

EViews 4 User's Guide



Quantitative Micro Software

EViews 4 User's Guide

Copyright © 1994–2002 Quantitative Micro Software, LLC

All Rights Reserved

Printed in the United States of America

ISBN 1-880411-28-8

Revised for EViews 4.1 - February 2002

This software product, including program code and manual, is copyrighted, and all rights are reserved by Quantitative Micro Software, LLC. The distribution and sale of this product are intended for the use of the original purchaser only. Except as permitted under the United States Copyright Act of 1976, no part of this product may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of Quantitative Micro Software.

Disclaimer

The authors and Quantitative Micro Software assume no responsibility for any errors that may appear in this manual or the EViews program. The user assumes all responsibility for the selection of the program to achieve intended results, and for the installation, use, and results obtained from the program.

Trademarks

Windows, Windows 95/98/2000/NT/Me, and Microsoft Excel are trademarks of Microsoft Corporation. PostScript is a trademark of Adobe Corporation. X11.2 and X12-ARIMA Version 0.2.7 are seasonal adjustment programs developed by the U. S. Census Bureau. Tramo/Seats is copyright by Agustin Maravall and Victor Gomez. All other product names mentioned in this manual may be trademarks or registered trademarks of their respective companies.

Quantitative Micro Software, LLC

4521 Campus Drive, #336, Irvine CA, 92612-2699

Telephone: (949) 856-3368

Fax: (949) 856-2044

e-mail: sales@eviews.com

web: www.eviews.com

March 11, 2002

Table of Contents

PREFACE	1
PART I. EViews FUNDAMENTALS	3
CHAPTER 1. INTRODUCTION	5
What Is EViews?	5
Installing and Running EViews	5
Windows Basics	6
The EViews Window	9
Closing EViews	12
Where To Go For Help	12
CHAPTER 2. A DEMONSTRATION	15
Creating a Workfile and Importing Data	15
Verifying the Data	18
Examining the Data	20
Estimating a Regression Model	22
Specification and Hypothesis Tests	24
Modifying the Equation	26
Forecasting from an Estimated Equation	28
Additional Issues	31
CHAPTER 3. EViews BASICS	33
Workfile Basics	33
Object Basics	41
Commands	53
For More Info.... ..	54
CHAPTER 4. BASIC DATA HANDLING	55
Samples	60
Importing Data	64
Exporting Data	70
Frequency Conversion	72
Commands for Basic Data Handling	75
Addendum: Reading ASCII Files	76
Addendum: Matrix Object Reading and Writing	84

CHAPTER 5. WORKING WITH DATA	87
Using Expressions87
Working with Series94
Working with Auto-series98
Working with Groups of Series102
Working with Scalars105
CHAPTER 6. EViews DATABASES	107
An Overview107
Database Basics108
Working with Objects in Databases113
Database Auto-Series119
The Database Registry121
Querying the Database123
Object Aliases and Illegal Names131
Maintaining the Database133
Foreign Format Databases135
Working with DRIPro Links141
Commands147
PART II. BASIC DATA ANALYSIS	149
CHAPTER 7. SERIES	151
Series Views151
Spreadsheet and Graph Views151
Descriptive Statistics152
Tests for Descriptive Stats156
Distribution Graphs164
One-Way Tabulation166
Correlogram167
Unit Root Test170
BDS Test170
Conversion Options173
Label174
Series Procs174
Generate by Equation174
Resampling175

Seasonal Adjustment	177
Exponential Smoothing	190
Hodrick-Prescott Filter	195
Commands	196
CHAPTER 8. GROUPS	199
Views from a Group Window	199
Group Members	199
Spreadsheet	199
Dated Data Table	200
Graphs	209
Descriptive Statistics	214
N-Way Tabulation	216
Principal Components	219
Correlations, Covariances, and Correlograms	221
Cross Correlations and Correlograms	221
Cointegration Test	222
Granger Causality	222
Label	223
Group Procedures	223
Commands	224
CHAPTER 9. STATISTICAL GRAPHS USING SERIES AND GROUPS	225
Distribution Graphs of Series	225
Scatter Diagrams with Fit Lines	233
Commands	242
CHAPTER 10. GRAPHS, TABLES, AND TEXT OBJECTS	243
Creating Graphs	243
Modifying Graphs	244
Multiple Graphs	250
Printing Graphs	252
Copying Graphs to Other Windows Programs	252
Graph Commands	253
Tables	253
Copying Tables to Other Windows Programs	254
Text Objects	255
Commands	255

PART III. BASIC SINGLE EQUATION ANALYSIS	257
CHAPTER 11. BASIC REGRESSION	259
Equation Objects	259
Specifying an Equation in EViews	260
Estimating an Equation in EViews	263
Equation Output	265
Working with Equations	272
Estimation Problems	276
Commands	277
CHAPTER 12. ADDITIONAL REGRESSION METHODS	279
Weighted Least Squares	279
Heteroskedasticity and Autocorrelation Consistent Covariances	281
Two-stage Least Squares	283
Nonlinear Least Squares	289
Generalized Method of Moments (GMM)	297
Commands	301
CHAPTER 13. TIME SERIES REGRESSION	303
Serial Correlation Theory	303
Testing for Serial Correlation	304
Estimating AR Models	307
ARIMA Theory	311
Estimating ARIMA Models	313
Diagnostic Evaluation	322
Polynomial Distributed Lags (PDLs)	323
Nonstationary Time Series	328
Unit Root Tests	329
Commands	341
CHAPTER 14. FORECASTING FROM AN EQUATION	343
Forecasting from Equations in EViews	343
Illustration	345
Forecast Basics	349
Forecasts with Lagged Dependent Variables	355
Forecasting with ARMA Errors	356
Forecasting Equations with Formulas	359

Forecasting with Nonlinear and PDL Specifications	364
Commands	364
CHAPTER 15. SPECIFICATION AND DIAGNOSTIC TESTS	367
Background	367
Types of Tests	367
Residual Tests	375
Specification and Stability Tests	379
Applications	389
Commands	393
PART IV. ADVANCED SINGLE EQUATION ANALYSIS	395
CHAPTER 16. ARCH AND GARCH ESTIMATION	397
The ARCH Specification	397
Estimating ARCH models in EViews	400
Working with ARCH Models	404
Asymmetric ARCH Models	407
The Component ARCH Model	412
Estimating and Interpreting Models in EViews	413
Examples	415
Commands	418
CHAPTER 17. DISCRETE AND LIMITED DEPENDENT VARIABLE MODELS	421
Binary Dependent Variable Models	421
Estimating Binary Models in EViews	423
Procedures for Binary Equations	433
Ordered Dependent Variable Models	438
Estimating Ordered Models in EViews	439
Views of Ordered Equations	442
Procedures for Ordered Equations	443
Censored Regression Models	444
Estimating Censored Models in EViews	445
Procedures for Censored Equations	449
Truncated Regression Models	454
Procedures for Truncated Equations	455
Count Models	458
Views of Count Models	462

Procedures for Count Models	462
Demonstrations	463
Commands	467
Technical Notes	467
CHAPTER 18. THE LOG LIKELIHOOD (LOGL) OBJECT	471
Overview	471
Specification	473
Estimation	479
LogL Views	480
LogL Procs	481
Troubleshooting	483
Limitations	484
Examples	486
PART V. MULTIPLE EQUATION ANALYSIS	493
CHAPTER 19. SYSTEM ESTIMATION	495
Background	495
System Estimation Methods	496
How to Create and Specify a System	498
Working With Systems	506
Commands	510
Technical Discussion	511
CHAPTER 20. VECTOR AUTOREGRESSION AND ERROR CORRECTION MODELS	519
Vector Autoregressions (VARs)	519
How to Estimate a VAR	520
VAR Estimation Output	521
Views and Procs of a VAR	522
Structural (Identified) VARs	531
Cointegration Test	537
Vector Error Correction (VEC) Models	547
A Note on EViews Backward Compatibility	550
CHAPTER 21. POOLED TIME SERIES, CROSS-SECTION DATA	551
Creating a Workfile for Pooled Data	551
The Pool Object	551
Importing Pooled Data	553

Exporting Pooled Data	558
Working with Pooled Data	558
Pooled Estimation	562
Commands	571
Technical Discussion	571
CHAPTER 22. STATE SPACE MODELS AND THE KALMAN FILTER	577
Background	577
Specifying a State Space Model in EViews	582
Working with the State Space	593
Converting from Version 3 Sspace	599
Commands	599
Technical Discussion	600
CHAPTER 23. MODELS	601
Overview	601
An Example Model	604
Building a Model	618
Working with the Model Structure	620
Specifying Scenarios	625
Using Add Factors	626
Solving the Model	629
Working with the Model Data	640
Commands	645
APPENDIX A. GLOBAL OPTIONS	647
Setting Options	647
APPENDIX B. DATE FORMATS	653
Date Specification	653
Implicit Dating	654
Special Date Functions	655
Examples	655
APPENDIX C. WILDCARDS	657
Wildcard Expressions	657
Using Wildcard Expressions	657
Source and Destination Patterns	658
Resolving Ambiguities	659

Wildcard versus Pool Identifier	660
APPENDIX D. ESTIMATION ALGORITHMS AND OPTIONS	663
Optimization Algorithms	663
Setting Estimation Options	666
Nonlinear Equation Solution Methods	671
APPENDIX E. GRADIENTS AND DERIVATIVES	675
Gradients	675
Derivatives	678
APPENDIX F. INFORMATION CRITERIA	683
Definitions	683
Using Information Criteria as a Guide to Model Selection	684
REFERENCES	685
INDEX	693

Preface

This manual describes the interactive use of EViews, a program for statistical and econometric analysis, and forecasting. For details on the EViews command language, as well as a description of the programming and matrix languages, we refer you to the companion volume—the EViews *Command and Programming Reference*.

The manual is divided into five parts:

- [Part I. “EViews Fundamentals” beginning on page 3](#)—introduces you to the basics of using EViews. In addition to a discussion of basic Windows operations, we explain how to use EViews to manage your data.
- [Part II. “Basic Data Analysis” beginning on page 149](#)—describes the use of EViews to perform basic analysis of data and to draw graphs and create tables describing your data.
- [Part III. “Basic Single Equation Analysis” on page 257](#)—discusses standard regression analysis: ordinary least squares, weighted least squares, two-stage least squares, nonlinear least squares, time series analysis, specification testing and forecasting.
- [Part IV. “Advanced Single Equation Analysis” beginning on page 395](#)—documents autoregressive conditional heteroskedasticity (ARCH) models, discrete and limited dependent variable models, and user specified likelihood estimation.
- [Part V. “Multiple Equation Analysis” on page 493](#)—describes estimation and forecasting with systems of equations, vector autoregression and error correction models, state space models, pooled cross-section/time series data, and model solution.

You should not feel a need to read the manual from cover-to-cover in order to use EViews. We recommend, however, that you glance at most of Part I to gain familiarity with the basic concepts and operation of the program. At a minimum, you should look over Chapters 1–3, especially the extended demonstration in [Chapter 2, “A Demonstration”, on page 15](#).

Part I. EViews Fundamentals

The following chapters document the fundamentals of working with EViews:

- The first three chapters contain introductory material. [Chapter 1, “Introduction”](#) describes the basics of installing EViews. [Chapter 3, “EViews Basics”](#) provides an overview of EViews basics. [Chapter 2, “A Demonstration”](#) guides you through a typical EViews session, introducing you to the basics of working with EViews.
- [Chapter 4, “Basic Data Handling”](#) and [Chapter 5, “Working with Data”](#) document the basics of working with data. We describe methods of getting your data into EViews, using the built-in tools to manipulate and manage your data, and exporting your data into spreadsheets, text files and other Windows applications.
- [Chapter 6, “EViews Databases”](#) contains more advanced material, describing the EViews database features and advanced data handling features.

We recommend that you read through most of the material in this section before beginning serious work with EViews. In particular, we suggest that you read the first three chapters prior to using the program.

Chapter 1. Introduction

What Is EViews?

EViews provides sophisticated data analysis, regression, and forecasting tools on Windows-based computers. With EViews you can quickly develop a statistical relation from your data and then use the relation to forecast future values of the data. Areas where EViews can be useful include: scientific data analysis and evaluation, financial analysis, macroeconomic forecasting, simulation, sales forecasting, and cost analysis.

EViews is a new version of a set of tools for manipulating time series data originally developed in the Time Series Processor software for large computers. The immediate predecessor of EViews was MicroTSP, first released in 1981. Though EViews was developed by economists and most of its uses are in economics, there is nothing in its design that limits its usefulness to economic time series. Even quite large cross-section projects can be handled in EViews.

EViews provides convenient visual ways to enter data series from the keyboard or from disk files, to create new series from existing ones, to display and print series, and to carry out statistical analysis of the relationships among series.

EViews takes advantage of the visual features of modern Windows software. You can use your mouse to guide the operation with standard Windows menus and dialogs. Results appear in windows and can be manipulated with standard Windows techniques.

Alternatively, you may use EViews' powerful command and batch processing language. You can enter and edit commands in the command window. You can create and store the commands in programs that document your research project for later execution.

Installing and Running EViews

Your copy of EViews 4.0 is distributed on a single CD-ROM. Installation is straightforward—simply insert your CD-ROM disc into a drive, wait briefly while the disc spins-up and the setup program launches, and then simply follow the prompts. If the disc does not spin-up, navigate to the drive using Windows Explorer, then click on the Setup icon.

We have also provided more detailed installation instructions in a separate sheet that you should have received with your EViews package. If you did not receive this sheet, please contact our office, or see our website: <http://www.eviews.com>.

Windows Basics

In this section, we provide a brief discussion of some useful techniques, concepts, and conventions that we will use in this manual. We urge those who desire more detail to obtain one of the (many) very good books on Windows.

The Mouse

EViews uses both buttons of the standard Windows mouse. Unless otherwise specified, when we say that you should *click* on an item, we mean a single click of the left mouse-button. *Double click* means to click the left mouse-button twice in rapid succession. We will often refer to *dragging* with the mouse; this means that you should click and hold the left mouse-button down while moving the mouse.

Window Control

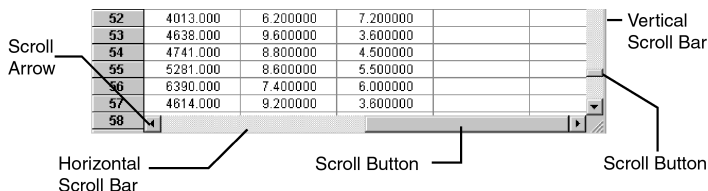
As you work, you may find that you wish to change the size of a window or temporarily move a window out of the way. Alternatively, a window may not be large enough to display all of your output, so that you want to move within the window in order to see relevant items. Windows provides you with methods for performing each of these tasks.

Changing the Active Window

When working in Windows, you may find that you have a number of open windows on your screen. The *active* (top-most) *window* is easily identified since its title bar will generally differ (in color and/or intensity) from the inactive windows. You can make a window active by clicking anywhere in the window, or by clicking on the word **Window** in the main menu, and selecting the window by clicking on its name.

Scrolling

Windows provides both horizontal and vertical *scroll bars* so that you can view information which does not fit inside the window (when all of the information in a window fits inside the viewable area, the scroll bars will be hidden).



The scroll box indicates the overall relative position of the window and the data. Here, the vertical scroll box is near the bottom, indicating that the window is showing the lower portion of our data. The size of the box also changes to show you the relative sizes of the

amount of data in the window and the amount of data that is off screen. Here, the current display covers roughly half of the horizontal contents of the window.

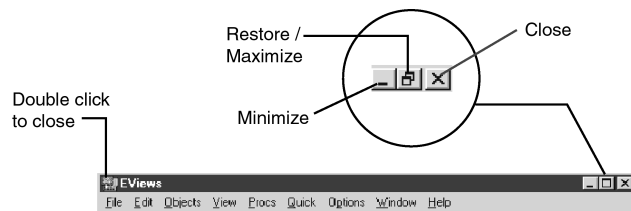
Clicking on the up, down, left, or right scroll arrows on the scroll bar will scroll the display one line in that direction. Clicking on the scroll bar on either side of a scroll box moves the information one screen in that direction.

If you hold down the mouse button while you click on or next to a scroll arrow, you will scroll continuously in the desired direction. To move quickly to any position in the window, drag the scroll box to the desired position.

Minimize/Maximize/Restore/Close

There may be times when you wish to move EViews out of the way while you work in another Windows program. Or you may wish to make the EViews window as large as possible by using the entire display area.

In the upper right-hand corner of each window, you will see a set of buttons which control the window display:



By clicking on the middle (Restore/Maximize) button, you can toggle between using your entire display area for the window, and using the original window size. *Maximize* (☐) uses your entire monitor display for the application window. *Restore* (☐) returns the window to its original size, allowing you to view multiple windows. If you are already using the entire display area for your window, the middle button will display the icon for restoring the window, otherwise it will display the icon for using the full screen area.

You can *minimize* your window by clicking on the minimize button in the upper right-hand corner of the window. To *restore* a program that has been minimized, click on the icon in your taskbar.

Lastly, the *close* button provides you with a convenient method for closing the window. To close all of your open EViews windows, you may also select **Window** in the main menu, and either **Close All**, or **Close All Objects**.

Moving and Resizing

You can move or change the size of the window (if it is not maximized or minimized). To move your window, simply click on the title bar (the top of your application window) and drag the window to a new location. To resize, simply put the cursor on one of the four sides or corners of the window. The cursor will change to a double arrow. Drag the window to the desired size, then release the mouse button.

Selecting and Opening Items

To select a single item, you should place the pointer over the item and single click. The item will now be highlighted. If you change your mind, you can change your selection by clicking on a different item, or you can cancel your selection by clicking on an area of the window where there are no items.

You can also select multiple items:

- To select sequential items, click on the first item you want to select, then drag the cursor to the last item you want to select and release the mouse button. All of the items will be selected. Alternatively, you can click on the first item, then hold down the SHIFT key and click on the last item.
- To select non-sequential items, click on the first item you want to select, then while holding the CTRL key, click on each additional item.
- You can also use CTRL-click to “unselect” items which have already been selected. In some cases it may be easier first to select a set of sequential items and then to unselect individual items.

Double clicking on an item will usually open the item. If you have multiple items selected, you can double click anywhere in the highlighted area.

Menus and Dialogs

Windows commands are accessed via *menus*. Most applications contain their own set of menus, which are located on the menu bar along the top of the application window. There are generally drop-down menus associated with the items in the main menu bar.

For example, the main EViews menu contains:



Selecting **File** from this menu will open a drop-down menu containing additional commands. We will describe the EViews menus in greater detail in the coming sections.

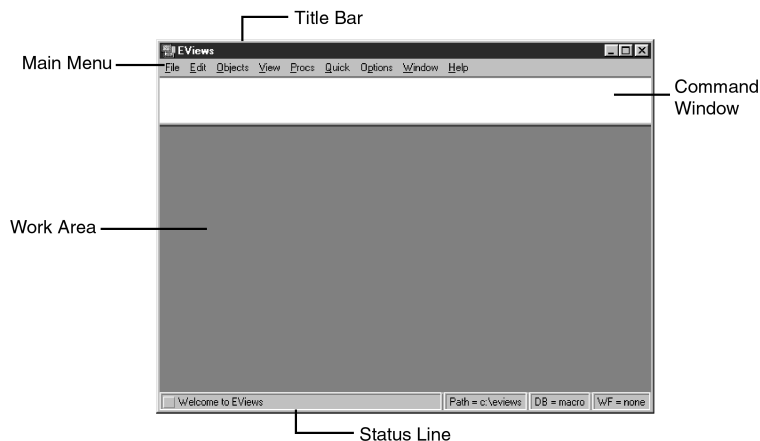
There are a few conventions which Windows uses in its menus that are worth remembering:

- A grayed-out command means the command is not currently available.
- An ellipse (...) following the command means that a dialog box (prompting you for additional input) will appear before the command is executed.
- A right-triangle (▸) means that additional (cascading) menus will appear if you select this item.
- A check mark (✓) indicates that the option listed in the menu is currently in effect. If you select the item again, the option will no longer be in effect and the check mark will be removed. This behavior will be referred to as *toggling*.
- Most menu items contain underlined characters representing keyboard shortcuts. You can use the keyboard shortcuts to the commands by pressing the ALT key, and then the underlined character. For example, ALT-F in EViews brings up the **F**ile drop-down menu.
- If you wish to close a menu without selecting an item, simply click on the menu name, or anywhere outside of the menu. Alternatively, you can press the ESC key.

We will often refer to entering information in *dialogs*. Dialogs are boxes that prompt for additional input when you select certain menu items. For example, when you select the menu item to run a regression, EViews opens a dialog prompting you for additional information about the specification, and often suggests default values for arguments. You can always tell when a menu item opens a dialog by the ellipses in the drop-down menu entry.

The EViews Window

If the program is installed correctly, you should see the EViews window when you launch the program. This is what the EViews window looks like:



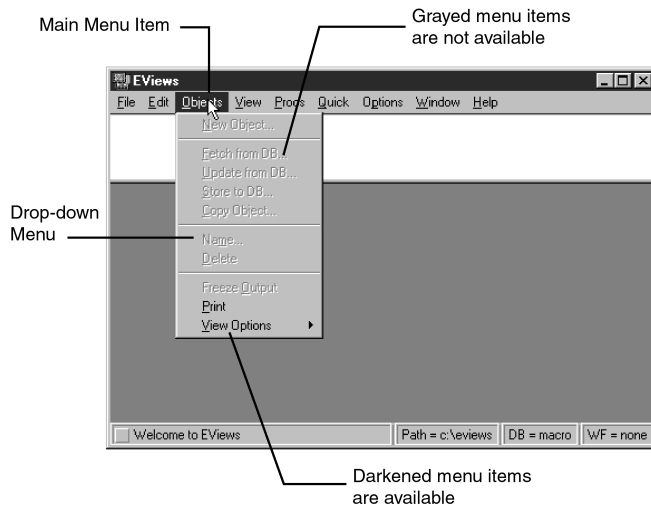
You should familiarize yourself with the following main areas in the EViews window.

The Title Bar

The *title bar*, labeled **EViews**, is at the very top of the main window. When EViews is the active program in Windows, the title bar has a color and intensity that differs from the other windows (generally it is darker). When another program is active, the EViews title bar will be lighter. If another program is active, EViews may be made active by clicking anywhere in the EViews window or by using ALT-TAB to cycle between applications until the EViews window is active.

The Main Menu

Just below the title bar is the *main menu*. If you move the cursor to an entry in the main menu and click on the left mouse button, a *drop-down menu* will appear. Clicking on an entry in the drop-down menu selects the highlighted item.

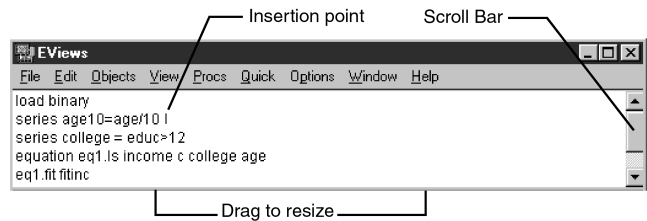


For example, here we click on the Objects entry in the main menu to reveal a drop-down menu. Notice that some of the items in the drop-down menu are listed in black and others are in gray. In menus, black items may be executed while the gray items are not available. In this example, you cannot create a **New Object** or **Store** an object, but you can **Print** and **View Options**. We will explain this behavior in our discussion of [“The Object Window” on page 46](#).

The Command Window

Below the menu bar is an area called the *command window*. EViews commands may be typed in this window. The command is executed as soon as you hit ENTER.

The vertical bar in the command window is called the *insertion point*. It shows where the letters that you type on the keyboard will be placed. As with standard word processors, if you have typed



something in the command area, you can move the insertion point by pointing to the new location and clicking the mouse. If the insertion point is not visible, it probably means that the command window is not active; simply click anywhere in the command window to tell EVIEWS that you wish to enter commands.

You can move the insertion point to previously executed commands, edit the existing command, and then press ENTER to execute the edited version of the command.

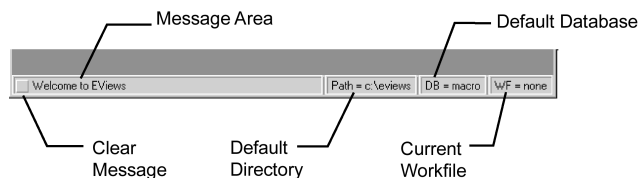
The command window supports Windows cut-and-paste so that you can easily move text between the command window, other EVIEWS text windows, and other Windows programs. The contents of the command area may also be saved directly into a text file for later use: make certain that the command window is active by clicking anywhere in the window, and then select **File/Save As...** from the main menu.

If you have entered more commands than will fit in your command window, EVIEWS turns the window into a standard scrollable window. Simply use the scroll bar or up and down arrows on the right-hand side of the window to see various parts of the list of previously executed commands.

You may find that the default size of the command window is too large or small for your needs. You can resize the command window by placing the cursor at the bottom of the command window, holding down the mouse button and dragging the window up or down. Release the mouse button when the command window is the desired size.

The Status Line

At the very bottom of the window is a *status line* which is divided into several sections.



The left section will sometimes contain status messages sent to you by EViews. These status messages can be cleared manually by clicking on the box at the far left of the status line. The next section shows the *default directory* that EViews will use to look for data and programs. The last two sections display the names of the default database and workfile. In later chapters, we will show you how to change both defaults.

The Work Area

The area in the middle of the window is the work area where EViews will display the various object windows that it creates. Think of these windows as similar to the sheets of paper you might place on your desk as you work. The windows will overlap each other with the foremost window being in focus or active. Only the active window has a darkened titlebar.

When a window is partly covered, you can bring it to the top by clicking on its titlebar or on a visible portion of the window. You can also cycle through the displayed windows by pressing the F6 or CTRL-TAB keys.

Alternatively, you may select a window by clicking on the **Window** menu item, and selecting the desired name.

You can move a window by clicking on its title bar and dragging the window to a new location. You can change the size of a window by clicking on any corner and dragging the corner to a new location.

Closing EViews

There are a number of ways to close EViews. You can always select **File/Exit** from the main menu, or you can press ALT-F4. Alternatively, you can click on the close box in the upper right-hand corner of the EViews window, or double click on the EViews icon in the upper left-hand corner of the window. If necessary, EViews will warn you and provide you with the opportunity to save any unsaved work.

Where To Go For Help

The EViews Manuals

This *User's Guide* describes how to use EViews to carry out your research. The earlier chapters deal with basic operations, the middle chapters cover basic econometric methods, and the later chapters describe more advanced methods.

Though we have tried to be complete, it is not possible to document every aspect of EViews. There are almost always several ways to do the same thing in EViews, and we cannot describe them all. In fact, one of the strengths of the program is that you will undoubtedly discover alternative, and perhaps more efficient, ways to get your work done.

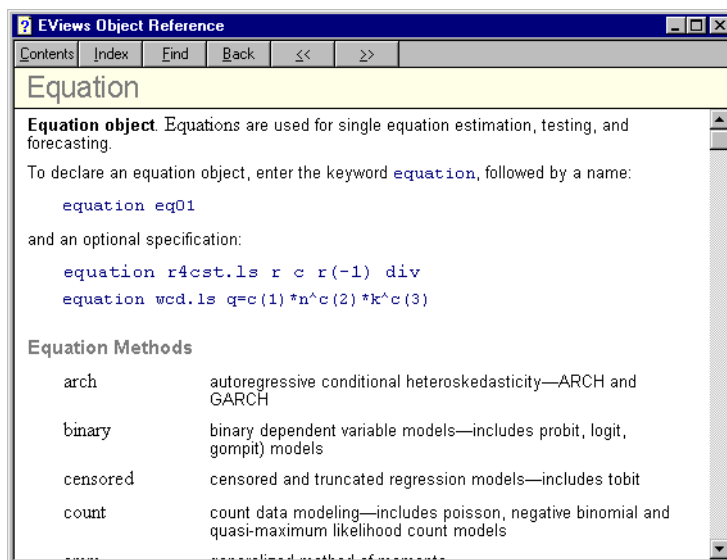
Most of the *User's Guide* explains the visual approach to using EViews. It describes how you can use your mouse to perform operations in EViews. To keep the explanations simple, we do not tell you about alternative ways to get your work done. For example, we will not remind you about the ALT- keyboard alternatives to using the mouse.

When we get to the discussion of the substantive statistical methods available in EViews, we will provide some technical information about the methods, and references to econometrics textbooks and other sources for additional information.

The Help System

Almost all of the EViews documentation may be viewed from within EViews by using the help system. To access the EViews help system, simply go to the main menu and select **Help**.

Since EViews uses standard Windows Help, the on-line manual is fully searchable and hypertext linked. You can set bookmarks to frequently accessed pages, and annotate the on-line documentation with your own notes.



In addition, the Help system will contain updates to the documentation that were made after the manuals went to press.

The World Wide Web

To supplement the information provided in the manuals and the help system, we have set up information areas on the Web that you may access using your favorite browser. You can

find answers to common questions about installing, using, and getting the most out of EViews.

Another popular area is our Download Section, which contains on-line updates to EViews Version 4, sample data and programs, and much more. Your purchase of EViews provides you with much more than the enclosed program and printed documentation. As we make minor changes and revisions to the current version of EViews, we will post them on our web site for you to download. As a valued QMS customer, you are free to download updates to the current version as often as you wish.

So set a bookmark to our site and visit often; the address is:

<http://www.eviews.com>.

Chapter 2. A Demonstration

In this chapter, we provide a demonstration of the basic features of EViews. The demonstration is not meant to be a comprehensive description of the program. A full description of the program begins in [Chapter 3](#).

This demo shows takes you through the following steps:

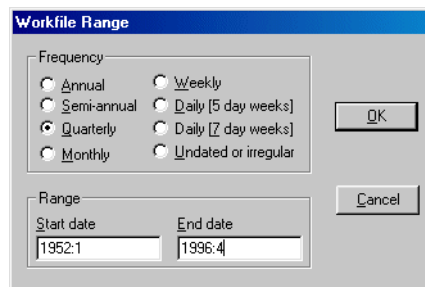
- importing data into EViews from an Excel spreadsheet
- examining the data and performing simple statistical analysis
- using regression analysis to model and forecast a statistical relationship
- performing specification and hypothesis testing
- plotting results

Creating a Workfile and Importing Data

The first step in the project is to read the data into an EViews workfile.

Before we describe the process of importing data, note that the demonstration data have been included in your EViews directory in both Excel spreadsheet and EViews workfile formats. If you wish to skip the discussion of importing data and go directly to the analysis part of the demonstration, you may load these data by selecting **File/Open/Workfile...** and opening DEMO.WF1.

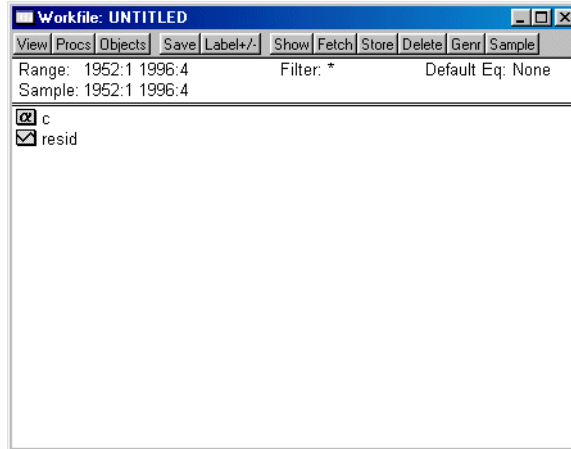
To create a workfile to hold your data, select **File/New/Workfile...**, which opens a dialog box where you will provide information about your data:



The screenshot shows the 'Workfile Range' dialog box. The 'Frequency' section has radio buttons for Annual, Semi-annual, Quarterly (selected), Monthly, Weekly, Daily [5 day weeks], Daily [7 day weeks], and Undated or irregular. The 'Range' section has text boxes for 'Start date' (1952:1) and 'End date' (1996:4). There are 'OK' and 'Cancel' buttons on the right side.

For our example, quarterly data are observed from the first quarter of 1952 to the end of 1996. You should set the workfile frequency to quarterly, and specify the start date 1952:1, and the end date 1996:4.

Once you have filled out the dialog, click on the **OK** button. EViews will create an untitled workfile, and will display a workfile window.

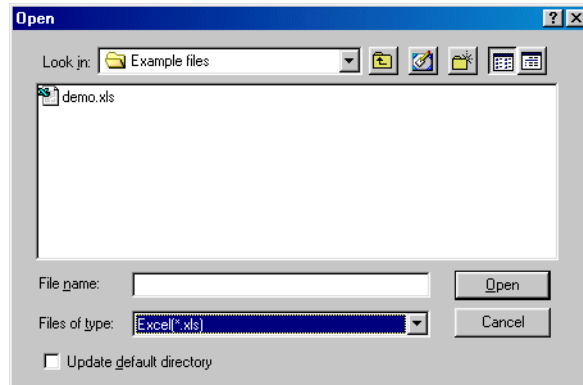


The workfile window is described in detail in “[The Workfile Window](#)” on page 35. For now, notice that the workfile window displays two pairs of dates: one for the range of dates contained in the workfile, and the second for the current workfile sample. Note also that the workfile contains the coefficient vector C and the series RESID. All EViews workfiles will contain these two objects.

The next step is to import data into the workfile. The data for the four variables used in the analysis have been provided in an Excel file named DEMO.XLS. The data in the DEMO.XLS file are arranged with each of the four series in columns, with names in the first row, and dates in the first column.

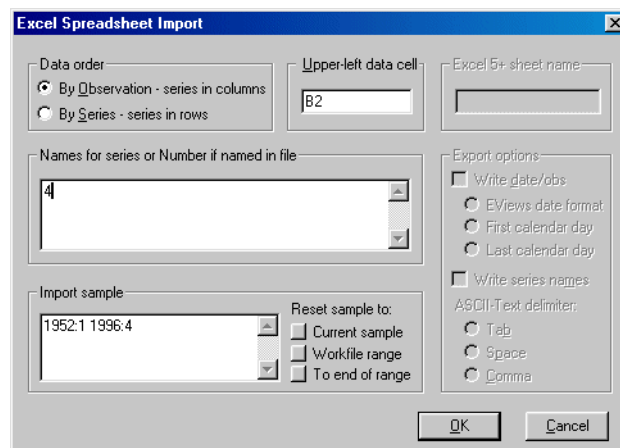
	A	B	C	D	E	F	G
1	OBS	GDP	PR	M1	RS		
2	1952:1	87.875	0.197561	126.537	1.64		
3	1952:2	88.125	0.198167	127.506	1.677667		
4	1952:3	89.625	0.200179	129.385	1.828667		
5	1952:4	92.875	0.201246	128.512	1.923667		
6	1953:1	94.625	0.201052	130.587	2.047333		
7	1953:2	95.55	0.201444	130.341	2.202667		
8	1953:3	95.425	0.202236	131.389	2.021667		
9	1953:4	94.175	0.202723	129.891	1.486333		
10	1954:1	94.075	0.203416	130.173	1.083667		
11	1954:2	94.2	0.203841	131.385	0.814333		
12	1954:3	95.45	0.204291	134.627	0.869667		
13	1954:4	97.36375	0.204374	134.252	1.036333		
14	1955:1	100.725	0.205603	136.413	1.256333		
15	1955:2	102.825	0.206227	136.471	1.614333		
16	1955:3	104.925	0.207762	138.377	1.861333		

To read these data, click on **Procs/Import/Read Text-Lotus-Excel...**

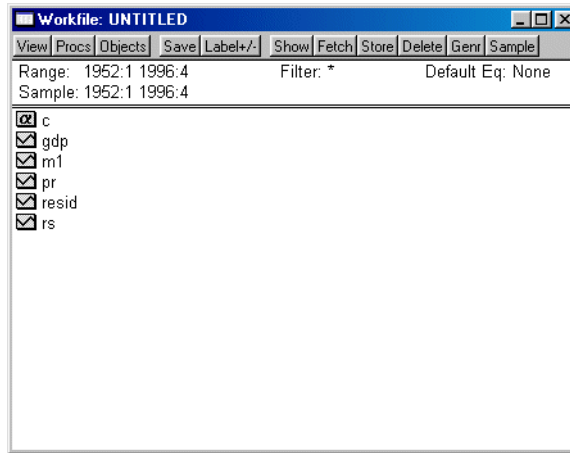


Locate the DEMO.XLS file (it should be in your EViews installation or “Example Files” directory) and double click on the file name. You can make finding the file a bit easier by choosing to display **Excel.xls** files from the **Files of type** combo box.

EViews will open the Excel spreadsheet import dialog:



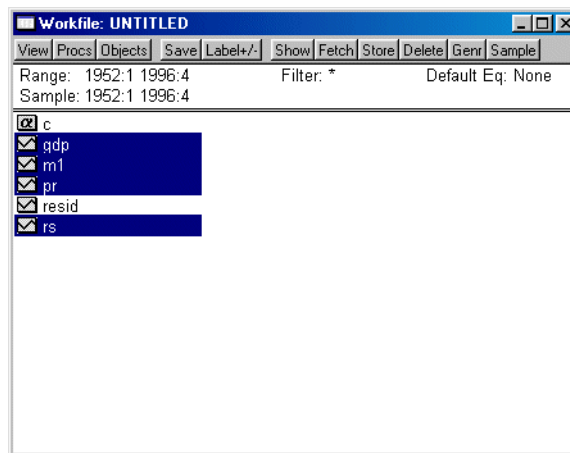
The default settings for order of data, upper-left data cell, and the sample to import should be appropriate for this Excel file. Since the names of the series are in the first row of the Excel file, you can simply enter the number of series (in this case you will want to enter “4”), in the **Names for series or Number of series if name in file** field of the dialog box. Click **OK**, and EViews will import the four series. These series will appear as icons in the workfile window:



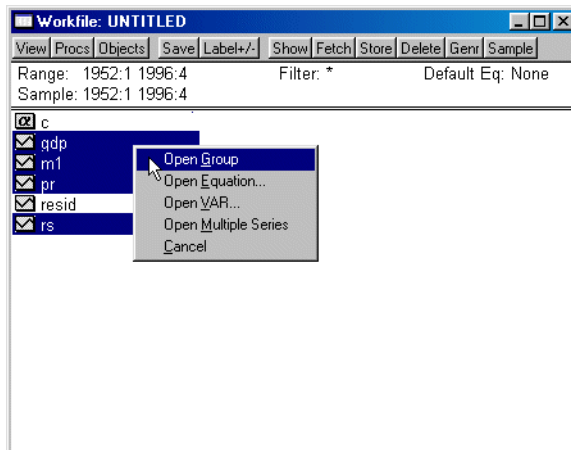
An alternative method of importing the data is to copy-and-paste the data from the Excel spreadsheet directly into EViews. This procedure is described in detail in [Chapter 4, “Basic Data Handling”](#), on page 55.

Verifying the Data

The first thing you should do after importing the data is to verify that the data have been read correctly. We will create a group object that allows us to examine all four series. Click on the name GDP in the workfile window, and then press CTRL and click on M1, PR, and RS. All four of the series should be highlighted:



Now place the cursor anywhere in the highlighted area and double click the left mouse button. EViews will open a popup menu providing you with several options:



Choose **Open Group**. EViews will create an untitled group object containing all four of the series. The default window for the group shows a spreadsheet view of the series:

The screenshot shows the EViews Group window titled "Group: UNTITLED Workfile: UNTITLED". The menu bar includes View, Procs, Objects, Print, Name, Freeze, Edit+/-, Smpl+/-, InsDel, and Tr. The spreadsheet view displays the following data:

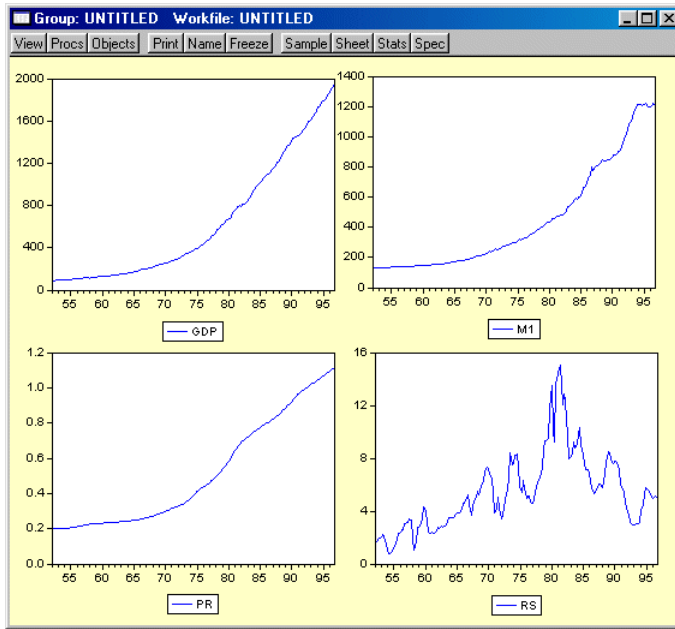
ob#	GDP	PR	M1	RS
ob#	GDP	PR	M1	RS
1952:1	87.87500	0.197561	126.5370	1.640000
1952:2	88.12500	0.198167	127.5060	1.677667
1952:3	89.62500	0.200179	129.3850	1.828667
1952:4	92.87500	0.201246	128.5120	1.923667
1953:1	94.62500	0.201052	130.5870	2.047333
1953:2	95.55000	0.201444	130.3410	2.202667
1953:3	95.42500	0.202236	131.3890	2.021667
1953:4	94.17500	0.202723	129.8910	1.486333
1954:1	94.07500	0.203416	130.1730	1.083667
1954:2	94.20000	0.203841	131.3850	0.814333
1954:3	95.45000	0.204291	134.6270	0.869667
1954:4	97.36375	0.204374	134.2520	1.036333
1955:1	100.7250	0.206603	136.4130	1.266333
1955:2	100.0000	0.206603	136.4130	1.266333
1955:3	100.0000	0.206603	136.4130	1.266333

You should compare the spreadsheet view with the top of the Excel worksheet to ensure that the first part of the data has been read correctly. You can use the scroll bars and scroll arrows on the right side of the window to verify the remainder of the data.

Once you are satisfied that the data are correct, you should save the workfile by clicking the **Save** button in the workfile window. A save dialog will open, prompting you for a workfile name and location. You should enter DEMO2, then click **OK**. EViews will save the workfile in the specified directory with the name DEMO2.WF1. A saved workfile can be opened later by selecting **File/Open/Workfile...** from the main menu.

Examining the Data

We can use basic EViews tools to examine the data in your group object in a variety of ways. For example, if you select **View/Multiple Graphs/Line** from the group object toolbar, EViews displays line graphs of each of the series in the group:



You can select **View/Descriptive Stats/Individual Samples** to compute descriptive statistics for each of the series in the group:

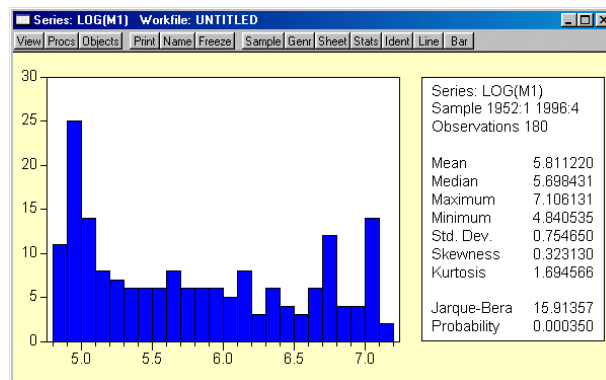
Group: UNTITLED		Workfile: DEMO			
	GDP	M1	PR	RS	
Mean	832.4190	445.0064	0.514106	5.412928	
Median	374.3000	298.3990	0.383802	5.057500	
Maximum	1948.225	1219.420	1.110511	15.08733	
Minimum	87.87500	126.5370	0.197561	0.814333	
Std. Dev.	564.2441	344.8315	0.303483	2.908939	
Skewness	0.846880	0.997778	0.592712	0.986782	
Kurtosis	2.346008	2.687096	1.828239	4.049883	
Jarque-Bera	24.68300	30.60101	20.81933	37.47907	
Probability	0.000004	0.000000	0.000030	0.000000	
Sum	113835.4	80101.16	92.53909	974.3270	
Sum Sq. Dev.	66988478	21284672	16.48625	1514.685	
Observations	180	180	180	180	

or click on **View/Correlations/Common Samples** to display the correlation matrix of the four series:

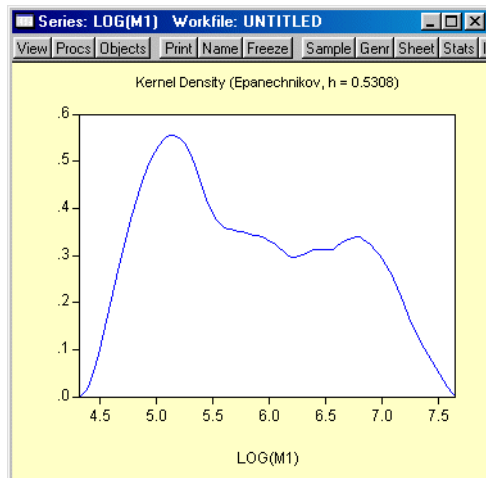
	GDP	M1	PR	RS
GDP	1.00000	0.995197	0.992475	0.333494
M1	0.995197	1.00000	0.990402	0.270059
PR	0.992475	0.990402	1.00000	0.412471
RS	0.333494	0.270059	0.412471	1.00000

We can also examine characteristics of the individual series. Since our regression analysis will be expressed in logarithms, we will work the log of M1. Select **Quick/Show...** then enter $\log(m1)$, and click OK. EViews will open a series window for LOG(M1).

Now select **View/Descriptive Statistics/Histogram and Stats** from the series toolbar to display the descriptive statistics for LOG(M1):



We can construct a smoothed version of the histogram by selecting **View/Distribution Graphs/Kernel Density...** and clicking on OK to accept the default options:



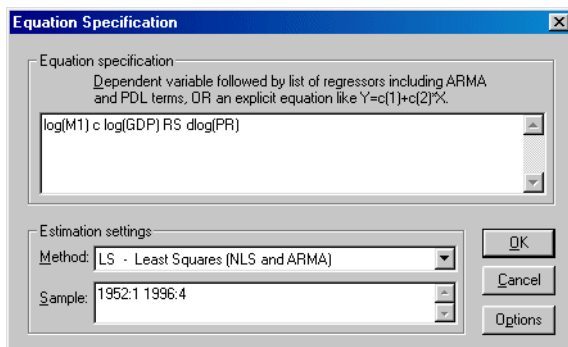
Estimating a Regression Model

We now estimate a regression model for M1 using data over the period from 1952:1–1992:4 and use this estimated regression to construct forecasts over the period 1993:1–2003:4. The model specification is

$$\log(M1_t) = \beta_1 + \beta_2 \log GDP_t + \beta_3 RS_t + \beta_4 \Delta \log PR_t + \epsilon_t \quad (2.1)$$

where $\log(M1)$ is the logarithm of the money supply, $\log(GDP)$ is the log of income, RS is the short term interest rate, and $\Delta \log(PR)$ is the log first difference of the price level (the approximate rate of inflation).

To estimate the model, we will create an equation object. Select **Quick** from the main menu and choose **Estimate Equation...** to open the estimation dialog. Enter the following equation specification:



Here we list the name of the dependent variable, followed by the names of each of the regressors, separated by spaces. We use expressions involving the functions `log` and `dlog` to represent the log transformations of `M1` and `GDP`, and the difference of the log transformation for `PR`. The built-in series name `C` stands for the constant in the regression.

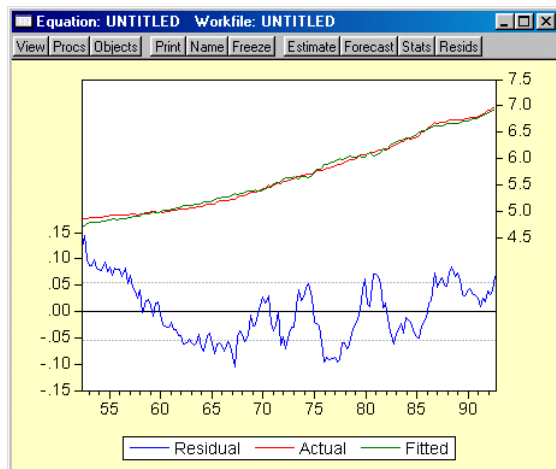
The dialog is initially set to estimate the equation using the **LS - Least Squares** method for the **Sample** 1952:1 1996:4. You should change the **Sample** to 1952:1 1992:4 to estimate the equation for the subsample of observations.

Click **OK** to estimate the equation using least squares and to display the regression results:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 10/19/97 Time: 22:43
Sample(adjusted): 1952:2 1992:4
Included observations: 163 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1.312383	0.032199	40.75850	0.0000
LOG(GDP)	0.772035	0.006537	118.1092	0.0000
RS	-0.020686	0.002516	-8.221196	0.0000
DLOG(PR)	-2.572204	0.942556	-2.728967	0.0071
R-squared	0.993274	Mean dependent var		5.692279
Adjusted R-squared	0.993147	S.D. dependent var		0.670253
S.E. of regression	0.055485	Akaike info criterion		-2.921176
Sum squared resid	0.489494	Schwarz criterion		-2.845256
Log likelihood	242.0759	F-statistic		7826.904
Durbin-Watson stat	0.140967	Prob(F-statistic)		0.000000

Note that the equation is estimated from 1952:2 to 1992:4 since one observation is dropped from the beginning of the estimation sample to account for the `dlog` difference term. The estimated coefficients are statistically significant, with t -statistic values well in excess of 2. The overall regression fit, as measured by the R^2 value, indicates a very tight fit. You can select **View/Actual, Fitted, Residual/Graph** in the equation toolbar to display a graph of the actual and fitted values for the dependent variable, along with the residuals:



Specification and Hypothesis Tests

We can use the estimated equation to perform hypothesis tests on the coefficients of the model. For example, to test the hypothesis that the coefficient on the price term is equal to 2, we will perform a Wald test. First, determine the coefficient of interest by selecting **View/Representations** from the equation toolbar:

```

Equation: UNTITLED  Workfile: UNTITLED
View Procs Objects Print Name Freeze Estimate Forecast Stats Resids
=====
Estimation Command:
=====
LS(M=500,C=0.0001,DERIV=AA,-SHOWOPTS) LOG(M1) C LOG(GDP) RS
DLOG(PR)

Estimation Equation:
=====
LOG(M1) = C(1) + C(2)*LOG(GDP) + C(3)*RS + C(4)*DLOG(PR)

Substituted Coefficients:
=====
LOG(M1) = 1.312383474 + 0.7720348992*LOG(GDP) - 0.02068603432*RS -
2.572203714*DLOG(PR)

```

Note that the coefficients are assigned in the order that the variables appear in the specification so that the coefficient for the PR term is labeled C(4). To test the restriction on C(4)

you should select **View/Coefficient Tests/Wald–Coefficient Restrictions...**, and enter the restriction $c(4) = 2$. EViews will report the results of the Wald test:

Wald Test:
Equation: EQ01

Test Statistic	Value	df	Probability
F-statistic	23.53081	(1, 159)	0.0000
Chi-square	23.53081	1	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-2 + C(4)	-4.572204	1.060945

Restrictions are linear in coefficients.

The low probability values indicate that the null hypothesis that $C(4) = 2$ is strongly rejected.

We should, however, be somewhat cautious of accepting this result without additional analysis. The low value of the Durbin-Watson statistic reported above is indicative of the presence of serial correlation in the residuals of the estimated equation. If uncorrected, serial correlation in the residuals will lead to incorrect estimates of the standard errors, and invalid statistical inference for the coefficients of the equation.

The Durbin-Watson statistic can be difficult to interpret. To perform a more general Breusch-Godfrey test for serial correlation in the residuals, select **View/Residual Tests/Serial Correlation LM Test...** from the equation toolbar, and specify an order of serial correlation to test against. Entering 1 yields a test against first-order serial correlation:

Breusch-Godfrey Serial Correlation LM Test:

F-statistic	813.0060	Probability	0.000000
Obs*R-squared	136.4770	Probability	0.000000

Test Equation:
 Dependent Variable: RESID
 Method: Least Squares
 Date: 10/19/97 Time: 22:45

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.006355	0.013031	-0.487683	0.6265
LOG(GDP)	0.000997	0.002645	0.376929	0.7067
RS	-0.000567	0.001018	-0.556748	0.5785
DLOG(PR)	0.404143	0.381676	1.058864	0.2913
RESID(-1)	0.920306	0.032276	28.51326	0.0000
R-squared	0.837282	Mean dependent var		1.21E-15
Adjusted R-squared	0.833163	S.D. dependent var		0.054969
S.E. of regression	0.022452	Akaike info criterion		-4.724644
Sum squared resid	0.079649	Schwarz criterion		-4.629744
Log likelihood	390.0585	F-statistic		203.2515
Durbin-Watson stat	1.770965	Prob(F-statistic)		0.000000

The top part of the output presents the test statistics and associated probability values. The test regression used to carry out the test is reported below the statistics.

The statistic labeled “Obs*R-squared” is the LM test statistic for the null hypothesis of no serial correlation. The (effectively) zero probability value strongly indicates the presence of serial correlation in the residuals.

Modifying the Equation

The test results suggest that we need to modify our original specification to take account of the serial correlation.

One approach is to include lags of the independent variables. To add variables to the existing equation, click on the **Estimate** button in the equation toolbar and edit the specification to include lags for each of the original explanatory variables:

```
log(m1) c log(gdp) rs dlog(pr) log(m1(-1)) log(gdp(-1)) rs(-1)
      dlog(pr(-1))
```

Note that lags are specified by including a negative number, enclosed in parentheses, following the series name. Click on **OK** to estimate the new specification and to display the results:

Dependent Variable: LOG(M1)
 Method: Least Squares
 Date: 10/19/97 Time: 22:48
 Sample(adjusted): 1952:3 1992:4
 Included observations: 162 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.071297	0.028248	2.523949	0.0126
LOG(GDP)	0.320338	0.118186	2.710453	0.0075
RS	-0.005222	0.001469	-3.554801	0.0005
DLOG(PR)	0.038615	0.341619	0.113036	0.9101
LOG(M1(-1))	0.926640	0.020319	45.60375	0.0000
LOG(GDP(-1))	-0.257364	0.123264	-2.087910	0.0385
RS(-1)	0.002604	0.001574	1.654429	0.1001
DLOG(PR(-1))	-0.071650	0.347403	-0.206246	0.8369
R-squared	0.999604	Mean dependent var		5.697490
Adjusted R-squared	0.999586	S.D. dependent var		0.669011
S.E. of regression	0.013611	Akaike info criterion		-5.707729
Sum squared resid	0.028531	Schwarz criterion		-5.555255
Log likelihood	470.3261	F-statistic		55543.30
Durbin-Watson stat	2.393764	Prob(F-statistic)		0.000000

Note that EViews has automatically adjusted the estimation sample to accommodate the additional lagged variables. We will save this equation in the workfile for later use. Press the **Name** button in the toolbar and name the equation EQLAGS.

Another common method of accounting for serial correlation is to include autoregressive (AR) and/or moving average (MA) terms in the equation. To estimate the model with an AR(1) error specification, you should make a copy of the previous equation by clicking **Objects/Copy Object...** EViews will create a new untitled equation containing all of the information from the previous equation. Press **Estimate** on the toolbar of the copy and modify the specification to read

```
log(m1) c log(gdp) rs dlog(pr) ar(1)
```

This specification removes the lagged terms, replacing them with an AR(1) specification. Click **OK**. EViews will report the estimation results, including the estimated first-order autoregressive coefficient of the error term:

Dependent Variable: LOG(M1)
 Method: Least Squares
 Date: 10/19/97 Time: 22:52
 Sample(adjusted): 1952:3 1992:4
 Included observations: 162 after adjusting endpoints
 Convergence achieved after 14 iterations

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1.050340	0.328390	3.198453	0.0017
LOG(GDP)	0.794929	0.049342	16.11057	0.0000
RS	-0.007395	0.001457	-5.075131	0.0000
DLOG(PR)	-0.008019	0.348689	-0.022998	0.9817
AR(1)	0.968100	0.018190	53.22283	0.0000
R-squared	0.999526	Mean dependent var		5.697490
Adjusted R-squared	0.999514	S.D. dependent var		0.669011
S.E. of regression	0.014751	Akaike info criterion		-5.564584
Sum squared resid	0.034164	Schwarz criterion		-5.469288
Log likelihood	455.7313	F-statistic		82748.93
Durbin-Watson stat	2.164265	Prob(F-statistic)		0.000000
Inverted AR Roots	.97			

The fit of the AR(1) model is roughly comparable to the lag model, but the somewhat higher values for both the Akaike and the Schwarz information criteria indicate that the previous lag model should be preferred. We will work with the lag model for the remainder of the demonstration.

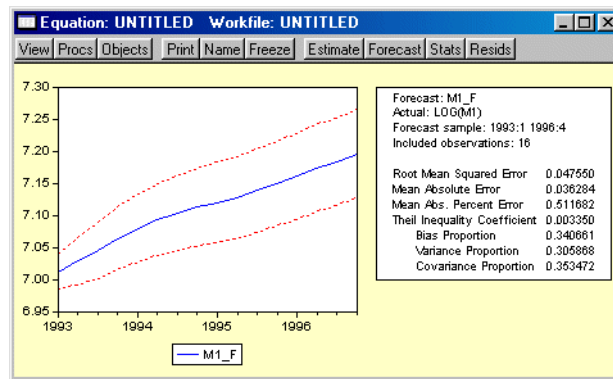
Forecasting from an Estimated Equation

We have been working with a subset of our data, so that we may compare forecasts based upon this model with the actual data for the post-estimation sample 1993:1–1996:4.

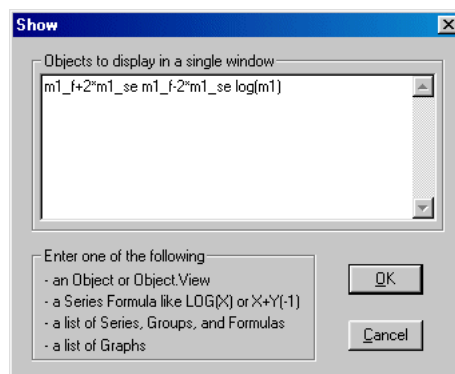
Click on the **Forecast** button in the EQLAGS equation toolbar to open the forecast dialog:

We set the forecast sample to 1993:1–1996:4 and provide names for both the forecasts and forecast standard errors so both will be saved as series in the workfile. The forecasted values will be saved in M1_F and the forecast standard errors will be saved in M1_SE.

Note also that we have elected to forecast the log of M1, not the level, and that we request both graphical and forecast evaluation output. The **Dynamic** option constructs the forecast for the sample period using only information available at the beginning of 1993:1. When you click **OK**, EViews displays both a graph of the forecasts, and statistics evaluating the quality of the fit to the actual data:

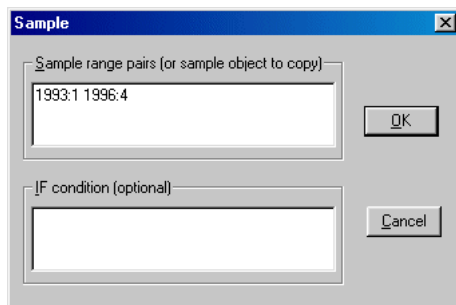


We can also plot the actual values of $\log(M1)$ against the forecasted values and the (approximate) 95% confidence intervals for the forecasts. First, we will create a new group containing these values by **Quick/Show...** and filling out the dialog as follows:

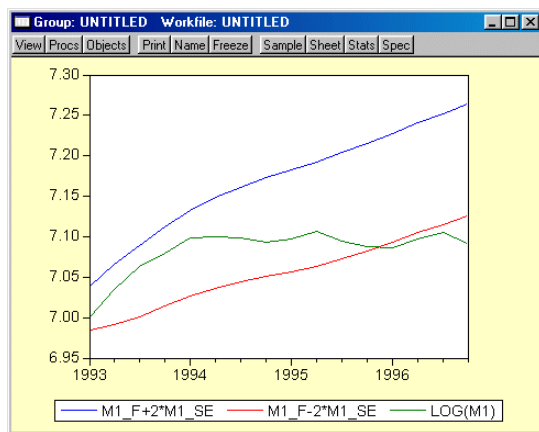


There are three expressions in the dialog. The first two represent the upper and lower bounds of the (approximate) 95% forecast interval as computed by evaluating the values of the point forecasts plus and minus two times the standard errors. The last expression represents the actual values of the dependent variable.

When you click **OK**, EViews opens an untitled group window containing a spreadsheet view of the data. Before plotting the data, we will change the sample of observations so that we only plot data for the forecast sample. Select **Quick/Sample...** or click on the **Sample** button in the group toolbar, and change the sample to include only the forecast period:

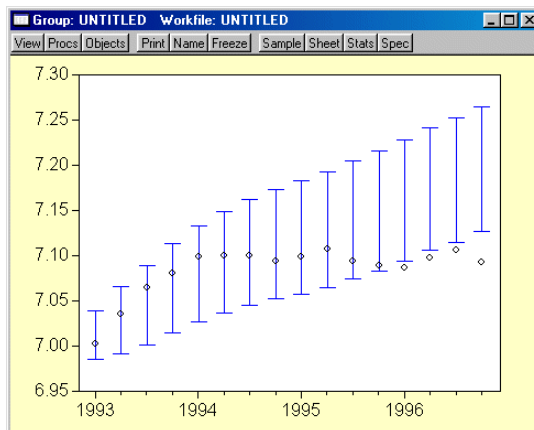


To plot the data for the forecast period, select **View/Graph/Line** from the group window:



The actual values of $\log(M1)$ are within the forecast interval for most of the forecast period, but fall below the lower bound of the 95% confidence interval beginning in 1996:1.

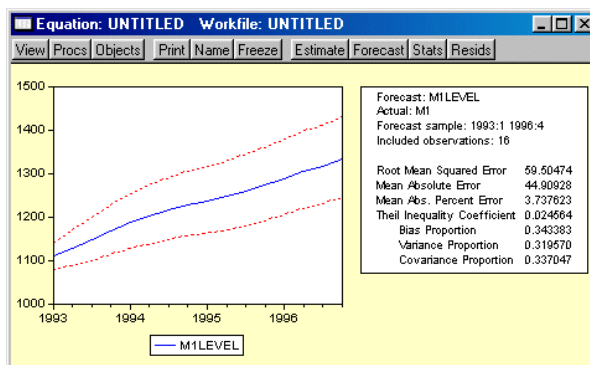
For an alternate view of these data, you can select **View/Graph/Error Bar**, which displays the graph as follows:



This graph clearly shows that the forecasts of $\log(M1)$ over-predict the actual values in the last four quarters of the forecast period.

We may also choose to examine forecasts of the level of M1. Click on the **Forecast** button in the EQLAGS equation toolbar to open the forecast dialog, and select M1 under the **Forecast of** option. Enter a new name to hold the forecasts, say M1LEVEL, and click **OK**.

EViews will present a graph of the forecast of the level of M1, along with the asymmetric confidence intervals for this forecast:



Additional Issues

It is worth noting that the example analysis above should be used for illustrative purposes only since there are a number of problems with the specification.

For one, there is quite a bit of serial correlation remaining in the EQLAGS specification. A test of serial correlation in the new equation (by selecting **View/Residual Tests/Serial**

Correlation LM Test..., and entering 1 for the number of lags) rejects the null hypothesis of no serial correlation in the reformulated equation:

Breusch-Godfrey Serial Correlation LM Test:

F-statistic	7.880369	Probability	0.005648
Obs*R-squared	7.935212	Probability	0.004848

Furthermore, there is evidence of autoregressive conditional heteroskedasticity (ARCH) in the residuals. Select **View/Residual Tests/ARCH LM Test...** and accept the default of 1. The ARCH test results strongly suggest the presence of ARCH in the residuals:

ARCH Test:

F-statistic	11.21965	Probability	0.001011
Obs*R-squared	10.61196	Probability	0.001124

In addition to serial correlation and ARCH, there is an even more fundamental problem with the above specification since, as the graphs attest, $\log(M1)$ exhibits a pronounced upward trend. We can, and should, perform tests for a unit root in this series. The presence of a unit root will indicate the need for further analysis.

Display the series window by clicking on **Window** and selecting the LOG(M1) series window from the menu. If the series window is closed, you may open a new window by selecting **Quick/Show...**, entering $\log(m1)$, and clicking OK.

To perform an Augmented Dickey-Fuller (ADF) test for nonstationarity of this series, select **View/Unit Root Test...** and click on OK to accept the default options. EViews will perform an ADF test and display the test results:

ADF Test Statistic	0.665471	1% Critical Value*	-3.4688
		5% Critical Value	-2.8780
		10% Critical Value	-2.5755

*MacKinnon critical values for rejection of hypothesis of a unit root.

The ADF test statistic value is greater than the critical values so that we cannot reject the null hypothesis of a unit root. The presence of a unit root suggests that we need to adopt more sophisticated statistical models. These techniques are discussed in [Chapter 13, “Time Series Regression”](#) and [Chapter 20, “Vector Autoregression and Error Correction Models”](#) of the *User’s Guide* which deal with time series and vector autoregression and vector error correction specifications, respectively.

Chapter 3. EViews Basics

Managing the variety of tasks associated with your work can be a complex and time-consuming process. Fortunately, EViews' innovative design takes much of the effort out of organizing your work, allowing you to concentrate on the substance of your project.

At the heart of the EViews design is the concept of an object. In brief, objects are collections of related information and operations that are bundled together into an easy-to-use unit. Virtually all of your work in EViews will involve using and manipulating various objects.

EViews holds all of its objects in object containers. You can think of object containers as filing cabinets or organizers for the various objects with which you are working. The most important object container in EViews is the workfile.

The remainder of this chapter describes basic techniques for working with objects and workfiles. While you may at first find the idea of objects to be a bit foreign, the basic concepts are easy to master and will form the foundation for your work in EViews. But don't feel that you have to understand all of the concepts the first time through. If you wish, you can begin working with EViews immediately, developing an intuitive understanding of objects and workfiles as you go.

Subsequent chapters will provide a more detailed description of working with the various types of objects and other types of object containers.

Workfile Basics

All EViews objects must be held in an object container. Most of your work in EViews will involve objects that are contained in a workfile, so your first step in any project will be to create a new workfile or to load an existing workfile into memory.

Workfiles have two primary characteristics. First, they are held in RAM for quick access to the objects in the workfile. Second, workfiles are characterized by a frequency and a range.

Data are often sampled at equally spaced intervals, or frequencies, over calendar time. When you set a workfile frequency, you tell EViews about the intervals between observations in your data. EViews has dated workfile types which handle annual, semi-annual, quarterly, monthly, weekly, and daily (5- or 7-day) data. For these workfiles, EViews will use all available calendar information in organizing and managing your data. For example, for weekly and daily data, EViews knows that some years contain days in each of 53 weeks, and that some years have 366 days, and will adjust the number of observations in a year accordingly.

Undated or irregular workfiles are those in which no dates are associated with the data—observations are simply numbered consecutively. Undated data are typically used for cross-section data, but may also be used in any situation where data are sampled irregularly; for example, financial data with frequent and irregular breaks for non-trading days.

The workfile range is a pair of dates or observation numbers describing the first and last observation to be held in the workfile.

Creating a Workfile

Your first step in EViews will usually be to create a workfile. One way to create a workfile is to click **File/New/Workfile...** and then to provide the necessary dialog information.

Select the appropriate frequency and enter the information for the workfile range. The **Start date** is the earliest date or observation you plan to use in the project and the **End date** is the latest date or observation. Don't worry if you don't know the exact start and end date; if you later find that your workfile isn't the right size, you can expand or contract the workfile range.

The rules for describing dates are quite simple:

- **Annual:** specify the year. Years from 1930–2029 may be identified using either 2 or 4-digit identifiers (e.g. “97” or “1997”). All other years must be identified with full year identifiers (e.g. “1776”, “2040”, “9789” or “50234”). Note that since 2-digit identifiers are assumed to be in either the 20th or 21st century, EViews cannot handle dates prior to A.D. 100.
- **Quarterly:** the year, followed by a colon or the letter “Q”, and then the quarter number. Examples: “1992:1”, “65:4”, “2002Q3”.
- **Monthly:** the year, followed by a colon or the letter “M”, and then the month number. Examples: “1956:1”, “1990M1”.
- **Semi-Annual:** the year, followed by a colon or the letter “S”, and then either “1” or “2” to denote the period. Examples: “1992:1”, “2024S2”.
- **Weekly and daily:** by default, you should specify these dates as month number, followed by a colon, followed by the day number, followed by a colon, followed by the year. Using the **Options/Dates-Frequency...** menu item, you can reverse the order of the day and month by switching to European notation.

For example, entering “8:10:97” indicates that you want your workfile to begin with August 10, 1997. If you have previously set your default date-frequency option to European notation, this date represents October 8, 1997.

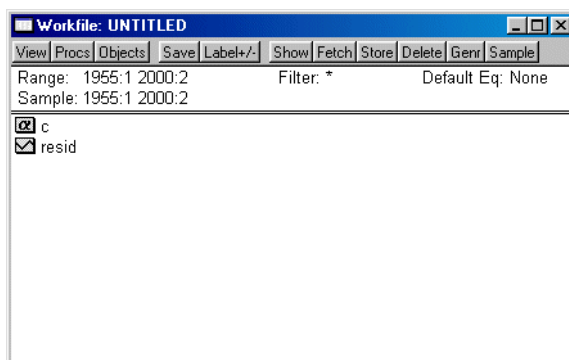
With weekly data, the day of the week associated with the starting date determines the beginning of the week. In the examples above, the first observations would be the week running from Sunday, August 10 through Saturday, August 16, 1997, or the week running from Wednesday, October 8, through Tuesday, October 14, 1997.

Alternatively, for quarterly, monthly, weekly, and daily data, you can enter just the year, and EViews will automatically specify the first and last observations for you.

In [Appendix B, “Date Formats”, beginning on page 653](#) we discuss the specification of dates in EViews in greater detail.

After you have finished supplying the information about the type of workfile and clicked **OK**, you will see the workfile window:

Here we have specified a workfile which will contain quarterly data from the first quarter of 1955 through the end of 1996. Since we have not yet saved the workfile, it is UNTITLED.

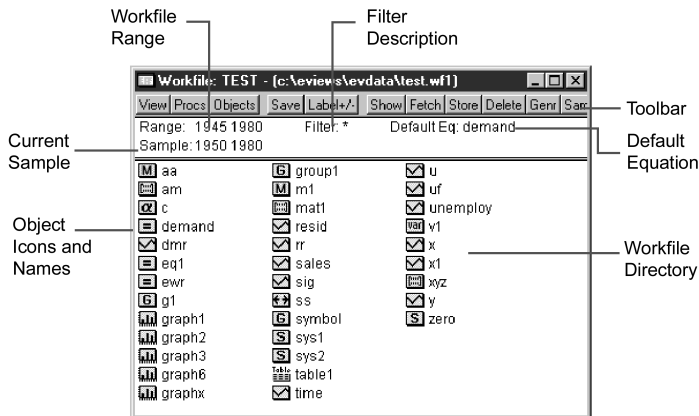


Note that there are two icons in this newly created workfile. These icons represent the objects that are contained in *every* workfile: a vector of coefficients, C , and a series of residuals, RESID. The little icon to the left identifies the type of object, an α for a coefficient vector and a tiny time series plot for a series.

Workfiles may also be created directly from EViews databases. See [Chapter 6](#) for further details.

The Workfile Window

After you have created a workfile and a number of objects, the workfile window will look something like this:



In the titlebar of the workfile window you will see the “**Workfile**” designation followed by the workfile name. If the workfile has not been saved, it will be designated “UNTITLED”. If the workfile has been saved to disk, you will see the name and the full disk path.

Just below the titlebar is a toolbar made up of a number of buttons. These buttons provide you with easy access to a number of useful workfile operations.

Below the toolbar are two lines of status information where EViews displays the range of the workfile, the current *sample* of the workfile (the range of observations that are to be used in calculations and statistical operations), the display filter (rule used in choosing a subset of objects to display in the workfile window), and the *default equation* (the last equation estimated or operated on). You may change the range, sample, and filter by double clicking on these labels and entering the relevant information in the dialog boxes. Double clicking on the equation label opens the equation.

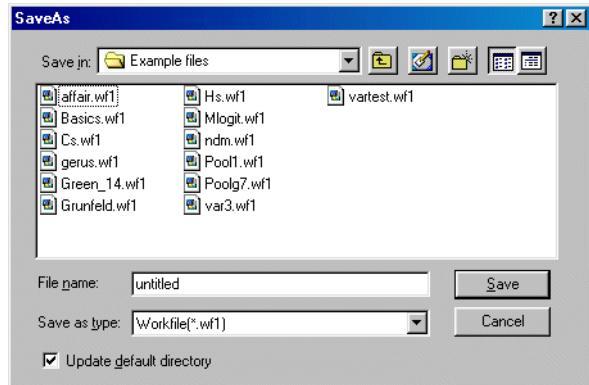
Lastly, you will see the workfile directory. In normal display mode, all named objects are listed in the directory by name and icon. The different types of objects and their icons are described in detail in “[Object Types](#)” on page 43.

It is worth remembering that the workfile window is a specific example of an object window. Object windows are discussed in “[The Object Window](#)” on page 46.

Saving Workfiles

You will want to name and save your workfile for future use. Push the **Save** button on the workfile toolbar to save a copy of the workfile on disk. You can also save the file using the **File/Save As...** or **File/Save...** choices from the main menu. A standard Windows file dialog box will open:

You can specify the target directory in the upper file menu labeled **Save in:**. You can navigate between directories in the standard Windows fashion—click once on the down arrow to access a directory tree; double clicking on a directory name in the display area gives you a list of all the files and subdirectories in that directory. Once you have worked your way to the right directory, type the name you want to give the workfile in the **File name:** box and push the **Save** button. Your workfile will be saved with the name you choose and the extension .WF1.



Alternatively, you could just type the full Windows path information and name, in the **File name:** box.

Once the workfile is named and saved, you can save subsequent updates or changes using the **Save** button on the toolbar, or **File/Save...** from the main menu. Selecting **Save** will update the existing workfile stored on disk. As with other Windows software, **File/Save As...** can be used to save the file with a new name. If the file you save to already exists, EViews will ask you whether you want to update the version on disk.

Note that workfiles saved in EViews Version 4 can, in general, be read by previous versions of EViews. Objects that are new to EViews Version 4 will, however, be removed from the workfile. We recommend that you take great caution when saving over your workfile using older versions of EViews.

Loading Workfiles

You can use **File/Open/Workfile...** to bring back a previously saved workfile. You will typically save your workfile containing all of your data and results at the end of the day, and later use **File/Open/Workfile...** to pick up where you left off.

When you select **File/Open/Workfile...** you will see a standard Windows file dialog. Simply navigate to the appropriate directory and double click on the name of the workfile to load it into RAM. The workfile window will open and all of the objects in the workfile will immediately be available.

For convenience, EViews keeps a record of the ten most recently used workfiles and programs at the bottom of the **File** menu. Select an entry and it will be opened in EViews.

Version 4 of EViews can read workfiles from all previous versions of EViews.

Save and Load Options

There are optional settings in the **File/Open...** and **File/Save As...** dialogs which provide you with additional control over the procedures which use files saved on disk.

Set Default Directory

All EViews file dialogs begin with a display of the contents of the *default directory*. You can always identify the default directory from the listing on the EViews status line. The default directory is set initially to be the directory containing the EViews program, but it can be changed at any time.

You can change the default directory by using the **File/Open...** or the **File/Save As...** menu items, navigating to the new directory, and checking the **Update Default Directory** box in the dialog. If you then open or save a workfile, the default directory will change to the one you have selected. The default directory may also be set from the **Options/File locations...** dialog. See [“File Locations” on page 648](#).

An alternative method for changing the default EViews directory is to use the `cd` command. Simply enter “`cd`” followed by the directory name in the command window (see [`cd`, `chdir` \(p. 156\)](#) of the *Command and Programming Reference* for details).

Using MicroTSP Files

You can read or write your workfile in a format that is compatible with MicroTSP. The **Files of Type:** and **Save as Type:** drop boxes allow you to handle DOS and Macintosh MicroTSP files. Simply click on the drop down box and select either **Old Dos Workfile** or **Old Mac Workfile**, as appropriate. You should be aware, however, that if you choose to save a workfile in MicroTSP format, only basic series data will be saved—the remainder of the workfile contents will be discarded.

Resizing Workfiles

You may decide to add data or make forecasts for observations beyond the ending date or before the starting date of your workfile. Alternatively, you may wish to remove extra observations from the start or end of the workfile.

To change the size of your workfile, select **Procs/Change Workfile Range...** and enter the beginning and ending observation of the workfile in the dialog. If you enter dates that encompass the original workfile range, EViews will expand the workfile without additional comment. If you enter a workfile range that does not encompass the original workfile range, EViews will warn you that data will be lost, and ask you to confirm the operation.

Sorting Workfiles

Basic data in workfiles are held in objects called series. If you click on **Procs/Sort Series...** in the workfile toolbar, you can sort all of the series in the workfile on the basis of the values of one or more of the series. A dialog box will open where you can provide the details about the sort.

If you list two or more series, EViews uses the values of the second series to resolve ties from the first series, and values of the third series to resolve ties from the second, and so forth. If you wish to sort in descending order, select the appropriate option in the dialog.

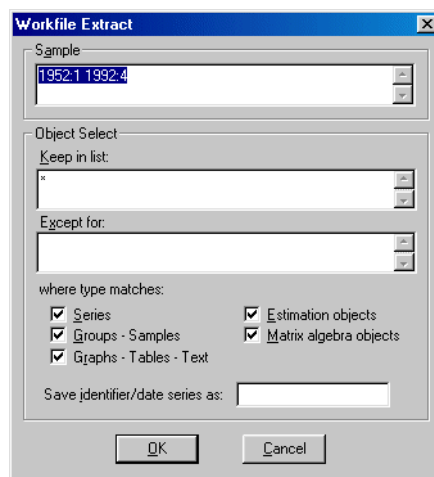
Note that if you are using a dated workfile, sorting the workfile will generally break the link between an observation and the corresponding date.

Extracting from a Workfile

You can use a workfile proc to create a subset of an existing workfile.

Simply select **Procs/Extract to new workfile...** and fill out the dialog. You will need to specify a sample, the types of objects (see “[Object Basics](#)” on page 41) you wish to keep, and lists of wildcard expressions describing the names of the objects to be kept and/or the objects to be dropped. You can also instruct EViews to create a series containing an observation indicator by entering a valid EViews name in the **Save identifier/date series** box.

When you click on **OK**, EViews will create a new workfile of an appropriate type containing your selections. If present, the new indicator series will contain the position of the observation in the original workfile.



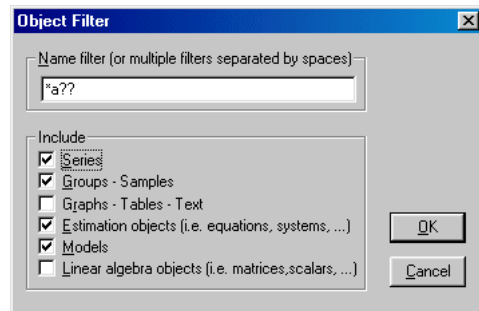
In most circumstances, the newly created workfile will have the same date format of the original workfile. Thus, if possible, EViews will extract a quarterly workfile to a quarterly workfile, and an annual workfile to an annual workfile. However, if the original workfile is dated and the extracting sample is non-contiguous, EViews will create the new workfile in an undated format. For example, if you extract the observations in the non-contiguous (monthly workfile) sample from Jan 1994 through Dec 1996 and Jan 1999 to December 2000, EViews will create an undated workfile containing data from designated subsample.

Changing the Workfile Display

Display Filter

When working with workfiles containing a large number of objects, it can become difficult to locate specific objects in the workfile window. You can solve this problem by using the workfile display filter to instruct EViews to display only a subset of objects in the workfile window. This subset can be defined on the basis of object name as well as object type.

Select **View/Display Filter...** or double click on the Filter description in the workfile window. The following dialog box will appear:



There are two parts to this dialog. In the edit field (blank space) of this dialog, you may place one or several name descriptions that include the standard wildcard characters: “*” (match any number of characters) and “?” (match any single character). Below the edit field are a series of check boxes corresponding to various types of EViews objects. EViews will display only objects of the specified types whose names match those in the edit field list.

The default string is “*”, which will display all objects of the specified types. However, if you enter the string

x*

only objects with names beginning with X will be displayed in the workfile window. Entering

x?y

displays all objects that begin with the letter X, followed by any single character and then ending with the letter Y. If you enter:

x* y* *z

all objects with names beginning with X or Y and all objects with names ending in Z will be displayed. Similarly, the more complicated expression:

??y* *z*

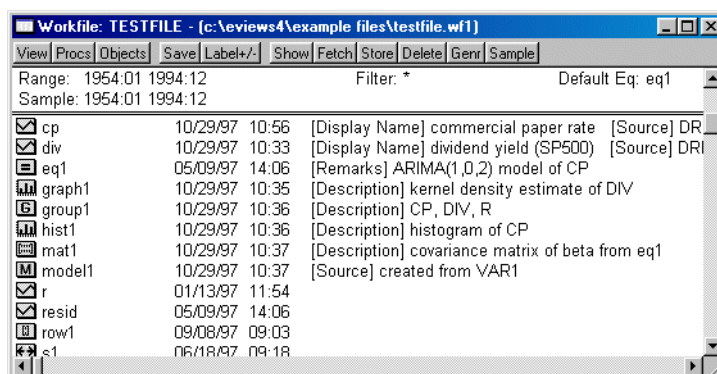
tells EViews to display all objects that begin with any two characters followed by a Y and any or no characters, and all objects that contain the letter Z. Wildcards may also be used

in more general settings—a complete description of the use of wildcards in EViews is provided in [Appendix C, “Wildcards”, on page 657](#).

When you specify a display filter, the Filter description in the workfile window changes to reflect your request. EViews always displays the current string used in matching names. Additionally, if you have chosen to display a subset of EViews object types, a “-” will be displayed in the Filter description at the top of the workfile window.

Display Comments

You can change the default workfile display to show additional information about your objects. If you select **View/Display Comments (Label + -)**, EViews will toggle between the standard workfile display format, and a display which provides additional information about the date the object was created or updated, as well as the label information that you may have attached to the object.



Display Letter Format

You can choose **View/Name Display...** in the workfile toolbar to specify whether EViews should use upper or lower case letters when it displays the workfile directory. The default is lower case.

Object Basics

Information in EViews is stored in *objects*. Each object consists of a collection of information related to a particular area of analysis. For example, a *series object* is a collection of information related to a set of observations on a particular variable. An *equation object* is a collection of information related to the relationship between a collection of variables.

Note that an object need not contain only one type of information. For example, an estimated equation object contains not only the coefficients obtained from estimation of the

equation, but also a description of the specification, the variance-covariance matrix of the coefficient estimates, and a variety of statistics associated with the estimates.

Associated with each type of object is a set of views and procedures which can be used with the information contained in the object. This association of views and procedures with the type of data contained in the object is what we term the *object oriented* design of EViews.

The object oriented design simplifies your work in EViews by organizing information as you work. For example, since an equation object contains all of the information relevant to an estimated relationship, you can move freely between a variety of equation specifications simply by working with different equation objects. You can examine results, perform hypothesis and specification tests, or generate forecasts at any time. Managing your work is simplified since only a single object is used to work with an entire collection of data and results.

This brief discussion provides only the barest introduction to the use of objects. The remainder of this section will provide a more general description of EViews objects. Subsequent chapters will discuss series, equations, and other object types in considerable detail.

Object Data

Each object contains various types of information. For example, series, matrix, vector, and scalar objects, all contain mostly numeric information. In contrast, equations and systems contain complete information about the specification of the equation or system, and the estimation results, as well as references to the underlying data used to construct the estimates. Graphs and tables contain numeric, text, and formatting information.

Since objects contain various kinds of data, you will want to work with different objects in different ways. For example, you might wish to compute summary statistics for the observations in a series, or you may want to perform forecasts based upon the results of an equation. EViews understands these differences and provides you with custom tools, called views and procedures, for working with an object's data.

Object Views

There is more than one way to examine the data in an object. Views are tabular and graphical windows that provide various ways of looking at the data in an object.

For example, a series object has a spreadsheet view, which shows the raw data, a line graph view, a bar graph view, a histogram-and-statistics view, and a correlogram view. Other views of a series include distributional plots, QQ-plots, and kernel density plots. Series views also allow you to compute simple hypothesis tests and statistics for various subgroups of your sample.

An equation object has a representation view showing the equation specification, an output view containing estimation results, an actual-fitted-residual view containing plots of fitted values and residuals, a covariance view containing the estimated coefficient covariance matrix, and various views for specification and parameter tests.

Views of an object are displayed in the object's window. Only one window can be opened for each object and each window displays only a single view of the object at a time. You can change views of an object using the **View** menu located in the object window's toolbar or the EViews main menu.

Perhaps the most important thing to remember about views is that views normally do not change data outside the object. Indeed, in most cases, changing views only changes the display format for the data, and not the data in the object itself.

Object Procedures



















Most EViews objects also have *procedures*, or *procs*. Like views, procedures often display tables or graphs in the object's window. Unlike views, however, procedures alter data, either in the object itself or in another object.

Many procedures create new objects. For example, a series object contains procedures for smoothing or seasonally adjusting time series data and creating a new series containing the smoothed or adjusted data. Equation objects contain procedures for generating new series containing the residuals, fitted values, or forecasts from the estimated equation.

You select procedures from the **Procs** menu on the object's toolbar or from the EViews main menu.

Object Types

The most common objects in EViews are series and equation objects. There are, however, a number of different types of objects, each of which serves a unique function. Most objects are represented by a unique icon which is displayed in the object container window:

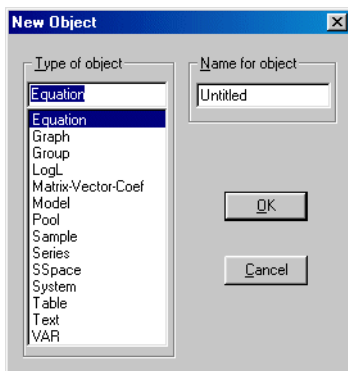
	Coefficient Vector		Scalar
	Equation		Series
	Graph		Sspace (State Space)
	Group		System
	Logl (Log Likelihood)		Sym (Symmetric Matrix)
	Matrix		Table
	Model		Text
	Pool (Time Series / Cross-Section)		Var (Vector Autoregression)
	Sample		Vector/Row Vector

Despite the fact that they are also objects, object containers do not have icons since they cannot be placed in other object containers—thus, workfiles and databases do not have icons since they cannot be placed in other workfiles or databases.

Creating, Selecting, and Opening Objects

Creating Objects

To create an object, you must first make certain that you have an open workfile container and that the workfile window is active. Next, select **Objects/New Object...** from the main menu. Until you have created or loaded a workfile, this selection is unavailable. After you click on the menu entry, you will see the following dialog box:



You can click on the type of object you want, optionally provide a name and then click on **OK**. For some object types, another dialog box will open prompting you to describe your object in more detail. For most objects, however, the object window will open immediately.

For example, if you select **Equation**, you will see a dialog box prompting you for additional information. Alternatively, if you click on **Series** and then select **OK**, you will see an object window (series window) displaying the spreadsheet view of an UNTITLED series:

Last updated: 07/21/00 - 09:23		
1954:01	NA	
1954:02	NA	
1954:03	NA	
1954:04	NA	
1954:05	NA	
1954:06	NA	
1954:07	NA	
1954:08	NA	
1954:09		

We will discuss object windows in greater detail in [“The Object Window” on page 46](#).

Objects can also be created by applying procedures to other objects or by freezing an object view (see [“Freezing Objects” on page 51](#)).

Selecting Objects

Creating a new object will not always be necessary. Instead, you may want to work with an existing object. One of the fundamental operations in EViews is *selecting* one or more objects from the workfile directory.

The easiest way to select objects is to point-and-click, using the standard Windows conventions for selecting contiguous or multiple items if necessary ([“Selecting and Opening Items” on page 8](#)). Keep in mind that if you are selecting a large number of items, you may find it useful to use the display filter before beginning to select items.

In addition, the **View** button in the workfile toolbar provides convenient selection shortcuts:

- **Select All** selects all of the objects in the workfile with the exception of the C coefficient vector and the RESID series.
- **Deselect All** eliminates any existing selections.

Opening Objects

Once you have selected your object or objects, you will want to open your selection, or create a new object containing the selected objects. You can do so by double clicking anywhere in the highlighted area.

If you double click on a single selected object, you open an object window.

If you select multiple graphs or series and double click, a pop-up menu appears giving you the option of creating and opening new objects (group, equation, VAR, graph) or displaying each of the selected objects in its own window.

Note that if you select multiple graphs and double click or select **View/Open as One Window**, all of the graphs are merged into a single graph that is displayed in a window.

Other multiple item selections are not valid, and will either issue an error or will simply not respond when you double click.

When you open an object, EViews will display the current view. In general, the current view of an object is the view that was displayed the last time the object was opened (if an object has never been opened, EViews will use a default view). The exception to this general rule is for those views that require significant computational time. In this latter case, the current view will revert to the default.

Showing Objects

An alternative method of selecting and opening objects is to “show” the item. Click on the **Show** button on the toolbar, or select **Quick/Show...** from the menu and type in the object name or names.

Showing an object works exactly as if you first selected the object or objects, and then opened your selection. If you enter a single object name in the dialog box, EViews will open the object as if you double clicked on the object name. If you enter multiple names, EViews will always open a single window to display results, creating a new object if necessary.

The **Show** button can also be used to display functions of series, also known as auto-series. All of the rules for auto-series that are outlined in [“Database Auto-Series” on page 119](#), will apply.

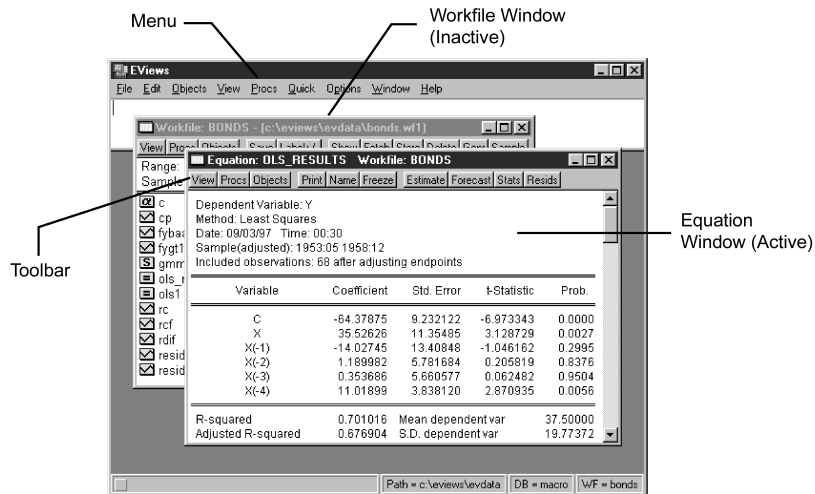
The Object Window

We have been using the term *object window* somewhat loosely in the previous discussion of the process of creating and opening objects. Object windows are the windows that are displayed when you open an object or object container. An object’s window will contain either a view of the object, or the results of an object procedure.

One of the more important features of EViews is that you can display object windows for a number of items at the same time. Managing these object windows is similar to the task of managing pieces of paper on your desk.

Components of the Object Window

Let’s look again at a typical object window:



Here, we see the equation window for OLS_RESULTS. First, notice that this is a standard window which can be closed, resized, minimized, maximized, and scrolled both vertically and horizontally. As in other Windows applications, you can make an object window active by clicking once on the titlebar, or anywhere in its window. Making an object window active is equivalent to saying that you want to work with that object.

Second, note that the titlebar of the object window identifies the object type, name, and object container (in this case, the BONDS workfile). If the object is itself an object container, the container information is replaced by directory information.

Lastly, at the top of the window there is a toolbar containing a number of buttons that provide easy access to frequently used menu items. These toolbars will vary across objects—the series object will have a different toolbar from an equation or a group or a VAR object.

There are, however, several buttons that are found on all object toolbars:

- The **View** button lets you change the view that is displayed in the object window. The available choices will differ, depending upon the object type.
- The **Procs** button provides access to a menu of procedures that are available for the object.
- The **Objects** button lets you manage your objects. You can store the object on disk, name, delete, copy, or print the object.
- The **Print** button lets you print the current view of the object (the window contents).
- The **Name** button allows you to name or rename the object.

- The **Freeze** button creates a new object graph, table, or text object out of the current view.

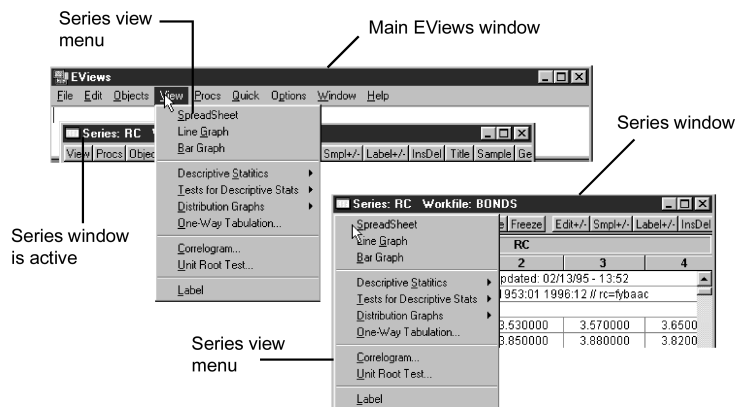
The other buttons on the series toolbar are specific to a series object and are described in [Chapter 7, “Series”, on page 151](#).

Menus and the Object Toolbar

As we have seen, the toolbar provides a shortcut to frequently accessed menu commands. There are a couple of subtle, but important, points associated with this relationship that deserve special emphasis:

- Since the toolbar simply provides a shortcut to menu items, you can always find the toolbar commands in the menus.
- This fact turns out to be quite useful if your window is not large enough to display all of the buttons on the toolbar. You can either enlarge the window so that all of the buttons are displayed, or you can access the command directly from the menu.
- The toolbar and menu *both* change with the object type. In particular, the contents of the View menu and the Procs menu will always change to reflect the type of object (series, equation, group, etc.) that is active.

The toolbars and menus themselves vary in how much they differ across objects. For example, the View and Procs drop-down menus differ for every object type. When the active window is displaying a series window, the menus provide access to series views and series procedures. Alternatively, when the active window is a group window, clicking on **View** or **Procs** provides access to the different set of items associated with group objects.



The figure above illustrates the relationship between the View toolbar button and the View menu when the series window is the active window. In the left side of the illustration, we see a portion of the *EViews window*, as it appears, after you click on View in the main

menu (note that the RC series window is the active window). On the right, we see a depiction of the *series window* as it appears after you click on the **View** button in the series toolbar. Since the two operations are identical, the two drop-down menus are identical.

In contrast to the View and Procs menus, the Objects menu does not, in general, vary across objects. An exception occurs, however, when an object container window (a workfile or database window) is active. In this case, clicking on **Objects** in the toolbar, or selecting **Objects** from the menu provides access to menu items for manipulating the objects in the container.

Working with Objects

Naming Objects

Objects may be named or unnamed. When you give an object a name, the name will appear in the directory of the workfile, and the object will be saved as part of the workfile when the workfile is saved.

You must name an object if you wish to keep its results. If you do not name an object, it will be called “UNTITLED”. Unnamed objects are not saved with the workfile, so they are deleted when the workfile is closed and removed from memory.

To name or rename an object, first open the object window by double clicking on its icon, or by clicking on **Show** on the workfile toolbar, and entering the object name. Next, click on the **Name** button on the object window, and enter the name (up to 16 characters), and optionally, a display name to be used when labelling the object in tables and graphs. If no display name is provided, EViews will use the object name.

You can also rename an object from the workfile window by selecting **Objects/Rename Selected...** and then specifying the new object name. This method saves you from first having to open the object.

The following names are reserved and should not be used as object names: ABS, ACOS, AR, ASIN, C, CON, CNORM, COEF, COS, D, DLOG, DNORM, ELSE, ENDIF, EXP, LOG, LOGIT, LPT1, LPT2, MA, NA, NRND, PDL, RESID, RND, SAR, SIN, SMA, SQR, and THEN.

EViews accepts both capital and lower case letters in the names you give to your series and other objects, but does not distinguish between names based on case. Its messages to you will follow normal capitalization rules. For example, ‘SALES’, ‘sales’, and ‘sAles’ are all the same object in EViews. For the sake of uniformity, we have written all examples of input using names in lower case, but you should feel free to use capital letters instead.

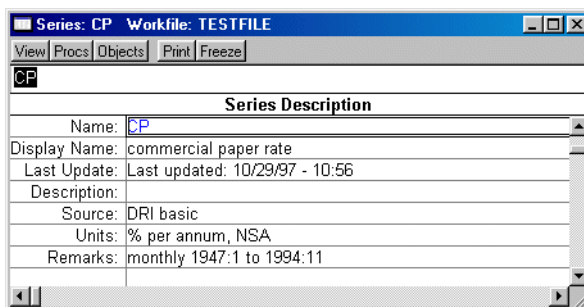
Despite the fact that names are not case sensitive, when you enter text information in an object, such as a plot legend, or label information, your capitalization will be fully preserved.

By default, EViews allows only one untitled object of a given type (one series, one equation, etc.). If you create a new untitled object of an existing type, you will be prompted to name the original object, and if you do not provide one, EViews will replace the original untitled object with the new object. The original object will not be saved. If you prefer, you can instruct EViews to retain all untitled objects during a session but you must still name the ones you want to save with the workfile. See [“Window and Font Options” on page 647](#).

Labeling Objects

In addition to the display name described above, EViews objects have label fields where you can provide extended annotation and commentary. To view these fields, select **View/Label** from the object window:

This is the label view of an unmodified object. By default, every time you modify the object, EViews automatically records the modification in a History field that will be appended at the bottom of the label view.



You can edit any of the fields, except the **Last Update** field.

Simply click in the field cell that you want to edit. All fields, except the **Remarks** and **History** fields, contain only one line. The **Remarks** and **History** fields can contain multiple lines. Press ENTER to add a new line to these two fields.

These annotated fields are most useful when you want to search for an object stored in an EViews database. Any text that is in the fields is searchable in an EViews database; see [“Querying the Database” on page 123](#), for further discussion.

Copying Objects

There are two distinct methods of duplicating the information in an object: copying and freezing.

If you select **Object/Copy** from the menu, EViews creates a new untitled object containing an exact copy of the original object. By exact copy, we mean that the new object duplicates all the features of the original (except for the name). It contains all of the views and procedures of the original object and can be used in future analyses just like the original object.

You can also copy an object from the workfile window. Simply highlight the object and click on **Object/Copy Selected...**, or click on **Object/Copy Selected...** and specify the destination name for the object.

We mention here that **Copy** is a very general and powerful operation with many additional features and uses. For example, you can copy objects across both workfiles and databases using wildcards and patterns. See [“Copying Objects” on page 115](#), for details on these additional features.

Copy-and-Pasting Objects

The standard EViews copy command makes a copy of the object in the same workfile. When two workfiles are in memory at the same time, you may copy objects between them using Copy-and-Paste.

Highlight the objects you wish to copy in the source workfile. Then select **Edit/Copy** from the main menu.

Select the destination workfile by clicking on its titlebar. Then select **Edit/Paste** from the main menu. EViews will place named copies of all of the highlighted objects in the destination workfile, prompting you to replace existing objects with the same name.

If the source and destination workfiles are of different frequency, frequency conversion (if possible) is applied to series objects before placing them in the destination workfile. See [“Frequency Conversion” on page 72](#), for the exact rules by which frequencies are converted.

Freezing Objects

The second method of copying information from an object is to *freeze* a view of the object. If you click **Object/Freeze Output** or press the **Freeze** button on the object’s toolbar, a table or graph object is created that duplicates the current *view* of the original object.

Before you press **Freeze**, you are looking at a view of an object in the object window. Freezing the view makes a copy of the view and turns it into an independent object that will remain even if you delete the original object. A frozen view does not necessarily show what is currently in the original object, but rather shows a snapshot of the object at the moment you pushed the button. For example, if you freeze a spreadsheet view of a series, you will see a view of a new *table object*; if you freeze a graphical view of a series, you will see a view of a new *graph object*.

The primary feature of freezing an object is that the tables and graphs created by freeze may be edited for presentations or reports. Frozen views do not change when the workfile sample or data change.

Deleting Objects

To delete an object or objects from your workfile, select the object or objects in the workfile directory. When you have selected everything you want to delete, click **Delete** or

Objects/Delete Selected on the workfile toolbar. EViews will prompt you to make certain that you wish to delete the objects.

Printing Objects

Choosing **View/Print Selected** from the workfile window prints the default view for all of the selected objects.

To print the currently displayed view of an object, push the **Print** button on the object window toolbar. You can also choose **File/Print** or **Objects/Print** on the main EViews menu bar.

You may print the default view of more than one object at a time by selecting the objects in the workfile window and choosing **View/Print Selected** from the workfile toolbar.

The print commands normally send a view or procedure output to the current Windows printer. You may specify instead that the output should be saved in the workfile as a table or graph, or spooled to an ASCII text file on disk. Details are provided in [Chapter 10, “Graphs, Tables, and Text Objects”](#), on page 243 and [Appendix A, “Global Options”](#), on page 647.

Storing Objects

EViews provides three ways to save your data on disk. You have already seen how to save entire workfiles, where all of the objects in the workfile are saved together in a single file with the .WF1 extension. You may also store individual objects in their own *data bank* files. They may then be fetched into other workfiles.

We will defer a full discussion of storing objects to data banks and databases until [Chapter 3](#). For now, note that when you are working with an object, you can place it in a data bank or database file by clicking on the **Objects/Store to DB...** button on the object's toolbar or menu. EViews will prompt you for additional information.

You can store several objects, by selecting them in the workfile window and then pressing the **Objects/Store selected to DB...** button on the workfile toolbar or menu.

Fetching Objects

You can fetch previously stored items from a data bank or database. One of the common methods of working with data is to create a workfile and then fetch previously stored data into the workfile as needed.

To fetch objects into a workfile, select **Objects/Fetch from DB...** from the workfile menu or toolbar. You will see a dialog box prompting you for additional information for the fetch: objects to be fetched, directory and database location, as applicable.

See “[Fetching Objects from the Database](#)” on page 114, for details on the advanced features of the fetch procedure.

Updating Objects

Updating works like fetching objects, but requires that the objects be present in the workfile. To update objects in the workfile, select them from the workfile window, and click on **Objects/Update from DB...** from the workfile menu or toolbar. The Fetch dialog will open, but with the objects to be fetched already filled in. Simply specify the directory and database location and click **OK**.

The selected objects will be replaced by their counterparts in the data bank or database.

See [Chapter 6, “EViews Databases”, on page 107](#), for additional details on the process of updating objects from a database.

Copy-and-Paste of Object Information

You can copy the list of object information displayed in a workfile or database window to the Windows clipboard and paste the list to other program files such as word processing files or spreadsheet files. Simply highlight the objects in the workfile directory window, select **Edit/Copy** (or click anywhere in the highlighted area, with the right mouse button, and select **Copy**). Then move to the application (word processor or spreadsheet) where you want to paste the list, and select **Edit/Paste**.

If only names are displayed in the window, EViews will copy a single line containing the highlighted names to the clipboard, with each name separated by a space. If the window contains additional information, either because **View/Display Comments (Label + /-)** has been chosen in a workfile window or a query has been carried out in a database window, each name will be placed in a separate line along with the additional information.

Note that if you copy-and-paste the list of objects into another EViews workfile, the objects themselves will be copied.

Commands

To create a new workfile, follow the `workfile` command with the name of the workfile. For example, if you type

```
workfile test1
```

EViews brings up the Workfile Range dialog to specify the range of the new workfile TEST1.

To save a workfile, follow the `save` command with a name for the saved workfile. For example,

```
save test2
```

saves the active workfile under the name TEST2 in the default directory.

See the *Command and Programming Reference* for a complete list of commands and options available in EViews.

For More Info...

This concludes our brief introduction to the EViews program. You should now be well on your way to a full understanding of the user-friendly EViews approach to forecasting and statistical analysis.

For further details on any aspect of the program, be certain to use your on-line help system, or consult the complete EViews *User's Guide* and the EViews *Command and Programming Reference*, both of which are provided in .PDF format on your CD-ROM.

Chapter 4. Basic Data Handling

The process of entering, reading, editing, manipulating, and generating data forms the foundation of most data analyses. Accordingly, most of your time in EViews will probably be spent working with data. EViews provides you with a sophisticated set of data manipulation tools that make these tasks as simple and straightforward as possible.

This chapter describes the fundamentals of working with data in EViews. There are three cornerstones of data handling in EViews: the two most common data objects, *series* and *groups*, and the use of *samples*. We begin with a brief description of series, groups, and samples, and then discuss basic input, output, and editing of data.

In the next chapter, we discuss EViews' powerful language for generating and manipulating the data in series and groups.

Data Objects

The actual numeric values that make up your data will generally be held in one or more of EViews' data objects (series, groups, matrices, vectors, scalars). For most users, series and groups will be, by far, the most important objects, so they will be the focus of our discussion. Matrices, vectors, and scalars are discussed at length in the *Command and Programming Reference*.

The following discussion is intended to provide only a brief introduction to the basics of series and groups. Our goal is to describe the basics of data input and output in EViews. An in-depth discussion of series and group objects follows in subsequent chapters.

Series

An EViews series contains a set of observations on a variable. Associated with each observation in the series is a date or observation label. For series in dated workfiles, the observations are presumed to be observed regularly over time. For undated data, the observations are not assumed to follow any particular frequency.

Creating a series

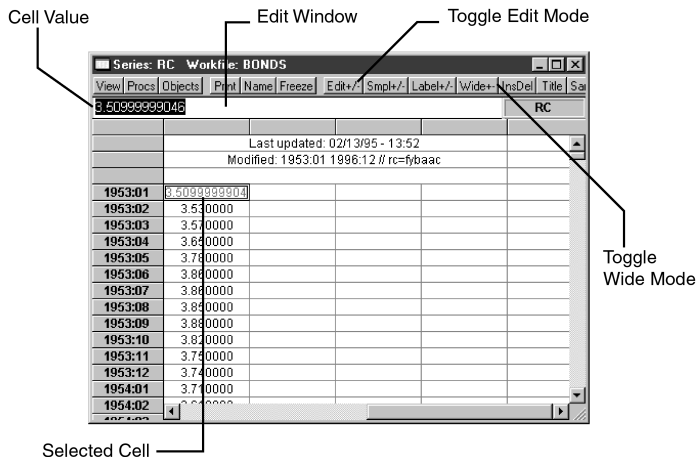
One method of creating a series is to select **Object/New Objects...** from the menu, and then to select **Series**. You may, at this time, provide a name for the series, or you can let the new series be untitled. Click **OK**. EViews will open a spreadsheet view of the new series object. All of the observations in the series will be assigned the missing value code "NA". You can then edit or use expressions to assign values for the series.

The second method of creating a series is to generate the series using mathematical expressions. Click on **Quick/Generate Series...** and enter the expression defining the series. We will discuss this method in depth in the next chapter.

Editing a series

You can edit individual values of the data in a series.

- First open the spreadsheet view of the series. If the series window display does not show the spreadsheet view, click on the **Sheet** button, or select **View/Spreadsheet**, to change the default view.
- Next, make certain that the spreadsheet window is in *edit mode*. EVIEWS provides you with the option of protecting the data in your series by turning off the ability to edit from the spreadsheet window. You can use the **Edit +/-** button on the toolbar to toggle between edit mode and *protected mode*:



Here we see an edit mode example for a series spreadsheet window. Notice the presence of the edit window just underneath the series toolbar containing the value of RC in 1953:01, and the double box around the selected cell in the spreadsheet—neither are present in protected mode.

- To change the value for an observation, select the cell, type in the value, and press ENTER. For example, to change the value of RC in 1953:01, simply click on the cell containing the value, type the new value in the edit window, and press ENTER.

Note that some cells are protected. If you select one of the protected cells, EVIEWS will display a message in the edit window telling you that the cell cannot be edited.

- When you have finished editing, you should protect yourself from inadvertently changing values of your data by clicking on **Edit +/-** to turn off edit mode.

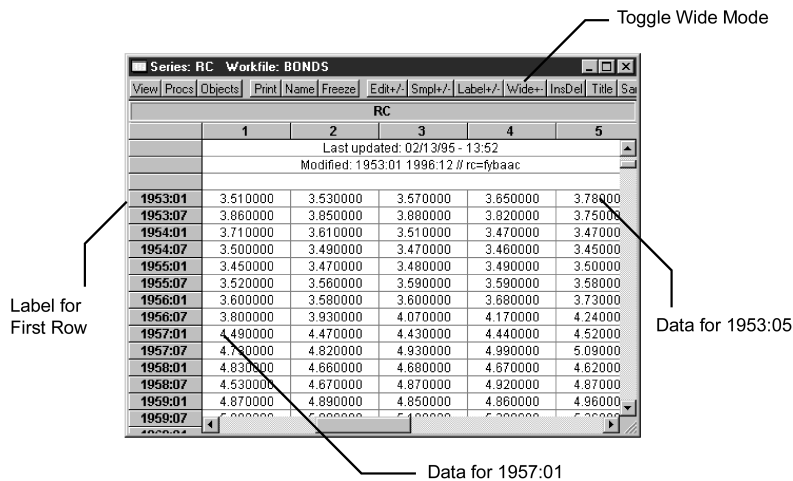
Changing the Spreadsheet Display

EViews provides you with several different ways of looking at your data in spreadsheet form.

The *narrow display* displays the observations for the series in a single column, with date labels in the margin.

The *wide display* arranges the observations from left to right and top to bottom, with the label for the first observation in the row displayed in the margin. For dated workfiles, EViews will, if possible, arrange the data in a form which matches the frequency of the data. Thus, semi-annual data will be displayed with two observations per row, quarterly data will contain four observations per row, and monthly data will contain six observations in each row.

You can change the display to show the observations in your series in a single column by clicking on the **Wide** +/- button on the spreadsheet view toolbar (you may need to resize the series window to make this button visible). For example, toggling the **Wide** +/- button switches the display between the compact display (as depicted), and the single column display:



This compact display format is useful when you wish to arrange the observations with data for a particular season displayed in the columns.

By default, all observations in the workfile are displayed, even those observations not in the current sample. By pressing **Smpl** +/- you can toggle between showing all observations in the workfile, and showing only those observations in the current sample.

There are two features that you should keep in mind as you toggle between the various display settings:

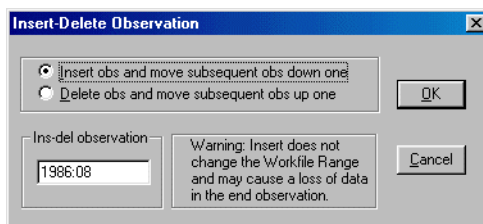
- If you choose to display only the observations in the current sample, EViews will switch to single column display.
- If you switch to wide display, EViews automatically turns off the display filter so that all observations in the workfile are displayed.

One consequence of this behavior is that if you begin with a narrow display of observations in the current sample, click on **Wide** +/- to switch to wide display, and then press the **Wide** +/- button again, EViews will provide a narrow display of all of the observations in the workfile. To return to the original narrow display of the current sample, you will need to press the **Smpl** +/- button again.

Inserting and deleting observations in a series

You can also insert and delete observations in the series. To insert an observation, first click on the cell where you want the new observation to appear. Next, click on the **InsDel** button. You will see a dialog asking whether you wish to insert or delete an observation at the current position:

You may change the location by editing the observation box. If you choose to insert an observation, EViews will insert a missing value at the appropriate position and push all of the observations down so that the last observation will be lost from the workfile. If you wish to preserve this observation, you will have to expand the workfile before inserting observations. If you choose to delete an observation, all of the remaining observations will move up, so that you will have a missing value at the end of the workfile range.



Groups

When working with multiple series, you will often want to create a group object to help you manage your data. A group is a list of series names (and potentially, mathematical expressions) that provides simultaneous access to all of the elements in the list.

With a group, you can refer to sets of variables using a single name. Thus, a set of variables may be analyzed, graphed, or printed using the group object, rather than each one of the individual series. Therefore, groups are often used in place of entering a lengthy list of names. Once a group is defined, you can use the group name in many places to refer to all of the series contained in the group.

You will also create groups of series when you wish to analyze or examine multiple series at the same time. For example, groups are used in computing correlation matrices, testing for cointegration and estimating a VAR or VEC, and graphing series against one another.

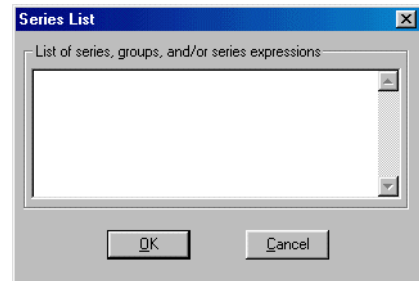
Creating Groups

There are several ways to create a group. Perhaps the easiest method is to select **Objects/ New Object...** from the menu or workfile toolbar, click on **Group**, and if desired, name the object.

You should enter the names of the series to be included in the group, and then click **OK**. A group window will open showing a spreadsheet view of the group.

You may have noticed that the dialog allows you to use group names and series expressions. If you include a group name, all of the series in the named group will be included in the new group.

For example, suppose that the group GR1 contains the series X, Y, and Z, and you create a new group GR2, which contains GR1 and the series A and B. Then GR2 will contain X, Y, Z, A and B. Bear in mind that only the series contained in GR1, not GR1 itself, are included in GR2; if you later add series to GR1, they will not be added to GR2.



Series expressions will be discussed in greater depth later. For now, it suffices to note that series expressions are mathematical expressions that may involve one or more series (e.g. $7/2$ or $3*X*Y/Z$). EViews will automatically evaluate the expressions for each observation and display the results as if they were an ordinary series. Users of spreadsheet programs will be familiar with this type of automatic recalculation.

For example, here is a spreadsheet view of an untitled group containing the series RC, and series expressions for the lag of RG, $RG(-1)$, and a series expression involving RC and RG.

An equivalent method of creating a group is to select **Quick/ Show...**, or to click on the **Show** button on the workfile toolbar, and then to enter the

obs	RC	RG(-1)	2*RC/RG
1953:01	3.510000	NA	NA
1953:02	3.530000	0.000000	NA
1953:03	3.570000	0.000000	NA
1953:04	3.650000	0.000000	2.579505
1953:05	3.780000	2.830000	2.478689
1953:06	3.860000	3.050000	2.482315
1953:07	3.860000	3.110000	2.634812
1953:08	3.850000	2.930000	2.610169
1953:09	3.880000	2.950000	2.703833
1953:10	3.820000	2.870000	2.872180
1953:11	3.750000	2.680000	2.798507
1953:12	3.740000	2.680000	2.888031
1954:01	3.710000	2.590000	2.991935
1954:02			

list of series, groups and series expressions to be included in the workfile. This method differs from using **Objects/New Object...** only in not allowing you to name the object at the time it is created.

You can also create an empty group that may be used for entering new data from the keyboard or pasting data copied from another Windows program. These methods are described in detail in “[Entering Data](#)” on page 64 and “[Copying and Pasting](#)” on page 65.

Editing in a Group

Editing data in a group is similar to editing data in a series. Open the group window, and click on **Sheet**, if necessary, to display the spreadsheet view. If the group spreadsheet is in protected mode, click on **Edit +/-** to enable edit mode, then select your cell, enter the new value, and press RETURN. The new number should appear in the spreadsheet.

Since groups are simply references to series, editing the series within a group changes the values in the original series.

As with series spreadsheet views, you may click on **Smpl +/-** to toggle between showing all of the observations in the workfile and showing only those observations in the current sample. Unlike the series window, the group window always shows series in a single column.

Samples

One of the most important concepts in EViews is the *sample* of observations. The sample is the set (often a subset) of observations in the workfile to be included in displays and in statistical procedures. Samples may be specified using ranges of observations and/or “if conditions” that observations must satisfy to be included.

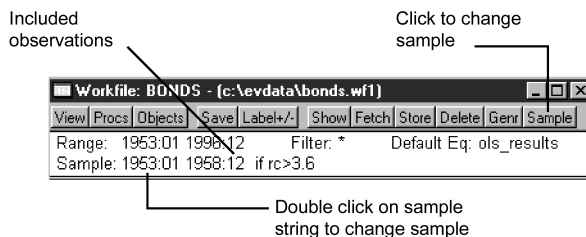
For example, you can tell EViews that you want to work with observations from 1953:1 to 1970:12 and 1995:1 to 1996:12. Or you may want to work with data from 1953:1 to 1958:12 where observations on the series RC exceed 3.6.

The Workfile Sample

When you create a workfile, the *global* or *workfile sample* is set initially to be the entire range of the workfile. The workfile sample tells EViews what set of observations you wish to use for subsequent operations. Unless you want to work with a different set of observations, you will not need to reset the workfile sample.

You can always tell the current workfile sample of observations by looking at the top of your workfile window:

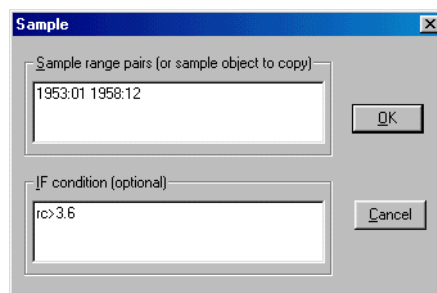
Here the sample consists of those observations between 1953:01 and 1958:12 for which the value of the RC exceeds 3.6.



Changing the Sample

There are three ways to set the global sample: you can click on the **Sample** button in the workfile toolbar, you can select **Objects/Sample...** from the menu, or you could click on the sample string in the workfile window display.

In the upper edit window you will enter one or more pairs of dates. Each pair identifies a starting and ending observation for a set of observations to be included in the sample. For example, if you entered the string “1950 1980 1990 1995”, EViews will use observations for 1950 through 1980 and observations for 1990 through 1995 in subsequent operations; observations from 1981 through 1989 will be excluded.



EViews provides some special shortcuts that may make entering sample range pairs easier. First, you can use the keyword “@all”, to refer to the entire workfile range. In the workfile above, entering “@all” in the dialog is equivalent to entering “1953:1 1996:12”. Furthermore, you may use “@first” and “@last” to refer to the first and last observation in the workfile. Thus, all three of the following sample ranges are identical:

```
@all
@first 1996:12
1953:1 @last
```

Sample range elements may contain mathematical expressions to create date offsets. This feature can be particularly useful in setting up a fixed width window of observations. For example, in the workfile above, the sample strings

```
1953:1 1953:1+11
```

define a sample that includes the 12 observations in the calendar year beginning in 1953:1.

While EViews expects date offsets that are integer values, there is nothing to stop you from adding or subtracting non-integer values—EViews will automatically convert the number to an integer. You should be warned, however, that the conversion behavior is not guaranteed to be well-defined. If you must use non-integer values, you are strongly encouraged to use the “@round”, “@floor” or “@ceil” functions to enforce the desired behavior.

The lower part of the sample window allows you to add conditions to the sample specification. The sample is the intersection of the set of observations defined by the range pairs in the upper window and the set of observations defined by the “if” conditions in the lower window. For example, if you enter:

```
Upper window: 1980 1993
Lower window: incm > 5000
```

the sample includes observations for 1980 through 1993 where the series INCM is greater than 5000.

Similarly, if you enter:

```
Upper window: 1958:1 1998:1
Lower window: gdp > gdp(-1)
```

all observations between the first quarter of 1958 and the last quarter of 1998, where GDP has risen from the previous quarter, will be included.

The “or” and “and” operators allow for the construction of more complex expressions. For example, suppose you now wanted to include in your analysis only those individuals whose income exceeds 5000 dollars per year and who have at least 13 years of education. Then you can enter:

```
Upper window: @all
Lower window: income > 5000 and educ > = 13
```

Multiple range pairs and “if” conditions may also be specified:

```
Upper window: 50 100 200 250
Lower window: income > = 4000 and educ > 12
```

includes undated workfile observations 50 through 100 and 200 through 250, where the series INCOME is greater than or equal to 4000 and the series EDUC is greater than 12.

You can create even more elaborate selection rules by including EViews built-in functions:

```
Upper window: 1958:1 1998:1
Lower window: (ed > = 6 and ed < = 13) or earn < @mean(earn)
```

includes all observations where the value of the variable ED falls between 6 and 13, or where the value of the variable EARN is lower than its mean. Note that you may use

parentheses to group the conditions and operators when there is potential ambiguity in the order of evaluation.

It is possible that one of the comparisons used in the conditioning statement will generate a missing value. For example, if an observation on INCM is missing, then the comparison $\text{INCM} > 5000$ is not defined for that observation. EViews will treat such missing values as though the condition were false, and the observation will not be included in the sample.

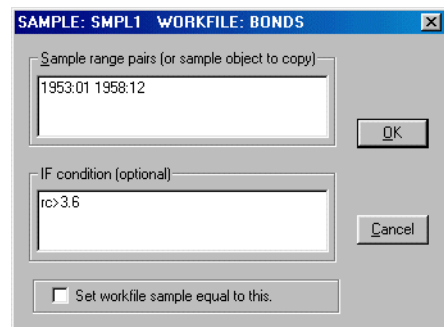
Sample Objects

As you have seen, it is possible to develop quite elaborate selection rules for the workfile sample. However, it can quickly become cumbersome and time-consuming to re-enter these rules if you change samples frequently. Fortunately, EViews provides you with a method of saving sample information in a sample object which can then be referred to by name. If you work with various well-defined subsets of your data, you will soon find sample objects to be indispensable.

Creating a Sample Object

To create a sample object, select **Objects/New...** When the New Objects dialog appears, select **Sample** and optionally, provide a name. If you do not provide a name, EViews will automatically assign one for you (sample objects may not be untitled). Click on **OK** and EViews will open a dialog:

Here is a partially filled in sample object dialog named SMPL1. Notice that while this dialog looks very similar to the one we described above for setting the sample, there are minor cosmetic differences: the name of the sample object appears in the title bar, and there is a check box for setting the workfile sample equal to this sample object.



These cosmetic differences reflect the two distinct purposes of the dialog: (1) to define the sample object, and (2) to set the workfile sample. Since EViews separates the act of defining the sample object from the act of setting the workfile sample, you can define the object without changing the workfile sample, and vice versa.

To define the sample object, you should fill out this dialog as before and click on **OK**. The sample object now appears in the workfile directory.

Using a Sample Object

Once created, a sample object can be referred to by its name almost anywhere that EViews prompts you for a sample description.

Alternatively, you can use a previously defined sample object directly to set the workfile sample. Simply open a sample object by double clicking on the name or icon. This will reopen the dialog. If you wish to change the sample object, edit the sample specification; otherwise, simply click **Set workfile sample** on the check box and click on **OK**.

Command Window Methods

You may find it easier to work with samples from the command window. Instead of specifying the sample by clicking on **Sample** or selecting **Quick/Sample...**, you can always enter the sample from the command window using the `smpl` command. Simply click on the command window to make it active, and type the keyword “`smpl`”, followed by the sample string:

```
smpl 1955:1 1958:12 if rc>3.6
```

and then press ENTER (notice, in the example above, the use of the keyword “IF” to separate the two parts of the sample specification). You should see the sample change in the workfile window.

If you wish to use sample objects, you will first create the sample object using a `sample` declaration, followed by the name to be given to the sample object, then the sample string:

```
sample mysample 1955:1 1958:12 if rc>3.6
```

To set the workfile sample using the sample object, just enter the `smpl` command, followed by the sample object name. For example,

```
smpl mysample
```

will set the workfile sample according to the rules contained in the sample object `MYSAMPLE`.

