q5+.5))/(1+exp(b12+b22*(q5+.5)))
               + exp(b13+b23*(q5+.5)))
scalar m3=pb_50-pa_50
scalar m4=pb_5-pa_5
#Marginal effects, 4 Year college
scalar pa_50 = exp(b13+b23*(q50-.5))/(1+exp(b12+b22*(q50-.5)))
                                   + exp(b13+b23*(q50-.5)))
```

print m1 m2 m3 m4 m5 m6 $\,$

Appendix A

gret1 commands

A.1 Estimation

- ar : Autoregressive estimation
- arima : ARMA model
- $\bullet\ {\rm corc}$: Cochrane-Orcutt estimation
- equation : Define equation within a system
- estimate : Estimate system of equations
- garch : GARCH model
- hccm : HCCM estimation
- heckit: Heckit estimation (2-step and MLE)
- hilu : Hildreth-Lu estimation
- hsk : Heteroskedasticity-corrected estimates
- lad : Least Absolute Deviation estimation
- logistic : Logistic regression
- logit : Logit regression
- mle : Maximum likelihood estimation
- mpols : Multiple-precision OLS
- nls : Nonlinear Least Squares

- ols : Ordinary Least Squares
- panel : Panel models
- poisson : Poisson estimation
- probit : Probit model
- pwe : Prais-Winsten estimator
- system : Systems of equations
- tobit : Tobit model
- tsls : Two-Stage Least Squares
- var : Vector Autoregression
- vecm : Vector Error Correction Model
- wls : Weighted Least Squares

A.2 Tests

- addto : Add variables to specified model
- adf : Augmented Dickey-Fuller test
- arch : ARCH test
- $\bullet \ {\rm chow}$: Chow test
- coeffsum : Sum of coefficients
- coint : Engle-Granger cointegration test
- coint2 : Johansen cointegration test
- cusum : CUSUM test
- hausman : Panel diagnostics
- kpss : KPSS stationarity test
- leverage : Influential observations
- lmtest : LM tests (obsolete-replaced by modtest)
- meantest : Difference of means
- omit : Omit variables
- omitfrom : Omit variables from specified model

- qlrtest : Quandt likelihood ratio test
- reset : Ramseys RESET
- restrict : Linear restrictions
- runs : Runs test
- testuhat : Normality of residual
- vartest : Difference of variances
- vif : Variance Inflation Factors

A.3 Transformation

- diff : First differences
- discrete : Mark variables as discrete
- dummify : Create sets of dummies
- lags : Create lags
- ldiff : Log-differences
- logs : Create logs
- multiply : Multiply variables
- rhodiff : Quasi-differencing
- sdiff : Seasonal differencing
- square : Create squares of variables

A.4 Statistics

- corr : Correlation coefficients
- corrgm : Correlogram
- freq : Frequency distribution
- hurst : Hurst exponent
- mahal : Mahalanobis distances
- pca : Principal Components Analysis

- pergm : Periodogram
- spearman : Spearmanss rank correlation
- summary : Descriptive statistics
- xtab : Cross-tabulate variables

A.5 Dataset

- addobs : Add observations
- append : Append data
- data : Import from database
- delete : Delete variables
- genr : Generate a new variable
- import : Import data
- info : Information on data set
- labels : Print labels for variables
- nulldata : Creating a blank dataset
- open : Open a data file
- rename : Rename variables
- set info : Edit attributes of variable
- setobs : Set frequency and starting observation
- setmiss : Missing value code
- smpl : Set the sample range
- store : Save data
- transpos : Transpose data
- varlist : Listing of variables

A.6 Graphs

- boxplot : Boxplots
- gnuplot : Create a gnuplot graph
- graph : Create ASCII graph
- plot : ASCII plot
- rmplot : Range-mean plot
- scatters : Multiple pairwise graphs

A.7 Printing

- equprint : Print model as equation
- outfile : Direct printing to file
- print : Print data or strings
- printf : Formatted printing
- tabprint : Print model in tabular form Prediction
- fcast : Generate forecasts
- fcasterr : Forecasts with confidence intervals
- fit : Generate fitted values

A.8 Programming

- break : Break from loop
- else
- end : End block of commands
- \bullet endif
- endloop : End a command loop
- function : Define a function
- if
- include : Include function definitions

- loop : Start a command loop
- matrix : Define or manipulate matrices
- run : Execute a script
- set : Set program parameters

A.9 Utilities

- criteria : Model selection criteria
- critical : Critical values
- help : Help on commands
- $\bullet \mbox{ modeltab}$: The model table
- pvalue : Compute p-values
- quit : Exit the program
- shell : Execute shell commands

Appendix B

Some Basic Probability Concepts

In this chapter, you learned some basic concepts about probability. Since the actual values that economic variables take on are not actually known before they are observed, we say that they are *random*. Probability is the theory that helps us to express uncertainty about the possible values of these variables. Each time we observe the outcome of a random variable we obtain an observation. Once observed, its value is known and hence it is no longer random. So, there is a distinction to be made between variables whose values are not yet observed (random variables) and those whose values have been observed (observations). Keep in mind, though, an observation is merely one of many possible values that the variables can take. Another draw will usually result in a different value being observed.

A probability distribution is just a mathematical statement about the possible values that our random variable can take on. The probability distribution tells us the relative frequency (or probability) with which each possible value is observed. In their mathematical form probability distributions can be rather complicated; either because there are too many possible values to describe succinctly, or because the formula that describes them is complex. In any event, it is common summarize this complexity by concentrating on some simple numerical characteristics that they possess. The numerical characteristics of these mathematical functions are often referred to as parameters. Examples are the mean and variance of a probability distribution. The mean of a probability distribution describes the average value of the random variable over all of its possible realizations. Conceptually, there are an infinite number of realizations therefore parameters are not known to us. As econometricians, our goal is to try to estimate these parameters using a finite amount of information available to us. We collect a number of realizations (called a sample) and then estimate the unknown parameters using a *statistic*. Just as a parameter is an unknown numerical characteristic of a probability distribution, a statistic is an observable numerical characteristic of a sample. Since the value of the statistic will be different for each sample drawn, it too is a random variable. The statistic is used to gain information about the parameter.

Expected values are used to summarize various numerical characteristics of a probability dis-

tributions. For instance, if X is a random variable that can take on the values 0,1,2,3 and these values occur with probability 1/6, 1/3, 1/3, and 1/6, respectively. The average value or mean of the probability distribution, designated μ , is obtained *analytically* using its expected value.

$$\mu = E[X] = \sum xf(x) = 0 \cdot \frac{1}{6} + 1 \cdot \frac{1}{3} + 2 \cdot \frac{1}{3} + 3 \cdot \frac{1}{6} = \frac{3}{2}$$
(B.1)

So, μ is a parameter. Its value can be obtained mathematically if we know the probability density function of the random variable, X. If this probability distribution is known, then there is no reason to take samples or to study statistics! We can ascertain the mean, or average value, of a random variable without every firing up our calculator. Of course, in the real world we only know that the value of X is not known before drawing it and we don't know what the actual probabilities are that make up the density function, f(x). In order to Figure out what the value of μ is, we have to resort to different methods. In this case, we try to infer what it is by drawing a sample and estimating it using a statistic.

One of the ways we bridge the mathematical world of probability theory with the observable world of statistics is through the concept of a *population*. A statistical population is the collection of individuals that you are interested in studying. Since it is normally too expensive to collect information on everyone of interest, the econometrician collects information on a *subset* of this population–in other words, he takes a *sample*.

The population in statistics has an analogue in probability theory. In probability theory one must specify the set of all possible values that the random variable can be. In the example above, a random variable is said to take on 0,1,2, or 3. This set must be complete in the sense that the variable cannot take on any other value. In statistics, the population plays a similar role. It consists of the set that is relevant to the purpose of your inquiry and that is possible to observe. Thus it is common to refer to parameters as describing characteristics of populations. Statistics are the analogues to these and describe characteristics of the sample.

This roundabout discussion leads me to an important point. We often use the words mean, variance, covariance, correlation rather casually in econometrics, but their meanings are quire different depending on whether we are referring to a probability distribution or a sample. When referring to the analytic concepts of mean, variance, covariance, and correlation we are specifically talking about characteristics of a probability distribution; these can only be ascertained through complete knowledge of the probability distribution functions. It is common to refer to them in this sense as population mean, population variance, and so on. These concepts do not have anything to do with samples or observations!

In statistics we attempt to estimate these (population) parameters using samples and explicit formulae. For instance, we might use the average value of a sample to estimate the average value of the population (or probability distribution).
	Probability Distribution	Sample
mean	$E[X] = \mu$	$\frac{1}{n}\sum x_i = \bar{x}$
variance	$E[X - \mu]^2 = \sigma^2$	$\frac{1}{n-1}\sum (x_i - \bar{x})^2 = s_x^2$

When you are asked to obtain the mean or variance of random variables, make sure you know whether the person asking wants the characteristics of the probability distribution or of the sample. The former requires knowledge of the probability distribution and the later requires a sample.

In **gretl** you are given the facility to obtain sample means, variances, covariances and correlations. You are also given the ability to compute tail probabilities using the normal, t-, F and chisquare distributions. First we'll examine how to get summary statistics.

Summary statistics usually refers to some basic measures of the numerical characteristics of your sample. In **gret1**, summary statistics can be obtained in at least two different ways. Once your data are loaded into the program, you can select Data>Summary statistics from the pull-down menu. Which leads to the output in Figure B.2. The other way to get summary statistics is from

						<u> </u>		· · · · ·
🚺 gre	tl							
Eile	Tools	Data	⊻iew	<u>A</u> dd	<u>S</u> ample	Variable	Model	Help
cola.gdt <u>I</u> con view								
ID#	Variab	le name	Granh	snerifi	ed vars			
0	const		Multipl	e grapi	hs			
1	id		C. mana		kiekiee			
2	choice	.	Correl	ary sua ation m	usuus atrix			
3	price		Cross	Tabula	tion	1		
5	display	e /	Princip	al com	ponents	- 14		
	appia	,	<u>M</u> ahala	anobis	distances	- E		
			C <u>r</u> oss-	correla	gram			
							Undal	ed: Full range 1 - 5466
		-	Ħ	🔁 🔹	> 🥢	β 🔄		
	_							

Figure B.1: Choosing summary statistics from the pull-down menu

the console or script. Recall, **gretl** is really just a language and the GUI is a way of accessing that language. So, to speed things up you can do this. Load the dataset and open up a console window. Then type **summary**. This produces summary statistics for all variables in memory. If you just want summary statistics for a subset, then simply add the variable names after **summary**, i.e., **summary x** gives you the summary statistics for the variable **x**.