Importing Data

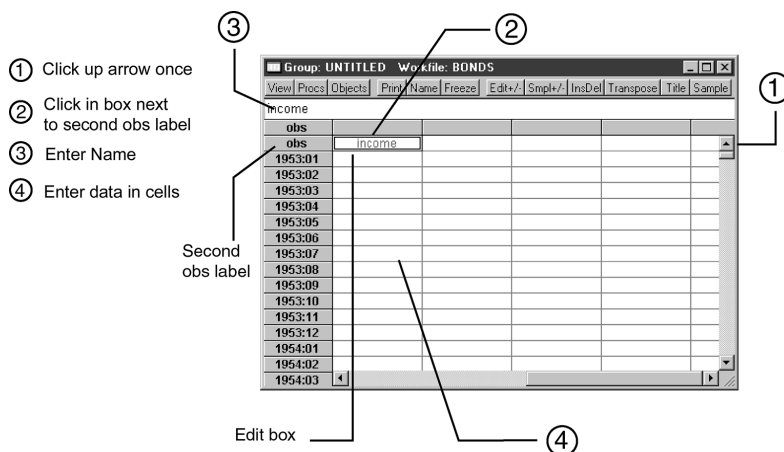
The data for your project may be available in a variety of forms. The data may be in a machine readable spreadsheet or text file that you created yourself or downloaded from the World Wide Web, or perhaps they are in a book or photocopy form.

There are several methods for getting such data into EViews. We outline the basics of data input into series and group objects from spreadsheet, text file, or printed formats, without a full discussion of working with workfiles or EViews and foreign databases. We also omit, for the moment, discussion of input into EViews matrix, vector and pool objects.

Entering Data

For small datasets in printed form, you may wish to enter the data by typing at the keyboard.

- Your first step is to open a temporary spreadsheet window in which you will enter the data. Choose **Quick/Empty Group (Edit Series)** from the main menu to open an untitled group window:



- The next step is to create and name the series. First click once on the up arrow to display the second **obs** label on the left-hand column. The row of cells next to the second **obs** label is where you will enter and edit series names.

Click once in the cell next to the second **obs** label, and enter your first series name. Here we have typed INCOME in the command window (the name in the cell changes as we type in the command window). Press RETURN.

- Repeat this procedure in subsequent columns for each additional series.

If you decide you want to rename one of your series, simply select the cell containing the series name, edit the name, and then press RETURN. EViews will prompt you to confirm the series rename.

- To enter the data, click on the appropriate cell and type the number. Pressing RETURN after entering a number will move you to the next cell. If you prefer, you can use the cursor keys to navigate the spreadsheet.
- When you are finished entering data, close the group window. If you wish, you can first name the untitled group by clicking on the **Name** button. If you do not wish to keep the group, answer **Yes** when EViews asks you to confirm the deletion.

Copying-and-Pasting

The Windows clipboard is a handy way to move data within EViews and between EViews and other software applications. It is a natural tool for importing data from Excel and other Windows applications that support Windows copy-and-paste.

Copying from Windows applications

The following discussion involves an example using an Excel spreadsheet, but the basic principles apply for other Windows applications.

Suppose you have bond yield and interest rate data in an Excel spreadsheet that you would like to bring into EViews.

Open the spreadsheet in Excel. Your first step is to highlight the cells to be imported into EViews. Since the column headings YIELD and INTEREST will be used as EViews variable names, you should highlight them as well. Since EViews understands dated data, and we are going to create a monthly workfile, you do not need to copy the date column. Instead, click on the column label B and drag to the column label C. The two columns of the spreadsheet will be highlighted:

	A	B	C	D	E	F	G
1	obs	yield	interest				
2	1953:01	3.51	0				
3	1953:02	3.53	0				
4	1953:03	3.57	0				
5	1953:04	3.65	2.83				
6	1953:05	3.78	3.05				
7	1953:06	3.86	3.11				
8	1953:07	3.86	2.93				
9	1953:08	3.85	2.95				
10	1953:09	3.88	2.87				
11	1953:10	3.82	2.66				

Select **Edit/Copy** to copy the highlighted data to the clipboard.

Pasting into New Series

Start EViews and create a new, or load an existing, monthly workfile containing the dates in the Excel spreadsheet (in our example 1953:1 through 1994:11). Make certain that the sample is set to include the same observations that you have copied onto the clipboard.

	A	B	C	D	E	F	G
1	obs	yield	interest				
2	1953:01	3.51	0				
3	1953:02	3.53	0				
4	1953:03	3.57	0				
5	1953:04	3.65	2.83				
6	1953:05	3.78	3.05				
7	1953:06	3.86	3.11				
8	1953:07	3.86	2.93				
9	1953:08	3.85	2.95				
10	1953:09	3.88	2.87				
11	1953:10	3.82	2.66				

Select **Quick/Empty Group (Edit Series)**. Note that the spreadsheet opens in edit mode so there is no need to click the **Edit** +/- button.

Here, we have created a monthly workfile with a range from 1953:1 to 1999:12. The first row of the EViews spreadsheet is labeled 1953:01. Since we are pasting in the series names, you should click on the up arrow in the scroll bar to make room for the series names.

Place the cursor in the upper-left cell, just to the right of the second obs label. Then select **Edit/Paste** from the main menu (*not* **Edit** +/- in the toolbar). The group spreadsheet will now contain the data from the clipboard.

Here, the series YIELD and INTEREST have been created in the workfile and can be used in any EViews procedure. You may now close the group window and delete the untitled group without losing the two series.

obs	YIELD	INTEREST
1953:01	3.510000	0.000000
1953:02	3.530000	0.000000
1953:03	3.570000	0.000000
1953:04	3.650000	2.830000
1953:05	3.780000	3.050000
1953:06	3.860000	3.110000
1953:07	3.860000	2.930000
1953:08	3.850000	2.950000
1953:09	3.880000	2.870000
1953:10		

Note that when importing data from the clipboard, EViews follows the Windows standard of tab-delimited free-format data with one observation per line. Since different applications use different whitespace and delimiter characters, attempting to cut-and-paste from nonstandard applications may produce unanticipated results.

Pasting into Existing Series

You can import data from the clipboard into an existing EViews series or group spreadsheet by using **Edit/Paste** in the same fashion. There are only a few additional issues to consider.

- To paste several series, you will first open a group window containing the existing series. The easiest way to do this is to click on **Show**, and then type the series names in the order they appear on the clipboard. Alternatively, you can create an untitled group by selecting the first series, click selecting each subsequent series (in order), and then double clicking to open.
- Next, make certain that the group window is in edit mode. If not, press the **Edit** +/- button to toggle between edit mode and protected mode. Place the cursor in the target cell, and select **Edit/Paste**.
- Finally, click on **Edit** +/- to return to protected mode.
- If you are pasting into a single series you will need to make certain that the series window is in edit mode, and that the series is viewed in a single column. If the series is in multiple columns, push on the **Smpl** +/- button. **Edit/Paste** the data and click on **Edit** +/- to protect the data.

Importing Data from an Spreadsheet or Text File

You can also read data directly from files created by other programs. Data may be in standard ASCII form or in either Lotus (.WKS, .WK1 or .WK3) or Excel (.XLS) spreadsheet formats.

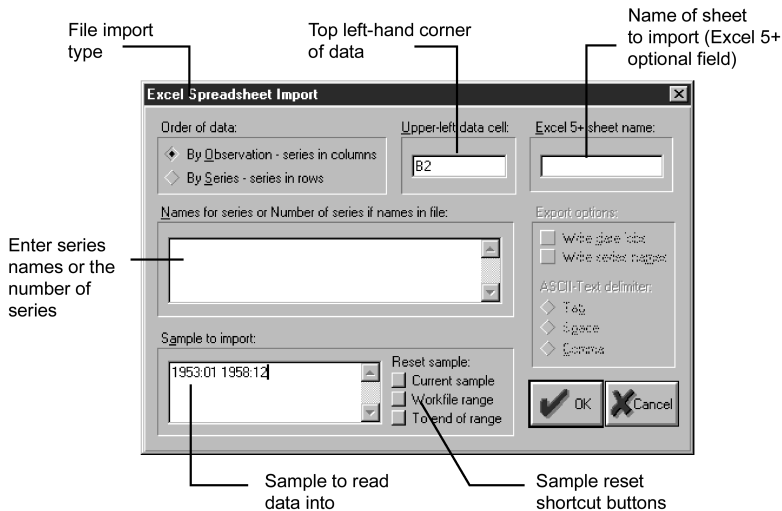
First make certain that you have an open workfile to receive the contents of the data import.

Next, click on **Procs/Import/Read Text-Lotus-Excel...** You will see a standard File dialog box asking you to specify the type and name of the file. Select a file type, navigate to the directory containing the file, and double click on the name. Alternatively, type in the name of the file that you wish to read (with full path information, if appropriate); if possible, EViews will automatically set the file type, otherwise it will treat the file as an ASCII file. Click on **Open**.

EViews will open a dialog prompting you for additional information about the import procedure. The dialog will differ greatly depending on whether the source file is a spreadsheet or an ASCII file.

Spreadsheet Import

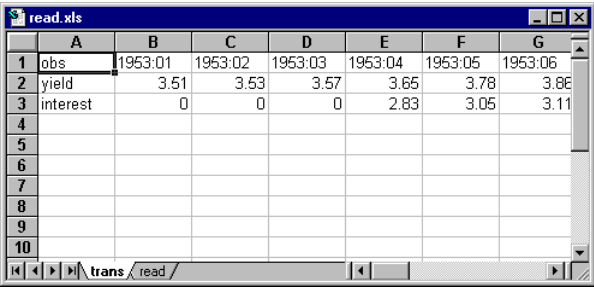
The title bar of the dialog will identify the type of file that you have asked EViews to read. Here is the dialog for importing an Excel 5 (or later versions of Excel), spreadsheet:



You will see slightly different versions of this dialog if you are reading a Lotus, or an Excel 4 (and earlier) file. Now fill in the dialog:

- First, you need to tell whether the data are ordered by observation or by series. **By observation** means that all of the data for the first observation are followed by all of the data for the second observation, etc. **By series** means that all of the data for the first variable are followed by all data for the second variable, etc. Another interpretation for “by observation” is that variables are arranged in columns while “by row” implies that all of the observations for a variable are in a single row.

Our Excel example above is organized by observation since each series is in a separate column. If the Excel data for YIELD and INTEREST were each contained in a single row as depicted, then the data should be read by series.



	A	B	C	D	E	F	G
1	obs	1953:01	1953:02	1953:03	1953:04	1953:05	1953:06
2	yield	3.51	3.53	3.57	3.65	3.78	3.86
3	interest	0	0	0	2.83	3.05	3.11
4							
5							
6							
7							
8							
9							
10							

- Next, tell EViews the location of the beginning cell (upper left-hand corner) of your actual data, not including any label or date information. In both examples above, the upper left-hand cell is B2.
- Enter the names of the series that you wish to read into the edit box. EViews reads spreadsheet data in contiguous blocks, so you should provide a name for each column or row (depending on the orientation of the data), even if you only wish to read selected rows.
- Alternatively, if the names that you wish to use for your series are contained in the file, you can simply provide the number of series to be read. The names must be adjacent to your data. If the data are organized by row and the starting cell is B2, then the names must be in column A, beginning at cell A2. If the data are organized by column beginning in B2, then the names must be in row 1, starting in cell B1. If, in the course of reading the data, EViews encounters an invalid cell name, it will automatically assign the next unused name with the prefix SER, followed by a number (e.g., SER01, SER02, etc.).
- Lastly, you should tell EViews the sample of data that you wish to import. EViews begins with the first observation in the file and assigns it to the first date in the sample for each variable. Each successive observation in the file is associated with successive observations in the sample. Thus, in an annual workfile, if you enter the sample:

1971 1975 1990 1991

in the import dialog, the first 5 observations will be assigned to the dates 1971–1975, and the sixth and seventh observations will be assigned to the dates 1990–1991. The data in the intermediate period will be unaffected by the importing procedure.

You should be warned that if you read into a sample which has more observations than are present in your input file, observations for which there are no corresponding inputs will be assigned missing values. For example, if you read into the sample defined as “1971 1990”, and there are only 10 observations in the input file, the observations from 1981 to 1990 will be assigned missing values.

When the dialog is first displayed, EViews by default enters the current workfile sample in the edit box. You should edit this string to reflect the desired sample. To make setting the sample easier, EViews provides you with three push-buttons which change the string in the edit box to commonly used values:

1. **Current sample** sets the dialog string to the current workfile sample.
 2. **Workfile range** sets the dialog string to the entire range of the workfile.
 3. **To end of range** sets the dialog string to all observations from the beginning of the current sample to the end of the workfile range.
- If you are reading data from an Excel 5 workbook file, there will be an additional edit box where you can enter the name of the sheet containing your data. If you do not enter a name, EViews will read the topmost sheet in the Excel workbook.
 - When the dialog is completely filled out, simply click **OK** and EViews will read your file, creating series and assigning values as requested.

ASCII Import

If you choose to read from an ASCII file, EViews will open an ASCII Text Import dialog. Fill out the dialog to read from the specified file.

The dialog box for ASCII file import is considerably more complicated than the corresponding spreadsheet dialog. While unfortunate, this complexity is necessary since there is no standard format for ASCII files. EViews provides you with a range of options to handle various types of ASCII files.

ASCII file importing is explained in considerable detail in [“Addendum: Reading ASCII Files”](#) beginning on page 76.

Exporting Data

EViews provides you with a number of methods for getting data from EViews into other applications.

Copying and Pasting

You can click and drag in a spreadsheet view or table of statistical results to highlight the cells you want to copy. Then click **Edit/Copy...** to put the data into the clipboard. You will see a dialog box asking whether to copy the numbers with the precision showing on your screen (formatted copy) or to copy the numbers at full precision (unformatted copy).

As a shortcut, you can highlight entire rows or columns of cells by clicking on the gray border that surrounds the spreadsheet. Dragging across the border selects multiple rows or columns. To copy several adjacent series from the spreadsheet, drag across their names in the top border. All of their data will be highlighted. Then click **Edit/Copy...** to put the data into the clipboard.

Once the data are on the clipboard, switch to the target application, and select **Edit/Paste**.

Exporting to a Spreadsheet or Text File

First, click on **Procs/Export/Write Text-Lotus-Excel...**, then enter the name and type of the output file in the file dialog. As you fill out the file dialog, keep in mind the following behavior:

- If you enter a file name with an extension, EViews will use the file extension to identify the file type. Files with common spreadsheet extensions (.XLS, .WK3, .WK1, and .WKS) will be saved to the appropriate spreadsheet type. All others will be saved as ASCII files.
- If you do not enter an extension, EViews will use the file type selected in the combo-box to determine the output type. Spreadsheet files will have the appropriate extensions appended to the name. ASCII files will be saved in the name provided in the dialog, without an extension. EViews will not append extensions to ASCII files unless you explicitly include one in the file name.
- Note that EViews cannot, at present, write into an existing file. The file that you select will, if necessary, be replaced.

Once you have specified the output file, click **OK** to open the export dialog.

Tip: if you highlight the series you wish to export before beginning the export procedure, the series names will be used to fill out the export dialog.

Spreadsheet Export

The dialogs for spreadsheet export are virtually identical to the dialogs for spreadsheet import. You should determine the orientation of your data, the series to export, and the sample of observations to be written.

Additionally, EViews provides you with checkboxes for determining whether to include the series names and/or the series dates in the spreadsheet. If you choose to write one or both to the spreadsheet, make certain that the starting cell for your data leaves the necessary room along the borders for the information. If the necessary room is not available, EViews will ignore the option—for example, if you choose to write your data beginning in cell A1, EViews will not write the names or dates.

ASCII Export

The ASCII export dialog is quite similar to the spreadsheet export dialog, but it contains a few additional options:

- You can change the text string to be used for writing missing values. Simply enter the text string in the edit field.
- EViews provides you with the option of separating data values with a tab, a space, or a comma. Click on the desired radio button.

You should be warned that if you attempt to write your data by series, EViews will write all of the observations for a series on a single line. If you have a reasonably long series of observations, these data may overflow the line-length of other programs

Frequency Conversion

As well as importing and exporting data in different formats, you will sometimes wish to combine data observed at different frequencies into a single common frequency. For example, you may have a series for which you have monthly data that you would like to use together with another series for which you have quarterly data. To use these series together they will first need to be converted to a common frequency.

Every series in EViews has an associated frequency. When a series is in a workfile, then the series is stored at the frequency of the workfile which contains the series. When a series is in a database, then each series is stored at its own frequency. Because all series in the same workfile must share a common frequency, moving a series from one workfile to another or from a database to a workfile will cause the series being moved to be converted to the frequency of the workfile into which it is being placed.

There are two types of frequency conversion: high frequency to low frequency conversion, and low frequency to high frequency conversion.

High Frequency To Low Frequency Conversion

If the series being imported has a higher frequency than the workfile, you may choose between a number of different conversion methods. There are six main alternatives:

- Average observations

- Sum observations
- First observation
- Last observation
- Maximum observation
- Minimum observation

In addition, there is an option **Conversion propagates NAs** which determines how missing data should be handled when carrying out the calculations. If **Conversion propagates NAs** is checked and a missing value appears anywhere in a calculation, the result for that period will be an NA. If **Conversion propagates NAs** is not checked, the calculations will be performed ignoring the missing values (although if all values for the period are missing, the result for that period will still be an NA).

For example, if you choose the **First observation** conversion method and fetch a daily series into a quarterly workfile, the value of the first quarter will be taken from the January 1 observation of the corresponding year. If no data exists for January 1, and **Conversion propagates NAs** is selected, an NA will be returned for the entire quarter. If **Conversion propagates NAs** is not selected, the quarter will be assigned the first available observation between January 1 and March 31.

If you fetch or copy an undated series into a dated workfile, the data will be copied into the workfile sequentially without any conversion, beginning at the starting observation number of the undated series (generally the first observation).

You can adjust the conversion method in three ways: as an option to the command line, as a series setting, or as a global option. The settings are applied in the following order:

- If you are using either the `copy` or `fetch` command and you provide an option to set the conversion method (see [copy \(p. 168\)](#) and [fetch \(p. 205\)](#) in the *Command and Programming Reference* for details), then EViews will use this method for all of the series listed in the command.
- Otherwise, if a series has a particular frequency conversion option set, choose **View/Conversion Options...** from the series object menu to view or change the option) that conversion method will be used for that particular series.
- Otherwise, the global option setting will be used (choose **Options/Frequency Conversion - Dates...** from the main menu to view or change the option). The global option is initially set to average the observations and to not propagate NAs.

As an example of controlling frequency conversion, say you have daily data consisting of high, low, and close series for a particular stock, from which you would like to construct a monthly workfile. If you use the default frequency conversion methods, the monthly work-

file will contain series which are an average of the daily observations. This is unlikely to be what you want. By setting the frequency conversion method of the high series to **Maximum observation**, of the low series to **Minimum observation**, and of the close series to **Last observation**, you can create a monthly workfile which will correctly follow the definitions of the series.

Low Frequency to High Frequency

EViews also provides a number of different interpolation methods for dealing with the case where the series being brought into the workfile has a lower frequency than the workfile. Since observing a series at a lower frequency provides fundamentally less information than observing the same series at a higher frequency, it is generally not possible to recover the high frequency series from the low frequency data. Consequently, the results from EViews' interpolation methods should be considered as suggestive rather than as providing the true values of the underlying series.

EViews supports the following interpolation methods:

- **Constant:** Constant with sum or average matched to the source data.
- **Quadratic:** Local quadratic with sum or average matched to the source data.
- **Linear:** Linear with last observation matched to the source data.
- **Cubic:** Cubic spline with last observation matched to the source data.
- **No conversion:** Disallow conversion (generates error message if mixed frequency).

Using an interpolation method which matches the average means that the average of the interpolated points for each period is equal to the source data point for that period. Similarly if the sum is matched, the interpolated points will sum to the source data point for the period, and if the last observation is matched, the last interpolated point will equal the source data point for the period.

For all methods, all relevant data from the low frequency series is used when forming the high frequency series, even if the range of the destination covers only part of the source.

The following describes the different methods in greater detail:

- **Constant: match average, Constant: match sum**—These two methods assign the same value to all observations in the high frequency series associated with a particular low frequency period. In one case, the value is chosen so that the average of the high frequency observation matches the low frequency observation (the value is simply repeated). In the other case, the value is chosen so that the sum of the high frequency observations matches the low frequency observation (the value is divided by the number of observations).

- **Quadratic: match average, Quadratic: match sum**—These two methods fit a local quadratic polynomial for each observation of the low frequency series, then use this polynomial to fill in all observations of the high frequency series associated with the period. The quadratic polynomial is formed by taking sets of three adjacent points from the source series and fitting a quadratic so that either the average or the sum of the high frequency points match to the low frequency data actually observed. For most points, one point before and one point after the period currently being interpolated are used to provide the three points. For end points, the two periods are both taken from the one side where data is available.

This method is a purely local method. The resulting interpolation curves are not constrained to be continuous at the boundaries between adjacent periods. Because of this, the method is better suited to situations where relatively few data points are being interpolated and the source data is fairly smooth.

- **Linear: match last**—This method assigns each value in the low frequency series to the last high frequency observation associated with the low frequency period, then places all intermediate points on straight lines connecting these points.
- **Cubic: match last**—This method assigns each value in the low frequency series to the last high frequency observation associated with the low frequency period, then places all intermediate points on a natural cubic spline connecting all the points.

A natural cubic spline is defined by the following properties:

1. Each segment of the curve is represented by a cubic polynomial.
2. Adjacent segments of the curve have the same level, first derivative and second derivative at the point where they meet.
3. The second derivative of the curve at the two global end points is equal to zero (this is the “natural” spline condition).

Cubic spline interpolation is a global interpolation method so that changing any one point (or adding an additional point) to the source series will affect all points in the interpolated series.

Commands for Basic Data Handling

To create a new series from an existing series, follow the `series` or `genr` command with a name for the new series, an equal sign and an expression involving the existing series:

```
series logy = log(y)
```

creates a new series named LOGY that is the natural log of the series Y.

To create a new group of series, follow the `group` command with a name for the group and a list of series to include in the group, each separated by a space:

```
group rhs c x1 x2 z
```

creates a group named RHS that contains the constant C (a series of ones) and the series X1, X2, Z.

To view the series or group, follow the `show` command with the name of the series or group:

```
show logy
```

To bring up the import dialog box, follow the `read` command with the full name of the file (including the file extension) to import from:

```
read c:\data\cps88.dat
```

To bring up the export dialog box, follow the `export` command with the full name of the file (including the file extension) to export:

```
write a:\usmacro.dat
```

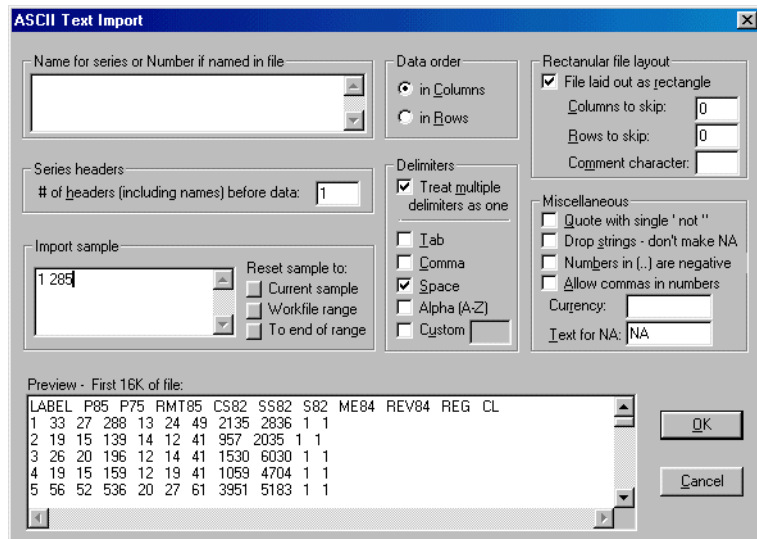
See the *Command and Programming Reference* for a complete list of commands and options available in EViews.

Addendum: Reading ASCII Files

If you instruct EViews to read from an ASCII file, the ASCII Text Import dialog will be displayed.

You may notice that the dialog is more complicated than the corresponding spreadsheet dialog. Since there is no standard format for ASCII files, we need to provide a variety of options to handle various types of files.

Note that the preview win-



dow at the bottom of the dialog shows you the first 16K of your file. You can use this information to set the various formatting options in the dialog.

You must provide the following information:

- **Names for series or Number of series if names in file.** If the file does not contain series names, or if you do not want to use the names in the file, list the names of the series in the order they appear in the file, separated by spaces.

If the names of the series are located in the file *before the start of the data*, you can tell EViews to use these names by entering a number representing the number of series to be read.

If possible, you should avoid using parentheses and mathematical symbols such as “*”, “+”, “-”, “/”, “^” in the names. If EViews tries to read the names from the file and encounters an invalid name, it will try to rename the series to a valid name by replacing invalid characters with underscores and numbers. For example, if the series is named X(-3) in the file, EViews will rename this series to X__3_01. If X__3_01 is already a series name, then EViews will name the series X__3_02, and so forth.

If EViews cannot name your series, say, because the name is a reserved name, or because the name is used by an object that is not a series, the series will be named SER01, SER02, etc.

You should be very careful in naming your series and listing the names in the dialog. If the name in the list or in the file is the same as an existing series name in the workfile, the data in the existing series will be overwritten.

- **Data order.** You need to specify how the data are organized in your file. If your data are ordered by observation so that each series is in a column, select **in Columns**. If your data are ordered by series so that all the data for the first series are followed by all the data for the second series and so on, select **in Rows**.
- **Import sample.** You should specify the sample in which to place the data from the file. EViews fills out the dialog with the current workfile sample, but you can edit the sample string or use the sample reset buttons to change the input sample. The input sample only sets the sample for the import procedure, it does not alter the workfile sample.

EViews fills all of the observations in the current sample using the data in the input file. There are a couple of rules to keep in mind:

1. EViews assigns values to all observations in the input sample. Observations outside of the input sample will not be changed.
2. If there are too few values in the input file, EViews will assign NAs to the extra

observations in the sample.

3. Once all of the data for the sample have been read, the remainder of the input file will be ignored.

In addition to the above information, you can use the following options to further control the way EViews reads in ASCII data.

EViews scans the first few lines of the source file and sets the default formatting options in the dialog based on what it finds. However, these settings are based on a limited number of lines and may not be appropriate. You may find that you need to reset these options.

Delimiters

Delimiters are the characters that your file uses to separate observations. You can specify multiple delimiters by selecting the appropriate entries. **Tab**, **Comma**, and **Space** are self-explanatory. The **Alpha** option treats any of the 26 characters from the alphabet as a delimiter.

For delimiters not listed in the option list, you can select the **Custom** option and specify the symbols you wish to treat as delimiters. For example, you can treat the slash “/” as a delimiter by selecting **Custom** and entering the character in the edit box. If you enter more than one character, *each* character will be treated as a delimiter. For example, if you type “//” in the **Custom** field, then the single slash “/” will be treated as a delimiter, instead of the double slash “//”. The double slash will be interpreted as two delimiters.

EViews provides you with the option of treating multiple delimiter characters as a single delimiter. For example, if “,” is a delimiter and the option **Treat multiple delimiters as one** is selected, EViews will interpret “,,” as a single delimiter. If the option is turned off, EViews will view this string as two delimiters surrounding a missing value.

Rectangular File Layout Options

To treat the ASCII file as a rectangular file, select the **File laid out as rectangle** option in the upper right-hand portion of the dialog. If the file is rectangular, EViews reads the file as a set of *lines*, with each new line denoting a new observation or a new series. If you turn off the rectangular option, EViews treats the whole file as one long string separated by delimiters and carriage returns.

Knowing that a file is rectangular makes ASCII reading much simpler since EViews knows how many values to expect on a given line. For files that are not rectangular, you will need to be precise about the number of series or observations that are in your file. For example, suppose that you have a non-rectangular file that is ordered in columns and you tell EViews that there are four series in the file. EViews will ignore new lines and will read a new observation after reading every four values.

If the file is rectangular, you can tell EViews to skip columns and/or rows.

For example, if you have a rectangular file and you type 3 in the **Rows to skip** field, EViews will skip the first three rows of the data file. Note that you can only skip the *first* few rows or columns; you cannot skip rows or columns in the middle of the file.

Series Headers

This option tells EViews how many “cells” to offset as series name headers before reading the data in the file. The way that cell offsets are counted differs depending on whether the file is in rectangular form or not.

For files in rectangular form, the offsets are given by rows (for data in columns) or by columns (for data in rows). For example, suppose your data file looks as follows:

LABEL	P85	P75	RMT85	CS82	SS82	S82	ME84	REU84	REG	CL
1	33	27	288	13	24	49	2135	2836	1	1
2	19	15	139	14	12	41	957	2035	1	1
3	26	20	196	12	14	41	1530	6030	1	1

There is a one line (row) gap between the series name line and the data for the first observation. In this case, you should set the series header offset as 2, one for the series name line and one for the gap. If there were no gap, then the correct offset would instead be 1.

For files that are in non-rectangular form, the offsets are given by the number of cells separated by the delimiters. For example, suppose you have a data file that looks as follows:

FTP	UEMP	MAN	LIC	GR	CLEAR	WM	NMAN	GOV	HE
WE	HOM	ACC	ASR						
260.35	11.0	455.5	178.15	215.98	93.4	558724.	538.1	133.9	2.98
117.18	8.60	39.17	306.18						
269.80	7.0	480.2	156.41	180.48	88.5	538584.	547.6	137.6	3.09
134.02	8.90	40.27	315.16						

The data are ordered in columns but each observation is recorded in two lines, the first line for the first 10 series and the second line for the remaining 4 series.

It is instructive to examine what happens if you incorrectly read this file as a rectangular file with 14 series and a header offset of 2. EViews will look for the series names in the first line, will skip the second line, and will begin reading data starting with the third line, treating each line as one observation. The first 10 series names will be read correctly, but since EViews will be unable to find the remaining four names on the first line, the remaining series will be named SER01–SER04. The data will also be read incorrectly. For example, the first four observations for the series GR will be 215.9800, NA, 180.4800, and NA, since EViews treats each line as a new observation.

To read this data file properly, you should turn off the rectangle file option and set the header offset to 1. Then EViews will read, from left to right, the first 14 values that are separated by a delimiter or carriage return and take them as series names. This corresponds to

the offset of 1 for headers. The next 14 observations are the first observations of the 14 series, and so on.

Miscellaneous Options

- **Quote with single ‘ not ‘.** The default behavior in EViews is to treat anything inside a pair of matching double quotes as one string, unless it is a number. This option treats anything inside a pair of matching single quotes as one string, instead of the double quotes. Since EViews does not support strings, the occurrence of a pair of matching double quotes will be treated as missing, unless the thing inside the pair of double quotes is a number.
- **Drop strings—don’t make NA.** Since EViews does not support strings, any input into a series observation that is not a number or delimiter will, by default, be treated as a missing observation. For example, 10b and 90:4 will both be treated as missing values (unless Alphabetic characters or “:” are treated as delimiters). The **Drop strings** option will skip these strings instead of treating them as NAs.

If you choose this option, the series names, which are strings, will also be skipped so that your series will be named using the EViews default names: SER01, SER02, and so on. If you wish to name your series, you should list the series names in the dialog.

Note that strings that are specified as missing observations in the **Text for NA** edit box will not be skipped and will be properly indicated as missing.

- **Numbers in () are negative.** By default, EViews treats parentheses as strings. However, if you choose this option, numbers in parentheses will be treated as negative numbers and will be read accordingly.
- **Allow commas in numbers.** By default, commas are treated as strings unless you specify them as a delimiter. For example, 1,000 will be read as either NA (unless you choose the drop string option, in which case it will be skipped) or as two observations, 1 and 0 (if the comma is a delimiter). However, if you choose to **Allow commas in numbers**, 1,000 will be read as the number 1000.
- **Currency.** This option allows you to specify a symbol for currency. For example, the default behavior treats \$10 as a string (which will either be NA or skipped) unless you specify “\$” as a delimiter. If you enter “\$” in the **Currency** option field, then \$10 will be read as the number 10.

The currency symbol can appear either at the beginning or end of a number but not in the middle. If you type more than one symbol in the field, each symbol will be treated as a currency code. Note that currency symbols are case sensitive. For example, if the Japanese yen is denoted by the “Y” prefix, you should enter “Y”, not “y”.

- **Text for NA.** This option allows you to specify a code for missing observations. The default is NA. You can use this option to read data files that use special values to indicate missing values, e.g., “.”, or “-99”.

You can specify only one code for missing observations. The entire **Text for NA** string will be treated as the missing value code.

Examples

In these examples, we demonstrate the ASCII import options using example data files downloaded from the World Wide Web. The first example file looks as follows:

X	Y	Z	A	B	AA	BB
5	3	10	-0.6008	NBA	10	8
10	3	7	-0.4837	NFL	7	2
2	2	5	1.2467	NFL	0	0
12	0	3	0.1705	NBA	5	1

This is a cross-section data set, seven series ordered in columns, each separated by a single space. Note that the B series takes string values, which will be replaced by NAs. If we type 7 series in the number of series field and use the default setting, EViews will correctly read in the data.

By default, EViews checks the **Treat multiple delimiters as one** option even though the series are delimited by a single space. If you do not check this option, the last series BB will not be read in and EViews creates a series named SER01. This strange behavior is caused by an extra space in the very first column of the data file, before the 1st and 3rd observations of the X series. EViews treats the very first space as a delimiter and looks for the first series data before the first extra space, which is missing. Therefore the first series is named SER01 with data NA, 10, NA, 12 and all other series are incorrectly imported.

To handle this case, EViews automatically ignores the delimiter before the first column data if you choose both the **Treat multiple delimiters as one** and the **File laid out as rectangle** options.

The top of the second example file looks like:

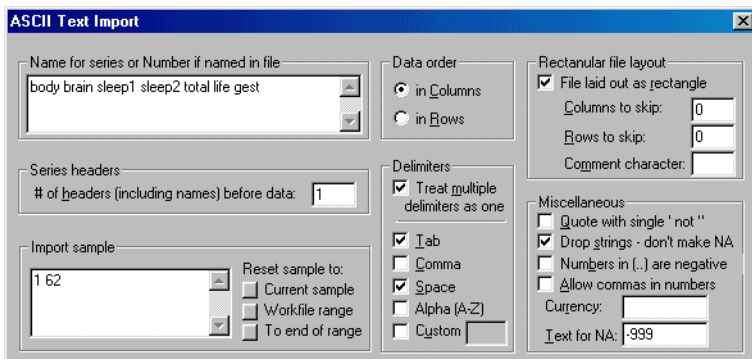
African elephant	6654.000	5712.000	-999.0	-999.0	3.3	38.6	645.0
African giant pouched rat	1.000	6.600	6.3	2.0	8.3	4.5	42.0
Arctic Fox	3.385	44.500	-999.0	-999.0	12.5	14.0	60.0
Arctic ground squirrel	.920	5.700	-999.0	-999.0	16.5	-999.0	25.0
Asian elephant	2547.000	4603.000	2.1	1.8	3.9	69.0	624.0
Baboon	10.550	179.500	9.1	.7	9.8	27.0	180.0

This is a cross-section data set, ordered in columns, with missing values coded as “-999.0”. There are eight series, each separated by spaces. The first series is the ID name in strings.

If we use the EViews defaults, there will be problems reading this file. The spaces in the ID description will generate spurious NA values in each row, breaking the rectangular format

of the file. For example, the first name will generate two NAs, since “African” is treated as one string, and “elephant” as another string.

You will need to use the **Drop strings** option to skip all of the strings in your data so that you don’t generate NAs. Fill out the ASCII dialog as follows:



Note the following:

- Since we skip the first string series, we list only the remaining seven series names.
- There are no header lines in the file so we set the offset to 0.
- If you are not sure whether the delimiter is a space or tab, mark both options. You should treat multiple delimiters as one.
- **Text for NA** should be entered exactly as it appears in the file. For this example you should enter “-999.0”, not “-999”.

The third example is a daily data file that looks as follows:

```

"Daily Corporate Bond Yields"
"Jan. 1, 1989 to Dec. 31, 1993"

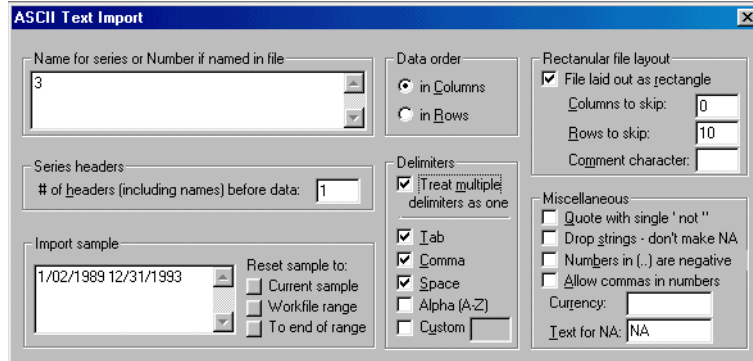
"Source: Federal Reserve Statistical Release H.15"
"Board of Governors of the Federal Reserve System"

"DATE = MM/DD/YY FORMAT"
"AA = MOODYS SEASONED AA"
"BAA = MOODYS SEASONED BAA"

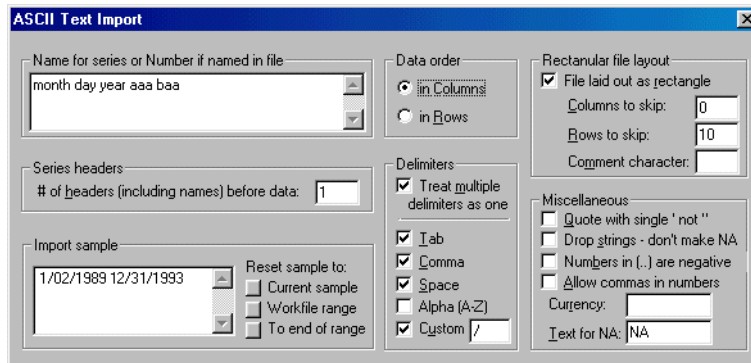
"DATE", "AA", "BAA"
01/02/89 0 0
01/03/89 9.88 10.71
01/04/89 9.86 10.71
01/05/89 9.89 10.75

```

This file has 10 lines of data description, line 11 is the series name header, and the data begin in line 12. The data are ordered in columns in rectangular form with missing values coded as a “0”. To read these data, you can instruct EViews to skip the first 10 rows of the rectangular file, and read three series with the names in the file.



The only problem with this method is that the DATE series will be filled with NAs since EViews treats the entry as a string (because of the “/” in the date entry). You can avoid this problem by identifying the slash as a delimiter using the **Custom** edit box. The first column will now be read as three distinct series since the two slashes are treated as delimiters. Therefore, we modify the option settings as follows:



Note the changes to the dialog entries:

- We now list five series names. We cannot use the file header since the line only contains three names.
- We skip 11 rows with no header offset since we want to skip the name header line.
- We specify the slash “/” as an additional delimiter in the **Custom** option field.

The month, day, and year will be read as separate series and can be used as a quick check of whether the data have been correctly read.

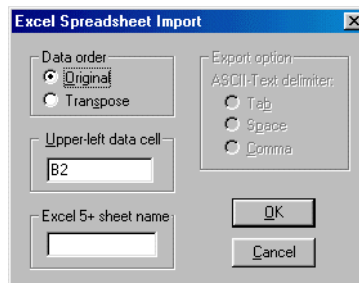
Addendum: Matrix Object Reading and Writing

The two available procedures for matrices allow you to import and export data directly to or from the matrix object. As with the standard EViews procedures, you can read or write from spreadsheet or from ASCII files.

Import Data (ASCII,XLS,WK?)...
Export Data (ASCII,XLS,WK?)...

To read from a file, select **Procs/Import Data (ASCII, .XLS, .WK?)...** EViews will open an import dialog.

Here, we depict the matrix import Excel dialog. As above, the corresponding ASCII dialog has many more options, since ASCII file reading is more complicated. Note that both the import and export dialogs differ little from the series import dialogs described above. The differences reflect the different nature of series and matrix input and output. For example, dialog options for series names and the sample are omitted since they do not apply to matrices.



To write the contents of the matrix to a file, select **Procs/Export Data (ASCII, .XLS, .WK?)...** and fill in the dialog.

In reading from a file, EViews first fills the matrix with NAs, puts the first data element in the (1,1) element of the matrix, and then continues reading the data by row or column according to the specified settings for **Ordering of data**. If this option is set as **Original**, EViews will read by row, filling the first row from left to right, and then continuing on to the next row. If the ordering is set as **Transpose**, EViews fills the matrix by columns, reading the first column from top to bottom and then continuing on to the next column.

ASCII files provide you with the option of reading your file as a rectangle. If your ASCII file is laid out as a rectangle, the contents of the rectangle will be placed in the matrix beginning at the (1,1) element of the matrix. For example, if you have a 3×3 matrix X, and read from the ASCII file containing:

```
1 2 3 4
5 6 7 8
9 10 11 12
```

using the **File laid out as rectangle** option, the matrix X will contain the corresponding rectangular portion of the ASCII file:

```
1 2 3
5 6 7
```

9 10 11

If you do not select the rectangular read option, EViews fills the matrix element-by-element, reading from the file line by line. Then X will contain:

1 2 3

4 5 6

7 8 9

Chapter 5. Working with Data

In the following discussion we describe EViews' powerful language for using expressions and generating and manipulating the data in series and groups. We first describe the fundamental rules for working with mathematical expressions in EViews, and then describe how to use these expressions in working with series and group data.

Using Expressions

One of the most powerful features of EViews is the ability to use and to process mathematical expressions. EViews contains an extensive library of built-in operators and functions that allow you to perform complicated mathematical operations on your data with just a few keystrokes. In addition to supporting standard mathematical and statistical operations, EViews provides a number of specialized functions for automatically handling the leads, lags and differences that are commonly found in time series data.

An EViews expression is a combination of numbers, series names, functions, and mathematical and relational operators. In practical terms, you will use expressions to describe all mathematical operations involving EViews objects.

As in other programs, you can use these expressions to calculate a new series from existing series, to describe a sample of observations, or to describe an equation for estimation or forecasting. However, EViews goes far beyond this simple use of expressions by allowing you to use expressions virtually anywhere you would use a series. We will have more on this important feature shortly, but first, we describe the basics of using expressions.

Operators

EViews expressions may include operators for the usual arithmetic operations. The operators for addition (+), subtraction (-), multiplication (*), division (/) and raising to a power (^) are used in standard fashion so that

$$5 + 6 * 7.0 / 3$$

$$7 + 3e-2 / 10.2345 + 6 * 10^2 + 3e3$$

$$3^2 - 9$$

are all valid expressions. Notice that explicit numerical values may be written in integer, decimal, or scientific notation.

In the examples above, the first expression takes 5 and adds to it the product of 6 and 7.0 divided by 3 ($5 + 14 = 19$); the last expression takes 3 raised to the power 2 and subtracts 9 ($9 - 9 = 0$). These expressions use the order of evaluation outlined below.

The “-” and “+” operators are also used as the unary minus (negation) and unary plus operators. It follows that

```
2-2
-2+2
2+++++++++--2
2---2
```

all yield a value of 0.

EViews follows the usual order in evaluating expressions from left to right, with operator precedence order (from highest precedence to lowest):

- unary minus (-), unary plus (+)
- exponentiation (^)
- multiplication (*), division (/)
- addition (+), subtraction (-)
- comparison (<, >, <=, >=, =)
- and, or

The last two sets of operators are used in logical expressions.

To enforce a particular order of evaluation, you can use parentheses. As in standard mathematical analysis, terms which are enclosed in parentheses are treated as a subexpression and evaluated first, from the innermost to the outermost set of parentheses. We strongly recommend the use of parentheses when there is any possibility of ambiguity in your expression.

To take some simple examples,

- -1^2 , evaluates to $(-1)^2 = 1$ since the unary minus is evaluated prior to the power operator.
- $-1 + -2 * 3 + 4$, evaluates to $-1 + -6 + 4 = -3$. The unary minus is evaluated first, followed by the multiplication, and finally the addition.
- $(-1 + -2) * (3 + 4)$, evaluates to $-3 * 7 = -21$. The unary minuses are evaluated first, followed by the two additions, and then the multiplication.
- $3 * ((2+3) * (7+4) + 3)$, evaluates to $3 * (5 * 11 + 3) = 3 * 58 = 174$.

A full listing of operators is presented in [Appendix A, “Operator and Function Reference”](#), on page 435 of the *Command and Programming Reference*.

Series Expressions

Much of the power of EViews comes from the fact that expressions involving series operate on every observation, or element, of the series in the current sample. For example, the series expression

$$2*y + 3$$

tells EViews to multiply every sample value of Y by 2 and then to add 3. We can also perform operations that work with multiple series. For example,

$$x/y + z$$

indicates that we wish to take every observation for X and divide it by the corresponding observation on Y, and add the corresponding observation for Z.

Series Functions

EViews contains an extensive library of built-in functions that operate on all of the elements of a series in the current sample. Some of the functions are “element functions” which return a value for each element of the series, while others are “summary functions” which return scalars, vectors or matrices, which may then be used in constructing new series or working in the matrix language (see [Chapter 4, “Matrix Language”, on page 55](#) of the *Command and Programming Reference* for a discussion of scalar, vector and matrix operations).

Most function names in EViews are preceded by the @-sign. For example, @mean returns the average value of a series taken over the current sample, and @abs takes the absolute value of each observation in the current sample.

All element functions return NAs when any input value is missing or invalid, or if the result is undefined. Functions which return summary information generally exclude observations for which data in the current sample are missing. For example, the @mean function will compute the mean for those observations in the sample that are non-missing.

A full description of the functions is presented in [Appendix A, “Operator and Function Reference”, on page 435](#) of the *Command and Programming Reference*.

Examples

We provide a few additional examples without additional comment:

```
@trend(1980:1)
@movav(x,4)
@mean(x)*y + @var(x)*z
1/@sqrt(2*@acos(-1))*exp(-1/2 * x^2)
```

```
d(x, 4)
```

The remainder of this chapter will provide additional examples of expressions involving functions.

Series Elements

At times, you may wish to access a particular observation for a series. EViews provides you with a special function, `@elem`, which allows you to use a specific value of a series.

`@elem` takes two arguments: the first argument is the name of the series, and the second is the date or observation identifier.

For example, suppose that you want to use the 1980:3 value of the quarterly series Y, or observation 323 of the undated series X. Then the functions:

```
@elem(y, 1980:3)
@elem(x, 323)
```

will return the values of the respective series in the respective periods.

Logical Expressions

A logical expression is one which evaluates to “true” or “false”. EViews also allows logical expressions to take the value “missing”, but for the moment, we will ignore this point until our discussion of missing values (see [“Missing Data” on page 92](#)).

Logical expressions may be used as part of a mathematical operation, as part of a sample statement, or as part of an if-condition in programs.

We have already seen examples of logical expressions in our discussion of samples and sample objects. For example, we saw the sample condition:

```
incm > 5000
```

which allowed us to select observations meeting the specified condition. This is an example of a logical expression—it is true for each observation on INCM that exceeds 5000; otherwise, it is false.

More generally, logical expressions are those involving the comparison operators, “<” (less than), “>” (greater than), “< =” (less than or equal to), “> =” (greater than or equal to), “=” (equal to), “< >” (not equal to).

As described above in the discussion of samples, you can use the “and” and “or” conjunction operators to build more complicated logical expressions:

```
(incm>5000 and educ>=13) or (incm>10000)
```

It is worth noting that EViews uses the number 1 to represent true and 0 to represent false. This internal representation means that you can create complicated expressions involving logical subexpressions. For example, you can use logical expressions to recode your data:

```
0*(inc<100)+(inc>=100 and inc<200)+2*(inc>=200)
```

which yields 0 if INC < 100, 1 if INC is greater than or equal to 100 and less than 200, and 2 for INC greater than or equal to 200.

The equality comparison operator “=” requires a bit more discussion, since the equal sign is used both in assigning values and in comparing values. We consider this issue in a bit more depth when we discuss creating and modifying series (see [“Working with Series” on page 94](#)). For now, note that if used in an expression,

```
incm = 2000
```

evaluates to true if income is exactly 2000, and false, otherwise.

Leads, Lags, and Differences

It is easy to work with lags or leads of your series. Simply use the series name, followed by the lag or lead enclosed in parentheses. Lags are specified as negative numbers and leads as positive numbers so that

```
income(-4)
```

is the fourth lag of the income series, while

```
sales(2)
```

is the second lead of sales.

While EViews expects lead and lag arguments to be integers, there is nothing to stop you from putting non-integer values in the parentheses. EViews will automatically convert the number to an integer; you should be warned, however, that the conversion behavior is not guaranteed to be systematic. If you must use non-integer values, you are strongly encouraged to use the `@round`, `@floor`, or `@ceil` functions to control the lag behavior.

In many places in EViews, you can specify a range of lead or lag terms. For example, when estimating equations, you can include expressions of the form

```
income(-1 to -4)
```

to represent all of the INCOME lags from 1 to 4. Similarly, the expressions

```
sales sales(-1) sales(-2) sales(-3) sales(-4)
sales(0 to -4)
```

```
sales(to -4)
```

are equivalent methods of specifying the level of SALES and all lags from 1 to 4.

EViews also has several built-in functions for working with difference data in either levels or in logs. The “D” and “DLOG” functions will automatically evaluate the differences for you. For example, instead of taking differences explicitly,

```
income - income(-1)
log(income) - log(income(-1))
```

you can use the equivalent expressions,

```
d(income)
dlog(income)
```

You can take higher order differences by specifying the difference order. For example, the expressions

```
d(income,4)
dlog(income,4)
```

represent the fourth-order differences of INCOME and log(INCOME).

If you wish to take seasonal differences, you should specify both the ordinary, and a seasonal difference term:

```
d(income,1,4)
dlog(income,1,4)
```

are first order differences with a seasonal difference at lag 4. If you want only the seasonal difference, specify the ordinary difference term to be 0:

```
d(income,0,4)
dlog(income,0,4)
```

Mathematical details are provided in [Appendix A, “Operator and Function Reference”](#), on [page 435](#) of the *Command and Programming Reference*.

Missing Data

Occasionally, you will encounter data that are not available for some periods or observations, or you may attempt to perform mathematical operations where the results are undefined (e.g., division by zero, log of a negative number). EViews uses the code NA (not available) to represent these missing values.

For the most part, you will never have to worry about NAs. EViews will generate NAs for you when appropriate, and will automatically exclude observations with NAs from statistical calculations. For example, if you are estimating an equation, EViews will use the set of

observations in the sample that have no missing values for the dependent and all of the independent variables.

There are, however, a few cases where you will need to work with NAs so you should be aware of some issues.

When you perform operations using multiple series, there may be alternative approaches for handling NAs. EViews will usually provide you with the option of *casewise exclusion* (common sample) or *listwise exclusion* (individual sample). With casewise exclusion, only those observations for which *all* of the series have non-missing data are used. This rule is always used, for example, in equation estimation. For listwise exclusion, EViews will use the maximum number of observations possible for each series, excluding observations separately for each series in the list of series. For example, when computing descriptive statistics for a group of series, you have the option to use a different sample for each series.

If you must work directly with NAs, just keep in mind that EViews NAs observe all of the rules of IEEE NaNs. This means that performing mathematical operations on NAs will generate missing values. Thus, each of the following expressions will generate missing values:

```
@log(-abs(x))
1/(x-x)
(-abs(x))^(1/3)
3*x + na
exp(x*na)
```

The equality and inequality comparison operators will work as expected with NA values treated as if they were any other value. Thus, the expressions

```
x = 5
y <> 5
x = na
y <> na
```

will all evaluate to “true” or “false” depending upon the value of each observation of the series.

In contrast, inequality comparisons using NAs are not logically defined, and will always return NA values:

```
x >= na
y < na
```

will yield NAs for all observations, irrespective of the values of X and Y. Similarly,

```
x > 5
```

will return a “true” or “false” for non-NA values of X, and NAs where X is missing.

If used in a mathematical operation, a logical expression resulting in an NA is treated as an ordinary missing value. For example, for observations where the series X contains NAs, the mathematical expression

```
5 * (x > 3)
```

will yield NAs. However, if the logical expression is used as part of a sample or if-statement, NA values are treated as “false”.

```
smpl 1 1000 if x > 3
```

```
smpl 1 1000 if x > 3 and x <> na
```

are equivalent since the condition `x > 3` implicitly tests for NA values. One consequence of this behavior is that

```
smpl 1 1000 if x <= na
```

will result in a sample with no observations.

Versions of EViews prior to 3.x followed the same IEEE rules for missing data with one important exception. In previous versions multiplying any number by zero resulted in zero. In EViews 3 and 4, the value NA times zero equals NA. Thus a recommended method of recoding (replacing) NA’s in the series X to a number Y will no longer work:

```
x = (x <> na) * x + (x = na) * y
```

works in Version 2, but not subsequent versions. A new `@nan` function has been provided for this purpose.

```
x = @nan(x, y)
```

recodes “NA” values of X to take the values in the series Y. See [Appendix A, “Operator and Function Reference”](#), on page 435 of the *Command and Programming Reference*.

Working with Series

One of the primary uses of expressions is to generate new series from existing data or to modify the values in an existing series. Used in combination with samples, expressions allow you to perform sophisticated transformations of your data, saving the results in new or existing series objects.

To create or modify a series, select **Quick/Generate Series...** or click on the **Genr** button on the workfile toolbar. EViews opens a window prompting you for additional information.

You should enter the assignment statement in the upper edit box, and the relevant sample period in the lower edit box.

The assignment statement is actually an implicit loop over observations. Beginning with the first observation in the sample, EViews will evaluate the assignment statement for each included observation.

Basic Assignment

You can type the series name, followed by an equal sign and then an expression. For every element of the sample, EViews will evaluate the expression on the right-hand side of the equality, and assign the value to the *destination series* on the left-hand side, creating the series if necessary.

For example, if there is no series named Y,

$$y = 2*x + 37*z$$

will first create the Y series and fill it with NAs. Then, for every observation in the current sample, EViews will fill each element of the Y series with the value of the expression. If Y does exist, EViews will only replace Y values in the current sample with the value of the expression. All observations not in the sample will be unchanged.

One special form of assignment occurs when the right-hand side expression is a constant expression:

$$y = 3$$

$$y = 37 * 2 + 3$$

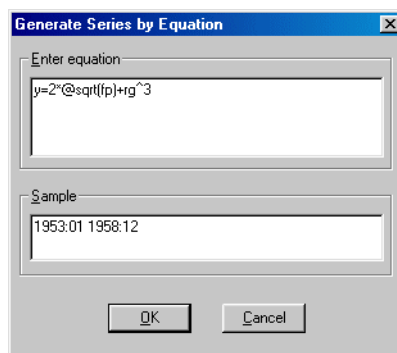
EViews will simply loop through all of the observations in the sample and assign the value of the constant.

Using Samples

By modifying the sample of observations used in assignment, you can splice together series using multiple **Genr** commands. For example, if we enter three **Genr** commands with different samples, first,

Upper window: `y = z`

Lower window: `@all if z<=1 and z>-1`



followed by a **Genr** with

```
Upper window:  y = -2 + 3*z
```

```
Lower window:  @all if z>1
```

and finally,

```
Upper window:  y = -.9 + .1*z
```

```
Lower window:  @all if z<=-1
```

we can generate Y as a piecewise linear function of the series Z.

Note that while it is possible to perform these types of operations using loops and IF-statements (see the *Command and Programming Reference*), we strongly urge you to use **Genr** and sample statements wherever possible since the latter approach is much more efficient.

Dynamic Assignment

Since EViews evaluates the assignment expression for each observation in the sample, you can perform dynamic assignment by using lagged values of the destination series on the right side of the equality. For example, suppose we have an annual workfile that ranges from 1945 to 1997. Then if we enter:

```
Upper window:  y = y + y(-1)
```

```
Lower window:  1946 1997
```

EViews will replace the Y series with the cumulative sum of Y. We begin with 1946, since we do not want to transform the first value in the workfile. Then for each period, EViews will take the current value of Y and add it to the lagged value of Y. The assignment is dynamic since as we successively move on to the next period, the lagged value of Y contains the cumulative sum.

Note that this procedure destroys the original data. To create a new series with the cumulative sums, you will have to perform the assignment in two steps, first making a copy of the original series, and then performing the dynamic assignment.

Implicit Assignment

You can make an implicit assignment by putting a simple formula on the left-hand side of the equal sign. EViews will examine your expression and select, as the destination series, the first valid series name on the left-hand side of the equality. Then for every observation in the sample, EViews will assign values using the implicit relationship. For example, if you enter:

```
log(y) = x
```

EViews will treat Y as the destination series, and evaluate $y=\exp(x)$ for every observation in the sample.

The following are examples of valid assignment statements where Y is the destination series:

```
1/y = z
log(y/x)/14.14 = z
log(@inv(y)*x) = z
2+y+3*z = 4*w
d(y) = nrnd
```

In general, EViews can solve for, or *normalize*, equations that use the following on the left-hand side of the equality: +, -, *, /, ^, log(), exp(), sqr(), d(), dlog(), @inv().

Since **Genr** is not a general equation solver, there will be situations in which EViews cannot normalize your equation. You cannot, for example, use the assignment statement

```
@tdist(y, 3) = x
```

since `@tdist` is not one of the functions that EViews knows how to invert. Similarly, EViews cannot solve for equations where the destination series appears more than once on the left side of the equality. For example, EViews cannot solve the equation,

```
x + 1/x = 5
```

In both cases, EViews will display the error message “Unable to normalize equation”.

Note that the destination series can appear on both sides of the equality. For example,

```
log(x) = x
```

is a legal assignment statement. EViews will normalize the expression and perform the assignment

```
x = exp(x)
```

so that X will be assigned the exponential of the original value of X. EViews will *not* solve for the values of X satisfying the equality $\log(x) = x$.

Command Window Assignment

You can create series and assign values from the command window. First, set the sample using the `sample` statement, then enter the assignment statement.

There are alternative forms for the assignment statement. First, if the series does not exist, you must use either the `series` or the `genr` keywords, followed by the assignment expression. The two statements

```
series y = exp(x)
genr y = exp(x)
```

are equivalent methods of generating the series Y. Once the series has been created, subsequent assignment statements do not require the keywords:

```
smpl @all
series y = exp(x)
smpl 1950 1990 if y>300
y = y/2
```

Examples

The following examples demonstrate how to perform some common tasks involving generating series (using the command line syntax):

- Time trend (with the value 0 in 1987:1)

```
series time = @trend(1987:1)
```

- Price index (with the value 1 in 1985:6)

```
series index = price/@elem(price, 1985:6)
```

- Dummy variables

```
series high_inc = income>12000
```

- Recode using logical expressions

```
series newser = x*(y>30) + z*(y<=30)
```

Working with Auto-series

Another important method of working with expressions is to use an expression *in place of* a series. EViews' powerful tools for expression handling allow you to substitute expressions virtually any place you would use a series—as a series object, as a group element, in equation specifications and estimation, and in models.

We term expressions that are used in place of series as *auto-series*, since the transformations in the expressions are *automatically* calculated without an explicit assignment statement.

Auto-series are most useful when you wish to see the behavior of a function of series, but do not want to keep the transformed series, or in cases where the underlying series data change frequently. Since the auto-series expressions are automatically recalculated whenever the underlying data change, they are never out-of-date.

"Creating" Auto-series

It is easy to create and use an auto-series—anywhere you might use a series name, simply enter an EViews expression. For example, suppose that you wish to plot the log of CP

against time for the period 1953:01 to 1958:12. One way to plot these values is to generate a new series, say LOGCP, and then plot the values. To generate the new series, enter

```
series logcp = log(cp)
```

in the command window, open the series LOGCP and select **View/Line Graph**.

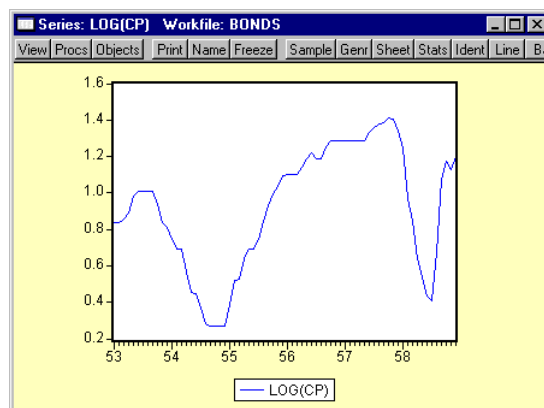
Alternatively, you can create and use an auto-series by clicking on the **Show** button, or selecting **Quick/Show...** and entering `log(cp)`. EViews will open a series window in spreadsheet view:

LOG(CP)	
Last updated: 09/23/97 - 08:18	
1953:01	0.837247
1953:02	0.837247
1953:03	0.858662
1953:04	0.891998
1953:05	0.982079
1953:06	1.011601
1953:07	1.011601
1953:08	1.011601
1953:09	1.007958
1953:10	0.936093
1953:11	0.936093

Note that in place of an actual series name, EViews substitutes the expression used to create the auto-series.

Furthermore, the auto-series may be treated as a standard series window so all of the series views and procedures are immediately available.

To display a time series graph of the LOG(CP) series, simply select **View/Line Graph**:

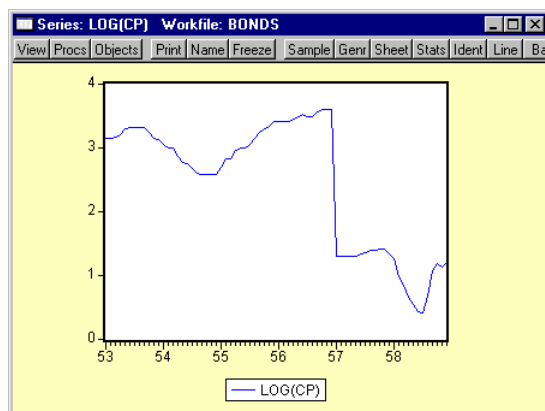


All other views and procedures are also accessible from the menus.

Note that if the data in the CP series are altered, the auto-series will reflect these changes. Suppose, for example, that we take the first four years of the CP series, and multiply it by a factor of 10:

```
smp1 1953:01 1956:12
cp = cp*10
smp1 1953:01 1958:12
```

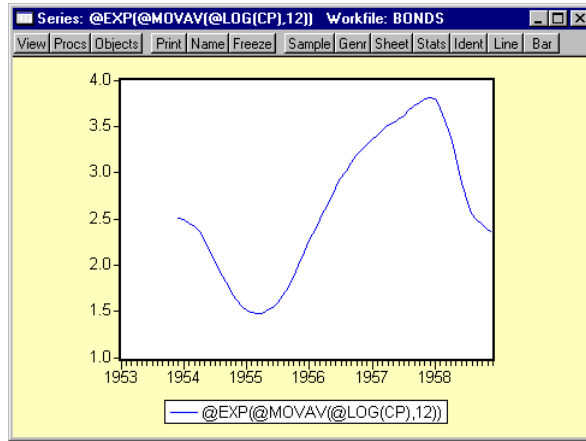
the auto-series graph will automatically change to reflect the new data



Similarly, you may use an auto-series to compute a 12 period, backward-looking, geometric moving average of the *original* CP data. The command:

```
show @exp(@movav(@log(cp), 12))
```

will display the auto-series containing the geometric moving average:

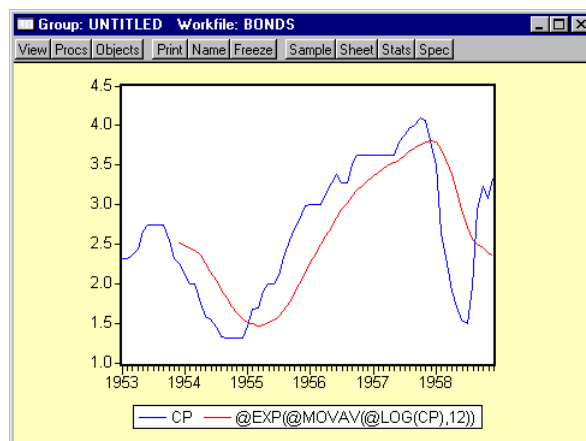


Using Auto-series in Groups

One of the more useful ways of working with auto-series is to include them in a group. Simply create the group as usual, using an expression in place of a series name, as appropriate. For example, if you select **Objects/New Object/Group**, and enter

```
cp @exp(@movav(@log(cp),12))
```

you will create a group containing two series: the ordinary series CP, and the auto-series representing the geometric moving average. We can then use the group object graphing routines to compare the original, with the smoothed series:



“Working with Groups of Series” on page 102 describes other useful techniques for working with auto-series.

Using Auto-Series in Estimation

One method of using auto-series in estimation is to allow expressions as right-hand side variables. Thus, you could estimate an equation with $\log(x)$ or $\exp(x+z)$ as an explanatory variable.

EViews goes a step beyond this use of auto-series, by allowing you to use auto-variables as the dependent variable in estimation. Thus, if you want to regress the log of Y on explanatory variables, you don't have to create a new variable LOGY. Instead, you can use the expression $\log(y)$ as your dependent variable.

When you forecast using an equation with an auto-series dependent variable, EViews will, if possible, forecast the untransformed dependent variable and adjust the estimated confidence interval accordingly. For example, if the dependent variable is specified as $\log(y)$, EViews will allow you to forecast the level of Y, and will compute the asymmetric confidence interval. See [Chapter 14, “Forecasting from an Equation”, on page 343](#) for additional details.

Working with Groups of Series

EViews provides specialized tools for working with groups of series that are held in the form of a group object. In Chapter 4 we used groups to import data from spreadsheets. Briefly, a group is a collection of one or more series identifiers or expressions. Note that a group does not contain the data in the individual series, only references to the data in the series.

To create a group, select **Objects/New Object/Group** and fill in the dialog with names of series and auto-series. Or you can select **Show** from the workfile toolbar and fill out the dialog. Alternatively, type the command `group` in the command window, followed by a name to be given to the group and then the series and auto-series names:

```
group macrolist gdp invest cons
```

creates the group MACROLIST containing the series GDP, INVEST and CONS. Similarly,

```
group altlist log(gdp) d(invest) cons/price
```

creates the group ALTLIST containing the log of the series GDP, the first difference of the series INVEST, and the CONS series divided by the PRICE series.

There are a few features of groups that are worth keeping in mind:

- A group is simply a list of series identifiers. It is not a copy of the data in the series. Thus, if you change the data for one of the series in the group, you will see the changes reflected in the group.

- If you delete a series from the workfile, it will disappear from any group that originally included the series. If the deleted series is the only series in a group, the group will also be deleted.
- Renaming a series changes the reference in every group containing the series.
- There are many routines in EViews where you can use a group name in place of a list of series. If you wish, for example, to use X1, X2 and X3 as right-hand side variables in a regression, you can instead create a group containing the series, and use the group in the regression.

We describe groups in greater detail in [Chapter 8, “Groups”, on page 199](#).

Accessing Individual Series in a Group

Groups, like other EViews objects, contain their own views and procedures. For now, note that you can access the individual elements of a named group as individual series.

To refer the n -th series in the group, simply append “(n)” to the group name. For example, consider the MACROLIST group, defined above. The expression MACROLIST(1) may be used to refer to GDP and MACROLIST(2) to refer to INVEST.

You can work with MACROLIST(1) as though it were any other series in EViews. You can display the series by clicking on the **Show** button on the toolbar and entering MACROLIST(1). You can include GDP in another group directly or indirectly. A group which contains

```
macrolist(1) macrolist(2)
```

will be identical to a group containing

```
gdp invest
```

We can also use the individual group members as part of expressions in generating new series:

```
series realgdp = macrolist(1)/price  
series y = 2*log(macrolist(3))
```

or in modifying the original series:

```
series macrolist(2) = macrolist(2)/price
```

Note that in this latter example the `series` keyword is required, despite the fact that the INVEST series already exists.

Other tools allow you to retrieve the number of series in a group by appending “.@count” to the group name:

```
scalar numgroup = macrolist.@count
```

and to retrieve the names of each of the series using “`@seriesname`”. These tools are described in greater detail in “[Group Data Members](#)” on page 28 of the *Command and Programming Reference*.

Illustration

Access to the members of a group, when paired with auto-series, provides a powerful set of tools for working with series data. As we saw above, auto-series provide you with dynamic updating of expressions. If we use the auto-series expression:

```
log(y)
```

the result will be automatically updated whenever the contents of the series Y changes.

A potential drawback of using auto-series is that expressions may be quite lengthy. For example, the two expressions:

```
log(gdp)/price + d(invest) * (cons + invest)
12345.6789 * 3.14159 / cons^2 + dlog(gdp)
```

are not suited to use as auto-series if they are to be used repeatedly in other expressions.

You can employ group access to make this style of working with data practical. First, create groups containing the expressions:

```
group g1 log(gdp)/price+d(invest)*(cons+invest)
group g2 12345.6789*3.14159/cons^2+dlog(gdp)
```

If there are spaces in the expression, the entire contents should be enclosed in parentheses.

You can now refer to the auto-series as G1(1) and G2(1). You can go even further by combining the two auto-series into a single group:

```
group myseries g1(1) g2(1)
```

and then referring to the series as MYSERIES(1) and MYSERIES(2). If you wish to skip the intermediate step of defining the subgroups G1 and G2, make certain that there are no spaces in the subexpression or that it is enclosed in parentheses. For example, the two expressions in the group ALTSERIES,

```
group altseries (log(gdp)/price) 3.141*cons/price
```

may be referred to as ALTSERIES(1) and ALTSERIES(2).

Working with Scalars

Scalar objects are different from series and groups in that they have no window views. Scalars are created by commands of the form

```
scalar scalar_name = number
```

where you assign a number to the scalar name. The number may be an expression or special functions that return a scalar.

To examine the contents of a scalar, you may enter the command `show`, followed by the name of the scalar. EViews will display the value of the scalar in the status line at the bottom of the EViews window, in the left-hand corner of the status line. For example,

```
scalar logl1=eq1.@logl  
show logl1
```

stores the log likelihood value of the equation object named EQ1 in a scalar named LOGL1, and displays the value in the status line. Alternatively, double clicking on the scalar name in the workfile window displays the value in the status line.

Chapter 6. EViews Databases

An EViews database resembles a workfile in that it is used to contain a collection of EViews objects. It differs from a workfile in two major ways. First, unlike a workfile, the entire database need not be loaded into memory in order to access an object inside it; an object can be fetched or stored directly to the database on disk. Second, the objects in a database are not restricted to being of a single frequency or range. A database could contain a collection of annual, monthly, and daily series, all with different numbers of observations.

EViews databases also differ from workfiles in that they support powerful query features which can be used to search through the database to find a particular series or a set of series with a common property. This makes databases ideal for managing large quantities of data.

While EViews has its own native storage format for databases, EViews also allows direct access to data stored in a variety of other formats through the same database interface. You can perform queries, copy objects to and from workfiles and other databases, and rename and delete objects within a database, all without worrying about in what format the data is actually stored.

An Overview

An EViews database is a set of files containing a collection of EViews objects. In this chapter we describe how to:

- Create a new database or open an existing database.
- Work with objects in the database, including how to store and fetch objects into workfiles, and how to copy, rename and delete objects in the database.
- Use auto-series to work with data directly from the database without creating a copy of the data in the workfile.
- Use the database registry to create shortcuts for long database names and to set up a search path for series names not found in the workfile.
- Perform a query on the database to get a list of objects with particular properties.
- Use object aliases to work with objects whose names are illegal or awkward.
- Maintain a database with operations such as packing, copying, and repairing.
- Work with remote database links to access data from remote sites.

Database Basics

What is an EViews Database?

An EViews native format database consists of a set of files on disk. There is a main file with the extension .EDB which contains the actual object data, and a number of index files with extensions such as .EO, .E1A and .E1B which are used to speed up searching operations on the database. In normal use, EViews manages these files for the user, so there is no need to be aware of this structure. However, if you are copying, moving, renaming, or deleting an EViews database from outside of EViews (using Windows Explorer for example), you should perform the operation on both the main database file and all the index files associated with the database. If you accidentally delete or damage an index file, EViews can regenerate it for you from the main data file using the repair command (see [“Maintaining the Database” on page 133](#)).

The fact that EViews databases are kept on disk rather than in memory has some important consequences. Any changes made to a database cause immediate changes to be made to the disk files associated with the database. Therefore, unlike workfiles, once a change is made to a database, there is no possibility of discarding the change and going back to the previously saved version. Because of this, you should take care when modifying a database, and should consider keeping regular backup copies of databases which you modify frequently.

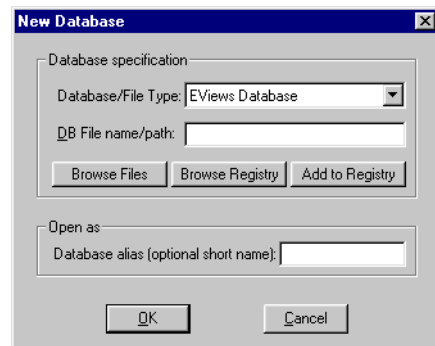
EViews also allows you to deal with a variety of foreign format databases through the same interface provided to EViews’ native format databases. Foreign databases can have many different forms, including files on disk, or data made available through some sort of network server. See [“Foreign Format Databases” on page 135](#) for a discussion of the different types of foreign databases that EViews can access.

Creating a Database

To create a database, simply select **File/New/Database...** from the main menu.

For a native EViews database, simply enter a name for the database in the field labeled **DB File name/path**, then click on the button marked **OK**. This will create a new EViews database in the current path.

To create a database in a different directory, you can enter the full path and database name in the **DB File name/path** edit field. Alternatively, you can browse to the desired directory. Simply click on the **Browse Files** button to call



up the common file dialog, and then navigate to the target directory. Enter the name of the new database in the “File name:” edit field, then click on the **OK** button to accept the information and close the file dialog. EViews will put the new path and filename in the **DB File name/path** edit field.

The **Database/File Type** field allows you to create different types of databases; see “[Foreign Format Databases](#)” on page 135 for a discussion of working with different database types.

The **Open As** field allows you to specify the shorthand that will be associated with this database. A shorthand is a short text label which is used to refer to the database in commands and programs. If you leave this field blank, a default shorthand will be assigned automatically. See “[Database Shorthands](#)” on page 111 for details.

The **Browse Registry** and **Add to Registry** buttons provide a convenient way to recall information associated with a previously registered database or to include the new database in the database registry; see “[The Database Registry](#)” on page 121 for details.

A database can also be created from the command line or in a program using the command:

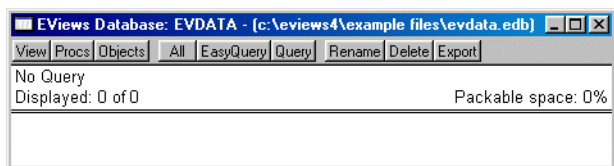
```
dbcreate db_name
```

where db_name is the name of the database using the same rules given above.

The Database Window

When you create a new database, a database window will open on the screen.

The database window provides a graphical interface which allows you to query the database, copy-and-paste objects to and from your work-



file, and perform basic maintenance on the database. Note that some database operations can also be carried out directly without first opening the database window.

To open a database window for an existing database, select **File/Open/Database...** from the main menu. The same dialog will appear as was used during database creation. To open an EViews database, use the **Browse Files** button to select a file using the common file dialog, then click on **OK** to open the file. A new window should appear representing the open database.

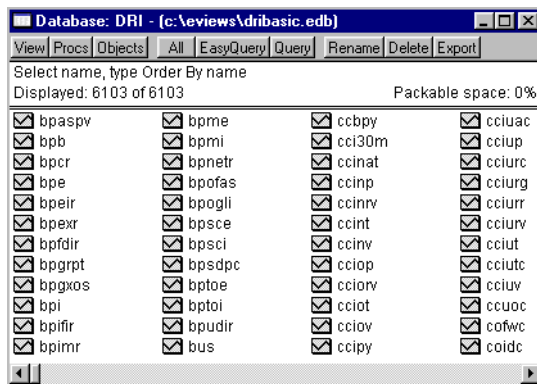
From the command line or in a program, you can open a database window by typing:

```
dbopen db_name
```

Unlike a workfile window, a database window does not display the contents of the database when it is first opened, although it does tell you how many objects are in the database. The second line of the window text shows the number of objects currently displayed (zero when the window is first opened) followed by the total number of objects stored in the database.

You can bring up an alphabetical listing of every object in the workfile by clicking on the **All** button:

As for a workfile, each object is preceded by a small icon that identifies the type of the object. When performing an **All** query, no other information about the object is visible. However, by double clicking on an object you can bring up a full description of the object including its name, type, modification date, frequency, start and end date (for series), and label.



For large databases, the **All** button generally displays too many objects and not enough information about each object. The database query features (“[Querying the Database](#)” on page 123) allow you to control precisely which objects should be displayed, and what information about each object should be visible. The text form of the query currently being displayed is always visible in the top line of the database window.

When working with foreign databases, the object names may appear in color to indicate that they are illegal names or that an alias has been attached to an object name. See “[Object Aliases and Illegal Names](#)” on page 131 for details.

The “Packable space” field in the database window displays the percentage of unused space in the database that can be recovered by a database pack operation; see “[Packing the Database](#)” on page 134 for details.

A brief technical note: having a database window open in EViews generally does not keep a file open at the operating system level. EViews will normally open files only when it is performing operations on those files. Consequently, multiple users can have a database open at the same time and can often perform operations simultaneously. There are some limits imposed by the fact that one user cannot read from a database that another user is writing to at the same time. However, EViews will detect this situation and continue to

retry the operation until the database becomes available. If the database does not become available within a specified time, EViews will generate an error stating that a “sharing violation” on the database has occurred.

For some foreign formats, even minor operations on a database may require full rewriting of the underlying file. In these cases, EViews will hold the file open as long as the database window is open in order to improve efficiency. The formats that currently behave this way are Aremos TSD files, RATS portable files and TSP portable files. When using these formats, only one user at a time may have an open database window for the file.

Database Shorthands

In many situations, EViews allows you to prefix an object name with a database identifier to indicate where the series is located. These database identifiers are referred to as “shorthands”. For example, the command:

```
fetch db1::x db2::y
```

indicates to EViews that the object named X is located in the database with the shorthand db1 and the object named y is located in the database with the shorthand db2.

Whenever a database is opened or created, it is assigned a shorthand. The shorthand can be specified by the user in the **Open as** field when opening a database, or using the “As” clause in the `dbopen` command (see [dbopen](#) (p. 183) of the *Command and Programming Reference*). If a shorthand is explicitly specified when opening a database, an error will occur if the shorthand is already in use.

If no shorthand is provided by the user, a shorthand is assigned automatically. The default value will be the name of the database after any path or extension information has been removed. If this shorthand is already in use, either because a database is already open with the same name, or because an entry in the database registry already uses the name, then a numerical suffix is appended to the shorthand, counting upwards until an unused shorthand is found.

For example, if we open two databases with the same name in a program:

```
dbopen test.edb
dbopen test.dat
```

then the first database will receive the shorthand “TEST” and the second database will receive the shorthand “TEST1”. If we then issue the command:

```
fetch test::x
```

the object will be fetched from the EViews database TEST.EDB. To fetch the object named X from the Haver database TEST.DAT we would use:

```
fetch test1::x
```

To minimize confusion, you should assign explicit shorthands to databases whenever ambiguity could arise. For example, we could explicitly assign the shorthand TEST_HAVER to the second database by replacing the second dbopen command with:

```
dbopen test.dat as test_haver
```

The shorthand attached to a database remains in effect until the database is closed. The shorthand assigned to an open database is displayed in the title bar of the database window.

The Default Database

In order to simplify common operations, EViews uses the concept of a *default database*. The default database is used in several places, the most important of which is as the default source or destination for store or fetch operations when an alternative database is not explicitly specified.

The default database is set by opening a new database window, or by clicking on an already open database window if there are multiple databases open on the screen. The name of the default database is listed in the status line at the bottom of the main EViews window (see [Chapter 3, “EViews Basics”, on page 33](#) for details). The concept is similar to that of the current workfile with one exception: when there are no currently open databases there is still a default database; when there are no currently open workfiles, the current workfile is listed as “none.”

EViews .DB? files

The database features described in this chapter were added in Version 3 of EViews. Previous versions of EViews and MicroTSP supported a much more limited set of database operations. Objects could be stored on disk in individual files, with one object per file. Essentially, the disk directory system was used as a database and each database entry had its own file. These files had the extension “.DB” for series, and .DB followed by an additional character for other types of objects. EViews refers to these collectively as .DB? files.

While the new database features added to EViews provide a superior method of archiving and managing your data, we have continued to support .DB? files to provide backward compatibility, and as a convenient method of distributing data to other programs. Series .DB files are now supported by a large number of programs including TSP, RATS, and SHAZAM. Also, some organizations such as the National Bureau of Economic Research (NBER), distribute data in .DB format.

Working with Objects in Databases

Since databases are simply containers of other EViews objects, most of your work with databases will involve moving objects into and out of them. The sections on storing, fetching and exporting objects discuss different ways of doing this.

You will also need to manage the objects inside a database. You can create duplicate copies of objects, change their names, or remove them from the database entirely. The sections on copying, renaming and deleting discuss how these operations can be carried out.

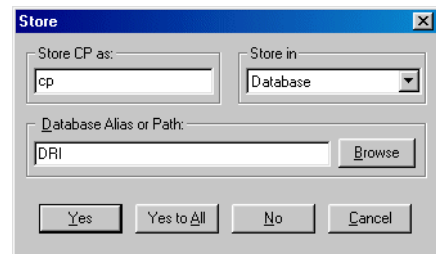
Storing Objects in the Database

An object can be stored in a database in a number of ways. If you have a workfile open on the screen and would like to store objects contained inside it into a database, just select the objects from the workfile window with the mouse, then click on the **Store** button in the workfile toolbar. A sequence of dialogs will come up, one for each object selected, which provide a number of options for renaming the object and determining where the object should be stored.

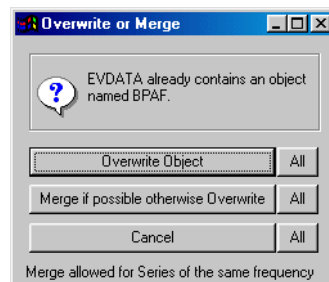
By default, the object will be stored in the current default database with the same name as in the workfile. Click **Yes** to store this object, or **Yes-to-All** to store all the selected objects using these settings.

If you would like to store the object with a different name, simply type the new name over the old name in the edit box labeled **Store**

object_name as: (where *object_name* is the name of the object currently being stored). If you would like to store the object in a different database, either type in the name of the new database in the text box marked **Database Alias or Path:** (see [“The Database Registry” on page 121](#) for an explanation of database aliases), or click on the button marked **Browse** to select the database name interactively. To store the object to disk as an EViews .DB? file, click on the arrow to the right of the field labeled **Store in:** and select **Individual .DB? files**. You may then specify a path in which to place the file using the field labeled **Path for DB files**.



If there is already an existing object in the database with the same name, EViews will display a dialog. The first and last of the three options are self explanatory. The second option can only be used if the object you are storing from the workfile and the object already in the database are both series of the same frequency. In this case, EViews will merge the data from the two series so that the new series in the database has all the observations from the series being stored, as well as any observations from the existing series which have not been overwritten. For example, if the existing series in the database is an annual series from 1950 to 1990, and the series being stored is an annual series from 1980 to 1995, the new series will run from 1950 to 1995, with data from the existing series for 1950 to 1979, and data from the new series for 1980 to 1995.

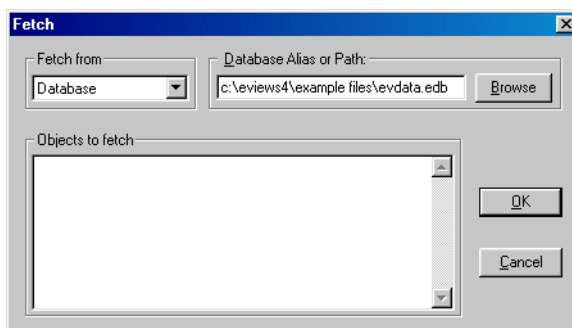


Fetching Objects from the Database

There are a number of ways to fetch objects from a database, most of which are similar to the methods for storing.

The first method is to click on the button marked **Fetch** on the toolbar of the workfile into which you would like to fetch the object. A dialog will come up which is similar to the dialog for store:

The dialog allows you to specify the names of the objects to fetch, and the database or directory from which to retrieve them.



Enter the names of the objects you would like to fetch in the field **Objects to Fetch**. Alternatively, you can use the mouse to select objects from the workfile window before clicking on the

Fetch button, in which case the names of these objects will appear automatically. The fields labeled **Database Alias or Path** and **Fetch from** are the same as for the store dialog with one exception. In addition to **EViews Database** and **Individual .DB? files**, **Fetch from** has an option titled **Search Databases**. This option tells EViews to search multiple databases for objects which match the specified names. To use this option you must first define a search order in the database registry; see [“The Database Registry” on page 121](#).

When you click on **OK**, EViews will fetch all the objects. If an object which is being fetched is already contained in the workfile, a dialog will appear asking whether to replace the object or not. Click on **Yes** to replace the object in the workfile or **No** to leave the object in the workfile unchanged.

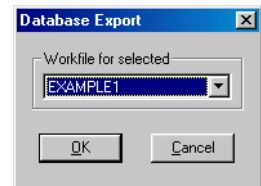
Because a workfile has a fixed frequency and range, fetching a series into a workfile may cause the data in the series to be modified to match the frequency and range of the workfile (see “[Frequency Conversion](#)” on page 72 for details). Be aware that loading a series into a workfile then saving it back into the database can cause truncation and frequency conversion of the series stored in the database.

Objects/Update selected from DB... from the workfile toolbar is the same as **Fetch** except that there is no overwrite warning message. If the object in the database is the same type as the one in the workfile, it is automatically overwritten. If it is of a different type, the fetch does not proceed. Update is also available from the **Objects** button in individual object windows.

Database Export

You can also move data into a workfile from the database window. From an open database window, select the objects you would like to copy using the mouse, then click on the button marked **Export** in the toolbar at the top of the database window. A dialog titled “Database Export” will appear on the screen:

When you click on the down arrow on the right of the field labeled **Workfile for selected objects:**, a list of all workfiles that are currently open should appear. Choose the workfile into which you would like to copy the objects. Clicking on the button marked **OK** will copy the selected objects to the chosen workfile.



In the list of open workfiles is an extra option called **New Workfile**. Choosing this option allows you to create a new workfile containing the objects you have selected. After you click on **OK**, a dialog will appear in which you can set the frequency and range of the workfile to be created. The default frequency is set to the lowest frequency of any of the objects selected, and the default range is set to cover all the data points contained in the objects. Clicking on **OK** will open a new workfile window and copy the selected objects into it, performing frequency conversions where necessary.

Copying Objects

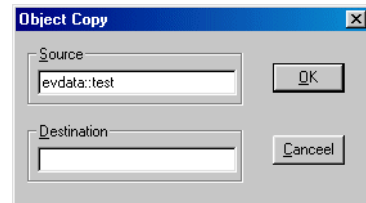
In addition to the above methods for moving objects, EViews provides general support for the copying and pasting of objects between any two EViews container objects (workfiles or databases). You can use these features to move objects between two databases or between

two workfiles, to create duplicate copies of objects within a workfile or database, or as an alternative method for store and fetch.

For copying objects between containers, the procedure is very similar no matter what types of container objects are involved. Before you start, make sure that the windows for both containers are open on the screen. In the container *from which* you would like to copy the objects, select the objects then click on **Edit/Copy** in the EViews program menu. Click on the container object *into which* you would like to paste the objects, then select **Edit/Paste** from the EViews program menu.

Depending on the types of the two containers, you will be presented with one or more dialogs. If you are performing a fetch or store operation, the same dialogs will appear as if you had carried out the operations using the toolbar buttons on the workfile window.

You can perform similar operations, as well as creating duplicate copies of an object within a container, by using the object copy procedure. From the main menu select **Object/Copy** (this may appear as **Object/Copy selected...**). The Object Copy dialog will be displayed.



The **Source** field specifies the object or objects you would like to copy, the **Destination** field specifies where you would like to copy them and what names they should be given.

The **Source** field should be filled in with an expression of the form:

```
source_db::source_pattern
```

where `source_db::` is optional, and indicates which database the objects should be copied from (if no database name is supplied, the source is taken to be the default workfile), and `source_pattern` is either a simple object name or a name pattern. A name pattern may include the wildcard characters “?” which matches any single character, and “*” which matches zero or more characters.

The **Destination** field should be filled in with an expression of the form:

```
dest_db::dest_name
```

where `dest_db::` is again optional, and indicates which database the objects should be copied to (if no database name is supplied, the destination is taken to be the default workfile), and `dest_name`, which is also optional, is the name to be given to the new copy of the object. If no name is given, the object will be copied with its existing name. If a pattern was used when specifying the source, a pattern must also be used when specifying the destination (see [“Source and Destination Patterns”](#) on page 658 for details).

For example, to copy an object from the database DB1 to the database DB2, keeping the existing name, you would fill in the dialog:

```
source:          db1::object_name
destination:     db2::
```

where OBJECT_NAME is the original name as displayed by EViews.

To copy all the objects in the database DB1 beginning with the letter X into the current workfile, changing the names so that they begin with Y, you would fill in the dialog

```
source:          db1::x*
destination:     y*
```

To make a duplicate copy of the object named ABC in the database DB1, giving it the new name XYZ you would fill in the dialog:

```
source:          db1::abc
destination:     db1::xyz
```

Renaming Objects in the Database

You may rename an object in the database by selecting the object in an open database window, then clicking on the button marked **Rename** in the database window toolbar. A dialog will come up in which you can modify the existing name or type in a new name. You can rename several objects at the same time using wildcard patterns and the `rename` command.

Deleting Objects From the Database

To delete objects from the database, select the objects in an open database window, then click on the button marked **Delete** on the database window toolbar. You can delete several objects at the same time using wildcard patterns. There is also a `delete` command. See the *Command and Programming Reference* for details.

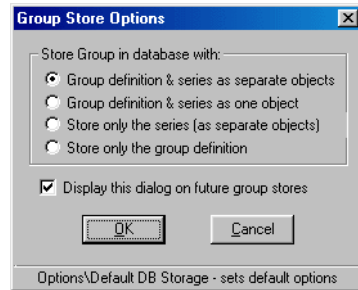
Store, Fetch, and Copy of Group Objects

A group object in EViews is essentially a list of series names that form the group. The data of each series are contained in the series object, not in the group object. When you do a store, fetch, or copy operation on a group object, an issue arises as to whether you want to do the operation on each of the series or to the group definition list.

Storing a Group Object

When you store a group object to a database, there are four available options:

- Store the group definition and the series as separate objects: stores the group object (only its definition information) and each of its series as separate objects in the database. If any of the series already exist in the database, EViews will ask whether to overwrite the existing series if in interactive mode, and will error if in batch mode.
- Store the group definition and the series as one object: stores each series within the group object. A group object that contains series data will have an icon G+ in the database directory window. A group object with only its definition information will have the usual icon G. If you use this option, you can store two different series with the same name (with one of the series as member of a group).
- Store only the series (as separate objects): only stores each series as separate objects in the database. If you want to store a long list of series into a database, you can create a temporary group object that contains those series and issue the store command only once.
- Store only the group definition: stores only the group definition information; none of the series data are stored in the database. This option is useful if you want to update the member data from the database but want to keep the group information (e.g. the dated data table settings) in the group.



By default, EViews will display a dialog asking you to select a group store option every time you store a group object. You can, however, instruct EViews to suppress the dialog and use the global option setting. Simply click on **Options/Database Default Storage Options...** to bring up a dialog that allows you both to set the global storage options, and to suppress the group store option dialog.

Fetching a Group Object

When you fetch a group object to a database, there are three options available:

- Fetch both group definition and the actual series: fetches both group definition and its series as separate objects. If any of the series defined in the group is not found in the database, the corresponding series will be created in the workfile filled with NAs. If any of the series already exist in the workfile, EViews will ask whether to overwrite the existing series or not if in interactive mode, and will error if in batch mode.



- Fetch only the series in the group: only fetches each series defined in the group. If the series exists both within the group object (with a G+ icon) and as a separate series object in the database, the series within the group object will be fetched.
- Fetch only the group definition: fetches only the group definition (but not the series data). If any of the series defined in the group does not exist in the workfile, EViews will create the corresponding series filled with NAs.

You can click on **Options/Database Default Storage Options...** to bring up a dialog that allows you both to set the global fetch options, and to suppress the fetch option dialog.

Copying Group Objects between Workfiles and Databases

You can also copy groups between different containers. The options that are available will differ depending on the type of source and destination container:

- Copy from workfile to database: same options as the store operation.
- Copy from database to workfile: same options as the fetch operation.
- Copy from workfile to workfile: both the group definition and series will be copied.
- Copy from database to database. If the group object contains only the group definition (with a G icon), only the group definition will be copied. If the group object also contains its series data (with a G+ icon), then the group will be copied containing the series data and the copied group will also appear with a G+ icon.

Database Auto-Series

We have described how to fetch series into a workfile. There is an alternative way of working with databases which allows you to make direct use of the series contained in a database without first copying the series. The advantage of this approach is that you need not go through the process of importing the data every time the database is revised. This approach follows the model of auto-series in EViews as described in [“Working with Auto-series” beginning on page 98](#).

There are many places in EViews where you can use a series expression, such as $\log(X)$, instead of a simple series name and EViews will automatically create a temporary auto-series for use in the procedure. This functionality has been extended so that you can now directly refer to a series in a database using the syntax:

```
db_name::object_name
```

where `db_name` is the shorthand associated with the database. If you omit the database name and simply prefix the object name with a double colon like this:

```
::object_name
```

EViews will look for the object in the default database.

A simple example is to generate a new series:

```
series lgdp = log(macro_db::gdp)
```

EViews will fetch the series named GDP from the database with the shorthand MACRO_DB, and put the log of GDP in a new series named LGDP in the workfile. It then deletes the series GDP from memory, unless it is in use by another object. Note that the generated series LGDP only contains data for observations within the current workfile sample.

You can also use auto-series in a regression. For example:

```
equation eq1.ls log(db1::y) c log(db2::x)
```

This will fetch the series named Y and X from the databases named DB1 and DB2, perform any necessary frequency conversions and end point truncation so that they are suitable for use in the current workfile, take the log of each of the series, then run the requested regression. Y and X are then deleted from memory unless they are otherwise in use.

The auto-series feature can be further extended to include automatic searching of databases according to rules set in the database registry (see [“The Database Registry” on page 121](#), for details). Using the database registry you can specify a list of databases to search whenever a series you request cannot be found in the workfile. With this feature enabled, the `series` command

```
series lgdp = log(gdp)
```

looks in the workfile for a series named GDP. If it is not found, EViews will search through the list of databases one by one until a series called GDP is found. When found, the series will be fetched into EViews so that the expression can be evaluated. Similarly, the regression

```
ls log(y) c log(x)
```

will fetch Y and/or X if they are not found in the workfile. Note that the regression output will label all variables with the database name from which they were imported.

In general, using auto-series directly from the database has the advantage that the data will be completely up to date. If the series in the database are revised, you do not need to repeat the step of importing the data into the workfile. You can simply reestimate the equation or model and EViews will automatically retrieve new copies of any data which are required.

There is one complication to this discussion which results from the rules which regulate the updating and deletion of auto-series in general. If there is an existing copy of an auto-series already in use in EViews, then a second use of the same expression will not cause the expression to be reevaluated (in this case reloaded from the database); it will simply make use of the existing copy. If the data in the database have changed since the last time the auto-series was loaded, the new expression will use the old data.

One implication of this behavior is that a copy of a series from a database can persist for any length of time if it is stored as a member in a group. For example, if you type:

```
show db1::y db2::x
```

this will create an untitled group in the workfile containing the expressions `db1::y` and `db2::x`. If the group window is left open and the data in the database are modified (for example by a `store` or a `copy` command), the group and its window will not update automatically. Furthermore, if the regression

```
ls log(db1::y) c log(db2::x)
```

is run again, this will use the copies of the series contained in the untitled group; it will not refetch the series from the database.

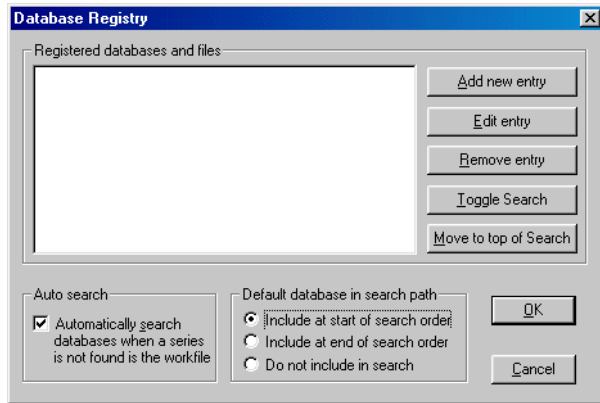
The Database Registry

The database registry is a file on disk that manages a variety of options which control database operations. It gives you the ability to assign short alias names that can be used in place of complete database paths, and it allows you to configure the automatic searching features of EViews.

Options/Database Registry...

from the main menu brings up the Database Registry dialog allowing you to view and edit the database registry:

The box labeled **Registry Entries** lists the databases that have been registered with EViews. The first time you bring up the dialog, the box should be empty. If you click on the **Add new entry** button, a second dialog appears.



There are three things you must specify in the dialog: the full name (including path) of the database, the alias which you would like to associate with the database, and the option for whether you wish to include the database in automatic searches.

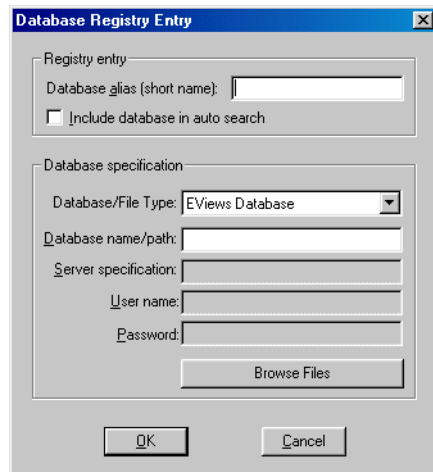
The full name and path of the database should be entered in the top edit field. Alternatively, click the **Browse** button to select your database interactively.

The next piece of information you must provide is a *database alias*: a short name that you can use in place of the full database path in EViews commands. The database alias will also be used by EViews to label database auto-series. For example, suppose you have a database named DRIBASIC in the subdirectory DATA on drive C. The following regression command is legal but awkward:

```
ls c:\data\dribasic::gdp c c:\data\dribasic::gdp (-1)
```

Long database names such as these may also cause output labels to truncate, making it difficult to see which series were used in a procedure.

By assigning DRIBASIC the alias DRI, we can use the more readable command:



```
ls dri::gdp c dri::gdp(-1)
```

and the regression output will be labeled with the shorter names. To minimize the possibility of truncation, we recommend the use of short alias names if you intend to make use of database auto-series.

Finally you should tell EViews if you want to include the database in automatic database searches by checking the **Include in auto search** checkbox. Clicking on **OK** will return you to the main Database Registry dialog with the new entry included in the list.

Any registry entry may be edited, deleted, switched on or off for searching, or moved to the top of the search order by highlighting the entry in the list and clicking the appropriate button to the right of the list box.

The remainder of the Database Registry dialog allows you to set options for automatic database searching. The **Auto-search** checkbox is used to control EViews behavior when you enter a command involving a series name which cannot be found in the current workfile. If this checkbox is selected, EViews will automatically search all databases that are registered for searching, before returning an error. If a series with the unrecognized name is found in any of the databases, EViews will create a database auto-series and continue with the procedure.

The last section of the dialog, **Default Database in Search Order**, lets you specify how the default database is treated in automatic database searches. Normally, when performing an automatic search, EViews will search through the databases contained in the **Registry Entries** window in the order that they are listed (provided that the **Include in auto search** box for that entry has been checked). These options allow you to assign a special role to the default database when performing a search.

- **Include at start of search order**—means that the current default database will be searched first, before searching the listed databases.
- **Include at end of search order**—means that the current default database will be searched last, after searching the listed databases.
- **Do not include in search**—means that the current default database will not be searched unless it is already one of the listed databases.

Querying the Database

A great deal of the power of the database comes from its extensive query capabilities. These capabilities make it easy to locate a particular object, or to perform operations on a set of objects which share similar properties.

The query capabilities of the database can only be used interactively from the database window. There are two ways of performing a query on the database: the easy mode and

the advanced mode. Both methods are really just different ways of building up a text query to the database. The easy mode provides a simpler interface for performing the most common types of queries. The advanced mode offers more flexibility at the cost of increased complexity.

Easy Queries

To perform an easy query, first open the database, then click on the **EasyQuery** button in the toolbar at the top of the database window. The Easy Query dialog will appear containing two text fields and a number of check boxes:

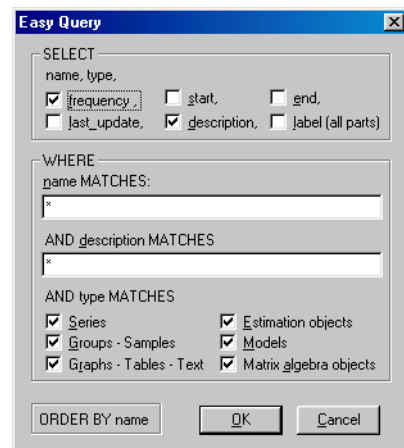
There are two main sections to this dialog: **Select** and **Where**. The **Select** section determines which fields to display for each object that meets the query condition. The **Where** section allows you to specify conditions that must be met for an object to be returned from the query. An Easy Query allows you to set conditions on the object name, object description, and/or object type.

The two edit fields (**name** and **description**) and the set of check boxes (**object type**) in the **Where** section provide three filters of objects that are returned from the query to the database. The filters are applied in sequence (using a logical ‘and’ operation) so that objects in the database must meet all of the criteria selected in order to appear in the results window of the query.

The **name** and **description** fields are each used to specify a *pattern expression* that the object must meet in order to satisfy the query. The simplest possible pattern expression consists of a single pattern. A pattern can either be a simple word consisting of alphanumeric characters, or a pattern made up of a combination of alphanumeric characters and the wildcard symbols “?” and “*”, where “?” means to match any one character and “*” means to match zero or more characters. For example:

```
pr?d*ction
```

would successfully match the words production, prediction, and predilection. Frequently used patterns include “s*” for words beginning in “S”, “*s” for words ending in “S”, and “*s*” for words containing “S”. Upper or lower case is not significant when searching for matches.



Matching is done on a word by word basis where at least one word in the text must match the pattern for it to match overall. Since object names in a database consist of only a single word, pattern matching for names consists of simply matching this word.

For descriptions, words are constructed as follows: each word consists of a set of consecutive alphanumeric characters, underlines, dollar signs, or apostrophes. However, the following list words are explicitly ignored: “a”, “an”, “and”, “any”, “are”, “as”, “be”, “between”, “by”, “for”, “from”, “if”, “in”, “is”, “it”, “not”, “must”, “of”, “on”, “or”, “should”, “that”, “the”, “then”, “this”, “to”, “with”, “when”, “where”, “while”. (This is done for reasons of efficiency, and to minimize false matches to patterns from uninteresting words). The three words “and”, “or”, and “not” are used for logical expressions.

For example,

```
bal. of p'ment: seas.adj. by X11
```

is broken into the following words: “bal”, “p'ment”, “seas”, “adj”, and “x11”. The words “of” and “by” are ignored.

A pattern expression can also consist of one or more patterns joined together with the logical operators “and”, “or” and “not” in a manner similar to that used in evaluating logical expressions in EViews. That is, the keyword `and` requires that both the surrounding conditions be met, the keyword `or` requires that either of the surrounding conditions be met, and the keyword `not` requires that the condition to the right of the operator is not met. For example:

```
s* and not *s
```

matches all objects which contain words which begin with, but do not end with, the letter S.

More than one operator can be used in an expression, in which case parentheses can be added to determine precedence (the order in which the operators are evaluated). Operators inside parentheses are always evaluated logically prior to operators outside parentheses. Nesting of parentheses is allowed. If there are no parentheses, the precedence of the operators is determined by the following rules: `not` is always applied first; `and` is applied second; and `or` is applied last. For example,

```
p* or s* and not *s
```

matches all objects which contain words beginning with P, or all objects which contain words which begin with, but do not end with, the letter S.

The third filter provided in the Easy Query dialog is the ability to filter by object type. Simply select the object types which you would like displayed, using the set of check boxes near the bottom of the dialog.

Advanced Queries

Advanced queries allow considerably more control over both the filtering and the results which are displayed from a query. Because of this flexibility, advanced queries require some understanding of the structure of an EViews database to be used effectively.

Each object in an EViews database is described by a set of fields. Each field is identified by a name. The current list of fields includes:

name	The name of the object.
type	The type of the object.
last_write	The time this object was last written to the database.
last_update	The time this object was last modified by EViews.
freq	The frequency of the data contained in the object.
start	The date of the first observation contained in the object.
end	The date of the last observation contained in the object.
obs	The number of data points stored in the series (including missing values).
description	A brief description of the object.
source	The source of the object.
units	The units of the object.
remarks	Additional remarks associated with the object.
history	Recent modifications of the object by EViews.
display_name	The EViews display name.

An advanced query allows you to examine the contents of any of these fields, and to select objects from the database by placing conditions on these fields. An advanced query can be performed by opening the database window, then clicking on the button marked **Query** in the toolbar at the top of the window. The Advanced Query dialog is displayed.

The first edit field labeled **Select:** is used to specify a list of all the fields that you would like displayed in the query results. Input into this text box consists of a series of field names separated by commas. Note that the name and type fields are always fetched automatically.

The ordering of display of the results of a query is determined by the **Order By** edit field. Any field name can be entered into this box, though some fields are likely to be more useful than others. The description field, for example, does not provide a useful ordering of the objects. The **Order By** field can be useful for

grouping together objects with the same value of a particular field. For example, ordering by `type` is an effective way to group together the results so that objects of the same type are placed together in the database window. The **Ascending** and **Descending** buttons can be used to reverse the ordering of the objects. For example, to see objects listed from those most recently written in the database to those least recently written, one could simply sort by the field `last_write` in **Descending** order.

The **Where** edit field is the most complicated part of the query. Input consists of a logical expression built up from conditions on the fields of the database. The simplest expression is an operator applied to a single field of the database. For example, to search for all series which are of monthly or higher frequencies (where higher frequency means containing more observations per time interval), the appropriate expression is:

```
freq >= monthly
```

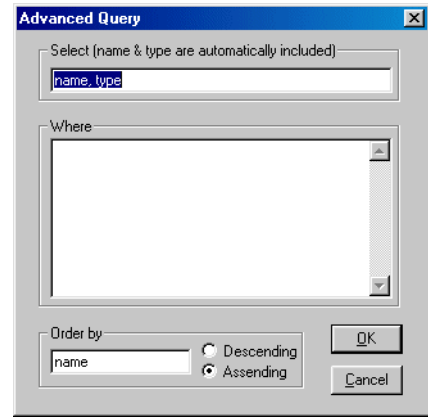
Field expressions can also be combined with the logical operators `and`, `or` and `not` with precedence following the same rules as those described above in the section on easy queries. For example, to query for all series of monthly or higher frequencies which begin before 1950 we could enter the expression:

```
freq >= monthly and start < 1950
```

Each field has its own rules as to the operators and constants which can be used with the field.

Name

The name field supports the operators “<”, “<=”, “>”, “>=”, “=”, and “<>” to perform typical comparisons on the name string using alphabetical ordering. For example,



```
name >= c and name < m
```

will match all objects with names beginning with letters from C to L. The name field also supports the operator “matches”. This is the operator which is used for filtering the name field in the easy query and is documented extensively in the previous section. Note that if `matches` is used with an expression involving more than one word, the expression must be contained in quotation marks. For example,

```
name matches "x* or y*" and freq = quarterly
```

is a valid query, while

```
name matches x* or y* and freq = quarterly
```

is a syntax error because the part of the expression that is related to the `matches` operator is ambiguous.

Type

The type field can be compared to the following object types in EViews using the “=” operator: `sample`, `equation`, `graph`, `table`, `text`, `program`, `model`, `system`, `var`, `pool`, `sspace`, `matrix`, `group`, `sym`, `matrix`, `vector`, `coef`, `series`. Relational operators are defined for the type field, although there is no particular logic to the ordering. The ordering can be used, however, to group together objects of similar types in the **Order By** field.

Freq

The frequency field has one of the following values:

u	Undated
a	Annual
s	Semiannual
q	Quarterly
m	Monthly
w	Weekly
5	5 day daily
7	7 day daily

Any word beginning with the letter above is taken to denote that particular frequency, so that monthly can either be written as “m” or “monthly”. Ordering over frequencies is defined so that a frequency with more observations per time interval is considered “greater” than a series with fewer observations per time interval. The operators “<”, “>”, “<=”, “>=”, “=”, “<>” are all defined according to these rules. For example,

```
freq <= quarterly
```

will match objects whose frequencies are quarterly, semiannual, annual or undated.

Start and End

Start and end dates use the following representation. A date from an annual series is written as an unadorned year number such as “1980”. A date from a semiannual series is written as a year number followed by an “S” followed by the six month period, for example “1980S2”. The same pattern is followed for quarterly and monthly data using the letters “Q” and “M” between the year and period number. Weekly, five day daily and 7 day daily data are denoted by a date in the format

```
mm/dd/yyyy
```

where *m* denotes a month digit, *d* denotes a day digit, and *y* denotes a year digit.

Operators on dates are defined in accordance with calendar ordering where an earlier date is less than a later date. Where a number of days are contained in a period, such as for monthly or quarterly data, an observation is ordered according to the first day of the period. For example,

```
start <= 1950
```

will include dates whose attributed day is the first of January 1950, but will not include dates which are associated with other days in 1950, such as the second, third, or fourth quarter of 1950. However, the expression

```
start < 1951
```

would include all intermediate quarters of 1950.

Last_write and Last_update

As stated above, `last_write` refers to the time the object was written to disk, while `last_update` refers to the time the object was last modified inside EViews. For example, if a new series was generated in a workfile, then stored in a database at some later time, `last_write` would contain the time that the store command was executed, while `last_update` would contain the time the new series was generated. Both of these fields contain date and time information which is displayed in the format:

```
mm/dd/yyyy hh:mm
```

where *m* represents a month digit, *d* represents a day digit, *y* represents a year digit, *h* represents an hour digit and *m* represents a minute digit.

The comparison operators are defined on the time fields so that earlier dates and times are considered less than later dates and times. A typical comparison has the form:

```
last_write >= mm/dd/yyyy
```

A day constant always refers to twelve o'clock midnight at the beginning of that day. There is no way to specify a particular time during the day.

Description, Source, Units, Remarks, History, Display_name

These fields contain the label information associated with each object (which can be edited using the Label view of the object in the workfile). Only one operator is available on these fields, the `matches` operator, which behaves exactly the same as the description field in the section on easy queries.

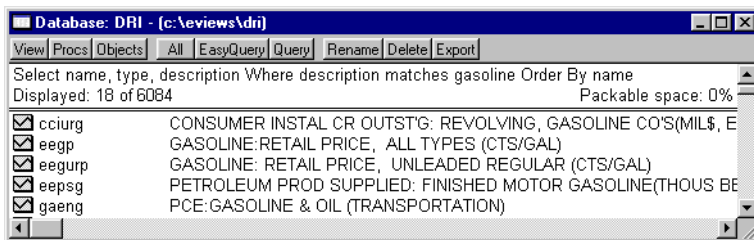
Query Examples

Suppose you are looking for data related to gasoline consumption and gasoline prices in the database named DRIBASIC. First open the database: click **File/Open**, select **Files of type: Database .edb** and locate the database. From the database window, click **Query** and fill in the Advanced Query dialog as follows:

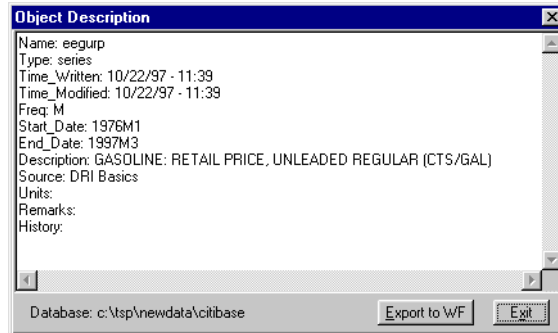
Select: name, type, description

Where: description matches gasoline

If there are any matches, the results are displayed in the database window like this:



To view the contents of all fields of an item, double click on its name. EViews will open an Object Description window that looks as follows:



To further restrict your search to series with at least quarterly frequency and to display the start and end dates of the results, click **Query** and again and modify the fields as follows:

Select: name, type, start, end, description

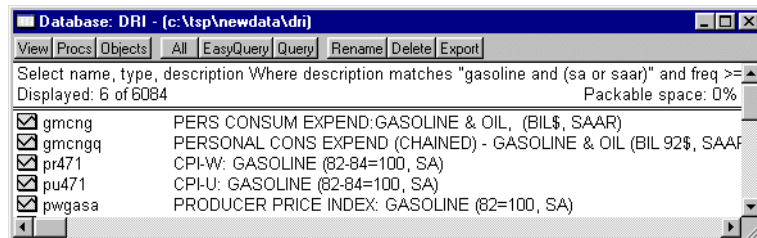
Where: description matches gasoline and freq>=q

If you are interested in seasonally adjusted series, which happen to contain sa or saar in their description in this database, further modify the fields to

Select: name, type, start, end, description

Where: description matches "gasoline and (sa or saar)" and freq>=q

The display of the query results now looks as follows:



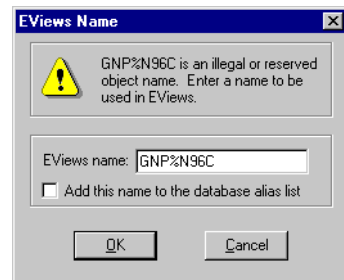
Object Aliases and Illegal Names

When working with a database, EViews allows you to create a list of aliases for that database which allow you to refer to objects inside a database by a different name. The most important use of this is when working with a database in a foreign format where some of the names used in the database are not legal EViews object names. However, the aliasing features of EViews can also be used in other contexts, such as to assign a shorter name to a series with an inconveniently long name.

The basic idea is as follows: each database can have one or more *object aliases* associated with it where each alias entry consists of the name of the object in the database and the name by which you would like it to be known in EViews.

The easiest way to create an object alias for an illegal name is simply to try and fetch the object with the illegal name into EViews. If you are working with query results, you can tell which object names are illegal because they will be displayed in the database window in red. When you try to fetch an object with an illegal name, a dialog will appear.

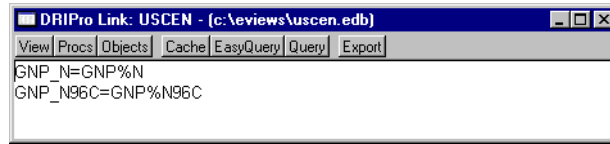
The field labeled **EViews Name:** initially contains the illegal name of the database object. You should edit this to form a legal EViews object name. In this example, we could change the name GNP%N96C to GNP_N_96C. The checkbox labeled **Add this name to the database alias list** (which by default is not checked), determines whether you want to create a permanent association between the name you have just typed and the illegal name. If you check the box, then whenever you use the edited object name in the future, EViews will take it to refer to the underlying illegal name. The edited name acts as an *alias* for the underlying name. It is as though you had renamed the object in the database to the new legal name, except that you have not actually modified the database itself, and your changes will not affect other users of the database.



When EViews displays an object in the database window for which an alias has been set, EViews will show the alias, rather than the underlying name of the object. In order to indicate that this substitution has been done, EViews displays the alias in blue.

Creating an alias can cause *shadowing* of object names. Shadowing occurs when you create an alias for an object in the database, but the name you use as an alias is the name of another object in the database. Because the existence of the alias will stop you from accessing the other object, that object is said to be shadowed. To indicate that an object name being displayed has been shadowed, EViews displays the name of shadowed objects in green. You will not be able to fetch an object which has been shadowed without modifying either its name or the alias which is causing it to be shadowed. Even if the shadowed series is explicitly selected with the mouse, operations performed on the series will use the series with the conflicting alias, not the shadowed series.

You can view a list of the aliases currently defined for any database by clicking on the **View** button at the top of the database window, then selecting **Object Aliases**. A list of all the aliases will be displayed in the window.



Each line represents one alias attached to the database and follows the format:

```
alias = database_object_name
```

You can edit the list of aliases to delete unwanted entries, or you can type in, or cut-and-paste, new entries into the file. You must follow the rule that both the set of aliases and the set of database names do not contain any repeated entries. (If you do not follow this rule, EViews will refuse to save your changes). To save any modifications you have made, simply switch back to the **Object Display** view of the database. EViews will prompt you for whether you want to save or discard your edits.

The list of currently defined database aliases for all databases is kept in the file OBALIAS.INI in the EViews installation directory. If you would like to replicate a particular set of aliases onto a different machine, you should copy this file to the other machine, or use a text editor to combine a portion of this file with the file already in use on the other machine. You must exit and restart EViews to be sure that EViews will reread the aliases from the file.

Maintaining the Database

In many cases an EViews database should function adequately without any explicit maintenance. Where maintenance is necessary, EViews provides a number of procedures to help you perform common tasks.

Database File Operations

Because EViews databases are spread across multiple files, all of which have the same name but different extensions, simple file operations like copy, rename and delete require multiple actions if performed outside of EViews. The **Procs** button in the database window toolbar contains the procedures **Copy the database**, **Rename the database**, and **Delete the database** that carry out the chosen operation on all of the files that make up the database.

Note that file operations do not automatically update the database registry. If you delete or rename a database that is registered, you should either create a new database with the same name and location, or edit the registry.

Packing the Database

If many objects are deleted from an EViews database without new objects being inserted, a large amount of unused space will be left in the database. In addition, if objects are frequently overwritten in the database, there will be a tendency for the database to grow gradually in size. The extent of growth will depend on circumstances, but a typical database is likely to stabilize at a size around 60% larger than what it would be if it were written in a single pass.

A database can be compacted down to its minimum size by using the pack procedure. Simply click on the button marked **Procs** in the toolbar at the top of the database window, then select the menu item **Pack the Database**. Depending on the size of the database and the speed of the computer which you are using, performing this operation may take a significant amount of time.

You can get some idea of the amount of space that will be reclaimed during a pack by looking at the Packable Space percentage displayed in the top right corner of the database window. A figure of 30%, for example, indicates that roughly a third of the database file consists of unused space. A more precise figure can be obtained from the **Database Statistics** view of a database. The number following the label “unused space” gives the number of bytes of unused space contained in the main database file.

Dealing with Errors

The best way to protect against damage to a database is to make regular backup copies of the database. This can be performed easily using the **Copy the Database** procedure documented above. EViews provides a number of other features to help you deal with damaged databases.

Damaged databases can be divided into two basic categories depending on how severely the database has been damaged. A database which can still be opened in a database window, but generates an error when performing some operations, may not be severely damaged, and may be able to be repaired. A database which can no longer be opened in a database window is severely damaged and will need to be rebuilt as a new database.

EViews has two procedures designed for working with databases which can be opened: **Test Database Integrity** and **Repair Database**. Both procedures are accessed by clicking on the button marked **Procs** in the database window toolbar, then selecting the appropriate menu item.

Test Database Integrity conducts a series of validity checks on the main database and index files. If an error is detected, a message box will be displayed providing some information as to the type of error found, and a suggestion as to how it might be dealt with. Because testing performs a large number of consistency checks on the database files, it may take considerable time to complete. You can monitor its progress by watching the

messages displayed in the status line at the bottom of the EViews window. Testing a database does not modify the database in any way, and will never create additional damage to a database.

Repair Database will attempt to automatically detect and correct simple problems in the database. Although care has been taken to make this command as safe as possible, it will attempt to modify a damaged database, so it is probably best to make a back up copy of a damaged database before running this procedure.

Rebuilding the Database

If the database is badly corrupted, it may not be possible for it to be repaired. In this case, EViews gives you the option of building a new database from the old one using the `dbrebuild` command. This operation can only be performed from the command line (since it may be impossible to open the database). The command is:

```
dbrebuild old_dbname new_dbname
```

The `dbrebuild` command does a low level scan through the main data file of the database `old_dbname` looking for any objects which can be recovered. Objects which it finds are copied into the new database `new_dbname`. This is a very time consuming process, but should be able to recover as much data as possible from even heavily damaged files.

Foreign Format Databases

While most of your work with databases will probably involve using EViews native format databases, EViews also gives you the ability to access data stored in a variety of other formats using the same database interface. You can perform queries, copy objects to and from workfiles and other databases, rename and delete objects within the database, add databases to your search path, and use EViews' name aliasing features, all without worrying about how the data is actually stored.

When copying objects, EViews preserves not only the data itself, but as much as possible of any date information and documentation associated with the object. Missing values are translated automatically.

To Convert Or Not To Convert?

Although EViews allows you to work with foreign files in their native format, in some cases you may be better off translating the entire foreign file into EViews format. If necessary, you can then translate the entire file back again when your work is complete. EViews native databases have been designed to support a certain set of operations efficiently, and while access to foreign formats has been kept as fast as possible, in some cases there will be substantial differences in performance depending on the format in use.

One significant difference is the time taken to search for objects using keywords in the description field. If the data is in EViews' format, EViews can typically query databases containing tens of thousands of series in a couple of seconds. When working with other formats, you may find that this same operation takes much longer, with the time increasing substantially as the database grows.

On the other hand, keeping the data in the foreign format may allow you to move between a number of applications without having to retranslate the file. This minimizes the number of copies of the data you have available, which may make the data easier to update and maintain.

Using EViews, you can either translate your data, or work with your data directly in the foreign format. You should choose between the two based on your particular needs.

Opening a Foreign Database

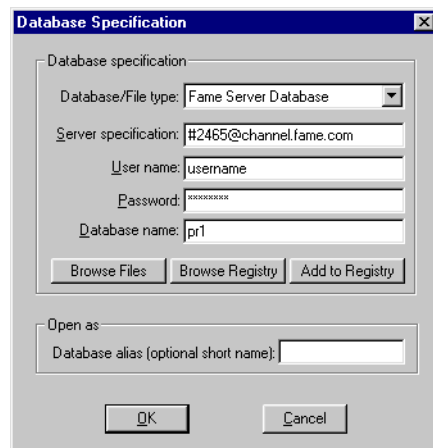
Working with foreign formats requires very little additional knowledge. To open a foreign database, simply select **File/Open/Database...** from the main menu to open the dialog. In the field **Database/File Type:** select the type of the foreign database or file you wish to open. If the database is a local file, you can then use the **Browse Files** button to locate the database in exactly the same way as for a native EViews database. You can create a new foreign format database by a similar procedure way using **File/New/Database...** from the main EViews menu.

If the database is accessed through a client-server model, selecting the dialog will change to show extra fields necessary for making the connection to the server. For example, when accessing a database located on a FAME server, the dialog will include fields for the FAME server, username and password.

Since access to a server requires many fields to be entered, you may wish to save this information as an entry in the database registry (see [“The Database Registry”](#) on page 121 for details).

There are special issues relating to working with DRIPro links. See [“DRIPro Link”](#) on page 137 for details.

You can also create and open foreign format files using the “dbopen” and “dbcreate” commands. You can either use an option to explicitly specify the foreign type, or let EViews determine the type using the file extension. See



[dbopen](#) (p. 183) and [dbcreate](#) (p. 181) in the *Command and Programming Reference* for details.

Copying a Foreign Database

Once you have opened a window to a foreign database, you can copy the entire database into a new format using **Procs/Copy the Database** from the database menus. A dialog will appear which allows you to specify the type and other attributes of the new database you would like to create.

When performing a database copy to a new format, objects which cannot be copied due to incompatibility between formats will result in error messages in the EViews command window, but will not halt the copying process. Upon completion, a message in the status line reports how many objects could not be copied.

Notes on Particular Formats

DRIPro Link

A DRIPro link is a special type of database which allows you to fetch data remotely over the internet from DRI's extensive collection of economic data. To use DRIPro access you must have a valid DRIPro account with DRI. There are special issues involved with using DRIPro links, which are discussed in detail in [“DRIPro Link” on page 137](#).