Gretl computes the sample mean, median, minimum, maximum, standard deviation (S.D.), coefficient of variation (C.V.), skewness and excess kurtosis for each variable in the data set. You

🙀 gretl: summary statistics								
K 🕹 🖬 (a) 18% ×							
	Summary Statis	stics, using the (observations 1 -	- 5466				
Variable	MEAN	MEDIAN	MIN	MAX				
id	911.50	911.50	1.0000	1822.0				
choice	0.33333	0.00000	0.00000	1.0000				
price	1.1851	1.1900	0.16000	2.9900				
feature	0.50878	1.0000	0.00000	1.0000				
display	0.36352	0.00000	0.00000	1.0000				
					≡			
Variable	s.D.	c.v.	SKEW	EXCSKURT				
id	526.01	0.57709	0.00000	-1.2000				
choice	0.47145	1.4143	0.70711	-1.5000				
price	0.30598	0.25818	0.24079	-0.34483				
feature	0.49997	0.98268	-0.035132	-1.9988				
display	0.48106	1.3233	0.56747	-1.6780				
					~			

Figure B.2: Choosing summary statistics from the pull-down menu yields these results.

may recall from your introductory statistics courses that there are an equal number of observations in your sample that are larger and smaller in value than the median. The standard deviation is the square root of your sample variance. The coefficient of variation is simply the standard deviation divided by the sample mean. Large values of the C.V. indicate that your mean is not very precisely measured. Skewness is a measure of the degree of symmetry of a distribution. If the left tail (tail at small end of the the distribution) extends over a relatively larger range of the variable than the right tail, the distribution is negatively skewed. If the right tail covers a larger range of values then it is positively skewed. Normal and t-distributions are symmetric and have zero skewness. The χ_n^2 is positively skewed. Excess kurtosis refers to the fourth sample moment about the mean of the distribution. 'Excess' refers to the kurtosis of the normal distribution, which is equal to three. Therefor if this number reported by **gret1** is positive, then the kurtosis is greater than that of the normal; this means that it is more peaked around the mean than the normal. If excess kurtosis is negative, then the distribution is flatter than the normal.

Sample Statistic	Formula	_
Mean	$\sum x_i/n = \bar{x}$	-
Variance	$\frac{1}{n-1}\sum (x_i - \bar{x})^2 = s_x^2$	
Standard Deviation	$s = \sqrt{s^2}$	You can also use gretl
Coefficient of Variation	$s/ar{x}$	
Skewness	$\frac{1}{n-1}\sum (x_i - \bar{x})^3 / s^3$	
Excess Kurtosis	$\frac{1}{n-1}\sum (x_i - \bar{x})^4 / s^4 - 3$	

to obtain tail prob-

abilities for various distributions. For example if $X \sim N(3,9)$ then $P(X \ge 4)$ is

$$P[X \ge 4] = P[Z \ge (4-3)/\sqrt{9}] = P[Z \ge 0.334] \doteq 0.3694$$
(B.2)

To obtain this probability, you can use the Tools>P-value finder from the pull-down menu. Then, give gret1 the value of X, the mean of the distribution and its standard deviation using the dialog box shown in Figure B.3. The result appears in Figure B.4. Gret1 is using the mean

Figure B.3: Dialog box for finding right hand side tail areas of various probability distributions.

📓 gretl: p-value finder 🛛 💷 🔀						
normal t chi-squ	uare F gamma binomial poisson					
mean std. deviation value	3 3 4					
	X Close					

Figure B.4: Results from the p value finder of $P[X \ge 4]$ where $X \sim N(3,9)$. Note, the area in the tail of this distribution to the right of 4 is .369441.

🛐 grett: p-value	
Standard normal: area to the right of 0.3333333 = 0.369441 (two-tailed value = 0.738883; complement = 0.261117)	

and standard deviation to covert the normal to a standard normal (i.e., z-score). As with nearly everything in gretl, you can use a script to do this as well. First, convert 4 from the $X \sim N(3,9)$ to a standard normal, $X \sim N(0,1)$. That means, subtract its mean, 3, and divide by its standard error, $\sqrt{9}$. The result is a scalar so, open a script window and type:

scalar z1 = (4-3)/sqrt(9)

Then use the cdf function to compute the tail probability of z1. For the normal cdf this is

scalar c1 = 1-cdf(z,z1)

The first argument of the cdf function, z, identifies the probability distribution and the second, z1, the number to which you want to integrate. So in this case you are integrating a standard normal cdf from minus infinity to z1=.334. You want the other tail (remember, you want the probability that Z is greater than 4) so subtract this value from 1.

In your book you are given another example $X \sim N(3,9)$ then find $P(4 \le X \le 6)$ is

$$P[4 \le X \le 6] = P[0.334 \le Z \le 1] = P[Z \le 1] - P[Z \le .33]$$
(B.3)

Take advantage of the fact that $P[Z \leq z] = 1 - P[Z > z]$ to obtain use the pvalue finder to obtain:

$$(1 - 0.1587) - (1 - 0.3694) = (0.3694 - 0.1587) = 0.2107$$
(B.4)

Note, this value differs slightly from the one given in your book due to rounding error that occurs from using the normal probability table. When using the table, the $P[Z \leq .334]$ was truncated to $P[Z \leq .33]$; this is because your tables are only taken out to two decimal places and a practical decision was made by the authors of your book to forgo interpolation (contrary to what your Intro to Statistics professor may have told you, it is hardly ever worth the effort to interpolate when you have to do it manually). **Gret1**, on the other hand computes this probability out to machine precision as $P[Z \leq \frac{1}{3}]$. Hence, a discrepancy occurs. Rest assured though that these results are, aside from rounding error, the same.

Using the cdf function makes this simple and accurate. The script is

```
scalar z1 = (4-3)/sqrt(9)
scalar z2 = (6-3)/sqrt(9)
scalar c1 = cdf(z,z1)
scalar c2 = cdf(z,z2)
scalar area = c2-c1
```

Appendix

Some Statistical Concepts

The hip data are used to illustrate computations for some simple statistics in your text.

C.1 Summary Statistics

Using a script or operating from the console, open the hip data, hip.gdt, and issue the summary command. This yields the results shown in Table C.1. This gives you the mean, median, minimum, maximum, standard deviation, coefficient of variation, skewness and excess kurtosis of your variable(s). Once the data are loaded, you can use gretl's language to generate these as well. For instance, genr hip_bar = mean(hip) yields the mean of the variable hip. To obtain the sample variance use genr s2hat = sum((hip-mean(hip)^2)/(\$nobs-1). The script below can be used to compute other summary statistics as discussed in your text.

```
open c:\userdata\gretl\data\poe\hip.gdt
summary
genr hip_bar = mean(hip)
genr s2hat = sum((hip-mean(hip))^2)/($nobs-1)
genr varYbar = s2hat/$nobs
genr sdYbar = sqrt(varYbar)
genr sig_tild = sqrt(sum((hip-mean(hip))^2)/($nobs))
genr mu3 = sum((hip-mean(hip))^3)/($nobs)
genr mu4 = sum((hip-mean(hip))^4)/($nobs)
```

Then, to estimate skewness, $S = \tilde{\mu}^3 / \tilde{\sigma}^3$, and kurtosis, $K = \tilde{\mu}^4 / \tilde{\sigma}^4$:

genr skew = mu3/sig_tild^3

Table C.1: Summary statistics from the hip data

```
? open c:\userdata\gretl\data\poe\hip.gdt
```

```
Read datafile c:\userdata\gretl\data\poe\hip.gdt periodicity: 1, maxobs: 50, observations range: 1-50
```

```
Listing 2 variables:
0) const 1) hip
```

? summary

Summary Statistics, using the observations 1 - 50
for the variable 'hip' (50 valid observations)

Mean	17.158
Median	17.085
Minimum	13.530
Maximum	20.400
Standard deviation	1.8070
C.V.	0.10531
Skewness	-0.013825
Ex. kurtosis	-0.66847

?

genr kurt = mu4/sig_tild^4

Note, in **gretl**'s built in summary command, the *excess* kurtosis is reported. The normal distribution has a theoretical kurtosis equal to 3 and the excess is measured relative to that. Hence, Excess $K = \tilde{\mu}^4 / \tilde{\sigma}^4 - 3$

If hip size in inches is normally distributed, $Y \sim N(\mu, \sigma^2)$. Based on our estimates, $Y \sim N(17.158, 3.265)$. The percentage of customers having hips greater than 18 inches can be estimated.

$$P(Y > 18) = P\left(\frac{Y - \mu}{\sigma} > \frac{18 - \mu}{\sigma}\right)$$
(C.1)

Replacing μ and σ by their estimates yields

genr zs = (18 - mean(hip))/sqrt(s2hat)
pvalue z zs

The last line actually computes the p-value associated with z-score. So, the pvalue command requests that a p-value be returned, the second argument (z) indicates the distribution to be used (in this case, z indicates the normal), and the final argument (zs) is the statistic itself, which is computed in the previous line.