DRIBase Database

The DRIBase system is a client server system used by DRI to provide databases at the client site which can be kept current by remote updates. Customers can also use DRIBase as a means of storing their own databases in an Sybase or Microsoft SQL Server system.

DRIBase access is only available in the Enterprise Edition of EViews.

In order to access DRIBase databases, the TSRT library from DRI must already be installed on the client machine. This will normally be done by DRI as part of the DRIBase installation procedure. DRIBase access

When working with DRIBase databases, the **Server specification** field should be set to contain the DRIBase database prefix, while the **Database name** field should contain the DRIBase bank name, including the leading “@” where appropriate. Note that these fields, as well as the **Username** and **Password** fields may be case sensitive, so make sure to preserve the case of any information given to you.

A DRIBase database has slightly different handling of frequencies than most other databases supported by EViews. See [“Issues with DRI Frequencies” on page 145](#) for details. You should also read [“Dealing with Illegal Names” on page 144](#) for a discussion of how DRI names are automatically remapped by EViews.

For further information on DRIBase, please contact DRI directly.

FAME

The FAME format is a binary format written by FAME database products. FAME provides a variety of products and services for working with time series data.

FAME access is only available in the Enterprise Edition of EViews.

In order to access FAME databases, a valid installation of FAME must already be installed. EViews makes use of the FAME C HLI library, and will error unless the FAME .DLLs are correctly installed on the machine. EViews currently supports only version 8 of the FAME libraries.

A local FAME database can have any file extension, and EViews supports access to a FAME database with any name. However, because many commands in EViews use the file extension to automatically detect the file type, you will generally find it easier to work with FAME databases which have the default “.DB” extension.

EViews also allows access to FAME databases located on a FAME Database Server. When working with a FAME server, the **Server specification** should be given in the form

```
#port_number@ip_address
```

For example, the server specification for access to a FAME/Channel database might appear as:

```
#2552@channel.fame.com
```

Access to a server will require a valid username and password for that server.

Please contact FAME directly for further information about the FAME database system and other FAME products.

Haver

The Haver database format is a binary format used by Haver Analytics when distributing data.

Haver access is only available in the Enterprise Edition of EViews.

The main difference between Haver databases and other file formats supported by EViews, is that Haver databases are read-only. You cannot create your own database in Haver format, nor can you modify an existing database. EViews will error if you try to do so.

Please contact Haver Analytics directly for further information about Haver Analytics data products.

AREMOS TSD

The TSD format is a portable ASCII file format written by the AREMOS package. Although EViews already has some support for TSD files through the `tsdftech`, `tsdstore`, `tsdload` and `tsdsave` commands, working with the database directly gives you an intuitive graphical interface to the data, and allows you to move data directly in and out of an EViews database without having to move the data through a workfile (which may force the data to be converted to a single frequency).

GiveWin/PcGive

The GiveWin/PcGive format is a binary file format used by GiveWin, PcGive versions 7 and 8, and PcFiml.

There are two issues when working with GiveWin/PcGive files. The first is that EViews is case insensitive when working with object names, while GiveWin and PcGive are case sensitive. Because of this, if you intend to work with a file in both packages, you should avoid having two objects with names distinguished only by case. If your files do not follow this rule, EViews will only be able to read the last of the objects with the same name. Any early objects will be invisible.

The second issue concerns files with mixed frequency. The GiveWin/PcGive file format does support series of mixed frequency, and EViews will write to these files accordingly. However, GiveWin itself appears to only allow you to read series from one frequency at a time, and will ignore (with error messages) any series which do not conform to the chosen frequency. Consequently, depending on your application, you may prefer to store series of only one frequency per GiveWin/PcGive file.

RATS 4.x

The RATS 4.x format is a binary format used by RATS version 4 on all platforms.

The main issue to be aware of when working with RATS 4.x format files is that the “.RAT” extension is also used by RATS version 3 files. EViews will neither read from nor write to RATS files in this earlier format. If you try to use EViews to open one of these files, EViews will error, giving you a message that the file has a version number which is not supported.

To work with a RATS version 3 file in EViews, you will first have to use RATS to translate the file to the version 4 format. To convert a version 3 file to a version 4 file, simply load the file into RATS and modify it in some way. When you save the file, RATS will ask you whether you would like to translate the file into the new format. One simple way to modify the file without actually changing the data is to rename a series in the file to the name which it already has. For example, if we have a version 3 file called “OLDFILE.RAT”, we can convert to a version 4 by first opening the file for editing in RATS:

```
dedit oldfile.rat
```

then listing the series contained in the file:

```
catalog
```

then renaming one of the series (say “X”) to its existing name

```
rename x x
```

and finally saving the file

```
save
```

At this point, you will be prompted whether you would like to translate the file into the version 4 format.

See the RATS documentation for details.

RATS Portable

The RATS portable format is an ASCII format which can be read and written by RATS. It is generally slower to work with than RATS native format, but the files are human readable and can be modified using a text editor.

You can read the contents of a RATS portable file into memory in RATS with the following commands:

```
open data filename.tr1  
data(format=portable) start end list_of_series  
close data
```

To write what is currently in memory in RATS to a RATS portable file, use:

```
open copy filename.tr1  
copy(format=portable) start end list_of_series  
close copy
```

See the RATS documentation for details.

TSP Portable

The TSP portable format is an ASCII format which can be read and written by copies of TSP on all platforms. The file consists of a translation of a TSP native databank (which typically have the extension “.TLB”) into a TSP program which, when executed, will regenerate the databank on the new machine.

To create a TSP portable file from a TSP databank file, use the `DBCOPY` command from within TSP:

```
dbcopy databank_name
```

To translate a TSP portable file back into a TSP databank file, simply execute the TSP file as a TSP program.

Once the data is in TSP databank format, you can use the TSP command

```
in databank_name
```

to set the automatic search to use this databank and the TSP command

```
out databank_name
```

to save any series which are created or modified back to the databank.

See the TSP documentation for details.

Working with DRIPro Links

EViews has the ability to remotely access databases hosted by DRI. Subscribers to DRI data services can use these features to access data directly from within EViews.

Although the interface to remote databases is very similar to that of local databases, there are some differences due to the nature of the connection. There are also some issues specifically related to accessing DRI data. The following sections document these differences.

Enabling DRI Access

In order to access DRI data services, you will need to have an active account with DRI. If you are not an existing DRI customer but may be interested in becoming one, you should contact DRI for details (see the booklet included with EViews for contact information). DRI will often provide trial access to the service to interested customers.

Access to DRI data will not be possible unless you have already installed and configured the DRIPro server software. If you have not already done this, you should follow the instructions in the DRIPro booklet included with EViews. You should test that the DRIPro software is functioning correctly before attempting to use EViews to connect remotely to DRI. If you have difficulties with getting the DRI software to work, you should contact DRI directly for technical support.

Creating a Database Link

A remote DRI database is represented in EViews by a *database link*. A database link resembles a local database, consisting of a set of files on disk, but instead of containing the data itself, a database link contains information as to how to access the remote data. A database

link also contains a cache in which copies of recently retrieved objects are kept, which can substantially reduce the time taken to perform some database operations.

You can create a database link by following a similar procedure to that used to create a local database. Select **File/New/Database...** from the main menu, then select **DRIPro Link** in the field **Database/File Type**. The dialog should change appearance so that a number of extra fields are displayed. Enter the name you would like to give the new database link in **Cache name/path**:. You may wish to name the database link after the DRI bank to which it links.

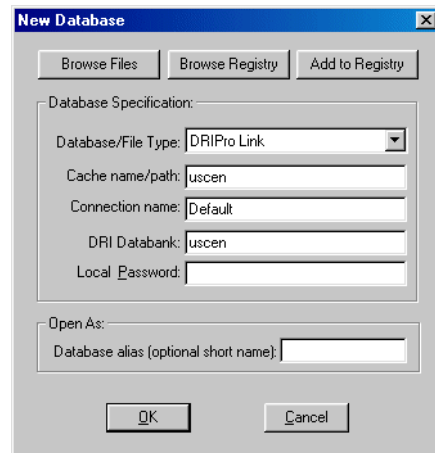
In the field **Connection name**: you should enter the name of the DRI connection you would like to use, as it appears in the **Connection Settings** box in the DRIPro configuration program. If you have only configured a single connection, and have not modified the connection name, the connection name will be DEFAULT, and this will be filled in automatically by EViews if you leave the field blank.

In the field **DRI Databank**: you should input the full name of the DRI bank to which you would like to connect, not including any leading @ sign. For example, to connect to the DRI U.S. Central database, you should enter “uscen”. (See the separate DRIPro booklet for a full list of DRI banks and their names). Each EViews database link can be associated with only one DRI databank, although you can create as many database links as you require.

The field **Local Password**: can be used to set a password that must be entered whenever you wish to use the database link. This should not be confused with your DRI username and password, which you must already have provided in the DRI configuration program. Accessing a database link which contains a local password will cause a dialog to appear which prompts the user to input the password. Access to the remote database is only provided if the remote password is valid. Leave this field blank if you do not want a password to be attached to the database link.

When you have finished filling in the dialog fields, click on the OK button. A new database will be created and a database window should appear on the screen.

The DRI database link window is very similar to a normal EViews database window. You should be able to perform basic query operations and simple fetching of series without any special instructions. Note, however, that it is not possible to modify a remote DRI database from within EViews, so operations which involve writing to the database have been



removed. There are a number of other complications related to dealing with DRI databases as described below.

Understanding the Cache

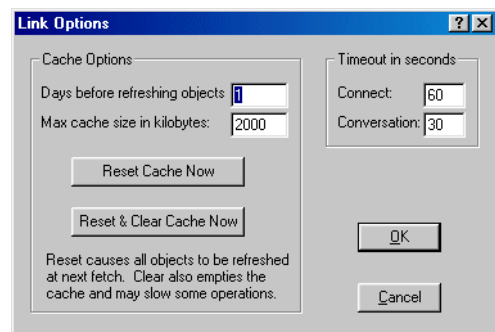
A database link includes a cache of recently fetched objects which is used to speed up certain operations on the database. In some circumstances, fetching an object from the database will simply retrieve a copy from the local cache, rather than fetching a fresh copy of the data from the remote site. Even if a fresh copy is retrieved, having a previous copy of the series in the cache can substantially speed up retrieval.

You can regulate the caching behavior of the database link in a number of different ways. The basic option which determines under what circumstances a new copy of the data should be fetched is the *days before refresh*. If you attempt to fetch an object from the database link, and the copy of the object currently in the cache was fetched more recently than the days before refresh value, then the object currently in the cache will be returned instead of a fresh copy being fetched. For example, if days before refresh is set to one, any object which has already been fetched today will be retrieved from the cache, while any object which has not yet been fetched today will be retrieved from the remote site. Similarly, if days before refresh is set to seven, then an object in the cache must be more than a week old before a new copy of the object will be fetched. If days before refresh is set to zero, then a new copy of the data is fetched every time it is used.

You can change the days before refresh setting by clicking on the **Procs** button at the top of the database link window, then choosing **Link Options...** from the pop-up menu. A dialog will appear:

The dialog contains a number of fields, one of which is labeled **Days before refreshing objects**. Type a new number in the field to change the value.

The same dialog also contains a button marked **Reset cache now**. This button can be used to modify the behavior documented above. Clicking on the button causes the cache to mark all objects in the cache as out of date, so that the next time each object is fetched, it is guaranteed that a fresh copy will be retrieved. This provides a simple way for you to be certain that the database link will not return any data fetched before a particular time.



The dialog also contains some options for managing the size of the cache. The field marked **Maximum cache size in kilobytes** can be used to set the maximum size that the

cache will be allowed to grow to on disk. If the cache grows above this size, a prompt will appear warning you that the cache has exceeded the limit and asking if you would like to compact the cache. Compacting is performed by deleting objects from oldest to newest until the cache size is reduced to less than three quarters of its maximum size. The cache is then packed to reclaim the empty space.

You can also completely clear the contents of the cache at any time by clicking on the button marked **Reset & Clear Cache Now**.

You can always examine the current contents of the database cache by clicking on the **Cache** button at the top of the database link window. This will display the names of all objects currently in the cache.

Configuring Link Options

The link options dialog also allows you to specify a number of timeout values. In most cases the default values will behave acceptably. If you believe you are having problems with EViews aborting the connection too early, or you would like to shorten the times so as to receive a timeout message sooner, then enter new values in the appropriate fields.

- **Connection timeout**—is the length of time, in seconds, that EViews will wait for a response when first connecting to DRI. Depending on the type of connection you are making to DRI, this can take a significant amount of time.
- **Conversation timeout**—is the length of time, in seconds, that EViews will wait for a response from DRI when carrying out a transaction after a connection has already been made.

The values are attached to a particular database link, and can be reset at any time.

Dealing with Illegal Names

DRI databanks contain a number of series with names which are not legal names for EViews objects. In particular, DRI names frequently contain the symbols “@”, “&” and “%”, none of which are legal characters in EViews object names. We have provided a number of features to allow you to work with these series within EViews.

Because the “@” symbol is so common in DRI names, while the underline symbol (which is a legal character in EViews) is unused, we have hard-coded the rule that all underlines in EViews are mapped into “@” symbols in DRI names when performing operations on an DRI database link. For example, if there is a series which DRI documentation suggests has the name JQIMET@UK, you should refer to this series inside EViews as JQIMET_UK. Note that when performing queries, EViews will automatically replace the “@” symbol by an underline in the object name before displaying the query results on the screen. Consequently, if you are fetching data by copying-and-pasting objects from a query window, you do not need to be aware of this translation.

For other illegal names, you should use the object aliasing features (see [“Object Aliases and Illegal Names” on page 131](#)) to map the names into legal EViews object names.

Issues with DRI Frequencies

DRI databases have a different structure than EViews databases. An EViews database can contain series with mixed frequencies. A DRI database can contain data of only a single frequency. In order that similar data may be grouped together, each DRI databank is actually composed of a series of separate databases, one for each frequency. When working with DRI data from within DRI software, you will often have to specify at exactly which frequency a particular series can be found. In some cases, a DRI databank may contain a series with the same name stored at several different frequencies.

Because this approach is inconsistent with the way that EViews’ own databases work, we have tried to create a simpler interface to DRI data where you do not need to keep track of the frequency of each series that you would like to fetch. Instead, you can simply fetch a series by name or by selecting it from the query window, and EViews will do whatever is necessary to find out the frequency for you.

An ambiguity can arise in doing this, where a series with the same name appears at a variety of different frequencies in the DRI databank. By default, EViews resolves this ambiguity by always fetching the *highest* frequency data available. EViews will then perform necessary frequency conversions using the standard rules for frequency conversion in EViews (see [“Frequency Conversion” on page 72](#)).

In many cases, this procedure will exactly replicate the results that would be obtained if the lower frequency data was fetched directly from DRI. In some cases (typically when the series in question is some sort of ratio or other expression of one or more series) the figures may not match up exactly. In this case, if you know that the DRI data exists at multiple frequencies and you are familiar with DRI frequency naming conventions, you can explicitly fetch a series from DRI at a particular frequency by using a modified form of the command line form of fetch. Simply add the DRI frequency in parentheses after the name of the series. For example, the command:

```
fetch x(Q) y(A)
```

will fetch the series X and Y from the current default database, reading the quarterly frequency copy of X and the annual frequency copy of Y. If you request a frequency at which the data is not available, you will receive an error message. You should consult DRI documentation for details on DRI frequencies.

Limitations of DRI Queries

Queries to DRI database links are more limited than those available for EViews databases. The following section documents the restrictions.

First, queries on DRI databases allow only a subset of the fields available in EViews databases to be selected. The fields supported are: name, type, freq, start, end, last_update and description.

Second, the only fields which can be used in “where” conditions in a query on a DRI database link are name and description. (EViews does not support queries by frequency because of the ambiguities arising from DRI frequencies noted above).

Each of these fields has only one operator, the “matches” operator, and operations on the two fields can only be joined together using the “and” operator.

The “matches” operator is also limited for queries on DRI databases, matching only a subset of the expressions available for EViews databases. In particular, the pattern expression in a query on an DRI database must either have the form

a or b or ... c

or the form

a and b and ... c

Mixing of “and” and “or” is not allowed, and the “not” operator is not supported.

Patterns, however, are allowed and follow the normal EViews rules where “?” denotes any single character and “*” denotes zero or more characters.

Sorting of results by field is not supported.

Dealing with Common Problems

As stated in the introduction, you must install and configure the DRI software before EViews will be able to connect to DRI. If you cannot connect to DRI using the DRIPro software, you should contact DRI directly for assistance.

Assuming that you have correctly configured your DRI connection, in most cases EViews will be able to recover adequately from unexpected problems which arise during a DRI session without user intervention. Sometimes this will require EViews to automatically disconnect then reconnect to DRI.

There are some circumstances in which EViews may have problems making a connection. In order to connect to DRI, EViews uses a program written by DRI called DRIPROSV. You can tell when this program is running by looking for the icon labeled “DRIPRO server” in the Windows taskbar. Because of problems that can arise with multiple connections, EViews will not attempt to use the program if it is already running. Instead, EViews will report an error message “DRI server software already running”. If there is another application which is using the connection to DRI, you can simply close down that program and the DRI server software should shut down automatically. If this is not the case, you may

have to close down the DRI server software manually. Simply click on the icon in the Windows taskbar with the right mouse button, then select **Close** from the pop-up menu.

You can also use this as a procedure for forcing the DRI connection to terminate. (For example to hang up the phone if you are using a direct modem rather than an Internet connection). Closing down the server software may cause EViews to report an error if it is currently carrying out a database transaction, but should otherwise be safe. EViews will restart the server software whenever it is needed.

Note that running other DRIPro software while EViews is using the DRI server software may cause EViews to behave unreliably.

Commands

To create a new EViews format database named USDB, type

```
dbcreate c:\evdata\usdb
```

To open the database DRIBASIC and make it the default database, type

```
dbopen dribasic
```

To open, if not already open, or to create a new database if one does not already exist, a database named US1, type

```
db us1
```

Note that this command also makes US1 the default database.

To make a backup copy US1_BAK of the entire database US1, type

```
dbcopy us1 us1_bak
```

To delete the entire US1 database, type

```
dbdelete us1
```

To store objects (named GDP and M1) in the default database, type

```
store gdp m1
```

To store the object named GDP into two different databases (named US1 and FINANCE1), type

```
store us1::gdp finance1::gdp
```

To store GDP into the default database and all objects whose name begins with the letter X into the database named US1, type

```
store(1) gdp us1::x*
```

The option 1 in parentheses instructs EViews to save space by storing all series in single rather than double precision.

To store two objects (GDP and M1) into individual .DB files, type

```
store(i) gdp c:\evdata\m1
```

To fetch two objects GDP and M1, searching all databases in the search order, type

```
fetch(d) gdp m1
```

To fetch two objects from individual .DB files, with frequency conversion by summation if necessary, type

```
fetch(i,c=s) gdp m1
```

To fetch all objects that have GDP in their names from two databases, with frequency conversion by averaging if necessary, type

```
fetch(c=a) us1::*gdp* dri::*gdp*
```

To make a backup copy of GDP within the US1 database, type

```
copy us1::gdp us1::gdp_bak
```

To copy all objects in the US1 database to another existing database WORLD1, type

```
copy us1::* world1::
```

Part II. Basic Data Analysis

The following chapters describe the EViews objects that you will use to perform basic data analysis.

- [Chapter 7, “Series”, beginning on page 151](#) describes the series object. Series are the basic unit of data in EViews and are the basis for all univariate analysis. This chapter documents the basic graphing and data analysis features associated with series.
- [Chapter 8, “Groups”, on page 199](#) documents the group object. Groups are collections of series which form the basis for a variety of multivariate graphing and data analyses.
- [Chapter 9, “Statistical Graphs Using Series and Groups”, on page 225](#) provides detailed documentation for exploratory data analysis using distribution graphs, kernel density, and scatterplot fit graphs.
- [Chapter 10, “Graphs, Tables, and Text Objects”, beginning on page 243](#) describes the creation and customization of tables and graphs.

Chapter 7. Series

EViews provides various statistical graphs, descriptive statistics, and procedures as *views* and *procedures* of a series. Once you have read or generated data into series objects using any of the methods described in [Chapter 4, “Basic Data Handling”](#), [Chapter 5, “Working with Data”](#), and [Chapter 6, “EViews Databases”](#), you are ready to perform statistical and graphical analysis using the data contained in the series.

Series views compute various statistics for a single series and display these statistics in various forms such as spreadsheets, tables, and graphs. The views range from a simple line graph, to kernel density estimators. Series procedures create new series from the data in existing series. These procedures include various seasonal adjustment methods, exponential smoothing methods, and the Hodrick-Prescott filter.

Methods which involve more than one series are described in [Chapter 8, “Groups”](#), on page 199, which outlines views and procedures for group objects.

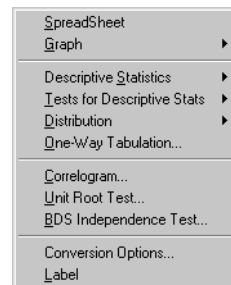
To access the views and procedures for series, open the series window by double clicking on the series name in the workfile, or by typing `show` followed by the name of the series in the command window.

Series Views

The series view drop-down menu is divided into four blocks. The first block lists views of the underlying data in the series. The second and third blocks provide access to general statistics; the views in the third block are mainly for time series. The fourth block contains the label view and allows you to assign default frequency conversion methods.

Spreadsheet and Graph Views

- **Spreadsheet** displays the raw data in the series in spreadsheet format.
- **Line Graph** plots the series against the date/observation number. This view is most useful for time series. See [Chapter 10, “Graphs, Tables, and Text Objects”](#), beginning on page 243 for a discussion of techniques for modifying and customizing the graphical display.
- **Bar Graph** plots the bar graph of the series. This view is useful for plotting series from a small data set that takes only a few distinct values.



- **Spike Graph** plots a spike graph of the series. The spike graph depicts values of the series as vertical spikes from the origin.
- **Seasonal Stacked Line/Seasonal Split Line.** These views plot the series reordered by season. The seasonal line graph view is currently available only for quarterly and monthly frequency workfiles.

The stacked view reorders the series into seasonal groups where the first season observations are ordered by year, and then followed by the second season observations, and so on. Also depicted are the horizontal lines identifying the mean of the series in each season.

The split view plots the line graph for each season on an annual horizontal axis.

Descriptive Statistics

This view displays various summary statistics for the series.

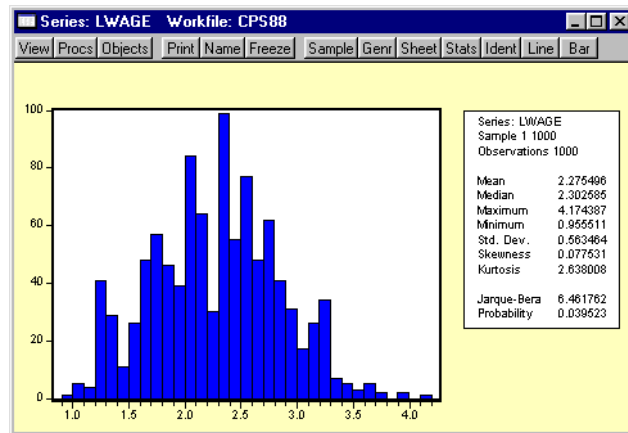
Histogram and Stats

This view displays the frequency distribution of your series in a histogram. The histogram divides the series range (the distance between the maximum and minimum values) into a number of equal length intervals or bins and displays a count of the number of observations that fall into each bin.

A complement of standard descriptive statistics are displayed along with the histogram. All of the statistics are calculated using observations in the current sample.

- **Mean** is the average value of the series, obtained by adding up the series and dividing by the number of observations.
- **Median** is the middle value (or average of the two middle values) of the series when the values are ordered from the smallest to the largest. The median is a robust measure of the center of the distribution that is less sensitive to outliers than the mean.
- **Max** and **Min** are the maximum and minimum values of the series in the current sample.

Histogram and Stats
Stats Table
Stats by Classification...



- **Std. Dev.** (standard deviation) is a measure of dispersion or spread in the series. The standard deviation is given by:

$$s = \sqrt{\left(\sum_{i=1}^N (y_i - \bar{y})^2 \right) / (N - 1)} \quad (7.1)$$

where N is the number of observations in the current sample and \bar{y} is the mean of the series.

- **Skewness** is a measure of asymmetry of the distribution of the series around its mean. Skewness is computed as:

$$S = \frac{1}{N} \sum_{i=1}^N \left(\frac{y_i - \bar{y}}{\hat{\sigma}} \right)^3 \quad (7.2)$$

where $\hat{\sigma}$ is an estimator for the standard deviation that is based on the biased estimator for the variance ($\hat{\sigma} = s\sqrt{(N-1)/N}$). The skewness of a symmetric distribution, such as the normal distribution, is zero. Positive skewness means that the distribution has a long right tail and negative skewness implies that the distribution has a long left tail.

- **Kurtosis** measures the peakedness or flatness of the distribution of the series. Kurtosis is computed as

$$K = \frac{1}{N} \sum_{i=1}^N \left(\frac{y_i - \bar{y}}{\hat{\sigma}} \right)^4 \quad (7.3)$$

where $\hat{\sigma}$ is again based on the biased estimator for the variance. The kurtosis of the normal distribution is 3. If the kurtosis exceeds 3, the distribution is peaked (leptokurtic) relative to the normal; if the kurtosis is less than 3, the distribution is flat (platykurtic) relative to the normal.

- **Jarque-Bera** is a test statistic for testing whether the series is normally distributed. The test statistic measures the difference of the skewness and kurtosis of the series with those from the normal distribution. The statistic is computed as:

$$\text{Jarque-Bera} = \frac{N - k}{6} \left(S^2 + \frac{(K - 3)^2}{4} \right) \quad (7.4)$$

where S is the skewness, K is the kurtosis, and k represents the number of estimated coefficients used to create the series.

Under the null hypothesis of a normal distribution, the Jarque-Bera statistic is distributed as χ^2 with 2 degrees of freedom. The reported Probability is the probability that a Jarque-Bera statistic exceeds (in absolute value) the observed value under the null hypothesis—a small probability value leads to the rejection of the null hypothesis.

sis of a normal distribution. For the LWAGE series displayed above, we reject the hypothesis of normal distribution at the 5% level but not at the 1% significance level.

Stats Table

Displays slightly more information than the Histogram/Stats view, with the numbers displayed in tabular form.

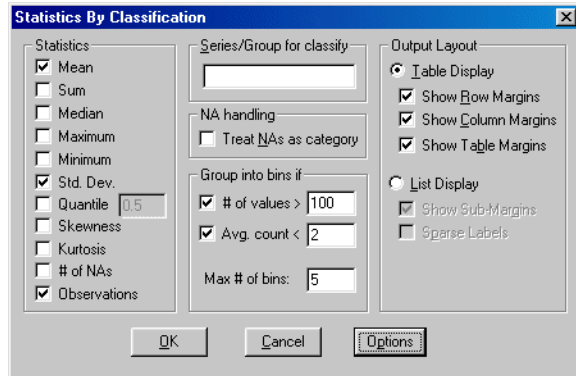
Stats by Classification

This view allows you to compute the descriptive statistics of a series for various subgroups of your sample. If you select **View/Descriptive Statistics/Stats by Classification...** a Statistics by Classification dialog box appears:

The **Statistics** option at the left allows you to choose the statistics you wish to compute.

In the **Series/Group for Classify** field enter series or group names that define your subgroups. You must type at least one name. Descriptive statistics will be calculated for each unique value of the classification series unless binning is selected. You may type more than one series or group

name; separate each name by a space. The quantile statistic requires an additional argument (a number between 0 and 1) corresponding to the desired quantile value. Click on the options button to choose between various methods of computing the quantiles. See [“CDF-Survivor-Quantile” on page 225](#) for details.



By default, EViews excludes observations which have missing values for any of the classification series. To treat NA values as a valid subgroup, select the **NA handling** option.

The **Layout** option allows you to control the display of the statistics. Table layout arrays the statistics in cells of two-way tables. The list form displays the statistics in a single line for each classification group.

The **Table** and **List** options are only relevant if you use more than one series as a classifier.

The **Sparse Labels** option suppresses repeating labels in list mode to make the display less cluttered.

The **Row Margins**, **Column Margins**, and **Table Margins** instruct EViews to compute statistics for aggregates of your subgroups. For example, if you classify your sample on the basis of gender and age, EViews will compute the statistics for each gender/age combination. If you elect to compute the marginal statistics, EViews will also compute statistics corresponding to each gender, and each age subgroup.

A classification may result in a large number of distinct values with very small cell sizes. By default, EViews automatically groups observations to maintain moderate cell sizes and numbers of categories. **Group into Bins** provides you with control over this process.

Setting the **# of values** option bins tell EViews to group data if the classifier series takes more than the specified number of distinct values.

The **Avg. count** option bins the series if the average count for each distinct value of the classifier series is less than the specified number.

The **Max # of bins** specifies the maximum number of subgroups to bin the series. Note that this number only provides you with approximate control over the number of bins.

The default setting is to bin the series into 5 subgroups if either the series takes more than 100 distinct values or if the average count is less than 2. If you do not want to bin the series, unmark both options.

For example, consider the following stats by classification view in table form:

Descriptive Statistics for LWAGE
Categorized by values of MARRIED and UNION
Date: 10/15/97 Time: 01:11
Sample: 1 1000
Included observations: 1000

Mean Median Std. Dev. Obs.	UNION		
	0	1	All
0	1.993829 1.906575 0.574636 305	2.387019 2.409131 0.395838 54	2.052972 2.014903 0.568689 359
MARRIED 1	2.368924 2.327278 0.557405 479	2.492371 2.525729 0.380441 162	2.400123 2.397895 0.520910 641
All	2.223001 2.197225 0.592757 784	2.466033 2.500525 0.386134 216	2.275496 2.302585 0.563464 1000

The header indicates that the table cells are categorized by two series MARRIED and UNION. These two series are dummy variables that take only two values and no binning was made. If the series were binned, intervals rather than a number would be displayed in the margins.

The upper left cell of the table indicates the reported statistics in each cell; in this case the median and the number of observations are reported in each cell. The row and column labeled **All** correspond to the **Row Margin** and **Column Margin** options described above.

Here is the same view in list form with sparse labels:

Descriptive Statistics for LWAGE
Categorized by values of MARRIED and UNION
Date: 10/15/97 Time: 01:08
Sample: 1 1000
Included observations: 1000

UNION	MARRIED	Mean	Median	Std. Dev.	Obs.
0	0	1.993829	1.906575	0.574636	305
	1	2.368924	2.327278	0.557405	479
	All	2.223001	2.197225	0.592757	784
1	0	2.387019	2.409131	0.395838	54
	1	2.492371	2.525729	0.380441	162
	All	2.466033	2.500525	0.386134	216
All	0	2.052972	2.014903	0.568689	359
	1	2.400123	2.397895	0.520910	641
	All	2.275496	2.302585	0.563464	1000

Tests for Descriptive Stats

Simple Hypothesis Tests

This view carries out simple hypothesis tests regarding the mean, median, and the variance of the series. These are all single sample tests; see [“Equality Tests by Classification” on page 159](#) for a description of two sample tests. If you select **View/Tests for Descriptive Stats/Simple Hypothesis Tests**, the Series Distribution Tests dialog box will be displayed.

Mean Test

Carries out the test of the null hypothesis that the mean μ of the series X is equal to a specified value m against the two-sided alternative that it is not equal to m :

$$\begin{aligned} H_0: \mu &= m \\ H_1: \mu &\neq m. \end{aligned} \quad (7.5)$$

If you do not specify the standard deviation of X , EViews reports a t -statistic computed as:

$$t = \frac{\bar{X} - m}{s / \sqrt{N}} \quad (7.6)$$

where \bar{X} is the sample mean of X , s is the unbiased sample standard deviation, and N is the number of observations of X . If X is normally distributed, under the null hypothesis the t -statistic follows a t -distribution with $N - 1$ degrees of freedom.

If you specify a value for the standard deviation of X , EViews also reports a z -statistic

$$z = \frac{\bar{X} - m}{\sigma / \sqrt{N}} \quad (7.7)$$

where σ is the specified standard deviation of X . If X is normally distributed with standard deviation σ , under the null hypothesis, the z -statistic has a standard normal distribution.

To carry out the mean test, type in the value of the mean under the null hypothesis in the edit field next to **Mean**. If you want to compute the z -statistic conditional on a known standard deviation, also type in a value for the standard deviation in the right edit field. You can type in any number or standard EViews expression in the edit fields.

Hypothesis Testing for LWAGE		
Date: 10/15/97 Time: 01:14		
Sample: 1 1000		
<u>Included observations: 1000</u>		
<u>Test of Hypothesis: Mean = 2</u>		
Sample Mean = 2.275496		
Sample Std. Dev. = 0.563464		
<u>Method</u>	<u>Value</u>	<u>Probability</u>
<u>t-statistic</u>	15.46139	0.00000

The reported probability value is the p -value, or marginal significance level, against a two-sided alternative. If this probability value is less than the size of the test, say 0.05, we reject the null hypothesis. Here, we strongly reject the null hypothesis for the two-sided test of equality. The probability value for a one-sided alternative is one half the p -value of the two-sided test.

Variance Test

Carries out the test of the null hypothesis that the variance of a series X is equal to a specified value σ^2 against the two-sided alternative that it is not equal to σ^2 :

$$\begin{aligned} H_0: \text{var}(x) &= \sigma^2 \\ H_1: \text{var}(x) &\neq \sigma^2. \end{aligned} \quad (7.8)$$

EViews reports a χ^2 statistic computed as:

$$\chi^2 = \frac{(N-1)s^2}{\sigma^2} \quad (7.9)$$

where N is the number of observations, s is the sample standard deviation, and \bar{X} is the sample mean of X . Under the null hypothesis and the assumption that X is normally distributed, the statistic follows a χ^2 distribution with $N - 1$ degrees of freedom. The probability value is computed as $\min(p, 1 - p)$, where p is the probability of observing a χ^2 -statistic as large as the one actually observed under the null hypothesis.

To carry out the variance test, type in the value of the variance under the null hypothesis in the field box next to **Variance**. You can type in any *positive* number or expression in the field.

Median Test

Carries out the test of the null hypothesis that the median of a series X is equal to a specified value m against the two-sided alternative that it is not equal to m :

$$\begin{aligned} H_0: \text{med}(x) &= m \\ H_1: \text{med}(x) &\neq m. \end{aligned} \quad (7.10)$$

EViews reports three rank-based, nonparametric test statistics. The principal references for this material are Conover (1980) and Sheskin (1997).

- **Binomial sign test.** This test is based on the idea that if the sample is drawn randomly from a binomial distribution, the sample proportion above and below the true median should be one-half. Note that EViews reports two-sided p -values for both the sign test and the large sample normal approximation (with continuity correction).
- **Wilcoxon signed ranks test.** Suppose that we compute the absolute value of the difference between each observation and the mean, and then rank these observations from high to low. The Wilcoxon test is based on the idea that the sum of the ranks for the samples above and below the median should be similar. EViews reports a p -value for the asymptotic normal approximation to the Wilcoxon T -statistic (correcting for both continuity and ties). See Sheskin (1997, pp. 82–94) and Conover (1980, p. 284).
- **Van der Waerden (normal scores) test.** This test is based on the same general idea as the Wilcoxon test, but is based on smoothed ranks. The signed ranks are smoothed by converting them to quantiles of the normal distribution (normal scores). EViews reports the two-sided p -value for the asymptotic normal test described by Conover (1980).

To carry out the median test, type in the value of the median under the null hypothesis in the edit box next to **Median**. You can type any numeric expression in the edit field.

Hypothesis Testing for LWAGE
 Date: 10/14/97 Time: 23:23
 Sample: 1 1000
Included observations: 1000

Test of Hypothesis: Median = 2.25

Sample Median = 2.302585

Method	Value	Probability
Sign (exact binomial)	532	0.046291
Sign (normal approximation)	1.992235	0.046345
Wilcoxon signed rank	1.134568	0.256556
van der Waerden (normal scores)	1.345613	0.178427

Median Test Summary

Category	Count	Mean Rank
Obs > 2.25	532	489.877820
Obs < 2.25	468	512.574786
Obs = 2.25	0	
Total	1000	

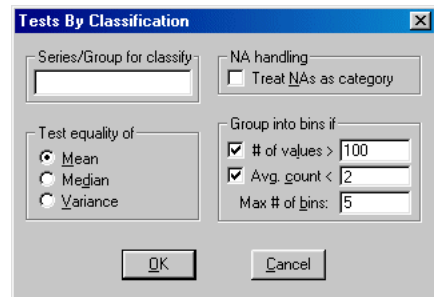
Equality Tests by Classification

This view allows you to test equality of the means, medians, and variances across subsamples (or subgroups) of a single series. For example, you can test whether mean income is the same for males and females, or whether the variance of education is related to race. The tests assume that the subsamples are independent.

For single sample tests, see the discussion of “[Simple Hypothesis Tests](#)” on page 156, above. For tests of equality across different series, see “[Equality Tests by Classification](#)” on page 159.

Select **View/Tests for Descriptive Stats/Equality Tests by Classification....** The Tests by Classification dialog box will appear.

First, select whether you wish to test the mean, the median or the variance. Specify the subgroups, the NA handling, and the grouping options as described in “[Stats by Classification](#)” beginning on page 154.



Mean Equality Test

This test is based on a single-factor, between-subjects, analysis of variance (ANOVA). The basic idea is that if the subgroups have the same mean, then the variability between the

sample means (between groups) should be the same as the variability within any subgroup (within group).

Denote the i -th observation in group g as $x_{g,i}$, where $i = 1, \dots, n_g$ for groups $g = 1, 2, \dots, G$. The between and within sums of squares are defined as

$$SS_B = \sum_{g=1}^G n_g (\bar{x}_g - \bar{x})^2 \quad (7.11)$$

$$SS_W = \sum_{g=1}^G \sum_{i=1}^{n_g} (x_{ig} - \bar{x}_g)^2 \quad (7.12)$$

where \bar{x}_g is the sample mean within group g and \bar{x} is the overall sample mean. The F -statistic for the equality of means is computed as

$$F = \frac{SS_B / (G - 1)}{SS_W / (N - G)} \quad (7.13)$$

where N is the total number of observations. The F -statistic has an F -distribution with $G - 1$ numerator degrees of freedom and $N - G$ denominator degrees of freedom under the null hypothesis of independent and identical normal distribution, with equal means and variances in each subgroup.

For tests with only two subgroups ($G = 2$), EViews also reports the t -statistic, which is simply the square root of the F -statistic with one numerator degree of freedom:

Test for Equality of Means of LWAGE
 Categorized by values of MARRIED and UNION
 Date: 10/14/97 Time: 12:28
 Sample: 1 1000
 Included observations: 1000

Method	df	Value	Probability
Anova F-statistic	(3, 996)	43.40185	0.0000

Analysis of Variance

Source of Variation	df	Sum of Sq.	Mean Sq.
Between	3	36.66990	12.22330
Within	996	280.5043	0.281631
Total	999	317.1742	0.317492

Category Statistics

UNION	MARRIED	Count	Mean	Std. Dev.	Std. Err. of Mean
0	0	305	1.993829	0.574636	0.032904
0	1	479	2.368924	0.557405	0.025468
1	0	54	2.387019	0.395838	0.053867
1	1	162	2.492371	0.380441	0.029890
	All	1000	2.275496	0.563464	0.017818

The analysis of variance table shows the decomposition of the total sum of squares into the between and within sum of squares, where

$$\text{Mean Sq.} = \text{Sum of Sq.}/\text{df}$$

The F -statistic is the ratio

$$F = \text{Between Mean Sq.}/\text{Within Mean Sq.}$$

Median (Distribution) Equality Tests

EViews computes various rank-based nonparametric tests of the hypothesis that the subgroups have the same general distribution, against the alternative that at least one subgroup has a different distribution.

In the two group setting, the null hypothesis is that the two subgroups are independent samples from the same general distribution. The alternative hypothesis may loosely be defined as “the values of (the first group) tend to differ from the values (of the second group)” (Conover 1980, p. 281). See also Bergmann, Ludbrook and Spooren (2000) for a more precise analysis of the issues involved.

We note that the “median” category in which we place these tests is somewhat misleading since the tests focus more generally on the equality of various statistics computed across subgroups. For example, the Wilcoxon test examines the comparability of mean ranks across subgroups. The categorization reflects common usage for these tests and various textbook definitions. The tests may, of course, have power against median differences.

- **Wilcoxon signed ranks test.** This test is computed when there are two subgroups. The test is identical to the Wilcoxon test outlined in the description of median tests on [page 158](#) but the division of the series into two groups is based upon the values of the classification variable instead of the value of the observation relative to the median.
- **Chi-square test for the median.** This is a rank-based ANOVA test based on the comparison of the number of observations above and below the overall median in each subgroup. This test is sometimes referred to as the *median test* (Conover, 1980).

Under the null hypothesis, the median chi-square statistic is asymptotically distributed as a χ^2 with $G - 1$ degrees of freedom. EViews also reports Yates’ continuity corrected statistic. You should note that the use of this correction is controversial (Sheskin, 1997, p. 218).

- **Kruskal-Wallis one-way ANOVA by ranks.** This is a generalization of the Mann-Whitney test to more than two subgroups. The idea behind the Mann-Whitney test is to rank the series from smallest value (rank 1) to largest, and to compare the sum of the ranks from subgroup 1 to the sum of the ranks from subgroup 2. If the groups have the same median, the values should be similar.

EViews reports the asymptotic normal approximation to the U-statistic (with continuity and tie correction) and the p -values for a two-sided test. For details, see Sheskin (1997) The test is based on a one-way analysis of variance using only ranks of the data. EViews reports the χ^2 chi-square approximation to the Kruskal-Wallis test statistic (with tie correction). Under the null hypothesis, this statistic is approximately distributed as a χ^2 with $G - 1$ degrees of freedom (see Sheskin, 1997).

- **van der Waerden (normal scores) test.** This test is analogous to the Kruskal-Wallis test, except that we smooth the ranks by converting them into normal quantiles (Conover, 1980). EViews reports a statistic which is approximately distributed as a χ^2 with $G - 1$ degrees of freedom under the null hypothesis. See the discussion of the Wilcoxon test for additional details on interpreting the test more generally as a test of a common subgroup distributions.

In addition to the test statistics and p -values, EViews reports values for the components of the test statistics for each subgroup of the sample. For example, the column labeled **Mean Score** contains the mean values of the van der Waerden scores (the smoothed ranks) for each subgroup.

Variance Equality Tests

Tests the null hypothesis that the variances in all G subgroups are equal against the alternative that at least one subgroup has a different variance. See Conover, *et al.* (1981) for a general discussion of variance testing.

- **F -test.** This test statistic is reported only for tests with two subgroups ($G = 2$). Compute the variance for each subgroup and denote the subgroup with the larger variance as L and the subgroup with the smaller variance as S . Then the F -statistic is given by

$$F = s_L^2 / s_S^2 \quad (7.14)$$

where s_g^2 is the variance in subgroup g . This F -statistic has an F -distribution with $n_L - 1$ numerator degrees of freedom and $n_S - 1$ denominator degrees of freedom under the null hypothesis of equal variance and independent normal samples

- **Siegel-Tukey test.** This test statistic is reported only for tests with two subgroups ($G = 2$). The test assumes the two subgroups are independent and have equal median. The test statistic is computed using the same steps as the Kruskal-Wallis test described above for the median equality tests on [page 162](#), with a different assignment of ranks. For the Siegel-Tukey test, first rank all observations from lowest to highest. Assign rank 1 to the lowest value. Then assign rank 2 to the *highest* value and rank 3 to the second highest value. Assign rank 4 to the second lowest value and rank 5 to the third lowest value, and so on. In other words, the ranking for the Siegel-Tukey test alternates from the lowest to the highest value for every other rank. EViews reports the normal approximation to the Siegel-Tukey statistic with a continuity correction (Sheskin, 1997, pp. 196–207).
- **Bartlett test.** This test compares the logarithm of the weighted average variance with the weighted sum of the logarithms of the variances. Under the joint null hypothesis that the subgroup variances are equal and that the sample is normally distributed, the test statistic is approximately distributed as a χ^2 with $G - 1$ degrees of freedom. Note, however, that the joint hypothesis implies that this test is sensitive to departures from normality. EViews reports the adjusted Bartlett statistic. For details, see Sokal and Rohlf (1995) and Judge, *et al.* (1985).
- **Levene test.** This test is based on an analysis of variance (ANOVA) of the absolute difference from the mean. The F -statistic for the Levene test has an approximate F -distribution with $G - 1$ numerator degrees of freedom and $N - G$ denominator degrees of freedom under the null hypothesis of equal variances in each subgroup (Levene, 1960).
- **Brown-Forsythe (modified Levene) test.** This is a modification of the Levene test in which we replace the absolute *mean* difference with the absolute *median* difference

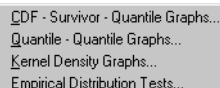
and appears to be a superior test in terms of robustness and power (Conover, *et al.* (1981), Brown and Forsythe (1974a, 1974b), Neter, *et al.* (1996)).

Distribution Graphs

These views display various graphs that characterize the empirical distribution of the series. A detailed description of these views may be found in [Chapter 9, “Statistical Graphs Using Series and Groups”](#), on page 225.

CDF-Survivor-Quantile

This view plots the empirical cumulative distribution, survivor, and quantile functions of the series together with plus/minus two standard error bands. EViews provides a number of alternative methods for performing these computations.



Quantile-Quantile

The quantile-quantile (QQ)-plot is a simple yet powerful tool for comparing two distributions. This view plots the quantiles of the chosen series against the quantiles of another series or a theoretical distribution.

Kernel Density

This view plots the kernel density estimate of the distribution of the series. The simplest nonparametric density estimate of a distribution of a series is the histogram. The histogram, however, is sensitive to the choice of origin and is not continuous. The kernel density estimator replaces the “boxes” in a histogram by “bumps” that are smooth (Silverman 1986). Smoothing is done by putting less weight on observations that are further from the point being evaluated.

EViews provides a number of kernel choices as well as control over bandwidth selection and computational method.

Empirical Distribution Tests

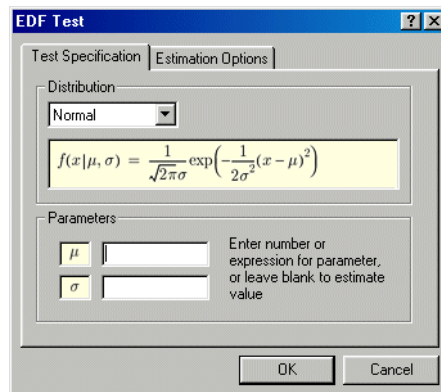
EViews provides built-in Kolmogorov-Smirnov, Lilliefors, Cramer-von Mises, Anderson-Darling, and Watson empirical distribution tests. These tests are based on the comparison between the empirical distribution and the specified theoretical distribution function. For a general description of empirical distribution function testing, see D’Agostino and Stephens (1986)

You can test whether your series is normally distributed, or whether it comes from, among others, an exponential, extreme value, logistic, chi-square, Weibull, or gamma distribution. You may provide parameters for the distribution, or EViews will estimate the parameters for you.

To carry out the test, simply double click on the series and select **View/Distribution/Empirical Distribution Tests...** from the series window.

There are two tabs in the dialog. The **Test Specification** tab allows you to specify the parametric distribution against which you want to test the empirical distribution of the series. Simply select the distribution of interest from the drop-down menu. The small display window will change to show you the parameterization of the specified distribution.

You can specify the values of any known parameters in the edit field or fields. If you leave any field blank, EViews will estimate the corresponding parameter using the data contained in the series.



The **Estimation Options** tab provides control over any iterative estimation that is required. You should not need to use this tab unless the output indicates failure in the estimation process. Most of the options in this tab should be self-explanatory. If you select **User-specified starting values**, EViews will take the starting values from the C coefficient vector.

It is worth noting that some distributions have positive probability on a restricted domain. If the series data take values outside this domain, EViews will report an out-of-range error. Similarly, some of the distributions have restrictions on domain of the parameter values. If you specify a parameter value that does not satisfy this restriction, EViews will report an error message.

The output from this view consists of two parts. The first part displays the test statistics and associated probability values.

Empirical Distribution Test for DPOW2
 Hypothesis: Normal
 Date: 01/09/01 Time: 09:11
 Sample: 1 1000
 Included observations: 1000

Method	Value	Adj. Value	Probability
Lilliefors (D)	0.294098	NA	0.0000
Cramer-von Mises (W2)	27.89617	27.91012	0.0000
Watson (U2)	25.31586	25.32852	0.0000
Anderson-Darling (A2)	143.6455	143.7536	0.0000

Here we show the output from a test for normality where both the mean and the variance are estimated from the series data. The first column, “Value” reports the asymptotic test

statistics while the second column “Adj. Value” reports test statistics that have a finite sample correction or adjusted for parameter uncertainty (in case the parameters are estimated). The third column reports p -value for the adjusted statistics.

All of the reported EViews p -values will account for the fact that parameters in the distribution have been estimated. In cases where estimation of parameters is involved, the distributions of the goodness-of-fit statistics are non-standard and distribution dependent, so that EViews may report a subset of tests and/or only a range of p -value. In this case, for example, EViews reports the Lilliefors test statistic instead of the Kolmogorov statistic since the parameters of the normal have been estimated. Details on the computation of the test statistics and the associated p -values may be found in Anderson and Darling (1952, 1954), Lewis (1961), Durbin (1970), Dallal and Wilkinson (1986), Davis and Stephens (1989), Csörgö and Faraway (1996) and Stephens (1986).

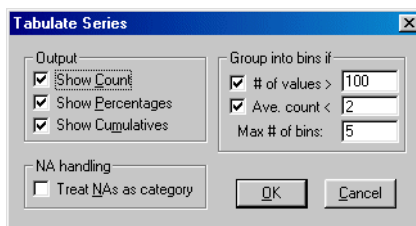
Method: Maximum Likelihood - d.f. corrected (Exact Solution)				
Parameter	Value	Std. Error	z-Statistic	Prob.
MU	0.142836	0.015703	9.096128	0.0000
SIGMA	0.496570	0.011109	44.69899	0.0000
Log likelihood	-718.4084	Mean dependent var.	0.142836	
No. of Coefficients	2	S.D. dependent var.	0.496570	

The second part of the output table displays the parameter values used to compute the theoretical distribution function. Any parameters that are specified to estimate are estimated by maximum likelihood (for the normal distribution, the estimate of the standard deviation is degree-of-freedom corrected if the mean is not specified *a priori*). For parameters that do not have a closed form analytic solution, the likelihood function is maximized using analytic first and second derivatives. These estimated parameters are reported with a standard error and p -value based on the asymptotic normal distribution.

One-Way Tabulation

This view tabulates the series in ascending order, optionally displaying the counts, percentage counts, and cumulative counts. When you select **View/One-Way Tabulation...** the Tabulate Series dialog box will be displayed.

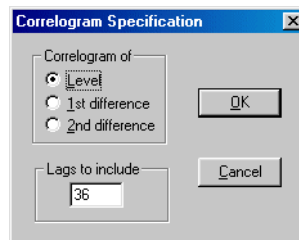
The Output options control which statistics to display in the table. You should specify the NA handling and the grouping options as described above in the discussion of “Stats by Classification” on page 154.



Cross-tabulation (n -way tabulation) is also available as a group view. See “N-Way Tabulation” on page 216 for details.

Correlogram

This view displays the autocorrelation and partial autocorrelation functions up to the specified order of lags. These functions characterize the pattern of temporal dependence in the series and typically make sense only for time series data. When you select **View/Correlogram...** the Correlogram Specification dialog box appears

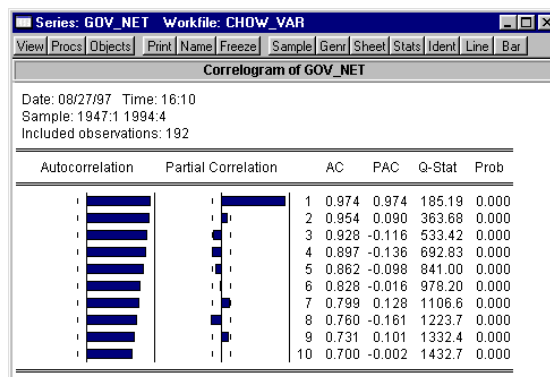


You may choose to plot the correlogram of the raw series (level) x , the first difference $d(x) = x - x(-1)$, or the second difference

$$d(x) - d(x(-1)) = x - 2x(-1) + x(-2)$$

of the series.

You should also specify the highest order of lag to display the correlogram; type in a positive integer in the field box. The series view displays the correlogram and associated statistics:



Autocorrelations (AC)

The autocorrelation of a series Y at lag k is estimated by:

$$\tau_k = \frac{\sum_{t=k+1}^T (Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{t=1}^T (Y_t - \bar{Y})^2} \quad (7.15)$$

where \bar{Y} is the sample mean of Y . This is the correlation coefficient for values of the series k periods apart. If τ_1 is nonzero, it means that the series is first order serially correlated. If τ_k dies off more or less geometrically with increasing lag k , it is a sign that the series obeys a low-order autoregressive (AR) process. If τ_k drops to zero after a small number of lags, it is a sign that the series obeys a low-order moving-average (MA) process. See “[Serial Correlation Theory](#)” on page 303, for a more complete description of AR and MA processes.

Note that the autocorrelations estimated by EViews differ slightly from theoretical descriptions of the estimator:

$$\tau_k = \frac{\sum_{t=k+1}^T ((Y_t - \bar{Y})(Y_{t-k} - \bar{Y})) / (T - K)}{\sum_{t=1}^T (Y_t - \bar{Y})^2 / T} \quad (7.16)$$

where $\bar{Y}_{t-k} = \sum Y_{t-k} / (T - k)$. The difference arises since, for computational simplicity, EViews employs the same overall sample mean \bar{Y} as the mean of both Y_t and Y_{t-k} . While both formulations are consistent estimators, the EViews formulation biases the result toward zero in finite samples.

The dotted lines in the plots of the autocorrelations are the approximate two standard error bounds computed as $\pm 2 / (\sqrt{T})$. If the autocorrelation is within these bounds, it is not significantly different from zero at (approximately) the 5% significance level.

Partial Autocorrelations (PAC)

The partial autocorrelation at lag k is the regression coefficient on Y_{t-k} when Y_t is regressed on a constant, Y_{t-1}, \dots, Y_{t-k} . This is a *partial* correlation since it measures the correlation of Y values that are k periods apart after removing the correlation from the intervening lags. If the pattern of autocorrelation is one that can be captured by an autoregression of order less than k , then the partial autocorrelation at lag k will be close to zero.

The PAC of a pure autoregressive process of order p , $AR(p)$, cuts off at lag p , while the PAC of a pure moving average (MA) process asymptotes gradually to zero.

EViews estimates the partial autocorrelation at lag k recursively by

$$\phi_k = \begin{cases} \tau_1 & \text{for } k = 1 \\ \frac{\tau_k - \sum_{j=1}^{k-1} \phi_{k-1,j} \tau_{k-j}}{1 - \sum_{j=1}^{k-1} \phi_{k-1,j} \tau_{k-j}} & \text{for } k > 1 \end{cases} \quad (7.17)$$

where τ_k is the estimated autocorrelation at lag k and

$$\phi_{k,j} = \phi_{k-1,j} - \phi_k \phi_{k-1,k-j}. \quad (7.18)$$

This is a consistent approximation of the partial autocorrelation. The algorithm is described in Box and Jenkins (1976, Part V, Description of computer programs). To obtain a more precise estimate of ϕ , simply run the regression

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \dots + \beta_{k-1} Y_{t-(k-1)} + \phi_k Y_{t-k} + e_t \quad (7.19)$$

where e_t is a residual. The dotted lines in the plots of the partial autocorrelations are the approximate two standard error bounds computed as $\pm 2/(\sqrt{T})$. If the partial autocorrelation is within these bounds, it is not significantly different from zero at (approximately) the 5% significance level.

Q-Statistics

The last two columns reported in the correlogram are the Ljung-Box Q -statistics and their p -values. The Q -statistic at lag k is a test statistic for the null hypothesis that there is no autocorrelation up to order k and is computed as:

$$Q_{LB} = T(T+2) \sum_{j=1}^k \frac{\tau_j^2}{T-j} \quad (7.20)$$

where τ_j is the j -th autocorrelation and T is the number of observations. If the series is not based upon the results of ARIMA estimation, then under the null hypothesis, Q is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of autocorrelations. If the series represents the residuals from ARIMA estimation, the appropriate degrees of freedom should be adjusted to represent the number of autocorrelations less the number of AR and MA terms previously estimated. Note also that some care should be taken in interpreting the results of a Ljung-Box test applied to the residuals from an ARMAX specification (see Dezhbaksh, 1990, for simulation evidence on the finite sample performance of the test in this setting).

The Q -statistic is often used as a test of whether the series is white noise. There remains the practical problem of choosing the order of lag to use for the test. If you choose too small a lag, the test may not detect serial correlation at high-order lags. However, if you

choose too large a lag, the test may have low power since the significant correlation at one lag may be diluted by insignificant correlations at other lags. For further discussion, see Ljung and Box (1979) or Harvey (1990, 1993).

Unit Root Test

This view carries out the Augmented Dickey-Fuller (ADF), GLS transformed Dickey-Fuller (DFGLS), Phillips-Perron (PP), Kwiatkowski, *et. al.* (KPSS), Elliot, Richardson and Stock (ERS) Point Optimal, and Ng and Perron (NP) unit root tests for whether the series (or its first or second difference) is stationary.

See [“Nonstationary Time Series” on page 328](#) for a discussion of stationary and nonstationary time series and additional details on how to carry out the unit roots tests in Eviews.

BDS Test

This view carries out the BDS test for independence, as described in Brock, Dechert, Scheinkman and LeBaron (1996).

The BDS test is a portmanteau test for time based dependence in a series. It can be used for testing against a variety of possible deviations from independence including linear dependence, non-linear dependence, or chaos.

The test can be applied to a series of estimated residuals to check whether the residuals are independent and identically distributed (iid). For example, the residuals from an ARMA model can be tested to see if there is any non-linear dependence in the series after the linear ARMA model has been fitted.

The idea behind the test is fairly simple. To perform the test, we first choose a distance, ϵ . We then consider a pair of points. If the observations of the series truly are iid, then for any pair of points, the probability of the distance between these points being less than or equal to epsilon will be constant. We denote this probability by $c_1(\epsilon)$.

We can also consider sets consisting of multiple pairs of points. One way we can choose sets of pairs is to move through the consecutive observations of the sample in order. That is, given an observation s , and an observation t of a series X , we can construct a set of pairs of the form

$$\{ \{X_s, X_t\}, \{X_{s+1}, X_{t+1}\}, \{X_{s+2}, X_{t+2}\}, \dots, \{X_{s+m-1}, X_{t+m-1}\} \} \quad (7.21)$$

where m is the number of consecutive points used in the set, or *embedding dimension*. We denote the joint probability of every pair of points in the set satisfying the epsilon condition by the probability $c_m(\epsilon)$.

The BDS test proceeds by noting that under the assumption of independence, this probability will simply be the product of the individual probabilities for each pair. That is, if the observations are independent,

$$c_m(\epsilon) = c_1^m(\epsilon). \quad (7.22)$$

When working with sample data, we do not directly observe $c_1(\epsilon)$ or $c_m(\epsilon)$. We can only estimate them from the sample. As a result, we do not expect this relationship to hold exactly, but only with some error. The larger the error, the less likely it is that the error is caused by random sample variation. The BDS test provides a formal basis for judging the size of this error.

To estimate the probability for a particular dimension, we simply go through all the possible sets of that length that can be drawn from the sample and count the number of sets which satisfy the ϵ condition. The ratio of the number of sets satisfying the condition divided by the total number of sets provides the estimate of the probability. Given a sample of n observations of a series X , we can state this in mathematical notation

$$c_{m,n}(\epsilon) = \frac{2}{(n-m+1)(n-m)} \sum_{s=1}^{n-m+1} \sum_{t=s+1}^{n-m+1} \prod_{j=0}^{m-1} I_\epsilon(X_{s+j}, X_{t+j}) \quad (7.23)$$

where I_ϵ is the indicator function

$$I_\epsilon(x, y) = \begin{cases} 1 & \text{if } |x - y| \leq \epsilon \\ 0 & \text{otherwise.} \end{cases} \quad (7.24)$$

Note that the statistics $c_{m,n}$ are often referred to as *correlation integrals*.

We can then use these sample estimates of the probabilities to construct a test statistic for independence

$$b_{m,n}(\epsilon) = c_{m,n}(\epsilon) - c_{1,n-m+1}(\epsilon)^m \quad (7.25)$$

where the second term discards the last $m - 1$ observations from the sample so that it is based on the same number of terms as the first statistic.

Under the assumption of independence, we would expect this statistic to be close to zero. In fact, it is shown in Brock et al. (1996) that

$$(\sqrt{n-m+1}) \frac{b_{m,n}(\epsilon)}{\sigma_{m,n}(\epsilon)} \rightarrow N(0, 1) \quad (7.26)$$

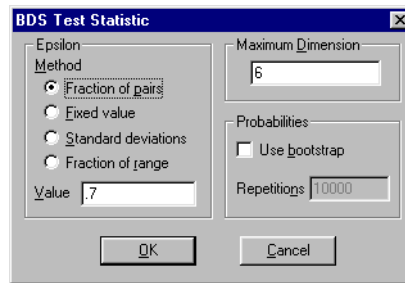
where

$$\sigma_{m,n}^2(\epsilon) = 4 \left(k^m + 2 \sum_{j=1}^{m-1} k^{m-j} c_1^{2j} + (m-1)^2 c_1^{2m} - m^2 k c_1^{2m-2} \right) \quad (7.27)$$

and where c_1 can be estimated using $c_{1,n}$. k is the probability of any triplet of points lying within ϵ of each other, and is estimated by counting the number of sets satisfying the sample condition

$$k_n(\epsilon) = \frac{2}{n(n-1)(n-2)} \sum_{t=1}^n \sum_{s=t+1}^n \sum_{r=s+1}^n (I_\epsilon(X_t, X_s)I_\epsilon(X_s, X_r) + I_\epsilon(X_t, X_r)I_\epsilon(X_r, X_s) + I_\epsilon(X_s, X_t)I_\epsilon(X_t, X_r)) \quad (7.28)$$

To calculate the BDS test statistic in EViews, simply open the series you would like to test in a window, and choose **View/BDS Independence Test...** A dialog will appear prompting you to input options.



To carry out the test, we must choose ϵ , the distance used for testing proximity of the data points, and the dimension m , the number of consecutive data points to include in the set.

The dialog provides several choices for how to specify ϵ :

- **Fraction of pairs:** ϵ is calculated so as to ensure a certain fraction of the total number of pairs of points in the sample lie within ϵ of each other.
- **Fixed value:** ϵ is fixed at a raw value specified in the units as the data series.
- **Standard deviations:** ϵ is calculated as a multiple of the standard deviation of the series.
- **Fraction of range:** ϵ is calculated as a fraction of the range (the difference between the maximum and minimum value) of the series.

The default is to specify ϵ as a fraction of pairs, since this method is most invariant to different distributions of the underlying series.

You must also specify the value used in calculating ϵ . The meaning of this value varies based on the choice of method. The default value of 0.7 provides a good starting point for the default method when testing shorter dimensions. For testing longer dimensions, you should generally increase the value of ϵ to improve the power of the test.

EViews also allows you to specify the maximum correlation dimension for which to calculate the test statistic. EViews will calculate the BDS test statistic for all dimensions from 2 to the specified value, using the same value of ϵ or each dimension. Note the same ϵ is used only because of calculational efficiency. It may be better to vary ϵ with the correlation dimension to maximize the power of the test.

In small samples or in series that have unusual distributions, the distribution of the BDS test statistic can be quite different from the asymptotic normal distribution. To compensate for this, EViews offers you the option of calculating bootstrapped p -values for the test statistic. To request bootstrapped p -values, simply check the **Use bootstrap** box, then specify the number of repetitions in the field below. A greater number of repetitions will provide a more accurate estimate of the p -values, but the procedure will take longer to perform.

When bootstrapped p -values are requested, EViews first calculates the test statistic for the data in the order in which it appears in the sample. EViews then carries out a set of repetitions where for each repetition a set of observations is randomly drawn without replacement from the original data with the same size as the original data. For each repetition, EViews recalculates the BDS test statistic for the randomly drawn data, then compares the statistic to that obtained from the original data. When all the repetitions are complete, EViews forms the final estimate of the bootstrapped p -value by dividing the lesser of the number of repetitions above or below the original statistic by the total number of repetitions, then multiplying by two (to account for the two tails).

As an example of a series where the BDS statistic will reject independence, consider a series generated by the non-linear moving average model:

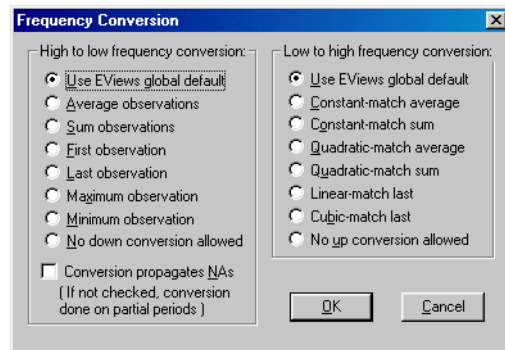
$$y_t = u_t + 8u_{t-1}u_{t-2} \quad (7.29)$$

where u_t is a normal random variable. On simulated data, the correlogram of this series shows no statistically significant correlations, yet the BDS test strongly rejects the hypothesis that the observations of the series are independent (note that the Q -statistics on the squared levels of the series also reject independence).

Conversion Options

When you fetch a series from an EViews database, it will automatically be converted to the frequency of the current workfile. The conversion options view allows you to set the method that will be used to perform these conversions.

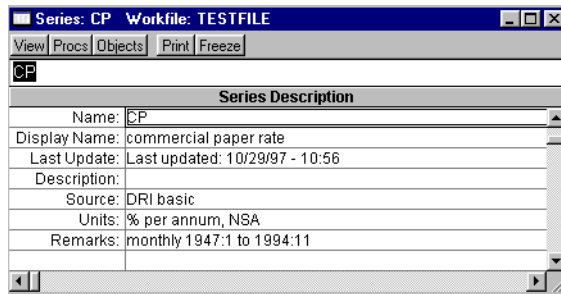
Each series has a frequency conversion method associated with it. The default conversion method is the one set by the EViews global options; see [Appendix A, “Global Options”](#), on page 647. You can change or check the conversion method by clicking **View/Conversion Options...**



Label

This view displays a description of the series object.

You can edit any of the field cells in the series label, except the **Last Update** cell which displays the date/time the series was last modified. Each field contains a single line, except for the **Remarks and History** fields which can contain up to 20 comment lines. Note that if you insert a line, the last (of the 20) line of these fields will be deleted.



The **Name** is the series name as it appears in the workfile; you can rename your series by editing this cell. If you fill in the **Display Name** field, this name may be used in tables and graphs in place of the standard object name. Unlike ordinary object names, **Display Names** may contain spaces and preserve capitalization (upper and lower case letters).

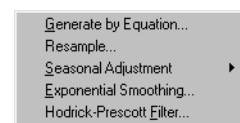
See [Chapter 6](#) for further discussion of label fields and their use in Database searches.

Series Procs

Series procedures may be used to generate new series that are based upon the data in the original series.

Generate by Equation

This is a general procedure that allows you to create new series by using expressions to transform values in the existing series. The



rules governing the generation of series are explained in detail in “[Series Expressions](#)” on page 89.

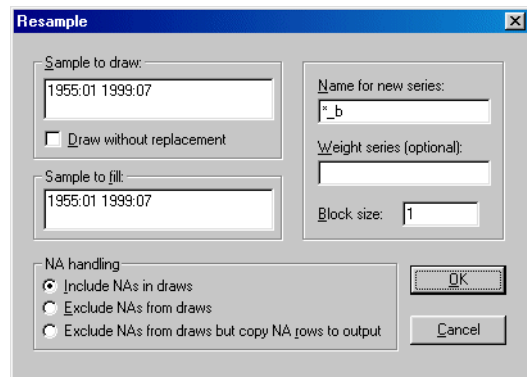
Resampling

The series resampling procedure selects from the observations in a series to create a new series (the resampled series). You may draw your new sample with replacement (allow a given observation to be drawn multiple times) or without replacement. When you select **Procs/Resample...** from the series window, you will be prompted to specify various options.

Input Sample

Describes the sample from which observations are to be drawn. The default is the current workfile sample.

If you select the **Draw without replacement** option, each row will be drawn at most once. This option requires the input sample to be at least as large as the output sample. If you do not select this option, each row will be drawn *with* replacement.



Output Sample

Specifies the sample into which the resampled series will be saved. Any value outside the output sample will not be changed. The default output sample is the current workfile sample. If you select the **Draw without replacement** option, the output sample cannot be larger than the input sample.

NA Handling

The default **Include NAs in draws** instructs EViews to draw from every observation in the input sample, including those that contain missing values. Alternatively, you may select the **Exclude NAs from draws** option so that you draw only from observations in the input sample that do not contain any missing values. Finally, the **Exclude NAs from draws but copy NA rows to output** option first copies matching observations in the input sample that contain missing values to the output sample. The remaining rows of the output sample are then filled by drawing from observations in the input sample that do not contain any missing values. This option keeps observations with missing values fixed and resamples those that do not contain any missing values.

Series Name

The new series will be named using the specified series name. You may provide a series name or a wildcard expression. If you use a wildcard expression, EViews will substitute the existing series name in place of the wildcard. For example, if you are sampling from the series X and specify “*_SMP” as the output series, EViews will save the results in the series X_SMP . You may not specify a destination series that is the same as the original series.

If another series with the specified name exists in the workfile, the values in the output sample will be overwritten with the resampled values. Any values outside the output sample will remain unchanged. If there is a non-series object with the specified name, EViews will return an error message.

Because of these naming conventions, your original series cannot be an auto-series. For example, if the original series is $X(-1)$ or $\text{LOG}(X)$, EViews will issue an error. You will have to generate a new series, say by setting $XLAG = X(-1)$ or $\text{LOGX} = \text{LOG}(X)$, and then resample from the newly generated series.

Weighting

By default, the procedure draws from each row in the input sample with equal probabilities. If you want to attach different probabilities to the rows (importance sampling), you can specify a name of an existing series that contains weights that are proportional to the desired probabilities in each row. The weight series must have non-missing non-negative values in the input sample, but the weights need not add up to 1 since EViews will normalize the weights.

Block Length

By default, sets the *block length* to 1, meaning that we draw one observation at a time from the input sample. If you specify a block length larger than 1, EViews will draw blocks of consecutive rows of the specified length. The blocks drawn in the procedure form a set of *overlapping moving* blocks in the input sample. The drawn blocks will be appended one after the other in the output series until it fills the output sample (the final block will be truncated if the block size is not an integer multiple of the output sample size). Block resampling with a block length larger than 1 makes the most sense when resampling time series data.

Block resampling requires a continuous output sample. Therefore a block length larger than 1 cannot be used when the output sample contains “gaps” or when you have selected the **Exclude NAs from draws but copy NA rows to output** option. If you choose **Exclude NAs from draws** option and the block length is larger than 1, the input sample will shrink in the presence of NAs in order to ensure that there are no missing values in any of the drawn blocks.

Seasonal Adjustment

Time series observed at quarterly and monthly frequencies often exhibit cyclical movements that recur every month or quarter. For example, ice cream sales may surge during summer every year and toy sales may reach a peak every December during Christmas sales. Seasonal adjustment refers to the process of removing these cyclical seasonal movements from a series and extracting the underlying trend component of the series.

The EViews seasonal adjustment procedures are available only for quarterly and monthly series. To seasonally adjust a series, click on **Procs/Seasonal Adjustment** in the series window toolbar and select the adjustment method (Census X12, X11 (Historical), Tramo/Seats or moving average).

Census X12

EViews provides a convenient front-end for accessing the U.S. Census Bureau's X12 seasonal adjustment program from within EViews. The X12 seasonal adjustment program X12A.EXE is publicly provided by the Census and is installed in your EViews directory.

When you request X12 seasonal adjustment from EViews, EViews will perform all of the following steps:

- write out a specification file and data file for the series.
- execute the X12 program in the background, using the contents of the specification file.
- read back the output file and saved data into your EViews workfile.

The following is a brief description of the EViews menu interface to X12. While some parts of X12 are not available via the menus, EViews also provides a more general command interface to the program (see [x12](#) (p. 388) of the *Command and Programming Reference*).

Users who desire a more detailed discussion of the X12 procedures and capabilities should consult the Census Bureau documentation. The full documentation for the Census program, *X12-ARIMA Reference Manual*, can be found in the DOCS subdirectory of your EViews directory in the PDF files (FINALPT1.PDF and FINALPT2.PDF).

To call the X12 seasonal adjustment procedure, select **Procs/Seasonal Adjustment/Census X12...** from the series window menu. A dialog will open with several tabs for setting the X12 options for seasonal adjustment, ARIMA estimation, trading day/holiday adjustment, outlier handling, and diagnostic output.

It is worth noting that when you open the X12 dialog, the options will be set to those from the previously executed X12 dialog. One exception to this rule is the outlier list in the Outliers tab, which will be cleared unless the previous seasonal adjustment was performed on the same series.

Seasonal Adjustment Options

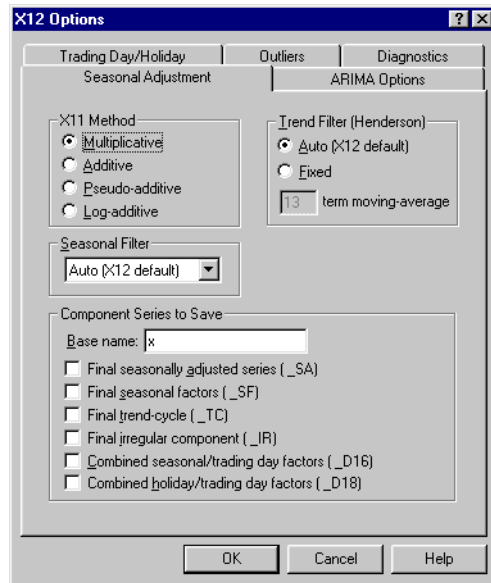
X11 Method specifies the form of the seasonal adjustment decomposition. A description of the four choices can be found in pages 75-77 of the *X12-ARIMA Reference Manual*. Be aware that the Pseudo-additive method must be accompanied by an ARIMA specification (see “ARIMA Options” on page 179 for details on specifying the form of your ARIMA).

Note that the multiplicative, pseudo-additive, and log-additive methods do not allow for zero or negative data.

The **Seasonal Filter** drop-down box allows you to select a seasonal moving average filter to be used when estimating the seasonal factors. The default **Auto (X12 default)** setting is an automatic procedure based on the moving seasonality ratio. For details on the remaining seasonal filters, consult the *X12-ARIMA Reference Manual*. To approximate the results from the previous X11 program’s default filter, choose the X11-default option. You should note the following:

- The seasonal filter specified in the dialog is used for all frequencies. If you wish to apply different filters to different frequencies, you will have to use the more general X12 command language described in detail in [x12 \(p. 388\)](#) of the *Command and Programming Reference*.
- X12 will not allow you to specify a 3×15 seasonal filter for series shorter than 20 years.
- The Census Bureau has confirmed that the X11-default filter option does not produce results which match those obtained from the previous version of X11. The difference arises due to changes in extreme value identification, replacement for the latest values, and the way the end weights of the Henderson filter is calculated. For comparability, we have retained the previous (historical) X11 routines as a separate procedure (see “[Census X11 \(Historical\)](#)” on page 184). Please note that the old X11 program is year 2000 compliant only through 2100 and supports only DOS 8.3 format filenames.

The **Trend Filter (Henderson)** settings allow you to specify the number of terms in the Henderson moving average used when estimating the trend-cycle component. You may use



any odd number greater than 1 and less than or equal to 101. The default is the automatic procedure used by X12.

You must provide a base name for the series stored from the X12 procedure in the **Name for Adjusted Series/Component Series to Save** edit box. To save a series returned from X12 in the workfile, click on the appropriate check box. The saved series will have the indicated suffix appended to the base name. For example, if you enter a base name of “X” and ask to save the seasonal factors (“_SF”), EViews will save the seasonal factors as X_SF.

You should take care when using long base names, since EViews must be able to create a valid series using the base name and any appended Census designations. In interactive mode, EViews will warn you that the resulting name exceeds the maximum series name length; in batch mode, EViews will create a name using a truncated base name and appended Census designations.

The dialog only allows you to store the four most commonly used series. You may, however, store any additional series as listed on Table 6-8 (p. 74) of the *X12-ARIMA Reference Manual* by running X12 from the command line (see [x12](#) (p. 388) of the *Command and Programming Reference*).

ARIMA Options

The X12 program also allows you to fit ARMA models to the series *prior* to seasonal adjustment. You can use X12 to remove deterministic effects (such as holiday and trading day effects) prior to seasonal adjustment and to obtain forecasts/backcasts that can be used for seasonal adjustment at the boundary of the sample. To fit an ARMA, select the **ARIMA Options** tab in the **X12 Options** dialog and fill in the desired options.

The **Data Transformation** setting allows you to transform the series before fitting an ARMA model. The **Auto** option selects between no transformation and a log transformation based on the Akaike information criterion. The **Logistic** option transforms the series y to $\log(y/(1-y))$ and is defined only for series with values that are strictly between 0 and 1. For the **Box-Cox** option, you must provide the parameter value λ for the transformation

$$\begin{cases} \log(y_t) & \text{if } \lambda = 0 \\ \lambda^2 + (y_t^\lambda - 1)/\lambda & \text{if } \lambda \neq 0 \end{cases} \quad (7.30)$$

See the “transform spec” (pp. 60–67) of the *X12-ARIMA Reference Manual* for further details.

ARIMA Specification allows you to choose between two different methods for specifying your ARIMA model. The **Specify in-line** option asks you to provide a single ARIMA specification to fit. The X12 syntax for the ARIMA specification is different from the one used by EViews and follows the Box-Jenkins notation “(p d q)(P D Q)” where

p	nonseasonal AR order
d	order of nonseasonal differences
q	nonseasonal MA order
P	(multiplicative) seasonal AR order
D	order of seasonal differences
Q	(multiplicative) seasonal MA order

The default specification “(0 1 1)(0 1 1)” is the seasonal IMA model

$$(1 - L)(1 - L^s)y_t = (1 - \theta_1 L)(1 - \theta_s L^s)\epsilon_t \quad (7.31)$$

Here are some other examples (L is the lag operator):

(1 0 0)	$(1 - \phi L)y_t = \epsilon_t$
(0 1 1)	$(1 - L)y_t = (1 - \theta L)\epsilon_t$
(1 0 1)(1 0 0)	$(1 - \phi_1 L)(1 - \phi_s L^s)y_t = (1 - \theta L)\epsilon_t$ where $s = 4$ for quarterly data and $s = 12$ for monthly data

You can skip lags using square brackets and explicitly specify the seasonal order after the parentheses:

([2 3] 0 0)	$(1 - \phi_2 L^2 - \phi_3 L^3)y_t = \epsilon_t$
(0 1 1)12	$(1 - L^{12})y_t = (1 - \theta L^{12})\epsilon_t$

See the *X12-ARIMA Reference Manual* (pp. 110–114) for further details and examples of ARIMA specification in X12. Note that there is a limit of 25 total AR, MA, and differencing coefficients in a model and that the maximum lag of any AR or MA parameter is 24 and the maximum number of differences in any ARIMA factor is 3.

Alternatively, if you choose **Select from file**, X12 will select an ARIMA model from a set of possible specifications provided in an external file. The selection process is based on a procedure developed by Statistics Canada for X11-ARIMA/88 and is described in the *X12-ARIMA Reference Manual* (p. 133). If you use this option, you will be asked to provide the name of a file that contains a set of possible ARIMA specifications. By default, EViews will use a file named X12A.MDL that contains a set of default specifications provided by Census (the list of specifications contained in this file is given below).

To provide your own list in a file, the ARIMA specification must follow the X12 syntax as explained in the ARIMA Specification section above. You must specify each model on a separate line, with an “X” at the end of each line except the last. You may also designate

one of the models as a “default” model by marking the end of a line with an asterisk “*” instead of “X”; see p. 133 of the *X12-ARIMA Reference Manual* for an explanation of the use of a default model. To ensure that the last line is read, it should be terminated by hitting the return key.

For example, the default file (X12A.MDL) provided by X12 contains the following specifications:

```
(0 1 1) (0 1 1) *
(0 1 2) (0 1 1) x
(2 1 0) (0 1 1) x
(0 2 2) (0 1 1) x
(2 1 2) (0 1 1)
```

There are two additional options for **Select from file**. **Select best** checks all models in the list and looks for the model with minimum forecast error; the default is to select the first model that satisfies the model selection criteria. **Select by out-of-sample-fit** uses out-of-sample forecast errors (by leaving out some of the observations in the sample) for model evaluation; the default is to use within-sample forecast errors.

The **Regressors** option allows you to include prespecified sets of exogenous regressors in your ARIMA model. Simply use the checkboxes to specify a constant term and/or (centered) seasonal dummy variables. Additional predefined regressors to capture trading day and/or holiday effects may be specified using the **Trading Day/Holiday** tab. You can also use the **Outlier** tab to capture outlier effects.

Trading Day and Holiday Effects

X12 provides options for handling trading day and/or holiday effects. To access these options, select the **Trading Day/Holiday** tab in the X12 Options dialog.

As a first step you should indicate whether you wish to make these adjustments in the ARIMA step or in the X11 seasonal adjustment step. To understand the distinction, note that there are two main procedures in the X12 program: the X11 seasonal adjustment step, and the ARIMA estimation step. The X11 step itself consists of several steps that decompose the series into the trend/cycle/irregular components. The X12 procedure may therefore be described as follows:

- optional preliminary X11 step (remove trading day/holiday effects from series, if requested).
- ARIMA step: fit an ARIMA model (with trading/holiday effects, if specified) to the series from step 1 or to the raw series.
- X11 step: seasonally adjust the series from step 2 using backcasts/forecasts from the ARIMA model.

While it is possible to perform trading day/holiday adjustments in *both* the X11 step and the ARIMA step, Census recommends against doing so (with a preference to performing the adjustment in the ARIMA step). EViews follows this advice by allowing you to perform the adjustment in only one of the two steps.

If you choose to perform the adjustment in the X11 step, there is an additional setting to consider. The checkbox **Apply only if significant (AIC)** instructs EViews to adjust only if warranted by examination of the Akaike information criterion.

It is worth noting that in X11, the significance tests for use of trading day/holiday adjustment are based on an F -est. For this, and a variety of other reasons the X12 procedure with “X11 settings” will not produce results that match those obtained from historical X11. To obtain comparable results, you must use the historical X11 procedure (see “[Census X11 \(Historical\)](#)” on page 184).

Once you select your adjustment method, the dialog will present additional adjustment options:

- **Trading Day Effects** — There are two options for trading day effects, depending on whether the series is a flow series or a stock series (such as inventories). For a flow series, you may adjust for day-of-week effects or only for weekday-weekend contrasts. Trading day effects for stock series are available only for monthly series and the day of the month in which the series is observed must be provided.
- **Holiday Effects** — Holiday effect adjustments apply only to flow series. For each holiday effect, you must provide a number that specifies the duration of that effect *prior* to the holiday. For example, if you select 8, the level of daily activity changes on the seventh day *before* the holiday and remains at the new level until the holiday (or a day before the holiday, depending on the holiday).

Note that the holidays are as defined for the United States and may not apply to other countries. For further details, see the *X12-ARIMA Reference Manual*, Tables 6–15 (p. 94) and 6–18 (p. 133).

Outlier Effects

As with trading day/holiday adjustments, outlier effects can be adjusted either in the X11 step or in the ARIMA step (see the discussion in “[Trading Day and Holiday Effects](#)” on page 181). However, outlier adjustments in the X11 step are done only to robustify the trading day/holiday adjustments in the X11 step. Therefore, in order to perform outlier adjustment in the X11 step, you must perform trading day/holiday adjustment in the X11 step. Only additive outliers are allowed in the X11 step; other types of outliers are available in the ARIMA step. For further information on the various types of outliers, see the *X12-ARIMA Reference Manual*, Tables 6–15 (p. 94) and 6–18 (p. 133).

If you do not know the exact date of an outlier, you may ask the program to test for an outlier (see “[Diagnostics](#)” on page 183).

Diagnostics

This tab provides options for various diagnostics. The **Sliding spans** and **Historical revisions** options test for stability of the adjusted series. While **Sliding spans** checks the change in adjusted series over a moving sample of fixed size (overlapping subspans), **Historical revisions** checks the change in adjusted series over an increasing sample as new observations are added to the sample. See the *X12-ARIMA Reference Manual* for further details and references of the testing procedure. You may also choose to display various diagnostic output:

- **Residual diagnostics** will report standard residual diagnostics (such as the autocorrelation functions and Q -statistics) to check the adequacy of the fitted ARIMA model. Note that this option requires estimation of an ARIMA model; if you do not provide an ARIMA model nor any exogenous regressors (including those from the **Trading day/Holiday** or **Outlier** tab), the diagnostics will be applied to the original series.
- **Outlier detection** automatically detects and reports outliers using the specified ARIMA model. This option requires an ARIMA specification or at least one exogenous regressor (including those from the **Trading day/Holiday** or **Outlier** tab); if no regression model is specified, the option is ignored.
- **Spectral plots** displays the spectra of the differenced seasonally adjusted series (SP1) and/or of the outlier modified irregular series (SP2). The red vertical dotted lines are the seasonal frequencies and the black vertical dashed lines are the trading day frequencies. If you observe peaks at these vertical lines it is an indication of inadequate adjustment. For further details, see Findley *et al.* (1998, section 3.1). If you request this option, data for the spectra will be stored in a matrix named *seriesname_SA_SP1* and *seriesname_SA_SP2* in your workfile. The first column of these matrices are the frequencies and the second column are 10 times the log spectra at the corresponding frequency.

X11/X12 Troubleshooting

The currently shipping versions of X11 and X12 as distributed by the Census (X11 executables dated 6/15/2000 and version 0.2.9 of X12) have the following limitation regarding directory length. First, you will not be able to run X11/X12 if you are running EViews from a shared directory on a server which has spaces in its name. The solution is to map that directory to a letter drive on your local machine. Second, the temporary directory path used by EViews to read and write data cannot have more than four subdirectories. This temporary directory used by EViews can be changed by selecting **Options/File Locations.../Temp File Path** in the main menu. If your temporary directory has more than four

subdirectories, change the Temp File Path to a writeable path that has fewer subdirectories. Note that if the path contains spaces or has more than 8 characters, it may appear in shortened form compatible with the old DOS convention.

Census X11 (Historical)

The Census X11.2 methods (multiplicative and additive) are the standard methods used by the U.S. Bureau of Census to seasonally adjust publicly released data. The X11 routines are separate programs provided by the Census and are installed in the EViews directory in the files X11Q2.EXE and X11SS.EXE. The documentation for these programs can also be found in your EViews directory as text files X11DOC1.TXT through X11DOC3.TXT.

The X11 programs may be run directly from DOS or from inside EViews. If you run the X11 programs from EViews by choosing the X11 options in the Seasonal Adjustment dialog, the adjusted series and the factor series will be automatically imported into your EViews workfile. X11 summary output and error messages will also be displayed in the series window at the end of the procedure.

The X11 method has many options, the most important of which are available in the Seasonal Adjustment dialog. However, there are other options not available in the EViews dialog; to use these other options, you should run the X11 programs from the DOS command line. All options available in the X11 methods are described in the X11DOC text files in your EViews directory.

You should note that there is a limit on the number of observations that you can seasonally adjust. X11 only works for quarterly and monthly frequencies, requires at least four full years of data, and can adjust only up to 20 years of monthly data and up to 30 years of quarterly data.

Moving Average Methods

Ratio to moving average—multiplicative

The algorithm works as follows. Denote the series to be filtered by y_t .

1. First compute the centered moving average of y_t as

$$x_t = \begin{cases} (0.5y_{t+6} + \dots + y_t + \dots + 0.5y_{t-6})/12 & \text{if monthly} \\ (0.5y_{t+2} + y_{t+1} + y_t + y_{t-1} + 0.5y_{t-2})/4 & \text{if quarterly} \end{cases} \quad (7.32)$$

2. Take the ratio $\tau_t = y_t/x_t$.
3. Compute the seasonal indices. For monthly series, the seasonal index i_m for month m is the average of τ_t using observations only for month m . For quarterly series,

the seasonal index i_q for quarter q is the average of τ_t using observations only for quarter q .

- We then adjust the seasonal indices so that they multiply to one. This is done by computing the seasonal factors as the ratio of the seasonal index to the geometric mean of the indices:

$$s = \begin{cases} i_m / (\sqrt[12]{i_1 i_2 \dots i_{12}}) & \text{if monthly} \\ i_q / (\sqrt[4]{i_1 i_2 i_3 i_4}) & \text{if quarterly} \end{cases} \quad (7.33)$$

- These s are the reported *scaling factors* in the series window and are saved as series if you provide a name in the field box. The interpretation is that the series y is s_j percent higher in period j relative to the adjusted series.
- The seasonally adjusted series is obtained by dividing y_t by the seasonal factors s_j .

Difference from moving average—additive

Suppose that we wish to filter y_t .

- First compute the centered moving average of y_t as in [Equation \(7.32\) on page 184](#).
- Take the difference $d_t = y_t - x_t$.
- Compute the seasonal indices. For monthly series, the seasonal index i_m for month m is the average of d_t using observations only for month m . For quarterly series, the seasonal index i_q for quarter q is the average of d_t using observations only for quarter q .
- We then adjust the seasonal indices so that they add up to zero. This is done by setting $s_j = i_j - \bar{i}$ where \bar{i} is the average of all seasonal indices. These s are the reported *scaling factors*. The interpretation is that the series y is s_j higher in period j relative to the adjusted series.
- The seasonally adjusted series is obtained by subtracting the seasonal factors s_j from y_t .

The main difference between X11 and the moving average methods is that the seasonal factors may change from year to year in X11. The seasonal factors are assumed to be constant for the moving average method.

Tramo/Seats

Tramo (“Time Series Regression with ARIMA Noise, Missing Observations, and Outliers”) performs estimation, forecasting, and interpolation of regression models with missing observations and ARIMA errors, in the presence of possibly several types of outliers. *Seats*

(“Signal Extraction in ARIMA Time Series”) performs an ARIMA-based decomposition of an observed time series into unobserved components. The two programs were developed by Victor Gomez and Agustin Maravall.

Used together, Tramo and Seats provide a commonly used alternative to the Census X12 program for seasonally adjusting a series. Typically, individuals will first “linearize” a series using Tramo and will then decompose the linearized series using Seats.

EViews provides a convenient front-end to the Tramo/Seats programs as a series proc. Simply select **Procs/Seasonal Adjustment/Tramo Seats...** and fill out the dialog. EViews writes an input file which is passed to Tramo/Seats via a call to a .DLL, and reads the output files from Tramo/Seats back into EViews (note: since EViews uses a new .DLL version of Tramo/Seats, results may differ from the older DOS version of the program).

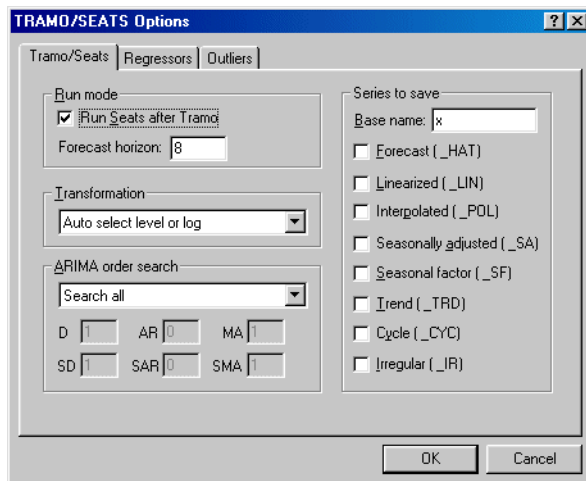
Since EViews only provides an interface to an external program, we cannot provide any technical details or support for Tramo/Seats itself. Users who are interested in the technical details should consult the original documentation *Instructions for the User* which is provided as a .PDF file in the DOCS/TRAMOSEATS subdirectory of your EViews directory.

Dialog Options

The Tramo/Seats interface from the dialog provides access to the most frequently used options. Users who desire more control over the execution of Tramo/Seats may use the command line form of the procedure as documented in the *Command and Programming Reference*.

The dialog contains three tabs. The main tab controls the basic specification of your Tramo/Seats run.

- **Run mode:** You can choose either to run only Tramo or you can select the **Run Seats after Tramo** checkbox to run both. In the latter case, EViews uses the input file produced by Tramo to run Seats. If you wish to run only Seats, you must use the command line interface.
- **Forecast horizon:** You may set the number of periods to forecast outside the cur-



rent sample. If you choose a number smaller than the number of forecasts required to run Seats, Tramo will automatically lengthen the forecast horizon as required.

- **Transformation:** Tramo/Seats is based on an ARIMA model of the series. You may choose to fit the ARIMA model to the level of the series or to the (natural) log of the series, or you select **Auto select level or log**. This option automatically chooses between the level model and the log transformed model using results from a trimmed range-mean regression; see the original Tramo/Seats documentation for further details.
- **ARIMA order search:** You may either specify the orders of the ARIMA model to fit or ask Tramo to search for the “best” ARIMA model. If you select **Fix order** in the combo box and specify the order of all of the ARIMA components, Tramo will use the specified values for all components where the implied ARIMA model is of the form

$$y_t = x_t' \beta + u_t$$

$$\phi(L)\delta(L)u_t = \theta(L)\epsilon_t$$

$$\delta(L) = (1 - L)^D(1 - L^s)^{SD}$$

$$\phi(L) = (1 + \phi_1 L + \dots + \phi_{AR} L^{AR})(1 + \Phi_1 L^s + \dots + \Phi_{SAR}(L^s)^{SAR})$$

$$\theta(L) = (1 + \theta_1 L + \dots + \theta_{MA} L^{MA})(1 + \Theta_1 L^s + \dots + \Theta_{SMA}(L^s)^{SMA})$$

with seasonal frequency s . When you fix the order of your ARIMA you should specify non-negative integers in the edit fields for D , SD , AR , SAR , MA , and SMA .

Alternatively, if you select **Fix only difference orders**, Tramo will search for the best ARMA model for differenced data of the orders specified in the edit fields.

You can also instruct Tramo to choose all orders. Simply choose **Search all** or **Search all and unit complex roots** to have Tramo find the best ARIMA model subject to limitations imposed by Tramo. The two options differ in the handling of complex roots. Details are provided in the original Tramo/Seats documentation.

Warning: if you choose to run Seats after Tramo, note that Seats has the following limit on the ARIMA orders $D \leq 3$, $AR \leq 3$, $MA \leq 3$, $SD \leq 2$, $SAR \leq 1$, $SMA \leq 1$.

- **Series to Save:** To save series output by Tramo/Seats in your workfile, provide a valid base name and check the series you wish to save. The saved series will have a post-fix appended to the basename as indicated in the dialog. If the saved series contains only missing values, it indicates that Tramo/Seats did not return the requested series; see [“Trouble Shooting” on page 189](#).

If Tramo/Seats returns forecasts for the selected series, EViews will append them at the end of the stored series. The workfile *range* must have enough observations after the current workfile *sample* to store these forecasts.

If you need access to series that are not listed in the dialog options, see [“Trouble Shooting” on page 189](#).

- User specified exogenous series: You may provide your own exogenous series to be used by Tramo. These must be a named series or a group in the current workfile and should not contain any missing values in the current sample *and* the forecast period.

If you selected a trading day adjustment option, you have the option of specifying exogenous series to be treated as a holiday series. The specification of the holiday series will depend on whether you chose a weekday/weekend adjustment or a 5-day adjustment. See the original Tramo/Seats documentation for further details.

If you are running Seats after Tramo, you must specify which component to allocate the regression effects. The Tramo default is to treat the regression effect as a separate additional component which is not included in the seasonally adjusted series.

EViews will write a separate data file for each entry in the exogenous series list which is passed to Tramo. If you have many exogenous series with the same specification, it is best to put them into one group.

- Easter/Trading day adjustment: These options are intended for monthly data; see the original Tramo/Seats documentation for details.
- Outlier detection: You may either ask Tramo to automatically detect possible outliers or you can specify your own outlier but not both. If you wish to do both, create a series corresponding to the known outlier and pass it as an exogenous series.

Similarly, the built-in intervention option in Tramo is not supported from the dialog. You may obtain the same result by creating the intervention series in EViews and passing it as an exogenous series. See the example below.

The original Tramo/Seats documentation provides definitions of the various outlier types and the method to detect them.

After you click **OK**, the series window will display the text output returned by Tramo/Seats. If you ran both Tramo and Seats, the output from Seats is appended at the end of Tramo output. Note that this text view will be lost if you change the series view. You should freeze the view into a text object if you wish to refer to the output file without having to run Tramo/Seats again.

It is worth noting that when you run Tramo/Seats, the dialog will generally contain the settings from the previous run of Tramo/Seats. A possible exception is the user specified outlier list which is cleared unless Tramo/Seats is called on the previously used series.

Comparing X12 and Tramo/Seats

Both X12 and Tramo/Seats are seasonal adjustment procedures based on extracting components from a given series. Methodologically, X12 uses a non-parametric moving average based method to extract its components, while Tramo/Seats bases its decomposition on an estimated parametric ARIMA model (the recent addition of ARIMA modelling in X12 appears to be used mainly to identify outliers and to obtain backcasts and forecasts for end-of-sample problems encountered when applying moving average methods.)

For the practitioner, the main difference between the two methods is that X12 does not allow missing values while Tramo/Seats will interpolate the missing values (based on the estimated ARIMA model). While both handle quarterly and monthly data, Tramo/Seats also handles annual and semi-annual data. See the sample programs in the **Example Files** directory for a few results that compare X12 and Tramo/Seats.

Trouble Shooting

Error handling

As mentioned elsewhere, EViews writes an input file which is passed to Tramo/Seats via a call to a .DLL. Currently the Tramo/Seats .DLL does not return error codes. Therefore, the only way to tell that something went wrong is to examine the output file. If you get an error message indicating that the output file was not found, the first thing you should do is to check for errors in the input file.

When you call Tramo/Seats, EViews creates two subdirectories called Tramo and Seats in a temporary directory. This temporary directory is taken from the global option **Options/File Locations.../Temp File Path** (note that long directory names with spaces may appear in shortened DOS form). The Temp File Path can be retrieved in a program by a call to the function `@temp_path` (p. 430) as described in the *Command and Programming Reference*.

The Tramo input file written by EViews will be placed in the subdirectory TRAMO and is named SERIE. A Seats input file written by Tramo is also placed in subdirectory TRAMO and is named SEATS.ITR.

The input file used by Seats is located in the SEATS subdirectory and is named SERIE2. If Seats is run alone, then EViews will create the SERIE2 file. When Tramo and Seats are called together, the Tramo file SEATS.ITR is copied into SERIE2.

If you encounter the error message containing the expression “output file not found”, it probably means that Tramo/Seats encountered an error in one of the input files. You should look for the input files SERIE and SERIE2 in your temp directories and check for any errors in these files.

Retrieving additional output

The output file displayed in the series window is placed in the OUTPUT subdirectory of the TRAMO and/or SEATS directories. The saved series are read from the files returned by Tramo/Seats that are placed in the GRAPH subdirectories. If you need to access other data files returned by Tramo/Seats that are not supported by EViews, you will have to read them back into the workfile using the `read` command from these GRAPH subdirectories. See the PDF documentation file for a description of these data file formats.

Warning: if you wish to examine these files, make sure to read these data files before you run the next Tramo/Seats procedure. EViews will clear these subdirectories before running the next Tramo/Seats command (this clearing is performed as a precautionary measure so that Tramo/Seats does not read results from a previous run).

Exponential Smoothing

Exponential smoothing is a simple method of adaptive forecasting. It is an effective way of forecasting when you have only a few observations on which to base your forecast. Unlike forecasts from regression models which use fixed coefficients, forecasts from exponential smoothing methods adjust based upon past forecast errors. For additional discussion, see Bowerman and O'Connell (1979).

To obtain forecasts based on exponential smoothing methods, choose **Procs/Exponential Smoothing**. The Exponential Smoothing dialog box appears:

You need to provide the following information:

- **Smoothing Method.** You have the option to choose one of the five methods listed.
- **Smoothing Parameters.** You can either specify the values of the smoothing parameters or let EViews estimate them.

To estimate the parameter, type the letter `e` (for estimate) in the edit field. EViews estimates the parameters by minimizing the sum of squared errors. Don't be surprised if the estimated damping parameters are close to one—it is a sign that the series is close to a random walk, where the most recent value is the best estimate of future values.

To specify a number, type the number in the field corresponding to the parameter. All parameters are constrained to be between 0 and 1; if you specify a number outside the unit interval, EViews will estimate the parameter.

- **Smoothed Series Name.** You should provide a name for the smoothed series. By default, EViews will generate a name by appending SM to the original series name, but you can enter any valid EViews name.
- **Estimation Sample.** You must specify the sample period upon which to base your forecasts (whether or not you choose to estimate the parameters). The default is the current workfile sample. EViews will calculate forecasts starting from the first observation after the end of the estimation sample.
- **Cycle for Seasonal.** You can change the number of seasons per year from the default of 12 for monthly or 4 for quarterly series. This option allows you to forecast from unusual data such as an undated workfile. Enter a number for the cycle in this field.

Single Smoothing (one parameter)

This single exponential smoothing method is appropriate for series that move randomly above and below a constant mean with no trend nor seasonal patterns. The smoothed series \hat{y}_t of y_t is computed recursively by evaluating

$$\hat{y}_t = \alpha y_t + (1 - \alpha)\hat{y}_{t-1} \quad (7.34)$$

where $0 < \alpha \leq 1$ is the *damping* (or *smoothing*) factor. The smaller is the α , the smoother is the \hat{y}_t series. By repeated substitution, we can rewrite the recursion as

$$\hat{y}_t = \alpha \sum_{s=0}^{t-1} (1 - \alpha)^s y_{t-s} \quad (7.35)$$

This shows why this method is called exponential smoothing—the forecast of y_t is a weighted average of the past values of y_t , where the weights decline exponentially with time.

The forecasts from single smoothing are constant for all future observations. This constant is given by

$$\hat{y}_{T+k} = \hat{y}_T \quad \text{for all } k > 0 \quad (7.36)$$

where T is the end of the estimation sample.

To start the recursion, we need an initial value for \hat{y}_t and a value for α . EViews uses the mean of the initial observations of y_t to start the recursion. Bowerman and O'Connell (1979) suggest that values of α around 0.01 to 0.30 work quite well. You can also let EViews estimate α to minimize the sum of squares of one-step forecast errors.

Double Smoothing (one parameter)

This method applies the single smoothing method twice (using the same parameter) and is appropriate for series with a linear trend. Double smoothing of a series y is defined by the recursions

$$\begin{aligned} S_t &= \alpha y_t + (1 - \alpha)S_{t-1} \\ D_t &= \alpha S_t + (1 - \alpha)D_{t-1} \end{aligned} \quad (7.37)$$

where S is the single smoothed series and D is the double smoothed series. Note that double smoothing is a single parameter smoothing method with damping factor $0 < \alpha \leq 1$.

Forecasts from double smoothing are computed as

$$\begin{aligned} \hat{y}_{T+k} &= \left(2 + \frac{\alpha k}{1 - \alpha}\right)S_T - \left(1 + \frac{\alpha k}{1 - \alpha}\right)D_T \\ &= \left(2S_T - D_T + \frac{\alpha}{1 - \alpha}(S_T - D_T)k\right) \end{aligned} \quad (7.38)$$

The last expression shows that forecasts from double smoothing lie on a linear trend with intercept $2S_T - D_T$ and slope $\alpha(S_T - D_T)/(1 - \alpha)$.

Holt-Winters—Multiplicative (three parameters)

This method is appropriate for series with a linear time trend and multiplicative seasonal variation. The smoothed series \hat{y}_t is given by

$$\hat{y}_{t+k} = (a + bk)c_{t+k} \quad (7.39)$$

where

$$\begin{aligned} a & \text{ permanent component (intercept)} \\ b & \text{ trend} \\ c_t & \text{ multiplicative seasonal factor} \end{aligned} \quad (7.40)$$

These three coefficients are defined by the following recursions

$$\begin{aligned} a(t) &= \alpha \frac{y_t}{c_t(t-s)} + (1 - \alpha)(a(t-1) + b(t-1)) \\ b(t) &= \beta(a(t) - a(t-1)) + (1 - \beta)b(t-1) \\ c_t(t) &= \gamma \frac{y_t}{a(t)} + (1 - \gamma)c_t(t-s) \end{aligned} \quad (7.41)$$

where $0 < \alpha, \beta, \gamma < 1$ are the damping factors and s is the seasonal frequency specified in the **Cycle for Seasonal** field box.

Forecasts are computed by

$$\hat{y}_{t+k} = (a(T) + b(T)k)c_{T+k-s} \quad (7.42)$$

where the seasonal factors are used from the last s estimates.

Holt-Winters—Additive (three parameter)

This method is appropriate for series with a linear time trend and additive seasonal variation. The smoothed series \hat{y}_t is given by

$$\hat{y}_{t+k} = a + bk + c_{t+k} \quad (7.43)$$

where a and b are the permanent component and trend as defined above in [Equation \(7.40\)](#) and c are the additive seasonal factors. The three coefficients are defined by the following recursions

$$\begin{aligned} a(t) &= \alpha(y_t - c_t(t-s)) + (1-\alpha)(a(t-1) + b(t-1)) \\ b(t) &= \beta(a(t) - a(t-1)) + 1 - \beta b(t-1) \\ c_t(t) &= \gamma(y_t - a(t+1)) - \gamma c_t(t-s) \end{aligned} \quad (7.44)$$

where $0 < \alpha, \beta, \gamma < 1$ are the damping factors and s is the seasonal frequency specified in the **Cycle for Seasonal** field box.

Forecasts are computed by

$$\hat{y}_{T+k} = a(T) + b(T)k + c_{T+k-s} \quad (7.45)$$

where the seasonal factors are used from the last s estimates.

Holt-Winters—No Seasonal (two parameters)

This method is appropriate for series with a linear time trend and no seasonal variation. This method is similar to the double smoothing method in that both generate forecasts with a linear trend and no seasonal component. The double smoothing method is more parsimonious since it uses only one parameter, while this method is a two parameter method. The smoothed series \hat{y}_t is given by

$$\hat{y}_{t+k} = a + bk \quad (7.46)$$

where a and b are the permanent component and trend as defined above in [Equation \(7.40\)](#).

These two coefficients are defined by the following recursions

$$\begin{aligned}
 a(t) &= \alpha y_t + (1 - \alpha)(a(t-1) + b(t-1)) \\
 b(t) &= \beta(a(t) - a(t-1)) + 1 - \beta b(t-1)
 \end{aligned}
 \tag{7.47}$$

where $0 < \alpha, \beta, \gamma < 1$ are the damping factors. This is an exponential smoothing method with two parameters.

Forecasts are computed by

$$\hat{y}_{T+k} = a(T) + b(T)k \tag{7.48}$$

These forecasts lie on a linear trend with intercept $a(T)$ and slope $b(T)$.

It is worth noting that Holt-Winters—No Seasonal, is *not* the same as additive or multiplicative with $\gamma = 0$. The condition $\gamma = 0$ only restricts the seasonal factors from changing over time so there are still (fixed) nonzero seasonal factors in the forecasts.

Illustration

As an illustration of forecasting using exponential smoothing we forecast data on monthly housing starts (HS) for the period 1985:01–1988:12 using the DRI Basics data for the period 1959:01–1984:12. These data are provided in the workfile HS.WF1. Load the workfile, highlight the HS series, double click, select **Procs/Exponential Smoothing...** We use the Holt-Winters—multiplicative method to account for seasonality, name the smoothed forecasts as HS_SM, and estimate all parameters over the period 1959:1–1984:12.

When you click **OK**, EViews displays the results of the smoothing procedure. The first part displays the estimated (or specified) parameter values, the sum of squared residuals, the root mean squared error of the forecast. The zero values for Beta and Gamma in this example mean that the trend and seasonal components are estimated as fixed and not changing.

```

Date: 10/15/97   Time: 00:57
Sample: 1959:01 1984:12
Included observations: 312
Method: Holt-Winters Multiplicative Seasonal
Original Series: HS
Forecast Series: HS_SM
-----
Parameters:   Alpha                0.7100
              Beta                  0.0000
              Gamma                  0.0000
Sum of Squared Residuals          40365.69
Root Mean Squared Error           11.37441
-----

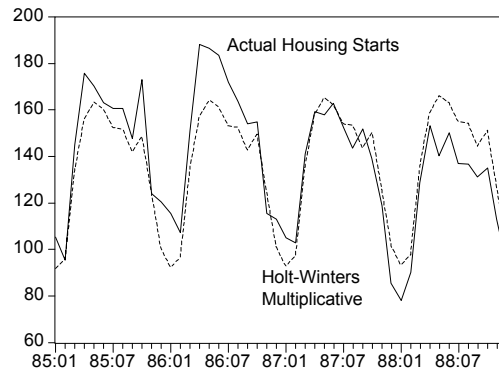
```

The second part of the table displays the mean (α), and trend (β) at the end of the estimation sample that are used for post-sample smoothed forecasts.

End of Period Levels:	Mean	134.6584
	Trend	0.064556
	Seasonals:	
	1984:01	0.680745
	1984:02	0.711559
	1984:03	0.992958
	1984:04	1.158501
	1984:05	1.210279
	1984:06	1.187010
	1984:07	1.127546
	1984:08	1.121792
	1984:09	1.050131
	1984:10	1.099288
	1984:11	0.918354
	1984:12	0.741837

For seasonal methods, the seasonal factors (γ) used in the forecasts are also displayed. The smoothed series in the workfile contains data from the beginning of the estimation sample to the end of the workfile range; all values after the estimation period are forecasts.

When we plot the actual values and the smoothed forecasts on a single graph, we get:



The forecasts from the multiplicative exponential smoothing method do a good job of tracking the seasonal movements in the actual series.

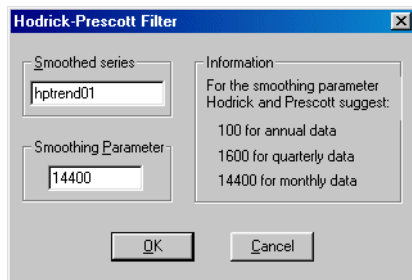
Hodrick-Prescott Filter

This is a smoothing method that is widely used among macroeconomists to obtain a smooth estimate of the long-term trend component of a series. The method was first used in a working paper (circulated in the early 1980's and published in 1997) by Hodrick and Prescott to analyze postwar U.S. business cycles.

Technically, the Hodrick-Prescott (HP) filter is a two-sided linear filter that computes the smoothed series s of y by minimizing the variance of y around s , subject to a penalty that constrains the second difference of s . That is, the HP filter chooses s to minimize:

$$\sum_{t=1}^T (y_t - s_t)^2 + \lambda \sum_{t=2}^{T-1} ((s_{t+1} - s_t) - (s_t - s_{t-1}))^2. \quad (7.49)$$

The penalty parameter λ controls the smoothness of the series σ . The larger the λ , the smoother the σ . As $\lambda = \infty$, s approaches a linear trend.



To smooth the series using the Hodrick-Prescott filter, choose **Procs/Hodrick-Prescott Filter...**

First, provide a name for the smoothed series. EViews will suggest a name, but you can always enter a name of your choosing. Next, specify an integer value for the smoothing parameter, λ . The default values in EViews are set to be:

$$\lambda = \begin{cases} 100 & \text{for annual data} \\ 1,600 & \text{for quarterly data} \\ 14,400 & \text{for monthly data} \end{cases} \quad (7.50)$$

EViews will round off any non-integer values.

When you click **OK**, EViews displays a graph of the filtered series together with the original series. Note that only data in the current workfile sample are filtered. Data for the smoothed series outside the current sample are filled with NAs.

Commands

The command syntax is to follow the name of the series with a dot and then the view or procedure name, with options specified in parentheses. For example, to view the histogram and descriptive statistics of a series named LWAGE, type

```
lwage.hist
```

To test whether the mean of the series HRS is equal to 3, type

```
hrs.teststat(mean=3)
```

To plot the quantiles of the series INJURE against the quantiles of the uniform distribution, type

```
injure.qqplot(u)
```

To plot the correlogram of the series GDP up to 20 lags, type

```
gdp.correl(20)
```

To smooth the series GDP by the Hodrick-Prescott filter with smoothing parameter 1600 and to save the smoothed series as GDP_HP, type

```
gdp.hpf(1600) gdp_hp
```

See “Series” (p. 39) in the *Command and Programming Reference* for a complete list of commands and options that are available for series objects.

Chapter 8. Groups

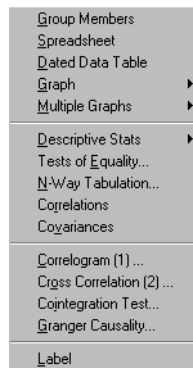
This chapter describes the views and procedures of a group object. With a group, you can compute various statistics that describe the relationship between multiple series and display them in various forms such as spreadsheets, tables, and graphs.

The remainder of this chapter assumes that you are already familiar with the basics of creating and working with a group. See the documentation of EViews features beginning with [Chapter 3](#) for relevant details on the basic operations.

Views from a Group Window

The group view drop-down menu is divided into four blocks:

- The views in the first block provide various ways of looking at the actual data in the group.
- The views in the second block display various basics statistics.
- The views in the third block are for specialized statistics typically computed using time series data.
- The fourth block contains the label view, which provides information regarding the group object.



Group Members

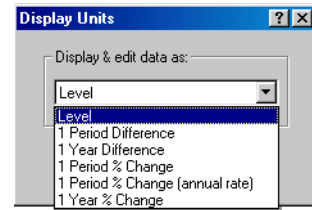
This view displays the member series in the group and allows you to alter the group. To change the group, simply edit the group window. You can add other series from the workfile, include expressions involving series, or you can delete series from the group.

Note that editing the window does not change the list of group members. Once you make your changes to the list, you must press the **UpdateGroup** button in the group window toolbar to save the changes.

Spreadsheet

This view displays the data, in spreadsheet form, for each series in the group. If you wish, you can flip the rows and columns of the spreadsheet by pressing the **Transpose** button. In transpose format, each row contains a series, and each column an observation or date. Pressing the **Transpose** button toggles between the two spreadsheet views.

You can change the display mode of your spreadsheet view to show common transformations of your data. By default, EViews displays the original values in the series. If you wish, you can change the spreadsheet display to show various differences of the series (in levels or percent changes). Simply click on the **Transform** button on the toolbar and select one of the display units from the drop-down menu.



If you select any method beside “Level”, EViews will add a header line to the spreadsheet display describing the transformation method used to generate the data displayed in the spreadsheet view.

You can edit the series data in either levels or transformed values. The **Edit +/-** on the group toolbar toggles the edit mode for the group. If you are in edit mode, an edit window appears in the top of the group window and a double-box is used to indicate the cell that is being edited.

obs	GDP	GDPR	M1	PR	% Change
1952:1	NA	NA	NA	NA	
1952:2	0.284405	-0.023482	0.785784	0.307046	
1952:3	1.702128	0.680234	1.473656	1.015001	
1952:4	3.626220	3.076665	-0.674730	0.533124	
1953:1	1.884253	1.982665	1.614635	-0.096489	
1953:2	0.977543	0.780835	-0.188380	0.195223	
1953:3	-0.130822	-0.521794	0.804045	0.393012	
1953:4	-1.309929	-1.547102	-1.140126	0.240907	
1954:1	-0.106185	-0.448669	0.217105	0.341994	
1954:2	0.132873	-0.075680	0.931069	0.208734	
1954:3	1.326964	1.103598	2.467557	0.220907	
1954:4	2.004976	1.963722	-0.278547	0.040481	
1955:1	3.452260	2.833753	1.606659	0.601446	
1955:2	2.084885	1.775873	0.042518	0.303594	
1955:3	2.042305	1.288608	1.396634	0.744130	
1955:4	1.596378	0.514826	-0.818778	1.075991	
1956:1					

Here, we are editing the data in the group in 1-period percent changes (note the label to the right of the edit field). If we change the 1952:4 value of the percent change in GDP, from 3.626 to 5, the values of GDP from 1952:4 to the end of the workfile will change to reflect the one-time increase in the value of GDP.

Dated Data Table

The dated data table view is used to construct tables for reporting and presenting data, forecasts, and simulation results. This view displays the series contained in the group in a variety of formats. You can also use this view to perform common transformations and frequency conversions, and to display data at various frequencies in the same table.

For example, suppose you wish to show your quarterly data for the CS and GDP series, with data for each year, along with an annual average, on a separate line:

	1988				1988
CS	3128.2	3147.8	3170.6	3202.9	3162.4
GDP	4655.3	4704.8	4734.5	4779.7	4718.6
	1989				1989
CS	3203.6	3212.2	3235.3	3242.0	3223.3
GDP	4817.6	4839.0	4839.0	4856.7	4838.1

The dated data table handles all of the work of setting up this table, and computing the summary values.

Alternatively, you may wish to display annual averages for each year up to the last, followed by the last four quarterly observations in your sample:

	1987	1988	1989	89:1	89:2	89:3	89:4
CS	3052.2	3162.4	3223.3	3203.6	3212.2	3235.3	3242.0
GDP	4540.0	4718.6	4838.1	4817.6	4839.0	4839.0	4856.7

Again, the dated data table may be used to automatically perform the required calculations and set up the table.

The dated data table is capable of creating more complex tables and performing a variety of other calculations. Note, however, that the dated data table view is currently available only for annual, semi-annual, quarterly, or monthly workfiles.

Creating and Specifying a Dated Data Table

To create a dated data table, create a group containing the series of interest and select **View/Dated Data Table**. The group window initially displays a default table view. The default is to show a single year of data on each line, along with a summary measure (the annual average).

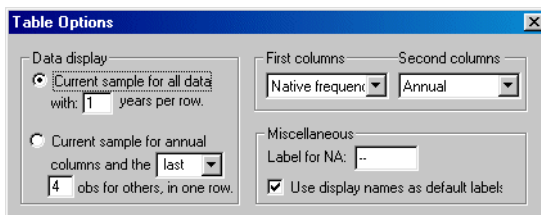
You can, however, set options to control the display of your data through the Table and Row Options dialogs. Note the presence of two new buttons on the group window toolbar, labeled **TabOptions** (for Table Options) and **RowOptions**. **TabOptions** sets the global options for the dated data table. These options will apply to all series in the group object. The **RowOptions** button allows you to override the global options for a particular series. Once you specify table and row options for your group, EViews will remember these options the next time you open the dated data table view for the group.

Table Setup

When you click on the **TabOptions** button, the Table Options dialog appears. The top half of the dialog provides options to control the general style of the table.

The buttons on the left hand side of the dialog allow you to choose between the two display formats described above:

- The first style displays the data for n years per row, where n is the positive integer specified in the edit field.
- The second style is a bit more complex. It allows you to specify, for data displayed at a frequency other than annual, the number of observations taken *from the end of the workfile sample* that are to be displayed. For data displayed at an annual frequency, EViews will display observations over the entire workfile sample.



The two combo boxes on the top right of the dialog supplement your dated display choice by allowing you to display your data at multiple frequencies in each row. The **First Columns** selection describes the display frequency for the first group of columns, while the **Second Columns** selection controls the display for the second group of columns. If you select the same frequency, only one set of results will be displayed.

In each combo box, you may choose among:

- Native frequency (the frequency of the workfile)
- Annual
- Quarterly
- Monthly

If necessary, EViews will perform any frequency conversion (to a lower frequency) required to construct the table.

The effects of these choices on the table display are best described by the following example. For purposes of illustration, assume that the current workfile sample is quarterly, with a range from 1997:1–2003:4.

Now suppose that you choose to display the first style (two years per row), with the first columns set to the native frequency, and the second columns set to annual frequency. Each row will contain eight quarters of data (the native frequency data) followed by the corresponding two annual observations (the annual frequency data):

	1997				1998				1997	1998
GDPR	1775.4	1784.9	1789.8	1799.6	1810.6	1820.9		1787.4	1825.8	
							1830.9	1841.0		
CS	681.0	687.2	691.6	696.0	700.2	704.4	708.7	713.3	688.9	706.6
PCGDPD	2.42	1.44	1.46	1.43	1.54	1.64	1.73	1.81	1.69	1.68
	1999				2000				1999	2000
GDPR		1860.9	1870.7	1880.9	1891.0	1900.9		1865.9	1905.7	
	1851.0						1910.6	1920.4		
CS	717.8	722.3	726.8	731.8	736.4	741.0	745.7	750.7	724.7	743.5
PCGDPD	1.88	1.94	2.00	2.05	2.09	2.13	2.16	2.20	1.97	2.15

EViews automatically performs the frequency conversion to annual data using the specified method (see [page 204](#)).

If you reverse the ordering of data types in the first and second columns so that the first columns display the annual data, and the second columns display the native frequency, the top of the table will contain:

	1997	1998	1997				1998			
GDPR	1787.4	1825.8	1775.4	1784.9	1789.8	1799.6	1810.6	1820.9	1830.9	1841.0
CS	688.9	706.6	681.0	687.2	691.6	696.0	700.2	704.4	708.7	713.3
PCGDPD	1.69	1.68	2.42	1.44	1.46	1.43	1.54	1.64	1.73	1.81
	1999	2000	1999				2000			
GDPR	1865.9	1905.7	1851.0	1860.9	1870.7	1880.9	1891.0	1900.9	1910.6	1920.4
CS	724.7	743.5	717.8	722.3	726.8	731.8	736.4	741.0	745.7	750.7
PCGDPD	1.97	2.15	1.88	1.94	2.00	2.05	2.09	2.13	2.16	2.20

Now, click on **TabOptions**, choose the second display style, and enter 4 in the edit box. Then specify **Annual frequency** for the first columns and **Native frequency** for the second columns. EViews will display the annual data for the workfile sample, followed by the last four quarterly observations:

	1997	1998	1999	2000	2001	2002	2003	03:1	03:2	03:3	03:4
GDPR	1787.4	1825.8	1865.9	1905.7	1945.4	1985.3	2024.2	2009.8	2019.4	2029.0	2038.5
CS	688.9	706.6	724.7	743.5	763.4	783.9	805.0	797.0	802.3	807.6	813.0
PCGDPD	1.69	1.68	1.97	2.15	2.26	2.33	2.37	2.36	2.36	2.37	2.38

The last four columns display the data for the last four observations of the current sample: 2003:1–2003:4. The remaining columns are the annual transformations of the current workfile sample 1997:1–2003:4.

Additional Table Options

The bottom of the Table Options dialog controls the default data transformations and numeric display for each series in the group. EViews allows you to use two rows, each with a different transformation and a different output format, to describe each series.

For each row, you specify the transformation method, frequency conversion method, and the number format.

Keep in mind that you may override the default transformation for a particular series using the **RowOptions** menu (see [page 206](#)).

Transformation Methods

The following transformations are available:

None (raw data)	No transformation
1 Period Difference	$(y - y(-1))$
1 Year Difference	$y - y(-f), \text{ where } f = \begin{cases} 1 & \text{for annual} \\ 2 & \text{for semi-annual} \\ 4 & \text{for quarterly} \\ 12 & \text{for monthly} \end{cases}$
1 Period% Change	$100 \times (y - y(-1)) / y(-1)$
1 Period% Change at Annual Rate	<p>Computes R such that</p> $(1 + r/100)^f$ <p>where f is defined above and r is the 1 period% change.</p>
1 Year% Change	$100 \times (y - y(-f)) / y(-f)$, where f is defined above.
No second row	Do not display a second row

We emphasize that the above transformation methods represent only the most commonly employed transformations. If you wish to construct your table with other transformations, you should add an appropriate auto-series to the group.

Frequency Conversion

The following frequency conversion methods are provided:

Average then Transform	First convert by taking the average, then transform the average, as specified.
Transform then Average	First transform the series, then take the average of the transformed series.
Sum then Transform	First convert by taking the sum, then transform the sum, as specified.
First Period	Convert by taking the first quarter of each year or first month of each quarter/year.
Last Period	Convert by taking the last quarter of each year or last month of each quarter/year.

The choice between **Average then Transform** and **Transform then Average** changes the ordering of the transformation and frequency conversion operations. The methods differ only for nonlinear transformations (such as the % change).

For example, if we specify:

The screenshot shows the 'Table Options' dialog box with the following settings:

- Data display:**
 - Current sample for all data with: 1 years per row.
 - Current sample for annual columns and the last 4 obs for others, in one row.
- Frequency Conversion:** First period
- Number format:**
 - Fixed decimal: 2
 - Fixed chars: 7
 - Auto format
- Miscellaneous:**
 - Use display names as default labels

EViews will display a table with data formatted in the following fashion:

	1997				1997
GDPR	1775.4	1784.9	1789.8	1799.6	7149.8
CS	681.0	687.2	691.6	696.0	2755.7
	1998				1998
GDPR	1810.6	1820.9	1830.9	1841.0	7303.3
CS	700.2	704.4	708.7	713.3	2826.5

If, instead, you change the **Frequency Conversion** to **First Period**, EViews will display a table of the form:

	1997				1997
GDPR	1775.4	1784.9	1789.8	1799.6	1775.4
CS	681.0	687.2	691.6	696.0	681.0
	1998				1998
GDPR	1810.6	1820.9	1830.9	1841.0	1810.6
CS	700.2	704.4	708.7	713.3	700.2

Below, we provide an example which illustrates the computation of the percentage change measures.

Formatting Options

EViews lets you choose between fixed decimal, fixed digit, and auto formatting of the numeric data. Generally, auto formatting will produce appropriate output formatting, but if not, simply select the desired method and enter an integer in the edit field. The options are:

Auto format	EViews chooses the format depending on the data.
Fixed decimal	Specify how many digits to display after the decimal point. This option aligns all numbers at the decimal point.
Fixed chars	Specify how many total characters to display for each number.

EViews will round your data prior to display in order to fit the specified format. This rounding is for display purposes only and does not alter the original data.

Row Options

These options allow you to override the row defaults specified by the Table Options dialog. You can specify a different transformation, frequency conversion method, and number format, for each series.

In the dialog that appears, select the series for which you wish to override the table default options. Then specify the transformation, frequency conversion, or number format you want to use for that series. The options are the same as those described above for the row defaults.

Other Options

Label for NA: allows you to define the symbol used to identify missing values in the table. Bear in mind that if you choose to display your data in transformed form, the transformation may generate missing values even if none of the raw data are missing. Dated data table transformations are explained above.

If your series has display names, you can use the display name as the label for the series in the table by selecting the **Use display names as default labels** option. See Chapter 3 for a discussion of display names and the label view.

Illustration

As an example, consider the following dated data table which displays both quarterly and annual data for GDPR and CS:

	1997				1997
GDPR	1775.4	1784.9	1789.8	1799.6	1787.4
	1.20	0.54	0.27	0.55	3.19
CS	681.0	687.2	691.6	696.0	688.9
	0.95	0.91	0.63	0.64	3.17
	1998				1998
GDPR	1810.6	1820.9	1830.9	1841.0	1825.8
	0.61	0.57	0.55	0.55	2.15
CS	700.2	704.4	708.7	713.3	706.6
	0.61	0.60	0.60	0.65	2.57

The first row of output for each of the series is not transformed, while the second row contains the 1-period percentage change in the series. The second row of GDPR is computed using the **Average then Transformed** setting while the second row of CS is computed with the **Transform then Average** option. We specified this alternative setting for CS by using the **RowOptions** dialog to override the table defaults.

The first four columns show the data in native frequency so the choice between **Average then Transform** and **Transform then Average** is irrelevant—each entry in the second row measures the 1-period (1-quarter) percentage change in the variable.

The 1-period percentage change in the last column is computed differently under the two methods. The **Average then Transformed** percentage change in GDP for 1998 measures the percentage change between the average value in 1997 and the average value in 1998. It is computed as:

$$100 \cdot (1825 - 1787) / 1787 \cong 2.15 \quad (8.1)$$

EViews computes this transformation using full precision for intermediate results, then displays the result using the specified number format.

The computation of the **Transform then Average** one-period change in CS for 1998 is a bit more subtle. Since we want a measure of the annual change, we first evaluate the *one-year* percentage change at each of the quarters in the year, and then average the results. For example, the one-year percentage change in 1998:1 is given by $100(700.2 - 681.0) / 681.0 = 2.82$ and the one-year percentage change in 1998:2 is $100(704.4 - 687.2) / 687.2 = 2.50$. Averaging these percentage changes yields

$$100 \left(\frac{700.2 - 681.0}{681.0} + \frac{704.5 - 687.2}{687.2} + \frac{708.7 - 691.6}{691.6} + \frac{713.3 - 696.0}{696.0} \right) / 4 \quad (8.2)$$

$$\cong 2.57$$

Note that this computation differs from evaluating the average of the one-quarter percentage changes for each of the quarters of the year.

Other Menu Items

- **Edit + /-** allows you to edit the row (series) labels as well as the actual data in the table. You will not be able to edit any of the computed ranks and any changes that you make to the row labels will only apply to the dated data table view.

We warn you that *if you edit a data cell, the underlying series data will also change*. This latter feature allows you to use dated data tables for data entry from published sources.

If you want to edit the data in the table but wish to keep the underlying data unchanged, first **Freeze** the table view and then apply **Edit** to the frozen table.

- **Font** allows you to choose the font, font style, and font size to be used in the table.
- **Title** allows you to add a title to the table.
- **Sample** allows you to change the sample to display in the table.

Here is an example of a table after freezing and editing:

	<u>1997</u>				<u>1997</u>
Gross Domestic Product (Billions \$ '92)	1775.4	1784.9	1789.8	1799.6	1787.4
One-period % change	(1.20)	(0.54)	(0.27)	(0.55)	(3.19)
Consumer Expenditure - Services (Billions \$ '92)	681.0	687.2	691.6	696.0	688.9
One-period % change	(0.95)	(0.91)	(0.63)	(0.64)	(3.17)
	<u>1998</u>				<u>1998</u>
Gross Domestic Product (Billions \$ '92)	1810.6	1820.9	1830.9	1841.0	1825.8
One-period % change	(0.61)	(0.57)	(0.55)	(0.55)	(2.15)
Consumer Expenditure - Services (Billions \$ '92)	700.2	704.4	708.7	713.3	706.6
One-period % change	(0.61)	(0.60)	(0.60)	(0.65)	(2.57)

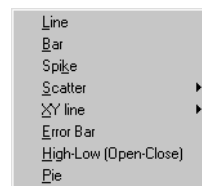
Graphs

These views display the series in the group in various graphical forms. You can create graph objects by freezing these views. [Chapter 10, “Graphs, Tables, and Text Objects”](#), on [page 243](#) explains how to edit and modify graph objects in EViews.

The **Graph** views display all series in a single graph. To display each series in a separate graph, see [“Multiple Graphs” on page 211](#).

Line and Bar Graphs

Displays a line or bar graph of the series in the group. Click anywhere in the background of the graph to modify the scaling options or line patterns.



Scatter

There are five variations on the scatter diagram view of a series.

Simple Scatter plots a scatter diagram with the first series on the horizontal axis and the remaining series on the vertical axis.



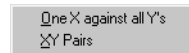
The remaining three options, **Scatter with Regression**, **Scatter with Nearest Neighbor Fit**, and **Scatter with Kernel Fit**, plot fitted lines of the first series against the second on top of the scatter diagram. They differ in how the fitted lines are calculated. All three graph views are described in detail in [“Scatter Diagrams with Fit Lines” beginning on page 233](#).

XY Pairs produces scatterplots of the first series in the group against the second, the third series against the fourth, and so forth.

XY Line

These views plot XY line graphs of the series in the group. They are similar to the scatterplot graphs, but with successive observations connected by lines. **One X against all Y's** will plot the first series in the group against all other series in the group. **XY pairs** will produce XY plots for successive pairs of series in the group.

You can also display symbols only (similar to a scatterplot), or lines and symbols for each XY graph. Click anywhere in the background of the view and change the line attributes for the selected line from **Lines only** to **Symbol only** or **Line & Symbol**.



See [Chapter 10, “Graphs, Tables, and Text Objects”](#), on page 243 for additional details on graph customization.

Error Bar

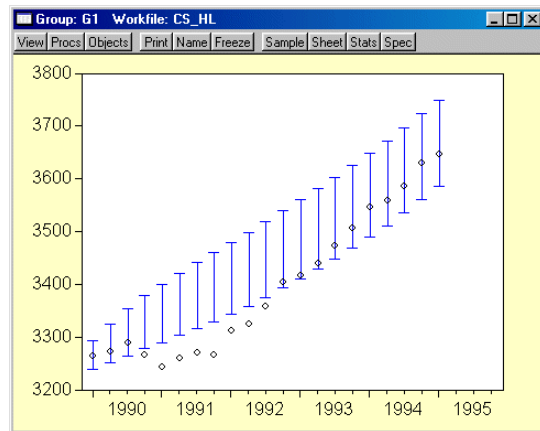
The Error Bar view plots error bars using the first two or three series in the group. The first series is used for the “high” value and the second series is the “low” value. The high and low values are connected with a vertical line. The (optional) third series is plotted as a small circle.

Note that EViews does not check the values of your high and low data for consistency. If the high value is *below* the low value, EViews will draw “outside half-lines” that do not connect.

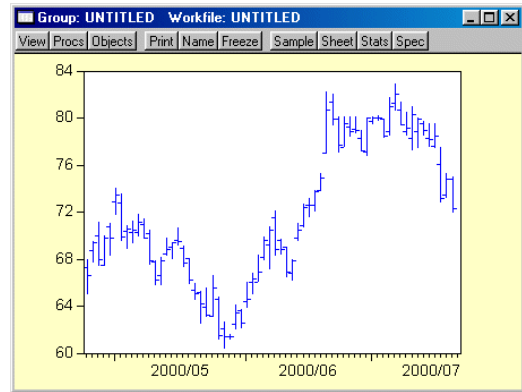
This view is commonly used to display confidence intervals for a statistic.

High-Low (Open-Close)

This view plots the first two to four series in the group as a high-low (open-close) chart. As the name suggests, this chart is commonly used by financial analysts to plot the daily high, low, opening and closing values of stock prices.



The first series in your group should be used to represent the “high” value and the second series should be the “low” value. The high and low values are connected with a vertical line. EViews will not check the values of your high and low data for consistency. If the high value is *below* the low value, EViews will draw “outside half-lines” that do not connect.



The third and fourth series are optional. If you provide only three series, the third series will be used as the “close” value in a high-low-close graph. The third series will be plotted as a right-facing horizontal line representing the “close” value.

If you provide four series, the third series will represent the “open” value and will be plotted as a left-facing horizontal line. The fourth series will be used to represent the “close” value. The close value will be plotted as a right-facing horizontal line.

Pie Graph

This view displays each observation as a pie chart, where the *percentage* of each series in the group is shown as a wedge in the pie.

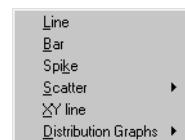
If a series has a negative or missing value, the series will simply be dropped from the pie for that observation. You can label the observation number to each pie; double click in the background of the pie chart and mark the **Label Pie** option in the Graph Options dialog.

Multiple Graphs

While **Graph** views display all series in a single graph, **Multiple Graphs** views display a separate graph for each series in the group.

Line and Bar Graphs

These views display a separate line graph or bar graph for each series in the group.

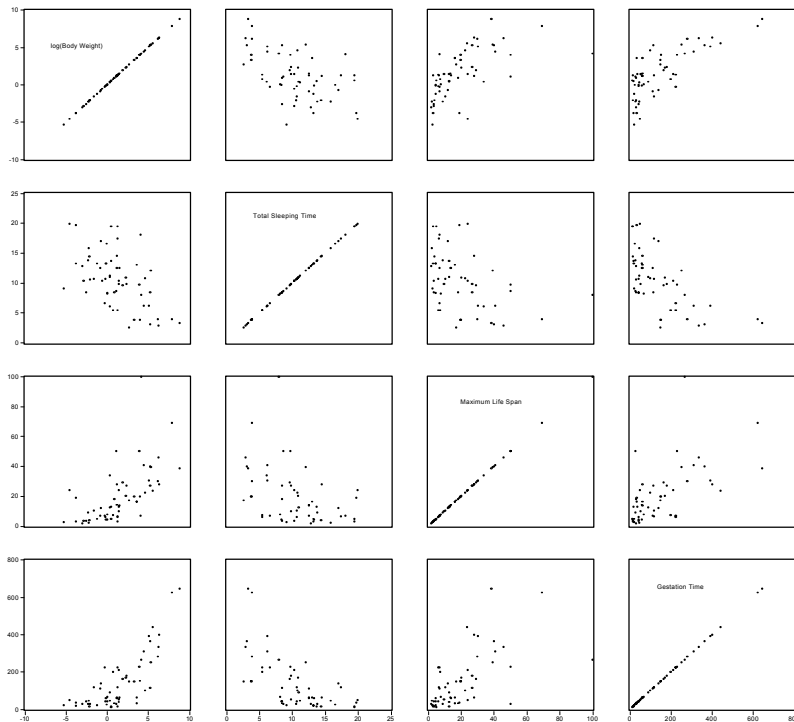


Scatter

First series against all. This view displays scatter plots with the first series in the group on the horizontal axis and the remaining series on the vertical axis, each in a separate graph. If there are G series in the group, $G - 1$ scatter plots will be displayed.

Matrix of all pairs (SCATMAT). This view displays the *scatterplot matrix*, where scatter plots for all possible pairs of series in the group are displayed as a matrix. The important feature of the scatterplot matrix is that the scatter plots are arranged in such a way that plots in the same column share a common horizontal scale, while plots in the same row share a common vertical scale.

If there are G series in the group, G^2 scatter plots will be displayed. The G plots on the main diagonal all lie on a 45 degree line, showing the distribution of the corresponding series on the 45 degree line. The $G(G - 1)/2$ scatters below and above the main diagonal are the same; they are repeated so that we can scan the plots both across the rows and across the columns.



Here is a scatter plot matrix that we copy-and-pasted directly into our document. Note that the resolution of the scatter plot matrix deteriorates quickly as the number of series in the group increases. You may want to freeze the view and modify the graph by moving the axis labels into the scatters on the main diagonal. You can also save more space by moving each scatter close to each other. Set the vertical and horizontal spacing by right-clicking and choosing the **Position and align graphs...** option.

XY line

This view plots the XY line graph of the first series on the horizontal X-axis and each of the remaining series on the vertical Y-axis in separate graphs. See the XY line view for **Graph** for more information on XY line graphs. If there are G series in the group, $G - 1$ XY line graphs will be displayed.

Distribution Graphs

CDF-Survivor-Quantile

This view displays the empirical cumulative distribution functions (CDF), survivor functions, and quantiles of each series in the group. These are identical to the series **CDF-Survivor-Quantile** view; see “[CDF-Survivor-Quantile](#)” on page 225 for a detailed description of how these graphs are computed and the available options.

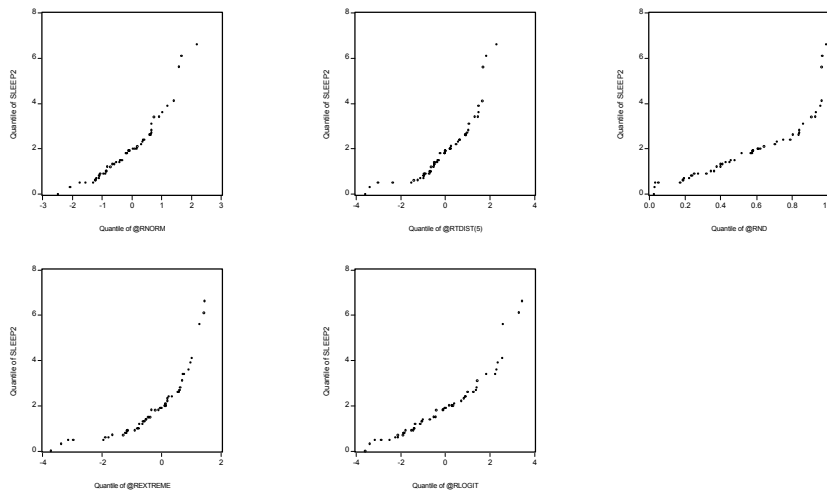
Quantile-Quantile

This view plots the quantiles of each series against the quantiles of a specified distribution or the empirical quantiles of another series. QQ-plots are explained in detail in “[Quantile-Quantile](#)” on page 227.

One useful application of group QQ-plots is to form a group of simulated series from different distributions and plot them against the quantiles of the series of interest. This way you can view, at a glance, the QQ-plots against various distributions. Suppose you want to know the distribution of the series SLEEP2. First, create a group containing random draws from the candidate distributions. For example,

```
group dist @rnorm @rtdist(5) @rextreme @rlogit @rnd
```

creates a group named DIST that contains simulated random draws from the standard normal, a t -distribution with 5 degrees of freedom, extreme value, logistic, and uniform distributions. Open the group DIST, choose **View/Multiple Graphs/Distribution Graphs/Quantile-Quantile**, select the **Series or Group** option and type in the name of the series SLEEP2 in the field of the QQ Plot dialog box.



The quantiles of SLEEP2 are plotted on the vertical axis of each graph. (We moved one of the graphs to make the plots a bit easier to see.) The QQ-plot of the underlying distribution should lie on a straight line. In this example, none of the QQ-plots lie on a line, indicating that the distribution of SLEEP2 does not match any of those in the group DIST.

Descriptive Statistics

This view displays the summary statistics of each series in the group. Details for each statistic are provided in [“Descriptive Statistics” on page 152](#).

- **Common Sample** computes the statistics using observations for which there are no missing values in any of the series in the group (casewise deletion of observations).
- **Individual Samples** computes the statistics using all nonmissing observations for each series (listwise deletion).

The two views are identical if there are no missing values, or if every series has missing observations at the same observation numbers.

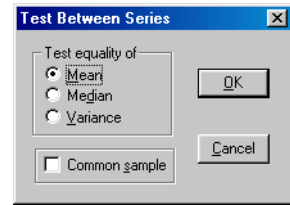
Tests of Equality

This view tests the null hypothesis that all series in the group have the same mean, median (distribution), or variance. All of these tests are described in detail in [“Equality Tests by Classification” on page 159](#).

The **Common sample** option uses only observations for which none of the series in the group has missing values.

Example

As an illustration, we demonstrate the use of this view to test for groupwise heteroskedasticity. Suppose we use data for seven countries over the period 1950–1992 and estimate a pooled OLS model (see [Chapter 21, “Pooled Time Series, Cross-Section Data”](#), on [page 551](#)). To test whether the residuals from this pooled regression are groupwise heteroskedastic, we test the equality of the variances of the residuals for each country.



First, save the residuals from the pooled OLS regression and make a group of the residuals corresponding to each country. This is most easily done by estimating the pooled OLS regression using a pool object and saving the residuals by selecting **Procs/Make Residuals** in the pool object menu or toolbar.

Next, open a group containing the residual series. One method is to highlight each residual series with the right mouse button, double click in the highlighted area and select **Open Group**. Alternatively, you can type `show`, followed by the names of the residual series, in the command window.

Select **View/Tests of Equality...**, and choose the **Variance** option in the Test Between Series dialog box.

Test for Equality of Variances between Series

Date: 10/20/97 Time: 15:24

Sample: 1950 1992

Included observations: 43

Method	df	Value	Probability
Bartlett	6	47.65089	1.39E-08
Levene	(6, 287)	5.947002	7.15E-06
Brown-Forsythe	(6, 287)	4.603232	0.000176

Category Statistics

Variable	Count	Std. Dev.	Mean Abs. Mean Diff.	Mean Abs. Median Diff.
RESID_CAN	42	387.3328	288.2434	275.5092
RESID_FRA	42	182.4492	143.0463	140.4258
RESID_GER	42	224.5817	169.6377	167.0994
RESID_ITA	42	173.4625	132.1824	131.2676
RESID_JAP	42	230.4443	185.5166	185.5166
RESID_UK	42	218.8625	159.4564	157.8945
RESID_US	42	340.9424	271.5252	265.4067
All	294	263.4411	192.8011	189.0171

Bartlett weighted standard deviation: 262.1580

The test statistics decisively reject the null hypothesis of equal variance of the residuals across countries, providing strong evidence of the presence of groupwise heteroskedasticity. You may want to adjust the denominator degrees of freedom to take account of the number of estimated parameters in the regression. The tests are, however, consistent even without the degrees of freedom adjustment.

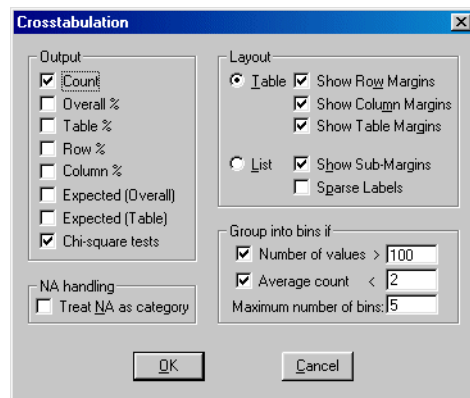
N-Way Tabulation

This view classifies the observations in the current sample into cells defined by the series in the group. You can display the cell counts in various forms and examine statistics for independence among the series in the group. **Select View/N-Way Tabulation...** which opens the tabulation dialog.

Many of the settings will be familiar from our discussion of one-way tabulation in [“One-Way Tabulation” on page 166](#).

Group into Bins If

If one or more of the series in the group is continuous and takes many distinct values, the number of cells becomes excessively large. This option provides you two ways to automatically bin the values of the series into subgroups.



- **Number of values** option bins the series if the series takes more than the specified number of distinct values.
- **Average count** option bins the series if the average count for each distinct value of the series is less than the specified number.
- **Maximum number of bins** specifies the approximate maximum number of subgroups to bin the series. The number of bins may be chosen to be smaller than this number in order to make the bins approximately the same size.

The default setting is to bin a series into approximately 5 subgroups if the series takes more than 100 distinct values or if the average count is less than 2. If you do not want to bin the series, unmark both options.

NA Handling

By default, EViews drops observations from the contingency table where any of the series in the group has a missing value. **Treat NA as category** option includes all observations and counts NAs in the contingency table as an explicit category.

Layout

This option controls the display style of the tabulation. The **Table** mode displays the categories of the first two series in $r \times c$ tables for each category of the remaining series in the group.

The **List** mode displays the table in a more compact, hierarchical form. The **Sparse Labels** option omits repeated category labels to make the list less cluttered. Note that some of the conditional χ^2 statistics are not displayed in list mode.

Output

To understand the options for output, consider a group with three series. Let (i, j, k) index the bin of the first, second, and third series, respectively. The number of observations in the (i, j, k) -th cell is denoted as n_{ijk} with a total of $N = \sum_i \sum_j \sum_k n_{ijk}$ observations.

- **Overall%** is the percentage of the total number of observations accounted for by the cell count.
- **Table%** is the percentage of the total number of observations in the conditional table accounted for by the cell count.
- **Row%** is the percentage of the number of observations in the row accounted for by the cell count.
- **Column%** is the percentage of the number of observations in the column accounted for by the cell count.

The *overall* expected count in the (i, j, k) -th cell is the number expected if *all* series in the group were independent of each other. This expectation is estimated by

$$\hat{n}_{ijk} = \left(\sum_i n_{ijk^*} / N \right) \left(\sum_j n_{ijk^*} / N \right) \left(\sum_k n_{ijk^*} / N \right) N. \quad (8.3)$$

The *table* expected count \tilde{n}_{ijk} is estimated by computing the expected count for the conditional table. For a given table, this expected value is estimated by:

$$\tilde{n}_{ijk^*} = \left(\sum_i n_{ijk^*} / N_{k^*} \right) \left(\sum_j n_{ijk^*} / N_{k^*} \right) N_{k^*} \quad (8.4)$$

where N_{k^*} is the total number of observations in the k^* table.

Chi-square Tests

If you select the **Chi-square tests** option, EViews reports χ^2 statistics for testing the independence of the series in the group. The test statistics are based on the distance between the actual cell count and the count expected under independence.

- **Overall (unconditional) independence among all series in the group.** EViews reports the following two test statistics for overall independence among all series in the group:

$$\text{Pearson } \chi^2 = \sum_{i,j,k} \frac{(\hat{n}_{i,j,k} - n_{i,j,k})^2}{\hat{n}_{i,j,k}} \quad (8.5)$$

$$\text{Likelihood ratio} = 2 \sum_{i,j,k} n_{i,j,k} \log\left(\frac{n_{i,j,k}}{\hat{n}_{i,j,k}}\right)$$

where n_{ijk} and \hat{n}_{ijk} are the actual and overall expected count in each cell. Under the null hypothesis of independence, the two statistics are asymptotically distributed χ^2 with $IJK - (I - 1) - (J - 1) - (K - 1) - 1$ degrees of freedom where I, J, K are the number of categories for each series.

These test statistics are reported at the top of the contingency table

Tabulation of LWAGE and UNION and MARRIED			
Date: 012/15/00 Time: 14:12			
Sample: 1 1000			
Included observations: 1000			
Tabulation Summary			
Variable	Categories		
LWAGE	5		
UNION	2		
MARRIED	2		
Product of Categories	20		
Test Statistics	df	Value	Prob
Pearson X2	13	174.5895	0.0000
Likelihood Ratio G2	13	167.4912	0.0000
WARNING: Expected value is less than 5 in 40.00% of cells (8 of 20).			

In this group there are three series LWAGE, UNION, and MARRIED, each with $I = 5$, $J = 2$, and $K = 2$ categories. Note the WARNING message: if there are many cells with expected value less than 5, the small sample distribution of the test statistic under the null hypothesis may deviate considerably from the asymptotic χ^2 distribution.

- **Conditional independence between series in the group.** If you display in table mode, EViews presents measures of association for *each* conditional table. These measures are analogous to the correlation coefficient; the larger the measure, the larger the association between the row series and the column series in the table. In addition to the Pearson χ^2 for the table, the following three measures of association are reported:

$$\text{Phi coefficient} = \sqrt{\tilde{\chi}^2 / \tilde{N}} \quad (8.6)$$

$$\text{Cramers V} = \sqrt{\tilde{\chi}^2 / ((\min\{r, c\} - 1)\tilde{N})} \quad (8.7)$$

$$\text{Contingency coefficient} = \sqrt{\tilde{\chi}^2 / (\tilde{\chi}^2 + N)} \quad (8.8)$$

where $\min(r, c)$ is the smaller of the number of row categories r or column categories c of the table, and \tilde{N} is the number of observations in the table. Note that all three measures are bounded between 0 and 1, a higher number indicating a stronger relation between the two series in the table. While the correlation coefficient only measures the linear association between two series, these nonparametric measures are robust to departures from linearity.

Table 1: Conditional table for MARRIED=0:

Count		UNION		Total
		0	1	
LWAGE	[0, 1)	0	0	0
	[1, 2)	167	8	175
	[2, 3)	121	44	165
	[3, 4)	17	2	19
	[4, 5)	0	0	0
	Total	305	54	359
<u>Measures of Association</u>		<u>Value</u>		
Phi Coefficient		0.302101		
Cramer's V		0.302101		
Contingency Coefficient		0.289193		
<u>Table Statistics</u>		<u>df</u>	<u>Value</u>	<u>Prob</u>
Pearson X2		2	32.76419	7.68E-08
Likelihood Ratio G2		2	34.87208	2.68E-08

Note: Expected value is less than 5 in 16.67% of cells (1 of 6).

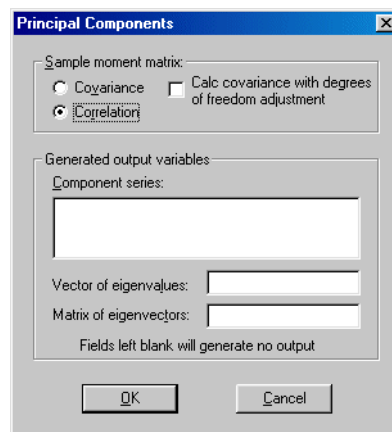
Bear in mind that these measures of association are computed for each two-way table. The conditional tables are presented at the top, and the unconditional tables are reported at the bottom of the view.

Principal Components

The principal components view of the group displays the Eugene-decomposition of the sample second moment of a group of series. Select **View/Principal Components...** to call up the dialog.

You may either decompose the sample covariance matrix or the correlation matrix computed for the series in the group. The sample second moment matrix is computed using data in the current workfile sample. If there are any missing values, the sample second moment is computed using the *common* sample where observations within the workfile range with missing values are dropped.

There is also a checkbox that allows you to correct for degrees of freedom in the computation of the covariances. If you select this option, EViews will divide the sum of squared deviations by $n - 1$ instead of n .



You may store the results in your workfile by simply providing the names in the appropriate fields. To store the first k principal component series, simply list k names in the Component series edit field, each separated by a space. Note that you cannot store more principal components than there are series in the group. You may also store the eigenvalues and eigenvectors in a named vector and matrix.

The principal component view displays output that looks as follows:

Date: 10/31/00 Time: 16:05

Sample: 1 74

Included observations: 74

Correlation of X1 X2 X3 X4

	Comp 1	Comp 2	Comp 3	Comp 4
Eigenvalue	3.497500	0.307081	0.152556	0.042863
Variance Prop.	0.874375	0.076770	0.038139	0.010716
Cumulative Prop.	0.874375	0.951145	0.989284	1.000000
Eigenvectors:				
Variable	Vector 1	Vector 2	Vector 3	Vector 4
X1	-0.522714	-0.164109	-0.236056	0.802568
X2	-0.512619	-0.074307	-0.660639	-0.543375
X3	-0.491857	-0.537452	0.640927	-0.241731
X4	0.471242	-0.823827	-0.311521	0.046838

The column headed by “Comp1” and “Vector1” corresponds to the first principal component, “Comp2” and “Vector2” denote the second principal component and so on. The row labeled “Eigenvalue” reports the eigenvalues of the sample second moment matrix in descending order from left to right. The Variance Prop. row displays the variance propor-

tion explained by each principal component. This value is simply the ratio of each eigenvalue to the sum of all eigenvalues. The Cumulative Prop. row displays the cumulative sum of the Variance Prop. row from left to right and is the variance proportion explained by principal components up to that order.

The second part of the output table displays the eigenvectors corresponding to each eigenvalue. The first principal component is computed as a linear combination of the series in the group with weights given by the first eigenvector. The second principal component is the linear combination with weights given by the second eigenvector and so on.

Correlations, Covariances, and Correlograms

Correlations and **Covariances** display the correlation and covariance matrices of the series in the group. The **Common Sample** view drops observations for which any one of the series has missing data in the current sample. The **Pairwise Samples** view computes each of the second moments using all non-missing observations for the relevant series. Note that **Pairwise Samples** returns entries that correspond to `@cov(x, y)` and `@cor(x, y)` functions. For unbalanced samples, the **Pairwise Samples** method uses the maximum number of observations, but may result in a non-positive definite matrix.

Correlogram displays the autocorrelations and partial autocorrelations of the first series in the group. See “[Correlogram](#)” on page 167, for a description of the correlogram view.

Cross Correlations and Correlograms

This view displays the cross correlations of the first two series in the group. The cross correlations between the two series x and y are given by

$$r_{xy}(l) = \frac{c_{xy}(l)}{\sqrt{c_{xx}(0)} \cdot \sqrt{c_{yy}(0)}}, \quad \text{where } l = 0, \pm 1, \pm 2, \dots \quad (8.9)$$

and

$$c_{xy}(l) = \begin{cases} \sum_{t=1}^{T-l} ((x_t - \bar{x})(y_{t+l} - \bar{y}))/T & l = 0, 1, 2, \dots \\ \sum_{t=1}^{T+l} ((y_t - \bar{y})(x_{t-l} - \bar{x}))/T & l = 0, -1, -2, \dots \end{cases} \quad (8.10)$$

Note that, unlike the autocorrelations, cross correlations are not necessarily symmetric around lag 0.

The dotted lines in the cross correlograms are the approximate two standard error bounds computed as $\pm 2/(\sqrt{T})$.

Cointegration Test

This view carries out the Johansen test for whether the series in the group are cointegrated or not. “[Cointegration Test](#)” on page 537 discusses the Johansen test in detail and describes how one should interpret the test results.

Granger Causality

Correlation does not necessarily imply causation in any meaningful sense of that word. The econometric graveyard is full of magnificent correlations, which are simply spurious or meaningless. Interesting examples include a positive correlation between teachers’ salaries and the consumption of alcohol and a superb positive correlation between the death rate in the UK and the proportion of marriages solemnized in the Church of England. Economists debate correlations which are less obviously meaningless.

The Granger (1969) approach to the question of whether x causes y is to see how much of the current y can be explained by past values of y and then to see whether adding lagged values of x can improve the explanation. y is said to be Granger-caused by x if x helps in the prediction of y , or equivalently if the coefficients on the lagged x ’s are statistically significant. Note that two-way causation is frequently the case; x Granger causes y and y Granger causes x .

It is important to note that the statement “ x Granger causes y ” does not imply that y is the effect or the result of x . Granger causality measures precedence and information content but does not by itself indicate causality in the more common use of the term.

When you select the **Granger Causality** view, you will first see a dialog box asking for the number of lags to use in the test regressions. In general it is better to use more rather than fewer lags, since the theory is couched in terms of the relevance of all past information. You should pick a lag length, l , that corresponds to reasonable beliefs about the longest time over which one of the variables could help predict the other.

EViews runs bivariate regressions of the form

$$\begin{aligned} y_t &= \alpha_0 + \alpha_1 y_{t-1} + \dots + \alpha_l y_{t-l} + \beta_1 x_{t-1} + \dots + \beta_l x_{t-l} + \epsilon_t \\ x_t &= \alpha_0 + \alpha_1 x_{t-1} + \dots + \alpha_l x_{t-l} + \beta_1 y_{t-1} + \dots + \beta_l y_{t-l} + u_t \end{aligned} \quad (8.11)$$

for all possible pairs of (x, y) series in the group. The reported F -statistics are the Wald statistics for the joint hypothesis:

$$\beta_1 = \beta_2 = \dots = \beta_l = 0 \quad (8.12)$$

for each equation. The null hypothesis is that x does *not* Granger-cause y in the first regression and that y does *not* Granger-cause x in the second regression. The test results are given by

Pairwise Granger Causality Tests
 Date: 10/20/97 Time: 15:31
 Sample: 1946:1 1995:4
 Lags: 4

Null Hypothesis:	Obs	F-Statistic	Probability
GDP does not Granger Cause CS	189	1.39156	0.23866
CS does not Granger Cause GDP		7.11192	2.4E-05

For this example, we cannot reject the hypothesis that GDP does not Granger cause CS but we do reject the hypothesis that CS does not Granger cause GDP. Therefore it appears that Granger causality runs one-way from CS to GDP and not the other way.

If you want to run Granger causality tests with other exogenous variables (*e.g.* seasonal dummy variables or linear trends) or if you want to carry out likelihood ratio (LR) tests, run the test regressions directly using equation objects.

Label

This view displays descriptions of the group. You can edit any of the field cells in the label view, except the Last Update cell which shows the date/time the group was last modified.

Name is the group name as it appears in the workfile; you can rename your group by editing this cell. If you fill in the Display Name cell, EViews will use this name in some of the tables and graphs of the group view. Unlike Names, Display Names may contain spaces and preserve capitalization (upper and lower case letters).

See [Chapter 6](#) for a discussion of the label fields and their use in database searches.

Group Procedures

There are three procedures available for groups.

- **Make Equation...** opens an Equation Specification dialog box with the first series in the group listed as the dependent variable and the remaining series as the regressors, including a constant term C. You can modify the specification as you wish.
- **Make Vector Autoregression...** opens an Unrestricted Vector Autoregression dialog box, where all series in the group are listed as endogenous variables in the VAR. See [Chapter 20](#), “[Vector Autoregression and Error Correction Models](#)”, on [page 519](#) for a discussion of specifying and estimating VARs in EViews.
- **Resample...** performs resampling on all of the series in the group. A full description of the resampling procedure is provided in “[Resampling](#)” on [page 175](#).

Commands

You can type the following commands as an alternative to choosing from menus in the group window toolbar. The general rule is to follow the name of the group with a dot and command name for the view or procedure with options specified in parentheses. For example, to view the simple scatter diagram of a group named GRP1, type

```
grp1.scat
```

To test whether the medians of all the series in group GP_WAGE are equal, type

```
gp_wage.testbet (med)
```

To plot the cross correlogram of the first two series in group GRP_MACRO up to 12 lags, type

```
grp_macro.cross (12)
```

Chapter 9. Statistical Graphs Using Series and Groups

EViews provides several methods for exploratory data analysis. In [Chapter 7](#) we document several graph views that may be used to characterize the distribution of a series. This chapter describes several bivariate scatterplot views which allow you to fit lines using parametric, and nonparametric procedures.

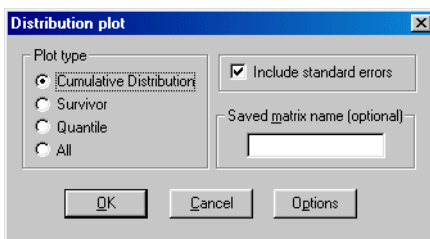
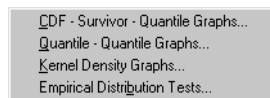
These views, which involve relatively complicated calculations or have a number of specialized options, are documented in detail below. While the discussion sometimes involves some fairly technical issues, you should not feel as though you need to master all of the details to use these views. The graphs correspond to familiar concepts, and are designed to be simple and easy to understand visual displays of your data. The EViews default settings should be sufficient for all but the most specialized of analyses. Feel free to explore each of the views, clicking on **OK** to accept the default settings.

Distribution Graphs of Series

The view menu of a series lists three graphs that characterize the empirical distribution of the series under **View/Distribution...**

CDF-Survivor-Quantile

This view plots the empirical cumulative distribution, survivor, and quantile functions of the series together with the plus or minus two standard error bands. Select **View/Distribution Graphs/CDF-Survivor-Quantile...**



- The **Cumulative Distribution** option plots the empirical cumulative distribution function (CDF) of the series. The CDF is the probability of observing a value from the series not exceeding a specified value r :

$$F_x(r) = \Pr(x \leq r) . \tag{9.1}$$

- The **Survivor** option plots the empirical survivor function of the series. The survivor function gives the probability of observing a value from the series at least as large as some specified value r and is equal to one minus the CDF:

$$S_x(r) = \Pr(x > r) = 1 - F_x(r). \quad (9.2)$$

- The **Quantile** option plots the empirical quantiles of the series. For $0 < q < 1$, the q -th quantile $x_{(q)}$ of x is a number such that

$$\begin{aligned} \Pr(x \leq x_{(q)}) &\leq q \\ \Pr(x \geq x_{(q)}) &\leq 1 - q. \end{aligned} \quad (9.3)$$

The quantile is the inverse function of the CDF; graphically, the quantile can be obtained by flipping the horizontal and vertical axis of the CDF.

- The **All** option plots the CDF, survivor, and quantiles.

Standard Errors

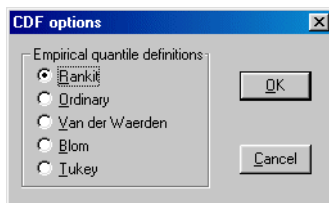
The **Include standard errors** option plots the approximate 95% confidence intervals together with the empirical distribution functions. The methodology for computing these intervals is described in detail in Conover (1980, pp. 114–116). Note that using this approach, we do not compute confidence intervals for the quantiles corresponding to the first and last few order statistics.

Saved matrix name

This optional edit field allows you to save the results in a matrix object. See [cdfplot](#) (p. 157) of the *Command and Programming Reference* for details on the structure of the saved matrix.

Options

EViews provides several methods of computing the empirical CDF used in the CDF and quantile computations.



Given a total of N observations, the CDF for value r is estimated as

Rankit (default)	$(r - 1/2)/N$
Ordinary	r/N
Van der Waerden	$r/(N + 1)$
Blom	$(r - 3/8)/(N + 1/4)$
Tukey	$(r - 1/3)/(N + 1/3)$

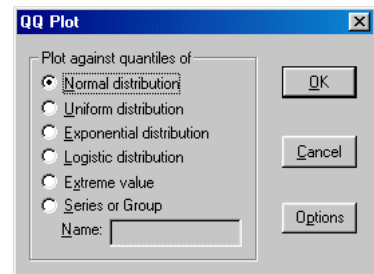
The various methods differ in how they adjust for the non-continuity of the CDF computation. The differences between these alternatives will become negligible as the sample size N grows.

Quantile-Quantile

The quantile-quantile (QQ)-plot is a simple yet powerful tool for comparing two distributions (Cleveland, 1994). This view plots the quantiles of the chosen series against the quantiles of another series or a theoretical distribution. If the two distributions are the same, the QQ-plot should lie on a straight line. If the QQ-plot does not lie on a straight line, the two distributions differ along some dimension. The pattern of deviation from linearity provides an indication of the nature of the mismatch.

To generate a QQ-plot, select **View/Distribution Graphs/Quantile-Quantile...** You can plot against the quantiles of the following theoretical distributions:

- **Normal.** Bell-shaped and symmetric distribution.
- **Uniform.** Rectangular density function. Equal probabilities associated with any fixed interval size in the support.
- **Exponential.** The unit exponential is a positively skewed distribution with a long right tail.
- **Logistic.** This symmetric distribution is similar to the normal, except that it has longer tails than the normal.
- **Extreme value.** The Type-I (minimum) extreme value is a negatively skewed distribution with a long left tail—it is very close to a lognormal distribution.



You can also plot against the quantiles of any series in your workfile. Type the names of the series or groups in the edit box, and select **Series** or **Group**. EVIEWS will compute a

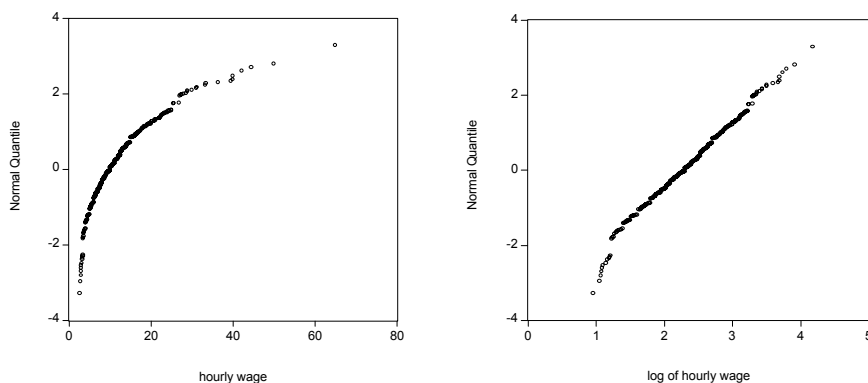
QQ-plot against each series in the list. You can use this option to plot against the quantiles of a simulated series from any distribution; see the example below.

The **Options** button provides you with several methods for computing the empirical quantiles. The options are explained in the CDF-Survivor-Quantile section above; the choice should not make much difference unless the sample is very small.

For additional details, see Cleveland (1994), or Chambers, et al. (1983, Chapter 6).

Illustration

Labor economists typically estimate wage earnings equations with the log of wage on the left-hand side instead of the wage itself. This is because the log of wage has a distribution more close to the normal than the wage, and classical small sample inference procedures are more likely to be valid. To check this claim, we can plot the quantiles of the wage and log of wage against those from the normal distribution. Highlight the series, double click, select **View/Distribution Graphs/Quantile-Quantile...**, and choose the (default) **Normal distribution** option:



If the distributions of the series on the vertical and horizontal axes match, the plots should lie on a straight line. The two plots clearly indicate that the log of wage has a distribution closer to the normal than the wage.

The concave shape of the QQ-plot for the wage indicates that the distribution of the wage series is positively skewed with a long right tail. If the shape were convex, it would indicate that the distribution is negatively skewed.

The QQ-plot for the log of wage falls nearly on a straight line except at the left end, where the plot curves downward. QQ-plots that fall on a straight line in the middle but curve upward at the left end and curve downward at the right end indicate that the distribution is leptokurtic and has a thicker tail than the normal distribution. If the plot curves downward

at the left, and upward at the right, it is an indication that the distribution is platykurtic and has a thinner tail than the normal distribution. Here, it appears that log wages are somewhat platykurtic.

If you want to compare your series with a distribution not in the option list, you can use the random number generator in EViews and plot against the quantiles of the simulated series from the distribution. For example, suppose we wanted to compare the distribution of the log of wage with the F -distribution with 10 numerator degrees of freedom and 50 denominator degrees of freedom. First generate a random draw from an $F(10,50)$ distribution using the command

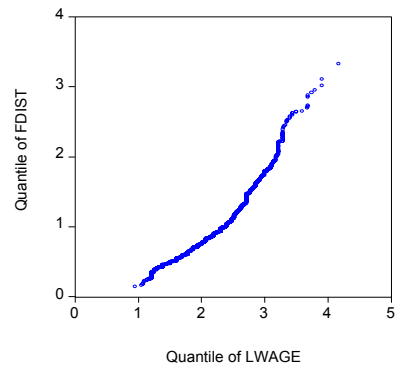
```
series fdist=@rfdist(10,50)
```

Then highlight the log of wage series, double click, select **View/Distribution Graphs/Quantile-Quantile...**, and choose the **Series or Group** option and type in the name of the simulated series (in this case `fdist`)

The plot is slightly convex, indicating that the distribution of the log of wage is slightly negatively skewed compared to the $F(10,50)$.

Kernel Density

This view plots the kernel density estimate of the distribution of the series. The simplest non-parametric density estimate of a distribution of a series is the histogram. You can view the histogram by selecting **View/Descriptive Statistics/Histogram and Stats**. The histogram, however, is sensitive to the choice of origin and is not continuous.

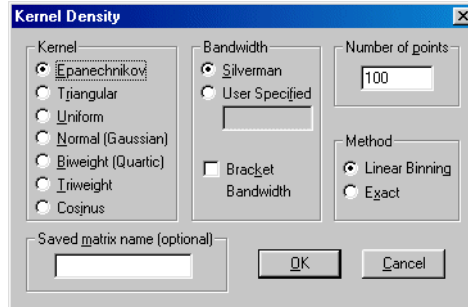


The kernel density estimator replaces the “boxes” in a histogram by “bumps” that are smooth (Silverman 1986). Smoothing is done by putting less weight on observations that are further from the point being evaluated. More technically, the kernel density estimate of a series X at a point x is estimated by:

$$f(x) = \frac{1}{Nh} \sum_{i=1}^N K\left(\frac{x - X_i}{h}\right), \quad (9.4)$$

where N is the number of observations, h is the bandwidth (or smoothing parameter) and K is a kernel weighting function that integrates to one.

When you choose **View/Distribution Graphs/Kernel Density...**, the Kernel Density dialog appears:



To display the kernel density estimates, you need to specify the following:

- **Kernel.** The kernel function is a weighting function that determines the shape of the bumps. EViews provides the following options for the kernel function K :

Epanechnikov (default)	$\frac{3}{4}(1 - u^2)I(u \leq 1)$
Triangular	$(1 - u)(I(u \leq 1))$
Uniform (Rectangular)	$\frac{1}{2}(I(u \leq 1))$
Normal (Gaussian)	$\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}u^2\right)$
Biweight (Quartic)	$\frac{15}{16}(1 - u^2)^2 I(u \leq 1)$
Triweight	$\frac{35}{32}(1 - u^2)^3 I(u \leq 1)$
Cosinus	$\frac{\pi}{4} \cos\left(\frac{\pi}{2}u\right) I(u \leq 1)$

where u is the argument of the kernel function and I is the indicator function that takes a value of one if its argument is true, and zero otherwise.

- **Bandwidth.** The bandwidth h controls the smoothness of the density estimate; the larger the bandwidth, the smoother the estimate. Bandwidth selection is of crucial importance in density estimation (Silverman 1986), and various methods have been suggested in the literature. The **Silverman** option (default) uses a data-based automatic bandwidth

$$h = 0.9kN^{-1/5}\min(s, R/1.34) \quad (9.5)$$

where N is the number of observations, s is the standard deviation, and R is the interquartile range of the series (Silverman 1986, equation 3.31). The factor k is a canonical bandwidth-transformation that differs across kernel functions (Marron and Nolan 1989; Härdle 1991). The canonical bandwidth-transformation adjusts the bandwidth so that the automatic density estimates have roughly the same amount of smoothness across various kernel functions.

To specify a bandwidth of your choice, mark **User Specified** option and type a non-negative number for the bandwidth in the field box. Although there is no general rule for the appropriate choice of the bandwidth, Silverman (1986, section 3.4) makes a case for undersmoothing by choosing a somewhat small bandwidth, since it is easier for the eye to smooth than it is to unsmooth.

The **Bracket Bandwidth** option allows you to investigate the sensitivity of your estimates to variations in the bandwidth. If you choose to bracket the bandwidth, EViews plots three density estimates using bandwidths $0.5h$, h , and $1.5h$.

- **Number of Points.** You must specify the number of points M at which you will evaluate the density function. The default is $M = 100$ points. Suppose the minimum and maximum value to be considered are given by X_L and X_U , respectively. Then $f(x)$ is evaluated at M equi-spaced points given by:

$$x_i = X_L + i \cdot \left(\frac{X_U - X_L}{M} \right), \text{ for } i = 0, 1, \dots, M-1. \quad (9.6)$$

EViews selects the lower and upper evaluation points by extending the minimum and maximum values of the data by two (for the normal kernel) or one (for all other kernels) bandwidth units.

- **Method.** By default, EViews utilizes the **Linear Binning** approximation algorithm of Fan and Marron (1994) to limit the number of evaluations required in computing the density estimates. For large samples, the computational savings are substantial.

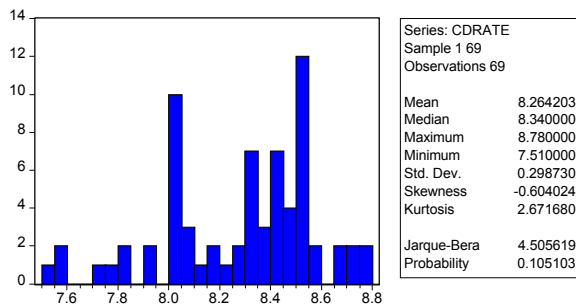
The **Exact** option evaluates the density function using all of the data points for each X_j , $j = 1, 2, \dots, N$ for each x_i . The number of kernel evaluations is therefore of order $O(NM)$, which, for large samples, may be quite time-consuming.

Unless there is a strong reason to compute the exact density estimate or unless your sample is very small, we recommend that you use the binning algorithm.

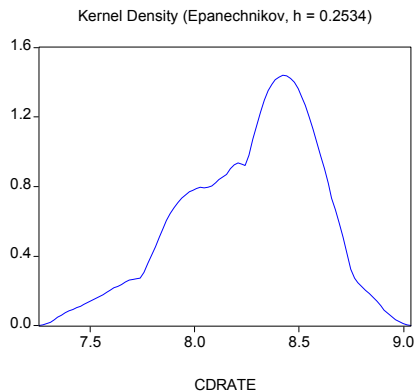
- **Saved matrix name.** This optional edit field allows you to save the results in a matrix object. See [kdensity](#) (p. 236) in the *Command and Programming Reference* for details on the structure of the saved matrix.

Illustration

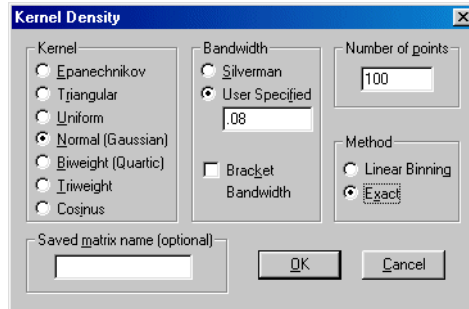
As an illustration of kernel density estimation, we use the three month CD rate data for 69 Long Island banks and thrifts used in Simonoff (1996). The histogram of the CD rate looks as follows:



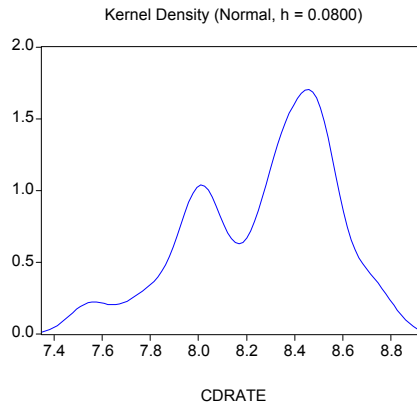
This histogram is a very crude estimate of the distribution of CD rates and does not provide us with much information about the underlying distribution. To view the kernel density estimate, select **View/Distribution Graphs/Kernel Density...** The default options produced the following view:



This density estimate seems to be oversmoothed. Simonoff (1996, chapter 3) uses a Gaussian kernel with bandwidth 0.08. To replicate his results, select **View/Distribution Graphs/Kernel Density...** and fill in the dialog box as follows:



Note that we select the **Exact** method option since there are only 69 observations to evaluate the kernel. The kernel density result is depicted below:

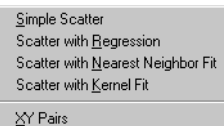


This density estimate has about the right degree of smoothing. Interestingly enough, this density has a trimodal shape with modes at the “focal” numbers 7.5, 8.0, and 8.5.

Scatter Diagrams with Fit Lines

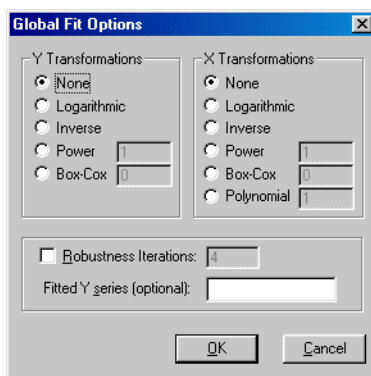
The view menu of a group includes four variants of scatterplot diagrams. **View/Graph/Scatter/Simple Scatter** plots a scatter diagram with the first series on the horizontal axis and the remaining series on the vertical axis.

The remaining three graphs, **Scatter with Regression**, **Scatter with Nearest Neighbor Fit**, and **Scatter with Kernel Fit** plot fitted lines for the second series.



Scatter with Regression

This view fits a bivariate regression of transformations of the second series in the group Y on transformations of the first series in the group X (and a constant).



The following transformations of the series are available for the bivariate fit:

None	y	x
Logarithmic	$\log(y)$	$\log(x)$
Inverse	$1/y$	$1/x$
Power	y^a	x^b
Box-Cox	$(y^a - 1)/a$	$(x^b - 1)/b$
Polynomial	—	$1, x, x^2, \dots, x^b$

where you specify the parameters a and b in the edit field. Note that the Box-Cox transformation with parameter zero is the same as the log transformation.

- If any of the transformed values are not available, EViews returns an error message. For example, if you take logs of negative values, noninteger powers of nonpositive values, or inverses of zeros, EViews will stop processing and issue an error message.
- If you specify a high-order polynomial, EViews may be forced to drop some of the high order terms to avoid collinearity.

When you click **OK**, EViews displays a scatter diagram of the series together with a line connecting the fitted values from the regression. You may optionally save the fitted values as a series. Type a name for the fitted series in the **Fitted Y series** edit field.

Robustness Iterations

The least squares method is very sensitive to the presence of even a few outlying observations. The **Robustness Iterations** option carries out a form of weighted least squares where outlying observations are given relatively less weight in estimating the coefficients of the regression.

For any given transformation of the series, the **Robustness Iteration** option carries out robust fitting with bisquare weights. Robust fitting estimates the parameters a , b to minimize the weighted sum of squared residuals

$$\sum_{i=1}^N r_i (y_i - a - x_i b)^2 \quad (9.7)$$

where y_i and x_i are the transformed series and the bisquare robustness weights r are given by

$$r = \begin{cases} (1 - e_i^2 / (36m^2))^2 & \text{for } |e_i / 6m| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (9.8)$$

where $e_i = y_i - a - x_i b$ is the residual from the previous iteration (the first iteration weights are determined by the OLS residuals), and m is the median of $|e_i|$. Observations with large residuals (outliers) are given small weights when forming the weighted sum of squared residuals.

To choose robustness iterations, click on the check box for **Robustness Iterations** and specify an integer for the number of iterations.

See Cleveland (1993) for additional discussion.

Scatter with Nearest Neighbor Fit

This view displays local polynomial regressions with bandwidth based on nearest neighbors. Briefly, for each data point in a sample, we fit a locally weighted polynomial regression. It is a local regression since we use only the subset of observations which lie in a neighborhood of the point to fit the regression model; it may be weighted so that observations further from the given data point are given less weight.

This class of regressions includes the popular *Loess* (also known as *Lowess*) techniques described by Cleveland (1993, 1994). Additional discussion of these techniques may be found in Fan and Gijbels (1996), and in Chambers, Cleveland, Kleiner, Tukey (1983).

Method

You should choose between computing the local regression at each data point in the sample, or using a subsample of data points.

- **Exact (full sample)** fits a local regression at every data point in the sample.
- **Cleveland subsampling** performs the local regression at only a subset of points. You should provide the size of the subsample M in the edit box.

The number of points at which the local regressions are computed is approximately equal to M . The actual number of points will depend on the distribution of the explanatory variable.

Since the exact method computes a regression at every data point in the sample, it may be quite time consuming when applied to large samples. For samples with over 100 observations, you may wish to consider subsampling.

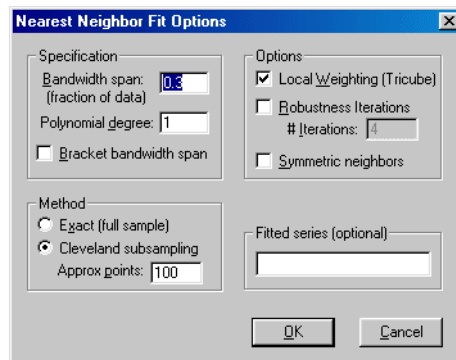
The idea behind subsampling is that the local regression computed at two adjacent points should differ by only a small amount. Cleveland subsampling provides an adaptive algorithm for skipping nearby points in such a way that the subsample includes all of the representative values of the regressor.

It is worth emphasizing that at each point in the subsample, EViews uses the entire sample in determining the neighborhood of points. Thus, each regression in the Cleveland subsample corresponds to an equivalent regression in the exact computation. For large data sets, the computational savings are substantial, with very little loss of information.

Specification

For each point in the sample selected by the **Method** option, we compute the fitted value by running a local regression using data around that point. The **Specification** option determines the rules employed in identifying the observations to be included in each local regression, and the functional form used for the regression.

Bandwidth span determines which observations should be included in the local regressions. You should specify a number α between 0 and 1. The span controls the smoothness of the local fit; a larger fraction α gives a smoother fit. The fraction α instructs EViews to



include the $\lfloor \alpha N \rfloor$ observations nearest to the given point, where $\lfloor \alpha N \rfloor$ is 100 α % of the total sample size, truncated to an integer.

Note that this standard definition of nearest neighbors implies that the number of points need not be symmetric around the point being evaluated. If desired, you can force symmetry by selecting the **Symmetric neighbors** option.

Polynomial degree specifies the degree of polynomial to fit in each local regression.

If you mark the **Bracket bandwidth span** option, EVIEWS displays three nearest neighbor fits with spans of 0.5α , α , and 1.5α .

Other Options

Local Weighting (Tricube) weights the observations of each local regression. The weighted regression minimizes the weighted sum of squared residuals

$$\sum_{i=1}^N w_i (y_i - a - x_i b_1 - x_i^2 b_2 - \dots - x_i^k b_k). \quad (9.9)$$

The tricube weights w are given by

$$w_i = \begin{cases} \left(1 - \left| \frac{d_i}{d(\lfloor \alpha N \rfloor)} \right| \right)^3 & \text{for } \left| \frac{d_i}{d(\lfloor \alpha N \rfloor)} \right| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (9.10)$$

where $d_i = |x_i - x|$ and $d(\lfloor \alpha N \rfloor)$ is the $\lfloor \alpha N \rfloor$ -th smallest such distance. Observations that are relatively far from the point being evaluated get small weights in the sum of squared residuals. If you turn this option off, each local regression will be unweighted with $w_i = 1$ for all i .

Robustness Iterations iterates the local regressions by adjusting the weights to down-weight outlier observations. The initial fit is obtained using weights w_i , where w_i is tricube if you choose **Local Weighting** and 1 otherwise. The residuals e_i from the initial fit are used to compute the robustness bisquare weights r_i as given on [page 235](#). In the second iteration, the local fit is obtained using weights $w_i r_i$. We repeat this process for the user specified number of iterations, where at each iteration the robustness weights r_i are recomputed using the residuals from the last iteration.

Symmetric Neighbors forces the local regression to include the same number of observations to the left and to the right of the point being evaluated. This approach violates the definition, though not the spirit, of nearest neighbor regression.

To save the fitted values as a series; type a name in the **Fitted series** field box. If you have specified subsampling, EVIEWS will linearly interpolate to find the fitted value of y for the

actual value of x . If you have marked the **Bracket bandwidth span** option, EViews saves three series with **_L**, **_M**, **_H** appended to the name, each corresponding to bandwidths of 0.5α , α , and 1.5α , respectively.

Note that Loess is a special case of nearest neighbor fit, with a polynomial of degree 1, and local tricube weighting. The default EViews options are set to provide Loess estimation.

Scatter with Kernel Fit

This view displays fits of local polynomial kernel regressions of the second series in the group Y on the first series in the group X. Both the nearest neighbor fit, described above, and the kernel fit are nonparametric regressions that fit local polynomials. The two differ in how they define “local” in the choice of bandwidth. The effective bandwidth in nearest neighbor regression varies, adapting to the observed distribution of the regressor. For the kernel fit, the bandwidth is fixed but the local observations are weighted according to a kernel function.

Extensive discussion may be found in Simonoff (1996), Hardle (1991), Fan and Gijbels (1996).

Local polynomial kernel regressions fit Y , at each value x , by choosing the parameters β to minimize the weighted sum-of-squared residuals:

$$m(x) = \sum_{i=1}^N (Y_i - \beta_0 - \beta_1(x - X_i) + \dots - \beta_k(x - X_i)^k)^2 K\left(\frac{x - X_i}{h}\right) \quad (9.11)$$

where N is the number of observations, h is the bandwidth (or smoothing parameter), and K is a kernel function that integrates to one. Note that the minimizing estimates of β will differ for each x .

When you select the **Scatter with Kernel Fit** view, the Kernel Fit dialog appears.

You will need to specify the form of the local regression, the kernel, the bandwidth, and other options to control the fit procedure.

Regression

Specify the order of polynomial k to be fit at each data point. The **Nadaraya-Watson** option sets $k = 0$ and locally fits a constant at each x . **Local Linear** sets $k = 1$ at each x . For higher order polynomials, mark the **Local Polynomial** option and type in an integer in the field box to specify the order of the polynomial.

Kernel

The kernel is the function used to weight the observations in each local regression. EViews provides the option of selecting one of the following kernel functions:

Epanechnikov (default)	$\frac{3}{4}(1 - u^2)I(u \leq 1)$
Triangular	$(1 - u)I(u \leq 1)$
Uniform (Rectangular)	$\frac{1}{2}I(u \leq 1)$
Normal (Gaussian)	$\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}u^2\right)$
Biweight (Quartic)	$\frac{15}{16}(1 - u^2)^2 I(u \leq 1)$
Triweight	$\frac{35}{32}(1 - u^2)^3 I(u \leq 1)$
Cosinus	$\frac{\pi}{4} \cos\left(\frac{\pi}{2}u\right) I(u \leq 1)$

where u is the argument of the kernel function and I is the indicator function that takes a value of one, if its argument is true, and zero otherwise.

Bandwidth

The bandwidth h determines the weights to be applied to observations in each local regression. The larger the h , the smoother the fit. By default, **EViews** arbitrarily sets the bandwidth to:

$$h = 0.15(X_U - X_L) \quad (9.12)$$

where $(X_U - X_L)$ is the range of X .

For nearest neighbor bandwidths, see **Scatter with Nearest Neighbor Fit**.

To specify your own bandwidth, mark **User Specified** and enter a nonnegative number for the bandwidth in the edit box.

Bracket Bandwidth option fits three kernel regressions using bandwidths $0.5h$, h , and $1.5h$.

Number of grid points

You must specify the number of points M at which to evaluate the local polynomial regression. The default is $M = 100$ points; you can specify any integer in the field. Suppose the range of the series X is $[X_L, X_U]$. Then the polynomial is evaluated at M equi-spaced points

$$x_i = X_L + i \cdot \left(\frac{X_U - X_L}{M} \right) \quad \text{for } i = 0, 1, \dots, M-1 \quad (9.13)$$

Method

Given a number of evaluation points, EViews provides you with two additional computational options: exact computation and linear binning.

The **Exact** method performs a regression at each x_i , using all of the data points (X_j, Y_j) , for $j = 1, 2, \dots, N$. Since the exact method computes a regression at every grid point, it may be quite time consuming when applied to large samples. In these settings, you may wish to consider the linear binning method.

The **Linear Binning** method (Fan and Marron 1994) approximates the kernel regression by binning the raw data X_j fractionally to the two nearest evaluation points, prior to evaluating the kernel estimate. For large data sets, the computational savings may be substantial, with virtually no loss of precision.

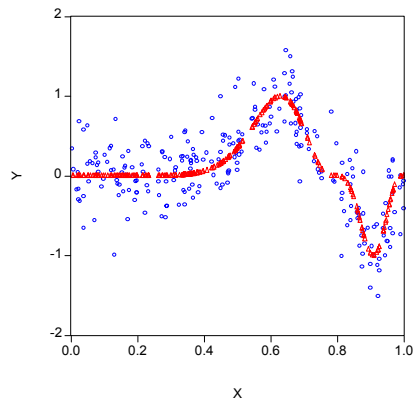
To save the fitted values as a series, type a name in the **Fitted Series** field box. EViews will save the fitted Y to the series, linearly interpolating points computed on the grid, to find the appropriate value. If you have marked the **Bracket Bandwidth** option, EViews saves three series with “_L”, “_M”, “_H” appended to the name, each corresponding to bandwidths 0.5α , α , and 1.5α , respectively.

Example

As an example, we estimate a bivariate relation for a simulated data set used in Hardle (1991). The data were generated by

```
series x=rnd
series y=sin(2*pi*x^3)^3+nrnd*(0.1^.5)
```

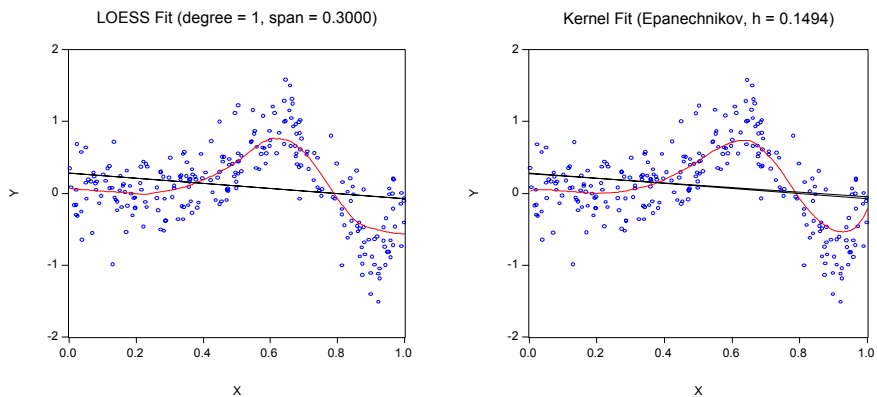
The simple scatter of Y and the “true” conditional mean of Y against X looks as follows:



The triangular shapes in the middle of the scatterplot trace out the “true” conditional mean of Y. Note that the true mean reaches a peak around $x = 0.6$, a valley around $x = 0.9$, and a saddle around $x = 0.8$.

To fit a nonparametric regression of Y on X, you first create a group containing the series Y and X. The order that you enter the series is important; the explanatory series variable must be the first series in the group. Highlight the series name X and then Y, double click in the highlighted area, select **Open Group**, and select **View/Graph/Scatter/Scatter with Nearest Neighbor Fit**, and repeat the procedure for **Scatter with Kernel Fit**.

The two fits, computed using the EVIEWS default settings, are shown together with the linear regression line:



Both local regression lines seem to capture the peak, but the kernel fit is more sensitive to the upturn in the neighborhood of $X = 1$. Of course, the fitted lines change as we modify the options, particularly when we adjust the bandwidth h and window width α .

Commands

The command

```
lwage.cdfplot(a)
```

plots the CDF, quantile and survive functions for LWAGE.

```
lwage.kdensity(k=n)
```

plots the kernel density estimate of LWAGE using a normal kernel and the automatic bandwidth selection.

```
lwage.kdensity(k=e,b=.25)
```

plots the kernel density estimate using the Epanechnikov kernel, a bandwidth of 0.25, and bracketing.

```
group aa age lwage  
aa.linefit(y1,x1)
```

creates a group AA containing the series LWAGE and AGE, and fits a regression line of log LWAGE on log AGE.

```
aa.linefit(y1,d=3)
```

fits log LWAGE to a third degree polynomial in AGE.

```
aa.nnfit  
aa.kerfit
```

uses the default settings to generate scatterplots with (Loess) nearest-neighbor and kernel regression fits to the data in AA.

Chapter 10. Graphs, Tables, and Text Objects

EViews objects (series, groups, equations, and so on) display their views as graphs, tables, and text. Views are dynamic: when the underlying object changes, or the active sample changes, so do the views. Often one would like to preserve the current view so that it does not change when the object changes. In EViews this is referred to as *freezing* the view. Freezing a view will create an object containing a “snapshot” of the contents of the view window. The type of object created varies with the original view: freezing a graphical view creates a graph object, freezing a tabular view creates a table object, and freezing a text view creates a text object.

Frozen views form the basis of most presentation output and EViews provides tools for customizing the appearance of these objects. This chapter describes the options available for controlling the appearance of graph, table, and text objects.

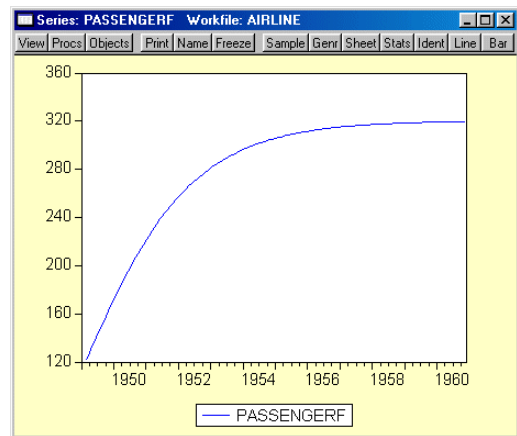
Creating Graphs

Graph objects are usually created by freezing a view. Simply press the **Freeze** button in an object window which contains a graph view.

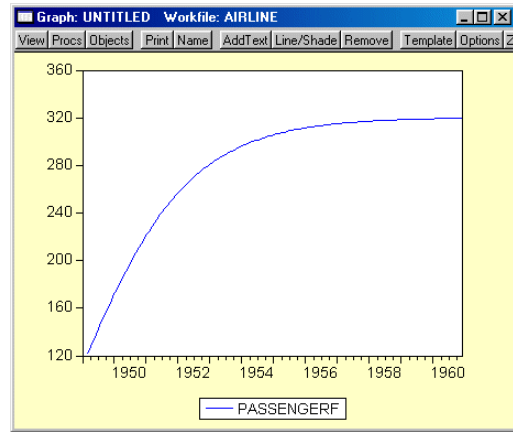
It is important to keep in mind the distinction between a graphical view of an object such as a series or a group, and a graph object created by freezing that view.

For example, suppose you wish to create a graph object containing a line graph of the series PASSENGERF. To display the line graph view of the series, select **View/Graph/Line** from the PASSENGERF series menu.

Notice the window titlebar which shows that this is a view of the series object PASSENGERF. If you wish to create a graph object for further customization, click on the **Freeze** button. EViews will create an UNTITLED graph object containing a snapshot of the view.



Here, the titlebar shows that we have an untitled Graph object. The contents of the two windows are identical since the graph object contains a copy of the contents of the original series view. Notice also that since we are working with a graph object, the menubar provides access to a new set of views and procedures which allow you to further modify the contents of the graph object.



As with other EViews objects, the UNTITLED graph will not be saved with the workfile. If you wish to store the frozen graph object in your workfile, you must name the graph object; press the **Name** button and provide a name.

Note that while you *can* modify the appearance of a graphical view of an object, any changes will be lost when the view is redrawn, *e.g.* when the object window is closed and reopened, when the workfile sample is modified, or when the data underlying the object are changed. If you would like to keep a customized graphical view, say for presentation purposes, you should create a graph object from the view.

You may also create a graph object by combining two or more existing named graph objects. Simply select all of the desired graphs and then double click on any one of the highlighted names. An alternative method of combining graphs is to select **Quick/Show...** and enter the names of the graphs.

Modifying Graphs

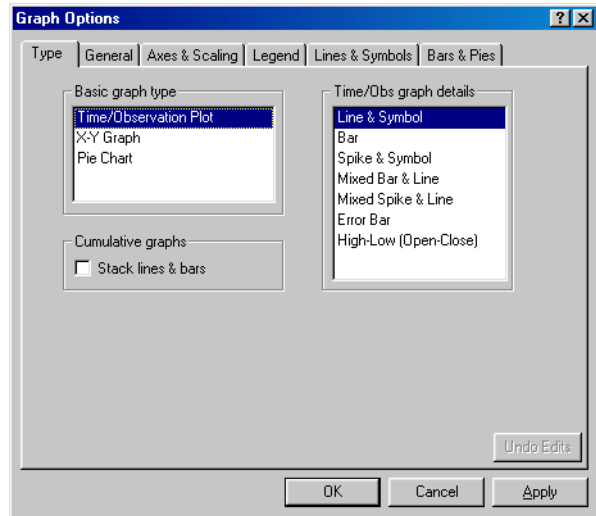
A graph object is made up of a number of elements: the plot area, the axes, the graph legend, and one or more pieces of added text or shading. To select one of these elements for editing, simply click in the area associated with it. A blue box will appear around the selected element. Once you have made your selection, you can click and drag to move the element around the graph, double click to bring up a dialog of options associated with the element, or use the toolbar or the right mouse button menus to remove the object entirely.

Alternatively, you may double click anywhere in the graph window to bring up the **Graph Options** tabbed dialog.

Changing Graph Types

The **Type** tab allows you to change the graph type. The available graph type depends on whether the graph involves a single series or more than one series. For example, the **Mixed Bar & Line**, **Mixed Spike & Line**, **Error Bar**, and **High-Low (Open-Close)** types are only available for graphs containing multiple lines.

Most of the graph types are self-explanatory, but a few comments may prove useful.



If you select the **Line & Symbol** and **Spike & Symbol** types, use the **Lines & Symbols** tab to control the line pattern and/or symbol type. For bar graphs and pie charts, use the **Bars & Pies** tab to control its appearance.

The **Error Bar** type is designed for displaying statistics with standard errors bands. This graph type shows a vertical error bar connecting the values for the first and second series. If the first series value is *below* the second series value, the bar will have outside half-lines. The (optional) third series is plotted as a symbol.

The **High-Low (Open-Close)** type displays up to four series. Data from the first two series (the high-low values) will be connected as a vertical line, the third series (the open value) as a left horizontal half-line, and the fourth series (the close value) as a right horizontal half-line. This graph type is commonly used to display daily stock price data.

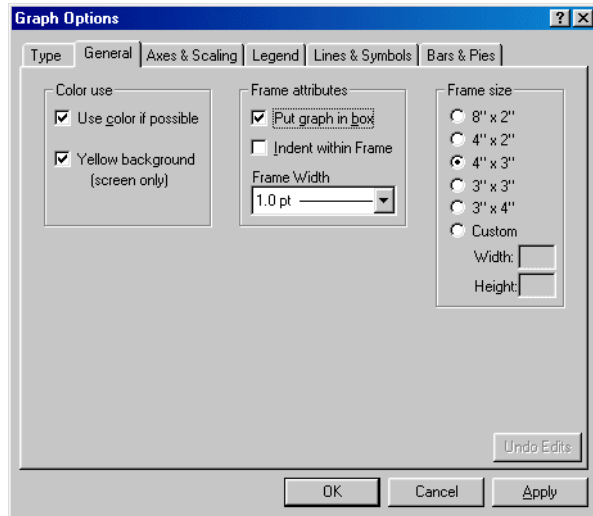
The **Stack lines & bars** option plots the series so that each line represents the sum of all preceding series. In other words, each series value is the vertical distance between each successive line. Note that if some (but not all) of the series have missing values, the sum will be cumulated with missing values replaced by zeros.

Once you have selected a graph type, click on **Apply** to change the graph type.

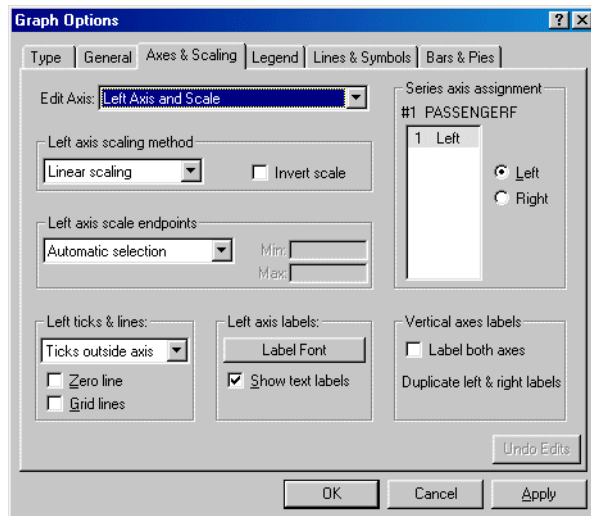
Changing Graph Size, Axes, Scaling, and Legend

The **General** tab controls basic display characteristics of the graph, including color usage, framing style, and perhaps most importantly, graph size. You can control the aspect ratio of your graph using the pre-defined ratios, or you can input a custom set of dimensions. Note that the values are displayed in “virtual inches”.

Be aware that if you added text in the graph with user specified (absolute) position, changing the graph frame size may change the *relative* position of the text in the graph.



Go change or edit axes, select the **Axes & Scaling** tab. Depending on its type, a graph can have up to four axes: left, bottom, right, and top. Each series is assigned an axis as displayed in the upper right combo box. You may change the assigned axis by first highlighting the series and then clicking on one of the available axis buttons. For example, to plot several series with a common scale, you should assign all series to the same axis. To plot two series with a dual left-right scale, assign different axes to the two series.



To edit an axis, select the desired axis from the drop down menu at the top of the dialog. For the Time/Observation Plot type, you may edit the bottom axis to control how the dates/observations are labeled.

To edit the graph legends, select the **Legend** tab. Note that if you place the legend using user specified (absolute) position, the relative position of the legend may change when you change the graph frame size.

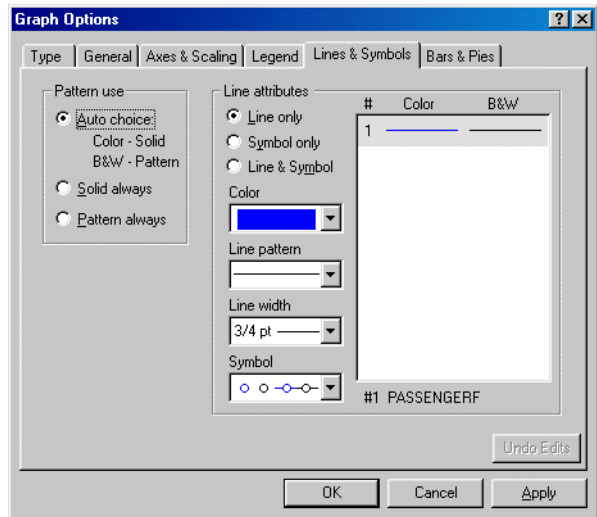
Customizing Lines & Symbols / Bars & Pies

The **Lines & Symbols** tab provides you with control over the drawing of all lines and symbols corresponding to the data in your graph.

You can choose to display lines, symbols, or both, and can customize the color, width, pattern, and symbol usage.

Once you make your choices, click on **Apply** to see the effect of the new settings. The current line and symbol settings for each of the graphs will be displayed in the window on the right hand side of the dialog.

The similar **Bars & Pies** tab allows you to control the display characteristics of your bar of pie graph. Among the elements you can set are the color, shading, and labeling of the graph elements.



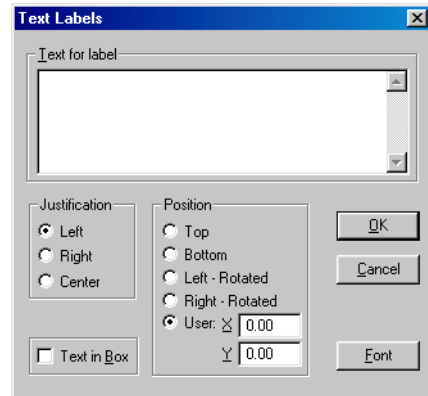
Adding and Editing Text

You can customize a graph by adding one or more lines of text anywhere in the graph. This can be useful for labeling a particular observation or period, or for adding titles or remarks to the graph. To add new text, simply click on the **AddText** button in the toolbar or select **Procs/Add text...**

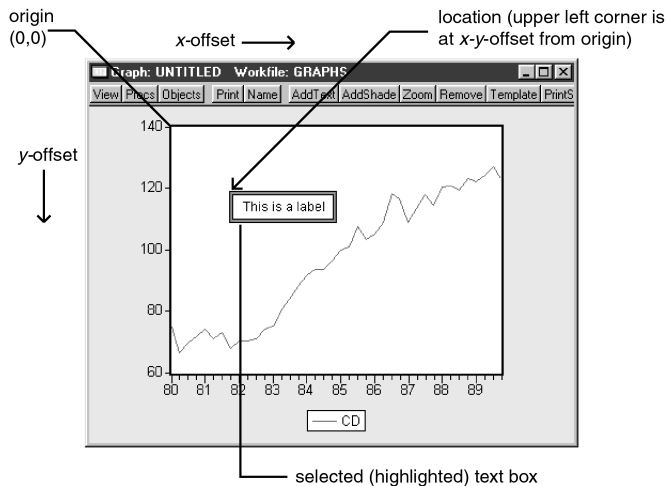
To modify an existing text, simply double click on the text. The Text Label dialog is displayed:

Enter the text you want to display in the large edit field. Spacing and capitalization (upper and lower case letters) will be preserved. If you want to enter more than one line, press the Enter key after each line.

- The **Justification** options determine how multiple lines will be aligned relative to each other.
- **Text in Box** puts the label in a rectangle box.
- **Font** allows you to select the fonts for the label.



The first four options in **Position** place the text at the indicated (relative) position outside the graph. You can also place the text by specifying its coordinates. Coordinates are set in virtual inches, with the origin at the upper left-hand corner of the graph.



The X-axis position increases as you move to the right of the origin, while the Y-axis increases as you move down from the origin. By default graphs are 4×3 virtual inches except scatter diagrams, which are 3×3 virtual inches. Thus, the $X = 4$, $Y = 3$ position refers to the lower right hand corner of the graph. Labels will be placed with the upper left-hand corner of the enclosing box at the specified coordinate.

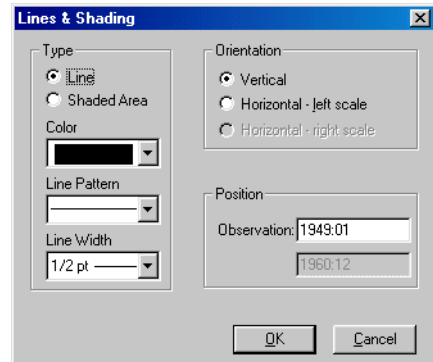
You can change the position of text added to the graph by selecting the text box and dragging it to the position you choose. After dragging to the desired position, you may double click on the text to bring up the Table Labels dialog and to check the coordinates of that position or to make changes to the text. Note that if you specify the text position using coordinates, the *relative* position of the text may change when you change the graph frame size.

Drawing Lines and Shades

You may draw lines or add a shaded area to the graph. From a graph object, click on the **Shade/Lines** button in the toolbar or select **Procs/Add shading....** The Lines & Shading dialog will appear:

Select whether you want to draw a line or add a shaded area and enter the appropriate information to position the line or shaded area horizontally or vertically. If you select **Vertical**, EViews will prompt you to position the line or shaded area at a given observation. If you select **Horizontal**, you must provide a data value at which to draw the line or shaded area.

You should also use this dialog to choose line patterns, width, and colors using the drop down menus.



To modify an existing line or shaded area, simply double click on it to bring up the dialog.

Removing Graph Elements

The **Remove** button in the graph toolbar removes the selected element of a frozen graph. For example, to remove text that you have placed on the graph, click on the text. A border will appear around the text. Press the **Remove** button and it will disappear. The same method can be applied to legends, scales, and shading.

Graph Templates

Having put a lot of effort into getting a graph to look just the way you want it, you may want to use the same options in another graph. EViews allows you to use any graph as a *template* for a new or existing graph. You can think of a template as a graph style that can be applied to other graphs.

First, name the graph object that you want to use as template. Then click the **Template** button in the toolbar of the graph object to which you wish to apply the template. Type the name of the graph object to use as a template. There are two template options:

- **Copy options** applies only the Graph Options settings of the template graph to the graph you are working on.
- **Copy options, text and shading** copies any text labels and shading in the template to the graph as well as its options.

If you select **Template** from a multiple graph window, the template options will be applied to each graph in the multiple graph object.

Multiple Graphs

Some views are made up of multiple graphs. Like single graph views, these may be turned into graph objects by freezing. For example, the impulse response view of a VAR can display multiple graphs in a single view.

You may also create a graph object containing multiple graphs by combining existing named graph objects. Simply select the desired graphs and then double click on any one of the highlighted names. An alternative method of combining graphs is to select **Quick/Show...** and enter the names of the graphs.

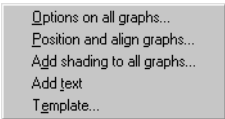
There are two ways to work with a multiple graph. You may change the settings for the multiple graph as a whole, or you may work with an individual graph component of the multiple graph.

Working With Multiple Graphs

EViews makes it easy to work with all of the graphs in a multiple graph. Simply select **Procs** from the graph menu and EViews will display a menu prompting you for additional choices.

These menu items set options that apply to all graphs in the graph object.

- To set a common graph attribute to all graphs, select the **Options on all graphs...** After selecting the desired options, check the **Apply page to all graphs** checkbox at the bottom of the tab.
- While each single graph in a multiple graph can be freely positioned by dragging the graph, you may wish to globally align graphs in columns and control the overall spacing between graphs. To globally position your graphs, select **Position and align graphs...**
- If all of your graphs share a common axis, you can draw lines or add shading to each graph in the object, by selecting **Add shading to all graphs...**



Options on all graphs...
Position and align graphs...
Add shading to all graphs...
Add text
Template...

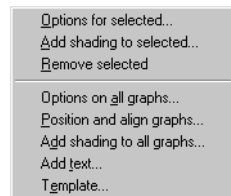
- Selecting **Add text...** allows you to annotate your multiple graph. Note that adding an item to the multiple graph differs from adding it to an individual graph since it will not move with the individual graph.

There is a shortcut method if you merely wish to set the options for all of your graphs. Simply double click anywhere in the background area of your graph, and EViews will open the Multiple Graph Options dialog.

Working with Individual Graphs

You may change the options for an individual graph in the usual fashion by double clicking on the graph to display the options dialog.

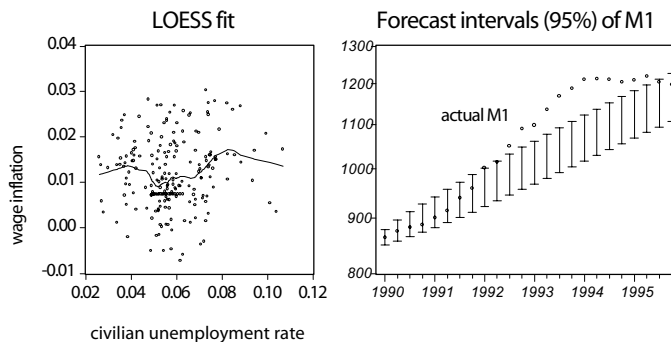
You can also perform various operations on individual graphs. Click on the graph and EViews will confirm the selection by surrounding the individual graph with a blue border. Select **Procs** or right mouse click to display a menu that allows you to set options, add shading or remove the selected graph. Deleting the selected graph can also be performed by pressing the **Remove** button on the graph toolbar.



In addition, you can place each individual graph at an arbitrary position by simply dragging the individual graph to the desired location.

An Example

Here is an example of an annotated and customized combined graph created in EViews and copy-and-pasted into a word processor:



Printing Graphs

Clicking on the **Print** button on the graph view or graph object window toolbar will print the graph. You can control printing options using **File/Print Setup** on the main EViews menu.

Most of the options are self-explanatory. If you wish to print your graph in color using your color printer, make certain that the **Print in color** box is checked. Conversely, if you are printing to a black and white printer, you should make certain this box is not checked so that EViews will substitute line patterns for colors.

Printing Graphs as PostScript Files

EViews uses the standard Windows printer services for all of its printing. This means that the standard Windows methods of setting up printing to a file will work. Under Windows 95/98/Me, it is very simple to set this up. From the taskbar, choose **Start/Settings/Printers**. Double click on **Add Printer**, click **Next**, click on **Local**, and select a PostScript printer. We have found that selecting the HP LaserJet 4M or the Apple LaserWriter Pro drivers (by clicking on the manufacturer, and then the printer name) appears to work well, though you may wish to experiment with different drivers depending upon your printing hardware.

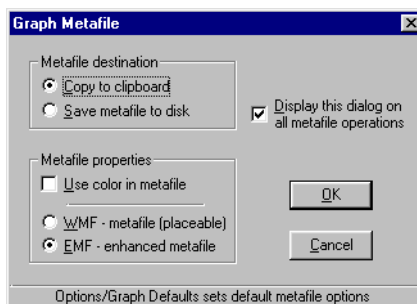
Once you have selected a printer, select FILE: to instruct Windows to print to a file and then tell Windows whether you wish this to be the default printer. Windows will then create a new printer icon representing file capture of the postscript. This “printer” may be selected from within any Windows application including EViews.

Copying Graphs to Other Windows Programs

You can incorporate an EViews graph view directly into a document in your Windows word processor. To do this, first activate the object window containing the graph you wish to move; click anywhere in the window so that the titlebar changes to a bright color. Then click **Edit/Copy** on the EViews main menu; the Copy Graph as Metafile dialog box appears.

You can copy the graph to the Windows clipboard or to a disk file in Windows metafile (WMF) or enhanced metafile (EMF) formats. You can request that the graph be in color and that its lines be in bold. We recommend that you copy graphs in black-and-white unless you will be printing to a color printer.

If you copy a graph to the clipboard, you can switch to your word processor and paste the



graph into your document. Standard programs such as Word or WordPerfect will give you a graph which can be sized, positioned, and modified within the program. You can also paste graphs into drawing programs, and make further modifications before pasting into your word processor or other software.

You can choose to hide this copy dialog so that you use the default settings for subsequent operations by unchecking the **Display this dialog...** box. If you wish to change the default settings or to turn on or off the display of the copy dialog, you should go to the Metafile tab of the global Graph options (**Options/Graphics Defaults...**).

When working with graph objects, EViews also allows you to create a metafile using a proc. Simply click on **Procs/Save graph as metafile...** and follow the dialogs to name and save the file.

Graph Commands

For those of you who wish to automate some of these procedures, for example to produce a regular report, EViews allows you to perform extensive graph customization from the command line or using programs. See “Graph” (p. 25) in the *Command and Programming Reference* for additional details.

Tables

Any views that are not graphical are displayed either as tables or text windows. Tables contain formatted text that is aligned in columns and rows. Examples of views displayed in tables are the Spreadsheet views and the Estimation Output views. Just as there are two types of graphs, there are two types of tables: table views and table objects. Table objects can be created by freezing table views. As with graph objects, table objects are “not live” in the sense that they do not reflect the current contents of the underlying object, but are based instead upon the contents of the object at the time the object was frozen.

Table Options

As with graph objects, there are editing options for tables that are more extensive than for views. These options are available from the **Procs** menu or as buttons on the table window toolbar. When you edit a table, you either highlight the cells you want to edit or refer to a rectangle specified by upper-left and lower-right cells. Cells are identified by the column letter and row number displayed in the margins of the table. If you highlight the cells before you push the editing button, the dialogs will propose that editing be applied to the highlighted cells.

- **Font** allows you to choose a font to be used in the table.
- **Insert-Delete (InsDel)** inserts or deletes rows or columns at the desired position in the table.

- **Column Width (Width)** changes the width of the columns in the table. The width is measured by the number of characters and depends on the font used.
- **Number Cell Format (Numbers)** controls the format for some or all of the numbers in the table. **Fixed characters** specifies the total number of characters to display, while **Fixed decimal** specifies the number of digits to the right of the decimal point. Note that the digits may not be visible if the cell width is not wide enough; use **Column Width** to increase the cell width. **Justification** entries set in the **Number Cell Format** dialog only apply to numbers. You should use **Justify** for justification of all selected cells.
- **Justification (Justify)** justifies all numbers and text for the selected cells.
- **Horizontal Lines (Lines)** adds or removes (double) lines from the specified cells. Note that lines will overlap with any other existing text or numbers in the cell. To place a line by itself, insert an empty row by clicking **InsDel**.
- **Grid + /-** toggles on or off the grid marking the cells in the table.
- **Title** allows you to put a header title at the top center of the table. This is different from **Name**, which provides the object name for the table in the workbook.
- **Edit + /-** toggles on or off edit mode that allows you edit text or numbers in the table.

Table Commands

For those of you who wish to automate some of these procedures, say to produce a monthly report, EViews allows you to perform considerable customization from the command line or using programs. For details, see [Chapter 5, “Working with Tables”](#), on [page 79](#) of the *Command and Programming Reference*.

Copying Tables to Other Windows Programs

You can also cut-and-paste a table into spreadsheet or word processing software. Highlight the area of the table you want to copy and then choose **Edit/Copy** from the main menu. A dialog box will provide you with the option of copying the numbers as they appear in the table, or at the highest internal precision. After you make a choice, switch to your application and select **Edit/Paste**.

EViews copies tables as rich text format (RTF) files, allowing you to preserve the formatting information that is built into the table. Thus, if you copy-and-paste a table from EViews into Microsoft Word or another program which supports RTF, you will create a nicely formatted table containing your results:

Dependent Variable: M1
 Method: Least Squares
 Date: 09/07/97 Time: 12:51
 Sample(adjusted): 1952:2 1979:4
 Included observations: 111 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-4.881480	1.532169	-3.185993	0.0019
PCGDPD	-0.033838	0.202541	-0.167066	0.8676
GDPD	0.008857	0.005484	1.615040	0.1093
M1(-1)	1.014426	0.013662	74.25210	0.0000
RD	-0.432699	0.378003	-1.144697	0.2549
R-squared	0.998632	Mean dependent var		214.6029
Adjusted R-squared	0.998581	S.D. dependent var		85.66340
S.E. of regression	3.227174	Akaike info criterion		5.225090
Sum squared resid	1103.953	Schwarz criterion		5.347140
Log likelihood	-284.9925	F-statistic		19350.15
Durbin-Watson stat	2.770462	Prob(F-statistic)		0.000000

Some word processors provide the option of pasting the contents of the clipboard as unformatted files. If you wish to paste the table as unformatted text, you should select **Edit/Paste Special**.

Text Objects

Some output views have no formatting and are simple displays of text information. Examples are representations of an equation and results from X-11 seasonal adjustment. If you freeze one of these views, you will create a text object.

You can also create a blank text object by selecting **Objects/New Object/Text** or by simply typing “text” in the command window. Text objects may be used whenever you wish to capture textual data that does not contain any formatting information.

Commands

The `freeze` command freezes the specified view of the named object. Provide a name for the frozen object in parentheses after the `freeze` command. For example, to freeze the histogram view of a series named `LWAGE` to a graph object named `LW_HIST`, type

```
freeze(lw_hist) lwage.hist
```

To freeze the simple scatter diagram of the series in the group `GRP1` to a graph named `GRA1`, type

```
freeze(gra1) grp1.scat
```

To combine two graph objects named `GRA1` and `GRA2` into a graph object named `BIGGRA`, type

```
freeze(biggra) gra1 gra2
```

See the *Command and Programming Reference* for a more complete discussion of these and related commands.

Part III. Basic Single Equation Analysis

The following chapters describe the EViews features for basic single equation analysis.

- [Chapter 11, “Basic Regression”, beginning on page 259](#) outlines the basics of ordinary least squares estimation in EViews.
- [Chapter 12, “Additional Regression Methods”, on page 279](#) discusses weighted least squares, two-stage least squares and nonlinear least square estimation techniques.
- [Chapter 13, “Time Series Regression”, on page 303](#) describes single equation regression techniques for the analysis of time series data: testing for serial correlation, estimation of ARMAX and ARIMAX models, using polynomial distributed lags, and unit root tests for nonstationary time series.
- [Chapter 14, “Forecasting from an Equation”, beginning on page 343](#) outlines the fundamentals of using EViews to forecast from estimated equations.
- [Chapter 15, “Specification and Diagnostic Tests”, beginning on page 367](#) describes specification testing in EViews.

Additional single equation techniques for autoregressive conditional heteroskedasticity, and discrete and limited dependent variable models are described in Part IV. Part V documents multiple equation analysis.

Chapter 11. Basic Regression

Single equation regression is one of the most versatile and widely used statistical techniques. Here, we describe the use of basic regression techniques in EViews: specifying and estimating a regression model, performing simple diagnostic analysis, and using your estimation results in further analysis.

Subsequent chapters discuss testing and forecasting, as well as more advanced and specialized techniques such as weighted least squares, two-stage least squares (TSLS), nonlinear least squares, ARIMA/ARIMAX models, generalized method of moments (GMM), GARCH models, and qualitative and limited dependent variable models. These techniques and models all build upon the basic ideas presented in this chapter.

You will probably find it useful to own an econometrics textbook as a reference for the techniques discussed in this and subsequent documentation. Standard textbooks that we have found to be useful are listed below (in generally increasing order of difficulty):

- Pindyck and Rubinfeld (1991), *Econometric Models and Economic Forecasts*, 3rd edition.
- Johnston and DiNardo (1997), *Econometric Methods*, 4th Edition.
- Greene (1997), *Econometric Analysis*, 3rd Edition.
- Davidson and MacKinnon (1993), *Estimation and Inference in Econometrics*.

Where appropriate, we will also provide you with specialized references for specific topics.

Equation Objects

Single equation regression estimation in EViews is performed using the *equation object*. To create an equation object in EViews: select **Objects/New Object/Equation** or **Quick/Estimate Equation...** from the main menu, or simply type the keyword `equation` in the command window.

Next, you will specify your equation in the Equation Specification dialog box that appears, and select an estimation method. Below, we provide details on specifying equations in EViews. EViews will estimate the equation and display results in the equation window.

The estimation results are stored as part of the equation object so they can be accessed at any time. Simply open the object to display the summary results, or to access EViews tools for working with results from an equation object. For example, you can retrieve the sum-of-squares from any equation, or you can use the estimated equation as part of a multi-equation model.

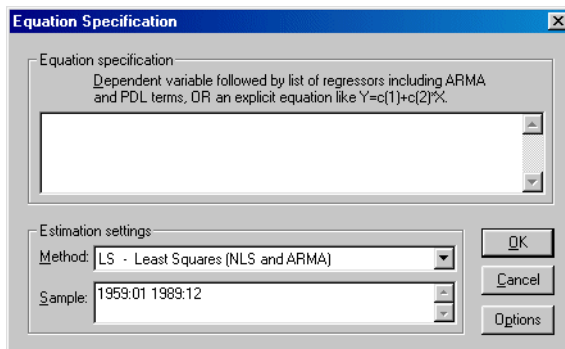
Specifying an Equation in EViews

When you create an equation object, a specification dialog box is displayed.

You need to specify three things in this dialog: the equation specification, the estimation method, and the sample to be used in estimation.

In the upper edit box, you can specify the equation: the dependent (left-hand side) and independent (right-hand side) variables and the functional form. There are two basic ways

of specifying an equation: “by list” and “by formula” or “by expression”. The list method is easier but may only be used with unrestricted linear specifications; the formula method is more general and must be used to specify nonlinear models or models with parametric restrictions.



Specifying an Equation by List

The simplest way to specify a linear equation is to provide a list of variables that you wish to use in the equation. First, include the name of the dependent variable or expression, followed by a list of explanatory variables. For example, to specify a linear consumption function, CS regressed on a constant and INC, type the following in the upper field of the Equation Specification dialog:

```
cs c inc
```

Note the presence of the series name C in the list of regressors. This is a built-in EViews series that is used to specify a constant in a regression. EViews does not automatically include a constant in a regression so you must explicitly list the constant (or its equivalent) as a regressor. The internal series C does not appear in your workfile, and you may not use it outside of specifying an equation. If you need a series of ones, you can generate a new series, or use the number 1 as an auto-series.

You may have noticed that there is a pre-defined object C in your workfile. This is the *default coefficient vector*—when you specify an equation by listing variable names, EViews stores the estimated coefficients in this vector, in the order of appearance in the list. In the example above, the constant will be stored in C(1) and the coefficient on INC will be held in C(2).

Lagged series may be included in statistical operations using the same notation as in generating a new series with a formula—put the lag in parentheses after the name of the series. For example, the specification:

```
cs cs(-1) c inc
```

tells EViews to regress CS on its own lagged value, a constant, and INC. The coefficient for lagged CS will be placed in C(1), the coefficient for the constant is C(2), and the coefficient of INC is C(3).

You can include a consecutive range of lagged series by using the word “to” between the lags. For example,

```
cs c cs(-1 to -4) inc
```

regresses CS on a constant, CS(-1), CS(-2), CS(-3), CS(-4), and INC. If you don't include the first lag, it is taken to be zero. For example,

```
cs c inc(to -2) inc(-4)
```

regresses CS on a constant, INC, INC(-1), INC(-2), and INC(-4).

You may include auto-series in the list of variables. If the auto-series expressions contain spaces, they should be enclosed in parentheses. For example,

```
log(cs) c log(cs(-1)) ((inc+inc(-1)) / 2)
```

specifies a regression of the natural logarithm of CS on a constant, its own lagged value, and a two period moving average of INC.

Typing the list of series may be cumbersome, especially if you are working with many regressors. If you wish, EViews can create the specification list for you. First, highlight the dependent variable in the workfile window by single clicking on the entry. Next, CTRL-click on each of the explanatory variables to highlight them as well. When you are done selecting all of your variables, double click on any of the highlighted series, and select **Open/Equation...** The Equation Specification dialog box should appear with the names entered in the specification field. The constant C is automatically included in this list; you must delete the C if you do not wish to include the constant.

Specifying an Equation by Formula

You will need to specify your equation using a formula when the list method is not general enough for your specification. Many, but not all, estimation methods allow you to specify your equation using a formula.

An equation formula in EViews is a mathematical expression involving regressors and coefficients. To specify an equation using a formula, simply enter the expression in the dia-

log in place of the list of variables. EViews will add an implicit additive disturbance to this equation and will estimate the parameters of the model using least squares.

When you specify an equation by list, EViews converts this into an equivalent equation formula. For example, the list,

$$\log(cs) \ c \ \log(cs(-1)) \ \log(inc)$$

is interpreted by EViews as,

$$\log(cs) = c(1) + c(2)*\log(cs(-1)) + c(3)*\log(inc)$$

Equations do not have to have a dependent variable followed by an equal sign and then an expression. The “=” sign can be anywhere in the formula, as in:

$$\log(urate) + c(1)*dmr = c(2)$$

The residuals for this equation are given by:

$$\epsilon = \log(urate) - c(1)dmr - c(2). \quad (11.1)$$

EViews will minimize the sum-of-squares of these residuals.

If you wish, you can specify an equation as a simple expression, without a dependent variable and an equal sign. If there is no equal sign, EViews assumes that the entire expression is the disturbance term. For example, if you specify an equation as

$$c(1)*x + c(2)*y + 4*z$$

EViews will find the coefficient values that minimize the sum of squares of the given expression, in this case $(C(1)*X + C(2)*Y + 4*Z)$. While EViews will estimate an expression of this type, since there is no dependent variable, some regression statistics (*e.g.* R-squared) are not reported and the equation cannot be used for forecasting. This restriction also holds for any equation that includes coefficients to the left of the equal sign. For example, if you specify,

$$x + c(1)*y = c(2)*z$$

EViews finds the values of C(1) and C(2) that minimize the sum of squares of $(X + C(1)*Y - C(2)*Z)$. The estimated coefficients will be identical to those from an equation specified using:

$$x = -c(1)*y + c(2)*z$$

but some regression statistics are not reported.

The two most common motivations for specifying your equation by formula are to estimate restricted and nonlinear models. For example, suppose that you wish to constrain the

coefficients on the lags on the variable X to sum to one. Solving out for the coefficient restriction leads to the following linear model with parameter restrictions:

$$y = c(1) + c(2)*x + c(3)*x(-1) + c(4)*x(-2) + (1-c(2)-c(3)-c(4))*x(-3)$$

To estimate a nonlinear model, simply enter the nonlinear formula. EViews will automatically detect the nonlinearity and estimate the model using nonlinear least squares. For details, see “[Nonlinear Least Squares](#)” on page 289.

One benefit to specifying an equation by formula is that you can elect to use a different coefficient vector. To create a new coefficient vector, choose **Objects/New Object...** and select **Matrix-Vector-Coef** from the main menu, type in a name for the coefficient vector, and click **OK**. In the New Matrix dialog box that appears, select **Coefficient Vector** and specify how many rows there should be in the vector. The object will be listed in the workfile directory with the coefficient vector icon (the little α).

You may then use this coefficient vector in your specification. For example, suppose you created coefficient vectors A and BETA, each with a single row. Then you can specify your equation using the new coefficients in place of C:

$$\log(cs) = a(1) + beta(1)*\log(cs(-1))$$

Estimating an Equation in EViews

Estimation Methods

Having specified your equation, you now need to choose an estimation method. Click on the **Method:** entry in the dialog and you will see a drop-down menu listing estimation methods.

Standard, single-equation regression is performed using least squares. The other methods are described in subsequent chapters.

Equations estimated by ordinary least squares and two-stage least squares, GMM, and ARCH can be specified with a formula. Nonlinear equations are not allowed with binary, ordered, censored, and count models, or in equations with ARMA terms.

LS	- Least Squares (NLS and ARMA)
TSLS	- Two-Stage Least Squares (TSNLS and ARMA)
ARCH	- Autoregressive Conditional Heteroskedasticity
GMM	- Generalized Method of Moments
BINARY	- Binary choice (logit, probit, extreme value)
ORDERED	- Ordered choice
CENSORED	- Censored data (tobit)
COUNT	- Integer count data

Estimation Sample

You should also specify the sample to be used in estimation. EViews will fill out the dialog with the current workfile sample, but you can change the sample for purposes of estimation by entering your sample string or object in the edit box (see “[Samples](#)” on page 60 for details). Changing the estimation sample does not affect the current workfile sample.

If any of the series used in estimation contain missing data, EViews will temporarily adjust the estimation sample of observations to exclude those observations (listwise exclusion). EViews notifies you that it has adjusted the sample by reporting the actual sample used in the estimation results:

```
Dependent Variable: Y
Method: Least Squares
Date: 08/19/97 Time: 10:24
Sample(adjusted): 1959:01 1989:12
Included observations: 340
Excluded observations: 32 after adjusting endpoints
```

Here we see the top of an equation output view. EViews reports that it has adjusted the sample. Out of the 372 observations in the period 1959:01–1989:12, EViews uses the 340 observations with observations for all of the relevant variables.

You should be aware that if you include lagged variables in a regression, the degree of sample adjustment will differ depending on whether data for the pre-sample period are available or not. For example, suppose you have nonmissing data for the two series M1 and IP over the period 1959:01–1989:12 and specify the regression as

```
m1 c ip ip(-1) ip(-2) ip(-3)
```

If you set the estimation sample to the period 1959:01–1989:12, EViews adjusts the sample to:

```
Dependent Variable: M1
Method: Least Squares
Date: 08/19/97 Time: 10:49
Sample: 1960:01 1989:12
Included observations: 360
```

since data for IP(-3) are not available until 1959:04. However, if you set the estimation sample to the period 1960:01–1989:12, EViews will not make any adjustment to the sample since all values of IP(-3) are available during the estimation sample.

Some operations, most notably estimation with MA terms and ARCH, do not allow missing observations in the middle of the sample. When executing these procedures, an error message is displayed and execution is halted if an NA is encountered in the middle of the sample. EViews handles missing data at the very start or the very end of the sample range by adjusting the sample endpoints and proceeding with the estimation procedure.

Estimation Options

EViews provides a number of estimation options. These options allow you to weight the estimating equation, to compute heteroskedasticity and auto-correlation robust covari-

ances, and to control various features of your estimation algorithm. These options are discussed in detail in “[Estimation Options](#)” on page 292.

Equation Output

When you click **OK** in the Equation Specification dialog, EViews displays the equation window displaying the estimation output view:

Dependent Variable: LOG(M1)
 Method: Least Squares
 Date: 08/18/97 Time: 14:02
 Sample: 1959:01 1989:12
 Included observations: 372

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1.699912	0.164954	-10.30539	0.0000
LOG(IP)	1.765866	0.043546	40.55199	0.0000
TB3	-0.011895	0.004628	-2.570016	0.0106
R-squared	0.886416	Mean dependent var		5.663717
Adjusted R-squared	0.885800	S.D. dependent var		0.553903
S.E. of regression	0.187183	Akaike info criterion		-0.505429
Sum squared resid	12.92882	Schwarz criterion		-0.473825
Log likelihood	97.00980	F-statistic		1439.848
Durbin-Watson stat	0.008687	Prob(F-statistic)		0.000000

Using matrix notation, the standard regression may be written as:

$$y = X\beta + \epsilon \quad (11.2)$$

where y is a T -dimensional vector containing observations on the dependent variable, X is a $T \times k$ matrix of independent variables, β is a k -vector of coefficients, and ϵ is a T -vector of disturbances. T is the number of observations and k is the number of right-hand side regressors.

In the output above, y is $\log(M1)$, X consists of three variables C, $\log(IP)$, and TB3, where $T = 372$ and $k = 3$.

Coefficient Results

Regression Coefficients

The column labeled “Coefficient” depicts the estimated coefficients. The least squares regression coefficients b are computed by the standard OLS formula

$$b = (X'X)^{-1}X'y \quad (11.3)$$

If your equation is specified by list, the coefficients will be labeled in the “Variable” column with the name of the corresponding regressor; if your equation is specified by formula, EViews lists the actual coefficients, C(1), C(2), etc.

For the simple linear models considered here, the coefficient measures the marginal contribution of the independent variable to the dependent variable, holding all other variables fixed. If present, the coefficient of the C is the constant or intercept in the regression—it is the base level of the prediction when all of the other independent variables are zero. The other coefficients are interpreted as the slope of the relation between the corresponding independent variable and the dependent variable, assuming all other variables do not change.

Standard Errors

The “Std. Error” column reports the estimated standard errors of the coefficient estimates. The standard errors measure the statistical reliability of the coefficient estimates—the larger the standard errors, the more statistical noise in the estimates. If the errors are normally distributed, there are about 2 chances in 3 that the true regression coefficient lies within one standard error of the reported coefficient, and 95 chances out of 100 that it lies within two standard errors.

The covariance matrix of the estimated coefficients is computed as,

$$\text{var}(b) = s^2(X'X)^{-1}; \quad s^2 = \hat{\varepsilon}'\hat{\varepsilon}/(T-k); \quad \hat{\varepsilon} = y - Xb \quad (11.4)$$

where $\hat{\varepsilon}$ is the residual. The standard errors of the estimated coefficients are the square roots of the diagonal elements of the coefficient covariance matrix. You can view the whole covariance matrix by choosing **View/Covariance Matrix**.

t-Statistics

The t -statistic, which is computed as the ratio of an estimated coefficient to its standard error, is used to test the hypothesis that a coefficient is equal to zero. To interpret the t -statistic, you should examine the probability of observing the t -statistic given that the coefficient is equal to zero. This probability computation is described below.

In cases where normality can only hold asymptotically, EViews will report a z -statistic instead of a t -statistic.

Probability

The last column of the output shows the probability of drawing a t -statistic (or a z -statistic) as extreme as the one actually observed, under the assumption that the errors are normally distributed, or that the estimated coefficients are asymptotically normally distributed.

This probability is also known as the p -value or the *marginal significance level*. Given a p -value, you can tell at a glance if you reject or accept the hypothesis that the true coefficient is zero against a two-sided alternative that it differs from zero. For example, if you are performing the test at the 5% significance level, a p -value lower than 0.05 is taken as evidence

to reject the null hypothesis of a zero coefficient. If you want to conduct a one-sided test, the appropriate probability is one-half that reported by EViews.

For the above example output, the hypothesis that the coefficient on TB3 is zero is rejected at the 5% significance level but not at the 1% level. However, if theory suggests that the coefficient on TB3 cannot be positive, then a one-sided test will reject the zero null hypothesis at the 1% level.

The p -values are computed from a t -distribution with $T - k$ degrees of freedom.

Summary Statistics

R-squared

The R-squared (R^2) statistic measures the success of the regression in predicting the values of the dependent variable within the sample. In standard settings, R^2 may be interpreted as the fraction of the variance of the dependent variable explained by the independent variables. The statistic will equal one if the regression fits perfectly, and zero if it fits no better than the simple mean of the dependent variable. It can be negative for a number of reasons. For example, if the regression does not have an intercept or constant, if the regression contains coefficient restrictions, or if the estimation method is two-stage least squares or ARCH.

EViews computes the (centered) R^2 as

$$R^2 = 1 - \frac{\hat{\epsilon}'\hat{\epsilon}}{(y - \bar{y})'(y - \bar{y})}; \quad \bar{y} = \sum_{t=1}^T y_t / T \quad (11.5)$$

where \bar{y} is the mean of the dependent (left-hand) variable.

Adjusted R-squared

One problem with using R^2 as a measure of goodness of fit is that the R^2 will never decrease as you add more regressors. In the extreme case, you can always obtain an R^2 of one if you include as many independent regressors as there are sample observations.

The adjusted R^2 , commonly denoted as \bar{R}^2 , penalizes the R^2 for the addition of regressors which do not contribute to the explanatory power of the model. The adjusted R^2 is computed as

$$\bar{R}^2 = 1 - (1 - R^2) \frac{T-1}{T-k} \quad (11.6)$$

The \bar{R}^2 is never larger than the R^2 , can decrease as you add regressors, and for poorly fitting models, may be negative.

Standard Error of the Regression (S.E. of regression)

The standard error of the regression is a summary measure based on the estimated variance of the residuals. The standard error of the regression is computed as

$$s = \sqrt{\frac{\hat{\epsilon}'\hat{\epsilon}}{T-k}} \quad (11.7)$$

Sum-of-Squared Residuals

The sum-of-squared residuals can be used in a variety of statistical calculations, and is presented separately for your convenience:

$$\hat{\epsilon}'\hat{\epsilon} = \sum_{t=1}^T (y_t - X_t'b)^2 \quad (11.8)$$

Log Likelihood

EViews reports the value of the log likelihood function (assuming normally distributed errors) evaluated at the estimated values of the coefficients. Likelihood ratio tests may be conducted by looking at the difference between the log likelihood values of the restricted and unrestricted versions of an equation.

The log likelihood is computed as

$$l = -\frac{T}{2}(1 + \log(2\pi) + \log(\hat{\epsilon}'\hat{\epsilon}/T)) \quad (11.9)$$

When comparing EViews output to that reported from other sources, note that EViews does not ignore constant terms.

Durbin-Watson Statistic

The Durbin-Watson statistic measures the serial correlation in the residuals. The statistic is computed as

$$DW = \frac{\sum_{t=2}^T (\hat{\epsilon}_t - \hat{\epsilon}_{t-1})^2}{\sum_{t=1}^T \hat{\epsilon}_t^2} \quad (11.10)$$

See Johnston and DiNardo (1997, Table D.5) for a table of the significance points of the distribution of the Durbin-Watson statistic.

As a rule of thumb, if the DW is less than 2, there is evidence of positive serial correlation. The DW statistic in our output is very close to one, indicating the presence of serial correlation in the residuals. See [“Serial Correlation Theory” beginning on page 303](#) for a more

extensive discussion of the Durbin-Watson statistic and the consequences of serially correlated residuals.

There are better tests for serial correlation. In “[Testing for Serial Correlation](#)” on page 304, we discuss the Q -statistic, and the Breusch-Godfrey LM test, both of which provide a more general testing framework than the Durbin-Watson test.

Mean and Standard Deviation (S.D.) of the Dependent Variable

The mean and standard deviation of y are computed using the standard formulae:

$$\bar{y} = \sum_{t=1}^T y_t/T; \quad s_y = \sqrt{\sum_{t=1}^T (y_t - \bar{y})^2 / (T-1)} \quad (11.11)$$

Akaike Information Criterion

The Akaike Information Criterion (AIC) is computed as:

$$\text{AIC} = -2l/T + 2k/T \quad (11.12)$$

where l is the log likelihood (given by [Equation \(11.9\)](#) on page 268).

The AIC is often used in model selection for non-nested alternatives—smaller values of the AIC are preferred. For example, you can choose the length of a lag distribution by choosing the specification with the lowest value of the AIC. See [Appendix F, “Information Criteria”](#), on page 683, for additional discussion.

Schwarz Criterion

The Schwarz Criterion (SC) is an alternative to the AIC that imposes a larger penalty for additional coefficients:

$$\text{SC} = -2l/T + (k \log T)/T \quad (11.13)$$

F-Statistic

The F -statistic reported in the regression output is from a test of the hypothesis that *all* of the slope coefficients (excluding the constant, or intercept) in a regression are zero. For ordinary least squares models, the F -statistic is computed as

$$F = \frac{R^2 / (k-1)}{(1-R^2) / (T-k)} \quad (11.14)$$

Under the null hypothesis with normally distributed errors, this statistic has an F -distribution with $k-1$ numerator degrees of freedom and $T-k$ denominator degrees of freedom.

The p -value given just below the F -statistic, denoted **Prob(F-statistic)**, is the marginal significance level of the F -est. If the p -value is less than the significance level you are testing, say 0.05, you reject the null hypothesis that all slope coefficients are equal to zero. For the example above, the p -value is essentially zero, so we reject the null hypothesis that all of the regression coefficients are zero. Note that the F -est is a joint test so that even if all the t -statistics are insignificant, the F -statistic can be highly significant.

Working With Equation Statistics

The regression statistics reported in the estimation output view are stored with the equation and are accessible through special “@-functions”. You can retrieve any of these statistics for further analysis by using these functions in `genr`, `scalar`, or `matrix` expressions. If a particular statistic is not computed for a given estimation method, the function will return an NA.

There are two kinds of “@-functions”: those that return a scalar value, and those that return matrices or vectors.

Keywords that return scalar values

@aic	Akaike information criterion
@coefcov(i,j)	covariance of coefficient estimates i and j
@coefs(i)	i -th coefficient value
@dw	Durbin-Watson statistic
@f	F -statistic
@hq	Hannan-Quinn information criterion
@jstat	J -statistic — value of the GMM objective function (for GMM)
@logl	value of the log likelihood function
@meandep	mean of the dependent variable
@ncoef	number of estimated coefficients
@r2	R-squared statistic
@rbar2	adjusted R-squared statistic
@regobs	number of observations in regression
@schwarz	Schwarz information criterion
@sddep	standard deviation of the dependent variable
@se	standard error of the regression
@ssr	sum of squared residuals

@stderrs(i)	standard error for coefficient i
@tstats(i)	t -statistic value for coefficient i
c(i)	i -th element of default coefficient vector for equation (if applicable)

Keywords that return vector or matrix objects

@cofcov	matrix containing the coefficient covariance matrix
@coefs	vector of coefficient values
@stderrs	vector of standard errors for the coefficients
@tstats	vector of t -statistic values for coefficients

If you use these functions without reference to an equation object, EViews will use the default equation. For example, the command:

```
series y = @dw
```

creates a series named Y and assigns to every observation, the value of the Durbin-Watson statistic for the default equation.

We strongly recommend, however, that you prepend the name of an equation object and a “.” to the statistic keyword. This instructs EViews to use the appropriate statistic for the named equation. For example:

```
series y = eq1.@dw
```

assigns to Y the value of the Durbin-Watson for the equation EQ1.

Functions that return a vector or matrix object should be assigned to the corresponding object type. For example, you should assign the results from @tstats to a vector:

```
vector tstats = eq1.@tstats
```

and the covariance matrix to a matrix:

```
matrix mycov = eq1.@cov
```

You can also access individual elements of these statistics:

```
scalar pvalue = 1-@cnorm(@abs(eq1.@tstats(4)))
scalar var1 = eq1.@covariance(1,1)
```

For documentation on using vectors and matrices in EViews, see [Chapter 4, “Matrix Language”](#), on page 55 of the *Command and Programming Reference*.

Working with Equations

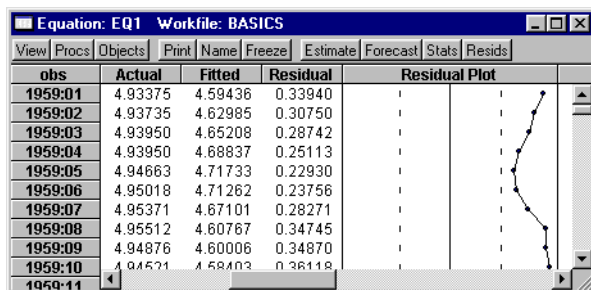
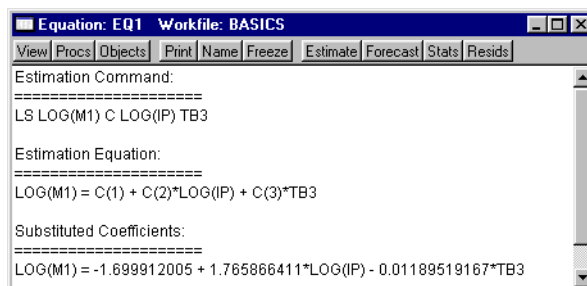
Views of an Equation

- **Representations.** Displays the equation in three forms: EViews command form, as an algebraic equation with symbolic coefficients, and as an equation with the estimated values of the coefficients.

You can cut-and-paste from the representations view into any application that supports the Windows clipboard.

- **Estimation Output.** Displays the equation output results described above.
- **Actual, Fitted, Residual.** These views display the actual and fitted values of the dependent variable and the residuals from the regression in tabular and graphical form. **Actual, Fitted, Residual Table** displays these values in table form.

Note that the actual value is always the sum of the fitted value and the residual. **Actual, Fitted, Residual Graph** displays a standard EViews graph of the actual values, fitted values, and residuals. **Residual Graph** plots only the residuals, while the **Standardized Residual Graph** plots the residuals divided by the estimated residual standard deviation.



- **Gradients and Derivatives...** Provides views which describe the gradients of the objective function and the information about the computation of any derivatives of the regression function. Details on these views are provided in [Appendix E, “Gradients and Derivatives”](#), on page 675.
- **Covariance Matrix.** Displays the covariance matrix of the coefficient estimates as a spreadsheet view. To save this covariance matrix as a matrix object, use the `@cov` function.

- **Coefficient Tests, Residual Tests, and Stability Tests.** These are views for specification and diagnostic tests and are described in detail in [Chapter 15, “Specification and Diagnostic Tests”](#), beginning on page 367.