C.2 Interval Estimation

Estimating a confidence interval using the hip data is also easy to do in **gretl**. Since the true variance, σ^2 , is not known, the t-distribution is used to compute the interval. The interval is

$$\bar{y} \pm t_c \frac{\hat{\sigma}}{\sqrt{N}} \tag{C.2}$$

where t_c is the desired critical value from the student-t distribution. In our case, N = 50 and the desired degrees of freedom for the t-distribution is N - 1 = 49. The **gretl** command **critical(t,49,.025** can be used to return the 0.025 critical value from the t_{49} distribution shown in Figure C.1 The computation is

```
open c:\userdata\gretl\data\poe\hip.gdt
genr s2hat = sum((hip-mean(hip))^2)/($nobs-1)
genr varYbar = s2hat/$nobs
genr sdYbar = sqrt(varYbar)
genr lb = mean(hip) - 2.01*sdYbar
genr ub = mean(hip) + 2.01*sdYbar
```

Figure C.1: Obtaining critical values from the t distribution using the console

```
gret1 console
gret1 console
gret1 console: type 'help' for a list of commands
cc = critical(t,49,.025)
Replaced scalar cc (ID 3) = 2.00958
?
```

which indicates that the interval [16.64,17.67] works 95% of the time. Note these numbers differ slightly from those in your book because we used 2.01 as our critical value. Hill et al. carry their critical value out to more decimal places and hence the difference. You can use **gretl**'s internal functions to improve accuracy. Replace 2.01 with critical(t,\$nobs-1,0.025) and see what happens!

genr lb = mean(hip) - critical(t,\$nobs-1,0.025)*sdYbar
genr ub = mean(hip) + critical(t,\$nobs-1,0.025)*sdYbar

C.3 Hypothesis Tests

Hypothesis tests are based on the same principles and use the same information that is used in the computation of confidence intervals. The first test is on the null hypothesis that hip size does not exceed 16.5 inches against the alternative that it does. Formally, $H_0: \mu = 16.5$ against the alternative $H_a: \mu > 16.5$. The test statistic is computed based on the sample average, \bar{Y} and is

$$t = \frac{Y - 16.5}{\hat{\sigma}/\sqrt{N}} \sim t_{N-1} \tag{C.3}$$

if the null hypothesis is true. Choosing the significance level, $\alpha = .05$, the right-hand side critical value for the t_{49} is 1.677. The average hip size is 17.1582 with standard deviation 1.807 so the test statistic is

$$t = \frac{17.1582 - 16.5}{1.807/\sqrt{50}} = 2.576 \tag{C.4}$$

The **gretl** code to produce this is:

```
open c:\userdata\gretl\data\poe\hip.gdt
genr s2hat = sum((hip-mean(hip))^2)/($nobs-1)
genr varYbar = s2hat/$nobs
genr sdYbar = sqrt(varYbar)
genr tstat = (mean(hip)-16.5)/(sdYbar)
scalar c = critical(t,49,0.025)
pvalue t 49 tstat
```

The scalar c = critical(t, 49, 0.025) statement can be used to get the $\alpha = 0.025$ critical value for the t distribution with 49 degrees of freedom. The next line, pvalue t 49 tstat, returns the p-value from the t distribution with 49 degrees of freedom for the computed statistic, tstat.

The two-tailed test is of the hypothesis, $H_0: \mu = 17$ against the alternative, $H_a: \mu \neq 17$.

$$t = \frac{\bar{Y} - 17}{\hat{\sigma}/\sqrt{N}} \sim t_{N-1} \tag{C.5}$$

if the null hypothesis is true. Choosing the significance level, $\alpha = .05$, the two sided critical value is ± 2.01 . Hence, you will reject the null hypothesis if t < -2.01 or if t > 2.01. The statistic is computed

$$t = \frac{17.1582 - 17}{1.807/\sqrt{50}} = .6191 \tag{C.6}$$

and you cannot reject the null hypothesis. The gretl code is:

```
genr tstat = (mean(hip)-17)/(sdYbar)
scalar c = critical(t,49,0.025)
pvalue t 49 tstat
```

C.4 Testing for Normality

Your book discusses the Jarque-Bera test for normality which is computed using the skewness and kurtosis of the least squares residuals. To compute the Jarque-Bera statistic, you'll first need to obtain the summary statistics from your data series.

From **gretl** script

```
open c:\userdata\gretl\data\poe\hip.gdt
summary
```

You could also use the point and click method to get the summary statistics. This is accomplished from the output window of your regression. Simply highlight the hip series and then choose Data>Summary statistics>selected variables from the pull-down menu. This yields the results in Table C.1.

One thing to note, **gretl** reports excess kurtosis rather than kurtosis. The excess kurtosis is measured relative to that of the normal distribution which has kurtosis of three. Hence, your computation is

$$JB = \frac{N}{6} \left(\text{Skewness}^2 + \frac{(\text{Excess Kurtosis})^2}{4} \right)$$
(C.7)

Which is

$$JB = \frac{50}{6} \left(-0.0138^2 + \frac{-0.66847^2}{4} \right) = .9325$$
(C.8)

Using the results in section C.1 for the computation of skewness and kurtosis, the gretl code is:

```
open c:\userdata\gretl\data\poe\hip.gdt
genr sig_tild = sqrt(sum((hip-mean(hip))^2)/($nobs))
genr mu3 = sum((hip-mean(hip))^3)/($nobs)
genr mu4 = sum((hip-mean(hip))^4)/($nobs)
genr skew = mu3/sig_tild^3
genr kurt = mu4/sig_tild^4
genr JB = ($nobs/6)*(skew^2+(kurt-3)^2/4)
pvalue X 2 JB
```

Appendix D

Using R with **gretl**

Another feature of **gretl** that makes it extremely powerful is its ability to work with another free program called R. R is actually a programming language for which many statistical procedures have been written. Although **gretl** is reasonably powerful, there are still many things that it won't do. The ability to export **gretl** data into R makes it possible to do some sophisticated analysis with relative ease.