Procedures of an Equation

- **Specify/Estimate...** brings up the Equation Specification dialog box so that you can modify your specification. You can edit the equation specification, or change the estimation method or estimation sample.
- **Forecast...** forecasts or fits values using the estimated equation. Forecasting using equations is discussed in [Chapter 14](#).
- **Make Residual Series...** saves the residuals from the regression as a series in the workfile. Depending on the estimation method, you may choose from three types of residuals: ordinary, standardized, and generalized. For ordinary least squares, only the ordinary residuals may be saved.
- **Make Regressor Group** creates an untitled group comprised of all the variables used in the equation (with the exception of the constant).
- **Make Gradient Group** creates a group containing the gradients of the objective function with respect to the coefficients of the model.
- **Make Derivative Group** creates a group containing the derivatives of the regression function with respect to the coefficients in the regression function.
- **Make Model** creates an untitled model containing a link to the estimated equation. This model can be solved in the usual manner. See [Chapter 23, “Models”](#), on page 601 for information on how to use models for forecasting and simulations.
- **Update Coefs from Equation** places the estimated coefficients of the equation in the coefficient vector. You can use this procedure to initialize starting values for various estimation procedures.

Default Equation

Following estimation, EViews often holds the estimated coefficients and their covariance matrix, the residuals, and some summary statistics in an untitled equation object. These results are available for use in a variety of subsequent computations including the specification and diagnostic tests described in [Chapter 15, “Specification and Diagnostic Tests”](#), beginning on page 367, and the computation of forecasts and model simulation in [Chapter 14, “Forecasting from an Equation”](#), on page 343 and [Chapter 23, “Models”](#), on page 601.

Untitled equations are not saved with the workfile. You may use the **Name** button on the equation toolbar to name your equation. The equation will be saved with the workfile

when the latter is saved. Once named, you can access the information in the equation at any time, even if you have just estimated several other models, or have not worked with the workfile for a long period of time.

For your convenience, EViews keeps track of a *default equation*. The default equation is the equation that is active or was the most recently active equation. The name of the default equation is shown at the upper right hand corner of the workfile window.

Residuals from an Equation

The residuals from the default equation are stored in a series object called RESID. RESID may be used directly as if it were a regular series, except in estimation.

RESID will be overwritten whenever you estimate an equation and will contain the residuals from the latest estimated equation. To save the residuals from a particular equation for later analysis, you should save them in a different series so they are not overwritten by the next estimation command. For example, you can copy the residuals into a regular EViews series called RES1 by the command

```
series res1 = resid
```

Even if you have already overwritten the RESID series, you can always create the desired series using EViews' built-in procedures if you still have the equation object. If your equation is named EQ1, open the equation window and select **Procs/Make Residual Series**, or enter

```
eq1.makesresid res1
```

to create the desired series.

Regression Statistics

You may refer to various regression statistics through the @-functions described above. For example, to generate a new series equal to FIT plus twice the standard error from the last regression, you can use the command

```
series plus = fit + 2*eq1.@se
```

To get the *t*-statistic for the second coefficient from equation EQ1, you could specify

```
eq1.@tstats(2)
```

To store the coefficient covariance matrix from EQ1 as a named symmetric matrix, you can use the command

```
sym ccov1 = eq1.@cov
```

See [“Keywords that return scalar values” on page 270](#) for additional details.

Storing and Retrieving an Equation

As with other objects, equations may be stored to disk in data bank or database files. You can also fetch equations from these files.

Equations may also be copied-and-pasted to, or from, workfiles or databases.

EViews even allows you to access equations directly from your databases or another workfile. You can estimate an equation, store it in a database, and then use it to forecast in several workfiles.

See [Chapter 3, “EViews Basics”, beginning on page 33](#) and [Chapter 6, “EViews Databases”, beginning on page 107](#) for additional information about objects, databases, and object containers.

Using Estimated Coefficients

The coefficients of an equation are listed in the representations view. By default, EViews will use the C coefficient vector when you specify an equation, but you may explicitly use other coefficient vectors in defining your equation.

These stored coefficients may be used as scalars in generating data. While there are easier ways of generating fitted values (see [“Forecasting from an Equation” on page 343](#)), for purposes of illustration, note that we can use the coefficients to form the fitted values from an equation. The command:

```
series cshat = eq1.c(1) + eq1.c(2)*gdp
```

forms the fitted value of CS, CSHAT, from the OLS regression coefficients and the independent variables from the equation object EQ1.

Note that while EViews will accept a series generating equation which does not explicitly refer to a named equation:

```
series cshat = c(1) + c(2)*gdp
```

and will use the existing values in the C coefficient vector, we strongly recommend that you always use named equations to identify the appropriate coefficients. In general, C will contain the correct coefficient values only immediately following estimation or a coefficient update. Using a named equation, or selecting **Procs/Update coefs from equation**, guarantees that you are using the correct coefficient values.

An alternative to referring to the coefficient vector is to reference the `@coefs` elements of your equation (see [page 270](#)). For example, the examples above may be written as

```
series cshat=eq1.@coefs(1)+eq1.@coefs(2)*gdp
```

EViews assigns an index to each coefficient in the order that it appears in the representations view. Thus, if you estimate the equation

```
equation eq01.ls y=c(10)+b(5)*y(-1)+a(7)*inc
```

where B and A are also coefficient vectors, then

- eq01.@coefs(1) contains C(10)
- eq01.@coefs(2) contains B(5)
- eq01.@coefs(3) contains A(7)

This method should prove useful in matching coefficients to standard errors derived from the @stderrs elements of the equation (see [Chapter 3, “Object, View and Procedure Reference”, beginning on page 19](#) of the *Command and Programming Reference*). The @coefs elements allow you to refer to both the coefficients and the standard errors using a common index.

If you have used an alternative named coefficient vector in specifying your equation, you can also access the coefficient vector directly. For example, if you have used a coefficient vector named BETA, you can generate the fitted values by issuing the commands

```
equation eq02.ls cs=beta(1)+beta(2)*gdp
series cshat=beta(1)+beta(2)*gdp
```

where BETA is a coefficient vector. Again, however, we recommend that you use the @coefs elements to refer to the coefficients of EQ02. Alternatively, you can update the coefficients in BETA prior to use by selecting **Procs/Update coefs from equation** from the equation window. Note that EViews does not allow you to refer to the named equation coefficients EQ02.BETA(1) and EQ02.BETA(2). You must instead use the expressions, EQ02.@COEFS(1) and EQ02.@COEFS(2).

Estimation Problems

Exact Collinearity

If the regressors are very highly collinear, EViews may encounter difficulty in computing the regression estimates. In such cases, EViews will issue an error message “Near singular matrix.” When you get this error message, you should check to see whether the regressors are *exactly* collinear. The regressors are exactly collinear if one regressor can be written as a linear combination of the other regressors. Under exact collinearity, the regressor matrix X does not have full column rank and the OLS estimator cannot be computed.

You should watch out for exact collinearity when you are using dummy variables in your regression. A set of mutually exclusive dummy variables and the constant term are exactly collinear. For example, suppose you have quarterly data and you try to run a regression with the specification

```
y c x @seas(1) @seas(2) @seas(3) @seas(4)
```

EViews will return a “Near singular matrix” error message since the constant and the four quarterly dummy variables are exactly collinear through the relation:

```
c = @seas(1) + @seas(2) + @seas(3) + @seas(4)
```

In this case, simply drop either the constant term or one of the dummy variables.

The textbooks listed above provide extensive discussion of the issue of collinearity.

Commands

To declare a new equation object, follow the equation command with a name for the equation object:

```
equation eq1
```

To estimate an equation by OLS, follow the equation name with a dot and the keyword “ls” or “est”, the name of the dependent variable, and the names of the independent variables, each separated by a space:

```
eq1.ls cs c gdp cpi
```

regresses CS on a constant, GDP, and CPI.

Alternatively, you can specify the equation by a formula with an equal sign:

```
eq1.ls cs = c(1) + c(2)*gdp + c(3)*cpi
```

You can define and estimate an equation in one command:

```
equation eq_sale.ls sales c trend orders industry_growth
```

estimates the specified equation and stores the results in an equation named EQ_SALE.

See [ls \(p. 245\)](#) in the *Command and Programming Reference* for a complete list of commands and options for single equation least squares estimation in EViews.

Chapter 12. Additional Regression Methods

This chapter discusses weighted least squares, heteroskedasticity and autocorrelation consistent covariance estimation, two-stage least squares (TSLS), nonlinear least squares, and generalized method of moments (GMM). Note that most of these methods are also available in systems of equations; see [Chapter 19](#).

Parts of this chapter refer to estimation of models which have autoregressive (AR) and moving average (MA) error terms. These concepts are discussed in greater depth in [Chapter 13](#).

Weighted Least Squares

Suppose that you have heteroskedasticity of known form, and that there is a series w , whose values are proportional to the *reciprocals* of the error standard deviations. You can use weighted least squares, with weight series w , to correct for the heteroskedasticity.

EViews performs weighted least squares by first dividing the weight series by its mean, then multiplying all of the data for each observation by the scaled weight series. The scaling of the weight series is a normalization that has no effect on the parameter results, but makes the weighted residuals more comparable to the unweighted residuals. The normalization does imply, however, that EViews weighted least squares is not appropriate in situations where the scale of the weight series is relevant, as in frequency weighting.

Estimation is then completed by running a regression using the weighted dependent and independent variables to minimize the sum-of-squared residuals

$$S(\beta) = \sum_t w_t^2 (y_t - x_t' \beta)^2 \quad (12.1)$$

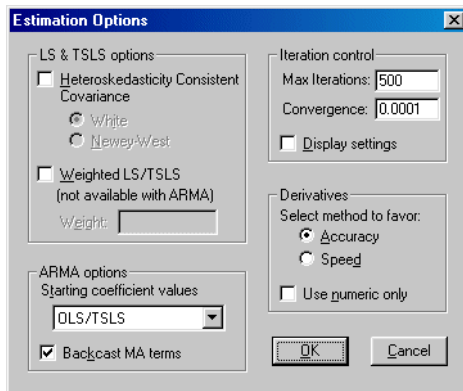
with respect to the k -dimensional vector of parameters β . In matrix notation, let W be a diagonal matrix containing the scaled w along the diagonal and zeroes elsewhere, and let y and X be the usual matrices associated with the left and right-hand side variables. The weighted least squares estimator is

$$b_{WLS} = (X'W'WX)^{-1}X'W'Wy, \quad (12.2)$$

and the estimated covariance matrix is

$$\hat{\Sigma}_{WLS} = s^2(X'W'WX)^{-1}. \quad (12.3)$$

To estimate an equation using weighted least squares, first go to the main menu and select **Quick/Estimate Equation...**, then choose **LS—Least Squares (NLS and ARMA)** from the combo box. Enter your equation specification and sample in the edit boxes, then push the **Options** button and click on the **Weighted LS/TSLS** option.



Fill in the blank after **Weight:** with the name of the series containing your weights, and click on **OK**. Click on **OK** again to accept the dialog and estimate the equation.

Dependent Variable: LOG(X)
 Method: Least Squares
 Date: 10/15/97 Time: 11:10
 Sample(adjusted): 1891 1983
 Included observations: 93 after adjusting endpoints
 Weighting series: POP

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.004233	0.012745	0.332092	0.7406
LOG(X(-1))	0.099840	0.112539	0.887163	0.3774
LOG(W(-1))	0.194219	0.421005	0.461322	0.6457

Weighted Statistics

R-squared	0.016252	Mean dependent var	0.009762
Adjusted R-squared	-0.005609	S.D. dependent var	0.106487
S.E. of regression	0.106785	Akaike info criterion	-1.604274
Sum squared resid	1.026272	Schwarz criterion	-1.522577
Log likelihood	77.59873	F-statistic	0.743433
Durbin-Watson stat	1.948087	Prob(F-statistic)	0.478376

Unweighted Statistics

R-squared	-0.002922	Mean dependent var	0.011093
Adjusted R-squared	-0.025209	S.D. dependent var	0.121357
S.E. of regression	0.122877	Sum squared resid	1.358893
Durbin-Watson stat	2.086669		

EViews will open an output window displaying the standard coefficient results, and both weighted and unweighted summary statistics. The weighted summary statistics are based on the fitted residuals, computed using the weighted data,

$$\tilde{u}_t = w_t(y_t - x_t' b_{WLS}). \quad (12.4)$$

The unweighted summary results are based on the residuals computed from the original (unweighted) data,

$$u_t = y_t - x_t' b_{WLS}. \quad (12.5)$$

Following estimation, the unweighted residuals are placed in the RESID series.

If the residual variance assumptions are correct, the weighted residuals should show no evidence of heteroskedasticity. If the variance assumptions are correct, the unweighted residuals should be heteroskedastic, with the reciprocal of the standard deviation of the residual at each period t being proportional to w_t .

The weighting option will be ignored in equations containing ARMA specifications. Note also that the weighting option is not available for binary, count, censored and truncated, or ordered discrete choice models.

Heteroskedasticity and Autocorrelation Consistent Covariances

When the form of heteroskedasticity is not known, it may not be possible to obtain efficient estimates of the parameters using weighted least squares. OLS provides consistent parameter estimates in the presence of heteroskedasticity but the usual OLS standard errors will be incorrect and should not be used for inference.

Before we describe the techniques for HAC covariance estimation, note that:

- Using the White heteroskedasticity consistent or the Newey-West HAC consistent covariance estimates does not change the point estimates of the parameters, only the estimated standard errors.
- There is nothing to keep you from combining various methods of accounting for heteroskedasticity and serial correlation. For example, weighted least squares estimation might be accompanied by White or Newey-West covariance matrix estimates.

Heteroskedasticity Consistent Covariances (White)

White (1980) has derived a heteroskedasticity consistent covariance matrix estimator which provides correct estimates of the coefficient covariances in the presence of heteroskedasticity of unknown form. The White covariance matrix is given by:

$$\hat{\Sigma}_W = \frac{T}{T-k} (X'X)^{-1} \left(\sum_{t=1}^T u_t^2 x_t x_t' \right) (X'X)^{-1}, \quad (12.6)$$

where T is the number of observations, k is the number of regressors, and u_t is the least squares residual.

EViews provides you the option to use the White covariance estimator in place of the standard OLS formula. Open the equation dialog and specify the equation as before, then push the **Options** button. Next, click on the check box labeled **Heteroskedasticity Consistent**

Covariance and click on the **White** radio button. Accept the options and click **OK** to estimate the equation.

EViews will estimate your equation and compute the variances using White's covariance estimator. You can always tell when EViews is using White covariances, since the output display will include a line to document this fact:

Dependent Variable: LOG(X)
 Method: Least Squares
 Date: 10/15/97 Time: 11:11
 Sample(adjusted): 1891 1983
 Included observations: 93 after adjusting endpoints
 Weighting series: POP
 White Heteroskedasticity-Consistent Standard Errors & Covariance

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.004233	0.012519	0.338088	0.7361
LOG(X(-1))	0.099840	0.137262	0.727369	0.4689
LOG(W(-1))	0.194219	0.436644	0.444800	0.6575

HAC Consistent Covariances (Newey-West)

The White covariance matrix described above assumes that the residuals of the estimated equation are serially uncorrelated. Newey and West (1987) have proposed a more general covariance estimator that is consistent in the presence of both heteroskedasticity and autocorrelation of unknown form. The Newey-West estimator is given by

$$\hat{\Sigma}_{NW} = \frac{T}{T-k} (X'X)^{-1} \hat{\Omega} (X'X)^{-1}, \quad (12.7)$$

where

$$\hat{\Omega} = \frac{T}{T-k} \left\{ \sum_{t=1}^T u_t^2 x_t x_t' + \sum_{v=1}^q \left[\left(1 - \frac{v}{q+1}\right) \sum_{t=v+1}^T (x_t u_t u_{t-v} x_{t-v}' + x_{t-v} u_{t-v} u_t x_t') \right] \right\} \quad (12.8)$$

and q , the truncation lag, is a parameter representing the number of autocorrelations used in evaluating the dynamics of the OLS residuals u_t . Following the suggestion of Newey and West, EViews sets q to

$$q = \text{floor}(4(T/100)^{2/9}). \quad (12.9)$$

To use the Newey-West method, push the **Options** button in the estimation dialog box. Check the box labeled **Heteroskedasticity Consistent Covariance** and press the **Newey-West** radio button.

Two-stage Least Squares

A fundamental assumption of regression analysis is that the right-hand side variables are uncorrelated with the disturbance term. If this assumption is violated, both OLS and weighted LS are biased and inconsistent.

There are a number of situations where some of the right-hand side variables are correlated with disturbances. Some classic examples occur when:

- There are endogenously determined variables on the right-hand side of the equation.
- Right-hand side variables are measured with error.

For simplicity, we will refer to variables that are correlated with the residuals as *endogenous*, and variables that are not correlated with the residuals as *exogenous* or *predetermined*.

The standard approach in cases where right-hand side variables are correlated with the residuals is to estimate the equation using *instrumental variables* regression. The idea behind instrumental variables is to find a set of variables, termed *instruments*, that are both (1) correlated with the explanatory variables in the equation, and (2) uncorrelated with the disturbances. These instruments are used to eliminate the correlation between right-hand side variables and the disturbances.

Two-stage least squares (TSLS) is a special case of instrumental variables regression. As the name suggests, there are two distinct stages in two-stage least squares. In the first stage, TSLS finds the portions of the endogenous and exogenous variables that can be attributed to the instruments. This stage involves estimating an OLS regression of each variable in the model on the set of instruments. The second stage is a regression of the original equation, with all of the variables replaced by the fitted values from the first-stage regressions. The coefficients of this regression are the TSLS estimates.

You need not worry about the separate stages of TSLS since EViews will estimate both stages simultaneously using instrumental variables techniques. More formally, let Z be the matrix of instruments, and let y and X be the dependent and explanatory variables. Then the coefficients computed in two-stage least squares are given by,

$$b_{TSLS} = (X'Z(Z'Z)^{-1}Z'X)^{-1}X'Z(Z'Z)^{-1}Z'y, \quad (12.10)$$

and the estimated covariance matrix of these coefficients is given by

$$\hat{\Sigma}_{TSLS} = s^2(X'Z(Z'Z)^{-1}Z'X)^{-1}, \quad (12.11)$$

where s^2 is the estimated residual variance (square of the standard error of the regression).

Estimating TSLS in EViews

To use two-stage least squares, open the equation specification box by choosing **Object/New Object/Equation...** or **Quick/Estimate Equation...**. Choose **TSLS** from the **Method:** combo box and the dialog will change to include an edit window where you will list the instruments. In the edit boxes, specify your dependent variable and independent variables and the list of instruments.

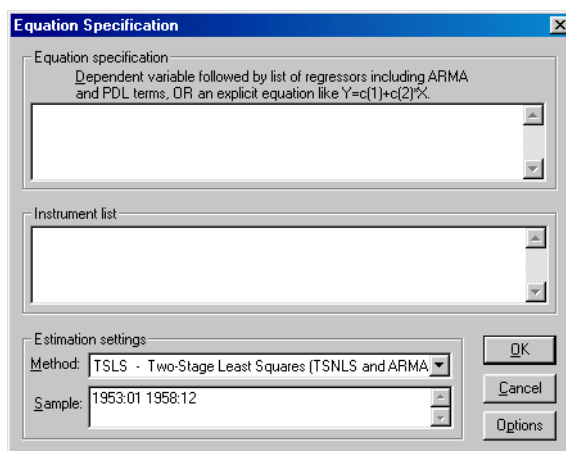
There are a few things to keep in mind as you enter your instruments:

- In order to calculate TSLS estimates, your specification must satisfy the *order condition* for identification, which says that there must be at least as many instruments as there are coefficients in your equation. There is an additional rank condition which must also be satisfied. See Davidson and MacKinnon (1994) and Johnston and DiNardo (1997) for additional discussion.
- For econometric reasons that we will not pursue here, any right-hand side variables that are not correlated with the disturbances can be used as instruments.
- The constant, C , is always a suitable instrument, so EViews will add it to the instrument list if you omit it.

For example, suppose you are interested in estimating a consumption equation relating consumption (CONS) to gross domestic product (GDP), lagged consumption (CONS(-1)), a trend variable (TIME) and a constant (C). GDP is endogenous and therefore correlated with the residuals. You may, however, believe that government expenditures (G), the log of the money supply (LM), lagged consumption, TIME, and C, are exogenous and uncorrelated with the disturbances, so that these variables may be used as instruments. Your equation specification is then,

```
cons c gdp cons(-1) time
```

and the instrument list is,



```
c gov cons(-1) time lm
```

This specification satisfies the order condition for identification, which requires that there are at least as many instruments (five) as there are coefficients (four) in the equation specification.

Furthermore, all of the variables in the consumption equation that are believed to be uncorrelated with the disturbances, (CONS(-1), TIME, and C), appear both in the equation specification and in the instrument list. Note that listing C as an instrument is redundant, since EViews automatically adds it to the instrument list.

Output from TSLS

Below, we present TSLS estimates from a regression of LOG(CS) on a constant and LOG(GDP), with the instrument list C LOG(CS(-1)) LOG(GDP(-1)):

Dependent Variable: LOG(CS)
 Method: Two-Stage Least Squares
 Date: 10/15/97 Time: 11:32
 Sample(adjusted): 1947:2 1995:1
 Included observations: 192 after adjusting endpoints
 Instrument list: C LOG(CS(-1)) LOG(GDP(-1))

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1.209268	0.039151	-30.88699	0.0000
LOG(GDP)	1.094339	0.004924	222.2597	0.0000
R-squared	0.996168	Mean dependent var		7.480286
Adjusted R-squared	0.996148	S.D. dependent var		0.462990
S.E. of regression	0.028735	Sum squared resid		0.156888
F-statistic	49399.36	Durbin-Watson stat		0.102639
Prob(F-statistic)	0.000000			

EViews identifies the estimation procedure, as well as the list of instruments in the header. This information is followed by the usual coefficient, t -statistics, and asymptotic p -values.

The summary statistics reported at the bottom of the table are computed using the formulas outlined in [Chapter 11](#). Bear in mind that all reported statistics are only asymptotically valid. For a discussion of the finite sample properties of TSLS, see Johnston and DiNardo (1997, pp. 355–358) or Davidson and MacKinnon (1984, pp. 221–224).

EViews uses the *structural residuals* $u_t = y_t - x_t' b_{TSLS}$ in calculating all of the summary statistics. For example, the standard error of the regression used in the asymptotic covariance calculation is computed as

$$s^2 = \sum_t u_t^2 / (T - k). \quad (12.12)$$

These structural residuals should be distinguished from the *second stage residuals* that you would obtain from the second stage regression if you actually computed the two-stage least squares estimates in two separate stages. The second stage residuals are given by

$\tilde{u}_t = \hat{y}_t - \hat{x}_t' b_{TSLS}$, where the \hat{y}_t and \hat{x}_t are the fitted values from the first-stage regressions.

We caution you that some of the reported statistics should be interpreted with care. For example, since different equation specifications will have different instrument lists, the reported R^2 for TSLS can be negative even when there is a constant in the equation.

Weighted TSLS

You can combine TSLS with weighted regression. Simply enter your TSLS specification as above, then press the **Options** button, select the **Weighted LS/TSLS** option and enter the weighting series.

Weighted two-stage least squares is performed by multiplying all of the data, including the instruments, by the weight variable, and estimating TSLS on the transformed model. Equivalently, EViews then estimates the coefficients using the formula,

$$b_{WTSLS} = (X'W'WZ(Z'W'WZ)^{-1}Z'W'WX)^{-1} \cdot X'W'WZ(Z'W'WZ)^{-1}Z'W'Wy \quad (12.13)$$

The estimated covariance matrix is

$$\hat{\Sigma}_{WTSLS} = s^2(X'W'WZ(Z'W'WZ)^{-1}Z'W'WX)^{-1}. \quad (12.14)$$

TSLS with AR errors

You can adjust your TSLS estimates to account for serial correlation by adding AR terms to your equation specification. EViews will automatically transform the model to a nonlinear least squares problem, and estimate the model using instrumental variables. Details of this procedure may be found in Fair (1984, pp. 210–214). The output from TSLS with an AR(1) specification looks as follows:

Dependent Variable: LOG(CS)
 Method: Two-Stage Least Squares
 Date: 10/15/97 Time: 11:42
 Sample(adjusted): 1947:2 1995:1
 Included observations: 192 after adjusting endpoints
 Convergence achieved after 4 iterations
 Instrument list: C LOG(CS(-1)) LOG(GDP(-1))

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1.420705	0.203266	-6.989390	0.0000
LOG(GDP)	1.119858	0.025116	44.58782	0.0000
AR(1)	0.930900	0.022267	41.80595	0.0000
R-squared	0.999611	Mean dependent var		7.480286
Adjusted R-squared	0.999607	S.D. dependent var		0.462990
S.E. of regression	0.009175	Sum squared resid		0.015909
F-statistic	243139.7	Durbin-Watson stat		1.931027
Prob(F-statistic)	0.000000			
Inverted AR Roots	.93			

The **Options** button in the estimation box may be used to change the iteration limit and convergence criterion for the nonlinear instrumental variables procedure.

First-order AR errors

Suppose your specification is:

$$\begin{aligned}
 y_t &= x_t' \beta + w_t \gamma + u_t \\
 u_t &= \rho_1 u_{t-1} + \epsilon_t
 \end{aligned}
 \tag{12.15}$$

where x_t is a vector of endogenous variables, and w_t is a vector of predetermined variables, which, in this context, may include lags of the dependent variable. z_t is a vector of instrumental variables not in w_t that is large enough to identify the parameters of the model.

In this setting, there are important technical issues to be raised in connection with the choice of instruments. In a widely quoted result, Fair (1970) shows that if the model is estimated using an iterative Cochrane-Orcutt procedure, all of the lagged left- and right-hand side variables (y_{t-1} , x_{t-1} , w_{t-1}) must be included in the instrument list to obtain consistent estimates. In this case, then the instrument list should include

$$(w_t, z_t, y_{t-1}, x_{t-1}, w_{t-1}). \tag{12.16}$$

Despite the fact the EViews estimates the model as a nonlinear regression model, the first stage instruments in TSLS are formed as if running Cochrane-Orcutt. Thus, if you choose to omit the lagged left- and right-hand side terms from the instrument list, EViews will automatically add each of the lagged terms as instruments. This fact is noted in your output.

Higher Order AR errors

The AR(1) result extends naturally to specifications involving higher order serial correlation. For example, if you include a single AR(4) term in your model, the natural instrument list will be

$$(w_t, z_t, y_{t-4}, x_{t-4}, w_{t-4}) \quad (12.17)$$

If you include AR terms from 1 through 4, one possible instrument list is

$$(w_t, z_t, y_{t-1}, \dots, y_{t-4}, x_{t-1}, \dots, x_{t-4}, w_{t-1}, \dots, w_{t-4}) \quad (12.18)$$

Note that while theoretically valid, this instrument list has a large number of overidentifying instruments, which may lead to computational difficulties and large finite sample biases (Fair (1984, p. 214), Davidson and MacKinnon (1993, pp. 222-224)). In theory, adding instruments should always improve your estimates, but as a practical matter this may not be so in small samples.

Examples

Suppose that you wish to estimate the consumption function by two-stage least squares, allowing for first-order serial correlation. You may then use two-stage least squares with the variable list,

```
cons c gdp ar(1)
```

and instrument list,

```
c gov log(m1) time cons(-1) gdp(-1)
```

Notice that the lags of both the dependent and endogenous variables (CONS(-1) and GDP(-1)), are included in the instrument list.

Similarly, consider the consumption function,

```
cons c cons(-1) gdp ar(1)
```

A valid instrument list is given by

```
c gov log(m1) time cons(-1) cons(-2) gdp(-1)
```

Here we treat the lagged left and right-hand side variables from the original specification as predetermined and add the lagged values to the instrument list.

Lastly, consider the specification,

```
cons c gdp ar(1) ar(2) ar(3) ar(4)
```

Adding all of the relevant instruments in the list, we have

```
c gov log(m1) time cons(-1) cons(-2) cons(-3) cons(-4) gdp(-1)
gdp(-2) gdp(-3) gdp(-4)
```

TSLS with MA errors

You can also estimate two-stage least squares variable problems with MA error terms of various orders. To account for the presence of MA errors, simply add the appropriate terms to your specification prior to estimation.

Illustration

Suppose that you wish to estimate the consumption function by two-stage least squares, accounting for first-order moving average errors. You may then use two-stage least squares with the variable list,

```
cons c gdp ma(1)
```

and instrument list,

```
c gov log(m1) time
```

EViews will add both first and second lags of CONS and GDP to the instrument list.

Technical Details

Most of the technical details are identical to those outlined above for AR errors. EViews transforms the model that is nonlinear in parameters (employing backcasting, if appropriate) and then estimates the model using nonlinear instrumental variables techniques.

Note that EViews augments the instrument list appropriately by adding lagged left- and right-hand side variables. There is an approximation involved here, however, in a truncation of the lag structure. In principle, each MA term involves an infinite number of AR terms. Clearly it is impossible to add an infinite number of lags to the instrument list. Instead, EViews performs an ad hoc approximation by adding a truncated set of instruments involving the MA order and an additional lag. If for example, you have an MA(5), EViews will add lagged instruments corresponding to lags 5 and 6.

Nonlinear Least Squares

Suppose that we have the regression specification

$$y_t = f(x_t, \beta) + \epsilon_t, \quad (12.19)$$

where f is a general function of the explanatory variables x_t and the parameters β . Least squares estimation chooses the parameter values that minimize the sum of squared residuals:

$$S(\beta) = \sum_t (y_t - f(x_t, \beta))^2 = (y - f(X, \beta))'(y - f(X, \beta)) \quad (12.20)$$

We say that a model is *linear in parameters* if the derivatives of f with respect to the parameters do not depend upon β ; if the derivatives are functions of β , we say that the model is *nonlinear in parameters*.

For example, consider the model given by

$$y_t = \beta_1 + \beta_2 \log L_t + \beta_3 \log K_t + \epsilon_t. \quad (12.21)$$

It is easy to see that this model is linear in its parameters, implying that it can be estimated using ordinary least squares.

In contrast, the equation specification

$$y_t = \beta_1 L_t^{\beta_2} K_t^{\beta_3} + \epsilon_t \quad (12.22)$$

has derivatives that depend upon the elements of β . There is no way to rearrange the terms in this model so that ordinary least squares can be used to minimize the sum-of-squared residuals. We must use nonlinear least squares techniques to estimate the parameters of the model.

Nonlinear least squares minimizes the sum-of-squared residuals with respect to the choice of parameters β . While there is no closed form solution for the parameter estimates, the estimates satisfy the first-order conditions:

$$(G(\beta))'(y - f(X, \beta)) = 0, \quad (12.23)$$

where $G(\beta)$ is the matrix of first derivatives of $f(X, \beta)$ with respect to β (to simplify notation we suppress the dependence of G upon X). The estimated covariance matrix is given by

$$\hat{\Sigma}_{NLLS} = s^2 (G(b_{NLLS})' G(b_{NLLS}))^{-1}. \quad (12.24)$$

where b_{NLLS} are the estimated parameters. For additional discussion of nonlinear estimation, see Pindyck and Rubinfeld (1991, pp. 231-245) or Davidson and MacKinnon (1993).

Estimating NLS Models in EViews

It is easy to tell EViews that you wish to estimate the parameters of a model using nonlinear least squares. EViews automatically applies nonlinear least squares to any regression equation that is nonlinear in its coefficients. Simply select **Object/New Object/Equation**, enter the equation in the equation specification dialog box and click **OK**. EViews will do all of the work of estimating your model using an iterative algorithm.

A full technical discussion of iterative estimation procedures is provided in [Appendix D](#), “Estimation Algorithms and Options”, beginning on page 663.

Specifying Nonlinear Least Squares

For nonlinear regression models, you will have to enter your specification in equation form using EViews expressions that contain direct references to coefficients. You may use elements of the default coefficient vector C (e.g. $C(1)$, $C(2)$, $C(34)$, $C(87)$), or you can define and use other coefficient vectors. For example,

$$y = c(1) + c(2) * (k^{c(3)} + 1^{c(4)})$$

is a nonlinear specification that uses the first through the fourth elements of the default coefficient vector, C .

To create a new coefficient vector, select **Objects/New Object/Matrix-Vector-Coef/Coefficient Vector** in the main menu and provide a name. You may now use this coefficient vector in your specification. For example, if you create a coefficient vector named CF , you can rewrite the specification above as

$$y = cf(11) + cf(12) * (k^{cf(13)} + 1^{cf(14)})$$

which uses the eleventh through the fourteenth elements of CF .

You can also use multiple coefficient vectors in your specification:

$$y = c(11) + c(12) * (k^{cf(1)} + 1^{cf(2)})$$

which uses both C and CF in the specification.

It is worth noting that EViews implicitly adds an additive disturbance to your specification. For example, the input

$$y = (c(1) * x + c(2) * z + 4)^2$$

is interpreted as $y_t = (c(1)x_t + c(2)z_t + 4)^2 + \epsilon_t$, and EViews will minimize

$$S(c(1), c(2)) = \sum_t (y_t - (c(1)x_t + c(2)z_t + 4)^2)^2. \quad (12.25)$$

If you wish, the equation specification may be given by a simple expression that does not include a dependent variable. For example, the input

$$(c(1) * x + c(2) * z + 4)^2$$

is interpreted by EViews as $-(c(1)x_t + c(2)z_t + 4)^2 = \epsilon_t$, and EViews will minimize

$$S(c(1), c(2)) = \sum_t (-(c(1)x_t + c(2)z_t + 4)^2)^2. \quad (12.26)$$

While EViews will estimate the parameters of this last specification, the equation cannot be used for forecasting and cannot be included in a model. This restriction also holds for

any equation that includes coefficients to the left of the equal sign. For example, if you specify

$$x + c(1) * y = z^{c(2)}$$

EViews will find the values of C(1) and C(2) that minimize the sum of squares of the implicit equation

$$x_t + c(1)y_t - z_t^{c(2)} = \epsilon_t, \quad (12.27)$$

but the estimated equation cannot be used in forecasting or included in a model, since there is no dependent variable.

Estimation Options

Starting Values. Iterative estimation procedures require starting values for the coefficients of the model. There are no general rules for selecting starting values for parameters. The closer to the true values the better, so if you have reasonable guesses for parameter values, these can be useful. In some cases, you can obtain good starting values by estimating a restricted version of the model using least squares. In general, however, you will have to experiment in order to find starting values.

EViews uses the values in the coefficient vector at the time you begin the estimation procedure as starting values for the iterative procedure. It is easy to examine and change these coefficient starting values.

To see the starting values, double click on the coefficient vector in the workfile directory. If the values appear to be reasonable, you can close the window and proceed with estimating your model.

If you wish to change the starting values, first make certain that the spreadsheet view of your coefficients is in edit mode, then enter the coefficient values. When you are finished setting the initial values, close the coefficient vector window and estimate your model.

You may also set starting coefficient values from the command window using the PARAM command. Simply enter the PARAM keyword, following by each coefficient and desired value:

```
param c(1) 153 c(2) .68 c(3) .15
```

sets C(1) = 153, C(2) = .68, and C(3) = .15.

See Appendix D, “Estimation Algorithms and Options” on page 663, for further details.

Derivative Methods. Estimation in EViews requires computation of the derivatives of the regression function with respect to the parameters. EViews provides you with the option of computing analytic expressions for these derivatives (if possible), or computing finite difference numeric derivatives in cases where the derivative is not constant. Furthermore, if

numeric derivatives are computed, you can choose whether to favor speed of computation (fewer function evaluations) or whether to favor accuracy (more function evaluations). Additional issues associated with ARIMA models are discussed in “[Estimation Options](#)” on page 318.

Iteration and Convergence Options. You can control the iterative process by specifying convergence criterion and the maximum number of iterations. Press the **Options** button in the equation dialog box and enter the desired values.

EViews will report that the estimation procedure has converged if the convergence test value is below your convergence tolerance. See “[Iteration and Convergence Options](#)” on page 669 for details.

In most cases, you will not need to change the maximum number of iterations. However, for some difficult to estimate models, the iterative procedure will not converge within the maximum number of iterations. If your model does not converge within the allotted number of iterations, simply click on the **Estimate** button, and, if desired, increase the maximum number of iterations. Click on **OK** to accept the options, and click on **OK** to begin estimation. EViews will start estimation using the last set of parameter values as starting values.

These options may also be set from the global options dialog. See [Appendix A](#), “[Estimation Defaults](#)” on page 649.

Output from NLS

Once your model has been estimated, EViews displays an equation output screen showing the results of the nonlinear least squares procedure. Below is the output from a regression of LOG(CS) on C, and the Box-Cox transform of GDP:

Dependent Variable: LOG(CS)
Method: Least Squares
Date: 10/15/97 Time: 11:51
Sample(adjusted): 1947:1 1995:1
Included observations: 193 after adjusting endpoints
Convergence achieved after 80 iterations
LOG(CS)= C(1)+C(2)*(GDP^C(3)-1)/C(3)

	Coefficient	Std. Error	t-Statistic	Prob.
C(1)	2.851780	0.279033	10.22024	0.0000
C(2)	0.257592	0.041147	6.260254	0.0000
C(3)	0.182959	0.020201	9.056824	0.0000
R-squared	0.997252	Mean dependent var		7.476058
Adjusted R-squared	0.997223	S.D. dependent var		0.465503
S.E. of regression	0.024532	Akaike info criterion		-4.562220
Sum squared resid	0.114350	Schwarz criterion		-4.511505
Log likelihood	443.2542	F-statistic		34469.84
Durbin-Watson stat	0.134628	Prob(F-statistic)		0.000000

If the estimation procedure has converged, EViews will report this fact, along with the number of iterations that were required. If the iterative procedure did not converge, EViews will report “Convergence not achieved after” followed by the number of iterations attempted.

Below the line describing convergence, EViews will repeat the nonlinear specification so that you can easily interpret the estimated coefficients of your model.

EViews provides you with all of the usual summary statistics for regression models. Provided that your model has converged, the standard statistical results and tests are *asymptotically* valid.

Weighted NLS

Weights can be used in nonlinear estimation in a manner analogous to weighted linear least squares. To estimate an equation using weighted nonlinear least squares, enter your specification, press the **Options** button and click on the **Weighted LS/TLS** option. Fill in the blank after **Weight:** with the name of the weight series and then estimate the equation.

EViews minimizes the sum of the weighted squared residuals:

$$S(\beta) = \sum_t w_t^2 (y_t - f(x_t, \beta))^2 = (y - f(X, \beta))' W' W (y - f(X, \beta)) \quad (12.28)$$

with respect to the parameters β , where w_t are the values of the weight series and W is the matrix of weights. The first-order conditions are given by

$$(G(\beta))' W' W (y - f(X, \beta)) = 0 \quad (12.29)$$

and the covariance estimate is computed as

$$\hat{\Sigma}_{WNLLS} = s^2 (G(b_{WNLLS})' W' W G(b_{WNLLS}))^{-1}. \quad (12.30)$$

NLS with AR errors

EViews will estimate nonlinear regression models with autoregressive error terms. Simply select **Objects/New Object/Equation...** or **Quick/Estimate Equation...** and specify your model using EViews expressions, followed by an additive term describing the AR correction enclosed in square brackets. The AR term should consist of a coefficient assignment for each AR term, separated by commas. For example, if you wish to estimate

$$\begin{aligned} CS_t &= c_1 + GDP_t^{c_2} + u_t \\ u_t &= c_3 u_{t-1} + c_4 u_{t-2} + \epsilon_t \end{aligned} \quad (12.31)$$

you should enter the specification

$$cs = c(1) + \text{gdp} \wedge c(2) + [\text{ar}(1)=c(3), \text{ar}(2)=c(4)]$$

See “How EViews Estimates AR Models” on page 310 for additional details. EViews does not currently estimate nonlinear models with MA errors, nor does it estimate weighted models with AR terms—if you add AR terms to a weighted nonlinear model, the weighting series will be ignored.

Nonlinear TSLS

Nonlinear two-stage least squares refers to an instrumental variables procedure for estimating nonlinear regression models involving functions of endogenous and exogenous variables and parameters. Suppose we have the usual nonlinear regression model:

$$y_t = f(x_t, \beta) + \epsilon_t, \quad (12.32)$$

where β is a k -dimensional vector of parameters, and x_t contains both exogenous and endogenous variables. In matrix form, if we have $m \geq k$ instruments z_t , nonlinear two-stage least squares minimizes

$$S(\beta) = (y - f(X, \beta))' Z(Z'Z)^{-1} Z'(y - f(X, \beta)) \quad (12.33)$$

with respect to the choice of β .

While there is no closed form solution for the parameter estimates, the parameter estimates satisfy the first-order conditions:

$$G(\beta)' Z(Z'Z)^{-1} Z'(y - f(X, \beta)) = 0 \quad (12.34)$$

with estimated covariance given by

$$\hat{\Sigma}_{TSNLLS} = s^2 (G(b_{TSNLLS})' Z(Z'Z)^{-1} Z' G(b_{TSNLLS}))^{-1}. \quad (12.35)$$

How to Estimate Nonlinear TSLS in EViews

EViews performs the estimation procedure in a single step so that you don't have to perform the separate stages yourself. Simply select **Object/New Object/Equation...** or **Quick/Estimate Equation...** Choose **TSLS** from the **Method:** combo box, enter your nonlinear specification and the list of instruments. Click **OK**.

With nonlinear two-stage least squares estimation, you have a great deal of flexibility with your choice of instruments. Intuitively you want instruments that are correlated with $G(\beta)$. Since G is nonlinear, you may begin to think about using more than just the exogenous and predetermined variables as instruments. Various nonlinear functions of these variables, for example, cross-products and powers, may also be valid instruments. One should be aware, however, of the possible finite sample biases resulting from using too many instruments.

Weighted Nonlinear Two-stage Least Squares

Weights can be used in nonlinear two-stage least squares estimation. Simply add weighting to your nonlinear TSLS specification above by pressing the **Options** button, selecting **Weighted LS/TSLS** option, and entering the name of the weight series.

The objective function for weighted TSLS is,

$$S(\beta) = (y - f(X, \beta))'W'WZ(Z'W'WZ)^{-1}Z'W'W(y - f(X, \beta)). \quad (12.36)$$

The reported standard errors are based on the covariance matrix estimate given by

$$\hat{\Sigma}_{WTSNLLS} = s^2(G(b)'W'WZ(Z'W'WZ)^{-1}Z'W'WG(b))^{-1} \quad (12.37)$$

where $b \equiv b_{WTSNLLS}$. Note that if you add AR or MA terms to a weighted specification, the weighting series will be ignored.

Nonlinear Two-stage Least Squares with AR errors

While we will not go into much detail here, note that EViews can estimate non-linear TSLS models where there are autoregressive error terms. EViews does not currently estimate nonlinear models with MA errors.

To estimate your model, simply open your equation specification window, and enter your nonlinear specification, including all AR terms, and provide your instrument list. For example, you could enter the regression specification

$$cs = \exp(c(1) + \text{gdp}^{\wedge}c(2)) + [\text{ar}(1)=c(3)]$$

with the instrument list

$$c \text{ gov}$$

EViews will transform the nonlinear regression model as described in [“Estimating AR Models” on page 307](#), and then estimate nonlinear TSLS on the transformed specification using the instruments C and GOV. For nonlinear models with AR errors, EViews uses a Gauss-Newton algorithm. See [“Optimization Algorithms” on page 663](#) for further details.

Solving Estimation Problems

EViews may not be able to estimate your nonlinear equation on the first attempt. Sometimes, the nonlinear least squares procedure will stop immediately. Other times, EViews may stop estimation after several iterations without achieving convergence. EViews might even report that it cannot improve the sums-of-squares. While there are no specific rules on how to proceed if you encounter these estimation problems, there are a few general areas you might want to examine.

Starting Values

If you experience problems with the very first iteration of a nonlinear procedure, the problem is almost certainly related to starting values. See the discussion above for how to examine and change your starting values.

Model Identification

If EViews goes through a number of iterations and then reports that it encounters a “Near Singular Matrix”, you should check to make certain that your model is identified. Models are said to be non-identified if there are multiple sets of coefficients which identically yield the minimized sum-of-squares value. If this condition holds, it is impossible to choose between the coefficients on the basis of the minimum sum-of-squares criterion.

For example, the nonlinear specification,

$$y_t = \beta_1\beta_2 + \beta_2^2x_t + \epsilon_t \quad (12.38)$$

is not identified, since any coefficient pair (β_1, β_2) is indistinguishable from the pair $(-\beta_1, -\beta_2)$ in terms of the sum-of-squared residuals.

For a thorough discussion of identification of nonlinear least squares models, see Davidson and MacKinnon (1993, Sections 2.3, 5.2 and 6.3).

Convergence Criterion

EViews may report that it is unable to improve the sums-of-squares. This result may be evidence of non-identification or model misspecification. Alternatively, it may be the result of setting your convergence criterion too low, which can occur if your nonlinear specification is particularly complex.

If you wish to change the convergence criterion, enter the new value in the Options dialog. Be aware that increasing this value increases the possibility that you will stop at a local minimum, and may hide misspecification or non-identification of your model.

See [“Setting Estimation Options” on page 666](#), for further details.

Generalized Method of Moments (GMM)

The starting point of GMM estimation is a theoretical relation that the parameters should satisfy. The idea is to choose the parameter estimates so that the theoretical relation is satisfied as “closely” as possible. The theoretical relation is replaced by its sample counterpart and the estimates are chosen to minimize the weighted distance between the theoretical and actual values. GMM is a robust estimator in that, unlike maximum likelihood estimation, it does not require information of the exact distribution of the disturbances. In fact, many common estimators in econometrics can be considered as special cases of GMM.

The theoretical relation that the parameters should satisfy are usually *orthogonality conditions* between some (possibly nonlinear) function of the parameters $f(\theta)$ and a set of instrumental variables z_t :

$$E(f(\theta)'Z) = 0, \quad (12.39)$$

where θ are the parameters to be estimated. The GMM estimator selects parameter estimates so that the sample correlations between the instruments and the function f are as close to zero as possible, as defined by the criterion function:

$$J(\theta) = (m(\theta))'Am(\theta), \quad (12.40)$$

where $m(\theta) = f(\theta)'Z$ and A is a weighting matrix. Any symmetric positive definite matrix A will yield a consistent estimate of q . However, it can be shown that a necessary (but not sufficient) condition to obtain an (asymptotically) efficient estimate of q is to set A equal to the inverse of the covariance matrix of the sample moments m .

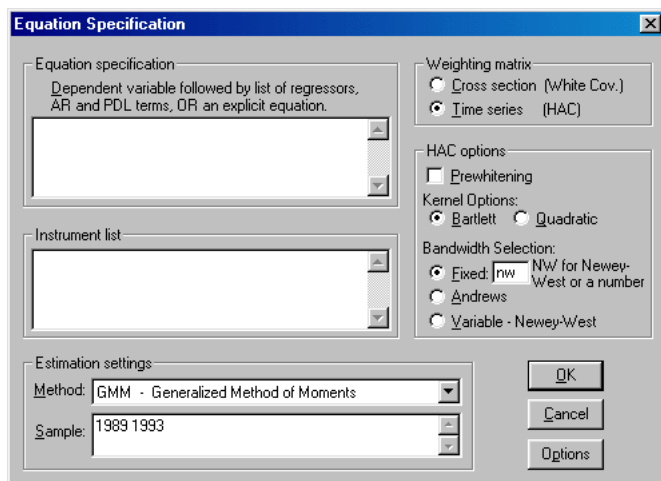
Many standard estimators, including all of the system estimators provided in EViews, can be set up as special cases of GMM. For example, the ordinary least squares estimator can be viewed as a GMM estimator, based upon the conditions that each of the right-hand variables is uncorrelated with the residual.

Estimation by GMM in EViews

To estimate an equation by GMM, either create a new equation object by selecting **Object/New Object/Equation**, or press the **Estimate** button in the toolbar of an existing equation. From the Equation Specification dialog choose **Estimation Method: GMM**. The estimation specification dialog will change as depicted below.

To obtain GMM estimates in EViews, you need to write the moment condition as an orthogonality condition between an expression including the parameters and a set of instrumental variables. There are two ways you can write the orthogonality condition: with and without a dependent variable.

If you specify the equation either by listing vari-



able names or by an expression with an equal sign, EViews will interpret the moment condition as an orthogonality condition between the instruments and the residuals defined by the equation. If you specify the equation by an expression without an equal sign, EViews will orthogonalize that expression to the set of instruments.

You must also list the names of the instruments in the **Instrument list** field box of the Equation Specification dialog box. For the GMM estimator to be identified, there must be at least as many instrumental variables as there are parameters to estimate. EViews will always include the constant in the list of instruments.

For example, if you type

Equation Specification: $y = c + x$

Instrument list: $c \ z \ w$

the orthogonality conditions given by

$$\begin{aligned}\sum (y_t - c(1) - c(2)x_t) &= 0 \\ \sum (y_t - c(1) - c(2)x_t)z_t &= 0 \\ \sum (y_t - c(1) - c(2)x_t)w_t &= 0\end{aligned}\tag{12.41}$$

If you enter an expression

Equation Specification: $c(1) * \log(y) + x^{c(2)}$

Instrument list: $c \ z \ z(-1)$

the orthogonality conditions are

$$\begin{aligned}\sum (c(1)\log y_t + x_t^{c(2)}) &= 0 \\ \sum (c(1)\log y_t + x_t^{c(2)})z_t &= 0 \\ \sum (c(1)\log y_t + x_t^{c(2)})z_{t-1} &= 0\end{aligned}\tag{12.42}$$

On the right part of the Equation Specification dialog are the options for selecting the weighting matrix A in the objective function. If you select **Weighting Matrix: Cross section (White Cov)**, the GMM estimates will be robust to heteroskedasticity of unknown form.

If you select **Weighting Matrix: Time series (HAC)**, the GMM estimates will be robust to heteroskedasticity and autocorrelation of unknown form. For the HAC option, you have to specify the kernel type and bandwidth.

- The **Kernel Options** determine the functional form of the kernel used to weight the autocovariances in computing the weighting matrix.

- The **Bandwidth Selection** option determines how the weights given by the kernel change with the lags of the autocovariances in the computation of the weighting matrix. If you select **Fixed** bandwidth, you may either enter a number for the bandwidth or type `nw` to use Newey and West’s fixed bandwidth selection criterion.
- The **Prewhitening** option runs a preliminary VAR(1) prior to estimation to “soak up” the correlation in the moment conditions.

The technical notes in [“Generalized Method of Moments \(GMM\)” on page 515](#) describe these options in more detail.

Example

Tauchen (1986) considers the problem of estimating the taste parameters β , γ from the Euler equation

$$(\beta R_{t+1} w_{t+1}^{-\gamma} - 1)' z_t = 0 \quad (12.43)$$

where we use instruments $z_t = (1, w_t, w_{t-1}, r_t, r_{t-1})'$. To estimate the parameters β , γ by GMM, fill in the Equation Specification dialog as

Equation Specification: `c(1)*r(+1)*w(+1)^(-c(2))-1`

Instrument list: `c w w(-1) r r(-1)`

The estimation result using the default HAC **Weighting Matrix** option looks as follows:

```

Dependent Variable: Implicit Equation
Method: Generalized Method of Moments
Date: 09/26/97   Time: 14:02
Sample(adjusted): 1891 1982
Included observations: 92 after adjusting endpoints
No prewhitening
Bandwidth: Fixed (3)
Kernel: Bartlett
Convergence achieved after: 7 weight matrices, 7 total coef iterations
C(1)*R(+1)*W(+1)^(-C(2))-1
Instrument list: C W W(-1) R R(-1)

```

	Coefficient	Std. Error	t-Statistic	Prob.
C(1)	0.934096	0.018751	49.81600	0.0000
C(2)	1.366396	0.741802	1.841995	0.0688
S.E. of regression	0.154084	Sum squared resid		2.136760
Durbin-Watson stat	1.903837	J-statistic		0.054523

Note that when you specify an equation without a dependent variable, EViews does not report some of the regression statistics such as the R-squared. The J -statistic reported at the bottom of the table is the minimized value of the objective function. The J -statistic can be used to carry out hypothesis tests from GMM estimation; see Newey and West (1987a). A simple application of the J -statistic is to test the validity of overidentifying restrictions when you have more instruments than parameters to estimate. In this example, we have

five instruments to estimate two parameters and so there are three overidentifying restrictions. Under the null hypothesis that the overidentifying restrictions are satisfied, the J -statistic times the number of regression observations is asymptotically χ^2 with degrees of freedom equal to the number of overidentifying restrictions. You can compute the test statistic as a named scalar in EViews using the commands

```
scalar overid=eq_gmm.@regobs*eq_gmm.@jstat
scalar overid_p=1-@cchisq(overid,3)
```

where EQ_GMM is the name of the equation containing the GMM estimates. The second command computes the p -value of the test statistic as a named scalar OVERID_P. To view the value of OVERID_P, double click on its name; the value will be displayed in the status line at the bottom of the EViews window.

Commands

To estimate an equation by weighted least squares, specify the weighting series in parentheses with the `w=` option after the `ls` command:

```
eq1.ls(w=1/pop) cs c gdp cpi
```

To estimate an equation by two-stage least squares, follow the `tsls` command with the dependent variable, the independent variables, an `@` sign, and a list of instruments:

```
equation eq2.tsls cs c gdp @ c cs(-1) gdp(-1)
```

To estimate an equation by GMM, follow the `gmm` command with the dependent variable, the independent variables, an `@` sign, and a list of instruments that define the orthogonality conditions:

```
equation eq3.gmm cs c gdp @ c cs(-1) gdp(-1)
```

You can set the starting values prior to nonlinear estimation by the command

```
param c(1) 1.5 c(2) 1
```

or

```
c(1) = 1.5
c(2) = 1
```

To declare a coefficient vector, specify the number of rows in parentheses and provide a name:

```
coef(4) beta
```

declares a four element coefficient vector named BETA filled with zeros.

See [“Equation” on page 21](#) of the *Command and Programming Reference* for a complete list of commands and options for single equation estimation in EViews.

Chapter 13. Time Series Regression

In this section we discuss single equation regression techniques that are important for the analysis of time series data: testing for serial correlation, estimation of ARMA models, using polynomial distributed lags, and testing for unit roots in potentially nonstationary time series.

The chapter focuses on the specification and estimation of time series models. A number of related topics are discussed elsewhere: standard multiple regression techniques are discussed in [Chapters 11](#) and [12](#), forecasting and inference are discussed extensively in [Chapters 14](#) and [15](#), vector autoregressions are discussed in [Chapter 20](#), and state space models and the Kalman filter are discussed in [Chapter 22](#).

Serial Correlation Theory

A common finding in time series regressions is that the residuals are correlated with their own lagged values. This serial correlation violates the standard assumption of regression theory that disturbances are not correlated with other disturbances. The primary problems associated with serial correlation are:

- OLS is no longer efficient among linear estimators. Furthermore, since prior residuals help to predict current residuals, we can take advantage of this information to form a better prediction of the dependent variable.
- Standard errors computed using the textbook OLS formula are not correct, and are generally understated.
- If there are lagged dependent variables on the right-hand side, OLS estimates are biased and inconsistent.

EViews provides tools for detecting serial correlation and estimation methods that take account of its presence.

In general, we will be concerned with specifications of the form:

$$\begin{aligned}y_t &= x_t' \beta + u_t \\u_t &= z_{t-1}' \gamma + \epsilon_t\end{aligned}\tag{13.1}$$

where x_t is a vector of explanatory variables observed at time t , z_{t-1} is a vector of variables known in the previous period, β and γ are vectors of parameters, u_t is a disturbance term, and ϵ_t is the innovation in the disturbance. The vector z_{t-1} may contain lagged values of u , lagged values of ϵ , or both.

The disturbance u_t is termed the *unconditional residual*. It is the residual based on the structural component ($x_t'\beta$) but not using the information contained in z_{t-1} . The innovation ϵ_t is also known as the *one-period ahead forecast error* or the *prediction error*. It is the difference between the actual value of the dependent variable and a forecast made on the basis of the independent variables and the past forecast errors.

The First-Order Autoregressive Model

The simplest and most widely used model of serial correlation is the first-order autoregressive, or AR(1), model. The AR(1) model is specified as

$$\begin{aligned}y_t &= x_t'\beta + u_t \\u_t &= \rho u_{t-1} + \epsilon_t\end{aligned}\tag{13.2}$$

The parameter ρ is the first-order serial correlation coefficient. In effect, the AR(1) model incorporates the residual from the past observation into the regression model for the current observation.

Higher-Order Autoregressive Models

More generally, a regression with an autoregressive process of order p , AR(p) error is given by

$$\begin{aligned}y_t &= x_t'\beta + u_t \\u_t &= \rho_1 u_{t-1} + \rho_2 u_{t-2} + \dots + \rho_p u_{t-p} + \epsilon_t\end{aligned}\tag{13.3}$$

The autocorrelations of a stationary AR(p) process gradually die out to zero, while the partial autocorrelations for lags larger than p are zero.

Testing for Serial Correlation

Before you use an estimated equation for statistical inference (*e.g.* hypothesis tests and forecasting), you should generally examine the residuals for evidence of serial correlation. EViews provides several methods of testing a specification for the presence of serial correlation.

The Durbin-Watson Statistic

EViews reports the Durbin-Watson (DW) statistic as a part of the standard regression output. The Durbin-Watson statistic is a test for first-order serial correlation. More formally, the DW statistic measures the linear association between adjacent residuals from a regression model. The Durbin-Watson is a test of the hypothesis $\rho = 0$ in the specification:

$$u_t = \rho u_{t-1} + \epsilon_t.\tag{13.4}$$

If there is no serial correlation, the DW statistic will be around 2. The DW statistic will fall below 2 if there is positive serial correlation (in the worst case, it will be near zero). If there is negative correlation, the statistic will lie somewhere between 2 and 4.

Positive serial correlation is the most commonly observed form of dependence. As a rule of thumb, with 50 or more observations and only a few independent variables, a DW statistic below about 1.5 is a strong indication of positive first order serial correlation. See Johnston and DiNardo (1997, Chapter 6.6.1) for a thorough discussion on the Durbin-Watson test and a table of the significance points of the statistic.

There are three main limitations of the DW test as a test for serial correlation. First, the distribution of the DW statistic under the null hypothesis depends on the data matrix x . The usual approach to handling this problem is to place bounds on the critical region, creating a region where the test results are inconclusive. Second, if there are lagged dependent variables on the right-hand side of the regression, the DW test is no longer valid. Lastly, you may only test the null hypothesis of no serial correlation against the alternative hypothesis of first-order serial correlation.

Two other tests of serial correlation—the Q -statistic and the Breusch-Godfrey LM test—overcome these limitations, and are preferred in most applications.

Correlograms and Q -statistics

If you select **View/Residual Tests/Correlogram- Q -statistics** on the equation toolbar, EViews will display the autocorrelation and partial autocorrelation functions of the residuals, together with the Ljung-Box Q -statistics for high-order serial correlation. If there is no serial correlation in the residuals, the autocorrelations and partial autocorrelations at all lags should be nearly zero, and all Q -statistics should be insignificant with large p -values.

Note that the p -values of the Q -statistics will be computed with the degrees of freedom adjusted for the inclusion of ARMA terms in your regression. There is evidence that some care should be taken in interpreting the results of a Ljung-Box test applied to the residuals from an ARMAX specification (see Dezhbaksh, 1990, for simulation evidence on the finite sample performance of the test in this setting).

Details on the computation of correlograms and Q -statistics are provided in greater detail in [Chapter 7, “Series”, on page 169](#).

Serial Correlation LM Test

Selecting **View/Residual Tests/Serial Correlation LM Test...** carries out the Breusch-Godfrey Lagrange multiplier test for general, high-order, ARMA errors. In the Lag Specification dialog box, you should enter the highest order of serial correlation to be tested.

The null hypothesis of the test is that there is no serial correlation in the residuals up to the specified order. EViews reports a statistic labeled “ F -statistic” and an “Obs*R-squared”

(NR^2 —the number of observations times the R-square) statistic. The NR^2 statistic has an asymptotic χ^2 distribution under the null hypothesis. The distribution of the F -statistic is not known, but is often used to conduct an informal test of the null.

See “[Serial Correlation LM Test](#)” on page 305 for further discussion of the serial correlation LM test.

Example

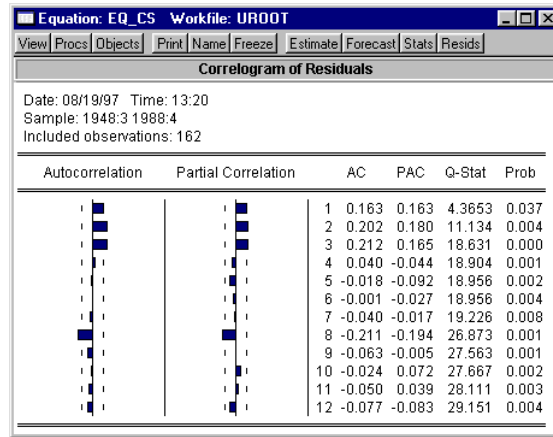
As an example of the application of these testing procedures, consider the following results from estimating a simple consumption function by ordinary least squares:

Dependent Variable: CS
 Method: Least Squares
 Date: 08/19/97 Time: 13:03
 Sample: 1948:3 1988:4
 Included observations: 162

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-9.227624	5.898177	-1.564487	0.1197
GDP	0.038732	0.017205	2.251193	0.0257
CS(-1)	0.952049	0.024484	38.88516	0.0000
R-squared	0.999625	Mean dependent var		1781.675
Adjusted R-squared	0.999621	S.D. dependent var		694.5419
S.E. of regression	13.53003	Akaike info criterion		8.066045
Sum squared resid	29106.82	Schwarz criterion		8.123223
Log likelihood	-650.3497	F-statistic		212047.1
Durbin-Watson stat	1.672255	Prob(F-statistic)		0.000000

A quick glance at the results reveals that the coefficients are statistically significant and the fit is very tight. However, if the error term is serially correlated, the estimated OLS standard errors are invalid and the estimated coefficients will be biased and inconsistent due to the presence of a lagged dependent variable on the right-hand side. The Durbin-Watson statistic is not appropriate as a test for serial correlation in this case, since there is a lagged dependent variable on the right-hand side of the equation.

Selecting **View/Residual Tests/Correlogram-Q-statistics** from this equation produces the following view:



The correlogram has spikes at lags up to three and at lag eight. The Q -statistics are significant at all lags, indicating significant serial correlation in the residuals.

Selecting **View/Residual Tests/Serial Correlation LM Test...** and entering a lag of 4 yields the following result:

Breusch-Godfrey Serial Correlation LM Test:			
F-statistic	3.654696	Probability	0.007109
Obs*R-squared	13.96215	Probability	0.007417

The test rejects the hypothesis of no serial correlation up to order four. The Q -statistic and the LM test both indicate that the residuals are serially correlated and the equation should be re-specified before using it for hypothesis tests and forecasting.

Estimating AR Models

Before you use the tools described in this section, you may first wish to examine your model for other signs of misspecification. Serial correlation in the errors may be evidence of serious problems with your specification. In particular, you should be on guard for an excessively restrictive specification that you arrived at by experimenting with ordinary least squares. Sometimes, adding improperly excluded variables to your regression will eliminate the serial correlation.

For a discussion of the efficiency gains from the serial correlation correction and some Monte-Carlo evidence, see Rao and Griliches (1969).

First-Order Serial Correlation

To estimate an AR(1) model in EViews, open an equation by selecting **Quick/Estimate Equation...** and enter your specification as usual, adding the expression “AR(1)” to the end of your list. For example, to estimate a simple consumption function with AR(1) errors,

$$\begin{aligned}CS_t &= c_1 + c_2GDP_t + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t\end{aligned}\tag{13.5}$$

you should specify your equation as

```
cs c gdp ar(1)
```

EViews automatically adjusts your sample to account for the lagged data used in estimation, estimates the model, and reports the adjusted sample along with the remainder of the estimation output.

Higher-Order Serial Correlation

Estimating higher order AR models is only slightly more complicated. To estimate an AR(k), you should enter your specification, followed by expressions for each AR term you wish to include. If you wish to estimate a model with autocorrelations from one to five:

$$\begin{aligned}CS_t &= c_1 + c_2GDP_t + u_t \\ u_t &= \rho_1u_{t-1} + \rho_2u_{t-2} + \dots + \rho_5u_{t-5} + \epsilon_t\end{aligned}\tag{13.6}$$

you should enter

```
cs c gdp ar(1) ar(2) ar(3) ar(4) ar(5)
```

By requiring that you enter all of the autocorrelations you wish to include in your model, EViews allows you great flexibility in restricting lower order correlations to be zero. For example, if you have quarterly data and want to include a single term to account for seasonal autocorrelation, you could enter

```
cs c gdp ar(4)
```

Nonlinear Models with Serial Correlation

EViews can estimate nonlinear regression models with additive AR errors. For example, suppose you wish to estimate the following nonlinear specification with an AR(2) error:

$$\begin{aligned}CS_t &= c_1 + GDP_t^{c_2} + u_t \\ u_t &= c_3u_{t-1} + c_4u_{t-2} + \epsilon_t\end{aligned}\tag{13.7}$$

Simply specify your model using EViews expressions, followed by an additive term describing the AR correction enclosed in square brackets. The AR term should contain a coefficient assignment for each AR lag, separated by commas:

$$cs = c(1) + gdp \wedge c(2) + [ar(1)=c(3), ar(2)=c(4)]$$

EViews transforms this nonlinear model by differencing, and estimates the transformed nonlinear specification using a Gauss-Newton iterative procedure (see [“How EViews Estimates AR Models” on page 310](#)).

Two-Stage Regression Models with Serial Correlation

By combining two-stage least squares or two-stage nonlinear least squares with AR terms, you can estimate models where there is correlation between regressors and the innovations as well as serial correlation in the residuals.

If the original regression model is linear, EViews uses the Marquardt algorithm to estimate the parameters of the transformed specification. If the original model is nonlinear, EViews uses Gauss-Newton to estimate the AR corrected specification.

For further details on the algorithms and related issues associated with the choice of instruments, see the discussion in [“TSLS with AR errors” beginning on page 286](#).

Output from AR Estimation

When estimating an AR model, some care must be taken in interpreting your results. While the estimated coefficients, coefficient standard errors, and t -statistics may be interpreted in the usual manner, results involving residuals differ from those computed in OLS settings.

To understand these differences, keep in mind that there are two different residuals associated with an AR model. The first are the estimated *unconditional residuals*,

$$\hat{u}_t = y_t - x_t' b, \quad (13.8)$$

which are computed using the original variables, and the estimated coefficients, b . These residuals are the errors that you would observe if you made a prediction of the value of y_t using contemporaneous information, but ignoring the information contained in the lagged residual.

Normally, there is no strong reason to examine these residuals, and EViews does not automatically compute them following estimation.

The second set of residuals are the estimated *one-period ahead forecast errors*, $\hat{\epsilon}$. As the name suggests, these residuals represent the forecast errors you would make if you computed forecasts using a prediction of the residuals based upon past values of your data, in addition to the contemporaneous information. In essence, you improve upon the uncondi-

tional forecasts and residuals by taking advantage of the predictive power of the lagged residuals.

For AR models, the residual-based regression statistics—such as the R^2 , the standard error of regression, and the Durbin-Watson statistic— reported by EViews are based on the one-period ahead forecast errors, $\hat{\epsilon}$.

A set of statistics that is unique to AR models is the estimated AR parameters, $\hat{\rho}_i$. For the simple AR(1) model, the estimated parameter $\hat{\rho}$ is the serial correlation coefficient of the unconditional residuals. For a stationary AR(1) model, the true ρ lies between -1 (extreme negative serial correlation) and $+1$ (extreme positive serial correlation). The stationarity condition for general AR(p) processes is that the inverted roots of the lag polynomial lie inside the unit circle. EViews reports these roots **as Inverted AR Roots** at the bottom of the regression output. There is no particular problem if the roots are imaginary, but a stationary AR model should have all roots with modulus less than one.

How EViews Estimates AR Models

Textbooks often describe techniques for estimating AR models. The most widely discussed approaches, the Cochrane-Orcutt, Prais-Winsten, Hatanaka, and Hildreth-Lu procedures, are multi-step approaches designed so that estimation can be performed using standard *linear* regression. All of these approaches suffer from important drawbacks which occur when working with models containing lagged dependent variables as regressors, or models using higher-order AR specifications; see Davidson and MacKinnon (1994, pp. 329–341), Greene (1997, p. 600–607).

EViews estimates AR models using nonlinear regression techniques. This approach has the advantage of being easy to understand, generally applicable, and easily extended to nonlinear specifications and models that contain endogenous right-hand side variables. Note that the nonlinear least squares estimates are asymptotically equivalent to maximum likelihood estimates and are asymptotically efficient.

To estimate an AR(1) model, EViews transforms the linear model

$$\begin{aligned} y_t &= x_t' \beta + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t \end{aligned} \tag{13.9}$$

into the nonlinear model,

$$y_t = \rho y_{t-1} + (x_t - \rho x_{t-1})' \beta + \epsilon_t, \tag{13.10}$$

by substituting the second equation into the first, and rearranging terms. The coefficients ρ and β are estimated simultaneously by applying a Marquardt nonlinear least squares algorithm to the transformed equation. See [Appendix D, “Estimation Algorithms and Options”](#), on page 663 for details on nonlinear estimation.

For a nonlinear AR(1) specification, EViews transforms the nonlinear model

$$\begin{aligned}y_t &= f(x_t, \beta) + u_t \\u_t &= \rho u_{t-1} + \epsilon_t\end{aligned}\tag{13.11}$$

into the alternative nonlinear specification

$$y_t = \rho y_{t-1} + f(x_t, \beta) - \rho f(x_{t-1}, \beta) + \epsilon_t\tag{13.12}$$

and estimates the coefficients using a Marquardt nonlinear least squares algorithm.

Higher order AR specifications are handled analogously. For example, a nonlinear AR(3) is estimated using nonlinear least squares on the equation

$$\begin{aligned}y_t &= (\rho_1 y_{t-1} + \rho_2 y_{t-2} + \rho_3 y_{t-3}) + f(x_t, \beta) - \rho_1 f(x_{t-1}, \beta) \\&\quad - \rho_2 f(x_{t-2}, \beta) - \rho_3 f(x_{t-3}, \beta) + \epsilon_t\end{aligned}\tag{13.13}$$

For details, see Fair (1984, pp. 210–214), and Davidson and MacKinnon (1996, pp. 331–341).

ARIMA Theory

ARIMA (autoregressive integrated moving average) models are generalizations of the simple AR model that use three tools for modeling the serial correlation in the disturbance:

- The first tool is the autoregressive, or AR, term. The AR(1) model introduced above uses only the first-order term but, in general, you may use additional, higher-order AR terms. Each AR term corresponds to the use of a lagged value of the residual in the forecasting equation for the unconditional residual. An autoregressive model of order p , AR(p) has the form

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \dots + \rho_p u_{t-p} + \epsilon_t.\tag{13.14}$$

- The second tool is the integration order term. Each integration order corresponds to differencing the series being forecast. A first-order integrated component means that the forecasting model is designed for the first difference of the original series. A second-order component corresponds to using second differences, and so on.
- The third tool is the MA, or moving average term. A moving average forecasting model uses lagged values of the forecast error to improve the current forecast. A first-order moving average term uses the most recent forecast error, a second-order term uses the forecast error from the two most recent periods, and so on. An MA(q) has the form:

$$u_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \dots + \theta_q \epsilon_{t-q}.\tag{13.15}$$

Please be aware that some authors and software packages use the opposite sign convention for the θ coefficients so that the signs of the MA coefficients may be reversed.

The autoregressive and moving average specifications can be combined to form an ARMA(p, q) specification

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \dots + \rho_p u_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \dots + \theta_q \epsilon_{t-q} \quad (13.16)$$

Although econometricians typically use ARIMA models applied to the residuals from a regression model, the specification can also be applied directly to a series. This latter approach provides a univariate model, specifying the conditional mean of the series as a constant, and measuring the residuals as differences of the series from its mean.

Principles of ARIMA Modeling (Box-Jenkins 1976)

In ARIMA forecasting, you assemble a complete forecasting model by using combinations of the three building blocks described above. The first step in forming an ARIMA model for a series of residuals is to look at its autocorrelation properties. You can use the correlogram view of a series for this purpose, as outlined in [“Correlogram” on page 167](#).

This phase of the ARIMA modeling procedure is called *identification* (not to be confused with the same term used in the simultaneous equations literature). The nature of the correlation between current values of residuals and their past values provides guidance in selecting an ARIMA specification.

The autocorrelations are easy to interpret—each one is the correlation coefficient of the current value of the series with the series lagged a certain number of periods. The partial autocorrelations are a bit more complicated; they measure the correlation of the current and lagged series after taking into account the predictive power of all the values of the series with smaller lags. The partial autocorrelation for lag 6, for example, measures the added predictive power of u_{t-6} when u_1, \dots, u_{t-5} are already in the prediction model. In fact, the partial autocorrelation is precisely the regression coefficient of u_{t-6} in a regression where the earlier lags are also used as predictors of u_t .

If you suspect that there is a distributed lag relationship between your dependent (left-hand) variable and some other predictor, you may want to look at their cross correlations before carrying out estimation.

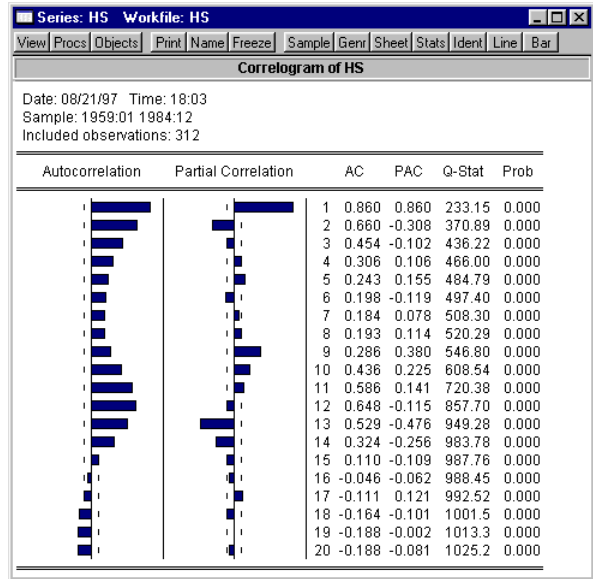
The next step is to decide what kind of ARIMA model to use. If the autocorrelation function dies off smoothly at a geometric rate, and the partial autocorrelations were zero after one lag, then a first-order autoregressive model is appropriate. Alternatively, if the autocorrelations were zero after one lag and the partial autocorrelations declined geometrically, a first-order moving average process would seem appropriate. If the autocorrelations appear

to have a seasonal pattern, this would suggest the presence of a seasonal ARMA structure (see “Seasonal ARMA Terms” on page 316).

For example, we can examine the correlogram of the DRI Basics housing series in the HS.WF1 workfile by selecting **View/Correlogram...** from the HS series toolbar:

The “wavy” cyclical correlogram with a seasonal frequency suggests fitting a seasonal ARMA model to HS.

The goal of ARIMA analysis is a parsimonious representation of the process governing the residual. You should use only enough AR and MA terms to fit the properties of the residuals. The Akaike information criterion and Schwarz criterion provided with each set of estimates may also be used as a guide for the appropriate lag order selection.



After fitting a candidate ARIMA specification, you should verify that there are no remaining autocorrelations that your model has not accounted for. Examine the autocorrelations and the partial autocorrelations of the innovations (the residuals from the ARIMA model) to see if any important forecasting power has been overlooked. EViews provides views for diagnostic checks after estimation.

Estimating ARIMA Models

EViews estimates general ARIMA specifications that allow for right-hand side explanatory variables. Despite the fact that these models are sometimes termed ARIMAX specifications, we will refer to this general class of models as ARIMA.

To specify your ARIMA model, you will:

- Difference your dependent variable, if necessary, to account for the order of integration.
- Describe your structural regression model (dependent variables and regressors) and add any AR or MA terms, as described above.

Differencing

The `d` operator can be used to specify differences of series. To specify first differencing, simply include the series name in parentheses after `d`. For example, `d(gdp)` specifies the first difference of GDP, or $GDP - GDP(-1)$.

More complicated forms of differencing may be specified with two optional parameters, n and s . `d(x, n)` specifies the n -th order difference of the series X :

$$d(x, n) = (1 - L)^n x, \quad (13.17)$$

where L is the lag operator. For example, `d(gdp, 2)` specifies the second order difference of GDP:

$$d(gdp, 2) = gdp - 2 * gdp(-1) + gdp(-2)$$

`d(x, n, s)` specifies n -th order ordinary differencing of X with a seasonal difference at lag s :

$$d(x, n, s) = (1 - L)^n (1 - L^s) x. \quad (13.18)$$

For example, `d(gdp, 0, 4)` specifies zero ordinary differencing with a seasonal difference at lag 4, or $GDP - GDP(-4)$.

If you need to work in logs, you can also use the `dlog` operator, which returns differences in the log values. For example, `dlog(gdp)` specifies the first difference of $\log(GDP)$ or $\log(GDP) - \log(GDP(-1))$. You may also specify the n and s options as described for the simple `d` operator, `dlog(x, n, s)`.

There are two ways to estimate integrated models in EViews. First, you may generate a new series containing the differenced data, and then estimate an ARMA model using the new data. For example, to estimate a Box-Jenkins ARIMA(1, 1, 1) model for M1, you can enter:

```
series dm1 = d(m1)
ls dm1 c ar(1) ma(1)
```

Alternatively, you may include the difference operator `d` directly in the estimation specification. For example, the same ARIMA(1,1,1) model can be estimated by the one-line command

```
ls d(m1) c ar(1) ma(1)
```

The latter method should generally be preferred for an important reason. If you define a new variable, such as DM1 above, and use it in your estimation procedure, then when you forecast from the estimated model, EViews will make forecasts of the dependent variable DM1. That is, you will get a forecast of the differenced series. If you are really interested in forecasts of the level variable, in this case M1, you will have to manually transform the

forecasted value and adjust the computed standard errors accordingly. Moreover, if any other transformation or lags of M1 are included as regressors, EViews will not know that they are related to DM1. If, however, you specify the model using the difference operator expression for the dependent variable, $d(m1)$, the forecasting procedure will provide you with the option of forecasting the level variable, in this case M1.

The difference operator may also be used in specifying exogenous variables and can be used in equations without ARMA terms. Simply include them in the list of regressors in addition to the endogenous variables. For example,

```
d(cs,2) c d(gdp,2) d(gdp(-1),2) d(gdp(-2),2) time
```

is a valid specification that employs the difference operator on both the left-hand and right-hand sides of the equation.

ARMA Terms

The AR and MA parts of your model will be specified using the keywords `ar` and `ma` as part of the equation. We have already seen examples of this approach in our specification of the AR terms above, and the concepts carry over directly to MA terms.

For example, to estimate a second-order autoregressive and first-order moving average error process ARMA(2,1), you would include expressions for the AR(1), AR(2), and MA(1) terms along with your other regressors:

```
c gov ar(1) ar(2) ma(1)
```

Once again, you need not use the AR and MA terms consecutively. For example, if you want to fit a fourth-order autoregressive model to take account of seasonal movements, you could use AR(4) by itself:

```
c gov ar(4)
```

You may also specify a pure moving average model by using only MA terms. Thus,

```
c gov ma(1) ma(2)
```

indicates an MA(2) model for the residuals.

The traditional Box-Jenkins or ARMA models do not have any right-hand side variables except for the constant. In this case, your list of regressors would just contain a C in addition to the AR and MA terms. For example,

```
c ar(1) ar(2) ma(1) ma(2)
```

is a standard Box-Jenkins ARMA (2,2).

Seasonal ARMA Terms

Box and Jenkins (1976) recommend the use of seasonal autoregressive (SAR) and seasonal moving average (SMA) terms for monthly or quarterly data with systematic seasonal movements. A SAR(p) term can be included in your equation specification for a seasonal autoregressive term with lag p . The lag polynomial used in estimation is the product of the one specified by the AR terms and the one specified by the SAR terms. The purpose of the SAR is to allow you to form the product of lag polynomials.

Similarly, SMA(q) can be included in your specification to specify a seasonal moving average term with lag q . The lag polynomial used in estimation is the product of the one defined by the MA terms and the one specified by the SMA terms. As with the SAR, the SMA term allows you to build up a polynomial that is the product of underlying lag polynomials.

For example, a second-order AR process without seasonality is given by

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \epsilon_t, \quad (13.19)$$

which can be represented using the lag operator L , $L^n x_t = x_{t-n}$ as

$$(1 - \rho_1 L - \rho_2 L^2)u_t = \epsilon_t. \quad (13.20)$$

You can estimate this process by including `ar(1)` and `ar(2)` terms in the list of regressors. With quarterly data, you might want to add a `sar(4)` expression to take account of seasonality. If you specify the equation as

```
sales c inc ar(1) ar(2) sar(4)
```

then the estimated error structure would be:

$$(1 - \rho_1 L - \rho_2 L^2)(1 - \phi L^4)u_t = \epsilon_t. \quad (13.21)$$

The error process is equivalent to:

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \phi u_{t-4} - \phi \rho_1 u_{t-5} - \phi \rho_2 u_{t-6} + \epsilon_t. \quad (13.22)$$

The parameter ϕ is associated with the seasonal part of the process. Note that this is an AR(6) process with nonlinear restrictions on the coefficients.

As another example, a second-order MA process without seasonality may be written

$$u_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2}, \quad (13.23)$$

or using lag operators,

$$u_t = (1 + \theta_1 L + \theta_2 L^2)\epsilon_t. \quad (13.24)$$

You can estimate this second-order process by including both the MA(1) and MA(2) terms in your equation specification.

With quarterly data, you might want to add `sma(4)` to take account of seasonality. If you specify the equation as

```
cs c ad ma(1) ma(2) sma(4)
```

then the estimated model is:

$$\begin{aligned} CS_t &= \beta_1 + \beta_2 AD_t + u_t \\ u_t &= (1 + \theta_1 L + \theta_2 L^2)(1 + \omega L^4)\epsilon_t \end{aligned} \quad (13.25)$$

The error process is equivalent to

$$u_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \omega \epsilon_{t-4} + \omega \theta_1 \epsilon_{t-5} + \omega \theta_2 \epsilon_{t-6}. \quad (13.26)$$

The parameter w is associated with the seasonal part of the process. This is just an MA(6) process with nonlinear restrictions on the coefficients. You can also include both SAR and SMA terms.

Output from ARIMA Estimation

The output from estimation with AR or MA specifications is the same as for ordinary least squares, with the addition of a lower block that shows the reciprocal roots of the AR and MA polynomials. If we write the general ARMA model using the lag polynomial $\rho(L)$ and $\theta(L)$ as

$$\rho(L)u_t = \theta(L)\epsilon_t, \quad (13.27)$$

then the reported roots are the roots of the polynomials

$$\rho(x^{-1}) = 0 \quad \text{and} \quad \theta(x^{-1}) = 0. \quad (13.28)$$

The roots, which may be imaginary, should have modulus no greater than one. The output will display a warning message if any of the roots violate this condition.

If ρ has a real root whose absolute value exceeds one or a pair of complex reciprocal roots outside the unit circle (that is, with modulus greater than one), it means that the autoregressive process is explosive.

If θ has reciprocal roots outside the unit circle, we say that the MA process is *noninvertible*, which makes interpreting and using the MA results difficult. However, noninvertibility poses no substantive problem, since as Hamilton (1994a, p. 65) notes, there is always an equivalent representation for the MA model where the reciprocal roots lie inside the unit circle. Accordingly, you should re-estimate your model with different starting values until you get a moving average process that satisfies invertibility. Alternatively, you may wish to turn off MA backcasting (see “[Backcasting MA terms](#)” on page 320).

If the estimated MA process has roots with modulus close to one, it is a sign that you may have over-differenced the data. The process will be difficult to estimate and even more difficult to forecast. If possible, you should re-estimate with one less round of differencing.

Consider the following example output from ARMA estimation:

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	8.614804	0.961559	8.959208	0.0000
AR(1)	0.983011	0.009127	107.7077	0.0000
SAR(4)	0.941898	0.018788	50.13275	0.0000
MA(1)	0.513572	0.040236	12.76402	0.0000
SMA(4)	-0.960399	0.000601	-1598.423	0.0000
R-squared	0.991557	Mean dependent var		6.978830
Adjusted R-squared	0.991484	S.D. dependent var		2.919607
S.E. of regression	0.269420	Akaike info criterion		0.225489
Sum squared resid	33.75296	Schwarz criterion		0.269667
Log likelihood	-47.98992	F-statistic		13652.76
Durbin-Watson stat	2.099958	Prob(F-statistic)		0.000000
Inverted AR Roots	.99	.98		
Inverted MA Roots	.99	-.00+.99i	-.00 -.99i	-.51
	-.99			

This estimation result corresponds to the following specification:

$$y_t = 8.61 + u_t \quad (13.29)$$

$$(1 - 0.98L)(1 - 0.94L^4)u_t = (1 + 0.51L)(1 - 0.96L^4)\epsilon_t$$

or equivalently, to

$$y_t = 0.0088 + 0.98y_{t-1} + 0.94y_{t-4} - 0.92y_{t-5} + \epsilon_t \quad (13.30)$$

$$+ 0.51\epsilon_{t-1} - 0.96\epsilon_{t-4} - 0.49\epsilon_{t-5}$$

Note that the signs of the MA terms may be reversed from those in textbooks. Note also that the inverted roots have moduli very close to one, which is typical for many macro time series models.

Estimation Options

ARMA estimation employs the same nonlinear estimation techniques described earlier for AR estimation. These nonlinear estimation techniques are discussed further in [Chapter 12, “Additional Regression Methods”](#), on page 290.

You may need to use the **Estimation Options** dialog box to control the iterative process. EViews provides a number of options that allow you to control the iterative procedure of the estimation algorithm. In general, you can rely on the EViews choices but on occasion you may wish to override the default settings.

Iteration Limits and Convergence Criterion

Controlling the maximum number of iterations and convergence criterion are described in detail in [“Iteration and Convergence Options” on page 669](#).

Derivative Methods

EViews always computes the derivatives of AR coefficients analytically and the derivatives of the MA coefficients using finite difference numeric derivative methods. For other coefficients in the model, EViews provides you with the option of computing analytic expressions for derivatives of the regression equation (if possible) or computing finite difference numeric derivatives in cases where the derivative is not constant. Furthermore, you can choose whether to favor speed of computation (fewer function evaluations) or whether to favor accuracy (more function evaluations) in the numeric derivative computation.

Starting Values for ARMA Estimation

As discussed above, models with AR or MA terms are estimated by nonlinear least squares. Nonlinear estimation techniques require starting values for all coefficient estimates. Normally, EViews determines its own starting values and for the most part this is an issue that you need not be concerned about. However, there are a few times when you may want to override the default starting values.

First, estimation will sometimes halt when the maximum number of iterations is reached, despite the fact that convergence is not achieved. Resuming the estimation with starting values from the previous step causes estimation to pick up where it left off instead of starting over. You may also want to try different starting values to ensure that the estimates are a global rather than a local minimum of the squared errors. You might also want to supply starting values if you have a good idea of what the answers should be, and want to speed up the estimation process.

To control the starting values for ARMA estimation, click on the **Options** button in the Equation Specification dialog. Among the options which EViews provides are several alternatives for setting starting values that you can see by accessing the drop-down menu labeled **Starting Coefficient Values for ARMA**.

EViews’ default approach is **OLS/TSLS**, which runs a preliminary estimation without the ARMA terms and then starts nonlinear estimation from those values. An alternative is to use fractions of the OLS or TSLS coefficients as starting values. You can choose **.8**, **.5**, **.3**, or you can start with all coefficient values set equal to zero.

The final starting value option is **User Supplied**. Under this option, EViews uses the coefficient values that are in the coefficient vector. To set the starting values, open a window for the coefficient vector C by double clicking on the icon, and editing the values.

To properly set starting values, you will need a little more information about how EViews assigns coefficients for the ARMA terms. As with other estimation methods, when you specify your equation as a list of variables, EViews uses the built-in C coefficient vector. It assigns coefficient numbers to the variables in the following order:

- First are the coefficients of the variables, in order of entry.
- Next come the AR terms in the order you typed them.
- The SAR, MA, and SMA coefficients follow, in that order.

Thus the following two specifications will have their coefficients in the same order:

```
y c x ma(2) ma(1) sma(4) ar(1)
y sma(4) c ar(1) ma(2) x ma(1)
```

You may also assign values in the C vector using the `param` command:

```
param c(1) 50 c(2) .8 c(3) .2 c(4) .6 c(5) .1 c(6) .5
```

The starting values will be 50 for the constant, 0.8 for X, 0.2 for AR(1), 0.6 for MA(2), 0.1 for MA(1) and 0.5 for SMA(4). Following estimation, you can always see the assignment of coefficients by looking at the **Representations** view of your equation.

You can also fill the C vector from any estimated equation (without typing the numbers) by choosing **Procs/Update Coefs from Equation** in the equation toolbar.

Backcasting MA terms

By default, EViews backcasts MA terms (Box and Jenkins, 1976). Consider an MA(q) model of the form

$$\begin{aligned} y_t &= X_t' \beta + u_t \\ u_t &= \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \dots + \theta_q \epsilon_{t-q} \end{aligned} \quad (13.31)$$

Given initial values, $\hat{\beta}$ and $\hat{\phi}$, EViews first computes the unconditional residuals \hat{u}_t for $t = 1, 2, \dots, T$, and uses the backward recursion:

$$\tilde{\epsilon}_t = \hat{u}_t - \hat{\phi}_1 \tilde{\epsilon}_{t+1} - \dots - \hat{\phi}_q \tilde{\epsilon}_{t+q} \quad (13.32)$$

to compute backcast values of ϵ to $\epsilon_{-(q-1)}$. To start this recursion, the q values for the innovations *beyond* the estimation sample are set to zero:

$$\tilde{\epsilon}_{T+1} = \tilde{\epsilon}_{T+2} = \dots = \tilde{\epsilon}_{T+q} = 0. \quad (13.33)$$

Next, a forward recursion is used to estimate the values of the innovations

$$\hat{\epsilon}_t = \hat{u}_t - \hat{\phi}_1 \hat{\epsilon}_{t-1} - \dots - \hat{\phi}_q \hat{\epsilon}_{t-q}, \quad (13.34)$$

using the backcasted values of the innovations (to initialize the recursion) and the actual residuals. If your model also includes AR terms, EViews will ρ -difference the \hat{u}_t to eliminate the serial correlation prior to performing the backcast.

Lastly, the sum of squared residuals (SSR) is formed as a function of the β and ϕ , using the fitted values of the lagged innovations:

$$\text{ssr}(\beta, \phi) = \sum_{t=p+1}^T (y_t - X_t' \beta - \phi_1 \hat{\epsilon}_{t-1} - \dots - \phi_q \hat{\epsilon}_{t-q})^2. \quad (13.35)$$

This expression is minimized with respect to β and ϕ .

The backcast step, forward recursion, and minimization procedures, are repeated until the estimates of β and ϕ converge.

If backcasting is turned off, the values of the pre-sample ϵ are set to zero:

$$\epsilon_{-(q-1)} = \dots = \hat{\epsilon}_0 = 0, \quad (13.36)$$

and forward recursion is used to solve for the remaining values of the innovations.

Dealing with Estimation Problems

Since EViews uses nonlinear least squares algorithms to estimate ARMA models, all of the discussion in [Chapter 12, “Solving Estimation Problems” on page 296](#), is applicable, especially the advice to try alternative starting values.

There are a few other issues to consider that are specific to estimation of ARMA models.

First, MA models are notoriously difficult to estimate. In particular, you should avoid high order MA terms unless absolutely required for your model as they are likely to cause estimation difficulties. For example, a single large spike at lag 57 in the correlogram does not necessarily require you to include an MA(57) term in your model unless you know there is something special happening every 57 periods. It is more likely that the spike in the correlogram is simply the product of one or more outliers in the series. By including many MA terms in your model, you lose degrees of freedom, and may sacrifice stability and reliability of your estimates.

If the underlying roots of the MA process have modulus close to one, you may encounter estimation difficulties, with EViews reporting that it cannot improve the sum-of-squares or that it failed to converge in the maximum number of iterations. This behavior may be a sign that you have over-differenced the data. You should check the correlogram of the series to determine whether you can re-estimate with one less round of differencing.

Lastly, if you continue to have problems, you may wish to turn off MA backcasting.

TOLS with ARIMA errors

Two-stage least squares or instrumental variable estimation with ARIMA poses no particular difficulties.

For a detailed discussion of how to estimate TOLS specifications with ARMA errors, see [“Two-stage Least Squares” on page 283](#).

Nonlinear Models with ARMA errors

EViews will estimate nonlinear ordinary and two-stage least squares models with autoregressive error terms. For details, see the extended discussion in [“Nonlinear Least Squares” beginning on page 289](#).

EViews does not currently estimate nonlinear models with MA errors. You can, however, use the state space object to specify and estimate these models (see [“ARMAX\(2, 3\) with a Random Coefficient” on page 586](#)).

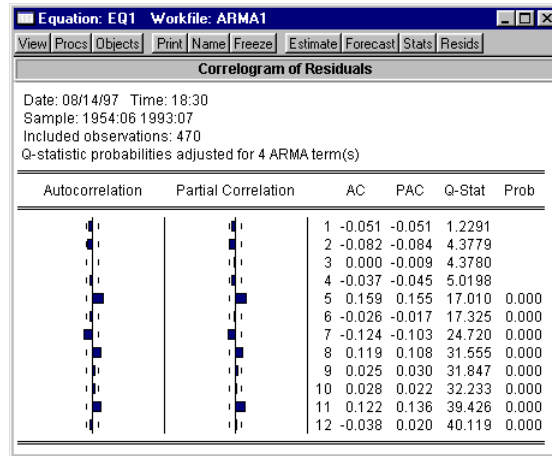
Weighted Models with ARMA errors

EViews does not have procedures to automatically estimate weighted models with ARMA error terms—if you add AR terms to a weighted model, the weighting series will be ignored. You can, of course, always construct the weighted series and then perform estimation using the weighted data and ARMA terms.

Diagnostic Evaluation

If your ARMA model is correctly specified, the residuals from the model should be nearly white noise. This means that there should be no serial correlation left in the residuals. The Durbin-Watson statistic reported in the regression output is a test for AR(1) in the absence of lagged dependent variables on the right-hand side. As discussed above, more general tests for serial correlation in the residuals can be carried out with **View/Residual Tests/Correlogram-Q-statistic** and **View/Residual Tests/Serial Correlation LM Test...**

For the example seasonal ARMA model, the residual correlogram looks as follows:



The correlogram has a significant spike at lag 5 and all subsequent Q -statistics are highly significant. This result clearly indicates the need for respecification of the model.

Polynomial Distributed Lags (PDLs)

A distributed lag is a relation of the type

$$y_t = w_t \delta + \beta_0 x_t + \beta_1 x_{t-1} + \dots + \beta_k x_{t-k} + \epsilon_t \quad (13.37)$$

The coefficients β describe the lag in the effect of x on y . In many cases, the coefficients can be estimated directly using this specification. In other cases, the high collinearity of current and lagged values of x will defeat direct estimation.

You can reduce the number of parameters to be estimated by using polynomial distributed lags (PDLs) to impose a smoothness condition on the lag coefficients. Smoothness is expressed as requiring that the coefficients lie on a polynomial of relatively low degree. A polynomial distributed lag model with order p restricts the β coefficients to lie on a p -th order polynomial of the form

$$\beta_j = \gamma_1 + \gamma_2(j - \bar{c}) + \gamma_3(j - \bar{c})^2 + \dots + \gamma_{p+1}(j - \bar{c})^p \quad (13.38)$$

for $j = 1, 2, \dots, k$, where \bar{c} is a pre-specified constant given by

$$\bar{c} = \begin{cases} (k)/2 & \text{if } p \text{ is even} \\ (k-1)/2 & \text{if } p \text{ is odd} \end{cases} \quad (13.39)$$

The PDL is sometimes referred to as an Almon lag. The constant \bar{c} is included only to avoid numerical problems that can arise from collinearity and does not affect the estimates of β .

This specification allows you to estimate a model with k lags of x using only p parameters (if you choose $p > k$, EViews will return a “Near Singular Matrix” error).

If you specify a PDL, EViews substitutes Equation (13.38) into Equation (13.37) yielding

$$y_t = \alpha + \gamma_1 z_1 + \gamma_2 z_2 + \dots + \gamma_{p+1} z_{p+1} + \epsilon_t \quad (13.40)$$

where

$$\begin{aligned} z_1 &= x_t + x_{t-1} + \dots + x_{t-k} \\ z_2 &= -\bar{c}x_t + (1 - \bar{c})x_{t-1} + \dots + (k - \bar{c})x_{t-k} \\ &\dots \\ z_{p+1} &= (-\bar{c})^p x_t + (1 - \bar{c})^p x_{t-1} + \dots + (k - \bar{c})^p x_{t-k} \end{aligned} \quad (13.41)$$

Once we estimate γ from Equation (13.40), we can recover the parameters of interest β , and their standard errors using the relationship described in Equation (13.38). This procedure is straightforward since β is a linear transformation of γ .

The specification of a polynomial distributed lag has three elements: the length of the lag k , the degree of the polynomial (the highest power in the polynomial) p , and the constraints that you want to apply. A near end constraint restricts the one-period lead effect of x on y to be zero:

$$\beta_{-1} = \gamma_1 + \gamma_2(-1 - \bar{c}) + \dots + \gamma_{p+1}(-1 - \bar{c})^p = 0. \quad (13.42)$$

A far end constraint restricts the effect of x on y to die off beyond the number of specified lags:

$$\beta_{k+1} = \gamma_1 + \gamma_2(k + 1 - \bar{c}) + \dots + \gamma_{p+1}(k + 1 - \bar{c})^p = 0. \quad (13.43)$$

If you restrict either the near or far end of the lag, the number of γ parameters estimated is reduced by one to account for the restriction; if you restrict both the near and far end of the lag, the number of γ parameters is reduced by two.

By default, EViews does not impose constraints.

How to Estimate Models Containing PDLs

You specify a polynomial distributed lag by the `pd1` term, with the following information in parentheses, each separated by a comma in this order:

- The name of the series.
- The lag length (the number of lagged values of the series to be included).
- The degree of the polynomial.
- A numerical code to constrain the lag polynomial (optional):

1	constrain the near end of the lag to zero.
2	constrain the far end.
3	constrain both ends.

You may omit the constraint code if you do not want to constrain the lag polynomial. Any number of `pdl` terms may be included in an equation. Each one tells EViews to fit distributed lag coefficients to the series and to constrain the coefficients to lie on a polynomial.

For example,

```
ls sales c pdl (orders,8,3)
```

fits SALES to a constant, and a distributed lag of current and eight lags of ORDERS, where the lag coefficients of ORDERS lie on a third degree polynomial with no endpoint constraints. Similarly,

```
ls div c pdl (rev,12,4,2)
```

fits DIV to a distributed lag of current and 12 lags of REV, where the coefficients of REV lie on a 4th degree polynomial with a constraint at the far end.

The `pdl` specification may also be used in two-stage least squares. If the series in the `pdl` is exogenous, you should include the PDL of the series in the instruments as well. For this purpose, you may specify `pdl (*)` as an instrument; all `pdl` variables will be used as instruments. For example, if you specify the TSLS equation as

```
sales c inc pdl (orders(-1),12,4)
```

with instruments

```
fed fed(-1) pdl (*)
```

the distributed lag of ORDERS will be used as instruments together with FED and FED(-1).

Polynomial distributed lags cannot be used in nonlinear specifications.

Example

The distributed lag model of industrial production (IP) on money (M1) yields the following results:

Dependent Variable: IP
 Method: Least Squares
 Date: 08/15/97 Time: 17:09
 Sample(adjusted): 1960:01 1989:12
 Included observations: 360 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	40.67568	0.823866	49.37171	0.0000
M1	0.129699	0.214574	0.604449	0.5459
M1(-1)	-0.045962	0.376907	-0.121944	0.9030
M1(-2)	0.033183	0.397099	0.083563	0.9335
M1(-3)	0.010621	0.405861	0.026169	0.9791
M1(-4)	0.031425	0.418805	0.075035	0.9402
M1(-5)	-0.048847	0.431728	-0.113143	0.9100
M1(-6)	0.053880	0.440753	0.122245	0.9028
M1(-7)	-0.015240	0.436123	-0.034944	0.9721
M1(-8)	-0.024902	0.423546	-0.058795	0.9531
M1(-9)	-0.028048	0.413540	-0.067825	0.9460
M1(-10)	0.030806	0.407523	0.075593	0.9398
M1(-11)	0.018509	0.389133	0.047564	0.9621
M1(-12)	-0.057373	0.228826	-0.250728	0.8022
R-squared	0.852398	Mean dependent var		71.72679
Adjusted R-squared	0.846852	S.D. dependent var		19.53063
S.E. of regression	7.643137	Akaike info criterion		6.943606
Sum squared resid	20212.47	Schwarz criterion		7.094732
Log likelihood	-1235.849	F-statistic		153.7030
Durbin-Watson stat	0.008255	Prob(F-statistic)		0.000000

Taken individually, none of the coefficients on lagged M1 are statistically different from zero. Yet the regression as a whole has a reasonable R^2 with a very significant F -statistic (though with a very low Durbin-Watson statistic). This is a typical symptom of high collinearity among the regressors and suggests fitting a polynomial distributed lag model.

To estimate a fifth-degree polynomial distributed lag model with no constraints, enter the commands:

```
smp1 59.1 89.12
ls ip c pdl(m1,12,5)
```

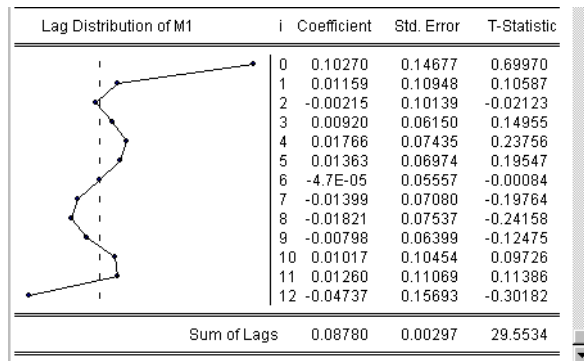
The following result is reported at the top of the equation window:

Dependent Variable: IP
 Method: Least Squares
 Date: 08/15/97 Time: 17:53
 Sample(adjusted): 1960:01 1989:12
 Included observations: 360 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	40.67311	0.815195	49.89374	0.0000
PDL01	-4.66E-05	0.055566	-0.000839	0.9993
PDL02	-0.015625	0.062884	-0.248479	0.8039
PDL03	-0.000160	0.013909	-0.011485	0.9908
PDL04	0.001862	0.007700	0.241788	0.8091
PDL05	2.58E-05	0.000408	0.063211	0.9496
PDL06	-4.93E-05	0.000180	-0.273611	0.7845
R-squared	0.852371	Mean dependent var	71.72679	
Adjusted R-squared	0.849862	S.D. dependent var	19.53063	
S.E. of regression	7.567664	Akaike info criterion	6.904899	
Sum squared resid	20216.15	Schwarz criterion	6.980462	
Log likelihood	-1235.882	F-statistic	339.6882	
Durbin-Watson stat	0.008026	Prob(F-statistic)	0.000000	

This portion of the view reports the estimated coefficients γ of the polynomial in Equation (13.38) on page 323. The terms PDL01, PDL02, PDL03, ..., correspond to z_1, z_2, \dots in Equation (13.40).

The implied coefficients of interest β_j in equation (1) are reported at the bottom of the table, together with a plot of the estimated polynomial:



The Sum of Lags reported at the bottom of the table is the sum of the estimated coefficients on the distributed lag and has the interpretation of the long run effect of M1 on IP, assuming stationarity.

Note that selecting **View/Coefficient Tests** for an equation estimated with PDL terms tests the restrictions on γ , not on β . In this example, the coefficients on the fourth- (PDL05) and fifth-order (PDL06) terms are individually insignificant and very close to zero. To test

the joint significance of these two terms, click **View/Coefficient Tests/Wald-Coefficient Restrictions...** and type

$$c(6) = 0, \quad c(7) = 0$$

in the Wald Test dialog box (see “[Wald Test \(Coefficient Restrictions\)](#)” on page 368 for an extensive discussion of Wald tests in EViews). EViews displays the result of the joint test:

Wald Test:			
Equation: IP_PDL			
Test Statistic	Value	df	Probability
F-statistic	0.039852	(2, 353)	0.9609
Chi-square	0.079704	2	0.9609
Null Hypothesis Summary:			
Normalized Restriction (= 0)	Value	Std. Err.	
C(6)	2.58E-05	2448.827	
C(7)	-4.93E-05	5550.537	

Restrictions are linear in coefficients.

There is no evidence to reject the null hypothesis, suggesting that you could have fit a lower order polynomial to your lag structure.

Nonstationary Time Series

The theory behind ARMA estimation is based on stationary time series. A series is said to be (weakly or covariance) *stationary* if the mean and autocovariances of the series do not depend on time. Any series that is not stationary is said to be *nonstationary*.

A common example of a nonstationary series is the *random walk*:

$$y_t = y_{t-1} + \epsilon_t, \quad (13.44)$$

where ϵ is a stationary random disturbance term. The series y has a constant forecast value, conditional on t , and the variance is increasing over time. The random walk is a difference stationary series since the first difference of y is stationary:

$$y_t - y_{t-1} = (1 - L)y_t = \epsilon_t. \quad (13.45)$$

A difference stationary series is said to be *integrated* and is denoted as $I(d)$ where d is the order of integration. The order of integration is the number of unit roots contained in the series, or the number of differencing operations it takes to make the series stationary. For the random walk above, there is one unit root, so it is an $I(1)$ series. Similarly, a stationary series is $I(0)$.

Standard inference procedures do not apply to regressions which contain an integrated dependent variable or integrated regressors. Therefore, it is important to check whether a

series is stationary or not before using it in a regression. The formal method to test the stationarity of a series is the unit root test.

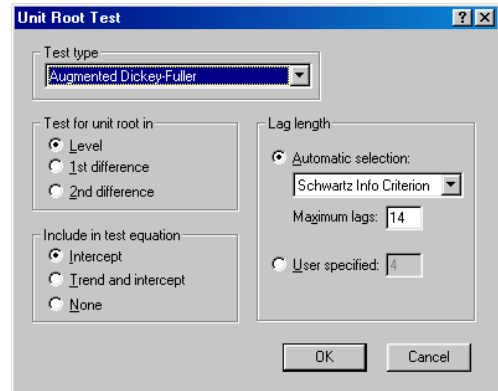
Unit Root Tests

EViews provides you with a variety of powerful tools for testing a series (or the first or second difference of the series) for the presence of a unit root. In addition to the existing Augmented Dickey-Fuller (1979) and Phillips-Perron (1998) tests, EViews now allows you to compute the GLS-detrended Dickey-Fuller (Elliot, Rothenberg, and Stock, 1996), Kwiatkowski, Phillips, Schmidt, and Shin (KPSS, 1992), Elliott, Rothenberg, and Stock Point Optimal (ERS, 1996), and Ng and Perron (NP, 2001) unit root tests. All of these tests are available as a view of a series.

Performing Unit Root Tests in EViews

The following discussion assumes that you are familiar with the basic forms of the unit root tests, and the associated options. We provide theoretical background for these tests in “[Basic Unit Root Theory](#)” beginning on page 333, and document the settings used when performing these tests.

To begin, double click on the series name to open the series window, and choose **View/Unit Root Test...**



You must specify four sets of options to carry out a unit root test. The first three settings (on the left-hand side of the dialog) determine the basic form of the unit root test. The fourth set of options (on the right-hand side of the dialog) consist of test specific advanced settings. You only need concern yourself with these settings if you wish to customize the calculation of your unit root test.

First, you should use the topmost combo box to select the type of unit root test that you wish to perform. You may choose one of six tests: ADF, DFGLS, PP, KPSS, ERS, and NP.

Next, specify whether you wish to test for a unit root in the level, first difference, or second difference of the series.

Lastly, choose your exogenous regressors. You can choose to include a constant, a constant and linear trend, or neither (there are limitations on these choices for some of the tests).

You can click on **OK** to compute the test using the specified settings, or you can customize your test using the advanced settings portion of the dialog.

The advanced settings for both the ADF and DFGLS tests allow you to specify how lagged difference terms p are to be included in the ADF test equation. You may choose to let EViews automatically select p , or you may specify a fixed positive integer value (if you choose automatic selection, you are given the additional option of selecting both the information criterion and maximum number of lags to be used in the selection procedure).

In this case, we have chosen to estimate an ADF test that includes a constant in the test regression and employs automatic lag length selection using a Schwarz Information Criterion (BIC) and a maximum lag length of 14. Applying these settings to data on the U. S. one-month Treasury bill rate for the period from March 1953 to July 1971, we can replicate Example 9.2 of Hayashi (2000, p. 596). The results are described below.

The first part of the unit root output provides information about the form of the test (the type of test, the exogenous variables, and lag length used), and contains the test output, associated critical values, and in this case, the p -value:

Null Hypothesis: TBILL has a unit root		
Exogenous: Constant		
Lag Length: 1 (Automatic based on SIC, MAXLAG=14)		
	t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic	-1.417410	0.5734
Test critical values:		
1% level	-3.459898	
5% level	-2.874435	
10% level	-2.573719	

*MacKinnon (1996) one-sided p -values.

The ADF statistic value is -1.417 and the associated one-sided p -value (for a test with 221 observations) is .573. In addition, EViews reports the critical values at the 1%, 5% and 10% levels. Notice here that the statistic t_{α} value is greater than the critical values so that we do not reject the null at conventional test sizes.

The second part of the output shows the intermediate test equation that EViews used to calculate the ADF statistic:

Augmented Dickey-Fuller Test Equation

Dependent Variable: D(TBILL)

Method: Least Squares

Date: 02/07/02 Time: 12:29

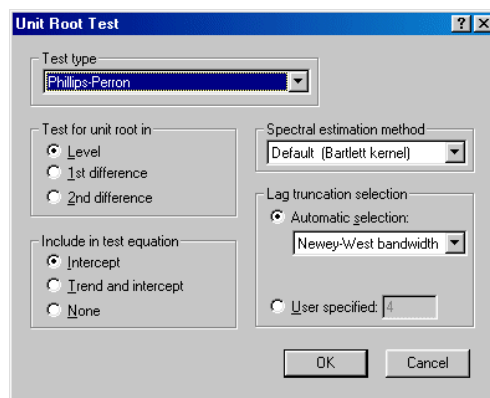
Sample: 1953:03 1971:07

Included observations: 221

Variable	Coefficient	Std. Error	t-Statistic	Prob.
TBILL(-1)	-0.022951	0.016192	-1.417410	0.1578
D(TBILL(-1))	-0.203330	0.067007	-3.034470	0.0027
C	0.088398	0.056934	1.552626	0.1220
R-squared	0.053856	Mean dependent var		0.013826
Adjusted R-squared	0.045175	S.D. dependent var		0.379758
S.E. of regression	0.371081	Akaike info criterion		0.868688
Sum squared resid	30.01882	Schwarz criterion		0.914817
Log likelihood	-92.99005	F-statistic		6.204410
Durbin-Watson stat	1.976361	Prob(F-statistic)		0.002395

If you had chosen to perform any of the other unit root tests (PP, KPSS, ERS, NP), the right side of the dialog would show the different options associated with the specified test. The options are associated with the method used to estimate the zero frequency spectrum term, f_0 , that is used in constructing the particular test statistic. As before, you only need pay attention to these settings if you wish to change from the EVIEWS defaults.

Here we have selected the PP test in the combo box. Note that the right-hand side of the dialog has changed, and now features a combo box for selecting the spectral estimation method. You may use this combo box to choose between various kernel or AR regression based estimators for f_0 . The entry labeled “Default” will show you the default estimator for the specific unit root test—here we see that the PP default uses a kernel sum-of-covariances estimator with Bartlett weights. If, instead, you had selected a NP test, the default entry would be “AR spectral-GLS”.



Lastly, you can control the lag length or bandwidth used for your spectral estimator. If you select one of the kernel estimation methods (Bartlett, Parzen, Quadratic Spectral), the dialog will give you a choice between using Newey-West or Andrews automatic bandwidth selection methods, or providing a user specified bandwidth. If, instead, you choose one of the AR spectral density estimation methods (AR Spectral - OLS, AR Spectral - OLS detrended, AR Spectral - GLS detrended), the dialog will prompt you to choose from various automatic lag length selection methods (using information criteria) or to provide a

user specified lag length. See [“Automatic Bandwidth and Lag Length Selection” on page 340](#).

Once you have chosen the appropriate settings for your test, click on the **OK** button. EViews reports the test statistic along with output from the corresponding test regression. For these tests, EViews reports the uncorrected estimate of the residual variance and the estimate of the frequency zero spectrum f_0 (labeled as the “HAC corrected variance”) in addition to the basic output. Running a PP test using the TBILL series yields:

Null Hypothesis: TBILL has a unit root		
Exogenous: Constant		
Bandwidth: 3.82 (Andrews using Bartlett kernel)		
	Adj. t-Stat	Prob.*
Phillips-Perron test statistic		
	-1.519035	0.5223
Test critical values:	1% level	-3.459898
	5% level	-2.874435
	10% level	-2.573719
*MacKinnon (1996) one-sided p-values.		
Residual variance (no correction)		0.141569
HAC corrected variance (Bartlett kernel)		0.107615

As with the ADF test, we fail to reject the null hypothesis of a unit root in the TBILL series at conventional significance levels.

Note that your test output will differ somewhat for alternative test specifications. For example, the KPSS output only provides the asymptotic critical values tabulated by KPSS:

Null Hypothesis: TBILL is stationary		
Exogenous: Constant		
Bandwidth: 11 (Newey-West Fixed using Bartlett kernel)		
	LM-Stat.	
Kwiatkowski-Phillips-Schmidt-Shin test statistic		
	1.537310	
Asymptotic critical values*:	1% level	0.739000
	5% level	0.463000
	10% level	0.347000
*Kwiatkowski-Phillips-Schmidt-Shin (1992, Table 1)		
Residual variance (no correction)		2.415060
HAC corrected variance (Bartlett kernel)		26.11028

Similarly, the NP test output will contain results for all four test statistics, along with the NP tabulated critical values.

A word of caution. You should note that the critical values reported by EViews are valid only for unit root tests of a data series, and will be invalid if the series is based on estimated values. For example, Engle and Granger (1987) proposed a two-step method to test

for cointegration. The test amounts to testing for a unit root in the residuals of a first stage regression. Since these residuals are estimates of the disturbance term, the asymptotic distribution of the test statistic differs from the one for ordinary series. The correct critical values for a subset of the tests may be found in Davidson and MacKinnon (1993, Table 20.2).

Basic Unit Root Theory

The following discussion outlines the basic features of unit root tests. By necessity, the discussion will be brief. Users who require detail should consult the original sources and standard references (see, for example, Davidson and MacKinnon, 1993, Chapter 20, Hamilton, 1994, Chapter 17, and Hayashi, 2000, Chapter 9).

Consider a simple AR(1) process:

$$y_t = \rho y_{t-1} + x_t' \delta + \epsilon_t, \quad (13.46)$$

where x_t are optional exogenous regressors which may consist of constant, or a constant and trend, ρ and δ are parameters to be estimated, and the ϵ_t are assumed to be white noise. If $|\rho| \geq 1$, y is a nonstationary series and the variance of y increases with time and approaches infinity. If $|\rho| < 1$, y is a (trend-)stationary series. Thus, the hypothesis of (trend-)stationarity can be evaluated by testing whether the absolute value of ρ is strictly less than one.

The unit root tests that EViews provides generally test the null hypothesis $H_0: \rho = 1$ against the one-sided alternative $H_1: \rho < 1$. In some cases, the null is tested against a point alternative. In contrast, the KPSS Lagrange Multiplier test evaluates the null of $H_0: \rho < 1$ against the alternative $H_1: \rho = 1$.

The Augmented Dickey-Fuller (ADF) Test

The standard DF test is carried out by estimating [Equation \(13.46\)](#) after subtracting y_{t-1} from both sides of the equation:

$$\Delta y_t = \alpha y_{t-1} + x_t' \delta + \epsilon_t, \quad (13.47)$$

where $\alpha = \rho - 1$. The null and alternative hypotheses may be written as

$$\begin{aligned} H_0: \alpha &= 0 \\ H_1: \alpha &< 0 \end{aligned} \quad (13.48)$$

and evaluated using the conventional t -ratio for α :

$$t_\alpha = \hat{\alpha} / (se(\hat{\alpha})) \quad (13.49)$$

where $\hat{\alpha}$ is the estimate of α , and $se(\hat{\alpha})$ is the coefficient standard error.

Dickey and Fuller (1979) show that under the null hypothesis of a unit root, this statistic does not follow the conventional Student's t -distribution, and they derive asymptotic

results and simulate critical values for various test and sample sizes. More recently, MacKinnon (1991, 1996) implements a much larger set of simulations than those tabulated by Dickey and Fuller. In addition, MacKinnon estimates response surfaces for the simulation results, permitting the calculation of Dickey-Fuller critical values and p -values for arbitrary sample sizes. The more recent MacKinnon critical value calculations are used by EViews in constructing test output.

The simple Dickey-Fuller unit root test described above is valid only if the series is an AR(1) process. If the series is correlated at higher order lags, the assumption of white noise disturbances ϵ_t is violated. The Augmented Dickey-Fuller (ADF) test constructs a parametric correction for higher-order correlation by assuming that the y series follows an AR(p) process and adding p lagged difference terms of the dependent variable y to the right-hand side of the test regression:

$$\Delta y_t = \alpha y_{t-1} + x_t' \delta + \beta_1 \Delta y_{t-1} + \beta_2 \Delta y_{t-2} + \dots + \beta_p \Delta y_{t-p} + v_t. \quad (13.50)$$

This augmented specification is then used to test (13.48) using the t -ratio (13.49). An important result obtained by Fuller is that the asymptotic distribution of the t -ratio for α is independent of the number of lagged first differences included in the ADF regression. Moreover, while the assumption that y follows an autoregressive (AR) process may seem restrictive, Said and Dickey (1984) demonstrate that the ADF test is asymptotically valid in the presence of a moving average (MA) component, provided that sufficient lagged difference terms are included in the test regression.

You will face two practical issues in performing an ADF test. First, you must choose whether to include exogenous variables in the test regression. You have the choice of including a constant, a constant and a linear time trend, or neither, in the test regression. One approach would be to run the test with both a constant and a linear trend since the other two cases are just special cases of this more general specification. However, including irrelevant regressors in the regression will reduce the power of the test to reject the null of a unit root. The standard recommendation is to choose a specification that is a plausible description of the data under both the null and alternative hypotheses. See, Hamilton (1994a, p. 501) for discussion.

Second, you will have to specify the number of lagged difference terms (which we will term the “lag length”) to be added to the test regression (0 yields the standard DF test; integers greater than 0 correspond to ADF tests). The usual (though not particularly useful) advice is to include a number of lags sufficient to remove serial correlation in the residuals. EViews provides both automatic and manual lag length selection options. For details, see “Automatic Bandwidth and Lag Length Selection” beginning on page 340.

Dickey-Fuller Test with GLS Detrending (DFGLS)

As noted above, you may elect to include a constant, or a constant and a linear time trend, in your ADF test regression. For these two cases, ERS (1996) propose a simple modification

of the ADF tests in which the data are detrended so that explanatory variables are “taken out” of the data prior to running the test regression.

ERS define a quasi-difference of y_t that depends on the value a representing the specific point alternative against which we wish to test the null:

$$d(y_t|a) = \begin{cases} y_t & \text{if } t = 1 \\ y_t - ay_{t-1} & \text{if } t > 1 \end{cases} \quad (13.51)$$

Next, consider an OLS regression of the quasi-differenced data $d(y_t|a)$ on the quasi-differenced $d(x_t|a)$:

$$d(y_t|a) = d(x_t|a)' \delta(a) + \eta_t \quad (13.52)$$

where x_t contains either a constant, or a constant and trend, and let $\hat{\delta}(a)$ be the OLS estimates from this regression.

All that we need now is a value for a . ERS recommend the use of $a = \bar{a}$, where

$$\bar{a} = \begin{cases} 1 - 7/T & \text{if } x_t = \{1\} \\ 1 - 13.5/T & \text{if } x_t = \{1, t\} \end{cases} \quad (13.53)$$

We now define the GLS detrended data, y_t^d , using the estimates associated with the \bar{a} :

$$y_t^d \equiv y_t - x_t' \hat{\delta}(\bar{a}) \quad (13.54)$$

Then the DFGLS test involves estimating the standard ADF test equation, (13.50), after substituting the GLS detrended y_t^d for the original y_t :

$$\Delta y_t^d = \alpha y_{t-1}^d + \beta_1 \Delta y_{t-1}^d + \dots + \beta_p y_{t-p}^d + v_t \quad (13.55)$$

Note that since the y_t^d are detrended, we do not include the x_t in the DFGLS test equation. As with the ADF test, we consider the t -ratio for $\hat{\alpha}$ from this test equation.

While the DFGLS t -ratio follows a Dickey-Fuller distribution in the constant only case, the asymptotic distribution differs when you include both a constant and trend. ERS (1996, Table 1, p. 825) simulate the critical values of the test statistic in this latter setting for $T = \{50, 100, 200, \infty\}$. Thus, the EViews lower tail critical values use the MacKinnon simulations for the no constant case, but are interpolated from the ERS simulated values for the constant and trend case. The null hypothesis is rejected for values that fall below these critical values.

The Phillips-Perron (PP) Test

Phillips and Perron (1988) propose an alternative (nonparametric) method of controlling for serial correlation when testing for a unit root. The PP method estimates the non-augmented DF test equation (13.47), and modifies the t -ratio of the α coefficient so that

serial correlation does not affect the asymptotic distribution of the test statistic. The PP test is based on the statistic:

$$\tau_\alpha = t_\alpha \left(\frac{\gamma_0}{f_0} \right)^{1/2} - \frac{T(f_0 - \gamma_0)(se(\hat{\alpha}))}{2f_0^{1/2}s} \quad (13.56)$$

where $\hat{\alpha}$ is the estimate, and t_α the t -ratio of α , $se(\hat{\alpha})$ is coefficient standard error, and s is the standard error of the test regression. In addition, γ_0 is a consistent estimate of the error variance in (13.47) (calculated as $(T-k)s^2/T$, where k is the number of regressors). The remaining term, f_0 , is an estimator of the residual spectrum at frequency zero.

There are two choices you will have make when performing the PP test. First, you must choose whether to include a constant, a constant and a linear time trend, or neither, in the test regression. Second, you will have to choose a method for estimating f_0 . EViews supports estimators for f_0 based on kernel-based sum-of-covariances, or on autoregressive spectral density estimation. See “[Frequency Zero Spectrum Estimation](#)” beginning on [page 338](#) for details.

The asymptotic distribution of the PP modified t -ratio is the same as that of the ADF statistic. EViews reports MacKinnon lower-tail critical and p -values for this test.

The Kwiatkowski, Phillips, Schmidt, and Shin (KPSS) Test

The KPSS (1992) test differs from the other unit root tests described here in that the series y_t is assumed to be (trend-) stationary under the null. The KPSS statistic is based on the the residuals from the OLS regression of y_t on the exogenous variables x_t :

$$y_t = x_t' \delta + u_t \quad (13.57)$$

The LM statistic is be defined as:

$$LM = \sum_t S(t)^2 / (T^2 f_0) \quad (13.58)$$

where f_0 , is an estimator of the residual spectrum at frequency zero and where $S(t)$ is a cumulative residual function:

$$S(t) = \sum_{r=1}^t \hat{u}_r \quad (13.59)$$

based on the residuals $\hat{u}_t = y_t - x_t' \hat{\delta}(0)$. We point out that the estimator of δ used in this calculation differs from the estimator for δ used by GLS detrending since it is based on a regression involving the original data, and not on the quasi-differenced data.

To specify the KPSS test, you must specify the set of exogenous regressors x_t and a method for estimating f_0 . See “[Frequency Zero Spectrum Estimation](#)” on [page 338](#) for discussion.

The reported critical values for the LM test statistic are based upon the asymptotic results presented in KPSS (Table 1, p. 166).