Quoting from the R web site

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R.

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity.

One of R's strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.

R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.

R can be downloaded from http://www.r-project.org/ which is referred to as CRAN or the comprehensive R archive network. To install R, you'll need to download it and follow the instructions given at the CRAN web site. Also, there is an appendix in the gretl manual about using R that you may find useful. The remainder of this brief appendix assumes that you have R installed and linked to gretl through the programs tab in the File>Preferences>General pull down menu. Make sure that the 'Command to launch GNR R' box points to the RGui.exe file associated with your installation of R.

To illustrate, open the *food.gdt* data in **gretl**.

```
open c:\userdata\gretl\data\poe\food.gdt
```

Now, select Tools>start GNU R from the pull-down menu. The current gretl data set, in this case *food.gdt*, will be transported into R's required format. You'll see the R console which is shown in Figure D.1.

R Console	_ 0	×
		^
R version 2.6.0 (2007-10-03) Copyright (C) 2007 The R Foundation for Statistical Computing ISBN 3-900051-07-0		
R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'license()' or 'licence()' for distribution details.		
Natural language support but running in an English locale		
R is a collaborative project with many contributors. Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications.		
Type 'demo()' for some demos, 'help()' for on-line help, or 'help.start()' for an HTML browser interface to help. Type 'q()' to quit R.		
> # load data from gretl > library(stats)		
> vnum <- as.double(R.version\$major) + (as.double(R.version\$minor) / 10.0)	I	
> if (vnum > 2.41) library(utils)		
> gretldata <- read.table("C:/Documents and Settings/Lee/Application Data/	/gret1\$	
> attach(gretldata)		
		ì

Figure D.1: The *R* console when called from **Gret**

In some versions of **gretl** this may not work (a bug?). To load the data in properly, type the following at the command prompt in R.

```
gretldata <-
    read.table("C:/userdata/myfiles/Rdata.tmp", header = TRUE )</pre>
```

This assumes that you have set gretl's user directory to C:\userdata\myfiles using the dialog box shown in Figure (D.2). Tools; Preferences; General The addition of Header = TRUE to the code

Figure D.2: Use this dialog to set the default location for **gret1** files to be written and read.

📓 gretl: options 🛛 🔀				
General Databases Programs HCCME Manuals				
Main gretl directory c:\Program Files\gretl\ Browse				
Tell me about gretl updates 🛛 🗹 Use locale setting for decimal point				
Emulate Windows look 📃 Allow shell commands				
Show icon view automatically				
Language preference Automatic				
Apply Cancel OK				

that gretl writes for you ensures that the variable names, which are included on the first row of the Rdata.tmp, get read into R properly. Then, to run the regression in R.

```
fitols <- lm(y~x,data=gretldata)</pre>
```

Figure D.3: The lm(y x, data=gretldata) command estimates a linear regression model with y as the dependent variable and x as an independent variable. *R* automatically includes an intercept. To print the results to the screen, you have to use the summary(anov) command.

	🔣 R Console	٦		2
	> summary(fitols)	_		ĺ
	Call: lm(formula = y ~ x, data = gretldata)			
	Residuals:			
	Min 10 Median 30 Max -223.025 -50.816 -6.324 67.879 212.044			
	Coefficients:			
	Estimate Std. Error t value Pr(> t) (Intercept) 83.416 43.410 1.922 0.0622.			