Elliot, Rothenberg, and Stock Point Optimal (ERS) Test

The ERS Point Optimal test is based on the quasi-differencing regression defined in Equations (13.52). Define the residuals from (13.52) as $\hat{\eta}_t(a) = d(y_t|a) - d(x_t|a)' \hat{\delta}(a)$, and let $SSR(a) = \sum \hat{\eta}_t^2(a)$ be the sum-of-squared residuals function. The ERS (feasible) point optimal test statistic of the null that $\alpha = 1$ against the alternative that $\alpha = \bar{a}$, is then defined as

$$P_T = (SSR(\bar{a}) - \bar{a}SSR(1))/f_0 \quad (13.60)$$

where f_0 is an estimator of the residual spectrum at frequency zero.

To compute the ERS test you must specify the set of exogenous regressors x_t and a method for estimating f_0 (see “Frequency Zero Spectrum Estimation” on page 338).

Critical values for the ERS test statistic are computed by interpolating the simulation results provided by ERS (1996, Table 1, p. 825) for $T = \{50, 100, 200, \infty\}$.

Ng and Perron (NP) Tests

Ng and Perron (2001) construct four test statistics that are based upon the GLS detrended data y_t^d . These test statistics are modified forms of Phillips and Perron Z_α and Z_t statistics, the Bhargava (1986) R_1 statistic, and the ERS Point Optimal statistic. First, define the term:

$$\kappa = \sum_{t=2}^T (y_{t-1}^d)^2 / T^2 \quad (13.61)$$

The modified statistics may then be written as

$$\begin{aligned} MZ_\alpha^d &= (T^{-1}(y_T^d)^2 - f_0)/(2\kappa) \\ MZ_t^d &= MZ_\alpha^d \times MSB \\ MSB^d &= (\kappa/f_0)^{1/2} \\ MP_T^d &= \begin{cases} (\bar{c}^2 \kappa - \bar{c}T^{-1}(y_T^d)^2)/f_0 & \text{if } x_t = \{1\} \\ (\bar{c}^2 \kappa + (1 - \bar{c})T^{-1}(y_T^d)^2)/f_0 & \text{if } x_t = \{1, t\} \end{cases} \end{aligned} \quad (13.62)$$

where

$$\bar{c} = \begin{cases} -7 & \text{if } x_t = \{1\} \\ -13.5 & \text{if } x_t = \{1, t\} \end{cases} \quad (13.63)$$

The NP tests require a specification for x_t and a choice of method for estimating f_0 (see

Frequency Zero Spectrum Estimation

Many of the unit root tests described above require a consistent estimate of the residual spectrum at frequency zero. EViews supports two classes of estimators for f_0 : kernel-based sum-of-covariances estimators, and autoregressive spectral density estimators.

Kernel Sum-of-Covariances Estimation

The kernel-based estimator of the frequency zero spectrum is based on a weighted sum of the autocovariances, with the weights are defined by a kernel function. The estimator takes the form

$$\hat{f}_0 = \sum_{j=-(T-1)}^{T-1} \hat{\gamma}(j) \cdot K(j/l) \quad (13.64)$$

where l is a bandwidth parameter (which acts as a truncation lag in the covariance weighting), K is a kernel function, and where $\hat{\gamma}(j)$, the j -th sample autocovariance of the residuals \tilde{u}_t , is defined as

$$\hat{\gamma}(j) = \sum_{t=j+1}^T (\tilde{u}_t \tilde{u}_{t-j}) / T \quad (13.65)$$

Note that the residuals \tilde{u}_t that EViews uses in estimating the autocovariance functions in (13.65) will differ depending on the specified unit root test:

Unit root test	Source of \tilde{u}_t residuals for kernel estimator
ADF, DFGLS	<i>not applicable.</i>
PP, ERS Point Optimal, NP	residuals from the Dickey-Fuller test equation, (13.47).
KPSS	residuals from the OLS test equation, (13.57).

EViews supports the following kernel functions:

Bartlett:	$K(x) = \begin{cases} 1 - x & \text{if } x \leq 1 \\ 0 & \text{otherwise} \end{cases}$
Parzen:	$K(x) = \begin{cases} 1 - 6x^2 + 6 x ^3 & \text{if } 0 \leq x \leq (1/2) \\ 2(1 - x)^3 & \text{if } (1/2) < x \leq 1 \\ 0 & \text{otherwise} \end{cases}$
Quadratic Spectral	$K(x) = \frac{25}{12\pi^2 x^2} \left(\frac{\sin(6\pi x/5)}{6\pi x/5} - \cos(6\pi x/5) \right)$

The properties of these kernels are described in Andrews (1991).

As with most kernel estimators, the choice of the bandwidth parameter l is of considerable importance. EViews allows you to specify a fixed parameter, or to have EViews select one using a data-dependent method. Automatic bandwidth parameter selection is discussed in “Automatic Bandwidth and Lag Length Selection” beginning on page 340.

Autoregressive Spectral Density Estimator

The autoregressive spectral density estimator at frequency zero is based upon the residual variance and estimated coefficients from the auxiliary regression:

$$\Delta \tilde{y}_t = \alpha \tilde{y}_{t-1} + \varphi \cdot \tilde{x}_t' \delta + \beta_1 \Delta \tilde{y}_{t-1} + \dots + \beta_p \Delta \tilde{y}_{t-p} + u_t \quad (13.66)$$

EViews provides three autoregressive spectral methods: OLS, OLS detrending, and GLS detrending, corresponding to difference choices for the data \tilde{y}_t . The following table summarizes the auxiliary equation estimated by the various AR spectral density estimators:

AR spectral method	Auxiliary AR regression specification
OLS	$\tilde{y}_t = y_t$, and $\varphi = 1$, $\tilde{x}_t = x_t$.
OLS detrended	$\tilde{y}_t = y_t - x_t' \hat{\delta}(0)$, and $\varphi = 0$.
GLS detrended	$\tilde{y}_t = y_t - x_t' \hat{\delta}(\bar{a}) = y_t^d$. and $\varphi = 0$.

where $\hat{\delta}(a)$ are the coefficient estimates from the regression defined in (13.52).

The AR spectral estimator of the frequency zero spectrum is defined as:

$$\hat{f}_0 = \hat{\sigma}_u^2 / (1 - \hat{\beta}_1 - \hat{\beta}_2 - \dots - \hat{\beta}_p) \quad (13.67)$$

where $\hat{\sigma}_u^2 = \sum \tilde{u}_t^2 / T$ is the residual variance, and $\hat{\beta}$ are the estimates from (13.66). We note here that EViews uses the non-degree-of-freedom estimator of the residual variance. As a result, spectral estimates computed in EViews may differ slightly from those obtained from other sources.

Not surprisingly, the spectrum estimator is sensitive to the number of lagged difference terms in the auxiliary equation. You may either specify a fixed parameter, or to have EViews automatically select one based on an information criterion. Automatic lag length selection is examined in “Automatic Bandwidth and Lag Length Selection” on page 340.

Default Settings

By default, EViews will choose the estimator of f_0 used by the authors of a given test specification. You may, of course, override the default settings and choose from either family of estimation methods. The default settings are listed below:

Unit root test	Frequency zero spectrum default method
ADF, DFGLS	<i>not applicable</i>
PP, KPSS	Kernel (Bartlett) sum-of-covariances
ERS Point Optimal	AR spectral regression (OLS)
NP	AR spectral regression (GLS-detrended)

Automatic Bandwidth and Lag Length Selection

There are three distinct situations in which EViews can automatically compute a bandwidth or a lag length parameter.

The first situation occurs when you are selecting the bandwidth parameter l for the kernel-based estimators of f_0 . For the kernel estimators, EViews provides you with the option of using the Newey-West (1994) or the Andrews (1991) data-based automatic bandwidth parameter methods. See the original sources for details. For those familiar with the Newey-West procedure, we note that EViews uses the lag selection parameter formulae given in the corresponding first lines of Table II-C. The Andrews method is based on an AR(1) specification.

The latter two occur when the unit root test requires estimation of a regression with a parametric correction for serial correlation as in the ADF and DFGLS test equation regressions, and in the AR spectral estimator for f_0 . In all of these cases, p lagged difference terms are added to a regression equation. The automatic selection methods choose p (less than the specified maximum) to minimize one of the following criteria:

Information criterion	Definition
Akaike (AIC)	$-2(l/T) + 2k/T$
Schwarz (SIC)	$-2(l/T) + k\log(T)/T$
Hannan-Quinn (HQ)	$-2(l/T) + 2k\log(\log(T))/T$
Modified AIC (MAIC)	$-2(l/T) + 2(k + \tau)/T$
Modified SIC (MSIC)	$-2(l/T) + (k + \tau)\log(T)/T$
Modified Hannan-Quinn (MHQ)	$-2(l/T) + 2(k + \tau)\log(\log(T))/T$

where the modification factor τ is computed as

$$\tau = \alpha^2 \sum_t \tilde{y}_{t-1}^2 / \hat{\sigma}_u^2 \quad (13.68)$$

for $\tilde{y}_t = y_t$, when computing the ADF test equation, and for \tilde{y}_t as defined in “Autoregressive Spectral Density Estimator” on page 339, when estimating f_0 . NP (2001) propose and examine the modified criteria, concluding with a recommendation of the MAIC.

For the information criterion selection methods, you must also specify an upper bound to the lag length. By default, EViews chooses a maximum lag of

$$k_{\max} = \text{int}(12(T/100)^{1/4}) \quad (13.69)$$

See Hayashi (2000, p. 594) for a discussion of the selection of this upper bound.

Commands

The command

```
equation eq_gdp.ls gdp c ar(1) ar(2) ma(1) ma(2)
```

fits an ARMA(2,2) model to the GDP series and stores the results in the equation object named EQ_GDP.

```
eq1.auto(4)
```

tests for serial correlation of up to order four in the residuals from equation EQ1.

```
eq1.correlogram(12)
```

displays the correlogram for the residuals in EQ1 up to lag 12.

```
equation eq2.ls gdp c pdl(m1,12,3)
```

fits a third degree polynomial to the coefficients of M1 up to twelve lags.

```
gdp.uroot(lag=4, const)
```

runs the ADF unit root test including a constant and four lags of first differences.

```
gdp.uroot(pp, trend, hac=bt, b=4.2)
```

runs the Phillips-Perron unit root test including a constant and linear trend with a Bartlett kernel and bandwidth of 4.2.

Chapter 14. Forecasting from an Equation

This chapter describes procedures for forecasting and computing fitted values from a single equation. The techniques described here are for forecasting with equation objects estimated using regression methods. Forecasts from equations estimated by specialized techniques, such as ARCH, binary, ordered, tobit, and count methods, are discussed in the corresponding chapters. Forecasting from a series using exponential smoothing methods is explained in “[Exponential Smoothing](#)” on page 190, and forecasting using multiple equations and models is described in [Chapter 23, “Models”](#), on page 601.

Forecasting from Equations in EViews

To illustrate the process of forecasting from an estimated equation, we begin with a simple example. Suppose we have data on the logarithm of monthly housing starts (HS) and the logarithm of the S&P index (SP) over the period 1959:01–1996:01. The data are contained in a workfile with range 1959:01–1998:12.

We estimate a regression of HS on a constant, SP, and the lag of HS, with an AR(1) to correct for residual serial correlation, using data for the period 1959:01–1990:12, and then use the model to forecast housing starts under a variety of settings. Following estimation, the equation results are held in the equation object EQ01:

Dependent Variable: HS
Method: Least Squares
Date: 10/18/97 Time: 12:44
Sample(adjusted): 1959:03 1990:01
Included observations: 371 after adjusting endpoints
Convergence achieved after 4 iterations

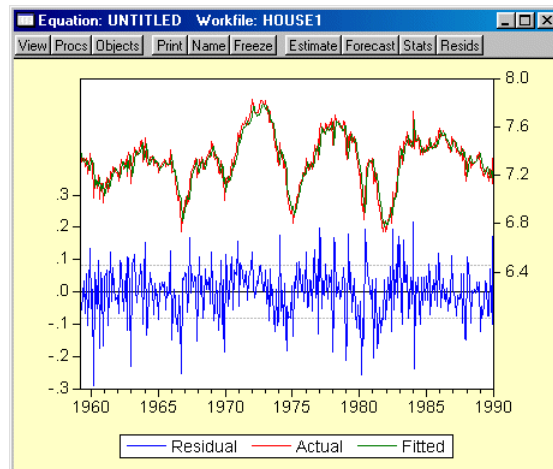
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.321924	0.117278	2.744975	0.0063
HS(-1)	0.952653	0.016218	58.74157	0.0000
SP	0.005222	0.007588	0.688249	0.4917
AR(1)	-0.271254	0.052114	-5.205027	0.0000

R-squared	0.861373	Mean dependent var	7.324051
Adjusted R-squared	0.860240	S.D. dependent var	0.220996
S.E. of regression	0.082618	Akaike info criterion	-2.138453
Sum squared resid	2.505050	Schwarz criterion	-2.096230
Log likelihood	400.6830	F-statistic	760.1338
Durbin-Watson stat	2.013460	Prob(F-statistic)	0.000000

Inverted AR Roots	- .27
-------------------	-------

Note that the estimation sample is adjusted by two observations to account for the first difference of the lagged endogenous variable used in deriving AR(1) estimates for this model.

To get a feel for the fit of the model, select **View/Actual, Fitted, Residual...**, then choose **Actual, Fitted, Residual Graph**:



The actual and fitted values depicted on the upper portion of the graph are virtually indistinguishable. This view provides little control over the process of producing fitted values, and does not allow you to save your fitted values. These limitations are overcome by using EViews built-in forecasting procedures to compute fitted values for the dependent variable.

How to Perform a Forecast

To forecast HS from this equation, push the **Forecast** button on the equation toolbar, or select **Procs/Forecast...**

You should provide the following information:

- **Series names.**
- **Forecasted series.** Fill in the edit box with the name to be given to the forecasted dependent variable. EViews suggests a name, but you can change it to any valid series name. The name should be different from the name of the dependent variable, since the forecast procedure will overwrite the data in the specified series.
- **S.E. (optional).** If desired, you may provide a name for the series to be filled with the forecast standard errors. If you do not provide a name, no forecast errors will be saved.

- **GARCH (optional)**. For models estimated by ARCH, you will be given a further option of saving forecasts of the conditional variances (GARCH terms). See [Chapter 16](#) for a discussion of GARCH estimation.
- **Forecasting method**. You have a choice between the following methods:
 - Dynamic**—calculates *multi-step forecasts* starting from the first period in the forecast sample.
 - Static**—calculates a sequence of *one-step ahead forecasts*, using actual, rather than forecasted values for lagged dependent variables.

and you can set the following options:

1. **Structural**—instructs EViews to ignore any ARMA terms in the equation when forecasting. By default, when your equation has ARMA terms, both dynamic and static solution methods form forecasts of the residuals. If you select **Structural**, all forecasts will ignore the residuals and will form predictions using only the structural part of the model.
2. **Sample range**. You must specify the sample to be used for the forecast. By default, EViews sets this sample to be the workfile sample. By specifying a sample outside the sample used in estimating your equation (the *estimation sample*), you can instruct EViews to produce out-of-sample forecasts.

Note that you are responsible for supplying the values for the independent variables in the out-of-sample forecasting period. For static forecasts, you must also supply the values for any lagged dependent variables.

- **Output**. You can choose to see the forecast output as a graph or a numerical forecast evaluation, or both. Forecast evaluation is only available if the forecast sample includes observations for which the dependent variable is observed.

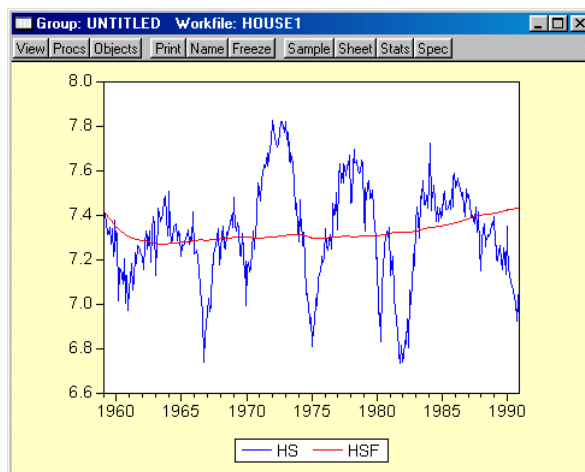
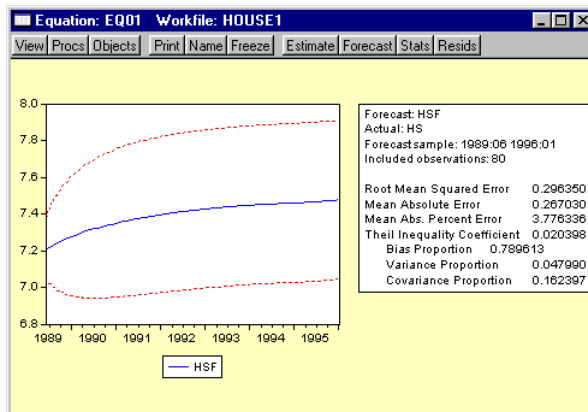
Illustration

Suppose we produce a dynamic forecast using EQ01 over the sample 1959:01 to 1996:01. The forecast values will be placed in the series HSF, and EViews will display both a graph of the forecasts and the plus and minus two standard error bands, as well as a forecast evaluation:

As noted in the output, the forecast values are saved in the series HSF. Since HSF is a standard EViews series, you may examine your forecasts using all of the standard tools for working with series objects.

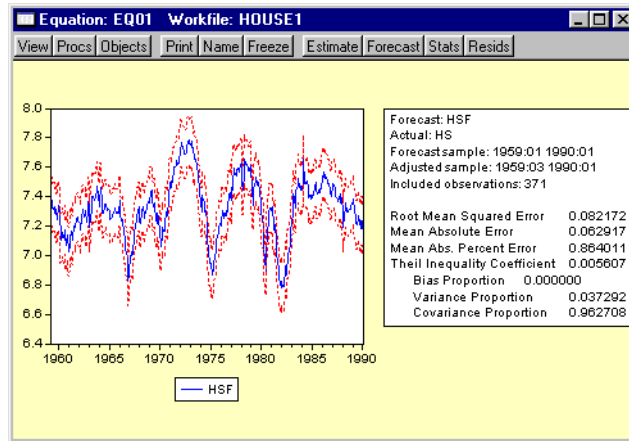
We can examine the actual versus fitted values by creating a group containing HS and HSF, and plotting the two series.

Click on **Quick/Show...** and enter HS and HSF. Then select **View/Graph/Line** to display the two series.



This is a dynamic forecast over the entire period from 1959:01 through 1996:01. For every period, the previously forecasted values for HS(-1) are used in forming a forecast of the subsequent value of HS. Note the considerable difference between this actual and fitted graph and the **Actual, Fitted, Residual Graph** depicted above.

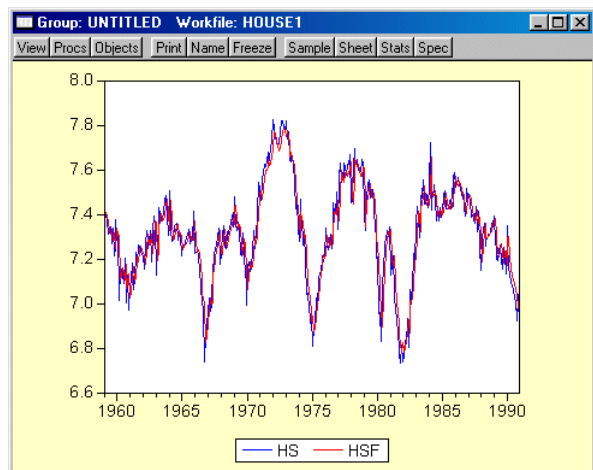
To perform a series of one-step ahead forecasts, click on **Forecast** on the equation toolbar, and select **Static** forecasts. EViews will display the forecast results:

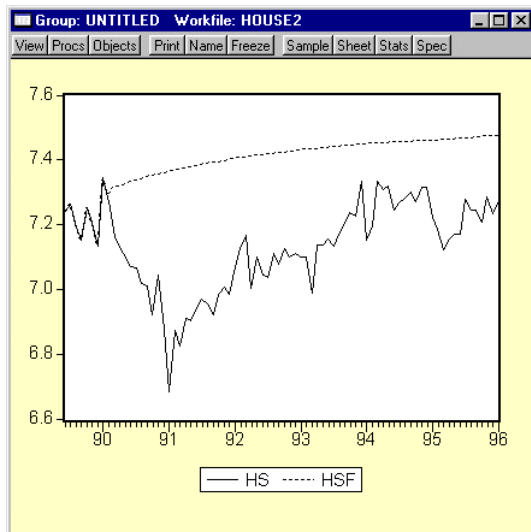


We can also compare the actual and fitted values from the static forecast by examining a line graph of a group containing HS and HSF.

The one-step ahead static forecasts are more accurate than the dynamic forecasts since, for each period, the actual value of $HS(-1)$ is used in forming the forecast of HS. These one-step ahead static forecasts are the same forecasts used in the **Actual, Fitted, Residual Graph** displayed above.

Lastly, we construct a dynamic forecast beginning in 1990:02 (the first period following the estimation sample) and ending in 1996:01. Keep in mind that data are available for SP for this entire period. The plot of the actual and the forecast values for 1989:01 to 1996:01 is given by:



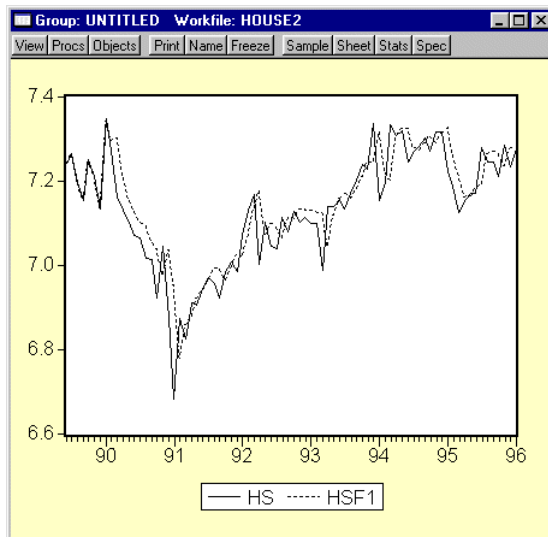


EViews backfills the forecast series prior to the forecast sample (up through 1990:01) and then dynamically forecasts HS for each subsequent period through 1996:01. This is the forecast that you would have constructed if, in 1990:01, you predicted values of HS from 1990:02 through 1996:01, given knowledge about the entire path of SP over that period.

The corresponding static forecast is displayed below:

Again, EViews backfills the values of the forecast series, HSF1, through 1990:01. This forecast is the one you would have constructed if, in 1990:01, you used all available data to estimate a model, and then used this estimated model to perform one-step ahead forecasts every month for the next six years.

The remainder of this chapter focuses on the details associated with the construction of these forecasts, and the corresponding forecast evaluations.



Forecast Basics

EViews stores the forecast results in the series specified in the **Forecast name** field. We will refer to this series as the *forecast series*.

The *forecast sample* specifies the observations for which EViews will try to compute fitted or forecasted values. If the forecast is not computable, a missing value will be returned. In some cases, EViews will carry out automatic adjustment of the sample to prevent a forecast consisting entirely of missing values (see “[Adjustment for Missing Values](#)” on [page 350](#), below). Note that the forecast sample may or may not overlap with the sample of observations used to estimate the equation.

For values not included in the forecast sample, there are two options. By default, EViews fills in the actual values of the dependent variable. If you turn off the **Insert actuals for out-of-sample** option, out-of-forecast-sample values will be filled with NAs.

As a consequence of these rules, *all data in the forecast series will be overwritten during the forecast procedure*. Existing values in the forecast series will be lost.

Computing Forecasts

For each observation in the forecast sample, EViews computes the fitted value of the dependent variable using the estimated parameters, the right-hand side exogenous variables, and, either the actual or estimated values for lagged endogenous variables and residuals. The method of constructing these forecasted values depends upon the estimated model and user specified settings.

To illustrate the forecasting procedure, we begin with a simple linear regression model with no lagged endogenous right-hand side variables, and no ARMA terms. Suppose that you have estimated the following equation specification:

$$y = c + xz$$

Now click on **Forecast**, specify a forecast period, and click **OK**.

For every observation in the forecast period, EViews will compute the fitted value of Y using the estimated parameters and the corresponding values of the regressors, X and Z :

$$\hat{y}_t = \hat{c}(1) + \hat{c}(2)x_t + \hat{c}(3)z_t. \quad (14.1)$$

You should make certain that you have valid values for the exogenous right-hand side variables for all observations in the forecast period. If any data are missing in the forecast sample, the corresponding forecast observation will be an NA.

Adjustment for Missing Values

There are two cases when a missing value will be returned for the forecast value. First, if any of the regressors have a missing value, and second, if any of the regressors are out of the range of the workfile. This includes the implicit error terms in AR models.

In the case of forecasts with no dynamic components in the specification (*i.e.* with no lagged endogenous or ARMA error terms), a missing value in the forecast series will not affect subsequent forecasted values. In the case where there are dynamic components, however, a single missing value in the forecasted series will propagate throughout all future values of the series.

As a convenience feature, EViews will move the starting point of the sample forward where necessary, until a valid forecast value is obtained. Without these adjustments, the user would have to figure out the appropriate number of presample values to skip, otherwise the forecast would consist entirely of missing values. For example, suppose you wanted to forecast dynamically from the following equation specification:

$$y \text{ c } y(-1) \text{ ar}(1)$$

If you specified the beginning of the forecast sample to the beginning of the workfile range, EViews will adjust forward the forecast sample by 2 observations, and will use the pre-forecast-sample values of the lagged variables (the loss of 2 observations occurs because the residual loses one observation due to the lagged endogenous variable so that the forecast for the error term can begin only from the third observation.)

Forecast Errors and Variances

Suppose the “true” model is given by:

$$y_t = x_t' \beta + \epsilon_t, \quad (14.2)$$

where ϵ_t is an independent, and identically distributed, mean zero random disturbance, and β is a vector of unknown parameters. Below, we relax the restriction that the ϵ 's be independent.

The true model generating y is not known, but we obtain estimates b of the unknown parameters β . Then, setting the error term equal to its mean value of zero, the (point) forecasts of y are obtained as

$$\hat{y}_t = x_t' b. \quad (14.3)$$

Forecasts are made with error, where the error is simply the difference between the actual and forecasted value $e_t = y_t - x_t' b$. Assuming that the model is correctly specified, there are two sources of forecast error: residual uncertainty and coefficient uncertainty.

Residual Uncertainty

The first source of error, termed *residual* or *innovation uncertainty*, arises because the innovations ϵ in the equation are unknown for the forecast period, and are replaced with their expectations. While the residuals are zero in expected value, the individual values are non-zero; the larger the variation in the individual errors, the greater the overall error in the forecasts.

The standard measure of this variation is the standard error of the regression (labeled “S.E. of regression” in the equation output). Residual uncertainty is usually the largest source of forecast error.

In dynamic forecasts, innovation uncertainty is compounded by the fact that lagged dependent variables and ARMA terms depend on lagged innovations. EViews also sets these equal to their expected values, which differ randomly from realized values. This additional source of forecast uncertainty tends to rise over the forecast horizon, leading to a pattern of increasing forecast errors. Forecasting with lagged dependent variables and ARMA terms is discussed in more detail below.

Coefficient Uncertainty

The second source of forecast error is *coefficient uncertainty*. The estimated coefficients b of the equation deviate from the true coefficients β in a random fashion. The standard error of the estimated coefficient, given in the regression output, is a measure of the precision with which the estimated coefficients measure the true coefficients.

The effect of coefficient uncertainty depends upon the exogenous variables. Since the estimated coefficients are multiplied by the exogenous variables x in the computation of forecasts, the more the exogenous variables deviate from their mean values, the greater is the forecast uncertainty.

Forecast Variability

The variability of forecasts is measured by the forecast standard errors. For a single equation without lagged dependent variables or ARMA terms, the forecast standard errors are computed as

$$\text{forecast se} = s \sqrt{1 + x_t'(X'X)^{-1}x_t} \quad (14.4)$$

where s is the standard error of regression. These standard errors account for both innovation (the first term) and coefficient uncertainty (the second term). Point forecasts made from linear regression models estimated by least squares are optimal in the sense that they have the smallest forecast variance among forecasts made by linear unbiased estimators. Moreover, if the innovations are normally distributed, the forecast errors have a t -distribution and forecast intervals can be readily formed.

If you supply a name for the forecast standard errors, EViews computes and saves a series of forecast standard errors in your workfile. You can use these standard errors to form forecast intervals. If you choose the **Do graph** option for output, EViews will plot the forecasts with plus and minus two standard error bands. These two standard error bands provide an approximate 95% forecast interval; if you (hypothetically) make many forecasts, the actual value of the dependent variable will fall inside these bounds 95 percent of the time.

Additional Details

EViews accounts for the additional forecast uncertainty generated when lagged dependent variables are used as explanatory variables (see [“Forecasts with Lagged Dependent Variables” on page 355](#)).

There are several other special cases, involving dependent variables that are defined by expression, where coefficient uncertainty is ignored. These cases are described in [“Forecasting Equations with Formulas” on page 359](#).

Forecast standard errors derived from equations estimated by nonlinear least squares and equations that include PDL (polynomial distributed lag) terms only account for the residual uncertainty ([“Forecasting with Nonlinear and PDL Specifications” on page 364](#)).

Forecast Evaluation

Suppose we construct a dynamic forecast for HS over the period 1990:02 to 1996:01 using our estimated housing equation. If the **Forecast evaluation** option is checked, and there are actual data for the forecasted variable for the forecast sample, EViews reports a table of statistical results evaluating the forecast:

Forecast: HSF	
Actual: HS	
Sample: 1990:02 1996:01	
<u>Include observations: 72</u>	
Root Mean Squared Error	0.318700
Mean Absolute Error	0.297261
Mean Absolute Percentage Error	4.205889
Theil Inequality Coefficient	0.021917
Bias Proportion	0.869982
Variance Proportion	0.082804
Covariance Proportion	0.047214

Note that EViews cannot compute a forecast evaluation if there are no data for the dependent variable for the forecast sample.

The forecast evaluation is saved in one of two formats. If you turn on the **Do graph** option, the forecasts are included along with a graph of the forecasts. If you wish to display the evaluations in their own table, you should turn off the **Do graph** option in the Forecast dialog box.

Suppose the forecast sample is $j = T + 1, T + 2, \dots, T + h$, and denote the actual and forecasted value in period t as y_t and \hat{y}_t , respectively. The reported forecast error statistics are computed as follows:

Root Mean Squared Error	$\sqrt{\sum_{t=T+1}^{T+h} (\hat{y}_t - y_t)^2 / h}$
Mean Absolute Error	$\sum_{t=T+1}^{T+h} \hat{y}_t - y_t / h$
Mean Absolute Percentage Error	$100 \sum_{t=T+1}^{T+h} \left \frac{\hat{y}_t - y_t}{y_t} \right / h$
Theil Inequality Coefficient	$\frac{\sqrt{\sum_{t=T+1}^{T+h} (\hat{y}_t - y_t)^2 / h}}{\sqrt{\sum_{t=T+1}^{T+h} \hat{y}_t^2 / h} + \sqrt{\sum_{t=T+1}^{T+h} y_t^2 / h}}$

The first two forecast error statistics depend on the scale of the dependent variable. These should be used as relative measures to compare forecasts for the same series across different models; the smaller the error, the better the forecasting ability of that model according to that criterion. The remaining two statistics are scale invariant. The Theil inequality coefficient always lies between zero and one, where zero indicates a perfect fit.

The mean squared forecast error can be decomposed as

$$\sum (\hat{y}_t - y_t)^2 / h = ((\sum \hat{y}_t / h) - \bar{y})^2 + (s_{\hat{y}} - s_y)^2 + 2(1 - r)s_{\hat{y}}s_y \quad (14.5)$$

where $\sum \hat{y}_t / h$, \bar{y} , $s_{\hat{y}}$, s_y are the means and (biased) standard deviations of \hat{y}_t and y , and r is the correlation between \hat{y} and y . The proportions are defined as:

Bias Proportion	$\frac{((\sum \hat{y}_t / h) - \bar{y})^2}{\sum (\hat{y}_t - y_t)^2 / h}$
Variance Proportion	$\frac{(s_{\hat{y}} - s_y)^2}{\sum (\hat{y}_t - y_t)^2 / h}$
Covariance Proportion	$\frac{2(1 - r)s_{\hat{y}}s_y}{\sum (\hat{y}_t - y_t)^2 / h}$

- The bias proportion tells us how far the mean of the forecast is from the mean of the actual series.
- The variance proportion tells us how far the variation of the forecast is from the variation of the actual series.
- The covariance proportion measures the remaining unsystematic forecasting errors.

Note that the bias, variance, and covariance proportions add up to one.

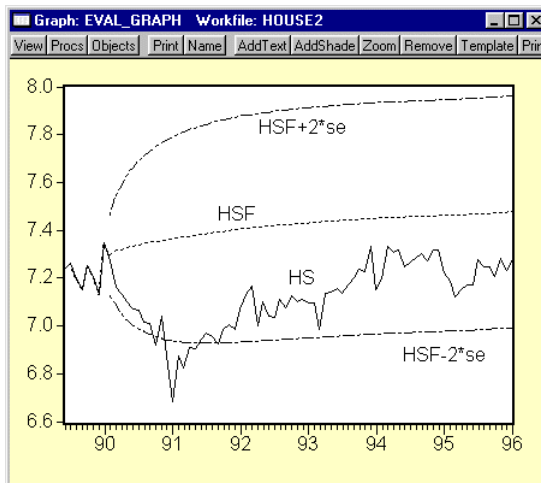
If your forecast is “good”, the bias and variance proportions should be small so that most of the bias should be concentrated on the covariance proportions. For additional discussion of forecast evaluation, see Pindyck and Rubinfeld (1991, Chapter 12).

For the example output, the bias proportion is large, indicating that the mean of the forecasts does a poor job of tracking the mean of the dependent variable. To check this, we will plot the forecasted series together with the actual series in the forecast sample with the two standard error bounds. Suppose we saved the forecasts and their standard errors as HSF and HSFSE, respectively. Then the plus and minus two standard error series can be generated by the commands

```
smp1 1990:02 1996:01
series hsf_high = hsf + 2*hsfse
series hsf_low = hsf - 2*hsfse
```

Create a group containing the four series. You can highlight the four series HS, HSF, HSF_HIGH, and HSF_LOW, double click on the selected area, and select **Open Group**, or you can select **Quick/Show...** and enter the four series names. Once you have the group open, select **View/Graph/Line**.

The forecasts completely miss the downturn at the start of the 1990's, but, subsequent to the recovery, track the trend reasonably well from 1992 to 1996.



Forecasts with Lagged Dependent Variables

Forecasting is complicated by the presence of lagged dependent variables on the right-hand side of the equation. For example, we can augment the earlier specification to include the first lag of Y :

$$y = \alpha + \beta x + \gamma z + \delta y(-1)$$

and click on the **Forecast** button and fill out the series names in the dialog as above. There is some question, however, as to how we should evaluate the lagged value of Y that appears on the right-hand side of the equation. There are two possibilities: dynamic forecasting and static forecasting.

Dynamic Forecasting

If you select dynamic forecasting, EViews will perform a multi-step forecast of Y , beginning at the start of the forecast sample. For our single lag specification above:

- The initial observation in the forecast sample will use the actual value of lagged Y . Thus, if S is the first observation in the forecast sample, EViews will compute

$$\hat{y}_S = \hat{c}(1) + \hat{c}(2)x_S + \hat{c}(3)z_S + \hat{c}(4)y_{S-1}, \quad (14.6)$$

where y_{S-1} is the value of the lagged endogenous variable in the period prior to the start of the forecast sample. This is the one-step ahead forecast.

- Forecasts for subsequent observations will use the previously *forecasted* values of Y :

$$\hat{y}_{S+k} = \hat{c}(1) + \hat{c}(2)x_{S+k} + \hat{c}(3)z_{S+k} + \hat{c}(4)\hat{y}_{S+k-1}. \quad (14.7)$$

- These forecasts may differ significantly from the one-step ahead forecasts.

If there are additional lags of Y in the estimating equation, the above algorithm is modified to account for the non-availability of lagged forecasted values in the additional period. For example, if there are three lags of Y in the equation:

- The first observation (S) uses the actual values for all three lags, y_{S-3} , y_{S-2} , and y_{S-1} .
- The second observation ($S+1$) uses actual values for y_{S-2} and y_{S-1} and the forecasted value \hat{y}_S of the first lag of y_{S+1} .
- The third observation ($S+2$) will use the actual values for y_{S-1} , and forecasted values \hat{y}_{S+1} and \hat{y}_S for the first and second lags of y_{S+2} .
- All subsequent observations will use the forecasted values for all three lags.

The selection of the start of the forecast sample is very important for dynamic forecasting. The dynamic forecasts are true multi-step forecasts (from the start of the forecast sample), since they use the recursively computed forecast of the lagged value of the dependent vari-

able. These forecasts may be interpreted as the forecasts for subsequent periods that would be computed using information available at the start of the forecast sample.

Dynamic forecasting requires that data for the exogenous variables be available for every observation in the forecast sample, and that values for any lagged dependent variables be observed at the start of the forecast sample (in our example, y_{S-1} , but more generally, any lags of y). If necessary, the forecast sample will be adjusted.

Any missing values for the explanatory variables will generate an NA for that observation and in all subsequent observations, via the dynamic forecasts of the lagged dependent variable.

Static Forecasting

Static forecasting performs a series of one-step ahead forecasts of the dependent variable:

- For each observation in the forecast sample, EViews computes

$$\hat{y}_{S+k} = \hat{c}(1) + \hat{c}(2)x_{S+k} + \hat{c}(3)z_{S+k} + \hat{c}(4)y_{S+k-1} \quad (14.8)$$

always using the actual value of the lagged endogenous variable.

Static forecasting requires that data for both the exogenous and any lagged endogenous variables be observed for every observation in the forecast sample. As above, EViews will, if necessary, adjust the forecast sample to account for pre-sample lagged variables. If the data are not available for any period, the forecasted value for that observation will be an NA. The presence of a forecasted value of NA does not have any impact on forecasts for subsequent observations.

A Comparison of Dynamic and Static Forecasting

Both methods will always yield identical results in the first period of a multi-period forecast. Thus, two forecast series, one dynamic and the other static, should be identical for the first observation in the forecast sample.

The two methods will differ for subsequent periods only if there are lagged dependent variables or ARMA terms.

Forecasting with ARMA Errors

Forecasting from equations with ARMA components involves some additional complexities. When you use the AR or MA specifications, you will need to be aware of how EViews handles the forecasts of the lagged residuals which are used in forecasting.

Structural Forecasts

By default, EViews will forecast values for the residuals using the estimated ARMA structure, as described below.

For some types of work, you may wish to assume that the ARMA errors are always zero. If you select the structural forecast option by checking **Structural (ignore ARMA)**, EViews computes the forecasts assuming that the errors are always zero. If the equation is estimated without ARMA terms, this option has no effect on the forecasts.

Forecasting with AR Errors

For equations with AR errors, EViews adds forecasts of the residuals from the equation to the forecast of the structural model that is based on the right-hand side variables.

In order to compute an estimate of the residual, EViews requires estimates or actual values of the lagged residuals. For the first observation in the forecast sample, EViews will use pre-sample data to compute the lagged residuals. If the pre-sample data needed to compute the lagged residuals are not available, EViews will adjust the forecast sample, and backfill the forecast series with actual values (see the discussion of “[Adjustment for Missing Values](#)” on page 350).

If you choose the **Dynamic** option, both the lagged dependent variable and the lagged residuals will be forecasted dynamically. If you select **Static**, both will be set to the actual lagged values. For example, consider the following AR(2) model:

$$\begin{aligned} y_t &= x_t' \beta + u_t \\ u_t &= \rho_1 u_{t-1} + \rho_2 u_{t-2} + \epsilon_t \end{aligned} \quad (14.9)$$

Denote the fitted residuals as $e_t = y_t - x_t' b$, and suppose the model was estimated using data up to $t = S - 1$. Then, provided that the x_t values are available, the static and dynamic forecasts for $t = S, S + 1, \dots$, are given by:

	static	dynamic
\hat{y}_S	$x_S' b + \hat{\rho}_1 e_{S-1} + \hat{\rho}_2 e_{S-2}$	$x_S' b + \hat{\rho}_1 e_{S-1} + \hat{\rho}_2 e_{S-2}$
\hat{y}_{S+1}	$x_{S+1}' b + \hat{\rho}_1 e_S + \hat{\rho}_2 e_{S-1}$	$x_{S+1}' b + \hat{\rho}_1 \hat{u}_S + \hat{\rho}_2 e_{S-1}$
\hat{y}_{S+2}	$x_{S+2}' b + \hat{\rho}_1 e_{S+1} + \hat{\rho}_2 e_S$	$x_{S+2}' b + \hat{\rho}_1 \hat{u}_{S+1} + \hat{\rho}_2 \hat{u}_S$

where the residuals $\hat{u}_t = \hat{y}_t - x_t' b$ are formed using the forecasted values of y_t . For subsequent observations, the dynamic forecast will always use the residuals based upon the multi-step forecasts, while the static forecast will use the one-step ahead forecast residuals.

Forecasting with MA Errors

In general, you need not concern yourselves with the details of MA forecasting, since EViews will do all of the work for you. For those of you who are interested in the details of dynamic forecasting, however, the following discussion should aid you in relating EViews results with those obtained from other sources.

The first step in computing forecasts using MA terms is to obtain fitted values for the innovations in the pre-forecast sample period. For example, if you are forecasting the values of y , beginning in period S , with a simple MA(q),

$$\hat{y}_S = \epsilon_S + \hat{\phi}_1 \epsilon_{S-1} + \dots + \hat{\phi}_q \epsilon_{S-q}, \quad (14.10)$$

you will need values for the lagged innovations, $\epsilon_{S-1}, \epsilon_{S-2}, \dots, \epsilon_{S-q}$.

To compute these pre-forecast innovations, EViews will first assign values for the q innovations prior to the start of the *estimation* sample, $\epsilon_0, \epsilon_{-1}, \epsilon_{-2}, \dots, \epsilon_{-q}$. If your equation is estimated with backcasting turned on, EViews will perform backcasting to obtain these values. If your equation is estimated with backcasting turned off, or if the forecast sample precedes the estimation sample, the initial values will be set to zero.

Given the initial values, EViews will fit the values of subsequent innovations, $\epsilon_1, \epsilon_2, \dots, \epsilon_q, \dots, \epsilon_{S-1}$, using forward recursion. The backcasting and recursion procedures are described in detail in the discussion of backcasting in ARMA models in [“Backcasting MA terms” on page 320](#).

Note the difference between this procedure and the approach for AR errors outlined above, in which the forecast sample is adjusted forward and the pre-forecast values are set to actual values.

The choice between dynamic and static forecasting has two primary implications:

- Once the q pre-sample values for the innovations are computed, dynamic forecasting sets subsequent innovations to zero. Static forecasting extends the forward recursion through the end of the estimation sample, allowing for a series of one-step ahead forecasts of both the structural model and the innovations.
- When computing static forecasts, EViews uses the entire estimation sample to backcast the innovations. For dynamic MA forecasting, the backcasting procedure uses observations from the beginning of the estimation sample to either the beginning of the forecast period, or the end of the estimation sample, whichever comes first.

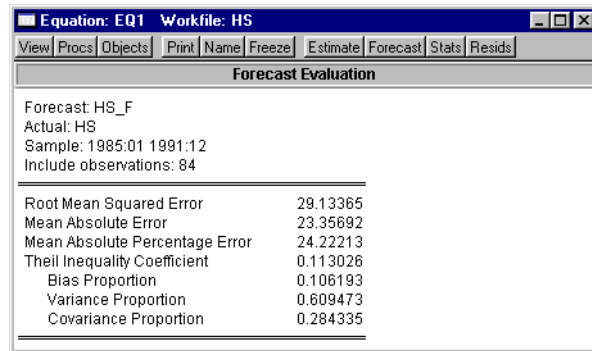
Example

As an example of forecasting from ARMA models, consider forecasting the monthly new housing starts (HS) series. The estimation period is 1959:01–1984:12 and we forecast for

the period 1985:01–1991:12. We estimated the following simple multiplicative seasonal autoregressive model:

$$hs \ c \ ar(1) \ sar(12)$$

To forecast from this estimated model, click **Forecast** on the equation toolbar. The forecast evaluation statistics for the model are shown below:



The large variance proportion indicates that the forecasts are not tracking the variation in the actual HS series. To plot the actual and forecasted series together with the two standard error bands, you can type

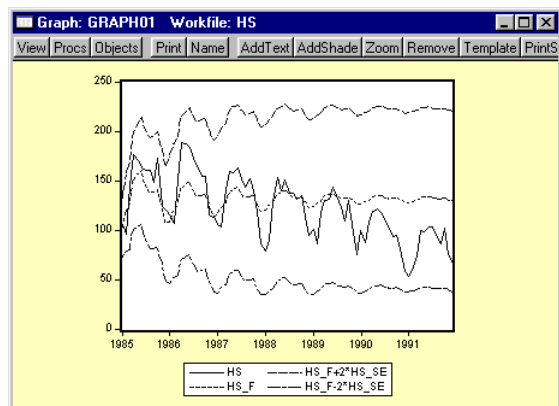
```
smp1 1985:01 1991:12
plot hs hs_f hs_f+2*hs_se hs_f-2*hs_se
```

where HS_F and HS_SE are the forecasts and standard errors of HS.

As indicated by the large variance proportion, the forecasts track the seasonal movements in HS only at the beginning of the forecast sample and quickly flattens out to the mean forecast value.

Forecasting Equations with Formulas

EViews allows estimation and forecasting with equations where the left-hand variable is a transformation specified by a formula. When forecasting from equations with formulas on the



left-hand side, three things determine the forecasting procedures and options that are available:

- whether the formula is linear or nonlinear
- whether the formula includes lagged variables
- whether the formula includes estimated coefficients

Point Forecasts

EViews always provides you with the option to forecast the transformed dependent variable. If the transformation can be normalized and solved for the first series in the formula, then EViews also provides you with the option to forecast the normalized series.

For example, suppose you estimated an equation with the specification

$$(\log(x) + z) \subset y$$

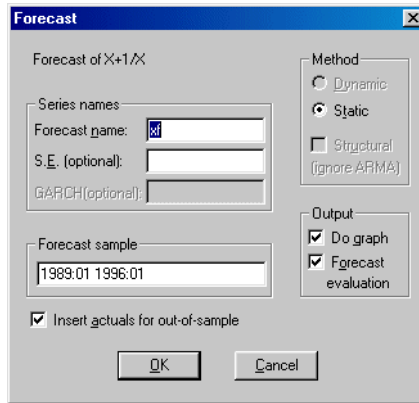
If you press the **Forecast** button, the Forecast dialog looks like this:

Notice that the dialog provides you with two choices for the series to forecast: the normalized series, X , and the dependent variable, $\text{LOG}(X) + Z$. X is the normalized series since it is the first series that appears on the left-hand side of the equation.

However, if you specify the equation as

$$x + 1/x = c(1) + c(2) * y$$

EViews will not be able to normalize the dependent variable and the Forecast dialog looks like this:



The dialog only allows you to forecast the transformed dependent variable, since EViews does not know how to normalize and solve for X . Note also that only static forecasts are available for this case. This restriction holds since EViews will not be able to solve for any lagged values of X on the right hand-side.

If the formula can be normalized, EViews will compute the forecasts of the transformed dependent variable by transforming the forecasts of the normalized series. This has important consequences when the formula includes lagged series. For example, consider the following two models:

```
series dy = d(y)
equation eq1.ls d(y) c x
equation eq2.ls dy c x
```

The dynamic forecasts of the first difference $D(Y)$ from the first equation will be numerically identical to those for DY from the second equation. However, the static forecasts for $D(Y)$ from the two equations will not be identical. This is because in the first equation, EViews knows that the dependent variable is a transformation of Y , so it will use the actual lagged value of Y in computing the static forecast of the first difference $D(Y)$. In the second equation, EViews simply views DY as an ordinary series, so that C and X are used to compute the static forecast.

Plotted Standard Errors

When you select **Do graph** in the forecast dialog, EViews will plot the forecasts, along with plus and minus two standard error bands. When you estimate an equation with an expression for the left-hand side, EViews will plot the standard error bands for either the normalized or the unnormalized expression, depending upon which term you elect to forecast.

If you elect to predict the normalized dependent variable, EViews will automatically account for the nonlinearity in the standard error transformation. The next section provides additional details on the procedure used to normalize the upper and lower bounds.

Saved Forecast Standard Errors

If you provide a name in this edit box, EViews will store the standard errors of the underlying series or expression that you chose to forecast.

When the dependent variable of the equation is a simple series or a formula involving only *linear* transformations, the saved standard errors will be exact (except where the forecasts do not account for coefficient uncertainty, as described below). If the dependent variable involves nonlinear transformations, the saved forecast standard errors will be exact if you choose to forecast the entire formula. If you choose to forecast the underlying endogenous series, the forecast uncertainty cannot be computed exactly, and EViews will provide a linear (first-order) approximation to the forecast standard errors.

Consider the following equations involving a formula dependent variable:

$$d(y) \subset x$$

$$\log(y) \subset x$$

For the first equation you may choose to forecast either Y or $D(Y)$. In both cases, the forecast standard errors will be exact, since the expression involves only linear transformations. The two standard errors will, however, differ in dynamic forecasts since the forecast standard errors for Y take into account the forecast uncertainty from the lagged value of Y . In the second example, the forecast standard errors for $\log(Y)$ will be exact. If, however, you request a forecast for Y itself, the standard errors saved in the series will be the approximate (linearized) forecast standard errors for Y .

Note that when EViews displays a graph view of the forecasts together with standard error bands, the standard error bands are always exact. Thus, in forecasting the underlying dependent variable in a nonlinear expression, the standard error bands will not be the same as those you would obtain by constructing series using the linearized standard errors saved in the workfile.

Suppose in our second example above that you store the forecast of Y and its standard errors in the workfile as the series $YHAT$ and SE_YHAT . Then the *approximate* two standard error bounds can be generated manually as:

```
series yhat_high1 = yhat + 2*se_yhat
```

```
series yhat_low1 = yhat - 2*se_yhat
```

These forecast error bounds will be symmetric about the point forecasts $YHAT$.

On the other hand, when EViews plots the forecast error bounds of Y, it proceeds in two steps. It first obtains the forecast of $\log(Y)$ and its standard errors (say LYHAT and SE_LYHAT) and forms the forecast error bounds on $\log(Y)$:

```
lyhat + 2*se_lyhat
lyhat - 2*se_lyhat
```

It then normalizes (inverts the transformation) of the two standard error bounds to obtain the prediction interval for Y:

```
series yhat_high2 = exp(lyhat + 2*se_lyhat)
series yhat_low2 = exp(lyhat - 2*se_lyhat)
```

Because this transformation is a non-linear transformation, these bands will not be symmetric around the forecast.

To take a more complicated example, suppose that you generate the series DLY and LY, and then estimate three equivalent models:

```
series dly =dlog(y)
series ly = log(y)
equation eq1.ls dlog(y) c x
equation eq2.ls d(ly) c x
equation eq3.ls dly c x
```

The estimated equations from the three models are numerically identical. If you choose to forecast the underlying dependent (normalized) series from each model, EQ1 will forecast Y, EQ2 will forecast LY (the log of Y), and EQ3 will forecast DLY (the log of the first difference of Y, $\log(Y) - \log(Y(-1))$). The forecast standard errors saved from EQ1 will be linearized approximations to the forecast standard error of Y, while those from the latter two will be exact for the forecast standard error of $\log Y$ and the log of the first difference of Y.

Static forecasts from all three models are identical because the forecasts from previous periods are not used in calculating this period's forecast when performing static forecasts. For dynamic forecasts, the log of the forecasts from EQ1 will be identical to those from EQ2 and the log first difference of the forecasts from EQ1 will be identical to the first difference of the forecasts from EQ2 and to the forecasts from EQ3. For static forecasts, the log first difference of the forecasts from EQ1 will be identical to the first difference of the forecasts from EQ2. However, these forecasts differ from those obtained from EQ3 because EViews does not know that the generated series DLY is actually a difference term so that it does not use the dynamic relation in the forecasts.

A final word of caution: when you have lagged dependent variables, you should avoid referring to the lagged series before the current series in a dependent variable expression. For example, consider the two equation specifications:

$$\begin{aligned}d(y) & c \ x \\ -y(-1)+y & c \ x\end{aligned}$$

Both models have the first difference of Y as the dependent variable and the estimation results are identical for the two models. However, if you forecast Y from the second model, EViews will try to calculate the forecasts of Y using leads of the actual series Y . These forecasts of Y will differ from those produced by the first model, and may not be what you expected.

Forecasting with Nonlinear and PDL Specifications

As explained above, forecast errors can arise from two sources: coefficient uncertainty and innovation uncertainty. For linear regression models, the forecast standard errors account for both coefficient and innovation uncertainty. However, if the model is nonlinear in the parameters (or if it contains a PDL specification), then the standard errors ignore coefficient uncertainty. EViews will display a message in the status line at the bottom of the EViews window when forecast standard errors only account for innovation uncertainty.

For example, consider the three specifications

$$\begin{aligned}\log(y) & c \ x \\ y & = c(1) + c(2) * x \\ y & = \exp(c(1) * x) \\ y & c \ x \text{ pdl}(z, 4, 2)\end{aligned}$$

Forecast standard errors from the first and second models account for both coefficient and innovation uncertainty since both models are linear in the coefficients. The third and fourth specifications have forecast standard errors that account only for residual uncertainty.

One additional case requires mention. Suppose you have the specification:

$$y - c(1) = c(3) + c(2) * x$$

Despite the fact that this specification is linear in the parameters, EViews will ignore coefficient uncertainty. Forecast standard errors for any specification that contains coefficients on the left-hand side of the equality will only reflect residual uncertainty.

Commands

To obtain static (one-step ahead) forecasts, follow the name of the estimated equation, a dot, the command `fit`, a name for the fitted series, and optionally a name for the standard errors of the fitted values:

```
eq1.fit yhat yhat_se
```

To obtain dynamic forecasts, follow the name of the estimated equation, a period, the command `forecast`, a name for the forecasts, and optionally a name for the standard errors of the forecasts:

```
eq1.forecast yh yh_se
```

See the *Command and Programming Reference* for a complete list of commands and options available for forecasting.

Chapter 15. Specification and Diagnostic Tests

Empirical research is usually an interactive process. The process begins with a specification of the relationship to be estimated. Selecting a specification usually involves several choices: the variables to be included, the functional form connecting these variables, and if the data are time series, the dynamic structure of the relationship between the variables.

Inevitably, there is uncertainty regarding the appropriateness of this initial specification. Once you estimate your equation, EViews provides tools for evaluating the quality of your specification along a number of dimensions. In turn, the results of these tests influence the chosen specification, and the process is repeated.

This chapter describes the extensive menu of specification test statistics that are available as views or procedures of an equation object. While we attempt to provide you with sufficient statistical background to conduct the tests, practical considerations ensure that many of the descriptions are incomplete. We refer you to standard statistical and econometric references for further details.

Background

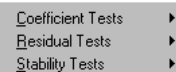
Each test procedure described below involves the specification of a null hypothesis, which is the hypothesis under test. Output from a test command consists of the sample values of one or more test statistics and their associated probability numbers (p -values). The latter indicate the probability of obtaining a test statistic whose absolute value is greater than or equal to that of the sample statistic if the null hypothesis is true. Thus, low p -values lead to the rejection of the null hypothesis. For example, if a p -value lies between 0.05 and 0.01, the null hypothesis is rejected at the 5 percent but not at the 1 percent level.

Bear in mind that there are different assumptions and distributional results associated with each test. For example, some of the test statistics have exact, finite sample distributions (usually t or F -distributions). Others are large sample test statistics with asymptotic χ^2 distributions. Details vary from one test to another and are given below in the description of each test.

Types of Tests

The **View** button on the equation toolbar gives you a choice among three categories of tests to check the specification of the equation.

Additional tests are discussed elsewhere in the *User's Guide*. These tests include unit root tests ([“Performing Unit Root Tests in EViews” on page 329](#)), the Granger causality test ([“Granger Causality” on](#)



page 222), tests specific to binary, order, censored, and count models (Chapter 17, “Discrete and Limited Dependent Variable Models”, on page 421), and the Johansen test for cointegration (“How to Perform a Cointegration Test” on page 538).

Coefficient Tests

These tests evaluate restrictions on the estimated coefficients, including the special case of tests for omitted and redundant variables.

Wald Test (Coefficient Restrictions)

Wald - Coefficient Restrictions...
Omitted Variables - Likelihood Ratio...
Redundant Variables - Likelihood Ratio...

The Wald test computes a test statistic based on the unrestricted regression. The Wald statistic measures how close the unrestricted estimates come to satisfying the restrictions under the null hypothesis. If the restrictions are in fact true, then the unrestricted estimates should come close to satisfying the restrictions.

How to Perform Wald Coefficient Tests

To demonstrate the calculation of Wald tests in EViews, we consider simple examples. Suppose a Cobb-Douglas production function has been estimated in the form:

$$\log Q = A + \alpha \log L + \beta \log K + \epsilon, \quad (15.1)$$

where Q , K and L denote value-added output and the inputs of capital and labor respectively. The hypothesis of constant returns to scale is then tested by the restriction: $\alpha + \beta = 1$.

Estimation of the Cobb-Douglas production function using annual data from 1947 to 1971 provided the following result:

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-2.327939	0.410601	-5.669595	0.0000
LOG(L)	1.591175	0.167740	9.485970	0.0000
LOG(K)	0.239604	0.105390	2.273498	0.0331
R-squared	0.983672	Mean dependent var		4.767586
Adjusted R-squared	0.982187	S.D. dependent var		0.326086
S.E. of regression	0.043521	Akaike info criterion		-3.318997
Sum squared resid	0.041669	Schwarz criterion		-3.172732
Log likelihood	44.48746	F-statistic		662.6819
Durbin-Watson stat	0.637300	Prob(F-statistic)		0.000000

Dependent Variable: LOG(Q)
Method: Least Squares
Date: 08/11/97 Time: 16:56
Sample: 1947 1971
Included observations: 25

The sum of the coefficients on LOG(L) and LOG(K) appears to be in excess of one, but to determine whether the difference is statistically relevant, we will conduct the hypothesis test of constant returns.

To carry out a Wald test, choose **View/Coefficient Tests/Wald-Coefficient Restrictions...** from the equation toolbar. Enter the restrictions into the edit box, with multiple coefficient restrictions separated by commas. The restrictions should be expressed as equations involving the estimated coefficients and constants (you may not include series names). The coefficients should be referred to as C(1), C(2), and so on, unless you have used a different coefficient vector in estimation.

To test the hypothesis of constant returns to scale, type the following restriction in the dialog box:

$$c(2) + c(3) = 1$$

and click **OK**. EViews reports the following result of the Wald test:

Wald Test:			
Equation: EQ1			
Test Statistic	Value	df	Probability
Chi-square	120.0177	1	0.0000
F-statistic	120.0177	(1, 22)	0.0000
Null Hypothesis Summary:			
Normalized Restriction (= 0)	Value	Std. Err.	
-1 + C(2) + C(3)	0.830779	0.075834	

Restrictions are linear in coefficients.

EViews reports an F -statistic and a Chi-square statistic with associated p -values. See “[Wald Test Details](#)” on page 371 for a discussion of these statistics. In addition, EViews reports the value of the normalized (homogeneous) restriction and an associated standard error. In this example, we have a single linear restriction so the two test statistics are identical, with the p -value indicating that we can decisively reject the null hypothesis of constant returns to scale.

To test more than one restriction, separate the restrictions by commas. For example, to test the hypothesis that the elasticity of output with respect to labor is 2/3 and the elasticity with respect to capital is 1/3, enter the restrictions as

$$c(2) = 2/3, \quad c(3) = 1/3$$

and EViews reports

Wald Test:

Equation: EQ1

Test Statistic	Value	df	Probability
Chi-square	53.99105	2	0.0000
F-statistic	26.99553	(2, 22)	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-2/3 + C(2)	0.924508	0.167740
-1/3 + C(1)	-2.661272	0.410601

Restrictions are linear in coefficients.

Note that in addition to the test statistic summary, we report the values of both of the normalized restrictions, along with their standard errors (the square roots of the diagonal elements of the restriction covariance matrix).

As an example of a nonlinear model with a nonlinear restriction, we estimate a production function of the form

$$\log Q = \beta_1 + \beta_2 \log(\beta_3 K^{\beta_4} + (1 - \beta_3)L^{\beta_4}) + \epsilon \quad (15.2)$$

and test the constant elasticity of substitution (CES) production function restriction $\beta_2 = 1/\beta_4$. This is an example of a nonlinear restriction. To estimate the (unrestricted) nonlinear model, you should select **Quick/Estimate Equation...** and then enter the following specification:

$$\log(q) = c(1) + c(2) * \log(c(3) * k^{c(4)} + (1 - c(3)) * l^{c(4)})$$

To test the nonlinear restriction, choose **View/Coefficient Tests/Wald-Coefficient Restrictions...** from the equation toolbar and type the following restriction in the Wald Test dialog box:

$$c(2) = 1/c(4)$$

The results are presented below:

Wald Test:

Equation: EQ2

Test Statistic	Value	df	Probability
Chi-square	0.028508	1	0.8659
F-statistic	0.028508	(1, 21)	0.8675

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(2) - 1/C(4)	1.292163	7.653088

Delta method computed using analytic derivatives.

Since this is a nonlinear equation, we focus on the Chi-square statistic which fails to reject the null hypothesis. Note that EViews reports that it used the delta method (with analytic derivatives) to compute the Wald restriction variance for the nonlinear restriction.

It is well-known that nonlinear Wald tests are not invariant to the way that you specify the nonlinear restrictions. In this example, the nonlinear restriction $\beta_2 = 1/\beta_4$ may equivalently be written as $\beta_2\beta_4 = 1$ or $\beta_4 = 1/\beta_2$ (for nonzero β_2 and β_4). For example, entering the restriction as

$$c(2) * c(4) = 1$$

yields:

Wald Test: Equation: EQ2			
Test Statistic	Value	df	Probability
Chi-square	104.5599	1	0.0000
F-statistic	104.5599	(1, 21)	0.0000

Null Hypothesis Summary:			
Normalized Restriction (= 0)	Value	Std. Err.	
-1 + C(2)*C(4)	0.835330	0.081691	

Delta method computed using analytic derivatives.

so that the test now decisively rejects the null hypothesis. We hasten to add that type of inconsistency is not unique to EViews, but is a more general property of the Wald test. Unfortunately, there does not seem to be a general solution to this problem (see Davidson and MacKinnon, 1993, Chapter 13).

Wald Test Details

Consider a general nonlinear regression model

$$y = f(\beta) + \epsilon \quad (15.3)$$

where y and ϵ are T -vectors and β is a k -vector of parameters to be estimated. Any restrictions on the parameters can be written as

$$H_0: g(\beta) = 0, \quad (15.4)$$

where g is a smooth function, $g: R^k \rightarrow R^q$, imposing q restrictions on β . The Wald statistic is then computed as

$$W = g(\beta)' \left(\frac{\partial g(\beta)}{\partial \beta} \hat{V}(b) \frac{\partial g(\beta)}{\partial \beta'} \right) g(\beta) |_{\beta=b} \quad (15.5)$$

where T is the number of observations and b is the vector of unrestricted parameter estimates, and where \hat{V} is an estimate of the b covariance. In the standard regression case, \hat{V} is given by

$$\hat{V}(b) = s^2 \left(\frac{\partial f(\beta)}{\partial \beta} \frac{\partial f(\beta)}{\partial \beta'} \right)^{-1} \Big|_{\beta=b} \quad (15.6)$$

where u is the vector of unrestricted residuals, and s^2 is the usual estimator of the unrestricted residual variance, $s^2 = (u'u)/(N-k)$, but the estimator of V may differ. For example, \hat{V} may be a robust variance matrix estimator computing using White or Newey-West techniques.

More formally, under the null hypothesis H_0 , the Wald statistic has an asymptotic $\chi^2(q)$ distribution, where q is the number of restrictions under H_0 .

For the textbook case of a linear regression model

$$y = X\beta + \epsilon \quad (15.7)$$

and linear restrictions

$$H_0: R\beta - r = 0, \quad (15.8)$$

where R is a known $q \times k$ matrix, and r is a q -vector, respectively. The Wald statistic in Equation (15.5) reduces to

$$W = (Rb - r)' (Rs^2(X'X)^{-1}R')^{-1} (Rb - r), \quad (15.9)$$

which is asymptotically distributed as $\chi^2(q)$ under H_0 .

If we further assume that the errors ϵ are independent and identically normally distributed, we have an exact, finite sample F -statistic:

$$F = \frac{W}{q} = \frac{(\tilde{u}'\tilde{u} - u'u)/q}{(u'u)/(T-k)}, \quad (15.10)$$

where \tilde{u} is the vector of residuals from the restricted regression. In this case, the F -statistic compares the residual sum of squares computed with and without the restrictions imposed.

We remind you that the expression for the finite sample F -statistic in (15.10) is for standard linear regression, and is not valid for more general cases (nonlinear models, ARMA specifications, or equations where the variances are estimated using other methods such as Newey-West or White). In non-standard settings, the reported F -statistic (which EViews always computes as W/q), does not possess the desired finite-sample properties. In these cases, while asymptotically valid, the F -statistic results should be viewed as illustrative and for comparison purposes only.

Omitted Variables

This test enables you to add a set of variables to an existing equation and to ask whether the set makes a significant contribution to explaining the variation in the dependent variable. The null hypothesis H_0 is that the additional set of regressors are not jointly significant.

The output from the test is an F -statistic and a likelihood ratio (LR) statistic with associated p -values, together with the estimation results of the unrestricted model under the alternative. The F -statistic is based on the difference between the residual sums of squares of the restricted and unrestricted regressions and is only valid in linear regression based settings. The LR statistic is computed as

$$LR = -2(l_r - l_u) \quad (15.11)$$

where l_r and l_u are the maximized values of the (Gaussian) log likelihood function of the unrestricted and restricted regressions, respectively. Under H_0 , the LR statistic has an asymptotic χ^2 distribution with degrees of freedom equal to the number of restrictions (the number of added variables).

Bear in mind that:

- The omitted variables test requires that the same number of observations exist in the original and test equations. If any of the series to be added contain missing observations over the sample of the original equation (which will often be the case when you add lagged variables), the test statistics cannot be constructed.
- The omitted variables test can be applied to equations estimated with linear LS, TSLS, ARCH (mean equation only), binary, ordered, censored, truncated, and count models. The test is available only if you specify the equation by listing the regressors, not by a formula.

To perform an LR test in these settings, you can estimate a separate equation for the unrestricted and restricted models over a common sample, and evaluate the LR statistic and p -value using scalars and the `@cchisq` function, as described above.

How to Perform an Omitted Variables Test

To test for omitted variables, select **View/Coefficient Tests/Omitted Variables-Likelihood Ratio...** In the dialog that opens, list the names of the test variables, each separated by at least one space. Suppose, for example, that the initial regression is

```
ls log(q) c log(l) log(k)
```

If you enter the list

`log (m) log (e)`

in the dialog, then EViews reports the results of the unrestricted regression containing the two additional explanatory variables, and displays statistics testing the hypothesis that the coefficients on the new variables are jointly zero. The top part of the output depicts the test results:

Omitted Variables: LOG(M) LOG(E)			
F-statistic	4.267478	Probability	0.028611
Log likelihood ratio	8.884940	Probability	0.011767

The F -statistic has an exact finite sample F -distribution under H_0 for linear models if the errors are independent and identically distributed normal random variables. The numerator degrees of freedom is the number of additional regressors and the denominator degrees of freedom is the number of observations less the total number of regressors. The log likelihood ratio statistic is the LR test statistic and is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of added regressors.

In our example, the tests reject the null hypothesis that the two series do not belong to the equation at a 5% significance level, but cannot reject the hypothesis at a 1% significance level.

Redundant Variables

The redundant variables test allows you to test for the statistical significance of a subset of your included variables. More formally, the test is for whether a subset of variables in an equation all have zero coefficients and might thus be deleted from the equation. The redundant variables test can be applied to equations estimated by linear LS, TSLS, ARCH (mean equation only), binary, ordered, censored, truncated, and count methods. The test is available only if you specify the equation by listing the regressors, not by a formula.

How to Perform a Redundant Variables Test

To test for redundant variables, select **View/Coefficient Tests/Redundant Variables-Likelihood Ratio...** In the dialog that appears, list the names of each of the test variables, separated by at least one space. Suppose, for example, that the initial regression is

`ls log (q) c log (l) log (k) log (m) log (e)`

If you type the list

`log (m) log (e)`

in the dialog, then EViews reports the results of the restricted regression dropping the two regressors, followed by the statistics associated with the test of the hypothesis that the coefficients on the two variables are jointly zero.

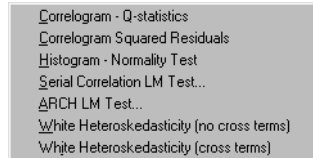
The test statistics are the F -statistic and the Log likelihood ratio. The F -statistic has an exact finite sample F -distribution under H_0 if the errors are independent and identically distributed normal random variables and the model is linear. The numerator degrees of freedom are given by the number of coefficient restrictions in the null hypothesis. The denominator degrees of freedom are given by the total regression degrees of freedom. The LR test is an asymptotic test, distributed as a χ^2 with degrees of freedom equal to the number of excluded variables under H_0 . In this case, there are two degrees of freedom.

Residual Tests

EViews provides tests for serial correlation, normality, heteroskedasticity, and autoregressive conditional heteroskedasticity in the residuals from your estimated equation. Not all of these tests are available for every specification.

Correlograms and Q -statistics

This view displays the autocorrelations and partial autocorrelations of the equation residuals up to the specified number of lags. Further details on these statistics and the Ljung-Box Q -statistics that are also computed are provided in [Chapter 7](#), “ Q -Statistics” on page 169.



This view is available for the residuals from least squares, two-stage least squares, nonlinear least squares and binary, ordered, censored, and count models. In calculating the probability values for the Q -statistics, the degrees of freedom are adjusted to account for estimated ARMA terms.

To display the correlograms and Q -statistics, push **View/Residual Tests/Correlogram- Q -statistics** on the equation toolbar. In the Lag Specification dialog box, specify the number of lags you wish to use in computing the correlogram.

Correlograms of Squared Residuals

This view displays the autocorrelations and partial autocorrelations of the squared residuals up to any specified number of lags and computes the Ljung-Box Q -statistics for the corresponding lags. The correlograms of the squared residuals can be used to check autoregressive conditional heteroskedasticity (ARCH) in the residuals; see also “[ARCH LM Test](#)” on page 377, below.

If there is no ARCH in the residuals, the autocorrelations and partial autocorrelations should be zero at all lags and the Q -statistics should not be significant; see [Chapter 7](#), page 167, for a discussion of the correlograms and Q -statistics.

This view is available for equations estimated by least squares, two-stage least squares, and nonlinear least squares estimation. In calculating the probability for Q -statistics, the degrees of freedom are adjusted for the inclusion of ARMA terms.

To display the correlograms and Q -statistics of the squared residuals, push **View/Residual Tests/Correlogram Squared Residuals** on the equation toolbar. In the Lag Specification dialog box that opens, specify the number of lags over which to compute the correlograms.

Histogram and Normality Test

This view displays a histogram and descriptive statistics of the residuals, including the Jarque-Bera statistic for testing normality. If the residuals are normally distributed, the histogram should be bell-shaped and the Jarque-Bera statistic should not be significant; see [Chapter 7, page 153](#), for a discussion of the Jarque-Bera test. This view is available for residuals from least squares, two-stage least squares, nonlinear least squares, and binary, ordered, censored, and count models.

To display the histogram and Jarque-Bera statistic, select **View/Residual Tests/Histogram-Normality**. The Jarque-Bera statistic has a χ^2 distribution with two degrees of freedom under the null hypothesis of normally distributed errors.

Serial Correlation LM Test

This test is an alternative to the Q -statistics for testing serial correlation. The test belongs to the class of asymptotic (large sample) tests known as Lagrange multiplier (LM) tests.

Unlike the Durbin-Watson statistic for AR(1) errors, the LM test may be used to test for higher order ARMA errors and is applicable whether or not there are lagged dependent variables. Therefore, we recommend its use (in preference to the DW statistic) whenever you are concerned with the possibility that your errors exhibit autocorrelation.

The null hypothesis of the LM test is that there is no serial correlation up to lag order p , where p is a pre-specified integer. The local alternative is ARMA(r, q) errors, where the number of lag terms $p = \max(r, q)$. Note that this alternative includes both AR(p) and MA(p) error processes, so that the test may have power against a variety of alternative autocorrelation structures. See Godfrey (1988), for further discussion.

The test statistic is computed by an auxiliary regression as follows. First, suppose you have estimated the regression

$$y_t = X_t\beta + \epsilon_t \quad (15.12)$$

where b are the estimated coefficients and ϵ are the errors. The test statistic for lag order p is based on the auxiliary regression for the residuals $e = y - X\hat{\beta}$:

$$e_t = X_t\gamma + \left(\sum_{s=1}^p \alpha_s e_{t-s} \right) + v_t. \quad (15.13)$$

Following the suggestion by Davidson and MacKinnon (1993), EViews sets any presample values of the residuals to 0. This approach does not affect the asymptotic distribution of the statistic, and Davidson and MacKinnon argue that doing so provides a test statistic which has better finite sample properties than an approach which drops the initial observations.

This is a regression of the residuals on the original regressors X and lagged residuals up to order p . EViews reports two test statistics from this test regression. The F -statistic is an omitted variable test for the joint significance of all lagged residuals. Because the omitted variables are residuals and not independent variables, the exact finite sample distribution of the F -statistic under H_0 is still not known, but we present the F -statistic for comparison purposes.

The Obs*R-squared statistic is the Breusch-Godfrey LM test statistic. This LM statistic is computed as the number of observations, times the (uncentered) R^2 from the test regression. Under quite general conditions, the LM test statistic is asymptotically distributed as a $\chi^2(p)$.

The serial correlation LM test is available for residuals from either least squares or two-stage least squares estimation. The original regression may include AR and MA terms, in which case the test regression will be modified to take account of the ARMA terms. Testing in 2SLS settings involves additional complications, see Wooldridge (1990) for details.

To carry out the test, push **View/Residual Tests/Serial Correlation LM Test...** on the equation toolbar and specify the highest order of the AR or MA process that might describe the serial correlation. If the test indicates serial correlation in the residuals, LS standard errors are invalid and should not be used for inference.

ARCH LM Test

This is a Lagrange multiplier (LM) test for autoregressive conditional heteroskedasticity (ARCH) in the residuals (Engle 1982). This particular specification of heteroskedasticity was motivated by the observation that in many financial time series, the magnitude of residuals appeared to be related to the magnitude of recent residuals. ARCH in itself does not invalidate standard LS inference. However, ignoring ARCH effects may result in loss of efficiency; see [Chapter 16](#) for a discussion of estimation of ARCH models in EViews.

The ARCH LM test statistic is computed from an auxiliary test regression. To test the null hypothesis that there is no ARCH up to order q in the residuals, we run the regression

$$e_t^2 = \beta_0 + \left(\sum_{s=1}^q \beta_s e_{t-s}^2 \right) + v_t, \quad (15.14)$$

where e is the residual. This is a regression of the squared residuals on a constant and lagged squared residuals up to order q . EViews reports two test statistics from this test regression. The F -statistic is an omitted variable test for the joint significance of all lagged squared residuals. The Obs*R-squared statistic is Engle's LM test statistic, computed as the number of observations times the R^2 from the test regression. The exact finite sample distribution of the F -statistic under H_0 is not known but the LM test statistic is asymptotically distributed $\chi^2(q)$ under quite general conditions. The ARCH LM test is available for equations estimated by least squares, two-stage least squares, and nonlinear least squares.

To carry out the test, push **View/Residual Tests/ARCH LM Test...** on the equation toolbar and specify the order of ARCH to be tested against.

White's Heteroskedasticity Test

This is a test for heteroskedasticity in the residuals from a least squares regression (White, 1980). Ordinary least squares estimates are consistent in the presence heteroskedasticity, but the conventional computed standard errors are no longer valid. If you find evidence of heteroskedasticity, you should either choose the robust standard errors option to correct the standard errors (see "[Heteroskedasticity Consistent Covariances \(White\)](#)" on page 281) or you should model the heteroskedasticity to obtain more efficient estimates using weighted least squares.

White's test is a test of the null hypothesis of no heteroskedasticity against heteroskedasticity of some unknown general form. The test statistic is computed by an auxiliary regression, where we regress the squared residuals on all possible (nonredundant) cross products of the regressors. For example, suppose we estimated the following regression:

$$y_t = b_1 + b_2 x_t + b_3 z_t + e_t \quad (15.15)$$

where the b are the estimated parameters and e the residual. The test statistic is then based on the auxiliary regression:

$$e_t^2 = \alpha_0 + \alpha_1 x_t + \alpha_2 z_t + \alpha_3 x_t^2 + \alpha_4 z_t^2 + \alpha_5 x_t z_t + v_t. \quad (15.16)$$

EViews reports two test statistics from the test regression. The F -statistic is an omitted variable test for the joint significance of all cross products, excluding the constant. It is presented for comparison purposes.

The Obs*R-squared statistic is White's test statistic, computed as the number of observations times the centered R^2 from the test regression. The exact finite sample distribution of the F -statistic under H_0 is not known, but White's test statistic is asymptotically dis-

tributed as a χ^2 with degrees of freedom equal to the number of slope coefficients (excluding the constant) in the test regression.

White also describes this approach as a general test for model misspecification, since the null hypothesis underlying the test assumes that the errors are both homoskedastic and independent of the regressors, and that the linear specification of the model is correct. Failure of any one of these conditions could lead to a significant test statistic. Conversely, a non-significant test statistic implies that none of the three conditions is violated.

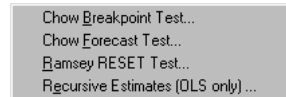
When there are redundant cross-products, EViews automatically drops them from the test regression. For example, the square of a dummy variable is the dummy variable itself, so that EViews drops the squared term to avoid perfect collinearity.

To carry out White's heteroskedasticity test, select **View/Residual Tests/White Heteroskedasticity**. EViews has two options for the test: cross terms and no cross terms. The cross terms version of the test is the original version of White's test that includes all of the cross product terms (in the example above, $x_t z_t$). However, with many right-hand side variables in the regression, the number of possible cross product terms becomes very large so that it may not be practical to include all of them. The no cross terms option runs the test regression using only squares of the regressors.

Specification and Stability Tests

EViews provides a number of test statistic views that examine whether the parameters of your model are stable across various subsamples of your data.

One recommended empirical technique is to split the T observations in your data set of observations into T_1 observations to be used for estimation, and $T_2 = T - T_1$ observations to be used for testing and evaluation. Using all available sample observations for estimation promotes a search for a specification that best fits that specific data set, but does not allow for testing predictions of the model against data that have not been used in estimating the model. Nor does it allow one to test for parameter constancy, stability and robustness of the estimated relationship. In time series work you will usually take the first T_1 observations for estimation and the last T_2 for testing. With cross section data you may wish to order the data by some variable, such as household income, sales of a firm, or other indicator variables and use a sub-set for testing.



There are no hard and fast rules for determining the relative sizes of T_1 and T_2 . In some cases there may be obvious points at which a break in structure might have taken place—a war, a piece of legislation, a switch from fixed to floating exchange rates, or an oil shock. Where there is no reason *a priori* to expect a structural break, a commonly used rule-of-thumb is to use 85 to 90 percent of the observations for estimation and the remainder for testing.

EViews provides built-in procedures which facilitate variations on this type of analysis.

Chow's Breakpoint Test

The idea of the breakpoint Chow test is to fit the equation separately for each subsample and to see whether there are significant differences in the estimated equations. A significant difference indicates a structural change in the relationship. For example, you can use this test to examine whether the demand function for energy was the same before and after the oil shock. The test may be used with least squares and two-stage least squares regressions.

To carry out the test, we partition the data into two or more subsamples. Each subsample must contain more observations than the number of coefficients in the equation so that the equation can be estimated. The Chow breakpoint test compares the sum of squared residuals obtained by fitting a single equation to the entire sample with the sum of squared residuals obtained when separate equations are fit to each subsample of the data.

EViews reports two test statistics for the Chow breakpoint test. The F -statistic is based on the comparison of the restricted and unrestricted sum of squared residuals and in the simplest case involving a single breakpoint, is computed as

$$F = \frac{(\tilde{u}'\tilde{u} - (u_1'u_1 + u_2'u_2))/k}{(u_1'u_1 + u_2'u_2)/(T - 2k)}, \quad (15.17)$$

where $\tilde{u}'\tilde{u}$ is the restricted sum of squared residuals, $u_i'u_i$ is the sum of squared residuals from subsample i , T is the total number of observations, and k is the number of parameters in the equation. This formula can be generalized naturally to more than one breakpoint. The F -statistic has an exact finite sample F -distribution if the errors are independent and identically distributed normal random variables.

The log likelihood ratio statistic is based on the comparison of the restricted and unrestricted maximum of the (Gaussian) log likelihood function. The LR test statistic has an asymptotic χ^2 distribution with degrees of freedom equal to $(m - 1)k$ under the null hypothesis of no structural change, where m is the number of subsamples.

One major drawback of the breakpoint test is that each subsample requires at least as many observations as the number of estimated parameters. This may be a problem if, for example, you want to test for structural change between wartime and peacetime where there are only a few observations in the wartime sample. The Chow forecast test, discussed below, should be used in such cases.

To apply the Chow breakpoint test, push **View/Stability Tests/Chow Breakpoint Test...** on the equation toolbar. In the dialog that appears, list the dates or observation numbers for the breakpoints. For example, if your original equation was estimated from 1950 to 1994, entering

1960

in the dialog specifies two subsamples, one from 1950 to 1959 and one from 1960 to 1994.
Typing

1960 1970

specifies three subsamples, 1950 to 1959, 1960 to 1969, and 1970 to 1994.

Chow's Forecast Test

The Chow forecast test estimates two models—one using the full set of data T , and the other using a long subperiod T_1 . A long difference between the two models casts doubt on the stability of the estimated relation over the sample period. The Chow forecast test can be used with least squares and two-stage least squares regressions.

EViews reports two test statistics for the Chow forecast test. The F -statistic is computed as

$$F = \frac{(\tilde{u}'\tilde{u} - u'u)/T_2}{u'u/(T_1 - k)}, \quad (15.18)$$

where $\tilde{u}'\tilde{u}$ is the residual sum of squares when the equation is fitted to all T sample observations, $u'u$ is the residual sum of squares when the equation is fitted to T_1 observations, and k is the number of estimated coefficients. This F -statistic follows an exact finite sample F -distribution if the errors are independent, and identically, normally distributed.

The log likelihood ratio statistic is based on the comparison of the restricted and unrestricted maximum of the (Gaussian) log likelihood function. Both the restricted and unrestricted log likelihood are obtained by estimating the regression using the whole sample. The restricted regression uses the original set of regressors, while the unrestricted regression adds a dummy variable for each forecast point. The LR test statistic has an asymptotic χ^2 distribution with degrees of freedom equal to the number of forecast points T_2 under the null hypothesis of no structural change.

To apply Chow's forecast test, push **View/Stability Tests/Chow Forecast Test...** on the equation toolbar and specify the date or observation number for the beginning of the forecasting sample. The date should be within the current sample of observations.

As an example, suppose we estimate a consumption function using quarterly data from 1947:1 to 1994:4 and specify 1973:1 as the first observation in the forecast period. The test reestimates the equation for the period 1947:1 to 1972:4, and uses the result to compute the prediction errors for the remaining quarters, and reports the following results:

Chow Forecast Test: Forecast from 1973:1 to 1994:4			
F-statistic	0.708348	Probability	0.951073
Log likelihood ratio	91.57088	Probability	0.376108

Neither of the forecast test statistics reject the null hypothesis of no structural change in the consumption function before and after 1973:1.

If we test the same hypothesis using the Chow breakpoint test, the result is

Chow Breakpoint Test: 1973:1			
F-statistic	38.39198	Probability	0.000000
Log likelihood ratio	65.75468	Probability	0.000000

Note that both of the breakpoint test statistics decisively reject the hypothesis from above. This example illustrates the possibility that the two Chow tests may yield conflicting results.

Ramsey's RESET Test

RESET stands for *Regression Specification Error Test* and was proposed by Ramsey (1969). The classical normal linear regression model is specified as

$$y = X\beta + \epsilon, \quad (15.19)$$

where the disturbance vector ϵ is presumed to follow the multivariate normal distribution $N(0, \sigma^2 I)$. Specification error is an omnibus term which covers any departure from the assumptions of the maintained model. Serial correlation, heteroskedasticity, or non-normality of all violate the assumption that the disturbances are distributed $N(0, \sigma^2 I)$. Tests for these specification errors have been described above. In contrast, RESET is a general test for the following types of specification errors:

- Omitted variables; X does not include all relevant variables.
- Incorrect functional form; some or all of the variables in y and X should be transformed to logs, powers, reciprocals, or in some other way.
- Correlation between X and ϵ , which may be caused, among other things, by measurement error in X , simultaneity, or the presence of lagged y values and serially correlated disturbances.

Under such specification errors, LS estimators will be biased and inconsistent, and conventional inference procedures will be invalidated. Ramsey (1969) showed that any or all of these specification errors produce a non-zero mean vector for ϵ . Therefore, the null and alternative hypotheses of the RESET test are

$$\begin{aligned}
 H_0: \epsilon &\sim N(0, \sigma^2 I) \\
 H_1: \epsilon &\sim N(\mu, \sigma^2 I) \quad \mu \neq 0
 \end{aligned}
 \tag{15.20}$$

The test is based on an augmented regression

$$y = X\beta + Z\gamma + \epsilon. \tag{15.21}$$

The test of specification error evaluates the restriction $\gamma = 0$. The crucial question in constructing the test is to determine what variables should enter the Z matrix. Note that the Z matrix may, for example, be comprised of variables that are not in the original specification, so that the test of $\gamma = 0$ is simply the omitted variables test described above.

In testing for incorrect functional form, the nonlinear part of the regression model may be some function of the regressors included in X . For example, if a linear relation

$$y = \beta_0 + \beta_1 X + \epsilon, \tag{15.22}$$

is specified instead of the true relation

$$y = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon \tag{15.23}$$

the augmented model has $Z = X^2$ and we are back to the omitted variable case. A more general example might be the specification of an additive relation

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon \tag{15.24}$$

instead of the (true) multiplicative relation

$$y = \beta_0 X_1^{\beta_1} X_2^{\beta_2} + \epsilon. \tag{15.25}$$

A Taylor series approximation of the multiplicative relation would yield an expression involving powers and cross-products of the explanatory variables. Ramsey's suggestion is to include powers of the predicted values of the dependent variable (which are, of course, linear combinations of powers and cross-product terms of the explanatory variables) in Z :

$$Z = [\hat{y}^2, \hat{y}^3, \hat{y}^4, \dots] \tag{15.26}$$

where \hat{y} is the vector of fitted values from the regression of y on X . The superscripts indicate the powers to which these predictions are raised. The first power is not included since it is perfectly collinear with the X matrix.

Output from the test reports the test regression and the F -statistic and log likelihood ratio for testing the hypothesis that the coefficients on the powers of fitted values are all zero. A study by Ramsey and Alexander (1984) showed that the RESET test could detect specification error in an equation which was known *a priori* to be misspecified but which nonetheless gave satisfactory values for all the more traditional test criteria—goodness of fit, test for first order serial correlation, high t -ratios.

To apply the test, select **View/Stability Tests/Ramsey RESET Test...** and specify the number of fitted terms to include in the test regression. The fitted terms are the powers of the fitted values from the original regression, starting with the square or second power. For example, if you specify 1, then the test will add \hat{y}^2 in the regression and if you specify 2, then the test will add \hat{y}^2 and \hat{y}^3 in the regression and so on. If you specify a large number of fitted terms, EViews may report a near singular matrix error message since the powers of the fitted values are likely to be highly collinear. The Ramsey RESET test is applicable only to an equation estimated by least squares.

Recursive Least Squares

In recursive least squares the equation is estimated repeatedly, using ever larger subsets of the sample data. If there are k coefficients to be estimated in the b vector, then the first k observations are used to form the first estimate of b . The next observation is then added to the data set and $k + 1$ observations are used to compute the second estimate of b . This process is repeated until all the T sample points have been used, yielding $T - k + 1$ estimates of the b vector. At each step the last estimate of b can be used to predict the next value of the dependent variable. The one-step ahead forecast error resulting from this prediction, suitably scaled, is defined to be a *recursive residual*.

More formally, let X_{t-1} denote the $(t - 1) \times k$ matrix of the regressors from period 1 to period $t - 1$, and y_{t-1} the corresponding vector of observations on the dependent variable. These data up to period $t - 1$ give an estimated coefficient vector, denoted by b_{t-1} . This coefficient vector gives you a forecast of the dependent variable in period t . The forecast is $x_t' b$, where x_t' is the row vector of observations on the regressors in period t . The forecast error is $y_t - x_t' b$, and the forecast variance is given by:

$$\sigma^2(1 + x_t'(X_t'X_t)^{-1}x_t). \quad (15.27)$$

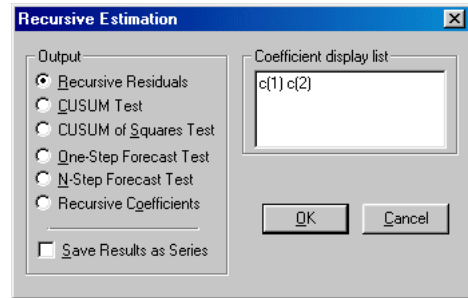
The recursive residual w_t is defined in EViews as

$$w_t = \frac{(y_t - x_t'b)}{(1 + x_t'(X_t'X_t)^{-1}x_t)^{1/2}}. \quad (15.28)$$

These residuals can be computed for $t = k + 1, \dots, T$. If the maintained model is valid, the recursive residuals will be independently and normally distributed with zero mean and constant variance σ^2 .

To calculate the recursive residuals, press **View/Stability Tests/Recursive Estimates (OLS only)...** on the equation toolbar.

There are six options available for the recursive estimates view. The recursive estimates view is only available for equations estimated by ordinary least squares without AR and MA terms. The **Save Results as Series** option allows you to save the recursive residuals and recursive coefficients as named series in the workfile; see “[Save Results as Series](#)” on page 388.



Recursive Residuals

This option shows a plot of the recursive residuals about the zero line. Plus and minus two standard errors are also shown at each point. Residuals outside the standard error bands suggest instability in the parameters of the equation.

CUSUM Test

The CUSUM test (Brown, Durbin, and Evans, 1975) is based on the cumulative sum of the recursive residuals. This option plots the cumulative sum together with the 5% critical lines. The test finds parameter instability if the cumulative sum goes outside the area between the two critical lines.

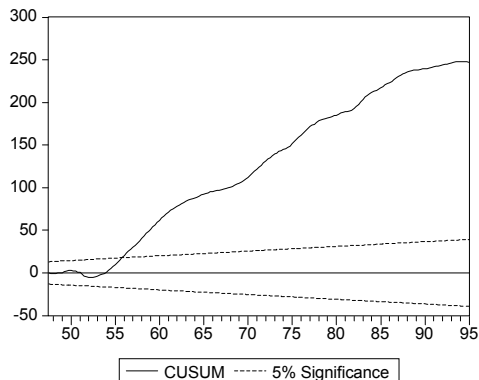
The CUSUM test is based on the statistic

$$W_t = \sum_{r=k+1}^t w_r / s, \quad (15.29)$$

for $t = k + 1, \dots, T$, where w is the recursive residual defined above, and s is the standard error of the regression fitted to all T sample points. If the β vector remains constant from period to period, $E(W_t) = 0$, but if β changes, W_t will tend to diverge from the zero mean value line. The significance of any departure from the zero line is assessed by reference to a pair of 5% significance lines, the distance between which increases with t . The 5% significance lines are found by connecting the points

$$[k, \pm 0.948(T-k)^{1/2}] \quad \text{and} \quad [T, \pm 3 \times 0.948(T-k)^{1/2}]. \quad (15.30)$$

Movement of W_t outside the critical lines is suggestive of coefficient instability. A sample CUSUM test is given below.



The test clearly indicates instability in the equation during the sample period.

CUSUM of Squares Test

The CUSUM of squares test (Brown, Durbin, and Evans, 1975) is based on the test statistic

$$W_t = \left(\sum_{r=k+1}^t w_r^2 \right) / \left(\sum_{r=k+1}^T w_r^2 \right). \quad (15.31)$$

The expected value of S under the hypothesis of parameter constancy is

$$E(S_t) = (t - k) / (T - k) \quad (15.32)$$

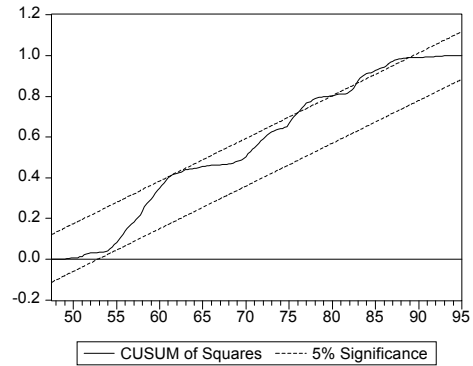
which goes from zero at $t = k$ to unity at $t = T$. The significance of the departure of S from its expected value is assessed by reference to a pair of parallel straight lines around the expected value. See Brown, Durbin, and Evans (1975) or Johnston and DiNardo (1997, Table D.8) for a table of significance lines for the CUSUM of squares test.

The CUSUM of squares test provides a plot of S_t against t and the pair of 5 percent critical lines. As with the CUSUM test, movement outside the critical lines is suggestive of parameter or variance instability.

The cumulative sum of squares is generally within the 5% significance lines, suggesting that the residual variance is somewhat stable.

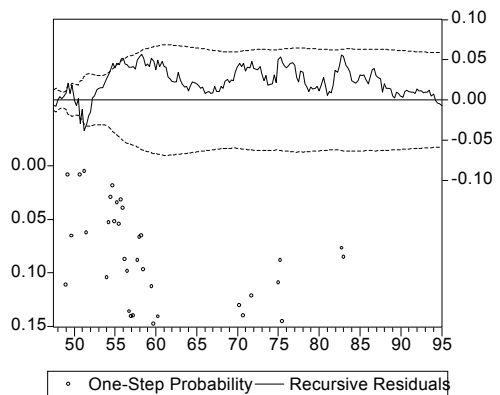
One-Step Forecast Test

If you look back at the definition of the recursive residuals given above, you will see that each recursive residual is the error in a one-step ahead forecast. To test whether the value of the dependent variable at time t might have come from the model fitted to all the data up to that point, each error can be compared with its standard deviation from the full sample.



The **One-Step Forecast Test** option produces a plot of the recursive residuals and standard errors and the sample points whose probability value is at or below 15 percent. The plot can help you spot the periods when your equation is least successful. For example, the one-step ahead forecast test might look like this:

The upper portion of the plot (right vertical axis) repeats the recursive residuals and standard errors displayed by the **Recursive Residuals** option. The lower portion of the plot (left vertical axis) shows the probability values for those sample points where the hypothesis of parameter constancy would be rejected at the 5, 10, or 15 percent levels. The points with p -values less than 0.05 correspond to those points where the recursive residuals go outside the two standard error bounds.



For the test equation, there is evidence of instability early in the sample period.

N-Step Forecast Test

This test uses the recursive calculations to carry out a sequence of Chow Forecast tests. In contrast to the single Chow Forecast test described earlier, this test does not require the specification of a forecast period—it automatically computes all feasible cases, starting with the smallest possible sample size for estimating the forecasting equation and then adding one observation at a time. The plot from this test shows the recursive residuals at

the top and significant probabilities (based on the F -statistic) in the lower portion of the diagram.

Recursive Coefficient Estimates

This view enables you to trace the evolution of estimates for any coefficient as more and more of the sample data are used in the estimation. The view will provide a plot of selected coefficients in the equation for all feasible recursive estimations. Also shown are the two standard error bands around the estimated coefficients.

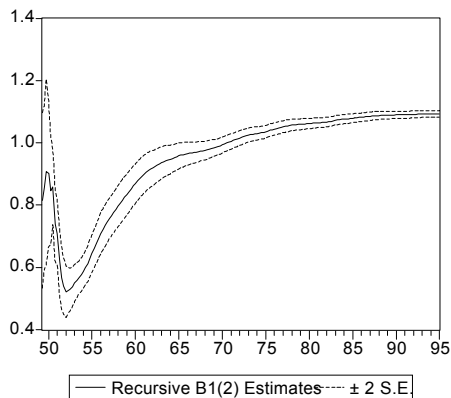
If the coefficient displays significant variation as more data is added to the estimating equation, it is a strong indication of instability. Coefficient plots will sometimes show dramatic jumps as the postulated equation tries to digest a structural break.

To view the recursive coefficient estimates, click the **Recursive Coefficients** option and list the coefficients you want to plot in the **Coefficient Display List** field of the dialog box. The recursive estimates of the marginal propensity to consume (coefficient C(2)), from the sample consumption function are provided below:

The estimated propensity to consume rises steadily as we add more data over the sample period, approaching a value of one.

Save Results as Series

The **Save Results as Series** checkbox will do different things depending on the plot you have asked to be displayed. When paired with the **Recursive Coefficients** option, **Save Results as Series** will instruct EViews to save all recursive coefficients and their standard errors in the workfile as named series. EViews will name the coefficients using the next available name of the form, R_C1, R_C2, ..., and the corresponding standard errors as R_C1SE, R_C2SE, and so on.



If you check the **Save Results as Series** box with any of the other options, EViews saves the recursive residuals and the recursive standard errors as named series in the workfile. EViews will name the residual and standard errors as R_RES and R_RESSE, respectively.

Note that you can use the recursive residuals to reconstruct the CUSUM and CUSUM of squares series.

Applications

In this section, we show how to carry out other specification tests in EViews. For brevity, the discussion is based on commands, but most of these procedures can also be carried out using the menu system.

A Wald test of structural change with unequal variance

The F -statistics reported in the Chow tests have an F -distribution only if the errors are independent and identically normally distributed. This restriction implies that the residual variance in the two subsamples must be equal.

Suppose now that we wish to compute a Wald statistic for structural change with unequal subsample variances. Denote the parameter estimates and their covariance matrix in subsample i as b_i and V_i for $i = 1, 2$. Under the assumption that b_1 and b_2 are independent normal, the difference $b_1 - b_2$ has mean zero and variance $V_1 + V_2$. Therefore, a Wald statistic for the null hypothesis of no structural change and independent samples can be constructed as

$$W = (b_1 - b_2)'(V_1 + V_2)^{-1}(b_1 - b_2), \quad (15.33)$$

which has an asymptotic χ^2 distribution with degrees of freedom equal to the number of estimated parameters in the b vector.

To carry out this test in EViews, we estimate the model in each subsample and save the estimated coefficients and their covariance matrix. For example, suppose we have a quarterly sample of 1947:1–1994:4 and wish to test whether there was a structural change in the consumption function in 1973:1. First, estimate the model in the first sample and save the results by the commands

```
coef(2) b1
smp1 1947:1 1972:4
equation eq_1.ls log(cs)=b1(1)+b1(2)*log(gdp)
sym v1=eq_1.@cov
```

The first line declares the coefficient vector, B1, into which we will place the coefficient estimates in the first sample. Note that the equation specification in the third line explicitly refers to elements of this coefficient vector. The last line saves the coefficient covariance matrix as a symmetric matrix named V1. Similarly, estimate the model in the second sample and save the results by the commands

```
coef(2) b2
smp1 1973.1 1994.4
equation eq_2.ls log(cs)=b2(1)+b2(2)*log(gdp)
sym v2=eq_2.@cov
```

To compute the Wald statistic, use the command

```
matrix wald=@transpose(b1-b2)*@inverse(v1+v2)*(b1-b2)
```

The Wald statistic is saved in the 1×1 matrix named WALD. To see the value, either double click on WALD or type “show wald”. You can compare this value with the critical values from the χ^2 distribution with 2 degrees of freedom. Alternatively, you can compute the p -value in EViews using the command

```
scalar wald_p=1-@cchisq(wald(1,1),2)
```

The p -value is saved as a scalar named WALD_P. To see the p -value, double click on WALD_P or type “show wald_p”. The p -value will be displayed in the status line at the bottom of the EViews window.

The Hausman test

A widely used class of tests in econometrics is the Hausman test. The underlying idea of the Hausman test is to compare two sets of estimates, one of which is consistent under both the null and the alternative and another which is consistent only under the null hypothesis. A large difference between the two sets of estimates is taken as evidence in favor of the alternative hypothesis.

Hausman (1978) originally proposed a test statistic for endogeneity based upon a direct comparison of coefficient values. Here we illustrate the version of the Hausman test proposed by Davidson and MacKinnon (1989, 1993), which carries out the test by running an auxiliary regression.

The following equation was estimated by OLS:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 08/13/97 Time: 14:12
Sample(adjusted): 1959:02 1995:04
Included observations: 435 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.022699	0.004443	-5.108528	0.0000
LOG(IP)	0.011630	0.002585	4.499708	0.0000
DLOG(PPI)	-0.024886	0.042754	-0.582071	0.5608
TB3	-0.000366	9.91E-05	-3.692675	0.0003
LOG(M1(-1))	0.996578	0.001210	823.4440	0.0000
R-squared	0.999953	Mean dependent var		5.844581
Adjusted R-squared	0.999953	S.D. dependent var		0.670596
S.E. of regression	0.004601	Akaike info criterion		-7.913714
Sum squared resid	0.009102	Schwarz criterion		-7.866871
Log likelihood	1726.233	F-statistic		2304897.
Durbin-Watson stat	1.265920	Prob(F-statistic)		0.000000

Suppose we are concerned that industrial production (IP) is endogenously determined with money (M1) through the money supply function. If this were the case, then OLS estimates will be biased and inconsistent. To test this hypothesis, we need to find a set of instrumental variables that are correlated with the “suspect” variable IP but not with the error term of the money demand equation. The choice of the appropriate instrument is a crucial step. Here we take the unemployment rate (URATE) and Moody’s AAA corporate bond yield (AAA) as instruments.

To carry out the Hausman test by artificial regression, we run two OLS regressions. In the first regression, we regress the suspect variable (log) IP on all exogenous variables and instruments and retrieve the residuals:

```
ls log(ip) c dlog(ppi) tb3 log(m1(-1)) urate aaa
series res_ip=resid
```

Then in the second regression, we re-estimate the money demand function including the residuals from the first regression as additional regressors. The result is:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 08/13/97 Time: 15:28
Sample(adjusted): 1959:02 1995:04
Included observations: 435 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.007145	0.007473	-0.956158	0.3395
LOG(IP)	0.001560	0.004672	0.333832	0.7387
DLOG(PPI)	0.020233	0.045935	0.440465	0.6598
TB3	-0.000185	0.000121	-1.527775	0.1273
LOG(M1(-1))	1.001093	0.002123	471.4894	0.0000
RES_IP	0.014428	0.005593	2.579826	0.0102
R-squared	0.999954	Mean dependent var		5.844581
Adjusted R-squared	0.999954	S.D. dependent var		0.670596
S.E. of regression	0.004571	Akaike info criterion		-7.924512
Sum squared resid	0.008963	Schwarz criterion		-7.868300
Log likelihood	1729.581	F-statistic		1868171.
Durbin-Watson stat	1.307838	Prob(F-statistic)		0.000000

If the OLS estimates are consistent, then the coefficient on the first stage residuals should not be significantly different from zero. In this example, the test (marginally) rejects the hypothesis of consistent OLS estimates (to be more precise, this is an asymptotic test and you should compare the t -statistic with the critical values from the standard normal).

Non-nested Tests

Most of the tests discussed in this chapter are nested tests in which the null hypothesis is obtained as a special case of the alternative hypothesis. Now consider the problem of choosing between the following two specifications of a consumption function:

$$\begin{aligned}
 H_1: \quad CS_t &= \alpha_1 + \alpha_2 GDP_t + \alpha_3 GDP_{t-1} + \epsilon_t \\
 H_2: \quad CS_t &= \beta_1 + \beta_2 GDP_t + \beta_3 CS_{t-1} + \epsilon_t
 \end{aligned}
 \tag{15.34}$$

These are examples of non-nested models since neither model may be expressed as a restricted version of the other.

The J -test proposed by Davidson and MacKinnon (1993) provides one method of choosing between two non-nested models. The idea is that if one model is the correct model, then the fitted values from the other model should not have explanatory power when estimating that model. For example, to test model H_1 against model H_2 , we first estimate model H_2 and retrieve the fitted values:

```
equation eq_cs2.ls cs c gdp cs(-1)
eq_cs2.fit cs2
```

The second line saves the fitted values as a series named CS2. Then estimate model H_1 including the fitted values from model H_2 . The result is:

Dependent Variable: CS				
Method: Least Squares				
Date: 8/13/97 Time: 00:49				
Sample(adjusted): 1947:2 1994:4				
Included observations: 191 after adjusting endpoints				
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	7.313232	4.391305	1.665389	0.0975
GDP	0.278749	0.029278	9.520694	0.0000
GDP(-1)	-0.314540	0.029287	-10.73978	0.0000
CS2	1.048470	0.019684	53.26506	0.0000
R-squared	0.999833	Mean dependent var		1953.966
Adjusted R-squared	0.999830	S.D. dependent var		848.4387
S.E. of regression	11.05357	Akaike info criterion		7.664104
Sum squared resid	22847.93	Schwarz criterion		7.732215
Log likelihood	-727.9219	F-statistic		373074.4
Durbin-Watson stat	2.253186	Prob(F-statistic)		0.000000

The fitted values from model H_2 enter significantly in model H_1 and we reject model H_1 .

We must also test model H_2 against model H_1 . Estimate model H_1 , retrieve the fitted values, and estimate model H_2 including the fitted values from model H_1 . The results of this “reverse” test are given by:

Dependent Variable: CS
 Method: Least Squares
 Date: 08/13/97 Time: 16:58
 Sample(adjusted): 1947:2 1994:4
 Included observations: 191 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1427.716	132.0349	-10.81318	0.0000
GDP	5.170543	0.476803	10.84419	0.0000
CS(-1)	0.977296	0.018348	53.26506	0.0000
CS1	-7.292771	0.679043	-10.73978	0.0000
R-squared	0.999833	Mean dependent var		1953.966
Adjusted R-squared	0.999830	S.D. dependent var		848.4387
S.E. of regression	11.05357	Akaike info criterion		7.664104
Sum squared resid	22847.93	Schwarz criterion		7.732215
Log likelihood	-727.9219	F-statistic		373074.4
Durbin-Watson stat	2.253186	Prob(F-statistic)		0.000000

The fitted values are again statistically significant and we reject model H_2 .

In this example, we reject both specifications, against the alternatives, suggesting that another model for the data is needed. It is also possible that we fail to reject both models, in which case the data do not provide enough information to discriminate between the two models.

Commands

All of the specification and diagnostic tests explained in this chapter are available in command form as views of a named equation. Follow the equation name with a dot and the view name of the test. For example, to carry out the Wald test of whether the third and fourth coefficients of the equation object EQ1 are both equal to zero, type

```
eq1.wald c(3)=0, c(4)=0
```

To carry out the serial correlation LM test of the residuals in equation EQ_Y up to 4 lags, type

```
eq_y.auto(4)
```

To display the recursive residuals of equation EQM1, type

```
eqm1.rls(r)
```

See “Equation” on page 21 of the *Command and Programming Reference* for a complete list of commands and options available for equation objects.

Part IV. Advanced Single Equation Analysis

The following sections describe EViews tools for the estimation and analysis of advanced single equation models.

- [Chapter 16, “ARCH and GARCH Estimation”](#), beginning on page 397, outlines the EViews tools for ARCH and GARCH modeling of the conditional variance, or volatility, of a variable.
- [Chapter 17, “Discrete and Limited Dependent Variable Models”](#), on page 421 documents EViews tools for estimating qualitative and limited dependent variable models. EViews provides estimation routines for binary or ordered (probit, logit, gompit), censored or truncated (tobit, etc.), and integer valued (count data).
- [Chapter 18, “The Log Likelihood \(LogL\) Object”](#), beginning on page 471 describes techniques for using EViews to estimate the parameters of maximum likelihood models where you may specify the form of the likelihood.

Multiple equation models and forecasting are described in [Part V. “Multiple Equation Analysis”](#) beginning on page 493.

Chapter 16. ARCH and GARCH Estimation

Most of the statistical tools in EViews are designed to model the conditional mean of a random variable. The tools described in this chapter differ by modeling the conditional variance, or volatility, of a variable.

There are several reasons that you may want to model and forecast volatility. First, you may need to analyze the risk of holding an asset or the value of an option. Second, forecast confidence intervals may be time-varying, so that more accurate intervals can be obtained by modeling the variance of the errors. Third, more efficient estimators can be obtained if heteroskedasticity in the errors is handled properly.

Autoregressive Conditional Heteroskedasticity (ARCH) models are specifically designed to model and forecast conditional variances. The variance of the dependent variable is modeled as a function of past values of the dependent variable and independent, or exogenous variables.

ARCH models were introduced by Engle (1982) and generalized as GARCH (Generalized ARCH) by Bollerslev (1986). These models are widely used in various branches of econometrics, especially in financial time series analysis. See Bollerslev, Chou, and Kroner (1992) and Bollerslev, Engle, and Nelson (1994) for recent surveys.

In the next section, the basic ARCH model will be described in detail. In subsequent sections, we consider the wide range of specifications available in EViews for modeling volatility. For brevity of discussion, we will use ARCH to refer to both ARCH and GARCH models, except where there is the possibility of confusion.

The ARCH Specification

In developing an ARCH model, you will have to provide two distinct specifications—one for the conditional mean and one for the conditional variance.

The GARCH(1,1) Model

In the standard GARCH(1,1) specification:

$$y_t = x_t' \gamma + \epsilon_t \quad (16.1)$$

$$\sigma_t^2 = \omega + \alpha \epsilon_{t-1}^2 + \beta \sigma_{t-1}^2 \quad (16.2)$$

the mean equation given in (16.1) is written as a function of exogenous variables with an error term. Since σ_t^2 is the one-period ahead forecast variance based on past information, it is called the *conditional variance*. The conditional variance equation specified in (16.2) is a function of three terms:

- The mean: ω .
- News about volatility from the previous period, measured as the lag of the squared residual from the mean equation: ϵ_{t-1}^2 (the ARCH term).
- Last period's forecast variance: σ_{t-1}^2 (the GARCH term).

The (1,1) in GARCH(1,1) refers to the presence of a first-order GARCH term (the first term in parentheses) and a first-order ARCH term (the second term in parentheses). An ordinary ARCH model is a special case of a GARCH specification in which there are no lagged forecast variances in the conditional variance equation.

ARCH models in EViews are estimated by the method of maximum likelihood under the assumption that the errors are conditionally normally distributed. For example, for the GARCH(1,1) model, the contribution to the log likelihood from observation t is

$$l_t = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log\sigma_t^2 - \frac{1}{2}(y_t - x_t'\gamma)^2 / \sigma_t^2, \quad (16.3)$$

where

$$\sigma_t^2 = \omega + \alpha(y_{t-1} - x_{t-1}'\gamma)^2 + \beta\sigma_{t-1}^2. \quad (16.4)$$

This specification is often interpreted in a financial context, where an agent or trader predicts this period's variance by forming a weighted average of a long term average (the constant), the forecasted variance from last period (the GARCH term), and information about volatility observed in the previous period (the ARCH term). If the asset return was unexpectedly large in either the upward or the downward direction, then the trader will increase the estimate of the variance for the next period. This model is also consistent with the volatility clustering often seen in financial returns data, where large changes in returns are likely to be followed by further large changes.

There are two alternative representations of the variance equation that may aid in the interpretation of the model:

- If we recursively substitute for the lagged variance on the right-hand side of (16.2), we can express the conditional variance as a weighted average of all of the lagged squared residuals:

$$\sigma_t^2 = \frac{\omega}{(1-\beta)} + \alpha \sum_{j=1}^{\infty} \beta^{j-1} \epsilon_{t-j}^2. \quad (16.5)$$

- We see that the GARCH(1,1) variance specification is analogous to the sample variance, but that it down-weights more distant lagged squared errors.

- The error in the squared returns is given by $v_t = \epsilon_t^2 - \sigma_t^2$. Substituting for the variances in the variance equation and rearranging terms we can write our model in terms of the errors:

$$\epsilon_t^2 = \omega + (\alpha + \beta)\epsilon_{t-1}^2 + v_t - \beta v_{t-1}. \quad (16.6)$$

- Thus, the squared errors follow a heteroskedastic ARMA(1,1) process. The autoregressive root which governs the persistence of volatility shocks is the sum of α plus β . In many applied settings, this root is very close to unity so that shocks die out rather slowly.

Regressors in the Variance Equation

Equation (16.2) may be extended to allow for the inclusion of exogenous or predetermined regressors, z , in the variance equation:

$$\sigma_t^2 = \omega + \alpha\epsilon_t^2 + \beta\sigma_{t-1}^2 + \pi z_t. \quad (16.7)$$

Note that the forecasted variances from this model are not guaranteed to be positive. You may wish to introduce regressors in a form where they are always positive to minimize the possibility that a single, large negative value generates a negative forecasted value. For example, you may want to set

$$z_t = \text{abs}(x_t). \quad (16.8)$$

The ARCH-M Model

The x in equation (16.2) represent exogenous or predetermined variables that are included in the mean equation. If we introduce the conditional variance into the mean equation, we get the ARCH-in-Mean (ARCH-M) model (Engle, Lilien and Robins, 1987):

$$y_t = x_t' \gamma + \sigma_t^2 + \epsilon_t. \quad (16.9)$$

The ARCH-M model is often used in financial applications where the expected return on an asset is related to the expected asset risk. The estimated coefficient on the expected risk is a measure of the risk-return tradeoff.

A variant of the ARCH-M specification uses the conditional standard deviation in place of the conditional variance in Equation (16.9).

The GARCH(p, q) Model

Higher order GARCH models, denoted GARCH(p, q), can be estimated by choosing either p or q greater than 1. The representation of the GARCH(p, q) variance is

$$\sigma_t^2 = \omega + \sum_{j=1}^p \beta_j \sigma_{t-j}^2 + \sum_{i=1}^q \alpha_i \epsilon_{t-i}^2 \quad (16.10)$$

where p is the order of the GARCH terms and q is the order of the ARCH term.

Estimating ARCH models in EViews

To estimate an ARCH or GARCH model, open the equation specification dialog by selecting **Quick/Estimate Equation...** or by selecting **Object/New Object/Equation....** Select **ARCH** from the method combo box.

You will need to specify both the mean and the variance equations, as well as the estimation technique and sample.

The Mean Equation

In the dependent variable edit box, you should enter the specification of the mean equation. You can enter the specification in list form by listing the dependent variable followed by the regressors. You should add the C to your specification if you wish to include a constant. If you have a more complex mean specification, you can enter your mean equation using a formula.

If your specification includes an ARCH-M term, you should click on the appropriate radio button in the upper right-hand side of the dialog.

The Variance Equation

ARCH Specification

Under the ARCH Specification label, you should choose the number of ARCH and GARCH terms. The default is to estimate with one ARCH and one GARCH term. This is by far the most popular specification.

To estimate the standard GARCH model as described above, click on the GARCH radio button. The other entries describe more complicated variants of the GARCH specification. We discuss each of these models later in the chapter.

Variance Regressors

In the edit box labeled **Variance Regressors**, you may optionally list variables you wish to include in the variance specification. Note that EViews will always include a constant as a variance regressor so that you do not need to add C to the list.

The distinction between the permanent and transitory regressors is discussed in [“The Component ARCH Model” on page 412](#).

Estimation Options

EViews provides you with access to a number of optional estimation settings. Simply click on the **Options** button and fill out the dialog as required.

Backcasting

By default, both the innovations used in initializing MA estimation and the initial variance required for the GARCH terms are computed using backcasting methods. Details on the MA backcasting procedure are provided in [“Backcasting MA terms” on page 320](#).

When computing backcast initial variances for GARCH, EViews first uses the coefficient values to compute the residuals of the mean equation, and then computes an exponential smoothing estimator of the initial values:

$$\sigma_0^2 = \epsilon_0^2 = \lambda^T \hat{\sigma}^2 + (1 - \lambda) \sum_{j=0}^{T-1} \lambda^{T-j-1} (\hat{\epsilon}_{T-j}^2), \quad (16.11)$$

where $\hat{\epsilon}$ are the residuals from the mean equation, $\hat{\sigma}^2$ is the unconditional variance estimate

$$\hat{\sigma}^2 = \sum_{t=1}^T \hat{\epsilon}_t^2 / T \quad (16.12)$$

and the smoothing parameter $\lambda = 0.7$. Alternatively, you can choose to initialize the GARCH process using the unconditional variance:

$$\sigma_0^2 = \epsilon_0^2 = \hat{\sigma}^2. \quad (16.13)$$

If you turn off backcasting, EViews will set the presample values of the residual to zero to initialize an MA, if present, and will set the presample values of the variance and squared residual to the unconditional variance using [\(16.13\)](#).

Our experience has been that GARCH models initialized using backcast exponential smoothing often outperform models initialized using the unconditional variance.

Heteroskedasticity Consistent Covariances

Click on the check box labeled **Heteroskedasticity Consistent Covariance** to compute the quasi-maximum likelihood (QML) covariances and standard errors using the methods described by Bollerslev and Wooldridge (1992).

You should use this option if you suspect that the residuals are not conditionally normally distributed. When the assumption of conditional normality does not hold, the ARCH parameter estimates will still be consistent, provided the mean and variance functions are

correctly specified. The estimates of the covariance matrix will not be consistent unless this option is specified, resulting in incorrect standard errors.

Note that the parameter estimates will be unchanged if you select this option; only the estimated covariance matrix will be altered.

Derivative Methods

EViews currently uses numeric derivatives in estimating ARCH models. You can control the method used in computing these derivatives to favor speed (fewer function evaluations) or to favor accuracy (more function evaluations).

Iterative Estimation Control

The likelihood functions of ARCH models are not always well-behaved so that convergence may not be achieved with the default estimation settings. You can use the options dialog to select the iterative algorithm (Marquardt, BHHH/Gauss-Newton), change starting values, increase the maximum number of iterations, or adjust the convergence criterion.

Starting Values

As with other iterative procedures, starting coefficient values are required. EViews will supply its own starting values for ARCH procedures using OLS regression for the mean equation. Using the Options dialog, you can also set starting values to various fractions of the OLS starting values, or you can specify the values yourself by choosing the User Specified option, and placing the desired coefficients in the default coefficient vector.

Examples

To estimate a standard GARCH(1,1) model with no regressors in the mean and variance equations,

$$\begin{aligned}R_t &= c + \epsilon_t \\ \sigma_t^2 &= \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2\end{aligned}\tag{16.14}$$

you should enter the various parts of your specification:

- Fill in the Mean Equation Specification edit box as
r c
- Enter 1 for the number of ARCH terms, and 1 for the number of GARCH terms, and select **GARCH (symmetric)**.
- Select **None** for the **ARCH-M term**.
- Leave blank the **Variance Regressors** edit box.

To estimate the ARCH(4)-M model

$$\begin{aligned}
 R_t &= \gamma_0 + \gamma_1 DUM_t + \gamma_2 \sigma_t + \epsilon_t \\
 \sigma_t^2 &= \omega + \alpha_1 \epsilon_{t-1}^2 + \alpha_2 \epsilon_{t-2}^2 + \alpha_3 \epsilon_{t-3}^2 + \alpha_4 \epsilon_{t-4}^2 + \gamma_3 DUM_t
 \end{aligned}
 \tag{16.15}$$

you should fill out the dialog in the following fashion:

- Enter the mean equation specification,
`r c dum`
- Enter “4” for the ARCH term and “0” for the GARCH term, and select **GARCH (symmetric)**.
- Select **Std. Dev.** for the **ARCH-M** term.
- Enter DUM in the Variance Regressors edit box.

Once you have filled in the Equation Specification dialog, click **OK** to estimate the model. ARCH models are estimated by the method of maximum likelihood, under the assumption that the errors are conditionally normally distributed. Because the variance appears in a non-linear way in the likelihood function, the likelihood function must be estimated using iterative algorithms. In the status line, you can watch the value of the likelihood as it changes with each iteration. When estimates converge, the parameter estimates and conventional regression statistics are presented in the ARCH object window.

ARCH Estimation Output

As an example, we fit a GARCH(1,1) model to the first difference of log daily U.S. Dollar/Japanese Yen exchange rates (R) using backcast values for the initial variances and Bollerslev-Wooldridge standard errors. The output is presented below.

By default, the estimation output header describes the estimation sample, and the methods used for computing the coefficient standard errors and the initial variance terms.

The main output from ARCH estimation is divided into two sections—the upper part provides the standard output for the mean equation, while the lower part, labeled “Variance Equation” contains the coefficients, standard errors, z -statistics and p -values for the coefficients of the variance equation. The ARCH parameters correspond to α and the GARCH parameters to β in [Equation \(16.2\) on page 397](#). The bottom panel of the output presents the standard set of regression statistics using the residuals from the mean equation. Note that measures such as R^2 may not be meaningful if there are no regressors in the mean equation. Here, for example, the R^2 is negative.

Dependent Variable: R
Method: ML - ARCH
Date: 10/16/00 Time: 10:13
Sample(adjusted): 2 5758
Included observations: 5757 after adjusting endpoints
Convergence achieved after 335 iterations
Bollerslev-Wooldrige robust standard errors & covariance
Variance backcast: ON

	Coefficient	Std. Error	z-Statistic	Prob.
C	-0.005032	0.005324	-0.945073	0.3446
Variance Equation				
C	0.015617	0.008729	1.788987	0.0736
ARCH(1)	0.208221	0.051805	4.019331	0.0001
GARCH(1)	0.785165	0.045302	17.33167	0.0000
R-squared	-0.000632	Mean dependent var		-0.020218
Adjusted R-squared	-0.001154	S.D. dependent var		0.604167
S.E. of regression	0.604515	Akaike info criterion		1.596745
Sum squared resid	2102.368	Schwarz criterion		1.601371
Log likelihood	-4592.231	Durbin-Watson stat		1.914803

In this example, the sum of the ARCH and GARCH coefficients ($\alpha + \beta$) is very close to one, indicating that volatility shocks are quite persistent. This result is often observed in high frequency financial data.

Working with ARCH Models

Once your model has been estimated, EViews provides a variety of views and procedures for inference and diagnostic checking.

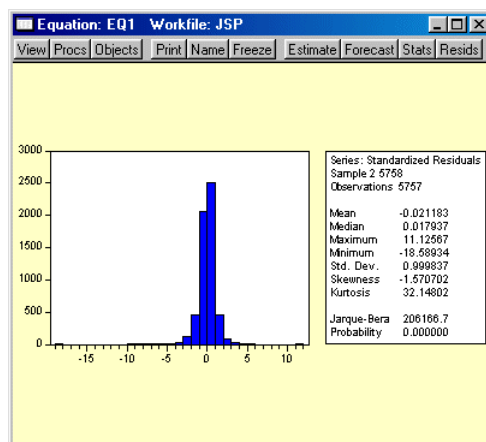
Views of ARCH Models

- **Actual, Fitted, Residual** view displays the residuals in various forms, such as table, graphs, and standardized residuals. You can save the residuals as a named series in your workfile using a procedure (see below).
- **Conditional SD Graph** plots the one-step ahead standard deviation σ_t for each observation in the sample. The observation at period t is the forecast for t made using information available in $t - 1$. You can save the conditional standard deviations as named series in your workfile using a procedure (see below).
- **Covariance Matrix** displays the estimated coefficient covariance matrix. Most ARCH models (except ARCH-M models) are block diagonal so that the covariance between the mean coefficients and the variance coefficients is very close to zero. If you include a constant in the mean equation, there will be two C's in the covariance matrix; the first C is the constant of the mean equation, and the second C is the constant of the variance equation.