	x 10.210 2.093 4.877 1.95e-05 ***			
	Signif. codes: 0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1			
	Residual standard error: 89.52 on 38 degrees of freedom Multiple R-Squared: 0.385, Adjusted R-squared: 0.3688			
	F-statistic: 23.79 on 1 and 38 DF, p-value: 1.946e-05			ſ
				ŀ
1			~ ~	

Before going further, let me comment on this terse piece of computer code. First, in R the symbol <- is used as the assignment operator; it assigns whatever is on the right hand side $(lm(y\sim x, data=gretldata))$ to the name you specify on the left (fitols). it can be reversed

-> if you want to call the object to its right what is computed on its left. Also, R does not bother to print results unless you ask for them. This is handier than you might think, since most programs produce a lot more output than you actually want and must be coerced into printing less. The 1m command stands for 'linear model' and in this example it contains two arguments within the parentheses. The first is your simple regression model. The dependent variable is y and the independent variable x. They are separated by the symbol which substitutes in this case for an equals sign. The other argument points to the data set that contains these two variables. This data set, pulled into R from gret1, is by default called gret1data. There are other options for the 1m command, and you can consult the substantial pdf manual to learn about them. In any event, you'll notice that when you enter this line and press the return key (which executes this line) Rresponds by issuing a command prompt, and no results! To print the results from your regression, you issue the command:

summary.lm(fitols)

which yields the output shown in Figure D.4. Then, to obtain the ANOVA table for this regression

anova(fitols)

This gives the result in Figure D.4. It's that simple! One thing to note about how R reports

Figure D.4: The anova(olsfit) command asks *R* to print the anova table for the regression results stored in olsfit._____

R Console	_ 🗆 🔀
	^
> anova(fitols)	
Analysis of Variance Table	
Response: y Df Sum Sq Mean Sq F value Pr(>F) x 1 190627 190627 23.789 1.946e-05 *** Residuals 38 304505 8013 	
Signif. codes: 0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1	_
>1	~
	> .;;

analysis of variance. It reports the explained variation (190627) in the top line and the unexplained variation in y (304505) below. It does not report total variation. To obtain the total, you just have to add the explained to the unexplained variation together (190627+304505=495132).

To do multiple regression in R, you have to put each of your independent variables (other than the intercept) into a matrix. A matrix is a rectangular array (which means it contains numbers arranged in rows and columns). You can think of a matrix as the rows and columns of numbers that appear in a spreadsheet program like MS Excel. Each row contains an observation on each of your independent variables; each column contains all of the observations on a particular variable. For instance suppose you have two variables, x1 and x2, each having 5 observations. These can be combined horizontally into the matrix, X. Computer programmers sometimes refer to this operation as *horizontal concatenation*. Concatenation essentially means that you connect or link objects in a series or chain; to concatenate horizontally means that you are binding one or more columns of numbers together.

The function in R that binds columns of numbers together is cbind. So, to horizontally concatenate x1 and x2 use the command

 $X \leq cbind(x1,x2)$

which takes

$$x1 = \begin{pmatrix} 2\\1\\5\\2\\7 \end{pmatrix}, \quad x2 = \begin{pmatrix} 4\\2\\1\\3\\1 \end{pmatrix}, \quad \text{and yields } X = \begin{pmatrix} 2&4\\1&2\\5&1\\2&3\\7&1 \end{pmatrix}$$

Then the regression is estimated using

fitols <- lm(y~X)</pre>

There is one more thing to mention about R that is very important and this example illustrates it vividly. R is case sensitive. That means that two objects x and X can mean two totally different things to R. Consequently, you have to be careful when defining and calling objects in R to get to distinguish lower from upper case letters.

D.1 Packages

The following is section is taken with very minor changes from Venables et al. (2006).

All R functions and datasets are stored in packages. Only when a package is loaded are its contents available. This is done both for efficiency (the full list would take more memory and would take longer to search than a subset), and to aid package developers, who are protected from name clashes with other code. The process of developing packages is described in section Creating R packages in Writing R Extensions. Here, we will describe them from a users point of view. To see which packages are installed at your site, issue the command library() with no arguments. To load a particular package (e.g., the MCMCpack package containing functions for estimating models in Chapter 16

> library(MCMCpack)

If you are connected to the Internet you can use the install.packages() and update.packages() functions (both available through the Packages menu in the Windows GUI). To see which packages are currently loaded, use

> search()

to display the search list.

To see a list of all available help topics in an installed package, use

> help.start()

to start the HTML help system, and then navigate to the package listing in the Reference section.

D.2 Stata Datasets

With R you can read in datasets in many different formats. Your textbook includes a dataset written in Stata's format and R can both read and write to this format. To read and write Stata's .dta files, you'll have to load the **foreign** package using the library command:

library(foreign)

Then, type

nels <- read.dta("c:/DATA/Stata/nels_small.dta")</pre>

and the dataset will be read directly into R. There are two things to note, though. First, the slashes in the filename are backwards from the Windows convention. Second, you need to point to the file in your directory structure and enclose the path/filename in double quotes. R looks for the the file where you've directed it and, provided it finds it, reads it into memory. It places the variable names from Stata into the object. Then, to retrieve a variable from the object you create (called in this example, data, use the syntax

pse <- nels\$psechoice</pre>

Now, you have created a new object called **pse** that contains the variable retrieved from the **nels** object called **psechoice**. This seems awkward at first, but believe it or not, it becomes pretty intuitive after a short time.

The command

attach(nels)

will take each of the columns of **nels** and allow you to refer to them by their variable names. So, instead of referring to **nels\$psechoice** you can directly ask for **psechoice** without using the **nels\$** prefix. For complex programs, using **attach()** may lead to unexpected results. If in doubt, it is probably a good idea to forgo this option. If you do decide to use it, you can later undo it using **detach(nels)**.

D.3 Final Thoughts

A very brief, but useful document can be found at http://cran.r-project.org/doc/contrib/ Farnsworth-EconometricsInR.pdf. This is a guide written by Grant Farnsworth about using R in econometrics. He gives some alternatives to using MCMCpack for the models discussed in Chapter 16.

Appendix E

Errata and Updates

- 2007-12-16 Page 40. Syntax for pvalue(t,\$df,t2) is fixed. This applies the script at the end of chapter 3 as well. Thanks to Greg Coleman.
- 2007-12-16 A new version of POEscripts.exe has been uploaded to the website, www.learneconometrics. com/gretl.
- 2008-5-30 The variable names in the script for chapter 12 do not match those in the dataset provided. FedFunds is F and Bonds is designated B. This will be fixed, eventually. Thanks to Peter Robertson for this and the following corrections.
- 2008-5-30 Page 67: phrase 'in tablular form' changed to 'in tabular form'. Peter Robertson.
- 2008-5-30 Page 68: phrase 'Anaylsis>ANOVA' changes to 'Analysis>ANOVA'. Peter Robertson.
- 2008-5-30 Page 73: 'it's' changed to 'its'. Peter Robertson.
- 2008-5-30 Page 93: 'were' changed to 'where'. Peter Robertson.
- 2008-5-30 Page 185: 'redidual' supposed to be 'residual. Peter Robertson.
- 2008-5-30 Page 196: 'and ARCH option has been added' changed to 'an ARCH option has been added...' Peter Robertson.
- 2008-5-30 Page 254: 'MCMCpack can also be use to . . .' changed to 'MCMCpack can also be used to . . .' Peter Robertson.
- 2008-7-23 The variable names in the script for chapter 12 now match those in the data file. I chose the longer, more descriptive variable names rather than the short ones used in POE. FedFunds is F and Bonds is B in POE. Thanks to Peter Robertson for this and the following corrections.
- 2008-7-23 New screen shots were generated for Chapter 12 to match the new variable names in the dataset. A new figure (Figure 12.15) was added to reflect changes made to the GUI in version 1.7.5 of gretl.

- 2008-9-26 In chapter 1 I added a figure (Figure 1.8) for the very useful function reference and updated most of the screen shots (version 1.7.8).
- 2008-9-26 Updated screen shots in chapter 2.
- 2009-1-17 gretl uses a new symbol to designate the logical 'and' (changes from | to ||. The script in chapter 15 is changed to reflect this. Thanks to Michel Pouchain.
- 2009-6-22 Section 16.2 was rewritten using **gretl**'s new multinomial logit function. The previous version of mnl.inp was rewritten as well and it now replicates the marginal effects in *POE* almost exactly.
- 2009-6-22 To improve readability and to reduce printing costs, the book was recompiled using LaTeX's fullpage package.
- 2009-7-20 Some of the screen shots were updated to reflect recent changes in gretl. Also, Chapter 10 was revised slightly to reflect changes in gretl.
- 2010-5-20 In chapter 2 (see page 29) there was a bug in the Monte Carlo script. There is not supposed to be a space between the double slashes and progressive, -progressive. I also added the -quiet option.
- 2010-5-20 Replaced the obsolete lmtest command with modtest.

Appendix

GNU Free Documentation License

Version 1.2, November 2002

Copyright © 2000,2001,2002 Free Software Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf

of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Ac-knowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/. Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

Bibliography

- Barro, Robert J. and Jong Wha Lee (1996), 'International measures of schooling years and schooling quality', *American Economic Review* 82(2), 218–223.
- Cottrell, Allin and Riccardo Jack Lucchetti (2007), *Gretl User's Guide*, Department of Economics and Wake Forest University and Dipartimento di Economia Università Politecnica delle Marche, http://ricardo.ecn.wfu.edu/pub//gretl/manual/PDF/gretl-guide.pdf.
- Doornik, J. A. and H. Hansen (1994), 'An omnibus test for univariate and multivariate normality', working paper, Nuffield College, Oxford.
- Greene, William H. (2003), Econometric Analysis, 5th edn, Prentice Hall, Upper Saddle River, N.J.
- Grunfeld, Yehuda (1958), The Determinants of Corporate Investment, PhD thesis, University of Chicago.
- Heckman, James J. (1979), 'Sample selection bias as a specification error', *Econometrica* **47**(1), 153–161.
- Hill, R. Carter, William E. Griffiths and Guay Lim (2007), Principles of Econometrics, third edn, John Wiley and Sons.
- Koop, Gary (2003), Bayesian Econometrics, John Wiley & Sons, Hoboken, NJ.
- Lancaster, Tony (2004), An Introduction to Modern Bayesian Econometrics, Blackwell Publishing, Ltd.
- Mixon Jr., J. Wilson and Ryan J. Smith (2006), 'Teaching undergraduate econometrics with gretl', Journal of Applied Econometrics 21, 1103–1107.
- Ramanathan, Ramu (2002), Introductory Econometrics with Applications, The Harcourt series in economics, 5th edn, Harcourt College Publishers, Fort Worth.
- Stock, James H. and Mark W. Watson (2006), Introduction to Econometrics, second edn, Addison Wesley, Boston, MA.
- Venables, W. N., D. M. Smith and R Development Core Team (2006), 'An introduction to R'.