- **Coefficient Tests** carries out standard hypothesis tests on the estimated coefficients. See “[Coefficient Tests](#)” on page 368 for details. Note that the likelihood ratio tests are not appropriate under a quasi-maximum likelihood interpretation of your results.
- **Residual Tests/Correlogram–Q-statistics** displays the correlogram (autocorrelations and partial autocorrelations) of the standardized residuals. This view can be used to test for remaining serial correlation in the mean equation and to check the specification of the mean equation. If the mean equation is correctly specified, all Q -statistics should not be significant. See “[Correlogram](#)” on page 167 for an explanation of correlograms and Q -statistics.
- **Residual Tests/Correlogram Squared Residuals** displays the correlogram (autocorrelations and partial autocorrelations) of the squared standardized residuals. This view can be used to test for remaining ARCH in the variance equation and to check the specification of the variance equation. If the variance equation is correctly specified, all Q -statistics should not be significant. See “[Correlogram](#)” on page 167 for an explanation of correlograms and Q -statistics. See also **Residual Tests/ARCH LM Test**.
- **Residual Tests/Histogram–Normality Test** displays descriptive statistics and a histogram of the standardized residuals. You can use the Jarque-Bera statistic to test whether the standardized residuals are normally distributed. If the standardized residuals are normally distributed, the Jarque-Bera statistic should not be significant. See “[Descriptive Statistics](#)” beginning on page 152 for an explanation of the Jarque-Bera test. For example, the histogram of the standardized residuals from the GARCH(1,1) model fit to the daily exchange rate looks as follows:

The residuals are highly leptokurtic and the Jarque-Bera statistic decisively rejects the hypothesis of normal distribution.

- **Residual Tests/ARCH LM Test** carries out Lagrange multiplier tests to test whether the standardized residuals exhibit additional ARCH. If the variance equation is correctly specified, there should be no ARCH left in the standardized residuals. See “[ARCH LM Test](#)” on page 377 for a discussion of testing. See also **Residual Tests/Correlogram Squared Residuals**.



ARCH Model Procedures

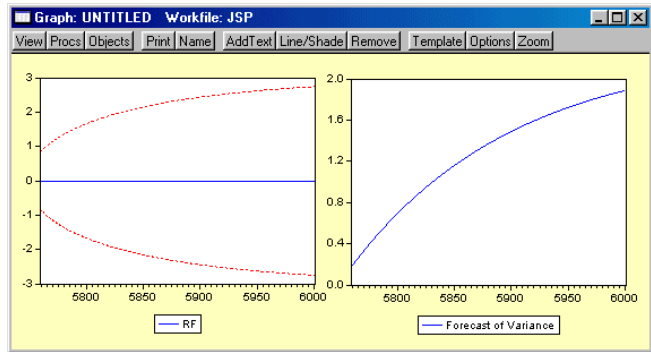
- **Make Residual Series** saves the residuals as named series in your workfile. You have the option to save the ordinary residuals, ϵ_t , or the standardized residuals, ϵ_t/σ_t . The residuals will be named RESID1, RESID2, and so on; you can rename the series with the **name** button in the series window.
- **Make GARCH Variance Series** saves the conditional variances σ_t^2 as named series in your workfile. The conditional variance series will be named GARCH01, GARCH02, and so on. Take the square root to get the conditional standard deviations as displayed by the **View/Conditional SD Graph**.
- **Forecast** uses the estimated ARCH model to compute static and dynamic forecasts of the mean, its forecast standard error, and the conditional variance. To save any of these forecasts in your workfile, type a name in the corresponding dialog box. If you choose the **Do graph** option, EViews displays the graphs of the forecasts and two standard deviation bands for the mean forecast. For example, suppose we estimated the following GARCH(1,1)-M model:

Dependent Variable: R
 Method: ML - ARCH
 Date: 10/16/00 Time: 11:12
 Sample(adjusted): 2 5758
 Included observations: 5757 after adjusting endpoints
 Convergence achieved after 403 iterations
 Bollerslev-Wooldrige robust standard errors & covariance
 Variance backcast: ON

	Coefficient	Std. Error	z-Statistic	Prob.
SQR(GARCH)	-0.021288	0.038087	-0.558935	0.5762
C	0.004083	0.017338	0.235492	0.8138
Variance Equation				
C	0.016429	0.009140	1.797553	0.0722
ARCH(1)	0.214350	0.051698	4.146199	0.0000
GARCH(1)	0.778253	0.046152	16.86295	0.0000
R-squared	0.000081	Mean dependent var		-0.020218
Adjusted R-squared	-0.000615	S.D. dependent var		0.604167
S.E. of regression	0.604352	Akaike info criterion		1.599669
Sum squared resid	2100.870	Schwarz criterion		1.605451
Log likelihood	-4599.646	F-statistic		0.116199
Durbin-Watson stat	1.917673	Prob(F-statistic)		0.976830

To construct dynamic forecasts of R using this model, click **Forecast** and fill in the Forecast dialog setting the sample after the estimation period. If you choose **Do graph**, the equation view changes to display the forecast results.

This graph is the forecast of R from the mean equation together with the two standard deviation bands. Although, the point forecasts look constant, they are in fact declining over the forecast period because of the negative



sign of the GARCH term in the mean equation (you can check this by looking at the spreadsheet view of the point forecasts.) The right graph is the forecast of the conditional variance σ_t^2 . Since the sum of the ARCH and GARCH terms ($\alpha + \beta$) is close to one, the volatility shocks are persistent, so that the forecasts of the conditional variance converge to the steady state quite slowly.

Additional Comments

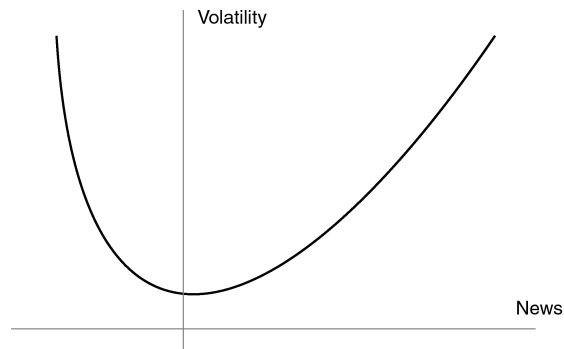
Several of the test results described above are formulated in terms of *standardized residuals*, ϵ_t/σ_t , which are defined as the conventional mean equation residuals divided by the conditional standard deviation.

If the model is correctly specified, the standardized residuals should be independent, and identically distributed random variables with mean zero and variance one. If the standardized residuals are also normally distributed, then the estimates are maximum likelihood estimates which are asymptotically efficient. However, even if the distribution of the residuals is not normal, the estimates are still consistent under quasi-maximum likelihood (QML) assumptions.

To carry out valid inference with QML, you should make certain to use the **Heteroskedasticity Consistent Covariance** option to estimate the standard errors.

Asymmetric ARCH Models

For equities, it is often observed that downward movements in the market are followed by higher volatilities than upward movements of the same magnitude. To account for this phenomenon, Engle and Ng (1993) describe a News Impact Curve with asymmetric response to good and bad news.



EViews estimates two models that allow for asymmetric shocks to volatility: TARCH and EGARCH.

The TARCH Model

TARCH or Threshold ARCH was introduced independently by Zakoïan (1994) and Glosten, Jaganathan, and Runkle (1993). The specification for the conditional variance is given by

$$\sigma_t^2 = \omega + \alpha \epsilon_{t-1}^2 + \gamma \epsilon_{t-1}^2 d_{t-1} + \beta \sigma_{t-1}^2 \quad (16.16)$$

where $d_t = 1$ if $\epsilon_t < 0$, and 0 otherwise.

In this model, good news ($\epsilon_t > 0$), and bad news ($\epsilon_t < 0$), have differential effects on the conditional variance—good news has an impact of α , while bad news has an impact of $(\alpha + \gamma)$. If $\gamma > 0$ we say that a *leverage effect* exists in that bad news increases volatility. If $\gamma \neq 0$, the news impact is asymmetric.

For higher order specifications of the TARCH model, EViews estimates

$$\sigma_t^2 = \omega + \sum_{i=1}^q \alpha_i \epsilon_{t-i}^2 + \gamma \epsilon_{t-1}^2 d_{t-1} + \sum_{j=1}^p \beta_j \sigma_{t-j}^2. \quad (16.17)$$

To estimate this model, specify your ARCH model in the usual fashion, but instead of selecting a GARCH specification, you should click on the **TARCH (asymmetric)** radio button listed under **ARCH Specification**.

The TARCH(1,1) model fitted to the daily exchange rate returns gives:

Method: ML - ARCH
 Date: 10/16/00 Time: 11:25
 Sample(adjusted): 2 5758
 Included observations: 5757 after adjusting endpoints
 Convergence achieved after 438 iterations
 Bollerslev-Wooldrige robust standard errors & covariance
 Variance backcast: ON

	Coefficient	Std. Error	z-Statistic	Prob.
C	-0.005262	0.005778	-0.910734	0.3624
Variance Equation				
C	0.015643	0.008727	1.792541	0.0730
ARCH(1)	0.206231	0.056786	3.631712	0.0003
(RESID<0)*ARCH(1)	0.004252	0.050441	0.084304	0.9328
GARCH(1)	0.784890	0.045006	17.43957	0.0000
R-squared	-0.000613	Mean dependent var		-0.020218
Adjusted R-squared	-0.001309	S.D. dependent var		0.604167
S.E. of regression	0.604562	Akaike info criterion		1.597084
Sum squared resid	2102.328	Schwarz criterion		1.602867
Log likelihood	-4592.206	Durbin-Watson stat		1.914840

The leverage effect term, γ , represented by (RESID < 0)*ARCH(1) in the output, is not significantly positive (even with a one-sided test) so there does not appear to be an asymmetric effect. Note that it is important that we use the quasi-likelihood robust standard errors since the residuals are highly leptokurtic.

Note that when forecasting with this model, EViews assumes that the distribution of the residuals is symmetric so that $d = 1$ half of the time. Since we cannot identify when this occurs, we arbitrarily set $d = 0.5$ for all observations.

The EGARCH Model

The EGARCH or Exponential GARCH model was proposed by Nelson (1991). The specification for the conditional variance is

$$\log \sigma_t^2 = \omega + \beta \log \sigma_{t-1}^2 + \alpha \left| \frac{\epsilon_{t-1}}{\sigma_{t-1}} \right| + \gamma \frac{\epsilon_{t-1}}{\sigma_{t-1}}. \quad (16.18)$$

Note that the left-hand side is the *log* of the conditional variance. This implies that the leverage effect is exponential, rather than quadratic, and that forecasts of the conditional variance are guaranteed to be nonnegative. The presence of leverage effects can be tested by the hypothesis that $\gamma > 0$. The impact is asymmetric if $\gamma \neq 0$.

There are two differences between the EViews specification of the EGARCH model and the original Nelson model. First, Nelson assumes that the ϵ follows a generalized error distribution, while EViews assumes normally distributed errors. Second, Nelson's specification for the log conditional variances differs slightly from the specification above:

$$\log \sigma_t^2 = \omega + \beta \log \sigma_{t-1}^2 + \alpha \left| \frac{\epsilon_{t-1}}{\sigma_{t-1}} - \sqrt{\frac{2}{\pi}} \right| + \gamma \frac{\epsilon_{t-1}}{\sigma_{t-1}}. \quad (16.19)$$

Estimating this model under the assumption of normal errors will yield identical estimates to those reported by EViews except for the intercept term w , which differ by $\alpha\sqrt{2/\pi}$.

$$\log \sigma_t^2 = \omega + \sum_{i=1}^p \beta_i \log \sigma_{t-i}^2 + \sum_{j=1}^q \left(\alpha_j \left| \frac{\epsilon_{t-j}}{\sigma_{t-j}} - \sqrt{\frac{2}{\pi}} \right| + \gamma_j \frac{\epsilon_{t-j}}{\sigma_{t-j}} \right). \quad (16.20)$$

To estimate an EGARCH model, simply select the EGARCH radio button under the ARCH specification settings. Applying this method to data on daily bond futures gives the following results:

Dependent Variable: BF
Method: ML - ARCH
Date: 10/16/00 Time: 11:47
Sample: 1 2114
Included observations: 2114
Convergence achieved after 15 iterations
Bollerslev-Wooldrige robust standard errors & covariance
Variance backcast: ON

	Coefficient	Std. Error	z-Statistic	Prob.
C	8.63E-05	0.000134	0.641819	0.5210
Variance Equation				
C	-0.129288	0.034851	-3.709784	0.0002
RES /SQR[GARCH](1)	0.062530	0.015138	4.130533	0.0000
RES/SQR[GARCH](1)	-0.025617	0.011189	-2.289447	0.0221
EGARCH(1)	0.991846	0.002925	339.0410	0.0000
R-squared	0.000000	Mean dependent var		
Adjusted R-squared	-0.001897	S.D. dependent var		
S.E. of regression	0.007024	Akaike info criterion		
Sum squared resid	0.104050	Schwarz criterion		
Log likelihood	7637.979	Durbin-Watson stat		

The leverage effect term, γ , denoted as RES/SQR[GARCH](1) in the output, is negative and statistically different from zero, indicating the existence of the leverage effect in future bonds returns during the sample period.

Plotting the Estimated News Impact Curve

What does the estimated news impact curve look like? Here we illustrate one method of using EViews to plot this curve. Our goal is to plot the volatility σ_t^2 , against the impact $z = \epsilon/\sigma$, where

$$\log \sigma_t^2 = \hat{\omega} + \hat{\beta} \log \sigma_{t-1}^2 + \hat{\alpha} |z_{t-1}| + \hat{\gamma} z_{t-1}. \quad (16.21)$$

We will fix last period's volatility σ_{t-1}^2 to the median of the estimated conditional variance series and estimate the one-period impact, conditional on last period's volatility.

Suppose you estimated the above EGARCH model in an undated workfile and saved the equation as an object named EQ1. First, generate the conditional variance series by clicking **Procs/Make GARCH Variance Series**. Next, store the median by entering the following command in the command window:

```
scalar med = @median(garch01)
```

where GARCH1 is the name of the conditional variance series.

Generate the z series, which will be the x -axis of the news impact curve, using the following commands:

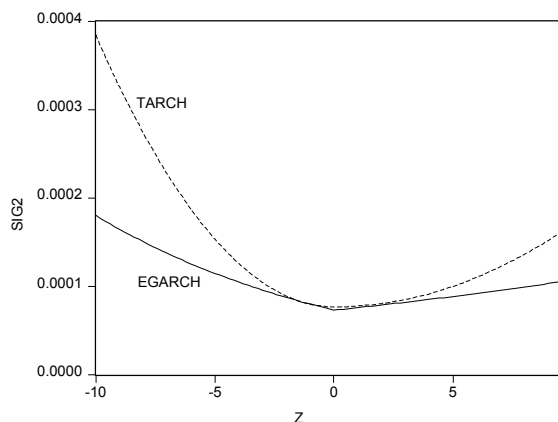
```
sml 1 100
series z = -10 + @trend(1)*20/100
```

This generates an equispaced z series between -10 and 10 . Then generate the σ^2 series by the command

```
series log(sig2) = eq1.c(2) + eq1.c(5)*log(med) +
eq1.c(3)*abs(z) + eq1.c(4)*z
```

where SIG2 is the name for the σ^2 series. Note that EViews will automatically create the series SIG2 from the log specification.

Finally, highlight the two series Z and SIG2 (in order to get the axis right), double click, **Open/as Group**, and **View/Graph/XY line/One X against all Y's**. Below we show a customized graph depicting the estimated news impact curves from TARCH and EGARCH models fitted to the daily futures return:



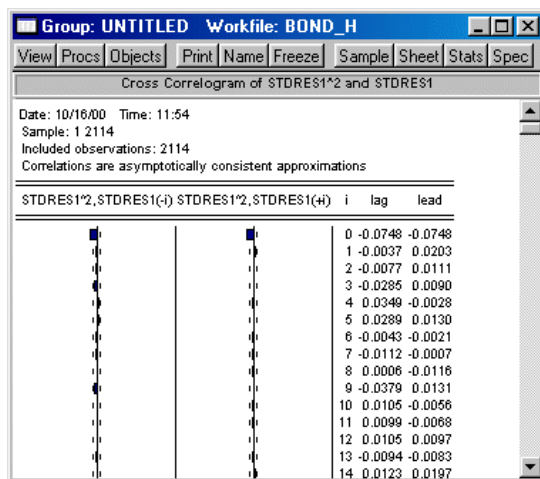
Checking for Asymmetry Using Cross Correlations

Estimating the TARCH and EGARCH models and testing the significance of the asymmetric terms is one way to test for asymmetric effects. Alternatively, we can look at the cross correlation between the squared standardized residuals z_t^2 and lagged standardized residuals z_{t-k} . These cross correlations should be zero for a symmetric GARCH model and negative for a TARCH or EGARCH asymmetric model.

To display the cross correlogram between z_t^2 and z_{t-k} , first save the standardized residuals from the estimated model in a series named STDRES1. Then type the command:

```
show stdres1^2 stdres1
```

where RESID1 is the name of the standardized residuals. Then **View/Cross Correlation (2)** ... and specify the lags to display the cross correlogram. The cross correlogram for the simple GARCH(1,1) model fit to future bonds returns is displayed below.



Because of the large sample size, the approximate two standard error bands (the dotted lines) are very tight around the estimated values. Note that the cross correlation only picks up linear associations between the two series and may miss nonlinear dependence between the two series.

The Component ARCH Model

The conditional variance in the GARCH(1,1) model:

$$\sigma_t^2 = \bar{\omega} + \alpha(\epsilon_{t-1}^2 - \bar{\omega}) + \beta(\sigma_{t-1}^2 - \bar{\omega}). \quad (16.22)$$

shows mean reversion to $\bar{\omega}$ which is a constant for all time. By contrast, the component model allows mean reversion to a varying level q_t , modeled as:

$$\begin{aligned}\sigma_t^2 - q_t &= \bar{\omega} + \alpha(\epsilon_{t-1}^2 - \bar{\omega}) + \beta(\sigma_{t-1}^2 - \bar{\omega}) \\ q_t &= \omega + \rho(q_{t-1} - \omega) + \phi(\epsilon_{t-1}^2 - \sigma_{t-1}^2).\end{aligned}\tag{16.23}$$

Here σ_t^2 is still the volatility, while q_t takes the place of ω and is the time varying long run volatility. The first equation describes the transitory component, $\sigma_t^2 - q_t$, which converges to zero with powers of $(\alpha + \beta)$. The second equation describes the long run component q_t , which converges to ω with powers of ρ . Typically ρ is between 0.99 and 1 so that q_t approaches ω very slowly. We can combine the transitory and permanent equations and write

$$\begin{aligned}\sigma_t^2 &= (1 - \alpha - \beta)(1 - \rho)\omega + (\alpha + \phi)\epsilon_{t-1}^2 - (\alpha\rho + (\alpha + \beta)\phi)\epsilon_{t-2}^2 \\ &\quad + (\beta - \phi)\sigma_{t-1}^2 - (\beta\rho - (\alpha + \beta)\phi)\sigma_{t-2}^2\end{aligned}\tag{16.24}$$

which shows that the component model is a (nonlinear) restricted GARCH(2,2) model.

You can include exogenous variables in the conditional variance equation of component models, either in the permanent or transitory equation (or both). The variables in the transitory equation will have an impact on the short run movements in volatility, while the variables in the permanent equation will affect the long run levels of volatility.

The **Asymmetric Component** option in the Equation Specification dialog combines the component model with the asymmetric TARCH model. This specification introduces asymmetric effects in the transitory equation and estimates models of the form:

$$\begin{aligned}y_t &= x_t' \pi + \epsilon_t \\ q_t &= \omega + \rho(q_{t-1} - \omega) + \phi(\epsilon_{t-1}^2 - \sigma_{t-1}^2) + \theta_1 z_{1t} \\ \sigma_t^2 - q_t &= \alpha(\epsilon_{t-1}^2 - q_{t-1}) + \gamma(\epsilon_{t-1}^2 - q_{t-1})d_{t-1} + \beta(\sigma_{t-1}^2 - q_{t-1}) + \theta_2 z_{2t}\end{aligned}\tag{16.25}$$

where z are the exogenous variables and d is the dummy variable indicating negative shocks. $\gamma > 0$ indicates the presence of transitory leverage effects in the conditional variance.

Estimating and Interpreting Models in EViews

To estimate component models in EViews, choose either the **Component ARCH** or **Asymmetric Component** option in the Equation Specification dialog. The estimation results of fitting the asymmetric component model to the bond futures return data are shown below:

Dependent Variable: BF
 Method: ML - ARCH
 Date: 10/16/00 Time: 12:41
 Sample: 1 2114
 Included observations: 2114
 Convergence achieved after 19 iterations
 Bollerslev-Wooldrige robust standard errors & covariance
 Variance backcast: ON

	Coefficient	Std. Error	z-Statistic	Prob.
C	0.000120	0.000133	0.899130	0.3686
Variance Equation				
Perm: C	2.99E-05	6.66E-06	4.485359	0.0000
Perm: [Q-C]	0.998114	0.001278	781.1750	0.0000
Perm: [ARCH-GARCH]	-0.004590	0.004750	-0.966355	0.3339
Tran: [ARCH-Q]	0.018299	0.012540	1.459306	0.1445
Tran: (RES<0)*[ARCH-Q]	0.030781	0.014738	2.088588	0.0367
Tran: [GARCH-Q]	0.939780	0.021482	43.74633	0.0000
R-squared	-0.000021	Mean dependent var		8.78E-05
Adjusted R-squared	-0.002868	S.D. dependent var		0.007017
S.E. of regression	0.007027	Akaike info criterion		-7.221226
Sum squared resid	0.104052	Schwarz criterion		-7.202497
Log likelihood	7639.836	Durbin-Watson stat		1.934630

The coefficients labeled “Perm:” are the coefficients for the permanent equation and those labeled “Trans:” correspond to the transitory equation. The estimate of the persistence in the long run component is $\hat{\rho} = 0.998$, indicating that the long run component converges very slowly to the steady state. The short run volatility component appears to be significantly different from zero. After setting the asymmetric effect to its mean, we can test the joint hypothesis that $(\alpha + 0.5\gamma + \beta = 0)$, by selecting **View/Coefficient Tests/Wald - Coefficient Restrictions...** and entering

$$c(5) + 0.5*c(6) + c(7) = 0$$

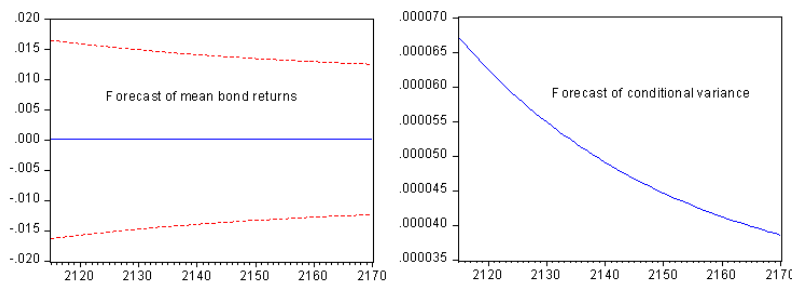
in the Wald Test dialog box. The result is given by:

Wald Test:			
Equation: BF_ACOMPONENT			
Test Statistic	Value	df	Probability
F-statistic	4613.622	(1, 2107)	0.0000
Chi-square	4613.622	1	0.0000

Null Hypothesis Summary:		
Normalized Restriction (= 0)	Value	Std. Err.
C(5) + 0.5*C(6) + C(7)	0.973469	69.77481

Restrictions are linear in coefficients.

To forecast from the component model, click **Forecast**, enter “2115 2170” as the forecast interval, and make certain that **Do Graph**, and is selected. Click **OK**. Here we have copy-and-pasted the customized graph output into our word processor.



Notice that the volatility forecast from the component model need not be monotonic. In this example, there is a short run fluctuating component which dies out in the long run.

To include exogenous regressors in the variance equation, type the names of the regressors in the **Variance Regressors** box in the following order: first list the series names to include in the permanent equation, followed by an @ sign, and then list the series names to include in the transitory equation. For example, to include HOL in the permanent equation and JAN, END in the transitory equation, type

```
hol @ jan end
```

and to include JAN in the transitory equation, type

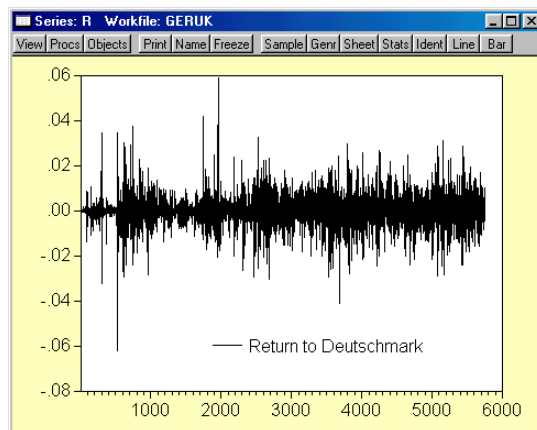
```
@ jan
```

Examples

Modeling the volatility in the US Dollar/Deutschmark exchange rates

As an illustration of ARCH modeling in EViews, we use daily US Dollar/Deutschmark exchange rate from 1971 to 1993. The dependent variable is the daily nominal return, $r_t = \log(s_t/s_{t-1})$, where s is the spot rate. The return series clearly shows volatility clustering, especially early in the sample.

We start from a general GARCH(1,1)-M model of the following form:



$$\begin{aligned}
 r_t &= \pi_0 + \pi_1 \sigma_t^2 + \epsilon_t + \theta \epsilon_{t-1} \\
 \sigma_t^2 &= \omega + \alpha \epsilon_{t-1}^2 + \beta \sigma_{t-1}^2 + \gamma \text{MON}_t.
 \end{aligned}
 \tag{16.26}$$

The mean equation includes the conditional variance and MA(1) errors. The MON series is a dummy variable for Monday, which is meant to capture weekend non-trading.

Dependent Variable: R
 Method: ML - ARCH
 Date: 10/16/00 Time: 13:34
 Sample(adjusted): 2 5764
 Included observations: 5763 after adjusting endpoints
 Estimation settings: tol= 1.0E-06, derivs=fast mixed (linear)
 Initial Values: C(1)=-2.18649, C(2)=-3.2E-05, C(3)=0.00500, C(4)=4.4E-05, C(5)=0.15000, C(6)=0.60000, C(7)=0.00000
 Convergence achieved after 251 iterations
 Bollerslev-Wooldridge robust standard errors & covariance
 MA backcast: 1, Variance backcast: ON

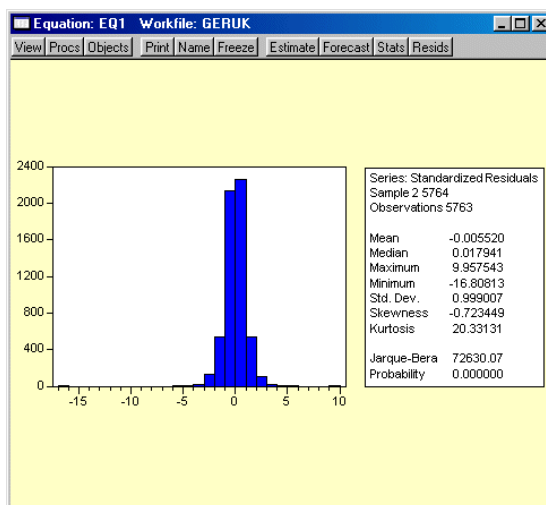
	Coefficient	Std. Error	z-Statistic	Prob.
GARCH	-0.860947	2.763612	-0.311530	0.7554
C	-7.31E-05	0.000103	-0.706830	0.4797
MA(1)	0.020325	0.015719	1.293035	0.1960
Variance Equation				
C	1.86E-06	3.02E-07	6.144539	0.0000
ARCH(1)	0.121845	0.022167	5.496754	0.0000
GARCH(1)	0.865308	0.014842	58.29979	0.0000
MON	-4.37E-06	6.28E-07	-6.956371	0.0000
R-squared	0.001056	Mean dependent var	-0.000128	
Adjusted R-squared	0.000015	S.D. dependent var	0.006628	
S.E. of regression	0.006628	Akaike info criterion	-7.404132	
Sum squared resid	0.252876	Schwarz criterion	-7.396043	
Log likelihood	21342.01	F-statistic	1.014057	
Durbin-Watson stat	1.973046	Prob(F-statistic)	0.413930	
Inverted MA Roots	-.02			

To check whether there are any ARCH effects left in the residuals, click **View/Residual Tests/ARCH LM Test...** and specify the order to test. The top part of the output from testing up to an ARCH(7) is given by

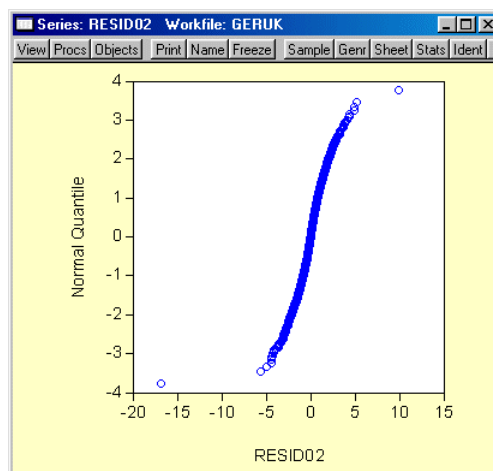
ARCH Test:			
F-statistic	0.125147	Probability	0.996585
Obs*R-squared	0.877118	Probability	0.996575

indicating that there does not appear to be any ARCH up to order 7.

To check whether the distribution of the standardized residuals look normal, click **View/Residual Tests/Histogram:**



The residuals are highly leptokurtic and the Jarque-Bera test decisively rejects the normal distribution. An alternative way to check the distribution of the residuals is to plot the quantiles. First save the residuals by clicking **Procs/Make Residual Series...** and choose **standardized**. In our example, EViews created a series named RESID02 containing the standardized residuals. Then in the residual series window choose **View/Distribution/Quantile-Quantile Graphs...** and plot against **Quantiles of the normal distribution**.

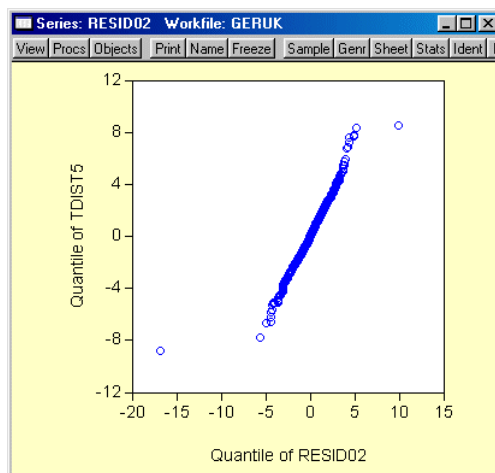


If the residuals are normally distributed, the QQ-plots should lie on a straight line; see [“Quantile-Quantile” on page 227](#) for details on QQ-plots. The plot shows that it is primarily a few large outliers that are driving the departure from normality. You can also experiment with other distributions. For example, in the ARCH literature, the t -distribution is com-

monly used to capture the thick tails in the distribution. Although there is no option for the t -distribution in the **Quantile-Quantile** view, you can simulate a draw from a t -distribution and check whether the distribution matches with the residuals. The command

```
series tdist5 = @qtdist(rnd,5)
```

simulates a random draw from the t -distribution with 5 degrees of freedom. Then in the QQ Plot dialog box choose **Series or Group** and type the name of the series (in this case `tdist5`).



The QQ-plot now lies mostly on a straight line, except for the two outlier observations.

The departure from normality of the residuals suggests using the robust standard errors option.

Commands

The ARCH procedure estimates GARCH models:

```
equation eq1.arch(1,1) sp500 c
```

estimates a GARCH(1,1) model with only a constant in the conditional mean equation.

```
eq1.arch(1,1) sp500 c @ d_mon
```

estimates a GARCH(1,1) model with a constant in the conditional mean equation and `D_MON` in the conditional variance equation.

```
eq1.makegarch garch1
```

stores the estimated conditional variance series from `EQ1` as a series named `GARCH1`.

[Chapter 18, “The Log Likelihood \(LogL\) Object”](#), beginning on [page 471](#) contains examples of using logl objects for estimating other univariate and multivariate GARCH models.

See the *Command and Programming Reference* for a complete list of commands and options available in ARCH estimation.

Chapter 17. Discrete and Limited Dependent Variable Models

The regression methods described in [Chapter 11, “Basic Regression”](#) and [Chapter 12, “Additional Regression Methods”](#) require that the dependent variable be observed on a continuous and unrestricted scale. It is quite common, however, for this condition to be violated, resulting in a non-continuous, or a limited dependent variable. We will distinguish between three types of these variables:

- qualitative (observed on a discrete or ordinal scale)
- censored or truncated
- integer valued

In this chapter we discuss estimation methods for several qualitative and limited dependent variable models. EViews provides estimation routines for binary or ordered (probit, logit, gompit), censored or truncated (tobit, etc.), and integer valued (count data) models.

Standard introductory discussion for the models presented in this chapter may be found in Greene (1997), Johnston and DiNardo (1997), and Maddala (1983). Wooldridge (1996) provides an excellent reference for quasi-likelihood methods and count models.

Binary Dependent Variable Models

In this class of models, the dependent variable, y may take on only two values— y might be a dummy variable representing the occurrence of an event, or a choice between two alternatives. For example, you may be interested in modeling the employment status of each individual in your sample (whether employed or not). The individuals differ in age, educational attainment, race, marital status, and other observable characteristics, which we denote as x . The goal is to quantify the relationship between the individual characteristics and the probability of being employed.

Theory

Suppose that a binary dependent variable, y , takes on values of zero and one. A simple linear regression of y on x is not appropriate, since among other things, the implied model of the conditional mean places inappropriate restrictions on the residuals of the model. Furthermore, the fitted value of y from a simple linear regression is not restricted to lie between zero and one.

Instead, we adopt a specification that is designed to handle the specific requirements of binary dependent variables. Suppose that we model the probability of observing a value of one as:

$$\Pr(y_i = 1 | x_i, \beta) = 1 - F(-x_i' \beta), \quad (17.1)$$

where F is a continuous, strictly increasing function that takes a real value and returns a value ranging from zero to one. The choice of the function F determines the type of binary model. It follows that

$$\Pr(y_i = 0 | x_i, \beta) = F(-x_i' \beta). \quad (17.2)$$

Given such a specification, we can estimate the parameters of this model using the method of maximum likelihood. The likelihood function is given by:

$$l(\beta) = \sum_{i=0}^n y_i \log(1 - F(-x_i' \beta)) + (1 - y_i) \log(F(-x_i' \beta)). \quad (17.3)$$

The first order conditions for this likelihood are nonlinear so that obtaining parameter estimates requires an iterative solution. By default, EViews uses a second derivative method for iteration and computation of the covariance matrix of the parameter estimates. As discussed below, EViews allows you to override these defaults using the Options dialog (see “[Second Derivative Methods](#)” on page 664 for additional details on the estimation methods).

There are two alternative interpretations of this specification that are of interest. First, the binary model is often motivated as a latent variables specification. Suppose that there is an unobserved latent variable y_i^* that is linearly related to x

$$y_i^* = x_i' \beta + u_i \quad (17.4)$$

where u_i is a random disturbance. Then the observed dependent variable is determined by whether y_i^* exceeds a threshold value:

$$y_i = \begin{cases} 1 & \text{if } y_i^* > 0 \\ 0 & \text{if } y_i^* \leq 0. \end{cases} \quad (17.5)$$

In this case, the threshold is set to zero, but the choice of a threshold value is irrelevant, so long as a constant term is included in x_i . Then

$$\Pr(y_i = 1 | x_i, \beta) = \Pr(y_i^* > 0) = \Pr(x_i' \beta + u_i > 0) = F_u(-x_i' \beta) \quad (17.6)$$

where F_u is the cumulative distribution function of u . Common models include probit (standard normal), logit (logistic), and gompit (extreme value) specifications for the F function.

In principle, the coding of the two numerical values of y is not critical since each of the binary responses only represents an event. Nevertheless, EViews requires that you code y as a zero-one variable. This restriction yields a number of advantages. For one, coding the

variable in this fashion implies that expected value of y is simply the probability that $y = 1$:

$$\begin{aligned} E(y_i|x_i, \beta) &= 1 \cdot \Pr(y_i = 1|x_i, \beta) + 0 \cdot \Pr(y_i = 0|x_i, \beta) \\ &= \Pr(y_i = 1|x_i, \beta). \end{aligned} \quad (17.7)$$

This convention provides us with a second interpretation of the binary specification: as a conditional mean specification. It follows that we can write the binary model as a regression model:

$$y_i = (1 - F(-x_i'\beta)) + \epsilon_i, \quad (17.8)$$

where ϵ_i is a residual representing the deviation of the binary y_i from its conditional mean. Then

$$\begin{aligned} E(\epsilon_i|x_i, \beta) &= 0 \\ \text{var}(\epsilon_i|x_i, \beta) &= F(-x_i'\beta)(1 - F(-x_i'\beta)). \end{aligned} \quad (17.9)$$

We will use the conditional mean interpretation in our discussion of binary model residuals (see [“Make Residual Series” on page 434](#)).

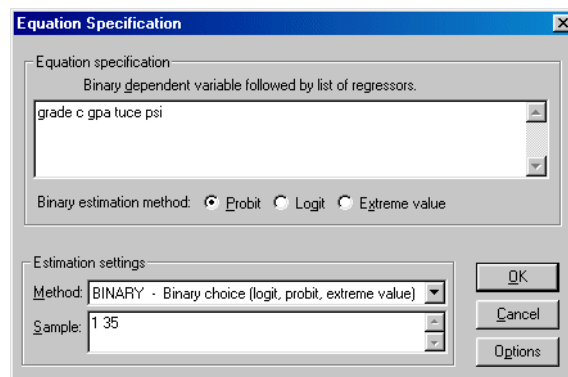
Estimating Binary Models in EViews

To estimate a binary dependent variable model, choose **Objects/New Object...** from the main menu and select the **Equation** object from the main menu. From the Equation Specification dialog, select the **BINARY** estimation method. The dialog will change to reflect your choice.

There are two parts to the binary model specification. First, in the **Equation Specification** field, you should type the name of the binary dependent variable followed by a list of regressors. You may not enter an explicit equation since binary estimation only supports specification by list. Next, select from among the three distributions for your error term:

Probit	$\Pr(y_i = 1 x_i, \beta) = 1 - \Phi(-x_i' \beta) = \Phi(x_i' \beta)$ <p>where Φ is the cumulative distribution function of the standard normal distribution.</p>
Logit	$\Pr(y_i = 1 x_i, \beta) = 1 - (e^{-x_i' \beta} / (1 + e^{-x_i' \beta}))$ $= e^{x_i' \beta} / (1 + e^{x_i' \beta})$ <p>which is based upon the cumulative distribution function for the logistic distribution.</p>
Extreme value (Gompit)	$\Pr(y_i = 1 x_i, \beta) = 1 - (1 - \exp(-e^{-x_i' \beta}))$ $= \exp(-e^{-x_i' \beta})$ <p>which is based upon the CDF for the Type-I extreme value distribution. Note that this distribution is skewed.</p>

For example, consider the probit specification example described in Greene (1997, p. 876) where we analyze the effectiveness of teaching methods on grades. The variable GRADE represents improvement on grades following exposure to the new teaching method PSI. Also controlling for alternative measures of knowledge (GPA and TUCE), we have the specification:



Once you have specified the model, click **OK**. EViews estimates the parameters of the model using iterative procedures, and will display information in the status line. EViews requires that the dependent variable be coded with the values zero-one with all other observations dropped from the estimation.

Following estimation, EViews displays results in the equation window. The top part of the estimation output is given by:

```

Dependent Variable: GRADE
Method: ML - Binary Probit
Date: 07/31/00 Time: 15:57
Sample: 1 32
Included observations: 32
Convergence achieved after 5 iterations
Covariance matrix computed using second derivatives
    
```

Variable	Coefficient	Std. Error	z-Statistic	Prob.
C	-7.452320	2.542472	-2.931131	0.0034
GPA	1.625810	0.693882	2.343063	0.0191
TUCE	0.051729	0.083890	0.616626	0.5375
PSI	1.426332	0.595038	2.397045	0.0165

The header contains basic information regarding the estimation technique (ML for maximum likelihood) and the sample used in estimation, as well as information on the number of iterations required for convergence, and on the method used to compute the coefficient covariance matrix.

Displayed next are the coefficient estimates, asymptotic standard errors, z -statistics and corresponding p -values.

Interpretation of the coefficient values is complicated by the fact that estimated coefficients from a binary model cannot be interpreted as the marginal effect on the dependent variable. The marginal effect of x_j on the conditional probability is given by:

$$\frac{\partial \mathbf{E}(y_i | x_i, \beta)}{\partial x_{ij}} = f(-x_i' \beta) \beta_j, \quad (17.10)$$

where $f(x) = dF(x)/dx$ is the density function corresponding to F . Note that β_j is weighted by a factor f that depends on the values of *all* of the regressors in x . The *direction* of the effect of a change in x_j depends only on the sign of the β_j coefficient. Positive values of β_j imply that increasing x_j will increase the probability of the response; negative values imply the opposite.

While marginal effects calculation is not provided as a built-in view or procedure, in “[Forecast](#)” on page 434, we show you how to use EViews to compute the marginal effects.

An alternative interpretation of the coefficients results from noting that the ratios of coefficients provide a measure of the relative changes in the probabilities:

$$\frac{\beta_j}{\beta_k} = \frac{\partial \mathbf{E}(y_i | x_i, \beta) / \partial x_{ij}}{\partial \mathbf{E}(y_i | x_i, \beta) / \partial x_{ik}}. \quad (17.11)$$

In addition to the summary statistics of the dependent variable, EViews also presents the following summary statistics:

Mean dependent var	0.343750	S.D. dependent var	0.482559
S.E. of regression	0.386128	Akaike info criterion	1.051175
Sum squared resid	4.174660	Schwarz criterion	1.234392
Log likelihood	-12.81880	Hannan-Quinn criter.	1.111907
Restr. log likelihood	-20.59173	Avg. log likelihood	-0.400588
LR statistic (3 df)	15.54585	McFadden R-squared	0.377478
Probability(LR stat)	0.001405		
Obs with Dep=0	21	Total obs	32
Obs with Dep=1	11		

First, there are several familiar summary descriptive statistics: the mean and standard deviation of the dependent variable, standard error of the regression, and the sum of the squared residuals. The latter two measures are computed in the usual fashion using the residuals:

$$e_i = y_i - E(y_i | x_i, \hat{\beta}) = y_i - (1 - F(-x_i' \hat{\beta})). \quad (17.12)$$

Additionally, there are several likelihood based statistics:

- **Log likelihood** is the maximized value of the log likelihood function $l(\hat{\beta})$.
- **Avg. log likelihood** is the log likelihood $l(\hat{\beta})$ divided by the number of observations n .
- **Restr. log likelihood** is the maximized log likelihood value, when all slope coefficients are restricted to zero, $l(\check{\beta})$. Since the constant term is included, this specification is equivalent to estimating the unconditional mean probability of “success”.
- The **LR statistic** tests the joint null hypothesis that all slope coefficients except the constant are zero and is computed as $-2(l(\check{\beta}) - l(\hat{\beta}))$. This statistic, which is only reported when you include a constant in your specification, is used to test the overall significance of the model. The number in parentheses is the degrees of freedom, which is the number of restrictions under test.
- **Probability(LR stat)** is the p -value of the LR test statistic. Under the null hypothesis, the LR test statistic is asymptotically distributed as a χ^2 variable, with degrees of freedom equal to the number of restrictions under test.
- **McFadden R-squared** is the likelihood ratio index computed as $1 - l(\hat{\beta})/l(\check{\beta})$, where $l(\check{\beta})$ is the restricted log likelihood. As the name suggests, this is an analog to the R^2 reported in linear regression models. It has the property that it always lies between zero and one.
- The various information criteria are detailed in [Appendix F, “Information Criteria”, beginning on page 683](#). For additional discussion, see Grasa (1989).

Estimation Options

The iteration limit and convergence criterion may be set in the usual fashion from the Options dialog. In addition, there are options that are specific to binary models. These options are described below.

Robust Standard Errors

For binary dependent variable models, EViews allows you to estimate the standard errors using quasi-maximum likelihood (**Huber/White**) or generalized linear model (**GLM**) methods. See [“Technical Notes” on page 467](#) for a discussion of these two methods.

Click **Options** in the Equation Specification dialog box and check the **Robust Covariance** box and select one of the two methods. When you estimate the binary model using this option, the header in the equation output will indicate the method used to compute the coefficient covariance matrix.

Starting Values

As with other estimation procedures, EViews allows you to specify starting values. In the options menu, select one of the items from the combo box. You can use the default EViews values, or you can choose a fraction of those values, zero coefficients, or user supplied values. To employ the latter, enter the coefficients in the C coefficient vector, and select **User Supplied** in the combo box.

The EViews default values are selected using a sophisticated algorithm that is specialized for each type of binary model. Unless there is a good reason to choose otherwise, we recommend that you use the default values.

Estimation Algorithm

By default, EViews uses quadratic hill-climbing to obtain parameter estimates. This algorithm uses the matrix of analytic second derivatives of the log likelihood in forming iteration updates and in computing the estimated covariance matrix of the coefficients.

If you wish, you can employ a different estimation algorithm: Newton-Raphson also employs second derivatives (without the diagonal weighting); BHHH uses first derivatives to determine both iteration updates and the covariance matrix estimates (see [Appendix D, “Estimation Algorithms and Options”, on page 663](#)). To employ one of these latter methods, click on **Options** in the Equation specification dialog box, and select the desired method.

Estimation Problems

In general, estimation of binary models is quite straightforward, and you should experience little difficulty in obtaining parameter estimates. There are a few situations, however, where you may experience problems.

First, you may get the error message “Dependent variable has no variance.” This error means that there is no variation in the dependent variable (the variable is always one or zero for all valid observations). This error most often occurs when EViews excludes the entire sample of observations for which y takes values other than zero or one, leaving too few observations for estimation.

You should make certain to recode your data so that the binary indicators take the values zero and one. This requirement is not as restrictive as it may first seem, since the recoding may easily be done using auto-series. Suppose, for example, that you have data where y takes the values 1000 and 2000. You could then use the boolean auto-series, “ $y=1000$ ”, or perhaps, “ $y<1500$ ”, as your dependent variable.

Second, you may receive an error message of the form “[xxxx] perfectly predicts binary response [success/failure]”, where xxxx is a sample condition. This error occurs when one of the regressors contains a separating value for which all of the observations with values below the threshold are associated with a single binary response, and all of the values above the threshold are associated with the alternative response. In this circumstance, the method of maximum likelihood breaks down.

For example, if all values of the explanatory variable $x > 0$ are associated with $y = 1$, then x is a perfect predictor of the dependent variable, and EViews will issue an error message and stop the estimation procedure.

The only solution to this problem is to remove the offending variable from your specification. Usually, the variable has been incorrectly entered in the model, as when a researcher includes a dummy variable that is identical to the dependent variable (for discussion, see Greene, 1997).

Thirdly, you may experience the error, “Non-positive likelihood value observed for observation [xxxx].” This error most commonly arises when the starting values for estimation are poor. The default EViews starting values should be adequate for most uses. You may wish to check the Options dialog to make certain that you are not using user specified starting values, or you may experiment with alternative user-specified values.

Lastly, the error message “Near-singular matrix” indicates that EViews was unable to invert the matrix required for iterative estimation. This will occur if the model is not identified. It may also occur if the current parameters are far from the true values. If you believe the latter to be the case, you may wish to experiment with starting values or the

estimation algorithm. The BHHH and quadratic hill-climbing algorithms are less sensitive to this particular problem than is Newton-Raphson.

Views of Binary Equations

EViews provides a number of standard views and procedures for binary models. For example, you can easily perform Wald or likelihood ratio tests by selecting **View/Coefficient Tests**, and then choosing the appropriate test. In addition, EViews allows you to examine and perform tests using the residuals from your model. The ordinary residuals used in most calculations are described above—additional residual types are defined below. Note that some care should be taken in interpreting test statistics that use these residuals since some of the underlying test assumptions may not be valid in the current setting.

There are a number of additional specialized views and procedures which allow you to examine the properties and performance of your estimated binary model.

Categorical Regressor Stats

This view displays descriptive statistics (mean and standard deviation) for each regressor. The descriptive statistics are computed for the whole sample, as well as the sample broken down by the value of the dependent variable y .

Expectation-Prediction (Classification) Table

This view displays 2×2 tables of correct and incorrect classification based on a user specified prediction rule, and on expected value calculations. Click on **View/Expectation-Prediction Table**. EViews opens a dialog prompting you to specify a prediction cutoff value, p , lying between zero and one. Each observation will be classified as having a predicted probability that lies above or below this cutoff.

After you enter the cutoff value and click on **OK**, EViews will display four (bordered) 2×2 tables in the equation window. Each table corresponds to a contingency table of the predicted response classified against the observed dependent variable. The top two tables and associated statistics depict the classification results based given the specified cutoff values.

Dependent Variable: GRADE

Method: ML - Binary Probit

Date: 07/31/00 Time: 15:57

Sample: 1 32

Included observations: 32

Prediction Evaluation (success cutoff C = 0.5)

	Estimated Equation			Constant Probability		
	Dep=0	Dep=1	Total	Dep=0	Dep=1	Total
P(Dep=1)≤C	18	3	21	21	11	32
P(Dep=1)>C	3	8	11	0	0	0
Total	21	11	32	21	11	32
Correct	18	8	26	21	0	21
% Correct	85.71	72.73	81.25	100.00	0.00	65.62
% Incorrect	14.29	27.27	18.75	0.00	100.00	34.38
Total Gain*	-14.29	72.73	15.62			
Percent Gain**	NA	72.73	45.45			

In the left-hand table, we classify observations as having predicted probabilities $\hat{p}_i = 1 - F(-x_i' \hat{\beta})$ that are above or below the specified cutoff value (here set to the default of 0.5). In the upper right-hand table, we classify observations using \bar{p} , the sample proportion of $y = 1$ observations. This probability, which is constant across individuals, is the value computed from estimating a model that includes only the intercept term, C.

“Correct” classifications are obtained when the predicted probability is less than or equal to the cutoff and the observed $y = 0$, or when the predicted probability is greater than the cutoff and the observed $y = 1$. In the example above, 18 of the Dep = 0 observations and 8 of the Dep = 1 observations are correctly classified by the estimated model.

It is worth noting that in the statistics literature, what we term the expectation-prediction table is sometimes referred to as the *classification table*. The fraction of $y = 1$ observations that are correctly predicted is termed the *sensitivity*, while the fraction of $y = 0$ observations that are correctly predicted is known as *specificity*. In EViews, these two values, expressed in percentage terms, are labeled “% Correct”. Overall, the estimated model correctly predicts 81.25% of the observations (85.71% of the Dep = 0 and 72.73% of the Dep = 1 observations).

The gain in the number of correct predictions obtained in moving from the right table to the left table provides a measure of the predictive ability of your model. The gain measures are reported in both absolute percentage increases (**Total Gain**), and as a percentage of the incorrect classifications in the constant probability model (**Percent Gain**). In the example above, the restricted model predicts that all 21 individuals will have Dep = 0. This prediction is correct for the 21 $y = 0$ observations, but is incorrect for the 11 $y = 1$ observations.

The estimated model improves on the Dep = 1 predictions by 72.73 percentage points, but does more poorly on the Dep = 0 predictions (-14.29 percentage points). Overall, the estimated equation is 15.62 percentage points better at predicting responses than the constant

probability model. This change represents a 45.45 percent improvement over the 65.62 percent correct prediction of the default model.

The bottom portion of the equation window contains analogous prediction results based upon expected value calculations.

	Estimated Equation			Constant Probability		
	Dep=0	Dep=1	Total	Dep=0	Dep=1	Total
E(# of Dep=0)	16.89	4.14	21.03	13.78	7.22	21.00
E(# of Dep=1)	4.11	6.86	10.97	7.22	3.78	11.00
Total	21.00	11.00	32.00	21.00	11.00	32.00
Correct	16.89	6.86	23.74	13.78	3.78	17.56
% Correct	80.42	62.32	74.20	65.62	34.38	54.88
% Incorrect	19.58	37.68	25.80	34.38	65.62	45.12
Total Gain*	14.80	27.95	19.32			
Percent Gain**	43.05	42.59	42.82			

In the left-hand table, we compute the expected number of $y = 0$ and $y = 1$ observations in the sample. For example, **E(# of Dep = 0)** is computed as

$$\sum_i \Pr(y_i = 0 | x_i, \beta) = \sum_i F(-x_i' \hat{\beta}), \quad (17.13)$$

where the cumulative distribution function F is for the normal, logistic, or extreme value distribution.

In the lower right-hand table, we compute the expected number of $y = 0$ and $y = 1$ observations for a model estimated with only a constant. For this restricted model, **E(# of Dep = 0)** is computed as $n(1 - \bar{p})$, where \bar{p} is the sample proportion of $y = 1$ observations. EViews also reports summary measures of the total gain and the percent (of the incorrect expectation) gain.

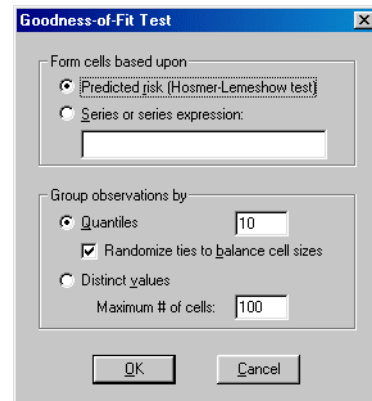
Among the 21 individuals with $y = 0$, the expected number of $y = 0$ observations in the estimated model is 16.89. Among the 11 observations with $y = 1$, the expected number of $y = 1$ observations is 6.86. These numbers represent roughly a 19.32 percentage point (42.82 percent) improvement over the constant probability model.

Goodness-of-Fit Tests

This view allows you to perform Pearson χ^2 -type tests of goodness-of-fit. EViews carries out two goodness-of-fit tests: Hosmer-Lemeshow (1989) and Andrews (1988a, 1988b). The idea underlying these tests is to compare the fitted expected values to the actual values by *group*. If these differences are “large”, we reject the model as providing an insufficient fit to the data.

Details on the two tests are described in the “[Technical Notes](#)” on page 467. Briefly, the tests differ in how the observations are grouped and in the asymptotic distribution of the test statistic. The Hosmer-Lemeshow test groups observations on the basis of the predicted probability that $y = 1$. The Andrews test is a more general test that groups observations on the basis of any series or series expression.

To carry out the test, select **View/Goodness-of-Fit Test...**



You must first decide on the grouping variable. You can select Hosmer-Lemeshow (predicted probability) grouping by clicking on the corresponding radio button, or you can select series grouping, and provide a series to be used in forming the groups.

Next, you need to specify the grouping rule. EVIEWS allows you to group on the basis of either distinct values or quantiles of the grouping variable.

If your grouping variable takes relatively few distinct values, you should choose the **Distinct values** grouping. EVIEWS will form a separate group for each distinct value of the grouping variable. For example, if your grouping variable is TUCE, EVIEWS will create a group for each distinct TUCE value and compare the expected and actual numbers of $y = 1$ observations in each group. By default, EVIEWS limits you to 100 distinct values. If the distinct values in your grouping series exceeds this value, EVIEWS will return an error message. If you wish to evaluate the test for more than 100 values, you must explicitly increase the maximum number of distinct values.

If your grouping variable takes on a large number of distinct values, you should select **Quantiles**, and enter the number of desired bins in the edit field. If you select this method, EVIEWS will group your observations into the number of specified bins, on the basis of the ordered values of the grouping series. For example, if you choose to group by TUCE, select **Quantiles**, and enter 10, EVIEWS will form groups on the basis of TUCE deciles.

If you choose to group by quantiles and there are ties in the grouping variable, EVIEWS may not be able to form the exact number of groups you specify unless tied values are assigned to different groups. Furthermore, the number of observations in each group may be very unbalanced. Selecting the **randomize ties** option randomly assigns ties to adjacent groups in order to balance the number of observations in each group.

Since the properties of the test statistics require that the number of observations in each group is “large”, some care needs to be taken in selecting a rule so that you do not end up with a large number of cells, each containing small numbers of observations.

By default, EViews will perform the test using Hosmer-Lemeshow grouping. The default grouping method is to form deciles. The test result using the default specification is given by:

Dependent Variable: GRADE
Method: ML - Binary Probit
Date: 07/31/00 Time: 15:57
Sample: 1 32
Included observations: 32
Andrews and Hosmer-Lemeshow Goodness-of-Fit Tests
Grouping based upon predicted risk (randomize ties)

	Quantile of Risk		Dep=0		Dep=1		Total Obs	H-L Value
	Low	High	Actual	Expect	Actual	Expect		
1	0.0161	0.0185	3	2.94722	0	0.05278	3	0.05372
2	0.0186	0.0272	3	2.93223	0	0.06777	3	0.06934
3	0.0309	0.0457	3	2.87888	0	0.12112	3	0.12621
4	0.0531	0.1088	3	2.77618	0	0.22382	3	0.24186
5	0.1235	0.1952	2	3.29779	2	0.70221	4	2.90924
6	0.2732	0.3287	3	2.07481	0	0.92519	3	1.33775
7	0.3563	0.5400	2	1.61497	1	1.38503	3	0.19883
8	0.5546	0.6424	1	1.20962	2	1.79038	3	0.06087
9	0.6572	0.8342	0	0.84550	3	2.15450	3	1.17730
10	0.8400	0.9522	1	0.45575	3	3.54425	4	0.73351
	Total		21	21.0330	11	10.9670	32	6.90863
H-L Statistic:			6.9086		Prob. Chi-Sq(8)		0.5465	
Andrews Statistic:			20.6045		Prob. Chi-Sq(10)		0.0240	

The columns labeled “Quantiles of Risk” depict the high and low value of the predicted probability for each decile. Also depicted are the actual and expected number of observations in each group, as well as the contribution of each group to the overall Hosmer-Lemeshow (H-L) statistic—large values indicate large differences between the actual and predicted values for that decile.

The χ^2 statistics are reported at the bottom of the table. Since grouping on the basis of the fitted values falls within the structure of an Andrews test, we report results for both the H-L and the Andrews test statistic. The p -value for the HL test is large while the value for the Andrews test statistic is small, providing mixed evidence of problems. Furthermore, the relatively small sample sizes suggest that caution is in order in interpreting the results.

Procedures for Binary Equations

In addition to the usual procedures for equations, EViews allows you to forecast the dependent variable and linear index, or to compute a variety of residuals associated with the binary model.

Forecast

EViews allows you to compute either the fitted probability, $\hat{p}_i = 1 - F(-x_i' \hat{\beta})$, or the fitted values of the index $x_i' \beta$. From the equation toolbar select **Procs/Forecast (Fitted Probability/Index)...**, and then click on the desired entry.

As with other estimators, you can select a forecast sample, and display a graph of the forecast. If your explanatory variables, x_t , include lagged values of the binary dependent variable y_t , forecasting with the **Dynamic** option instructs EViews to use the fitted values \hat{p}_{t-1} , to derive the forecasts, in contrast with the **Static** option, which uses the actual (lagged) y_{t-1} .

Neither forecast evaluations nor automatic calculation of standard errors of the forecast are currently available for this estimation method. The latter can be computed using the variance matrix of the coefficients displayed by **View/Covariance Matrix**, or using the `@covariance` function.

You can use the fitted index in a variety of ways, for example, to compute the marginal effects of the explanatory variables. Simply forecast the fitted index and save the results in a series, say XB. Then the auto-series `@dnorm(-xb)`, `@dlogistic(-xb)`, or `@dextreme(-xb)` may be multiplied by the coefficients of interest to provide an estimate of the derivatives of the expected value of y_i with respect to the j -th variable in x_i :

$$\frac{\partial E(y_i | x_i, \beta)}{\partial x_{ij}} = f(-x_i' \beta) \beta_j. \quad (17.14)$$

Make Residual Series

Procs/Make Residual Series gives you the option of generating one of the following three types of residuals:

Ordinary	$e_{oi} = y_i - \hat{p}_i$
Standardized	$e_{si} = \frac{y_i - \hat{p}_i}{\sqrt{\hat{p}_i(1 - \hat{p}_i)}}$
Generalized	$e_{gi} = \frac{(y_i - \hat{p}_i)f(-x_i' \hat{\beta})}{\hat{p}_i(1 - \hat{p}_i)} \phi$

where $\hat{p}_i = 1 - F(-x_i' \hat{\beta})$ is the fitted probability, and the distribution and density functions F and f , depend on the specified distribution.

The ordinary residuals have been described above. The standardized residuals are simply the ordinary residuals divided by an estimate of the theoretical standard deviation. The

generalized residuals are derived from the first order conditions that define the ML estimates. The first order conditions may be regarded as an orthogonality condition between the generalized residuals and the regressors x .

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^N \frac{(y_i - (1 - F(-x_i' \beta)))f(-x_i' \beta)}{F(-x_i' \beta)(1 - F(-x_i' \beta))} \cdot x_i = \sum_{i=1}^N e_{g,i} \cdot x_i. \quad (17.15)$$

This property is analogous to the orthogonality condition between the (ordinary) residuals and the regressors in linear regression models.

The usefulness of the generalized residuals derives from the fact that you can easily obtain the score vectors by multiplying the generalized residuals by each of the regressors in x . These scores can be used in a variety of LM specification tests (see Chesher, Lancaster and Irish (1985), and Gourieroux, Monfort, Renault, and Trognon (1987)). We provide an example below.

Demonstrations

You can easily use the results of a binary model in additional analysis. Here, we provide demonstrations of using EViews to plot a probability response curve and to test for heteroskedasticity in the residuals.

Plotting Probability Response Curves

You can use the estimated coefficients from a binary model to examine how the predicted probabilities vary with an independent variable. To do so, we will use the EViews built-in modeling features.

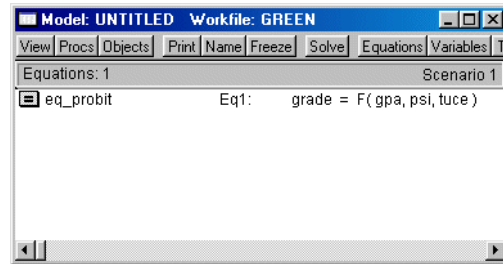
For the probit example above, suppose we are interested in the effect of teaching method (PSI) on educational improvement (GRADE). We wish to plot the fitted probabilities of GRADE improvement as a function of GPA for the two values of PSI, fixing the values of other variables at their sample means.

We will perform the analysis using a grid of values for GPA from 2 to 4. First, we will create a series containing the values of GPA for which we wish to examine the fitted probabilities for GRADE. The easiest way to do this is to use the `@trend` function to generate a new series:

```
series gpa_plot=2+(4-2)*@trend/(@obs(@trend)-1)
```

`@trend` creates a series that begins at 0 in the first observation of the sample, and increases by 1 for each subsequent observation, up through `@obs-1`.

Next, we will use a model object to define and perform the desired computations. The following discussion skims over many of the useful features of EViews models. Those wishing greater detail should consult [Chapter 23, “Models”](#), beginning on page 601.

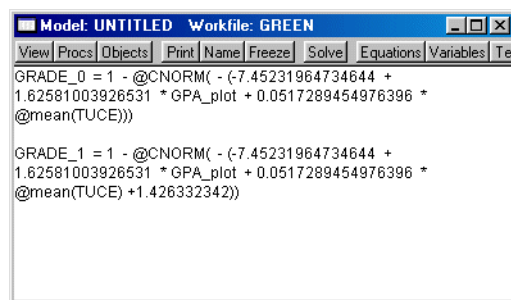


First, we create a model out of the estimated equation by selecting **Procs/Make Model** from the equation toolbar. EViews will create an untitled model object containing a link to the estimated equation and will open the model window.

Next we want to edit this model specification so that calculations are performed using our simulation values. To do so we must first break the link between the original equation and the model specification by selecting **Procs/Links/Break All Links**. Next, click on the Text button or select **View/Source Text** to display the text editing screen.

We wish to create two separate equations: one with the value of PSI set to 0 and one with the value of PSI set to 1 (you can, of course, use copy-and-paste to aid in creating the additional equation). We will also edit the specification so that references to GPA are replaced with the series of simulation values GPA_PLOT, and references to TUCE are replaced with the calculated mean, “@MEAN(TUCE)”. The GRADE_0 equation sets PSI to 0, while the GRADE_1 contains an additional expression, 1.426332342, which is the coefficient on the PSI variable.

Once you have edited your model, click on **Solve** and set the “Active” solution scenario to “Actuals”. This tells EViews that you wish to place the solutions in the series “GRADE_0” and “GRADE_1” as specified in the equation definitions. You can safely ignore the remaining solution settings and simply click on **OK**. EViews will report that your model has solved successfully.



You are now ready to plot results. Select **Object/New Object/Group**, and enter

```
gpa_plot grade_0 grade_1
```

EViews will open an untitled group window containing these two series. Select **View/Graph/XY line** to display the probability of GRADE improvement plotted against GPA for those with and without PSI (and with the TUCE evaluated at means).

EViews will open a group window containing these series. Select **View/Graph/XY line** from the group toolbar to display a graph of your results.

We have annotated the graph slightly so that you can better judge the effect of the new teaching methods (PSI) on the GPA—Grade Improvement relationship.

Testing for Heteroskedasticity

As an example of specification tests for binary dependent variable models, we carry out the LM test for heteroskedasticity using the artificial regression method described by Davidson and MacKinnon (1993, section 15.4).

We test the null hypothesis of

homoskedasticity against the alternative of heteroskedasticity of the form

$$\text{var}(u_i) = \exp(2z_i'\gamma), \quad (17.16)$$

where γ is an unknown parameter. In this example, we take PSI as the only variable in z . The test statistic is the explained sum of squares from the regression

$$\frac{(y_i - \hat{p}_i)}{\sqrt{\hat{p}_i(1 - \hat{p}_i)}} = \frac{f(-x_i'\hat{\beta})}{\sqrt{\hat{p}_i(1 - \hat{p}_i)}} x_i' b_1 + \frac{f(-x_i'\hat{\beta})(-x_i'\hat{\beta})}{\sqrt{\hat{p}_i(1 - \hat{p}_i)}} z_i' b_2 + v_i, \quad (17.17)$$

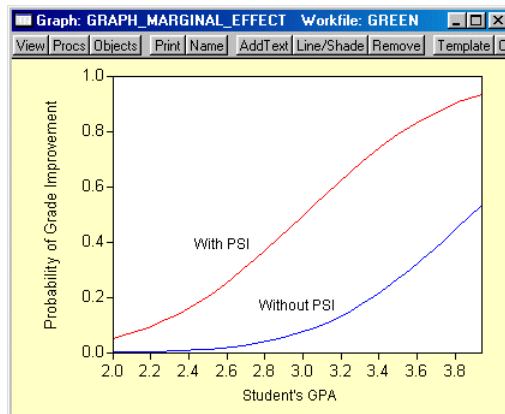
which is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of variables in z (in this case 1).

To carry out the test, we first retrieve the fitted probabilities \hat{p}_i and fitted index $x_i'\hat{\beta}$. Click on the **Forecast** button and first save the fitted probabilities as P_HAT and then the index as XB (you will have to click **Forecast** twice to save the two series).

Next, the dependent variable in the test regression may be obtained as the standardized residual. Select **Procs/Make Residual Series...** and select **Standardized Residual**. We will save the series as BRMR_Y.

Lastly, we will use the built-in EViews functions for evaluating the normal density and cumulative distribution function to create a group object containing the independent variables:

```
series fac=@dnorm(-xb)/@sqrt(p_hat*(1-p_hat))
group brmr_x fac (gpa*fac) (tuce*fac) (psi*fac)
```



Then run the artificial regression by clicking on **Quick/Estimate Equation...**, selecting **Least Squares**, and entering:

```
brmr_y brmr_x (psi*(-xb)*fac)
```

You can obtain the fitted values by clicking on the **Forecast** button in the equation toolbar of this artificial regression. The LM test statistic is the sum of squares of these fitted values. If the fitted values from the artificial regression are saved in BRMR_YF, the test statistic can be saved as a scalar named LM_TEST:

```
scalar lm_test=@sumsq(brmr_yf)
```

which contains the value 1.5408. You can compare the value of this test statistic with the critical values from the chi-square table with one degree of freedom. To save the p -value as a scalar, enter the command:

```
scalar p_val=1-@cchisq(lm_test,1)
```

To examine the value of LM_TEST or P_VAL, double click on the name in the workfile window; the value will be displayed in the status line at the bottom of the EViews window. The p -value in this example is roughly 0.21, so we have little evidence against the null hypothesis of homoskedasticity.

Ordered Dependent Variable Models

EViews estimates the ordered-response model of Aitchison and Silvey (1957) under a variety of assumptions about the latent error distribution. In ordered dependent variable models, the observed y denotes outcomes representing ordered or ranked categories. For example, we may observe individuals who choose between one of four educational outcomes: less than high school, high school, college, advanced degree. Or we may observe individuals who are employed, partially retired, or fully retired.

As in the binary dependent variable model, we can model the observed response by considering a latent variable y_i^* that depends linearly on the explanatory variables x_i :

$$y_i^* = x_i' \beta + \epsilon_i \quad (17.18)$$

where ϵ_i are independent and identically distributed random variables. The observed y_i is determined from y_i^* using the rule:

$$y_i = \begin{cases} 0 & \text{if } y_i^* \leq \gamma_1 \\ 1 & \text{if } \gamma_1 < y_i^* \leq \gamma_2 \\ 2 & \text{if } \gamma_2 < y_i^* \leq \gamma_3 \\ \vdots & \vdots \\ M & \text{if } \gamma_M < y_i^* \end{cases} \quad (17.19)$$

It is worth noting that the actual values chosen to represent the categories in y are completely arbitrary. All the ordered specification requires is for ordering to be preserved so that $y_i^* < y_j^*$ implies that $y_i < y_j$.

It follows that the probabilities of observing each value of y are given by

$$\begin{aligned} \Pr(y_i = 0 | x_i, \beta, \gamma) &= F(\gamma_1 - x_i' \beta) \\ \Pr(y_i = 1 | x_i, \beta, \gamma) &= F(\gamma_2 - x_i' \beta) - F(\gamma_1 - x_i' \beta) \\ \Pr(y_i = 2 | x_i, \beta, \gamma) &= F(\gamma_3 - x_i' \beta) - F(\gamma_2 - x_i' \beta) \\ &\dots \\ \Pr(y_i = M | x_i, \beta, \gamma) &= 1 - F(\gamma_M - x_i' \beta) \end{aligned} \quad (17.20)$$

where F is the cumulative distribution function of ϵ .

The threshold values γ are estimated along with the β coefficients by maximizing the log likelihood function:

$$l(\beta, \gamma) = \sum_{i=1}^N \sum_{j=0}^M \log(\Pr(y_i = j | x_i, \beta, \gamma)) \cdot 1(y_i = j) \quad (17.21)$$

where $1(\cdot)$ is an indicator function which takes the value 1 if the argument is true, and 0 if the argument is false. By default, EViews uses analytic second derivative methods to obtain parameter and variance matrix of the estimated coefficient estimates (see [“Quadratic hill-climbing \(Goldfeld-Quandt\)” on page 664](#)).

Estimating Ordered Models in EViews

Suppose that the dependent variable DANGER is an index ordered from 1 (least dangerous animal) to 5 (most dangerous animal). We wish to model this ordered dependent variable as a function of the explanatory variables, BODY, BRAIN and SLEEP. Note that the values that we have assigned to the dependent variable are not relevant, only the ordering implied by those values. EViews will estimate an identical model if the dependent variable is recorded to take the values 1, 2, 3, 4, 5 or 10, 234, 3243, 54321, 123456.

To estimate this model, choose **Objects/New Object...** from the main menu and select the **Equation** object from the main menu. From the Equation Specification dialog, select estimation method **ORDERED**. The standard estimation dialog will change to match this specification.

There are three parts to specifying an ordered variable model: the equation specification, the error specification, and the sample specification. First, in the **Equation Specification** field you should type the name of the ordered dependent variable followed by the list of your regressors. In our example, you will enter:

```
danger body brain sleep
```

Ordered estimation only supports specification by list so you may not enter an explicit equation.

Also keep in mind that:

- A separate constant term is not separately identified from the limit points γ , so EViews will ignore any constant term in your specification. Thus, the model

```
danger c body brain sleep
```

is equivalent to the specification above.

- EViews requires the dependent variable to be integer valued, otherwise you will see an error message, and estimation will stop. This is not, however, a serious restriction, since you can easily convert the series into an integer using `@round`, `@floor` or `@ceil` in an auto-series expression.

Next, select between the ordered logit, ordered probit, and the ordered extreme value models by choosing one of the three distributions for the latent error term.

Lastly, specify the estimation sample.

Now click on **OK**, EViews will estimate the parameters of the model using iterative procedures.

Once the estimation procedure converges, EViews will display the estimation results in the equation window. The first part of the table contains the usual header information, including the assumed error distribution, estimation sample, iteration and convergence information, number of distinct values for y , and the method of computing the coefficient covariance matrix.

Dependent Variable: DANGER
 Method: ML - Ordered Probit
 Date: 09/13/97 Time: 10:00
 Sample(adjusted): 1 61
 Included observations: 58
 Excluded observations: 3 after adjusting endpoints
 Number of ordered indicator values: 5
 Convergence achieved after 5 iterations
 Covariance matrix computed using second derivatives

	Coefficient	Std. Error	z-Statistic	Prob.
BODY	0.006346	0.003262	1.945385	0.0517
BRAIN	-0.003506	0.001822	-1.924244	0.0543
SLEEP	-0.158596	0.040440	-3.921741	0.0001

Below the header information are the coefficient estimates and asymptotic standard errors, and the corresponding z-statistics and significance levels. The estimated coefficients of the ordered model must be interpreted with care (see Greene (1997, section 19.8) or Johnston and DiNardo (1997, section 13.9)).

The *sign* of $\hat{\beta}_j$ shows the direction of the change in the probability of falling in the end-point rankings ($y = 0$ or $y = 1$) when x_{ij} changes. $\Pr(y = 0)$ changes in the *opposite* direction of the sign of $\hat{\beta}_j$ and $\Pr(y = M)$ changes in the *same* direction as the sign of $\hat{\beta}_j$. The effects on the probability of falling in any of the middle rankings are given by:

$$\frac{\partial \Pr(y = k)}{\partial \beta_j} = \frac{\partial F(\gamma_{k+1} - x_i' \beta)}{\partial \beta_j} - \frac{\partial F(\gamma_k - x_i' \beta)}{\partial \beta_j} \quad (17.22)$$

for $k = 1, 2, \dots, M - 1$. It is impossible to determine the signs of these terms, *a priori*.

The lower part of the estimation output, labeled “Limit Points”, presents the estimates of the γ coefficients and the associated standard errors and probability values:

LimitPoints				
Limit_2:C(4)	-2.382697	0.512993	-4.644695	0.0000
Limit_3:C(5)	-1.598777	0.484884	-3.297237	0.0010
Limit_4:C(6)	-1.028655	0.465433	-2.210104	0.0271
Limit_5:C(7)	-0.241152	0.445500	-0.541307	0.5883
Akaike info criterion	11.74480	Schwarz criterion	11.99348	
Log likelihood	-333.5993	Hannan-Quinn criter.	11.84167	
Avg. log likelihood	-5.751712			

Note that the coefficients are labeled both with the identity of the limit point, and the coefficient number. Just below the limit points are the summary statistics for the equation.

Estimation Problems

Most of the previous discussion of estimation problems for binary models (page 428) also holds for ordered models. In general, these models are well-behaved and will require little intervention.

There are cases, however, where problems will arise. First, EViews currently has a limit of 750 total coefficients in an ordered dependent variable model. Thus, if you have 25 right-hand side variables, and a dependent variable with 726 distinct values, you will be unable to estimate your model using EViews.

Second, you may run into identification problems and estimation difficulties if you have some groups where there are very few observations. If necessary, you may choose to combine adjacent groups and re-estimate the model.

EViews may stop estimation with the message “Parameter estimates for limit points are non-ascending”, most likely on the first iteration. This error indicates that parameter values for the limit points were invalid, and that EViews was unable to adjust these values to make them valid. Make certain that if you are using user defined parameters, the limit points are strictly increasing. Better yet, we recommend that you employ the EViews starting values since they are based on a consistent first-stage estimation procedure, and should therefore be quite well-behaved.

Views of Ordered Equations

EViews provides you with several views of an ordered equation. As with other equations, you can examine the specification and estimated covariance matrix as well as perform Wald and likelihood ratio tests on coefficients of the model. In addition, there are several views that are specialized for the ordered model:

- **Dependent Variable Frequencies** — computes a one-way frequency table for the ordered dependent variable for the observations in the estimation sample. EViews presents both the frequency table and the cumulative frequency table in levels and percentages.
- **Expectation-Prediction Table** — classifies observations on the basis of the predicted response. EViews performs the classification on the basis of maximum predicted probability as well as the expected probability.

Dependent Variable: DANGER
 Method: ML - Ordered Probit
 Date: 09/13/97 Time: 10:00
 Sample(adjusted): 1 61
 Included observations: 58
 Excluded observations: 3 after adjusting endpoints
 Prediction table for ordered dependent variable

Value	Count	Count of obs with Max Prob	Error	Sum of all Probabilities	Error
1	18	27	-9	18.571	-0.571
2	14	16	-2	13.417	0.583
3	10	0	10	9.163	0.837
4	9	8	1	8.940	0.060
5	7	7	0	7.909	-0.909

There are two columns labeled “Error”. The first measures the difference between the observed count and the number of observations where the probability of that response is highest. For example, 18 individuals reported a value of 1 for DANGER, while 27 individuals had predicted probabilities that were highest for this value. The actual count minus the predicted is -9 . The second error column measures the difference between the actual number of individuals reporting the value, and the sum of all of the individual probabilities for that value.

Procedures for Ordered Equations

Make Ordered Limit Vector/Matrix

The full set of coefficients and the covariance matrix may be obtained from the estimated equation in the usual fashion (see [“Working With Equation Statistics”](#) on page 270). In some circumstances, however, you may wish to perform inference using only the estimates of the γ coefficients and the associated covariances.

The **Make Ordered Limit Vector** and **Make Ordered Limit Covariance Matrix** procedures provide a shortcut method of obtaining the estimates associated with the γ coefficients. The first procedure creates a vector (using the next unused name of the form LIMITS01, LIMITS02, etc.) containing the estimated γ coefficients. The latter procedure creates a symmetric matrix containing the estimated covariance matrix of the γ . The matrix will be given an unused name of the form VLIMITS01, VLIMITS02, etc., where the “V” is used to indicate that these are the variances of the estimated limit points.

Forecasting using Models

You cannot forecast directly from an estimated ordered model since the dependent variable represents categorical or rank data. EViews does, however, allow you to forecast the probability associated with each category. To forecast these probabilities, you must first create a

model. Choose **Procs/Make Model** and EViews will open an untitled model window containing a system of equations, with a separate equation for the probability of each ordered response value.

To forecast from this model, simply click the Solve button in the model window toolbar. If you select Scenario 1 as your solution scenario, the default settings will save your results in a set of named series with “_1” appended to the end of the each underlying name. See [Chapter 23, “Models”, beginning on page 601](#) for additional detail on modifying and solving models.

For this example, the series I_DANGER_1 will contain the fitted linear index $x_i' \hat{\beta}$. The fitted probability of falling in category 1 will be stored as a series named DANGER_1_1, the fitted probability of falling in category 2 will be stored as a series named DANGER_2_1, and so on. Note that for each observation, the fitted probability of falling in each of the categories sums up to one.

Make Residual Series

The generalized residuals of the ordered model are the derivatives of the log likelihood with respect to a hypothetical unit- x variable. These residuals are defined to be uncorrelated with the explanatory variables of the model (see Chesher and Irish (1987), and Gouriéroux, Monfort, Renault and Trognon (1987) for details), and thus may be used in a variety of specification tests.

To create a series containing the generalized residuals, select **View/Make Residual Series...**, enter a name or accept the default name, and click **OK**. The generalized residuals for an ordered model are given by:

$$e_{gi} = \frac{f(\gamma_g - x_i' \hat{\beta}) - f(\gamma_{g-1} - x_i' \hat{\beta})}{F(\gamma_g - x_i' \hat{\beta}) - F(\gamma_{g-1} - x_i' \hat{\beta})}, \quad (17.23)$$

where $\gamma_0 = -\infty$, and $\gamma_{M+1} = \infty$.

Censored Regression Models

In some settings, the dependent variable is only partially observed. For example, in survey data, data on incomes above a specified level are often top-coded to protect confidentiality. Similarly desired consumption on durable goods may be censored at a small positive or zero value. EViews provides tools to perform maximum likelihood estimation of these models and to use the results for further analysis.

Theory

Consider the following latent variable regression model

$$y_i^* = x_i' \beta + \sigma \epsilon_i, \quad (17.24)$$

where σ is a scale parameter. The scale parameter σ is identified in censored and truncated regression models, and will be estimated along with the β .

In the canonical *censored regression model*, known as the *tobit*, the observed data y are given by

$$y_i = \begin{cases} 0 & \text{if } y_i^* \leq 0 \\ y_i^* & \text{if } y_i^* > 0 \end{cases} \quad (17.25)$$

In other words, all negative values of y_i^* are coded as 0. We say that these data are *left censored* at 0. Note that this situation differs from a *truncated regression model* where negative values of y_i^* are dropped from the sample. More generally, EViews allows for both left and right censoring at arbitrary limit points so that

$$y_i = \begin{cases} \underline{c}_i & \text{if } y_i^* \leq \underline{c}_i \\ y_i^* & \text{if } \underline{c}_i < y_i^* \leq \bar{c}_i \\ \bar{c}_i & \text{if } \bar{c}_i < y_i^* \end{cases} \quad (17.26)$$

where $\underline{c}_i, \bar{c}_i$ are fixed numbers representing the censoring points. If there is no left censoring, then we can set $\underline{c}_i = -\infty$. If there is no right censoring, then $\bar{c}_i = \infty$. The canonical tobit model is a special case with $\underline{c}_i = 0$ and $\bar{c}_i = \infty$.

The parameters β, σ are estimated by maximizing the log likelihood function

$$\begin{aligned} l(\beta, \sigma) = & \sum_{i=1}^N \log f((y_i - x_i' \beta) / \sigma) \cdot 1(\underline{c}_i < y_i < \bar{c}_i) \\ & + \log(F((\underline{c}_i - x_i' \beta) / \sigma)) \cdot 1(y_i = \underline{c}_i) \\ & + \log(1 - F((\bar{c}_i - x_i' \beta) / \sigma)) \cdot 1(y_i = \bar{c}_i) \end{aligned} \quad (17.27)$$

where f, F are the density and cumulative distribution functions of ϵ .

Estimating Censored Models in EViews

Consider the model

$$\text{HRS}_i = \beta_1 + \beta_2 \text{AGE}_i + \beta_3 \text{EDU}_i + \beta_4 \text{KID}_i + \epsilon_i, \quad (17.28)$$

where hours worked (HRS) is left censored at zero. To estimate this model, select **Quick/Estimate Equation...** from the main menu. Then from the Equation Specification dialog,

select the **CENSORED** estimation method. The dialog will change to provide a number of different input options.

Specifying the Regression Equation

In the **Equation Specification** field, enter the name of the censored dependent variable followed by a list of regressors. In our example, you will enter

```
hrs c age edu kid
```

Censored estimation only supports specification by list so you may not enter an explicit equation.

Next, select one of the three distributions for the error term. EViews allows you three possible choices for the distribution of ϵ :

Standard normal	$E(\epsilon) = 0, \text{var}(\epsilon) = 1$
Logistic	$E(\epsilon) = 0, \text{var}(\epsilon) = \pi^2/3$
Extreme value (Type I)	$E(\epsilon) \approx 0.5772$ (Euler's constant), $\text{var}(\epsilon) = \pi^2/6$

Specifying the Censoring Points

You must also provide information about the censoring points of the dependent variable. There are two cases to consider: (1) where the limit points are known for all individuals, and (2) where the censoring is by indicator and the limit points are known only for individuals with censored observations.

Limit Points Known

You should enter an expressions for the left and right censoring points in the edit fields as required. Note that if you leave an edit field blank, EViews will assume that there is no censoring of observations of that type.

For example, in the canonical tobit model the data are censored on the left at zero, and are uncensored on the right. This case may be specified as:

```
Left edit field:    0
Right edit field:  [blank]
```

Similarly, top-coded censored data may be specified as:

```
Left edit field:  [blank]
Right edit field: 20000
```


while the more general case of left and right censoring is given by:

Left edit field: 10000

Right edit field: 20000

EViews also allows more general specifications where the censoring points are known but differ across observations. Simply enter the name of the series or auto-series containing the censoring points in the appropriate edit field. For example,

Left edit field: lowinc

Right edit field: vcens1+10

specifies a model with LOWINC censoring on the left-hand side, and right censoring at the value of VCENS1 + 10.

Limit Points Not Known

In some cases, the hypothetical censoring point is unknown for some individuals (c_i and \bar{c}_i are not observed for all observations). This situation often occurs with data where censoring is indicated with a zero-one dummy variable, but no additional information is provided about potential censoring points.

EViews provides you an alternative method of describing data censoring that matches this format. Simply select the **Field is zero/one indicator of censoring** option in the estimation dialog, and enter the series expression for the censoring indicator(s) in the appropriate edit field(s). Observations with a censoring indicator of one are assumed to be censored while those with a value of zero are assumed to be actual responses.

For example, suppose that we have observations on the length of time that an individual has been unemployed (U), but that some of these observations represent ongoing unemployment at the time the sample is taken. These latter observations may be treated as right censored at the reported value. If the variable RCENS is a dummy variable representing censoring, you can click on the **Field is zero/one indicator of censoring** setting and enter:

Left edit field: [blank]

Right edit field: rcens

in the edit fields. If the data are censored on both the left and the right, use separate binary indicators for each form of censoring:

Left edit field: lcens

Right edit field: rcens

where LCENS is also a binary indicator.

Once you have specified the model, click **OK**. EViews will estimate the parameters of the model using appropriate iterative techniques.

A Comparison of Censoring Methods

An alternative to specifying index censoring is to enter a very large positive or negative value for the censoring limit for non-censored observations. For example, you could enter `-1e-100` and `1e100` as the censoring limits for an observation on a completed unemployment spell. In fact, any limit point that is “outside” the observed data will suffice.

While this latter approach will yield the same likelihood function and therefore the same parameter values and coefficient covariance matrix, there is a drawback to the artificial data approach. The presence of a censoring value implies that you can evaluate the conditional mean of the observed dependent variable, as well as the ordinary and standardized residuals. All of the calculations that use residuals will, however, be based upon the arbitrary artificial data and will be invalid.

If you specify your censoring by index, you are telling EViews that you do not have information about the censoring for those observations that are not censored. And if an observation is left censored, you may not have information about the right censoring limit. In these circumstances, you should specify your censoring by index so that EViews will prevent you from computing the conditional mean of the dependent variable and the associated residuals.

Interpreting the Output

If your model converges, EViews will display the estimation results in the equation window. The first part of the table presents the usual header information, including information about the assumed error distribution, estimation sample, estimation algorithms, and number of iterations required for convergence.

EViews also provides information about the specification for the censoring. If the estimated model is the canonical tobit with left-censoring at zero, EViews will label the method as a TOBIT. For all other censoring methods, EViews will display detailed information about form of the left and/or right censoring.

Here, we have the header output from a left censored model where the censoring is specified by value:

```
Dependent Variable: Y_PT
Method: ML - Censored Normal (TOBIT)
Date: 09/14/97   Time: 08:27
Sample: 1 601
Included observations: 601
Convergence achieved after 8 iterations
Covariance matrix computed using second
derivatives
```

Below the header are the usual results for the coefficients, including the asymptotic standard errors, z -statistics, and significance levels. As in other limited dependent variable models, the estimated coefficients do not have a direct interpretation as the marginal effect of the associated regressor j for individual i , x_{ij} . In censored regression models, a change in x_{ij} has two effects: an effect on the mean of y , given that it is observed, and an effect on the probability of y being observed (see McDonald and Moffitt, 1980).

In addition to results for the regression coefficients, EViews reports an additional coefficient named SCALE, which is the estimated scale factor σ . This scale factor may be used to estimate the standard deviation of the residual, using the known variance of the assumed distribution. For example, if the estimated SCALE has a value of 0.446 for a model with extreme value errors, the implied standard error of the error term is $0.5977 = 0.466\pi/\sqrt{6}$.

Most of the other output is self-explanatory. As in the binary and ordered models above, EViews reports summary statistics for the dependent variable and likelihood based statistics. The regression statistics at the bottom of the table are computed in the usual fashion, using the residuals $\hat{\epsilon}_i = y_i - E(y_i|x_i, \hat{\beta}, \hat{\sigma})$ from the observed y .

Views of Censored Equations

Most of the views that are available for a censored regression are familiar from other settings. The residuals used in the calculations are defined below.

The one new view is the **Categorical Regressor Stats** view, which presents means and standard deviations for the dependent and independent variables for the estimation sample. EViews provides statistics computed over the entire sample, as well as for the left censored, right censored and non-censored individuals.

Procedures for Censored Equations

EViews provides several procedures which provide access to information derived from your censored equation estimates.

Make Residual Series

Select **Procs/Make Residual Series**, and select from among the three types of residuals. The three types of residuals for censored models are defined as:

Ordinary	$e_{oi} = y_i - E(y_i x_i, \hat{\beta}, \hat{\sigma})$
Standardized	$e_{si} = \frac{y_i - E(y_i x_i, \hat{\beta}, \hat{\sigma})}{\sqrt{\text{var}(y_i x_i, \hat{\beta}, \hat{\sigma})}}$
Generalized	$e_{gi} = -\frac{f((c_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma F((c_i - x_i'\hat{\beta})/\hat{\sigma})} \cdot 1(y_i \leq c_i)$ $-\frac{f'((c_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma F((c_i - x_i'\hat{\beta})/\hat{\sigma})} \cdot 1(c_i < y_i \leq \bar{c}_i)$ $+\frac{f((\bar{c}_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma(1-F((\bar{c}_i - x_i'\hat{\beta})/\hat{\sigma}))} \cdot 1(y_i \leq \bar{c}_i)$

where f , F are the density and distribution functions, and where 1 is an indicator function which takes the value 1 if the condition in parentheses is true, and 0 if it is false. All of the above terms will be evaluated at the estimated β and σ . See the discussion of forecasting for details on the computation of $E(y_i|x_i, \beta, \sigma)$.

The generalized residuals may be used as the basis of a number of LM tests, including LM tests of normality (see Lancaster, Chesher and Irish (1985), Chesher and Irish (1987), and Gourioux, Monfort, Renault and Trognon (1987); Greene (1997), provides a brief discussion and additional references).

Forecasting

EViews provides you with the option of forecasting the expected dependent variable, $E(y_i|x_i, \beta, \sigma)$, or the expected latent variable, $E(y_i^*|x_i, \beta, \sigma)$. Select **Forecast** from the equation toolbar to open the forecast dialog.

To forecast the expected *latent variable*, click on **Index - Expected latent variable**, and enter a name for the series to hold the output. The forecasts of the expected latent variable $E(y_i^*|x_i, \beta, \sigma)$ may be derived from the latent model using the relationship

$$\hat{y}_i^* = E(y_i^*|x_i, \hat{\beta}, \hat{\sigma}) = x_i'\hat{\beta}. \quad (17.29)$$

To forecast the expected *observed dependent variable*, you should select **Expected dependent variable**, and enter a series name. These forecasts are computed using the relationship:

$$\begin{aligned} \hat{y}_i &= E(y_i|x_i, \hat{\beta}, \hat{\sigma}) = \underline{c}_i \cdot \Pr(y_i = \underline{c}_i|x_i, \hat{\beta}, \hat{\sigma}) \\ &+ E(y_i^*|\underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma}) \cdot \Pr(\underline{c}_i < y_i^* < \bar{c}_i|x_i, \hat{\beta}, \hat{\sigma}) \\ &+ \bar{c}_i \cdot \Pr(y_i = \bar{c}_i|x_i, \hat{\beta}, \hat{\sigma}) \end{aligned} \tag{17.30}$$

Note that these forecasts always satisfy $\underline{c}_i \leq \hat{y}_i \leq \bar{c}_i$. The probabilities associated with being in the various classifications are computed by evaluating the cumulative distribution function of the specified distribution. For example, the probability of being at the lower limit is given by:

$$\Pr(y_i = \underline{c}_i|x_i, \hat{\beta}, \hat{\sigma}) = \Pr(y_i^* \leq \underline{c}_i|x_i, \hat{\beta}, \hat{\sigma}) = F((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma}). \tag{17.31}$$

Censored Model Illustration

As an example, we replicate Fair’s (1978) tobit model that estimates the incidence of extramarital affairs. The dependent variable, number of extramarital affairs (Y_PT), is left censored at zero and the errors are assumed to be normally distributed. The bottom portion of the output is presented below:

	Coefficient	Std. Error	z-Statistic	Prob.
C	7.608487	3.905837	1.947979	0.0514
Z1	0.945787	1.062824	0.889881	0.3735
Z2	-0.192698	0.080965	-2.380015	0.0173
Z3	0.533190	0.146602	3.636997	0.0003
Z4	1.019182	1.279524	0.796532	0.4257
Z5	-1.699000	0.405467	-4.190231	0.0000
Z6	0.025361	0.227658	0.111399	0.9113
Z7	0.212983	0.321145	0.663198	0.5072
Z8	-2.273284	0.415389	-5.472657	0.0000
Error		Distribution		
SCALE:C(10)	8.258432	0.554534	14.89256	0.0000
R-squared	0.151569	Mean dependent var		1.455907
Adjusted R-squared	0.138649	S.D. dependent var		3.298758
S.E. of regression	3.061544	Akaike info criterion		2.378473
Sum squared resid	5539.472	Schwarz criterion		2.451661
Log likelihood	-704.7311	Hannan-Quinn criter.		2.406961
Avg. log likelihood	-1.172597			
Left censored obs	451	Right censored obs		0
Uncensored obs	150	Total obs		601

Tests of Significance

EViews does not, by default, provide you with the usual likelihood ratio test of the overall significance for the tobit and other censored regression models. There are several ways to perform this test (or an asymptotically equivalent test).

First, you can use the built-in coefficient testing procedures to test the exclusion of all of the explanatory variables. Select the redundant variables test and enter the names of all of the explanatory variables you wish to exclude. EViews will compute the appropriate likelihood ratio test statistic and the p -value associated with the statistic.

To take an example, suppose we wish to test whether the variables in the Fair tobit, above, contribute to the fit of the model. Select **View/Coefficient Tests/Redundant Variables - Likelihood Ratio...** and enter all of the explanatory variables:

```
z1 z2 z3 z4 z5 z6 z7 z8
```

EViews will estimate the restricted model for you and compute the LR statistic and p -value. In this case, the value of the test statistic is 80.01, which for eight degrees-of-freedom, yields a p -value of less than 0.000001.

Alternatively, you could test the restriction using the Wald test by selecting **View/Coefficient Tests/Wald - Coefficient Restrictions...**, and entering the restriction that:

```
c (2) =c (3) =c (4) =c (5) =c (6) =c (7) =c (8) =c (9) =0
```

The reported statistic is 68.14, with a p -value of less than 0.000001.

Lastly, we demonstrate the direct computation of the LR test. Suppose the Fair tobit model estimated above is saved in the named equation EQ_TOBIT. Then you could estimate an equation containing only a constant, say EQ_RESTR, and place the likelihood ratio statistic in a scalar:

```
scalar lrstat=-2*(eq_restr.@logl-eq_tobit.@logl)
```

Next, evaluate the chi-square probability associated with this statistic:

```
scalar lrprob=1-@cchisq(lrstat, 8)
```

with degrees of freedom given by the number of coefficient restrictions in the constant only model. You can double click on the LRSTAT icon or the LRPROB icon in the workfile window to display the results in the status line.

A Specification Test for the Tobit

As a rough diagnostic check, Pagan and Vella (1989) suggest plotting Powell's (1986) symmetrically trimmed residuals. If the error terms have a symmetric distribution centered at zero (as assumed by the normal distribution), so should the trimmed residuals. To construct the trimmed residuals, first save the forecasts of the index (expected latent variable): click **Forecast**, choose **Index-Expected latent variable**, and provide a name for the fitted index, say XB. The trimmed residuals are obtained by dropping observations for which $x_i' \hat{\beta} < 0$, and replacing y_i with $2(x_i' \hat{\beta})$ for all observations where $y_i < 2(x_i' \hat{\beta})$. The trimmed residuals RES_T can be obtained by the commands

```

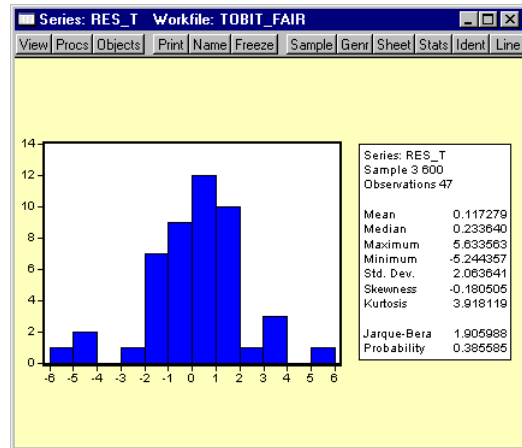
series res_t=(y_pt<=2*xb)*(y_pt-xb) +(y_pt>2*xb)*xb
smpl if xb<0
series res_t=na
smpl @all

```

The histogram of the trimmed residual is depicted below.

This example illustrates the possibility that the number of observations that are lost by trimming can be quite large; out of the 601 observations in the sample, only 47 observations are left after trimming.

The tobit model imposes the restriction that the coefficients that determine the probability of being censored are the same as those that determine the conditional mean of the uncensored observations. To test this restriction, we carry out the LR test by comparing the (restricted) tobit to the unrestricted log likelihood that is the sum of a probit and a truncated regression (we discuss truncated regression in detail in the following section). Save the tobit equation in the workfile by pressing the **Name** button, and enter a name, say EQ_TOBIT.



To estimate the probit, first create a dummy variable indicating uncensored observations by the command

```
series y_c=(y_pt>0)
```

Then estimate a probit by replacing the dependent variable Y_PT by Y_C. A simple way to do this is to press **Objects/Copy Object...** from the tobit equation toolbar. From the new untitled equation window that appears, press **Estimate**, replace the dependent variable with Y_C and choose **Method: BINARY** and click **OK**. Save the probit equation by pressing the **Name** button, say as EQ_BIN.

To estimate the truncated model, press **Objects/Copy Object...** from the tobit equation toolbar again. From the new untitled equation window that appears, press **Estimate**, mark the **Truncated sample** option, and click **OK**. Save the truncated regression by pressing the **Name** button, say as EQ_TR.

Then the LR test statistic and its *p*-value can be saved as a scalar by the commands

```
scalar lr_test=2*(eq_bin.@logl+eq_tr.@logl-eq_tobit.@logl)
```

```
scalar lr_pval=1-@cchisq(lr_test,eq_tobit.@ncoef)
```

Double click on the scalar name to display the value in the status line at the bottom of the EViews window. For the example data set, the p -value is 0.066 which rejects the tobit model at the 10% level but not at the 5% level.

For other specification tests for the tobit, see Greene (1997, 20.3.4) or Pagan and Vella (1989).

Truncated Regression Models

A close relative of the censored regression model is the truncated regression model. Suppose that an observation is not observed whenever the dependent variable falls below one threshold, or exceeds a second threshold. This sampling rule occurs, for example, in earnings function studies for low-income families that exclude observations with incomes above a threshold, and in studies of durables demand among individuals who purchase durables.

The general two-limit truncated regression model may be written as:

$$y_i^* = x_i' \beta + \epsilon_i \quad (17.32)$$

where $y_i = y_i^*$ is only observed if:

$$c_i < y_i^* < \bar{c}_i. \quad (17.33)$$

If there is no lower truncation, then we can set $c_i = -\infty$. If there is no upper truncation, then we set $\bar{c}_i = \infty$.

The log likelihood function associated with these data is given by

$$l(\beta, \sigma) = \sum_{i=1}^N \log f((y_i - x_i' \beta) / \sigma) \cdot 1(c_i < y_i < \bar{c}_i) - \log(F((\bar{c}_i - x_i' \beta) / \sigma) - F((c_i - x_i' \beta) / \sigma)). \quad (17.34)$$

The likelihood function is maximized with respect to β and σ , using standard iterative methods.

Estimating a Truncated Model in EViews

Estimation of a truncated regression model follows the same steps as estimating a censored regression:

- Select **Quick/Estimate Equation...** from the main menu, and in the Equation Specification dialog, select the **CENSORED** estimation method. The censored and truncated regression dialog will appear.

- Enter the name of the truncated dependent variable and the list of the regressors in the **Equation Specification** field, and select one of the three distributions for the error term. You must enter your specification by list.
- Indicate that you wish to estimate the truncated model by checking the **Truncated sample** option.
- Specify the truncation points of the dependent variable by entering the appropriate expressions in the two edit fields. If you leave an edit field blank, EViews will assume that there is no truncation along that dimension.

You should keep a few points in mind. First, truncated estimation is only available for models where the truncation points are known, since the likelihood function is not otherwise defined. If you attempt to specify your truncation points by index, EViews will issue an error message indicating that this selection is not available.

Second, EViews will issue an error message if any values of the dependent variable are outside the truncation points. Furthermore, EViews will automatically exclude any observations that are exactly equal to a truncation point. Thus, if you specify zero as the lower truncation limit, EViews will issue an error message if any observations are less than zero, and will exclude any observations where the dependent variable exactly equals zero.

The cumulative distribution function and density of the assumed distribution will be used to form the likelihood function, as described above.

Procedures for Truncated Equations

EViews provides the same procedures for truncated equations as for censored equations. The residual and forecast calculations differ to reflect the truncated dependent variable and the different likelihood function.

Make Residual Series

Select **Procs/Make Residual Series**, and select from among the three types of residuals. The three types of residuals for censored models are defined as:

Ordinary	$e_{oi} = y_i - E(y_i^* \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})$
Standardized	$e_{si} = \frac{y_i - E(y_i^* \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})}{\sqrt{\text{var}(y_i^* \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})}}$
Generalized	$e_{gi} = - \frac{f'((y_i - x_i' \hat{\beta}) / \hat{\sigma})}{\sigma f((y_i - x_i' \hat{\beta}) / \hat{\sigma})} - \frac{f((\bar{c}_i - x_i' \hat{\beta}) / \hat{\sigma}) - f((\underline{c}_i - x_i' \hat{\beta}) / \hat{\sigma})}{\sigma (F((\bar{c}_i - x_i' \hat{\beta}) / \hat{\sigma}) - F((\underline{c}_i - x_i' \hat{\beta}) / \hat{\sigma}))}$

where f , F , are the density and distribution functions. Details on the computation of $E(y_i | \underline{c}_i < y_i < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})$ are provided below.

The generalized residuals may be used as the basis of a number of LM tests, including LM tests of normality (see Chesher and Irish (1984, 1987), and Gourieroux, Monfort and Trognon (1987); Greene (1997) provides a brief discussion and additional references).

Forecasting

EViews provides you with the option of forecasting the expected observed dependent variable, $E(y_i | x_i, \hat{\beta}, \hat{\sigma})$, or the expected latent variable, $E(y_i^* | x_i, \hat{\beta}, \hat{\sigma})$.

To forecast the expected latent variable, select **Forecast** from the equation toolbar to open the forecast dialog, click on **Index - Expected latent variable**, and enter a name for the series to hold the output. The forecasts of the expected latent variable $E(y_i^* | x_i, \hat{\beta}, \hat{\sigma})$ are computed using

$$\hat{y}_i^* = E(y_i^* | x_i, \hat{\beta}, \hat{\sigma}) = x_i' \hat{\beta}. \quad (17.35)$$

To forecast the expected observed dependent variable for the truncated model, you should select **Expected dependent variable**, and enter a series name. These forecasts are computed using:

$$\hat{y}_i = E(y_i^* | \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma}) \quad (17.36)$$

so that the expectations for the latent variable are taken with respect to the conditional (on being observed) distribution of the y_i^* . Note that these forecasts always satisfy the inequality $\underline{c}_i < \hat{y}_i < \bar{c}_i$.

It is instructive to compare this latter expected value with the expected value derived for the censored model in [Equation \(17.30\)](#) above (repeated here for convenience):

$$\begin{aligned} \hat{y}_i &= E(y_i|x_i, \hat{\beta}, \hat{\sigma}) = \underline{c}_i \cdot \Pr(y_i = \underline{c}_i|x_i, \hat{\beta}, \hat{\sigma}) \\ &+ E(y_i^*|\underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma}) \cdot \Pr(\underline{c}_i < y_i^* < \bar{c}_i|x_i, \hat{\beta}, \hat{\sigma}) \\ &+ \bar{c}_i \cdot \Pr(y_i = \bar{c}_i|x_i, \hat{\beta}, \hat{\sigma}). \end{aligned} \tag{17.37}$$

The expected value of the dependent variable for the truncated model is the first part of the middle term of the censored expected value. The differences between the two expected values (the probability weight and the first and third terms) reflect the different treatment of latent observations that do not lie between \underline{c}_i and \bar{c}_i . In the censored case, those observations are included in the sample and are accounted for in the expected value. In the truncated case, data outside the interval are not observed and are not used in the expected value computation.

Illustration

As an example, we reestimate the Fair tobit model from above, truncating the data so that observations at or below zero are removed from the sample. The output from truncated estimation of the Fair model is presented below:

```

Dependent Variable: Y_PT
Method: ML - Censored Normal (TOBIT)
Date: 10/13/97   Time: 22:45
Sample(adjusted): 452 601
Included observations: 150 after adjusting endpoints
Truncated sample
Left censoring (value) at zero
Convergence achieved after 8 iterations
Covariance matrix computed using second derivatives

```

	Coefficient	Std. Error	z-Statistic	Prob.
C	12.37288	5.178306	2.389368	0.0169
Z1	-1.336872	1.453133	-0.919993	0.3576
Z2	-0.044792	0.116141	-0.385670	0.6997
Z3	0.544182	0.220119	2.472218	0.0134
Z4	-2.142896	1.787720	-1.198675	0.2307
Z5	-1.423128	0.600472	-2.370014	0.0178
Z6	-0.316721	0.322327	-0.982609	0.3258
Z7	0.621428	0.478827	1.297813	0.1944
Z8	-1.210037	0.552131	-2.191578	0.0284

Error Distribution				
SCALE:C(10)	5.379557	0.688875	7.809196	0.0000
R-squared	0.654664	Mean dependent var		1.455907
Adjusted R-squared	0.649405	S.D. dependent var		3.298758
S.E. of regression	1.953229	Akaike info criterion		1.333891
Sum squared resid	2254.726	Schwarz criterion		1.407079
Log likelihood	-390.8342	Hannan-Quinn criter.		1.362379
Avg. log likelihood	-0.650306			
Left censored obs	0	Right censored obs		0
Uncensored obs	150	Total obs		150

Note that the header information indicates that the model is a truncated specification, and that the sample information at the bottom of the screen shows that there are no left and right censored observations.

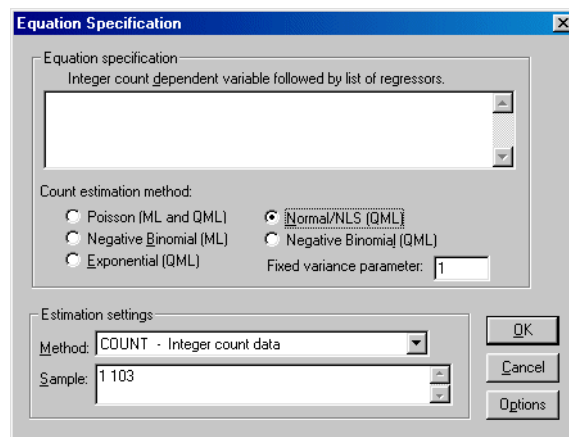
Count Models

Count models are employed when y takes integer values that represent the number of events that occur—examples of count data include the number of patents filed by a company, and the number of spells of unemployment experienced over a fixed time interval.

EViews provides support for the estimation of several models of count data. In addition to the standard poisson and negative binomial maximum likelihood (ML) specifications, EViews provides a number of quasi-maximum likelihood (QML) estimators for count data.

Estimating Count Models in EViews

To estimate a count data model, select **Quick/Estimate Equation...** from the main menu, and select **COUNT** as the estimation method. EViews displays the count estimation dialog into which you will enter the dependent and explanatory variable regressors, select a type of count model, and if desired, set estimation options.



There are three parts to the specification of the count model:

- In the upper edit field, you should list the dependent variable and the independent variables. You must specify your model by list. The list of explanatory variables specifies a model for the conditional mean of the dependent variable:

$$m(x_i, \beta) = E(y_i | x_i, \beta) = \exp(x_i' \beta). \quad (17.38)$$

- Next, click on **Options** and, if desired, change the default estimation algorithm, convergence criterion, starting values, and method of computing the coefficient covariance.
- Lastly, select one of the entries listed under count estimation method, and if appropriate, specify a value for the variance parameter. Details for each method are provided in the following discussion.

Poisson Model

For the Poisson model, the conditional density of y_i given x_i is

$$f(y_i|x_i, \beta) = \frac{e^{-m(x_i, \beta)} m(x_i, \beta)^{y_i}}{y_i!} \quad (17.39)$$

where y_i is a non-negative integer valued random variable. The maximum likelihood estimator (MLE) of the parameter β is obtained by maximizing the log likelihood function

$$l(\beta) = \sum_{i=1}^N y_i \log m(x_i, \beta) - m(x_i, \beta) - \log(y_i!). \quad (17.40)$$

Provided the conditional mean function is correctly specified and the conditional distribution of y is Poisson, the MLE $\hat{\beta}$ is consistent, efficient, and asymptotically normally distributed, with variance matrix consistently estimated by

$$V = \text{var}(\hat{\beta}) = \left(\sum_{i=1}^N \left(\frac{\partial \hat{m}_i}{\partial \beta} \frac{\partial \hat{m}_i}{\partial \beta'} \right) / \hat{m}_i \right)^{-1} \quad (17.41)$$

where $\hat{m}_i = m(x_i, \hat{\beta})$.

The Poisson assumption imposes restrictions that are often violated in empirical applications. The most important restriction is the equality of the (conditional) mean and variance:

$$v(x_i, \beta) = \text{var}(y_i|x_i, \beta) = \text{E}(y_i|x_i, \beta) = m(x_i, \beta). \quad (17.42)$$

If the mean-variance equality does not hold, the model is misspecified. EViews provides a number of other estimators for count data which relax this restriction.

We note here that the Poisson estimator may also be interpreted as a quasi-maximum likelihood estimator. The implications of this result are discussed below.

Negative Binomial (ML)

One common alternative to the Poisson model is to estimate the parameters of the model using maximum likelihood of a negative binomial specification. The log likelihood for the negative binomial distribution is given by

$$\begin{aligned}
l(\beta, \eta) &= \sum_{i=1}^N y_i \log(\eta^2 m(x_i, \beta)) \\
&- (y_i + 1/\eta^2) \log(1 + \eta^2 m(x_i, \beta)) \\
&+ \log \Gamma(y_i + 1/\eta^2) - \log(y_i!) - \log \Gamma(1/\eta^2)
\end{aligned} \tag{17.43}$$

where η^2 is a variance parameter to be jointly estimated with the conditional mean parameters β . EVIEWS estimates the log of η^2 , and labels this parameter as the “SHAPE” parameter in the output. Standard errors are computed using the inverse of the information matrix.

The negative binomial distribution is often used when there is *overdispersion* in the data, so that $v(x_i, \beta) > m(x_i, \beta)$, since the following moment conditions hold:

$$\begin{aligned}
E(y_i | x_i, \beta) &= m(x_i, \beta) \\
\text{var}(y_i | x_i, \beta) &= m(x_i, \beta)(1 + \eta^2 m(x_i, \beta))
\end{aligned} \tag{17.44}$$

η^2 is therefore a measure of the extent to which the conditional variance exceeds the conditional mean.

Consistency and efficiency of the negative binomial ML requires that the conditional distribution of y be negative binomial.

Quasi-maximum Likelihood (QML)

We can perform maximum likelihood estimation under a number of alternative distributional assumptions. These *quasi-maximum likelihood (QML) estimators* are robust in the sense that they produce consistent estimates of the parameters of a correctly specified conditional mean, even if the distribution is incorrectly specified.

This robustness result is exactly analogous to the situation in ordinary regression, where the normal ML estimator (least squares) is consistent, even if the underlying error distribution is not normally distributed. In ordinary least squares, all that is required for consistency is a correct specification of the conditional mean $m(x_i, \beta) = x_i' \beta$. For QML count models, all that is required for consistency is a correct specification of the conditional mean $m(x_i, \beta)$.

The estimated standard errors computed using the inverse of the information matrix will not be consistent unless the conditional distribution of y is correctly specified. However, it is possible to estimate the standard errors in a robust fashion so that we can conduct valid inference, even if the distribution is incorrectly specified.

EViews provides options to compute two types of robust standard errors. Click **Options** in the Equation Specification dialog box and mark the **Robust Covariance** option. The **Huber/White** option computes QML standard errors, while the **GLM** option computes standard errors corrected for overdispersion. See “[Technical Notes](#)” on page 467 for details on these options.

Further details on QML estimation are provided by Gourioux, Monfort, and Trognon (1994a, 1994b). Wooldridge (1996) provides an excellent summary of the use of QML techniques in estimating parameters of count models. See also the extensive related literature on Generalized Linear Models (McCullagh and Nelder, 1989).

Poisson

The Poisson MLE is also a QMLE for data from alternative distributions. Provided that the conditional mean is correctly specified, it will yield consistent estimates of the parameters β of the mean function. By default, EViews reports the ML standard errors. If you wish to compute the QML standard errors, you should click on **Options**, select **Robust Covariances**, and select the desired covariance matrix estimator.

Exponential

The log likelihood for the exponential distribution is given by

$$l(\beta) = \sum_{i=1}^N -\log m(x_i, \beta) - y_i / (m(x_i, \beta)). \quad (17.45)$$

As with the other QML estimators, the exponential QMLE is consistent even if the conditional distribution of y_i is not exponential, provided that m_i is correctly specified. By default, EViews reports the robust QML standard errors.

Normal

The log likelihood for the normal distribution is

$$l(\beta) = \sum_{i=1}^N -\frac{1}{2} \left(\frac{y_i - m(x_i, \beta)}{\sigma} \right)^2 - \frac{1}{2} \log(\sigma^2) - \frac{1}{2} \log(2\pi). \quad (17.46)$$

For *fixed* σ^2 and correctly specified m_i , maximizing the normal log likelihood function provides consistent estimates even if the distribution is not normal. Note that maximizing the normal log likelihood for a fixed σ^2 is equivalent to minimizing the sum of squares for the nonlinear regression model:

$$y_i = m(x_i, \beta) + \epsilon_i. \quad (17.47)$$

EViews sets $\sigma^2 = 1$ by default. You may specify any other (positive) value for σ^2 by changing the number in the **Fixed variance parameter** field box. By default, EViews reports the robust QML standard errors when estimating this specification.

Negative Binomial

If we maximize the negative binomial log likelihood, given above, for *fixed* η^2 , we obtain the QMLE of the conditional mean parameters β . This QML estimator is consistent even if the conditional distribution of y is not negative binomial, provided that m_i is correctly specified.

EViews sets $\eta^2 = 1$ by default, which is a special case known as the geometric distribution. You may specify any other (positive) value by changing the number in the **Fixed variance parameter** field box. For the negative binomial QMLE, EViews by default reports the robust QMLE standard errors.

Views of Count Models

EViews provides a full complement of views of count models. You can examine the estimation output, compute frequencies for the dependent variable, view the covariance matrix, or perform coefficient tests. Additionally, you can select **View/Actual, Fitted, Residual...** and pick from a number of views describing the ordinary residuals $e_{oi} = y_i - m(x_i, \hat{\beta})$, or you can examine the correlogram and histogram of these residuals. For the most part, all of these views are self-explanatory.

Note, however, that the LR test statistics presented in the summary statistics at the bottom of the equation output, or as computed under the **View/Coefficient Tests/Redundant Variables - Likelihood Ratio...** have a known asymptotic distribution only if the conditional distribution is correctly specified. Under the weaker GLM assumption that the true variance is proportional to the nominal variance, we can form a quasi-likelihood ratio, $QLR = LR/\hat{\sigma}^2$, where $\hat{\sigma}^2$ is the estimated proportional variance factor. This QLR statistic has an asymptotic χ^2 distribution under the assumption that the mean is correctly specified and that the variances follow the GLM structure. EViews does not compute the QLR statistic, but it can be estimated by computing an estimate of $\hat{\sigma}^2$ based upon the standardized residuals. We provide an example of the use of the QLR test statistic below.

If the GLM assumption does not hold, then there is no usable QLR test statistic with a known distribution; see Wooldridge (1996).

Procedures for Count Models

Most of the procedures are self-explanatory. Some details are required for the forecasting and residual creation procedures.

- **Forecast...** provides you the option to forecast the dependent variable y_i or the predicted linear index $x_i\beta$. Note that for all of these models the forecasts of y_i are given by $\hat{y}_i = m(x_i, \hat{\beta})$ where $m(x_i, \hat{\beta}) = \exp(x_i\hat{\beta})$.

- **Make Residual Series...** provides the following three types of residuals for count models:

Ordinary	$e_{oi} = y_i - m(x_i, \hat{\beta})$
Standardized (Pearson)	$e_{si} = \frac{y_i - m(x_i, \hat{\beta})}{\sqrt{v(x_i, \hat{\beta}, \hat{\gamma})}}$
Generalized	$e_g = (\text{varies})$

where the γ represents any additional parameters in the variance specification. Note that the specification of the variances may vary significantly between specifications. For example, the Poisson model has $v(x_i, \hat{\beta}) = m(x_i, \hat{\beta})$, while the exponential has $v(x_i, \hat{\beta}) = m(x_i, \hat{\beta})^2$.

The generalized residuals can be used to obtain the score vector by multiplying the generalized residuals by each variable in x . These scores can be used in a variety of LM or conditional moment tests for specification testing; see Wooldridge (1996).

Demonstrations

A Specification Test for Overdispersion

Consider the model

$$\text{NUMB}_i = \beta_1 + \beta_2 \text{IP}_i + \beta_3 \text{FEB}_i + \epsilon_i, \quad (17.48)$$

where the dependent variable NUMB is the number of strikes, IP is a measure of industrial production, and FEB is a February dummy variable, as reported in Kennan (1985, Table 1).

The results from Poisson estimation of this model are presented below:

Dependent Variable: NUMB
 Method: ML/QML - Poisson Count
 Date: 09/14/97 Time: 10:58
 Sample: 1 103
 Included observations: 103

Convergence achieved after 4 iterations

Covariance matrix computed using second derivatives

Variable	Coefficient	Std. Error	z-Statistic	Prob.
C	1.725630	0.043656	39.52764	0.0000
IP	2.775334	0.819104	3.388254	0.0007
FEB	-0.377407	0.174520	-2.162539	0.0306
R-squared	0.064502	Mean dependent var		5.495146
Adjusted R-squared	0.045792	S.D. dependent var		3.653829
S.E. of regression	3.569190	Akaike info criterion		5.583421
Sum squared resid	1273.912	Schwarz criterion		5.660160
Log likelihood	-284.5462	Hannan-Quinn criter.		5.614503
Restr. log likelihood	-292.9694	Avg. log likelihood		-2.762584
LR statistic (2 df)	16.84645	LR index (Pseudo-R2)		0.028751
Probability(LR stat)	0.000220			

Cameron and Trivedi (1990) propose a regression based test of the Poisson restriction $v(x_i, \beta) = m(x_i, \beta)$. To carry out the test, first estimate the Poisson model and obtain the fitted values of the dependent variable. Click **Forecast** and provide a name for the forecasted dependent variable, say NUMB_F. The test is based on an auxiliary regression of $e_{oi}^2 - y_i$ on \hat{y}_i^2 and testing the significance of the regression coefficient. For this example, the test regression can be estimated by the command

```
ls (numb-numb_f)^2-numb numb_f^2
```

yielding the following results:

Dependent Variable: (NUMB-NUMB_F)^2-NUMB
 Method: Least Squares
 Date: 09/14/97 Time: 11:05
 Sample: 1 103
 Included observations: 103

Variable	Coefficient	Std. Error	t-Statistic	Prob.
NUMB F^2	0.238874	0.052115	4.583571	0.0000
R-squared	0.043930	Mean dependent var		6.872929
Adjusted R-squared	0.043930	S.D. dependent var		17.65726
S.E. of regression	17.26506	Akaike info criterion		8.544908
Sum squared resid	30404.41	Schwarz criterion		8.570488
Log likelihood	-439.0628	Durbin-Watson stat		1.711805

The t -statistic of the coefficient is highly significant, leading us to reject the Poisson restriction. Moreover, the estimated coefficient is significantly positive, indicating overdispersion in the residuals.

An alternative approach, suggested by Wooldridge (1996) is to regress $e_{si} - 1$, on \hat{y}_i . To perform this test, select **Procs/Make Residual Series...** and select **Standardized**. Save the results in a series, say SRESID. Then estimating the regression specification:

```
sresid^2-1 numbf
```

yields the results:

Dependent Variable: SRESID^2-1
 Method: Least Squares
 Date: 10/06/97 Time: 16:05
 Sample: 1 103
 Included observations: 103

Variable	Coefficient	Std. Error	t-Statistic	Prob.
NUMBF	0.221238	0.055002	4.022326	0.0001
R-squared	0.017556	Mean dependent var		1.161573
Adjusted R-squared	0.017556	S.D. dependent var		3.138974
S.E. of regression	3.111299	Akaike info criterion		5.117619
Sum squared resid	987.3786	Schwarz criterion		5.143199
Log likelihood	-262.5574	Durbin-Watson stat		1.764537

Both tests suggest the presence of overdispersion, with the variance approximated by $v = m(1 + 0.23m)$.

Given the evidence of overdispersion and the rejection of the Poisson restriction, we will re-estimate the model, allowing for mean-variance inequality. Our approach will be to estimate the two-step negative binomial QMLE specification (termed the *quasi-generalized pseudo-maximum likelihood estimator* by Gourieroux, Monfort, and Trognon (1984a, b)) using the estimate of $\hat{\eta}^2$ derived above. To compute this estimator, simply select **Negative Binomial (QML)** and enter 0.22124 in the edit field for **Fixed variance parameter**.

We will use the GLM variance calculations, so you should click on **Option** in the Equation Specification dialog and mark the **Robust Covariance** and **GLM** options. The estimation results are shown below:

Dependent Variable: NUMB
 Method: QML - Negative Binomial Count
 Date: 10/11/97 Time: 23:53
 Sample: 1 103
 Included observations: 103
 QML parameter used in estimation: 0.22124
 Convergence achieved after 3 iterations
 GLM Robust Standard Errors & Covariance
 Variance factor estimate = 2.465660162
 Covariance matrix computed using second derivatives

Variable	Coefficient	Std. Error	z-Statistic	Prob.
C	1.724906	0.102543	16.82135	0.0000
IP	2.833103	1.919447	1.475999	0.1399
FEB	-0.369558	0.377376	-0.979285	0.3274
R-squared	0.064374	Mean dependent var		5.495146
Adjusted R-squared	0.045661	S.D. dependent var		3.653829
S.E. of regression	3.569435	Akaike info criterion		5.174385
Sum squared resid	1274.087	Schwarz criterion		5.251125
Log likelihood	-263.4808	Hannan-Quinn criter.		5.205468
Restr. log likelihood	-522.9973	Avg. log likelihood		-2.558066
LR statistic (2 df)	519.0330	LR index (Pseudo-R2)		0.496210
Probability(LR stat)	0.000000			

The header indicates that the estimated GLM variance factor is 2.4, suggesting that the negative binomial ML would not have been an appropriate specification. Nevertheless, the negative binomial QML should be consistent, and under the GLM assumption, the standard errors should be consistently estimated. It is worth noting that the coefficients on IP and FEB, which were strongly statistically significant in the Poisson specification, are no longer significantly different from zero at conventional significance levels.

Quasi-likelihood Ratio Statistic

As described by Wooldridge (1996), specification testing using likelihood ratio statistics requires some care when based upon QML models. We illustrate here the differences between a standard LR test for significant coefficients and the corresponding QLR statistic.

From the results above, we know that the overall likelihood ratio statistic for the Poisson model is 16.85, with a corresponding p -value of 0.0002. This statistic is valid under the assumption that $m(x_i, \beta)$ is specified correctly and that the mean-variance equality holds.

We can decisively reject the latter hypothesis, suggesting that we should derive the QML estimator with consistently estimated covariance matrix under the GLM variance assumption. While EViews does not automatically adjust the LR statistic to reflect the QML assumption, it is easy enough to compute the adjustment by hand. Following Wooldridge, we construct the QLR statistic by dividing the original LR statistic by the estimated GLM variance factor.

Suppose that the estimated QML equation is named EQ1. Then you can use EViews to compute p -value associated with this statistic, placing the results in scalars using the following commands:

```
scalar qlr = eq1.@logl/2.226420477
scalar qpval = 1-@cchisq(qlr, 2)
```

You can examine the results by clicking on the scalar objects in the workfile window and viewing the results in the status line. The QLR statistic is 7.5666, and the p -value is 0.023. The statistic and p -value are valid under the weaker conditions that the conditional mean is correctly specified, and that the conditional variance is proportional (but not necessarily equal) to the conditional mean.

Commands

The following are some examples of carrying out the above procedures using commands.

To estimate a logit of LFP on a constant, AGE, EDU, and EDU squared, type

```
equation eq1.binary(d=1) lfp c age edu edu^2
```

To carry out the Hosmer-Lemeshow goodness-of-fit test for the estimated equation EQ1, type

```
eq1.fittest(h)
```

To save the generalized residuals of EQ1 as named series RES_G, type

```
eq1.makesresid(g) res_g
```

To forecast (static) from EQ1 the index $x_i'\beta$ as named series XB, type

```
eq1.fit(i) xb
```

For a complete list of commands and options available for estimation methods in this chapter, see the *Command and Programming Reference*.

Technical Notes

Huber/White (QML) Standard Errors

The Huber/White options for robust standard errors computes the quasi-maximum likelihood (or pseudo-ML) standard errors

$$\text{var}_{QML}(\hat{\beta}) = \hat{H}^{-1} \hat{g} \hat{g}' \hat{H}^{-1}, \quad (17.49)$$

where \hat{g} and \hat{H}^{-1} are the gradient (or score) and Hessian of the log likelihood evaluated at the ML estimates.

Note that these standard errors are *not* robust to heteroskedasticity in binary dependent variable models. They are robust to certain misspecifications of the underlying distribution of y .

GLM Standard Errors

Many of the discrete and limited dependent variable models described in this chapter belong to a class of models known as *generalized linear models* (GLM). The assumption of GLM is that the distribution of the dependent variable y_i belongs to the exponential family and that the conditional mean of y_i is a (smooth) nonlinear transformation of the linear part $x_i'\beta$:

$$E(y_i|x_i, \beta) = h(x_i'\beta). \quad (17.50)$$

Even though the QML covariance is robust to general misspecification of the conditional distribution of y_i , it does not possess any efficiency properties. An alternative consistent estimate of the covariance is obtained if we impose the GLM condition that the (true) variance of y_i is proportional to the variance of the distribution used to specify the log likelihood:

$$\text{var}(y_i|x_i, \beta) = \sigma^2 \text{var}_{ML}(y_i|x_i, \beta). \quad (17.51)$$

In other words, the ratio of the (conditional) variance to the mean is some constant σ^2 that is independent of x . The most empirically relevant case is $\sigma^2 > 1$, which is known as *overdispersion*. If this proportional variance condition holds, a consistent estimate of the GLM covariance is given by

$$\text{var}_{GLM}(\hat{\beta}) = \hat{\sigma}^2 \text{var}_{ML}(\hat{\beta}), \quad (17.52)$$

where

$$\hat{\sigma}^2 = \frac{1}{N-K} \cdot \sum_{i=1}^N \frac{(y_i - \hat{y}_i)^2}{\sqrt{v(x_i, \hat{\beta}, \hat{\gamma})}} = \frac{1}{N-K} \cdot \sum_{i=1}^N \frac{\hat{u}_i^2}{\sqrt{(v(x_i, \hat{\beta}, \hat{\gamma}))}}. \quad (17.53)$$

If you select GLM standard errors, the estimated proportionality term $\hat{\sigma}^2$ is reported as the variance factor estimate in EViews.

For more discussion on GLM and the phenomenon of overdispersion, see McCullough and Nelder (1989) or Fahrmeir and Tutz (1994).

The Hosmer-Lemeshow Test

Let the data be grouped into $j = 1, 2, \dots, J$ groups, and let n_j be the number of observations in group j . Define the number of $y_i = 1$ observations and the average of predicted values in group j as

$$\begin{aligned}
 y(j) &= \sum_{i \in j} y_i \\
 \bar{p}(j) &= \sum_{i \in j} \hat{p}_i / n_j = \sum_{i \in j} (1 - F(-x_i' \hat{\beta})) / n_j
 \end{aligned}
 \tag{17.54}$$

The Hosmer-Lemeshow test statistic is computed as

$$HL = \sum_{j=1}^J \frac{(y(j) - n_j \bar{p}(j))^2}{n_j \bar{p}(j)(1 - \bar{p}(j))}.
 \tag{17.55}$$

The distribution of the HL statistic is not known, however Hosmer and Lemeshow (1989, p.141) report evidence from extensive simulation indicating that when the model is correctly specified, the distribution of the statistic is well approximated by a χ^2 distribution with $J - 2$ degrees of freedom. Note that these findings are based on a simulation where J is close to n .

The Andrews Test

Let the data be grouped into $j = 1, 2, \dots, J$ groups. Since y is binary, there are $2J$ cells into which any observation can fall. Andrews (1988a, 1988b) compares the $2J$ vector of the actual number of observations in each cell to those predicted from the model, forms a quadratic form, and shows that the quadratic form has an asymptotic χ^2 distribution if the model is specified correctly.

Andrews suggests three tests depending on the choice of the weighting matrix in the quadratic form. EViews uses the test that can be computed by an auxiliary regression as described in Andrews (1988a, 3.18) or Andrews (1988b, 17).

Briefly, let \tilde{A} be an $n \times J$ matrix with element $\tilde{a}_{ij} = 1(i \in j) - \hat{p}_i$, where the indicator function $1(i \in j)$ takes the value one if observation i belongs to group j with $y_i = 1$, and zero otherwise (we drop the columns for the groups with $y = 0$ to avoid singularity). Let B be the $n \times K$ matrix of the contributions to the score $\partial l(\beta) / \partial \beta'$. The Andrews test statistic is n times the R^2 from regressing a constant (one) on each column of \tilde{A} and B . Under the null hypothesis that the model is correctly specified, nR^2 is asymptotically distributed χ^2 with J degrees of freedom.

Chapter 18. The Log Likelihood (LogL) Object

EViews contains customized procedures which help solve the majority of the estimation problems that you might encounter. On occasion, however, you may come across an estimation specification which is not included among these specialized routines. This specification may be an extension of an existing procedure, or it could be an entirely new class of problem.

Fortunately, EViews provides you with tools to estimate a wide variety of specifications through the *log likelihood (logl)* object. The *logl* object provides you with a general, open-ended tool for estimating a broad class of specifications by maximizing a likelihood function with respect to parameters.

When working with a log likelihood object, you will use EViews' series generation capabilities to describe the log likelihood contribution of each observation in your sample as a function of unknown parameters. You may supply analytical derivatives of the likelihood for one or more parameters, or you can simply let EViews calculate numeric derivatives automatically. EViews will search for the parameter values that maximize the specified likelihood function, and will provide estimated standard errors for these parameter estimates.

In this chapter we provide an overview and describe the general features of the *logl* object. We also give examples of specifications which may be estimated using the object. The examples include: multinomial logit, unconditional maximum likelihood AR(1) estimation, Box-Cox regression, disequilibrium switching models, least squares with multiplicative heteroskedasticity, probit specifications with heteroskedasticity, probit with grouped data, nested logit, zero-altered Poisson models, Heckman sample selection models, Weibull hazard models, GARCH(1,1) with *t*-distributed errors, GARCH with coefficient restrictions, EGARCH with a generalized error distribution, and multivariate GARCH.

Overview

Most of the work in estimating a model using the *logl* object is in creating the text specification which will be used to evaluate the likelihood function.

If you are familiar with the process of generating series in EViews, you should find it easy to work with the *logl* specification, since the likelihood specification is merely a list of series assignment statements which are evaluated iteratively during the course of the maximization procedure. All you need to do is write down a set of statements which, when evaluated, will describe a series containing the contributions of each observation to the log likelihood function.

To take a simple example, suppose you believe that your data are generated by the conditional heteroskedasticity regression model:

$$\begin{aligned}y_t &= \beta_1 + \beta_2 x_t + \beta_3 z_t + \epsilon_t \\ \epsilon_t &\sim N(0, \sigma^2 z_t^\alpha)\end{aligned}\tag{18.1}$$

where x , y , and z are the observed series (data) and $\beta_1, \beta_2, \beta_3, \sigma, \alpha$ are the parameters of the model. The log likelihood function (the log of the density of the observed data) for a sample of T observations can be written as

$$\begin{aligned}l(\beta, \alpha, \sigma) &= -\frac{T}{2}(\log(2\pi) + \log\sigma^2) - \frac{\alpha}{2} \sum_{t=1}^T \log(z_t) - \sum_{t=1}^T \frac{(y_t - \beta_1 - \beta_2 x_t - \beta_3 z_t)^2}{\sigma^2 z_t^\alpha} \\ &= \sum_{t=1}^T \left\{ \log\phi\left(\frac{y_t - \beta_1 - \beta_2 x_t - \beta_3 z_t}{\sigma z_t^{\alpha/2}}\right) - \frac{1}{2} \log(\sigma^2 z_t^\alpha) \right\}\end{aligned}\tag{18.2}$$

where ϕ is the standard normal density function.

Note that we can write the log likelihood function as a sum of the log likelihood contributions for each observation t :

$$l(\beta, \alpha, \sigma) = \sum_{t=1}^T l_t(\beta, \alpha, \sigma)\tag{18.3}$$

where the individual contributions are given by

$$l_t(\beta, \alpha, \sigma) = \log\phi\left(\frac{y_t - \beta_1 - \beta_2 x_t - \beta_3 z_t}{\sigma z_t^{\alpha/2}}\right) - \frac{1}{2} \log(\sigma^2 z_t^\alpha)\tag{18.4}$$

Suppose that you know the true parameter values of the model, and you wish to generate a series in EViews which contains the contributions for each observation. To do this, you could assign the known values of the parameters to the elements C(1) to C(5) of the coefficient vector, and then execute the following list of assignment statements as commands or in an EViews program:

```
series res = y - c(1) - c(2)*x - c(3)*z
series var = c(4) * z^c(5)
series logl1 = log(@dnorm(res/@sqrt(var))) - log(var)/2
```

The first two statements describe series which will contain intermediate results used in the calculations. The first statement creates the residual series, RES, and the second statement creates the variance series, VAR. The series LOGL1 contains the set of log likelihood contributions for each observation.

Now suppose instead that you do not know the true parameter values of the model, and would like to estimate them from the data. The maximum likelihood estimates of the parameters are defined as the set of parameter values which produce the largest value of the likelihood function evaluated across all the observations in the sample.

The `logl` object makes finding these maximum likelihood estimates easy. Simply create a new log likelihood object, input the assignment statements above into the `logl` specification view, then ask EViews to estimate the specification.

In entering the assignment statements, you need only make two minor changes to the text above. First, the `series` keyword must be removed from the beginning of each line (since the likelihood specification implicitly assumes it is present). Second, an extra line must be added to the specification which identifies the name of the series in which the likelihood contributions will be contained. Thus, you should enter the following into your log likelihood object:

```
@logl logl1
res = y - c(1) - c(2)*x - c(3)*z
var = c(4) * z^c(5)
logl1 = log(@dnorm(res/@sqrt(var))) - log(var)/2
```

The first line in the log likelihood specification, `@logl logl1`, tells EViews that the series `LOGL1` should be used to store the likelihood contributions. The remaining lines describe the computation of the intermediate results, and the actual likelihood contributions.

When you tell EViews to estimate the parameters of this model, it will execute the assignment statements in the specification repeatedly for different parameter values, using an iterative algorithm to search for the set of values that maximize the sum of the log likelihood contributions. When EViews can no longer improve the overall likelihood, it will stop iterating and will report final parameter values and estimated standard errors in the estimation output.

The remainder of this chapter discusses the rules for specification, estimation and testing using the likelihood object in greater detail.

Specification

To create a likelihood object, choose **Objects/New Object.../LogL** or type “logl” in the command window. The likelihood window will open with a blank specification view. The specification view is a text window into which you enter a list of statements which describe your statistical model, and in which you set options which control various aspects of the estimation procedure.

Specifying the Likelihood

As described in the overview above, the core of the likelihood specification is a set of assignment statements which, when evaluated, generate a series containing the log likelihood contribution of each observation in the sample. There can be as many or as few of these assignment statements as you wish.

Each likelihood specification must contain a control statement which provides the name of the series which is used to contain the likelihood contributions. The format of this statement is

```
@logl series_name
```

where `series_name` is the name of the series which will contain the contributions. This control statement may appear anywhere in the `logl` specification.

Whenever the specification is evaluated, whether for estimation or for carrying out a View or Proc, each assignment statement will be evaluated at the current parameter values, and the results stored in a series with the specified name. If the series does not exist, it will be created automatically. If the series already exists, EViews will use the existing series for storage, and will overwrite the data contained in the series.

If you would like to remove one or more of the series used in the specification after evaluation, you can use the `@temp` statement:

```
@temp series_name1 series_name2 ...
```

This statement tells EViews to delete any series in the list after evaluation of the specification is completed. Deleting these series may be useful if your `logl` creates a lot of intermediate results, and you do not want the series containing these results to clutter your workfile.

Parameter Names

In the example above, we used the coefficients `C(1)` to `C(5)` as names for our unknown parameters. More generally, any element of a named coefficient vector which appears in the specification will be treated as a parameter to be estimated.

In the conditional heteroskedasticity example, you might choose to use coefficients from three different coefficient vectors: one vector for the mean equation, one for the variance equation, and one for the variance parameters. You would first create three named coefficient vectors by the commands

```
coef(3) beta
coef(1) scale
coef(1) alpha
```

You could then write the likelihood specification as

```
@logl logl1
res = y - beta(1) - beta(2)*x - beta(3)*z
var = scale(1)*z^alpha(1)
logl1 = log(@dnorm(res/@sqrt(var))) - log(var)/2
```

Since all elements of named coefficient vectors in the specification will be treated as parameters, you should make certain that all coefficients really do affect the value of one or more of the likelihood contributions. If a parameter has no effect upon the likelihood, you will experience a singularity error when you attempt to estimate the parameters.

Note that all objects other than coefficient elements will be considered fixed and will not be updated during estimation. For example, suppose that SIGMA is a named scalar in your workfile. Then if you redefine the subexpression for VAR as

```
var = sigma*z^alpha(1)
```

EViews will not estimate SIGMA. The value of SIGMA will remain fixed at its value at the start of estimation.

Order of Evaluation

The logl specification contains one or more assignment statements which generate the series containing the likelihood contributions. EViews always evaluates from top to bottom when executing these assignment statements, so expressions which are used in subsequent calculations should always be placed first.

EViews must also iterate through the observations in the sample. Since EViews iterates through both the equations in the specification and the observations in the sample, you will need to specify the order in which the evaluation of observations and equations occurs.

By default, EViews evaluates the specification *by observation* so that *all of the assignment statements* are evaluated for the *first observation*, then for the second observation, and so on across all the observations in the estimation sample. This is the correct order for recursive models where the likelihood of an observation depends on previously observed (lagged) values, as in AR or ARCH models.

You can change the order of evaluation so EViews evaluates the specification *by equation*, so *the first assignment statement* is evaluated for *all the observations*, then the second assignment statement is evaluated for all the observations, and so on for each of the assignment statements in the specification. This is the correct order for models where aggregate statistics from intermediate series are used as input to subsequent calculations.

You can explicitly select which method of evaluation you would like by adding a statement to the likelihood specification. To force evaluation by equation, simply add a line containing the keyword “@byeqn”. To explicitly state that you require evaluation by observation, the “@byobs” keyword can be used. If no keyword is provided, @byobs is assumed.

In the conditional heteroskedasticity example above, it does not matter whether the assignment statements are evaluated by equation (line by line) or by observation, since the results do not depend upon the order of evaluation.

However, if the specification has a recursive structure, or if the specification requires the calculation of aggregate statistics based on intermediate series, you must select the appropriate evaluation order if the calculations are to be carried out correctly.

As an example of the @byeqn statement, consider the following specification:

```
@logl robust1
@byeqn
res1 = y-c(1)-c(2)*x
delta = @abs(res1)/6/@median(@abs(res1))
weight = (delta<1)*(1-delta^2)^2
robust1 = -(weight*res1^2)
```

This specification performs robust regression by downweighting outlier residuals at each iteration. The assignment statement for DELTA computes the median of the absolute value of the residuals in each iteration, and this is used as a reference point for forming a weighting function for outliers. The @byeqn statement instructs EViews to compute all residuals RES1 at a given iteration before computing the median of those residuals when calculating the DELTA series.

Analytic Derivatives

By default, when maximizing the likelihood and forming estimates of the standard errors, EViews computes numeric derivatives of the likelihood function with respect to the parameters. If you would like to specify an analytic expression for one or more of the derivatives, you may use the @deriv statement. The @deriv statement has the form:

```
@deriv pname1 sname1 pname2 sname2 ...
```

where pname is a parameter in the model and sname is the name of the corresponding derivative series generated by the specification.

For example, consider the following likelihood object that specifies a multinomial logit model:

```
' multinomial logit with 3 outcomes
@logl logl1
```

```

xb2 = b2(1)+b2(2)*x1+b2(3)*x2
xb3 = b3(1)+b3(2)*x1+b3(3)*x2
denom = 1+exp(xb2)+exp(xb3)
' derivatives wrt the 2nd outcome params
@deriv b2(1) grad21 b2(2) grad22 b2(3) grad23
grad21 = d2-exp(xb2)/denom
grad22 = grad21*x1
grad23 = grad21*x2
' derivatives wrt the 3rd outcome params
@deriv b3(1) grad31 b3(2) grad32 b3(3) grad33
grad31 = d3-exp(xb3)/denom
grad32 = grad31*x1
grad33 = grad31*x2
' specify log likelihood
logl1 = d2*xb2+d3*xb3-log(1+exp(xb2)+exp(xb3))

```

See Greene (1997), Chapter 19.7.1 for a discussion of multinomial logit models. There are three possible outcomes, and the parameters of the three regressors (X_1 , X_2 and the constant) are normalized relative to the first outcome. The analytic derivatives are particularly simple for the multinomial logit model and the two `@deriv` statements in the specification instruct EViews to use the expressions for GRAD21, GRAD22, GRAD23, GRAD31, GRAD32, and GRAD33, instead of computing numeric derivatives.

When working with analytic derivatives, you may wish to check the validity of your expressions for the derivatives by comparing them with numerically computed derivatives. EViews provides you with tools which will perform this comparison at the current values of parameters or at the specified starting values. See the discussion of the **Check Derivatives** view of the likelihood object in the Command Reference at the end of this Chapter.

Derivative Step Sizes

If analytic derivatives are not specified for any of your parameters, EViews numerically evaluates the derivatives of the likelihood function for those parameters. The step sizes used in computing the derivatives are controlled by two parameters: r (relative step size) and m (minimum step size). Let $\theta^{(i)}$ denote the value of the parameter θ at iteration i . Then the step size at iteration $i + 1$ is determined by

$$s^{(i+1)} = \max(r\theta^{(i)}, m) \quad (18.5)$$

The two-sided numeric derivative is evaluated as

$$\frac{f(\theta^{(i)} + s^{(i+1)}) - f(\theta^{(i)} - s^{(i+1)})}{2s^{(i+1)}} \quad (18.6)$$

while the one-sided numeric derivative is evaluated as

$$\frac{f(\theta^{(i)} + s^{(i+1)}) - f(\theta^{(i)})}{s^{(i+1)}} \quad (18.7)$$

where f is the likelihood function. Two-sided derivatives are more accurate, but require roughly twice as many evaluations of the likelihood function and so take about twice as long to evaluate.

The `@derivstep` statement can be used to control the step size and method used to evaluate the derivative at each iteration. The `@derivstep` keyword should be followed by sets of three arguments: the name of the parameter to be set (or the keyword `@all`), the relative step size, and the minimum step size.

The default setting is (approximately):

```
@derivstep(1) @all 1.49e-8 1e-10
```

where “1” in the parentheses indicates that one-sided numeric derivatives should be used and `@all` indicates that the following setting applies to all of the parameters. The first number following `@all` is the relative step size and the second number is the minimum step size. The default relative step size is set to the square root of machine epsilon (1.49×10^{-8}) and the minimum step size is set to $m = 10^{-10}$.

The step size can be set separately for each parameter in a single or in multiple `@derivstep` statements. The evaluation method option specified in parentheses is a global option; it cannot be specified separately for each parameter.

For example, if you include the line

```
@derivstep(2) c(2) 1e-7 1e-10
```

the relative step size for coefficient `C(2)` will be increased to $m = 10^{-7}$ and a two-sided derivative will be used to evaluate the derivative. In a more complex example,

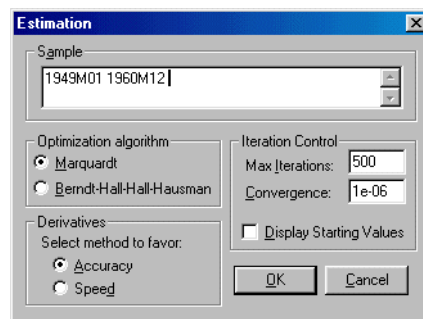
```
@derivstep(2) @all 1.49e-8 1e-10 c(2) 1e-7 1e-10 c(3) 1e-5 1e-8
```

computes two-sided derivatives using the default step sizes for all coefficients except `C(2)` and `C(3)`. The values for these latter coefficients are specified directly.

Estimation

Once you have specified the logl object, you can ask EViews to find the parameter values which maximize the likelihood parameters. Simply click the **Estimate** button in the likelihood window toolbar to open the Estimation Options dialog.

There are a number of options which allow you to control various aspects of the estimation procedure. See “[Setting Estimation Options](#)” on page 666 for a discussion of these options. The default settings, however, should provide a good start for most problems. When you click on OK, EViews will begin estimation using the current settings.



Starting Values

Since EViews uses an iterative algorithm to find the maximum likelihood estimates, the choice of starting values is important. For problems in which the likelihood function is globally concave, it will influence how many iterations are taken for estimation to converge. For problems where the likelihood function is not concave, it may determine which of several local maxima is found. In some cases, estimation will fail unless reasonable starting values are provided.

By default, EViews uses the values stored in the coefficient vector or vectors prior to estimation. If a @param statement is included in the specification, the values specified in the statement will be used instead.

In our conditional heteroskedasticity regression example, one choice for starting values for the coefficients of the mean equation coefficients are the simple OLS estimates, since OLS provides consistent point estimates even in the presence of (bounded) heteroskedasticity. To use the OLS estimates as starting values, first estimate the OLS equation by the command

```
equation eq1.ls y c x z
```

After estimating this equation, the elements C(1), C(2), C(3) of the C coefficient vector will contain the OLS estimates. To set the variance scale parameter C(4) to the estimated OLS residual variance, you can type the assignment statement in the command window

```
c(4) = eq1.@se^2
```

For the final heteroskedasticity parameter C(5), you can use the residuals from the original OLS regression to carry out a second OLS regression, and set the value of C(5) to the appropriate coefficient. Alternatively, you can arbitrarily set the parameter value using a simple assignment statement:

```
c(5) = 1
```

Now, if you estimate the logl specification immediately after carrying out the OLS estimation and subsequent commands, it will use the values that you have placed in the C vector as starting values.

As noted above, an alternative method of initializing the parameters to known values is to include a @param statement in the likelihood specification. For example, if you include the line

```
@param c(1) 0.1 c(2) 0.1 c(3) 0.1 c(4) 1 c(5) 1
```

in the specification of the logl, EViews will always set the starting values to $C(1) = C(2) = C(3) = 0.1$, $C(4) = C(5) = 1$.

See also the discussion of starting values in [“Starting Coefficient Values” on page 667](#).

Estimation Sample

EViews uses the sample of observations specified in the Estimation Options dialog when estimating the parameters of the log likelihood. EViews evaluates each expression in the logl for every observation in the sample at current parameter values, using the *by observation* or *by equation* ordering. All of these evaluations follow the standard EViews rules for evaluating series expressions.

If there are missing values in the log likelihood series at the initial parameter values, EViews will issue an error message and the estimation procedure will stop. In contrast to the behavior of other EViews built-in procedures, logl estimation performs no endpoint adjustments or dropping of observations with missing values when estimating the parameters of the model.

LogL Views

- **Likelihood Specification:** displays the window where you specify and edit the likelihood specification.
- **Estimation Output:** displays the estimation results obtained from maximizing the likelihood function.
- **Covariance Matrix:** displays the estimated covariance matrix of the parameter estimates. These are computed from the inverse of the sum of the outer product of the first derivatives evaluated at the optimum parameter values. To save this covariance matrix as a (SYM) MATRIX, you may use the @COV function.
- **Wald Coefficient Tests...:** performs the Wald coefficient restriction test. See Chapter 14, Coefficient Tests, for a discussion of Wald tests.

- **Gradients:** displays view of the gradients (first derivatives) of the log likelihood at the current parameter values (if the model has not yet been estimated), or at the converged parameter values (if the model has been estimated). These views may prove to be useful diagnostic tools if you are experiencing problems with convergence.
- **Check Derivatives:** displays the values of the numeric derivatives and analytic derivatives (if available) at the starting values (if a `@param` statement is included), or at current parameter values (if there is no `@param` statement).

LogL Procs

- **Estimate...:** brings up a dialog to set estimation options, and to estimate the parameters of the log likelihood.
- **Make Model:** creates an untitled model object out of the estimated likelihood specification.
- **Make Gradient Group:** creates an untitled group of the gradients (first derivatives) of the log likelihood at the estimated parameter values. These gradients are often used in constructing Lagrange multiplier tests.
- **Update Coefs from LogL:** updates the coefficient vector(s) with the estimates from the likelihood object. This procedure allows you to export the maximum likelihood estimates for use as starting values in other estimation problems.

Most of these procedures should be familiar to you from other EViews estimation objects. We describe below the features that are specific to the logl object.

Estimation Output

In addition to the coefficient and standard error estimates, the standard output for the logl object describes the method of estimation, sample used in estimation, date and time that the logl was estimated, evaluation order, and information about the convergence of the estimation procedure.

LogL: MLOGIT
Method: Maximum Likelihood (Marquardt)
Date: 10/19/00 Time: 14:26
Sample: 1 1000
Included observations: 1000
Evaluation order: By observation
Estimation settings: tol= 1.0E-09
Initial Values: B2(1)=-1.08356, B2(2)=0.90467, B2(3)=-0.06786, B3(1)=-0.69842, B3(2)=-0.33212, B3(3)=0.32981
Convergence achieved after 7 iterations

	Coefficient	Std. Error	z-Statistic	Prob.
B2(1)	-0.521793	0.205568	-2.538302	0.0111
B2(2)	0.994358	0.267963	3.710798	0.0002
B2(3)	0.134983	0.265655	0.508115	0.6114
B3(1)	-0.262307	0.207174	-1.266122	0.2055
B3(2)	0.176770	0.274756	0.643371	0.5200
B3(3)	0.399166	0.274056	1.456511	0.1453
Log likelihood	-1089.415	Akaike info criterion	2.190830	
Avg. log likelihood	-1.089415	Schwarz criterion	2.220277	
Number of Coefs.	6	Hannan-Quinn criter.	2.202022	

EViews also provides the log likelihood value, average log likelihood value, number of coefficients, and three Information Criteria. By default, the starting values are not displayed. Here we have used the Estimation Options dialog to instruct EViews to display the estimation starting values in the output.

Gradients

The gradient summary, table and graph view allow you to examine the gradients of the likelihood. These gradients are computed at the current parameter values (if the model has not yet been estimated), or at the converged parameter values (if the model has been estimated). See [Appendix E, “Gradients and Derivatives”](#), on page 675 for additional details.

You may find this view to be a useful diagnostic tool when experiencing problems with convergence or singularity. One common problem leading to singular matrices is a zero derivative for a parameter due to an incorrectly specified likelihood, poor starting values, or a lack of model identification. See the discussion below for further details.

LogL: MLOGIT Workfile: MLOGIT									
View	Procs	Objects	Print	Name	Freeze	MergeText	Estimate	Stats	Spec
Gradients at estimated parameters									
obs	B2(1)	B2(2)	B2(3)	B3(1)	B3(2)				
1	0.617633	0.446035	0.449061	-0.332790	-0.241				
2	0.660916	0.310714	0.074241	-0.308290	-0.14				
3	0.665234	0.330282	0.518881	-0.355333	-0.17				
4	-0.420696	-0.360283	-0.076988	-0.284167	-0.24				
5	0.576206	0.516712	0.306708	-0.303744	-0.27				
6	-0.394766	-0.296582	-0.147463	0.694417	0.52				
7	-0.386750	-0.269690	-0.047604	-0.292850	-0.20				
8	0.582613	0.509899	0.362163	-0.311741	-0.27				
9	-0.288308	-0.076096	-0.252996	-0.379763	-0.10				
10	0.553928	0.549244	0.256273	-0.290475	-0.28				
11	-0.301615	-0.086085	-0.056264	0.674758	0.19				
12	-0.259970	-0.029439	-0.251003	-0.396446	-0.04				
13	0.689356	0.244054	0.363363	-0.346518	-0.12				
14									

Check Derivatives

You can use the **Check Derivatives** view to examine your numeric derivatives or to check the validity of your expressions for the analytic derivatives. If the logl specification contains a @param statement, the derivatives will be evaluated at the specified values, otherwise, the derivatives will be computed at the current coefficient values.

The first part of this view displays the names of the user supplied derivatives, step size parameters, and the coefficient values at which the derivatives are evaluated. The relative and minimum step sizes shown in this example are the default settings.

The second part of the view computes the sum (over all individuals in the sample) of the numeric and, if applicable, the analytic derivatives for each coefficient. If appropriate, EViews will also compute the largest individual difference between the analytic and the numeric derivatives in both absolute, and percentage terms.

Coefficient	User	Rel. Step	Min. Step	Coef. Value
B2(1)	GRAD21	1.49E-08	1.00E-10	1.000000
B2(2)	GRAD22	1.49E-08	1.00E-10	1.000000
B2(3)	GRAD23	1.49E-08	1.00E-10	1.000000
B3(1)	GRAD31	1.49E-08	1.00E-10	1.000000
B3(2)	GRAD32	1.49E-08	1.00E-10	1.000000
B3(3)	GRAD33	1.49E-08	1.00E-10	1.000000

Coefficient	Sum Over Obs.		Maximum Diff.	
	Numeric	User	Absolute	Percent
B2(1)	-122.5118	-122.5117	3.65E-08	7.62E-06
B2(2)	-48.24256	-48.24256	-4.32E-08	0.001530
B2(3)	-65.16444	-65.16444	4.28E-08	0.000892
B3(1)	-134.5118	-134.5117	3.65E-08	7.62E-06
B3(2)	-77.25805	-77.25805	-4.13E-08	0.001530
B3(3)	-64.14204	-64.14204	4.28E-08	0.002391

Troubleshooting

Because the logl object provides a great deal of flexibility, you are more likely to experience problems with estimation using the logl object than with EViews' built-in estimators.

If you are experiencing difficulties with estimation the following suggestions may help you in solving your problem:

- **Check your likelihood specification.** A simple error involving a wrong sign can easily stop the estimation process from working. You should also verify that the parameters of the model are really identified (in some specifications you may have to impose a normalization across the parameters). Also, every parameter which appears in the model must feed directly or indirectly into the likelihood contributions. The **Check Derivatives** view is particularly useful in helping you spot the latter problem.
- **Choose your starting values.** If any of the likelihood contributions in your sample cannot be evaluated due to missing values or because of domain errors in mathematical operations (logs and square roots of negative numbers, division by zero,

etc.) the estimation will stop immediately with the message: “Cannot compute @logl due to missing values”. In other cases, a bad choice of starting values may lead you into regions where the likelihood function is poorly behaved. You should always try to initialize your parameters to sensible numerical values. If you have a simpler estimation technique available which approximates the problem, you may wish to use estimates from this method as starting values for the maximum likelihood specification.

- **Make sure lagged values are initialized correctly.** In contrast to most other estimation routines in EViews, the logl estimation procedure will not automatically drop observations with NAs or lags from the sample when estimating a log likelihood model. If your likelihood specification involves lags, you will either have to drop observations from the beginning of your estimation sample, or you will have to carefully code the specification so that missing values from before the sample do not cause NAs to propagate through the entire sample (see the AR(1) and GARCH examples for a demonstration).

Since the series used to evaluate the likelihood are contained in your workfile (unless you use the `@temp` statement to delete them), you can examine the values in the log likelihood and intermediate series, to find problems involving lags and missing values.

- **Verify your derivatives.** If you are using analytic derivatives, use the **Check Derivatives** view to make sure you have coded the derivatives correctly. If you are using numerical derivatives, consider specifying analytic derivatives or adjusting the options for derivative method or step size.
- **Reparametrize your model.** If you are having problems with parameter values causing mathematical errors, you may wish to consider reparameterizing the model to restrict the parameter within its valid domain. See the discussion below for examples.

Most of the error messages you are likely to see during estimation are self-explanatory. The error message “near singular matrix” may be less obvious. This error message occurs when EViews is unable to invert the matrix of the sum of the outer product of the derivatives so that it is impossible to determine the direction of the next step of the optimization. This error may indicate a wide variety of problems, including bad starting values, but will almost always occur if the model is not identified, either theoretically, or in terms of the available data.

Limitations

The likelihood object can be used to estimate parameters that maximize (or minimize) a variety of objective functions. Although the main use of the likelihood object will be to specify a log likelihood, you can specify least squares and minimum distance estimation

problems with the likelihood object as long as the objective function is additive over the sample.

You should be aware that the algorithm used in estimating the parameters of the log likelihood is not well suited to solving arbitrary maximization or minimization problems. The algorithm forms an approximation to the Hessian of the log likelihood, based on the sum of the outer product of the derivatives of the likelihood contributions. This approximation relies on both the functional form and statistical properties of maximum likelihood objective functions, and may not be a good approximation in general settings. Consequently, you may or may not be able to obtain results with other functional forms. Furthermore, the standard error estimates of the parameter values will only have meaning if the series describing the log likelihood contributions are (up to an additive constant) the individual contributions to a correctly specified, well-defined theoretical log likelihood.

Currently, the expressions used to describe the likelihood contribution must follow the rules of EViews series expressions. This restriction implies that we do not allow matrix operations in the likelihood specification. In order to specify likelihood functions for multiple equation models, you may have to write out the expression for the determinants and quadratic forms. Although possible, this may become tedious for models with more than two or three equations. See the multivariate GARCH sample programs for examples of this approach.

Additionally, the `logl` object does not directly handle optimization subject to general inequality constraints. There are, however, a variety of well-established techniques for imposing simple inequality constraints. We provide examples below. The underlying idea is to apply a monotonic transformation to the coefficient so that the new coefficient term takes on values only in the desired range. The commonly used transformations are the `@exp` for one-sided restrictions and the `@logit` and `@arctan` for two-sided restrictions.

You should be aware of the limitations of the transformation approach. First, the approach only works for relatively simple inequality constraints. If you have several cross-coefficient inequality restrictions, the solution will quickly become intractable. Second, in order to perform hypothesis tests on the untransformed coefficient, you will have to obtain an estimate of the standard errors of the associated expressions. Since the transformations are generally nonlinear, you will have to compute linear approximations to the variances yourself (using the delta method). Lastly, inference will be poor near the boundary values of the inequality restrictions.

Simple One-Sided Restrictions

Suppose you would like to restrict the estimate of the coefficient of X to be no larger than 1. One way you could do this is to specify the corresponding subexpression as follows:

```
' restrict coef on x to not exceed 1
```

```
res1 = y - c(1) - (1-exp(c(2)))*x
```

Note that EViews will report the point estimate and the standard error for the parameter $C(2)$, not the coefficient of X . To find the standard error of the expression $1 - \exp(c(2))$, you will have to use the delta method; see for example Greene (1997), Theorems 4.15 and 4.16.

Simple Two-Sided Restrictions

Suppose instead that you want to restrict the coefficient for X to be between -1 and 1 . Then you can specify the expression as:

```
' restrict coef on x to be between -1 and 1
res1 = y - c(1) - (2*@logit(c(2))-1)*x
```

Again, EViews will report the point estimate and standard error for the parameter $C(2)$. You will have to use the delta method to compute the standard error of the transformation expression $2 * @logit(c(2)) - 1$.

More generally, if you want to restrict the parameter to lie between L and H , you can use the transformation

```
(H-L)*@logit(c(1)) + L
```

where $C(1)$ is the parameter to be estimated. In the above example, $L = -1$ and $H = 1$.

Examples

In this section, we provide extended examples of working with the logl object to estimate a multinomial logit and a maximum likelihood AR(1) specification. Example programs for these and several other specifications are provided in your default EViews data directory. If you set your default directory to point to the EViews data directory, you should be able to issue a RUN command for each of these programs to create the logl object and to estimate the unknown parameters.

Multinomial Logit (mlogit1.prg)

In this example, we demonstrate how to specify and estimate a simple multinomial logit model using the logl object. Suppose the dependent variable Y can take one of three categories 1, 2, and 3. Further suppose that there are data on two regressors, X_1 and X_2 that vary across observations (individuals). Standard examples include variables such as age and level of education. Then the multinomial logit model assumes that the probability of observing each category in Y is given by:

$$\Pr(y_i = j) = \frac{\exp(\beta_{0j} + \beta_{1j}x_{1i} + \beta_{2j}x_{2i})}{\sum_{k=1}^3 \exp(\beta_{0k} + \beta_{1k}x_{1i} + \beta_{2k}x_{2i})} = P_{ij} \quad (18.8)$$

for $j = 1, 2, 3$. Note that the parameters β are specific to each category so there are $3 \times 3 = 9$ parameters in this specification. The parameters are not all identified unless we impose a normalization (see for example Greene, 1997, chapter 19.7), so we normalize the parameters of the first choice category $j = 1$ to be all zero:

$$\beta_{0,1} = \beta_{1,1} = \beta_{2,1} = 0.$$

The log likelihood function for the multinomial logit can be written as

$$l = \sum_{i=1}^N \sum_{j=1}^3 d_{ij} \log(P_{ij}) \quad (18.9)$$

where d_{ij} is a dummy variable that takes the value 1 if observation i has chosen alternative j and 0 otherwise. The first-order conditions are

$$\frac{\partial l}{\partial \beta_{kj}} = \sum_{i=1}^N (d_{ij} - P_{ij}) x_{ki} \quad (18.10)$$

for $k = 0, 1, 2$ and $j = 1, 2, 3$.

We have provided, in the Example Files subdirectory of your default EViews directory, a workfile MLOGIT.WK1 containing artificial multinomial data. The program begins by loading this workfile.

```
' load artificial data
%evworkfile = @evpath + "\example files\logl\mlogit"
load "{%evworkfile}"
```

from the EViews example directory.

Next, we declare the coefficient vectors that will contain the estimated parameters for each choice alternative.

```
' declare parameter vector
coef(3) b2
coef(3) b3
```

As an alternative, we could have used the default coefficient vector C.

We then set up the likelihood function by issuing a series of append statements:

```
mlogit.append xb2 = b2(1)+b2(2)*x1+b2(3)*x2
mlogit.append xb3 = b3(1)+b3(2)*x1+b3(3)*x2
```

```
' define prob for each choice
mlogit.append denom = 1+exp(xb2)+exp(xb3)
mlogit.append pr1 = 1/denom
mlogit.append pr2 = exp(xb2)/denom
mlogit.append pr3 = exp(xb3)/denom
' specify likelihood
mlogit.append logl1 = (1-dd2-dd3)*log(pr1)
      +dd2*log(pr2)+dd3*log(pr3)
```

Since the analytic derivatives for the multinomial logit are particularly simple, we also specify the expressions for the analytic derivatives to be used during estimation and the appropriate `@deriv` statements:

```
' specify analytic derivatives
for!i = 2 to 3
mlogit.append @deriv b{!i}(1) grad{!i}1 b{!i}(2) grad{!i}2
      b{!i}(3) grad{!i}3
mlogit.append grad{!i}1 = dd{!i}-pr{!i}
mlogit.append grad{!i}2 = grad{!i}1*x1
mlogit.append grad{!i}3 = grad{!i}1*x2
next
```

Note that if you were to specify this likelihood interactively, you would simply type the expression that follows each `append` statement directly into the MLOGIT object.

This concludes the actual specification of the likelihood object. Before estimating the model, we get the starting values by estimating a series of binary logit models.

```
' get starting values from binomial logit
equation eq2.binary(d=1) dd2 c x1 x2
b2 = eq2.@coefs
equation eq3.binary(d=1) dd3 c x1 x2
b3 = eq3.@coefs
```

To check whether you have specified the analytic derivatives correctly, choose **View/Check Derivatives** or use the command

```
show mlogit.checkderiv
```

If you have correctly specified the analytic derivatives, they should be fairly close to the numeric derivatives.

We are now ready to estimate the model. Either click the **Estimate** button or use the command

```
' do MLE
mlogit.ml(showopts, m=1000, c=1e-5)
show mlogit.output
```

Note that you can examine the derivatives for this model using the **Gradient Table** view, or you can examine the series in the workfile containing the gradients. You can also look at the intermediate results, and log likelihood values. For example, to look at the likelihood contributions for each individual, simply double click on the LOGL1 series.

AR(1) Model (ar1.prg)

In this example, we demonstrate how to obtain full maximum likelihood estimates of an AR(1). The maximum likelihood procedure uses the first observation in the sample, in contrast to the built-in AR(1) procedure in EViews which treats the first observation as fixed and maximizes the conditional likelihood for the remaining observations by nonlinear least squares.

As an illustration, we first generate data that follows an AR(1) process:

```
' make up data
create m 80 89
rndseed 123
series y=0
smp1 @first+1 @last
y = 1+0.85*y(-1) + nrnd
```

The exact Gaussian likelihood function for an AR(1) model is given by

$$f(y, \theta) = \begin{cases} \frac{1}{\sigma\sqrt{2\pi(1-\rho^2)}} \exp\left\{-\frac{(y_t - c/(1-\rho^2))^2}{2(\sigma^2/(1-\rho^2))}\right\} & t = 1 \\ \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{(y_t - c - \rho y_{t-1})^2}{2(\sigma^2)}\right\} & t > 0 \end{cases} \quad (18.11)$$

where c is the constant term, ρ is the AR(1) coefficient, and σ^2 is the error variance, all to be estimated (see for example Hamilton, 1994a, chapter 5.2).

Since the likelihood function evaluation differs for the first observation in our sample, we create a dummy variable indicator for the first observation:

```
' create dummy variable for first obs
```

```
series d1 = 0
smpl @first @first
d1 = 1
smpl @all
```

Next, we declare the coefficient vectors to store the parameter estimates and initialize them with the least squares estimates.

```
' set starting values to LS (drops first obs)
equation eq1.ls y c ar(1)
coef(1) rho = c(2)
coef(1) s2 = eq1.@se^2
```

We then specify the likelihood function. We make use of the `@recode` function to differentiate the evaluation of the likelihood for the first observation from the remaining observations. Note: the `@recode` function used here uses the updated syntax for this function—please double-check the current documentation for details.

```
' set up likelihood
' uses new @recode syntax 6/98
logl ar1
ar1.append @logl logl1
ar1.append var = @recode(d1=1,s2(1)/(1-rho(1)^2),s2(1))
ar1.append res = @recode(d1=1,y-c(1)/(1-rho(1)),y-c(1)-
    rho(1)*y(-1))
ar1.append sres = res/@sqrt(var)
ar1.append logl1 = log(@dnorm(sres))-log(var)/2
```

The likelihood specification uses the built-in function `@dnorm` for the standard normal density. The second term is the Jacobian term that arises from transforming the standard normal variable to one with non-unit variance. (You could, of course, write out the likelihood for the normal distribution without using the `@dnorm` function.)

The program displays the MLE together with the least squares estimates

```
' do MLE
ar1.ml(showopts, m=1000, c=1e-5)
show ar1.output
' compare with EViews AR(1) which ignores first obs
show eq1.output
```

Additional Examples

The following additional example programs can be found in the “Example Files” subdirectory of your default EViews directory.

- **Conditional logit** (clogit1.prg): estimates a conditional logit with 3 outcomes and both individual specific and choice specific regressors. The program also displays the prediction table and carries out a Hausman test for independence of irrelevant alternatives (IIA). See Greene (1997, chapter 19.7) for a discussion of multinomial logit models.
- **Box-Cox transformation** (boxcox1.prg): estimates a simple bivariate regression with an estimated Box-Cox transformation on both the dependent and independent variables. Box-Cox transformation models are notoriously difficult to estimate and the results are very sensitive to starting values.
- **Disequilibrium switching model** (diseq1.prg): estimates the switching model in exercise 15.14–15.15 of Judge et al. (1985, pages 644–646). Note that there are some typos in Judge et al. (1985, pages 639–640). The program uses the likelihood specification in Quandt (1988, page 32, equations 2.3.16–2.3.17).
- **Multiplicative heteroskedasticity** (hetero1.prg): estimates a linear regression model with multiplicative heteroskedasticity. Replicates the results in Greene (1997, example 12.14).
- **Probit with heteroskedasticity** (hprobit1.prg): estimates a probit specification with multiplicative heteroskedasticity. See Greene (1997, example 19.7).
- **Probit with grouped data** (gprobit1.prg): estimates a probit with grouped data (proportions data). Estimates the model in Greene (1997, exercise 19.6).
- **Nested logit** (nlogit1.prg): estimates a nested logit model with 2 branches. Tests the IIA assumption by a Wald test. See Greene (1997, chapter 19.7.4) for a discussion of nested logit models.
- **Zero-altered Poisson model** (zpoiss1.prg): estimates the zero-altered Poisson model. Also carries out the non-nested LR test of Vuong (1989). See Greene (1997, chapter 19.9.6) for a discussion of zero-altered Poisson models and Vuong’s non-nested likelihood ratio test.
- **Heckman sample selection model** (heckman1.prg): estimates Heckman’s two equation sample selection model by MLE using the two-step estimates as starting values.
- **Weibull hazard model** (weibull1.prg): estimates the uncensored Weibull hazard model described in Greene (1997, example 20.18). The program also carries out one of the conditional moment tests in Greene (1997, example 20.19).

- **GARCH(1,1) with t-distributed errors** (`arch_t1.prg`): estimates a GARCH(1,1) model with t -distribution. The log likelihood function for this model can be found in Hamilton (1994a, equation 21.1.24, page 662).
- **GARCH with coefficient restrictions** (`garch1.prg`): estimates an MA(1)-GARCH(1,1) model with coefficient restrictions in the conditional variance equation. This model is estimated by Bollerslev, Engle, and Nelson (1994, equation 9.1, page 3015) for different data.
- **EGARCH with generalized error distributed errors** (`egarch1.prg`): estimates Nelson's (1991) exponential GARCH with generalized error distribution. The specification and likelihood is described in Hamilton (1994a, pages 668–669).
- **Multivariate GARCH** (`bv_garch.prg` and `tv_garch.prg`): estimates the bi- or the tri-variate version of the BEKK GARCH specification (Engle and Kroner, 1995).

Part V. Multiple Equation Analysis

In this section, we document EViews tools for multiple equation estimation, forecasting and data analysis.

- Chapters 19–22 describe estimation techniques for systems of equations (“[System Estimation](#)” on page 495), VARs and VECs (“[Vector Autoregression and Error Correction Models](#)” on page 519), and state space models (“[State Space Models and the Kalman Filter](#)” beginning on page 577).
- Chapter 21, “[Pooled Time Series, Cross-Section Data](#)”, on page 551 outlines tools for working with pooled time series, cross-section data, and estimating standard equation specifications which account for the pooled structure of the data.
- Chapter 22, “[State Space Models and the Kalman Filter](#)”, on page 577 describes the use of EViews’ state space and Kalman filter tools for modeling structural time series models.
- Chapter 23, “[Models](#)”, beginning on page 601 describes the use of model objects to forecast from multiple equation estimates, or to perform multivariate simulation.

Chapter 19. System Estimation

This chapter describes methods of estimating the parameters of systems of equations. We describe least squares, weighted least squares, seemingly unrelated regression (SUR), weighted two-stage least squares, three-stage least squares, full-information maximum likelihood (FIML), and generalized method of moments (GMM) estimation techniques.

Once you have estimated the parameters of your system of equations, you may wish to forecast future values or perform simulations for different values of the explanatory variables. [Chapter 23, “Models”, on page 601](#) describes the use of models to forecast from an estimated system of equations or to perform single and multivariate simulation.

Background

A *system* is a group of equations containing unknown parameters. Systems can be estimated using a number of multivariate techniques that take into account the interdependencies among the equations in the system.

The general form of a system is

$$f(y_t, x_t, \beta) = \epsilon_t, \quad (19.1)$$

where y_t is a vector of endogenous variables, x_t is a vector of exogenous variables, and ϵ_t is a vector of possibly serially correlated disturbances. The task of estimation is to find estimates of the vector of parameters β .

EViews provides you with a number of methods of estimating the parameters of the system. One approach is to estimate each equation in the system separately, using one of the single equation methods described earlier in this manual. A second approach is to estimate, simultaneously, the complete set of parameters of the equations in the system. The simultaneous approach allows you to place constraints on coefficients across equations and to employ techniques that account for correlation in the residuals across equations.

While there are important advantages to using a system to estimate your parameters, they do not come without cost. Most importantly, if you misspecify one of the equations in the system and estimate your parameters using single equation methods, only the misspecified equation will be poorly estimated. If you employ system estimation techniques, the poor estimates for the misspecification equation may “contaminate” estimates for other equations.

At this point, we take care to distinguish between systems of equations and models. A *model* is a group of known equations describing endogenous variables. Models are used to solve for values of the endogenous variables, given information on other variables in the model.

Systems and models often work together quite closely. You might estimate the parameters of a system of equations, and then create a model in order to forecast or simulate values of the endogenous variables in the system. We discuss this process in greater detail in [Chapter 23, “Models”, on page 601](#).

System Estimation Methods

EViews will estimate the parameters of a system of equations using:

- Ordinary least squares.
- Equation weighted regression.
- Seemingly unrelated regression (SUR).
- System two-state least squares.
- Weighted two-stage least squares.
- Three-stage least squares.
- Full information maximum likelihood (FIML).
- Generalized method of moments (GMM).

The equations in the system may be linear or nonlinear, and may contain autoregressive error terms.

In the remainder of this section, we describe each technique at a general level. Users who are interested in the technical details are referred to the [“Technical Discussion” on page 511](#).

Ordinary Least Squares

This technique minimizes the sum-of-squared residuals for each equation, accounting for any cross-equation restrictions on the parameters of the system. If there are no such restrictions, this method is identical to estimating each equation using single-equation ordinary least squares.

Cross-Equation Weighting

This method accounts for cross-equation heteroskedasticity by minimizing the weighted sum-of-squared residuals. The equation weights are the inverses of the estimated equation variances, and are derived from unweighted estimation of the parameters of the system. This method yields identical results to unweighted single-equation least squares if there are no cross-equation restrictions.

Seemingly Unrelated Regression

The seemingly unrelated regression (SUR) method, also known as the multivariate regression, or Zellner's method, estimates the parameters of the system, accounting for heteroskedasticity, and contemporaneous correlation in the errors across equations. The estimates of the cross-equation covariance matrix are based upon parameter estimates of the unweighted system.

Note that EViews estimates a more general form of SUR than is typically described in the literature, since it allows for cross-equation restrictions on parameters.

Two-Stage Least Squares

The system two-stage least squares (STLS) estimator is the system version of the single equation two-stage least squares estimator described above. STLS is an appropriate technique when some of the right-hand side variables are correlated with the error terms, and there is neither heteroskedasticity, nor contemporaneous correlation in the residuals. EViews estimates STLS by applying TSL equation by equation to the unweighted system, enforcing any cross-equation parameter restrictions. If there are no cross-equation restrictions, the results will be identical to unweighted single-equation TSL.

Weighted Two-Stage Least Squares

The weighted two-stage least squares (WTLS) estimator is the two-stage version of the weighted least squares estimator. WTLS is an appropriate technique when some of the right-hand side variables are correlated with the error terms, and there is heteroskedasticity, but no contemporaneous correlation in the residuals.

EViews first applies STLS to the unweighted system. The results from this estimation are used to form the equation weights, based upon the estimated equation variances. If there are no cross-equation restrictions, these first-stage results will be identical to unweighted single-equation TSL.

Three-Stage Least Squares

Three-stage least squares (3SLS) is the two-stage least squares version of the SUR method. It is an appropriate technique when right-hand side variables are correlated with the error terms, and there is both heteroskedasticity, and contemporaneous correlation in the residuals.

EViews applies TSL to the unweighted system, enforcing any cross-equation parameter restrictions. These estimates are used to form an estimate of the full cross-equation covariance matrix which, in turn, is used to transform the equations to eliminate the cross-equation correlation. TSL is applied to the transformed model.

Full Information Maximum Likelihood (FIML)

Full Information Maximum Likelihood (FIML) estimates the likelihood function under the assumption that the contemporaneous errors have a joint normal distribution. Provided that the likelihood function is correctly specified, FIML is fully efficient.

Generalized Method of Moments (GMM)

The GMM estimator belongs to a class of estimators known as M-estimators that are defined by minimizing some criterion function. GMM is a robust estimator in that it does not require information of the exact distribution of the disturbances.

GMM estimation is based upon the assumption that the disturbances in the equations are uncorrelated with a set of instrumental variables. The GMM estimator selects parameter estimates so that the correlations between the instruments and disturbances are as close to zero as possible, as defined by a criterion function. By choosing the weighting matrix in the criterion function appropriately, GMM can be made robust to heteroskedasticity and/or autocorrelation of unknown form.

Many standard estimators, including all of the system estimators provided in EViews, can be set up as special cases of GMM. For example, the ordinary least squares estimator can be viewed as a GMM estimator, based upon the conditions that each of the right-hand side variables is uncorrelated with the residual.

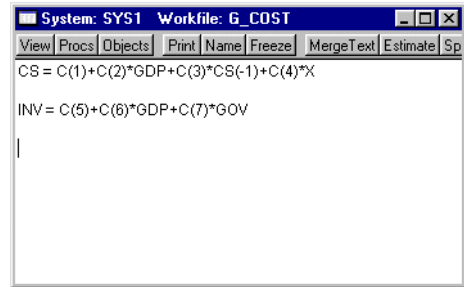
How to Create and Specify a System

To estimate the parameters of your system of equations, you should first create a system object and specify the system of equations. Click on **Objects/New Object/System** or type `system` in the command window. The system object window should appear. When you first create the system, the window will be blank. You will fill the system specification window with text describing the equations, and potentially, lines describing the instruments and the parameter starting values.

Equations

Enter your equations, by formula, using standard EViews expressions. The equations in your system should be behavioral equations with unknown coefficients and an implicit error term.

Consider the specification of a simple two equation system. You can use the default EViews coefficients, C(1), C(2), and so on, or you can use other coefficient vectors, in which case you should first declare them by clicking **Objects/New Object/Matrix-Vector-Coef/Coefficient Vector** in the main menu.



There are some general rules for specifying your equations:

- Equations can be nonlinear in their variables, coefficients, or both. Cross equation coefficient restrictions may be imposed by using the same coefficients in different equations. For example,

$$\begin{aligned}y &= c(1) + c(2) * x \\z &= c(3) + c(2) * z + (1 - c(2)) * x\end{aligned}$$

- You may also impose adding up constraints. Suppose for the equation

$$y = c(1) * x_1 + c(2) * x_2 + c(3) * x_3$$

you wish to impose $C(1) + C(2) + C(3) = 1$. You can impose this restriction by specifying the equation as

$$y = c(1) * x_1 + c(2) * x_2 + (1 - c(1) - c(2)) * x_3$$

- The equations in a system may contain autoregressive (AR) error specifications, but not MA, SAR, or SMA error specifications. You must associate coefficients with each AR specification. Enclose the entire AR specification in square brackets and follow each AR with an “=”-sign and a coefficient. For example,

$$cs = c(1) + c(2) * gdp + [ar(1)=c(3), ar(2)=c(4)]$$

You can constrain all of the equations in a system to have the same AR coefficient by giving all equations the same AR coefficient number, or you can estimate separate AR processes, by assigning each equation its own coefficient.

- Equations in a system need not have a dependent variable followed by an equal sign and then an expression. The “=”-sign can be anywhere in the formula, as in:

$$\log(\text{unemp} / (1 - \text{unemp})) = c(1) + c(2) * \text{dmr}$$

You can also write the equation as a simple expression without a dependent variable, as in

$$(c(1) * x + c(2) * y + 4) ^ 2$$

When encountering an expression that does not contain an equal sign, EViews sets the entire expression equal to the implicit error term.

If an equation should not have a disturbance, it is an identity, and should not be included in a system. If necessary, you should solve out for any identities to obtain the behavioral equations.

You should make certain that there is no identity linking all of the disturbances in your system. For example, if each of your equations describes a fraction of a total, the sum of the equations will always equal one, and the sum of the disturbances will identically equal zero. You will need to drop one of these equations to avoid numerical problems.

Instruments

If you plan to estimate your system using two-stage least squares, three-stage least squares, or GMM, you must specify the instrumental variables to be used in estimation. There are several ways to specify your instruments, with the appropriate form depending on whether you wish to have identical instruments in each equation, and whether you wish to compute the projections on an equation-by-equation basis, or whether you wish to compute a restricted projection using the stacked system.

In the simplest (default) case, EViews will form your instrumental variable projections on an equation-by-equation basis. If you prefer to think of this process as a two-step (2SLS) procedure, the first-stage regression of the variables in your model on the instruments will be run separately for each equation.

In this setting, there are two ways to specify your instruments. If you would like to use identical instruments in every equations, you should include a line beginning with the keyword “@INST” or “INST”, followed by a list of all the exogenous variables to be used as instruments. For example, the line

```
@inst gdp(-1 to -4) x gov
```

instructs EViews to use these six variables as instruments for all of the equations in the system. System estimation will involve a separate projection for each equation in your system.

You may also specify different instruments for each equation by appending an “@”-sign at the end of the equation, followed by a list of instruments for that equation. For example,

```
cs = c(1)+c(2)*gdp+c(3)*cs(-1) @ cs(-1) inv(-1) gov
inv = c(4)+c(5)*gdp+c(6)*gov @ gdp(-1) gov
```

The first equation uses CS(-1), INV(-1), GOV, and a constant as instruments, while the second equation uses GDP(-1), GOV, and a constant as instruments.

Lastly, you can mix the two methods. Any equation without individually specified instruments will use the instruments specified by the `@inst` statement. The system

```
@inst gdp(-1 to -4) x gov
cs = c(1)+c(2)*gdp+c(3)*cs(-1)
inv = c(4)+c(5)*gdp+c(6)*gov @ gdp(-1) gov
```

will use the instruments GDP(-1), GDP(-2), GDP(-3), GDP(-4), X, GOV, and C, for the CS equation, but only GDP(-1), GOV, and C, for the INV equation.

As noted above, the EViews default behavior is to perform the instrumental variables projection on an equation-by-equation basis. You may, however, wish to perform the projections on the stacked system. Notably, where the number of instruments is large, relative to the number of observations, stacking the equations and instruments prior to performing the projection may be the only feasible way to compute 2SLS estimates.

To designate instruments for a stacked projection, you should use the `@stackinst` statement (note: this statement is only available for systems estimated by 2SLS or 3SLS; it is not available for systems estimated using GMM).

In a `@stackinst` statement, the “@STACKINST” keyword should be followed by a list of stacked instrument specifications. Each specification is a comma delimited list of series enclosed in parentheses (one per equation), describing the instruments to be constrained in a stacked specification.

For example, the following `@stackinst` specification creates two instruments in a three equation model:

```
@stackinst (z1, z2, z3) (m1, m1, m1)
```

This statement instructs EViews to form two stacked instruments, one by stacking the separate series Z1, Z2, and Z3, and the other formed by stacking M1 three times. The first-stage instrumental variables projection is then of the variables in the stacked system on the stacked instruments.

When working with systems that have a large number of equations, the above syntax may be unwieldy. For these cases, EViews provides a couple of shortcuts. First, for instruments that are identical in all equations, you may use an “*” after the comma to instruct EViews to repeat the specified series. Thus, the above statement is equivalent to

```
@stackinst (z1, z2, z3) (m1, *)
```

Second, for non-identical instruments, you may specify a set of stacked instruments using an EViews group object, so long as the number of variables in the group is equal to the number of equations in the system. Thus, if you create a group Z with

```
group z z1 z2 z3
```

the above statement can be simplified to:

```
@stackinst z (m1,*)
```

You can, of course, combine ordinary instrument and stacked instrument specifications. This situation is equivalent to having common and equation specific coefficients for variables in your system. Simply think of the stacked instruments as representing common (coefficient) instruments, and ordinary instruments as representing equation specific (coefficient) instruments. For example, consider the system given by

```
@stackinst (z1,z2,z3) (m1,*)
@inst ia
y1 = c(1)*x1
y2 = c(1)*x2
y3 = c(1)*x3 @ ic
```

The stacked instruments for this specification may be represented as:

$$\begin{bmatrix} Z1 & M1 & IA & 0 & 0 & 0 \\ Z2 & M1 & 0 & IA & 0 & 0 \\ Z3 & M1 & 0 & 0 & IA & IC \end{bmatrix} \quad (19.2)$$

so it is easy to see that this specification is equivalent to the following stacked specification

```
@stackinst (z1, z2, z3) (m1, *) (ia, 0, 0) (0, ia, 0) (0, 0, ia)
(0, 0, ic)
```

since the common instrument specification

```
@inst ia
```

is equivalent to

```
@stackinst (ia, 0, 0) (0, ia, 0) (0, 0, ia)
```

Additional Comments

- If you include a “C” in the stacked instrument list, it will not be included in the individual equations. If you do not include the “C” as a stacked instrument, it will be included as an instrument in every equation, whether specified explicitly or not.
- You should list all exogenous right-hand side variables as instruments for a given equation.

- Identification requires that there should be at least as many instruments (including the constant) in each equation as there are right-hand side variables in that equation.
- The `@stackinst` statement is only available for estimation by 2SLS and 3SLS. It is not currently supported for GMM.
- If you estimate your system using a method that does not use instruments, all instrument specification lines will be ignored.

Starting Values

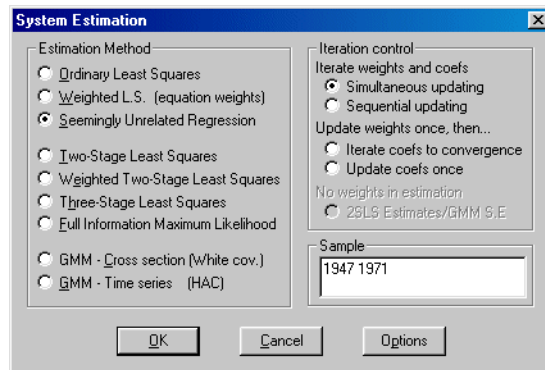
For systems that contain nonlinear equations, you can include a line that begins with `param` to provide starting values for some or all of the parameters. List pairs of parameters and values. For example,

```
param c(1) .15 b(3) .5
```

sets the initial values of `C(1)` and `B(3)`. If you do not provide starting values, EViews uses the values in the current coefficient vector.

How to Estimate a System

Once you have created and specified your system, push the **Estimate** button on the toolbar.



First select the estimation method. The **GMM-Cross section** option uses a weighting matrix that is robust to heteroskedasticity and contemporaneous correlation of unknown form, while the **GMM-Time series (HAC)** option extends this robustness to autocorrelation of unknown form.

When you select the **GMM-Time series (HAC)** option, the dialog box expands to display additional options for specifying the weighting matrix. The new options will appear at the lower right part of the dialog control. These options control the computation of the heteroskedasticity and autocorrelation robust (HAC) weighting matrix. See “[Technical Discussion](#)” on page 511 for a more detailed discussion of these options.

The **Kernel Options** determines the functional form of the kernel used to weight the autocovariances to compute the weighting matrix. The **Bandwidth Selection** option determines how the weights given by the kernel change with the lags of the autocovariances in the computation of the weighting matrix. If you select **Fixed** bandwidth, you may enter a number for the bandwidth or type `nw` to use Newey and West’s fixed bandwidth selection criterion.

The **Prewhitening** option runs a preliminary VAR(1) prior to estimation to “soak up” the correlation in the moment conditions.

Iteration Control

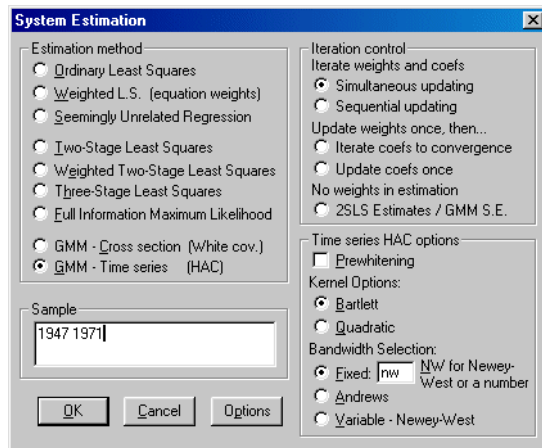
For weighted least squares, SUR, weighted TSLS, 3SLS, GMM, and nonlinear systems of equations, there is an additional estimation issue involving the procedure for computing the GLS weighting matrix and the coefficient vector. In general, you can elect to use the EViews default option with no additional difficulties, but there may be times when you wish to exercise greater control over the computation.

The estimation option controls the method of iterating over coefficients, over the weighting matrices, or both:

- **One-Step Weighting Matrix—Iterate Coefs** is the default method.

By default, EViews carries out a first-stage estimation of the coefficients using no weighting matrix (the identity matrix). Using starting values obtained from OLS (or TSLS, if there are instruments), EViews iterates until the coefficients converge. If the model is linear, this procedure involves a single OLS or TSLS regression.

The residuals from this first-stage iteration are used to form a consistent estimate of the weighting matrix.



In the second stage of the procedure, EViews uses the estimated weighting matrix in forming new estimates of the coefficients. If the model is nonlinear, EViews iterates the coefficient estimates until convergence.

- **One-Step Weighting Matrix—One-Step Coefs** performs the first-stage estimation of the coefficients, and constructs an estimate of the weighting matrix. In the second stage, EViews does not iterate the coefficients to convergence, instead performing a single coefficient iteration step.
- **Iterate Weights and Coefs—Sequential** repeats the default method, described above, until both the coefficients and the weighting matrix converge.
- **Iterate Weights and Coefs—Simultaneous** updates both the coefficients and the weighting matrix at each iteration. These steps are then repeated until both the coefficients and weighting matrix converge. This is the iteration method employed in previous versions of EViews.

Note that all four of the estimation techniques yield results that are asymptotically efficient. For linear models, the two **Iterate Weights and Coefs** options are equivalent to each other, and the two **One-Step Weighting Matrix** options are equivalent to each other, since obtaining coefficient estimates does not require iteration.

There is a last technique which is not fully efficient, but which does produce estimated covariances and standard errors which are robust to heteroskedasticity or heteroskedasticity and serial correlation.

- The **2SLS Estimates/GMM S.E.** computes first-stage estimates of the parameters assuming an identity weighting matrix. These estimates are then used to compute a coefficient covariance matrix that is robust to cross-section heteroskedasticity (White) or heteroskedasticity and autocorrelation (Newey-West).

Other Options

The **Options** button in the System Estimation dialog allows you to set a number of options for estimation, including convergence criterion, maximum number of iterations, and derivative calculation settings. See [“Setting Estimation Options” on page 666](#) for additional discussion of estimation options.

Estimation Output

The system estimation output contains parameter estimates, standard errors, and *t*-statistics for each of the coefficients in the system. Additionally, EViews reports the determinant of the residual covariance matrix, and, for FIML estimates, the maximized likelihood value.

In addition, EViews reports a set of summary statistics for each equation. The R^2 statistic, Durbin-Watson statistic, standard error of the regression, sum-of-squared residuals, etc., are computed for each equation using the standard definitions, based on the residuals from the system estimation procedure.

You may access most of these results using regression statistics functions. See [Chapter 11, page 270](#) for a discussion of the use of these functions, and [Chapter 3, “Object, View and Procedure Reference”, on page 19](#) of the *Command and Programming Reference* for a full listing of the available functions for systems.

Working With Systems

After obtaining estimates, the system object provides a number of tools for examining the equation results, and performing inference and specification testing.

System Views

Some of the system views are familiar from the discussion in previous chapters:

- You can examine the estimated covariance matrix by selecting the **Coefficient Covariance Matrix** view.
- **Wald Coefficient Tests...** performs hypothesis tests on the coefficients. These views are discussed in greater depth in [“Wald Test \(Coefficient Restrictions\)” on page 368](#).
- The **Estimation Output** view displays the coefficient estimates and summary statistics for the system. You may also access this view by pressing **Stats** on the system toolbar.

Other views are very familiar, but differ slightly in name or output, from their single equation counterparts:

- **System Specification** displays the specification window for the system. The specification window may also be displayed by pressing **Spec** on the toolbar.
- **Residual Graphs** displays a separate graph of the residuals from each equation in the system.
- **Endogenous Table** presents a spreadsheet view of the endogenous variables in the system.
- **Endogenous Graph** displays graphs of each of the endogenous variables.

The last two views are specific to systems:

- **Residual Correlation Matrix** computes the contemporaneous correlation matrix for the residuals of each equation.

- **Residual Covariance Matrix** computes the contemporaneous covariance matrix for the residuals. See also the function `@residcov` in [Chapter 3, “Object, View and Procedure Reference”](#), on page 45 of the *Command and Programming Reference*.

System Procs

One notable difference between systems and single equation objects is that there is no forecast procedure for systems. To forecast or perform simulation using an estimated system, you must use a model object.

EViews provides you with a simple method of incorporating the results of a system into a model. If you select **Procs/Make Model**, EViews will open an untitled model object containing the estimated system. This model can be used for forecasting and simulation. An alternative approach, creating the model and including the system object by name, is described in [“Building a Model”](#) on page 618.

There are other procedures for working with the system:

- **Estimate...** opens the dialog for estimating the system of equations. It may also be accessed by pressing **Estimate** on the system toolbar.
- **Make Residuals** creates a number of series containing the residuals for each equation in the system. The residuals will be given the next unused name of the form RESID01, RESID02, etc., in the order that the equations are specified in the system.
- **Make Endogenous Group** creates an untitled group object containing the endogenous variables.

Example

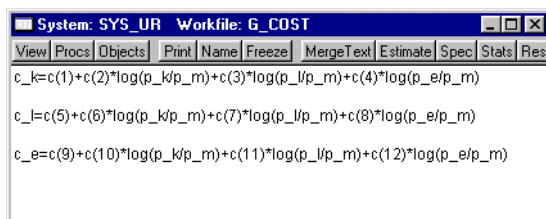
As an illustration of the process of estimating a system of equations in EViews, we estimate a translog cost function using data from Berndt and Wood (1975) as tabulated in Greene (1997). The translog cost function has four factors with three equations of the form:

$$\begin{aligned}
 c_K &= \beta_K + \delta_{KK} \log\left(\frac{p_K}{p_M}\right) + \delta_{KL} \log\left(\frac{p_L}{p_M}\right) + \delta_{KE} \log\left(\frac{p_E}{p_M}\right) + \epsilon_K \\
 c_L &= \beta_L + \delta_{LK} \log\left(\frac{p_K}{p_M}\right) + \delta_{LL} \log\left(\frac{p_L}{p_M}\right) + \delta_{LE} \log\left(\frac{p_E}{p_M}\right) + \epsilon_L \\
 c_E &= \beta_E + \delta_{EK} \log\left(\frac{p_K}{p_M}\right) + \delta_{EL} \log\left(\frac{p_L}{p_M}\right) + \delta_{EE} \log\left(\frac{p_E}{p_M}\right) + \epsilon_E
 \end{aligned} \tag{19.3}$$

where c_i and p_i are the cost share and price of factor i , respectively. β and δ are the parameters to be estimated. Note that there are cross equation coefficient restrictions that ensure symmetry of the cross partial derivatives.

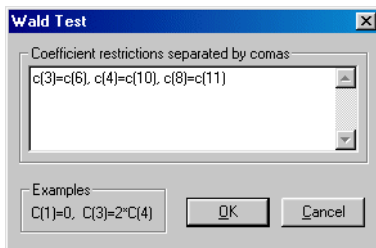
We first estimate this system without imposing the cross equation restrictions and test whether the symmetry restrictions hold. Create a system by clicking **Objects/New Object/System** in the main toolbar or type `system` in the command window. Press the **Name** button and type in the name “SYS_UR” to name the system.

Next, type in the system window and specify the system as:



We estimate this model by full information maximum likelihood (FIML). FIML is invariant to the equation that is dropped. Press the **Estimate** button and choose **Full Information Maximum Likelihood**. EViews presents the estimated coefficients and regression statistics for each equation.

To test the symmetry restrictions, select **View/Wald Coefficient Tests...**, fill in the dialog:



and click OK. The result:

Wald Test:
System: SYS_UR

Test Statistic	Value	df	Probability
Chi-square	0.431610	3	0.9336

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(3) - C(6)	-0.010891	35.74745
C(4) - C(10)	0.030891	23.35663
C(8) - C(11)	0.057997	21.16539

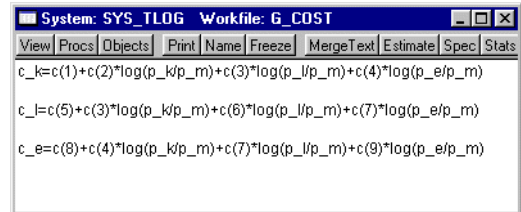
Restrictions are linear in coefficients.

fails to reject the symmetry restrictions. To estimate the system imposing the symmetry restrictions, copy the object using **Objects/Copy Object**, click **View/System Specification** and modify the system.

We have named the system

SYS_TLOG. Note that to impose symmetry in the translog specification, we have restricted the coefficients on the cross-price terms to be the same (we have also renumbered the 9 remaining coefficients so that they are consecutive).

The restrictions are imposed by using the same coefficients in each equation. For example, the coefficient on the $\log(P_L/P_M)$ term in the C_K equation, $C(3)$, is the same as the coefficient on the $\log(P_K/P_M)$ term in the C_L equation.



To estimate this model using FIML, click **Estimate** and choose **Full Information Maximum Likelihood**. The top part of the equation describes the estimation specification, and provides coefficient and standard error estimates, t -statistics, p -values, and summary statistics:

System: SYS_TLOG
 Estimation Method: Full Information Maximum Likelihood (Marquardt)
 Date: 08/07/97 Time: 13:17
 Sample: 1947 1971
 Convergence achieved after 9 iterations

	Coefficient	Std. Error	t-Statistic	Prob.
C(1)	0.056672	0.002803	20.21585	0.0000
C(2)	0.029082	0.008310	3.499789	0.0008
C(3)	0.000274	0.009810	0.027925	0.9778
C(4)	-0.010639	0.004931	-2.157489	0.0346
C(5)	0.252923	0.002172	116.4713	0.0000
C(6)	0.076621	0.010717	7.149671	0.0000
C(7)	-0.003500	0.007603	-0.460326	0.6468
C(8)	0.043860	0.002651	16.54573	0.0000
C(9)	0.020018	0.011049	1.811797	0.0746
Log Likelihood		344.4730		
Determinant residual covariance		2.16E-16		

The log likelihood value reported at the bottom of the first part of the table can be used to construct likelihood ratio tests.

Since maximum likelihood assumes the errors are multivariate normal, we may wish to test whether the residuals are normally distributed. Click **Procs/Make Residuals** and EViews opens an untitled group window containing the residuals of each equation in the system. Then to compute descriptive statistics for each residual in the group, select **View/Descriptive Stats** from the group window toolbar:

The Jarque-Bera statistic rejects the hypothesis of normal distribution for the second equation but not for the other equations.

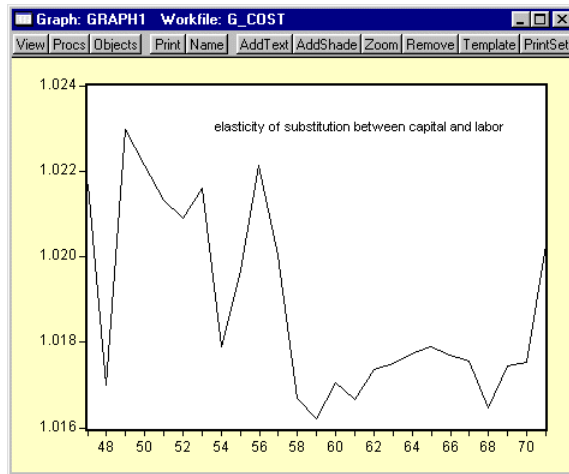
The estimated coefficients of the translog cost function can be used to construct estimates of the elasticity of substitution between factors of production. For example, the elasticity of substitution between capital and labor is given by $1 + c(3)/(C_K * C_L)$. Note that the elasticity of substitution

is not a constant, and depends on the values of C_K and C_L . To create a series containing the elasticities computed for each observation, select **Quick/Generate Series...**, and enter:

$$es_k1 = 1 + sys1.c(3) / (c_k * c_l)$$

To plot the series of elasticity of substitution between capital and labor for each observation, double click on the series name ES_KL in the workfile and select **View/Line Graph**:

Group: UNTITLED Workfile: G_COST			
View	Procs	Objects	Print Name Freeze Sample Sheet Stats Spec
	RESID01	RESID02	RESID03
Mean	0.000118	0.000168	7.84E-06
Median	-0.000451	0.000231	-0.000843
Maximum	0.007941	0.018095	0.002967
Minimum	-0.005602	-0.011877	-0.002677
Std. Dev.	0.003233	0.005456	0.001814
Skewness	0.712642	0.878560	0.133671
Kurtosis	2.952042	6.464994	1.505250
Jarque-Bera	2.118471	15.72240	2.401710
Probability	0.346721	0.000385	0.300937
Sum	0.002980	0.004210	0.001960
Sum Sq. Dev.	0.000251	0.000714	7.90E-05
Observations	25	25	25



While it varies over the sample, the elasticity of substitution is generally close to one, which is consistent with the assumption of a Cobb-Douglas cost function.

Commands

To create a new system, follow the system command with a name for the system:


```
system demand1
```

creates and opens the window for a new system named DEMAND1.

To estimate a system, follow the name of the system with a dot and the name of the estimation procedure. For example,

```
sys1.fiml
```

estimates SYS1 by full information maximum likelihood.

See “System” on page 45 of the *Command and Programming Reference* for a complete list of commands and options available for system objects.

Technical Discussion

While the discussion to follow is expressed in terms of a balanced system of linear equations, the analysis carries forward in a straightforward way to unbalanced systems containing nonlinear equations.

Denote a system of m equations, in stacked form, as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} X_1 & 0 & \dots & 0 \\ 0 & X_2 & & \vdots \\ & & \ddots & 0 \\ 0 & \dots & 0 & X_M \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_M \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_M \end{bmatrix} \quad (19.4)$$

where y_m is T vector, X_m is a $T \times k_m$ matrix, and β_m is a k_m vector of coefficients. The error terms ϵ have an $MT \times MT$ covariance matrix V . The system may be written in compact form as:

$$y = X\beta + \epsilon. \quad (19.5)$$

Under the standard assumptions, the residual variance matrix from this stacked system is given by

$$V = E(\epsilon\epsilon') = \sigma^2(I_M \otimes I_T). \quad (19.6)$$

Other residual structures are of interest. First, the errors may be heteroskedastic across the m equations. Second, they may be heteroskedastic and contemporaneously correlated. We can characterize both of these cases by defining the $M \times M$ matrix of contemporaneous correlations, Σ , where the (i,j) -th element of Σ is given by $\sigma_{ij} = E(\epsilon_{it}\epsilon_{jt})$ for all t . If the errors are contemporaneously uncorrelated, then, $\sigma_{ij} = 0$ for $i \neq j$, and we can write:

$$V = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_M^2) \otimes I_T \quad (19.7)$$

More generally, if the errors are heteroskedastic and contemporaneously correlated:

$$V = \Sigma \otimes I_T. \quad (19.8)$$

Lastly, at the most general level, there may be heteroskedasticity, contemporaneous correlation, and autocorrelation of the residuals. The general variance matrix of the residuals may be written:

$$V = \begin{pmatrix} \sigma_{11}\Sigma_{11} & \sigma_{12}\Sigma_{12} & \dots & \sigma_{1M}\Sigma_{1M} \\ \sigma_{21}\Sigma_{21} & \sigma_{22}\Sigma_{22} & & \vdots \\ & & \ddots & \\ \sigma_{M1}\Sigma_{M1} & \dots & & \sigma_{MM}\Sigma_{MM} \end{pmatrix} \quad (19.9)$$

where Σ_{ij} is an autocorrelation matrix for the i -th and j -th equations.

Ordinary Least Squares

The OLS estimator of the estimated variance matrix of the parameters is valid under the assumption that $V = \Sigma \otimes I_T$. The estimator for β is given by:

$$b_{LS} = (X'X)^{-1}X'y, \quad (19.10)$$

and the variance estimator is given by

$$\text{var}(b_{LS}) = s^2(X'X)^{-1}, \quad (19.11)$$

where s^2 is the residual variance estimate for the stacked system.

Weighted Least Squares

The weighted least squares estimator is given by:

$$b_{WLS} = (X'\hat{V}^{-1}X)^{-1}X'\hat{V}^{-1}y \quad (19.12)$$

where $\hat{V} = \text{diag}(s_{11}, s_{22}, \dots, s_{MM}) \otimes I_T$ is a consistent estimator of V , and s_{ii} is the residual variance estimator,

$$s_{ij} = ((y_i - X_i b_{LS})'(y_j - X_j b_{LS}))/\max(T_i, T_j) \quad (19.13)$$

where the inner product is taken over the non-missing common elements of i and j . The \max function in Equation (19.13) is designed to handle the case of unbalanced data by down-weighting the covariance terms. Provided the missing values are asymptotically negligible, this yields a consistent estimator of the variance elements. Note also that there is no adjustment for degrees-of-freedom.

When specifying your estimation specification, you are given a choice of which coefficients to use in computing the s_{ij} . If you choose not to iterate the weights, the OLS coefficient estimates will be used to estimate the variances. If you choose to iterate the weights,

the current parameter estimates (which may be based on the previously computed weights) are used in computing the s_{ij} . This latter procedure may be iterated until the weights and coefficients converge.

The estimator for the coefficient variance matrix is:

$$\text{var}(b_{WLS}) = (X' \hat{V}^{-1} X)^{-1}. \quad (19.14)$$

The weighted least squares estimator is efficient, and the variance estimator consistent, under the assumption that there is heteroskedasticity, but no serial or contemporaneous correlation in the residuals.

It is worth pointing out that if there are no cross-equation restrictions on the parameters of the model, weighted LS on the entire system yields estimates that are identical to those obtained by equation-by-equation LS. Consider the following simple model:

$$\begin{aligned} y_1 &= X_1 \beta_1 + \epsilon_1 \\ y_2 &= X_2 \beta_2 + \epsilon_2 \end{aligned} \quad (19.15)$$

If β_1 and β_2 are unrestricted, the WLS estimator given in Equation (19.14) yields,

$$b_{WLS} = \begin{bmatrix} ((X_1' X_1) / s_{11})^{-1} ((X_1' y_1) / s_{11}) \\ ((X_2' X_2) / s_{22})^{-1} ((X_2' y_2) / s_{22}) \end{bmatrix} = \begin{bmatrix} (X_1' X_1)^{-1} X_1' y_1 \\ (X_2' X_2)^{-1} X_2' y_2 \end{bmatrix}. \quad (19.16)$$

The expression on the right is equivalent to equation-by-equation OLS. Note, however, that even without cross-equation restrictions, the standard errors are not the same in the two cases.

Seemingly Unrelated Regression (SUR)

SUR is appropriate when all the right-hand side regressors X are assumed to be exogenous, and the errors are heteroskedastic and contemporaneously correlated so that the error variance matrix is given by $V = \Sigma \otimes I_T$. Zellner's SUR estimator of β takes the form

$$b_{SUR} = (X'(\hat{\Sigma} \otimes I_T)^{-1} X)^{-1} X'(\hat{\Sigma} \otimes I_T)^{-1} y, \quad (19.17)$$

where $\hat{\Sigma}$ is a consistent estimate of Σ with typical element s_{ij} , for all i and j .

If you include AR terms in equation j , EViews transforms the model (see “[Estimating AR Models](#)” on page 307) and estimates the following equation:

$$y_{jt} = X_{jt} \beta_j + \left(\sum_{r=1}^{p_j} \rho_{jr} (y_{j(t-r)} - X_{j(t-r)}) \right) + \epsilon_{jt} \quad (19.18)$$

where ϵ_j is assumed to be serially independent, but possibly correlated contemporaneously across equations. At the beginning of the first iteration, we estimate the equation by

nonlinear LS and use the estimates to compute the residuals $\hat{\epsilon}$. We then construct an estimate of Σ using $s_{ij} = (\hat{\epsilon}_i' \hat{\epsilon}_j) / \max(T_i, T_j)$ and perform nonlinear GLS to complete one iteration of the estimation procedure. These iterations may be repeated until the coefficients and weights converge.

Two-Stage Least Squares (TSLS) and Weighted TSLS

TSLS is a single equation estimation method that is appropriate when some of the variables in X are endogenous. Write the j -th equation of the system as

$$Y\Gamma_j + XB_j + \epsilon_j = 0 \quad (19.19)$$

or, alternatively,

$$y_j = Y_j\gamma_j + X_j\beta_j + \epsilon_j = Z_j\delta_j + \epsilon_j \quad (19.20)$$

where $\Gamma_j' = (-1, \gamma_j', 0)$, $B_j' = (\beta_j', 0)$, $Z_j' = (Y_j', X_j')$ and $\delta_j' = (\gamma_j', \beta_j')$. Y is the matrix of endogenous variables and X is the matrix of exogenous variables.

In the first stage, we regress the right-hand side endogenous variables y_j on all exogenous variables X and get the fitted values

$$\hat{Y}_j = X(X'X)^{-1}X'Xy_j. \quad (19.21)$$

In the second stage, we regress y_j on \hat{Y}_j and X_j to get

$$\hat{\delta}_{2SLS} = (\hat{Z}_j' \hat{Z}_j)^{-1} \hat{Z}_j' y_j. \quad (19.22)$$

where $\hat{Z}_j = (\hat{Y}_j, X_j)$.

Weighted TSLS applies the weights in the second stage so that

$$\hat{\delta}_{W2SLS} = (\hat{Z}_j' \hat{V}^{-1} \hat{Z}_j)^{-1} \hat{Z}_j' \hat{V}^{-1} y_j \quad (19.23)$$

where the elements of the variance matrix are estimated in the usual fashion using the residuals from unweighted TSLS.

If you choose to iterate the weights, X is estimated at each step using the current values of the coefficients and residuals.

Three-Stage Least Squares (3SLS)

Since TSLS is a single equation estimator that does not take account of the covariances between residuals, it is not, in general, fully efficient. 3SLS is a system method that estimates all of the coefficients of the model, then forms weights and reestimates the model using the estimated weighting matrix. It should be viewed as the endogenous variable analogue to the SUR estimator described above.

The first two stages of 3SLS are the same as in TSLS. In the third stage, we apply feasible generalized least squares (FGLS) to the equations in the system in a manner analogous to the SUR estimator.

SUR uses the OLS residuals to obtain a consistent estimate of the cross-equation covariance matrix Σ . This covariance estimator is not, however, consistent if any of the right-hand side variables are endogenous. 3SLS uses the 2SLS residuals to obtain a consistent estimate of Σ .

$$\hat{\delta}_{3SLS} = (Z(\hat{\Sigma}^{-1} \otimes X(X'X)^{-1}X')Z)^{-1}Z(\hat{\Sigma}^{-1} \otimes X(X'X)^{-1}X')y, \quad (19.24)$$

where $\hat{\Sigma}$ has typical element

$$s_{ij} = ((y_i - Z_i\hat{\gamma}_{2SLS})'(y_j - Z_j\hat{\gamma}_{2SLS}))/\max(T_i, T_j). \quad (19.25)$$

If you choose to iterate the weights, the current coefficients and residuals will be used to estimate $\hat{\Sigma}$.

Generalized Method of Moments (GMM)

The basic idea underlying GMM is simple and intuitive. We have a set of theoretical moment conditions that the parameters of interest θ should satisfy. We denote these moment conditions as

$$E(m(y, \theta)) = 0. \quad (19.26)$$

The method of moments estimator is defined by replacing the moment condition (19.26) by its sample analog

$$\left(\sum_t m(y_t, \theta) \right) / T = 0. \quad (19.27)$$

However, condition (19.27) will not be satisfied for any θ when there are more restrictions m than there are parameters θ . To allow for such overidentification, the GMM estimator is defined by minimizing the following criterion function:

$$\sum_t m(y_t, \theta)A(y_t, \theta)m(y_t, \theta) \quad (19.28)$$

which measures the “distance” between m and zero. A is a weighting matrix that weights each moment condition. Any symmetric positive definite matrix A will yield a consistent estimate of θ . However, it can be shown that a necessary (but not sufficient) condition to obtain an (asymptotically) efficient estimate of θ is to set A equal to the inverse of the covariance matrix Ω of the sample moments m . This follows intuitively, since we want to put less weight on the conditions that are more imprecise.

To obtain GMM estimates in EViews, you must be able to write the moment conditions in Equation (19.26) as an orthogonality condition between the residuals of a regression equation, $u(y, \theta, X)$, and a set of instrumental variables, Z , so that

$$m(\theta, y, X, Z) = Z'u(\theta, y, X) \quad (19.29)$$

For example, the OLS estimator is obtained as a GMM estimator with the orthogonality conditions

$$X'(y - X\beta) = 0. \quad (19.30)$$

For the GMM estimator to be identified, there must be at least as many instrumental variables Z as there are parameters θ . See the section on “[Generalized Method of Moments \(GMM\)](#)” beginning on page 297 for additional examples of GMM orthogonality conditions.

An important aspect of specifying a GMM problem is the choice of the weighting matrix A . EViews uses the optimal $A = \hat{\Omega}^{-1}$, where $\hat{\Omega}$ is the estimated covariance matrix of the sample moments m . EViews uses the consistent TSLS estimates for the initial estimate of θ in forming the estimate of Ω .

White’s Heteroskedasticity Consistent Covariance Matrix

If you choose the **GMM-Cross section** option, EViews estimates Ω using White’s heteroskedasticity consistent covariance matrix

$$\hat{\Omega}_W = \hat{\Gamma}(0) = \frac{1}{T-k} \left(\sum_{t=1}^T Z_t' u_t u_t' Z_t \right) \quad (19.31)$$

where u is the vector of residuals, and Z_t is a $k \times p$ matrix such that the p moment conditions at t may be written as $m(\theta, y_t, X_t, Z_t) = Z_t' u(\theta, y_t, X_t)$.

Heteroskedasticity and Autocorrelation Consistent (HAC) Covariance Matrix

If you choose the **GMM-Time series** option, EViews estimates Ω by

$$\hat{\Omega}_{HAC} = \hat{\Gamma}(0) + \left(\sum_{j=1}^{T-1} k(j, q) (\hat{\Gamma}(j) - \hat{\Gamma}'(j)) \right) \quad (19.32)$$

where

$$\hat{\Gamma}(j) = \frac{1}{T-k} \left(\sum_{t=j+1}^T Z_{t-j}' u_t u_{t-j}' Z_t \right). \quad (19.33)$$

You also need to specify the *kernel* κ and the *bandwidth* q .

Kernel Options

The kernel κ is used to weight the covariances so that $\hat{\Omega}$ is ensured to be positive semi-definite. EViews provides two choices for the kernel, Bartlett and quadratic spectral (QS). The Bartlett kernel is given by

$$\kappa(j, q) = \begin{cases} 1 - (j/q) & 0 \leq j \leq q \\ 0 & \text{otherwise} \end{cases} \quad (19.34)$$

while the quadratic spectral (QS) kernel is given by

$$k(j/q) = \frac{25}{12(\pi x)^2} \left(\frac{\sin(6\pi x/5)}{6\pi x/5} - \cos(6\pi x/5) \right) \quad (19.35)$$

where $x = j/q$. The QS has a faster rate of convergence than the Bartlett and is smooth and not truncated (Andrews 1991). Note that even though the QS kernel is not truncated, it still depends on the bandwidth q (which need not be an integer).

Bandwidth Selection

The bandwidth q determines how the weights given by the kernel change with the lags in the estimation of Ω . Newey-West fixed bandwidth is based solely on the number of observations in the sample and is given by

$$q = \text{int}(4(T/100)^{2/9}) \quad (19.36)$$

where $\text{int}(\cdot)$ denotes the integer part of the argument.

EViews also provides two “automatic”, or data dependent bandwidth selection methods that are based on the autocorrelations in the data. Both methods select the bandwidth according to

$$q = \begin{cases} \text{int}(1.1447(\hat{\alpha}(1)T)^{1/3}) & \text{for the Bartlett kernel} \\ 1.3221(\hat{\alpha}(2)T)^{1/5} & \text{for the QS kernel} \end{cases} \quad (19.37)$$

The two methods, Andrews and Variable-Newey-West, differ in how they estimate $\hat{\alpha}(1)$ and $\hat{\alpha}(2)$.

Andrews (1991) is a parametric method that assumes the sample moments follow an AR(1) process. We first fit an AR(1) to each sample moment (19.29) and estimate the autocorrelation coefficients $\hat{\rho}_i$ and the residual variances $\hat{\sigma}_i^2$ for $i = 1, 2, \dots, zn$, where z is the number of instrumental variables and n is the number of equations in the system. Then $\hat{\alpha}(1)$ and $\hat{\alpha}(2)$ are estimated by

$$\begin{aligned} \hat{\alpha}(1) &= \left(\sum_{i=1}^{zn} \frac{4\hat{\rho}_i^2 \hat{\sigma}_i^4}{(1-\hat{\rho}_i)^6 (1+\hat{\rho}_i)^2} \right) / \left(\sum_{i=1}^{zn} \frac{\hat{\sigma}_i^4}{(1-\hat{\rho}_i)^4} \right) \\ \hat{\alpha}(2) &= \left(\sum_{i=1}^{zn} \frac{4\hat{\rho}_i^2 \hat{\sigma}_i^4}{(1-\hat{\rho}_i)^8} \right) / \left(\sum_{i=1}^{zn} \frac{\hat{\sigma}_i^4}{(1-\hat{\rho}_i)^4} \right) \end{aligned} \quad (19.38)$$

Note that we weight all moments equally, including the moment corresponding to the constant.

Newey-West (1994) is a nonparametric method based on a truncated weighted sum of the estimated cross-moments $\hat{T}(j)$. $\hat{\alpha}(1)$ and $\hat{\alpha}(2)$ are estimated by

$$\hat{\alpha}(p) = \left(\frac{l' F(p) l}{l' F(0) l} \right) \quad (19.39)$$

where l is a vector of ones and

$$F(p) = \Gamma(0) + \sum_{i=1}^L i^p (\hat{T}(i) + \hat{T}'(i)), \quad (19.40)$$

for $p = 1, 2$.

One practical problem with the Newey-West method is that we have to choose a lag selection parameter L . The choice of L is arbitrary, subject to the condition that it grow at a certain rate. EViews sets the lag parameter to

$$L = \begin{cases} \text{int}(4(T/100)^{2/9}) & \text{for the Bartlett kernel} \\ T & \text{for the QS kernel} \end{cases} \quad (19.41)$$

Prewhitening

You can also choose to prewhiten the sample moments m to “soak up” the correlations in m prior to GMM estimation. We first fit a VAR(1) to the sample moments

$$m_t = A m_{t-1} + v_t. \quad (19.42)$$

Then the variance $\hat{\Omega}$ of m is estimated by $\hat{\Omega} = (I - A)^{-1} \hat{\Omega}^* (I - A)^{-1}$ where $\hat{\Omega}^*$ is the variance of the residuals v_t and is computed using any of the above methods. The GMM estimator is then found by minimizing the criterion function:

$$u' Z \hat{\Omega}^{-1} Z' u \quad (19.43)$$

Note that while Andrews and Monahan (1992) adjust the VAR estimates to avoid singularity when the moments are near unit root processes, EViews does not perform this eigenvalue adjustment.

Chapter 20. Vector Autoregression and Error Correction Models

The structural approach to time series modeling uses economic theory to model the relationship among the variables of interest. Unfortunately, economic theory is often not rich enough to provide a dynamic specification that identifies all of these relationships. Furthermore, estimation and inference are complicated by the fact that endogenous variables may appear on both the left and right sides of equations.

These problems lead to alternative, non-structural approaches to modeling the relationship among several variables. This chapter describes the estimation and analysis of vector autoregression (VAR) and the vector error correction (VEC) models. We also describe tools for testing the presence of cointegrating relationships among several non-stationary variables.

Vector Autoregressions (VARs)

The vector autoregression (VAR) is commonly used for forecasting systems of interrelated time series and for analyzing the dynamic impact of random disturbances on the system of variables. The VAR approach sidesteps the need for structural modeling by treating every endogenous variable in the system as a function of the lagged values of all of the endogenous variables in the system.

The mathematical representation of a VAR is

$$y_t = A_1 y_{t-1} + \dots + A_p y_{t-p} + B x_t + \epsilon_t \quad (20.1)$$

where y_t is a k vector of endogenous variables, x_t is a d vector of exogenous variables, A_1, \dots, A_p and B are matrices of coefficients to be estimated, and ϵ_t is a vector of innovations that may be contemporaneously correlated but are uncorrelated with their own lagged values and uncorrelated with all of the right-hand side variables.

Since only lagged values of the endogenous variables appear on the right-hand side of the equations, simultaneity is not an issue and OLS yields consistent estimates. Moreover, even though the innovations ϵ_t may be contemporaneously correlated, OLS is efficient and equivalent to GLS since all equations have identical regressors.

As an example, suppose that industrial production (IP) and money supply (M1) are jointly determined by a VAR and let a constant be the only exogenous variable. Assuming that the VAR contains two lagged values of the endogenous variables, it may be written as

$$\begin{aligned} IP_t &= a_{11}IP_{t-1} + a_{12}M1_{t-1} + b_{11}IP_{t-2} + b_{12}M1_{t-2} + c_1 + \epsilon_{1t} \\ M1_t &= a_{21}IP_{t-1} + a_{22}M1_{t-1} + b_{21}IP_{t-2} + b_{22}M1_{t-2} + c_2 + \epsilon_{2t} \end{aligned} \quad (20.2)$$

where a_{ij} , b_{ij} , c_i are the parameters to be estimated.

How to Estimate a VAR

To specify a VAR in EViews, you must first create a var object. Select **Quick/Estimate VAR...** or type `var` in the command window. The **Basics** tab of the VAR Specification dialog will prompt you to define the structure of your VAR.

You should fill out the dialog with the appropriate information:

- Select the VAR type: **Unrestricted VAR** or **Vector Error Correction (VEC)**. What we have been calling a VAR is actually an *unrestricted* VAR. VECs are explained below.
- Set the estimation sample.
- Enter the lag specification in the appropriate edit box. This information is entered in pairs: each pair of numbers defines a *range* of lags. For example, the lag pair shown above

1 4

tells EViews to use the first *through* fourth lags of all the endogenous variables in the system as right-hand side variables.

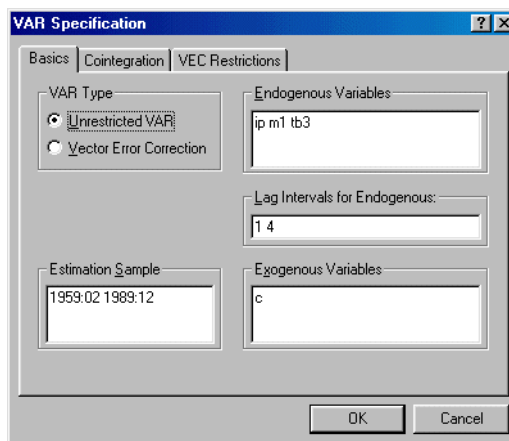
You can add any number of lag intervals, all entered in pairs. The lag specification

2 4 6 9 12 12

uses lags 2–4, 6–9, and 12.

- Enter the names of endogenous and exogenous series in the appropriate edit boxes. Here we have listed M1, IP, and TB3 as endogenous series, and have used the special series C as the constant exogenous term. If either list of series was longer, we could have created a named group object containing the list and then entered the group name.

The remaining dialog tabs (**Cointegration** and **Restrictions**) are relevant only for VEC models and are explained below.



VAR Estimation Output

Once you have specified the VAR, click **OK**. EViews will display the estimation results in the VAR window.

Each column in the table corresponds to an equation in the VAR. For each right-hand side variable, EViews reports the estimated coefficient, its standard error, and the t -statistic. For example, the coefficient for IP(-1) in the TB3 equation is 0.095984.

EViews displays additional information below the coefficient summary. The first part of the additional output presents standard OLS regression statistics for each equation. The results are

	IP	M1	TB3
IP(-1)	1.253934 (0.05401) [23.2147]	0.253215 (0.17769) [1.42501]	0.095984 (0.05021) [1.91170]
IP(-2)	-0.187774 (0.08557) [-2.19448]	-0.230187 (0.28149) [-0.81774]	0.015590 (0.07954) [0.19601]
IP(-3)	-0.003780 (0.08556) [-0.04423]	-0.153515 (0.28146) [-0.54547]	-0.173824 (0.07953) [-2.18673]

computed separately for each equation, using the appropriate residuals and are displayed in the corresponding column. The numbers at the very bottom of the table are the summary statistics for the VAR system as a whole.

R-squared	0.999221	0.999915	0.968018
Adj. R-squared	0.999195	0.999912	0.966937
Sum sq. resids	113.8813	1232.453	98.39849
S.E. equation	0.566385	1.863249	0.526478
Log likelihood	-306.3509	-744.5662	-279.4628
Akaike AIC	-1.102274	1.279331	-1.248405
Schwarz SC	-0.964217	1.417388	-1.110348
Mean dependent	70.97919	339.7451	6.333891
S.D. dependent	19.95932	198.6301	2.895381
Determinant Residual Covariance		0.259637	
Log Likelihood		-1318.390	
Akaike Information Criteria		-1.136514	
Schwarz Criteria		-0.722342	

The determinant of the residual covariance is computed as

$$|\hat{\Omega}| = \det\left(\frac{1}{T-p} \sum_t \hat{\epsilon}_t \hat{\epsilon}_t'\right) \quad (20.3)$$

where p is the number of parameters per equation in the VAR. The log likelihood value is computed assuming a multivariate normal (Gaussian) distribution as

$$l = -\frac{T}{2}\{k(1 + \log 2\pi) + \log|\hat{\Omega}|\} \quad (20.4)$$

and the two information criteria are computed as

$$\begin{aligned} AIC &= -2l/T + 2n/T \\ SC &= -2l/T + n \log T/T \end{aligned} \quad (20.5)$$

where $n = k(d + pk)$ is the total number of estimated parameters in the VAR. These information criteria can be used for model selection such as determining the lag length of the VAR, with smaller values of the information criterion being preferred. It is worth noting that some reference sources may define the AIC/SC differently, either omitting the “inessential” constant terms from the likelihood, or not dividing by T (see also [Appendix F, “Information Criteria”](#), on page 683 for additional discussion of information criteria).

Views and Procs of a VAR

Once you have estimated a VAR, EViews provides various views to work with the estimated VAR. In this section, we discuss views that are specific to VARs. For other views and procedures, see the general discussion of system views in [Chapter 19, “System Estimation”](#), beginning on page 495.

Diagnostic Views

A set of diagnostic views are provided under the menus **View/Lag Structure** and **View/Residual Tests** in the VAR window. These views should help you check the appropriateness of the estimated VAR.

Lag Structure

AR Roots Table/Graph

Reports the *inverse* roots of the characteristic AR polynomial; see Lütkepohl (1991). The estimated VAR is stable (stationary) if all roots have modulus less than one and lie inside the unit circle. If the VAR is not stable, certain results (such as impulse response standard errors) are not valid. There will be kp roots, where k is the number of endogenous variables and p is the largest lag. If you estimated a VEC with r cointegrating relations, $k - r$ roots should be equal to unity.

Pairwise Granger Causality Tests.

Carries out pairwise Granger causality tests and tests whether an endogenous variable can be treated as exogenous. For each equation in the VAR, the output displays χ^2 (Wald) statistics for the joint significance of each of the other lagged endogenous variables in that equation. The statistic in the last row (**All**) is the χ^2 statistic for joint significance of all other lagged endogenous variables in the equation.

Warning: if you have estimated a VEC, the lagged variables that are tested for exclusion are only those that are first differenced. The lagged level terms in the cointegrating equations (the error correction terms) are not tested.

Lag Exclusion Tests.

Carries out lag exclusion tests for each lag in the VAR. For each lag, the χ^2 (Wald) statistic for the joint significance of all endogenous variables at that lag is reported for each equation separately and jointly (last column).

Lag Length Criteria.

Computes various criteria to select the lag order of an unrestricted VAR. You will be prompted to specify the maximum lag to “test” for. The table displays various information criteria for all lags up to the specified maximum. (If there are no exogenous variables in the VAR, the lag starts at 1; otherwise the lag starts at 0.) The table indicates the selected lag from each column criterion by an asterisk “*”. For columns 4–7, these are the lags with the smallest value of the criterion.

All the criteria are discussed in Lütkepohl (1991, Section 4.3). The sequential modified likelihood ratio (LR) test is carried out as follows. Starting from the maximum lag, test the hypothesis that the coefficients on lag l are jointly zero using the χ^2 statistics

$$LR = (T - m) \{ \log |\Omega_{\ell-1}| - \log |\Omega_{\ell}| \} \sim \chi^2(k^2) \quad (20.6)$$

where m is the number of parameters per equation under the alternative. Note that we employ Sims’ (1980) small sample modification which uses $(T - m)$ rather than T . We compare the modified LR statistics to the 5% critical values starting from the maximum lag, and decreasing the lag one at a time until we first get a rejection. The alternative lag order from the first rejected test is marked with an asterisk (if no test rejects, the minimum lag will be marked with an asterisk). It is worth emphasizing that even though the individual tests have size 0.05, the overall size of the test will not be 5%; see the discussion in Lütkepohl (1991, pp. 125–126).

Residual Tests

Correlograms

Displays the pairwise cross-correlograms (sample autocorrelations) for the estimated residuals in the VAR for the specified number of lags. The cross-correlograms can be displayed in three different formats. There are two tabular forms, one ordered by variables (**Tabulate by Variable**) and one ordered by lags (**Tabulate by Lag**). The **Graph** form displays a matrix of pairwise cross-correlograms. The dotted line in the graphs represent plus or minus two times the asymptotic standard errors of the lagged correlations (computed as $1/\sqrt{T}$).

Portmanteau Autocorrelation Test

Computes the multivariate Box-Pierce/Ljung-Box Q -statistics for residual serial correlation up to the specified order (see Lütkepohl, 1991, 4.4.21 & 4.4.23 for details). We report both the Q -statistics and the adjusted Q -statistics (with a small sample correction). Under the null hypothesis of no serial correlation up to lag h , both statistics are *approximately* distributed χ^2 with degrees of freedom $k^2(h-p)$ where p is the VAR lag order. The asymptotic distribution is approximate in the sense that it requires the MA coefficients to be zero for lags $i > h-p$. Therefore, this approximation will be poor if the roots of the AR polynomial are close to one and h is small. In fact, the degrees of freedom becomes negative for $h < p$.

Autocorrelation LM Test

Reports the multivariate LM test statistics for residual serial correlation up to the specified order. The test statistic for lag order h is computed by running an auxiliary regression of the residuals u_t on the original right-hand regressors and the lagged residual u_{t-h} , where the missing first h values of u_{t-h} are filled with zeros. See Johansen (1995a, p. 22) for the formula of the LM statistic. Under the null hypothesis of no serial correlation of order h , the LM statistic is asymptotically distributed χ^2 with k^2 degrees of freedom.

Normality Test

Reports the multivariate extensions of the Jarque-Bera residual normality test, which compares the third and fourth moments of the residuals to those from the normal distribution. For the multivariate test, you must choose a factorization of the k residuals that are orthogonal to each other (see “[Impulse Responses](#)” on page 527 for additional discussion of the need for orthogonalization).

Let P be a $k \times k$ factorization matrix such that

$$v_t = Pu_t \sim N(0, I_k) \quad (20.7)$$

where u_t is the demeaned residuals. Define the third and fourth moment vectors $m_3 = \sum_t v_t^3/T$ and $m_4 = \sum_t v_t^4/T$. Then

$$\sqrt{T} \begin{bmatrix} m_3 \\ m_4 - 3 \end{bmatrix} \rightarrow N \left(0, \begin{bmatrix} 6I_k & 0 \\ 0 & 24I_k \end{bmatrix} \right) \quad (20.8)$$

under the null hypothesis of normal distribution. Since each component is independent of each other, we can form a χ^2 statistic by summing squares of any of these third and fourth moments.

EViews provides you with choices for the factorization matrix P :

- **Cholesky** (Lütkepohl 1991, pp. 155-158): P is the inverse of the lower triangular Cholesky factor of the residual covariance matrix. The resulting test statistics depend on the ordering of the variables in the VAR.
- **Inverse Square Root of Residual Correlation Matrix** (Doornik and Hansen 1994): $P = HA^{-1/2}H'V$ where A is a diagonal matrix containing the eigenvalues of the residual correlation matrix on the diagonal, H is a matrix whose columns are the corresponding eigenvectors, and V is a diagonal matrix containing the inverse square root of the residual variances on the diagonal. This P is essentially the inverse square root of the residual correlation matrix. The test is invariant to the ordering and to the scale of the variables in the VAR. As suggested by Doornik and Hansen (1994), we perform a small sample correction to the transformed residuals v_t before computing the statistics.
- **Inverse Square Root of Residual Covariance Matrix** (Urzua 1997): $P = GD^{-1/2}G'$ where D is the diagonal matrix containing the eigenvalues of the residual covariance matrix on the diagonal and G is a matrix whose columns are the corresponding eigenvectors. This test has a specific alternative, which is the quartic exponential distribution. According to Urzua, this is the “most likely” alternative to the multivariate normal with finite fourth moments since it can approximate the multivariate Pearson family “as close as needed.” As recommended by Urzua, we make a small sample correction to the transformed residuals v_t before computing the statistics. This small sample correction differs from the one used by Doornik and Hansen (1994); see Urzua (1997, Section D).
- **Factorization from Identified (Structural) VAR**: $P = B^{-1}A$ where A , B are estimated from the structural VAR model. This option is available only if you have estimated the factorization matrices A and B using the structural VAR (see [page 531](#), below).

EViews reports test statistics for each orthogonal component (labeled RESID1, RESID2, and so on) and for the joint test. For individual components, the estimated skewness m_3 and kurtosis m_4 are reported in the first two columns together with the p -values from the $\chi^2(1)$ distribution (in square brackets). The Jarque-Bera column reports

$$T \left\{ \frac{m_3^2}{6} + \frac{(m_4 - 3)^2}{24} \right\} \quad (20.9)$$

with p -values from the $\chi^2(2)$ distribution. *Note: in contrast to the Jarque-Bera statistic computed in the series view, this statistic is not computed using a degrees of freedom correction.*

For the joint tests, we will generally report

$$\begin{aligned}
\lambda_3 &= Tm_3' m_3 / 6 \rightarrow \chi^2(k) \\
\lambda_4 &= T(m_4 - 3)'(m_4 - 3) / 24 \rightarrow \chi^2(k) \\
\lambda &= \lambda_3 + \lambda_4 \rightarrow \chi^2(2k).
\end{aligned}
\tag{20.10}$$

If, however, you choose Urzua's (1997) test, λ will not only use the sum of squares of the "pure" third and fourth moments but will also include the sum of squares of all *cross* third and fourth moments. In this case, λ is asymptotically distributed as a χ^2 with $k(k+1)(k+2)(k+7)/24$ degrees of freedom.

White Heteroskedasticity Test

These tests are the extension of White's (1980) test to systems of equations as discussed by Kelejian (1982) and Doornik (1995). The test regression is run by regressing each cross product of the residuals on the cross products of the regressors and testing the joint significance of the regression. The **No Cross Terms** option uses only the levels and squares of the original regressors, while the **With Cross Terms** option includes all non-redundant cross-products of the original regressors in the test equation. The test regression always includes a constant term as a regressor.

The first part of the output displays the joint significance of the regressors excluding the constant term for each test regression. You may think of each test regression as testing the constancy of each element in the residual covariance matrix separately. Under the null of no heteroskedasticity or (no misspecification), the non-constant regressors should not be jointly significant.

The last line of the output table shows the LM chi-square statistics for the joint significance of all regressors in the system of test equations (see Doornik, 1995, for details). The system LM statistic is distributed as a χ^2 with degrees of freedom mn , where $m = k(k+1)/2$ is the number of cross-products of the residuals in the system and n is the number of the common set of right-hand side variables in the test regression.

Notes on Comparability

Many of the diagnostic tests given above may be computed "manually" by estimating the VAR using a system object and selecting **View/Wald Coefficient Tests...** We caution you that the results from the system will not match those from the VAR dlogistic views for a various reasons:

- The system object will, in general, use the maximum possible observations for each equation in the system. By contrast, VAR objects force a balanced sample in case there are missing values.
- The estimates of the weighting matrix used in system estimation do not contain a degrees of freedom correction (the residual sums-of-squares are divided by T rather

than by $T - k$), while the VAR estimates do perform this adjustment. Even though estimated using comparable specifications and yielding identifiable coefficients, the test statistics from system SUR and the VARs will show small (asymptotically insignificant) differences.

Impulse Responses

A shock to the i -th variable not only directly affects the i -th variable but is also transmitted to all of the other endogenous variables through the dynamic (lag) structure of the VAR. An impulse response function traces the effect of a one-time shock to one of the innovations on current and future values of the endogenous variables.

If the innovations ϵ_t are contemporaneously uncorrelated, interpretation of the impulse response is straightforward. The i -th innovation $\epsilon_{i,t}$ is simply a shock to the i -th endogenous variable $y_{i,t}$. Innovations, however, are usually correlated, and may be viewed as having a common component which cannot be associated with a specific variable. In order to interpret the impulses, it is common to apply a transformation P to the innovations so that they become uncorrelated

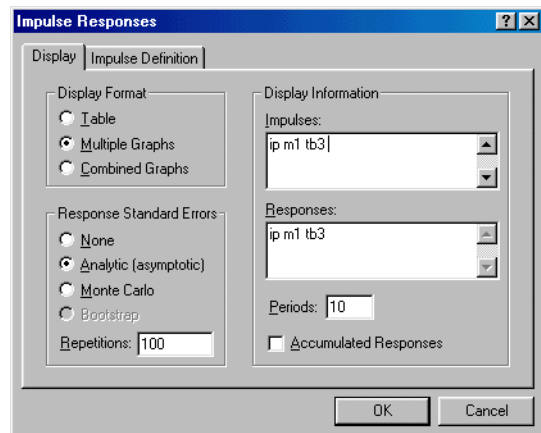
$$v_t = P\epsilon_t \sim (0, D) \quad (20.11)$$

where D is a *diagonal* covariance matrix. As explained below, EViews provides several options for the choice of P .

To obtain the impulse response functions, first estimate a VAR. Then select **View/Impulse Response...** from the VAR toolbar. You will see a dialog box with two tabs: **Display** and **Impulse Definition**.

The **Display** tab provides the following options:

- **Display Format:** displays results as a table or graph. Keep in mind that if you choose the **Combined Graphs** option, the **Response Standard Errors** option will be ignored and the standard errors will not be displayed. Note also that the output table format is ordered by response variables, not by impulse variables.



- **Display Information:** you should enter the variables for which you wish to generate innovations (**Impulses**) and the variables for which you wish to observe the

responses (**Responses**). You may either enter the name of the endogenous variables or the numbers corresponding to the ordering of the variables. For example, if you specified the VAR as GDP, M1, CPI, then you may either type

GDP CPI M1

or

1 3 2

The order in which you enter these variables only affects the display of results.

You should also specify a positive integer for the number of periods to trace the response function. To display the accumulated responses, check the **Accumulate Response** box. For stationary VARs, the impulse responses should die out to zero and the accumulated responses should asymptote to some (non-zero) constant.

- **Response Standard Errors:** provides options for computing the response standard errors. Note that analytic and/or Monte Carlo standard errors are currently not available for certain **Impulse** options and for vector error correction (VEC) models. If you choose **Monte Carlo** standard errors, you should also specify the number of repetitions to use in the appropriate edit box.

If you choose the table format, the estimated standard errors will be reported in parentheses below the responses. If you choose to display the results in multiple graphs, the graph will contain the plus/minus two standard error bands about the impulse responses. The standard error bands are not displayed in combined graphs.

The **Impulse** tab provides the following options for transforming the impulses:

- **Residual—One Unit** sets the impulses to one unit of the residuals. This option ignores the units of measurement and the correlations in the VAR residuals so that no transformation is performed. The responses from this option are the MA coefficients of the infinite MA order Wold representation of the VAR.
- **Residual—One Std. Dev.** sets the impulses to one standard deviation of the residuals. This option ignores the correlations in the VAR residuals.
- **Cholesky** uses the inverse of the Cholesky factor of the residual covariance matrix to orthogonalize the impulses. This option imposes an ordering of the variables in the VAR and attributes all of the effect of any common component to the variable that comes first in the VAR system. Note that responses can change dramatically if you change the ordering of the variables. You may specify a different VAR ordering by reordering the variables in the **Cholesky Ordering** edit box.

The **(d.f. adjustment)** option makes a small sample degrees of freedom correction when estimating the residual covariance matrix used to derive the Cholesky factor. The (i,j) -th element of the residual covariance matrix with degrees of freedom cor-

rection is computed as $\sum_t e_{i,t} e_{j,t} / (T - p)$ where p is the number of parameters per equation in the VAR. The **(no d.f. adjustment)** option estimates the (i, j) -th element of the residual covariance matrix as $\sum_t e_{i,t} e_{j,t} / T$. *Note: previous versions of EViews computed the impulses using the Cholesky factor from the residual covariance matrix with no degrees of freedom adjustment.*

- **Generalized Impulses** as described by Pesaran and Shin (1998) constructs an orthogonal set of innovations that does not depend on the VAR ordering. The generalized impulse responses from an innovation to the j -th variable are derived by applying a variable specific Cholesky factor computed with the j -th variable at the top of the Cholesky ordering.
- **Structural Decomposition** uses the orthogonal transformation estimated from the structural factorization matrices. This approach is not available unless you have estimated the structural factorization matrices as explained in [“Structural \(Identified\) VARs” on page 531](#).
- **User Specified** allows you to specify your own impulses. Create a matrix (or vector) that contains the impulses and type the name of that matrix in the edit box. If the VAR has k endogenous variables, the impulse matrix must have k rows and 1 or k columns, where each column is a impulse vector.

For example, say you have a $k = 3$ variable VAR and wish to apply simultaneously a positive one unit shock to the first variable and a negative one unit shock to the second variable. Then you will create a 3×1 impulse matrix containing the values 1, -1, and 0. Using commands, you can enter:

```
matrix(3,1) shock
shock.fill(by=c) 1, -1, 0
```

and type the name of the matrix SHOCK in the edit box.

Variance Decomposition

While impulse response functions trace the effects of a shock to one endogenous variable on to the other variables in the VAR, *variance decomposition* separates the variation in an endogenous variable into the component shocks to the VAR. Thus, the variance decomposition provides information about the relative importance of each random innovation in affecting the variables in the VAR.

To obtain the variance decomposition, select **View/Variance Decomposition...** from the var object toolbar. You should provide the same information as for impulse responses above. Note that since non-orthogonal factorization will yield decompositions that do not satisfy an adding up property, your choice of factorization is limited to orthogonal factorizations.

The table format displays a separate variance decomposition for each endogenous variable. The second column, labeled “S.E.”, contains the forecast error of the variable at the given forecast horizon. The source of this forecast error is the variation in the current and future values of the innovations to each endogenous variable in the VAR. The remaining columns give the percentage of the forecast variance due to each innovation, with each row adding up to 100.

As with the impulse responses, the variance decomposition based on the Cholesky factor can change dramatically if you alter the ordering of the variables in the VAR. For example, the first period decomposition for the first variable in the VAR ordering is completely due to its own innovation.

Factorization based on structural orthogonalization is available only if you have estimated the structural factorization matrices as explained in [“Structural \(Identified\) VARs” on page 531](#). Note that the forecast standard errors should be identical to those from the Cholesky factorization *if the structural VAR is just identified*. For over-identified structural VARs, the forecast standard errors may differ in order to maintain the adding up property.

Procs of a VAR

Most of the procedures (procs) available for a VAR are common to those available for a system object (see [“System Procs” on page 507](#)). Here we discuss only those procs that are unique to the VAR object.

Make System

This proc creates a system object that contains an equivalent VAR specification. If you want to estimate a non-standard VAR, you may use this proc as a quick way to specify a VAR in a system object which you can then modify to meet your needs. For example, while the VAR object requires each equation to have the same lag structure, you may want to relax this restriction. To estimate a VAR with unbalanced lag structure, use the **Make System** proc to create a VAR system with a balanced lag structure and edit the system specification to meet the desired lag specification.

The **By Variable** option creates a system whose specification (and coefficient number) is ordered by variables. Use this option if you want to edit the specification to exclude lags of a specific variable from some of the equations. The **By Lag** option creates a system whose specification (and coefficient number) is ordered by lags. Use this option if you want to edit the specification to exclude certain lags from some of the equations.

For vector error correction (VEC) models, treating the coefficients of the cointegrating vector as additional unknown coefficients will make the resulting system unidentified. In this case, EViews will create a system object where the coefficients for the cointegrating vectors are fixed at the estimated values from the VEC. If you want to estimate the coefficients of

the cointegrating vector in the system, you may edit the specification, but you should make certain that the resulting system is identified.

You should also note that while the standard VAR can be estimated efficiently by equation-by-equation OLS, this is generally not the case for the modified specification. You may wish to use one of the system-wide estimation methods (e.g. SUR) when estimating non-standard VARs using the system object.

Estimate Structural Factorization

This procedure is used to estimate the factorization matrices for a structural (or identified) VAR. The full details of this procedure is given in [“Structural \(Identified\) VARs” on page 531](#). You must first estimate the structural factorization matrices using this proc in order to use the structural options in impulse responses and variance decompositions.

Structural (Identified) VARs

The main purpose of structural VAR (SVAR) estimation is to obtain non-recursive orthogonalization of the error terms for impulse response analysis. This alternative to the recursive Cholesky orthogonalization requires the user to impose enough restrictions to identify the orthogonal (structural) components of the error terms.

Let y_t be a k -element vector of the endogenous variables and let $\Sigma = E[e_t e_t']$ be the residual covariance matrix. Following Amisano and Giannini (1997), the class of SVAR models that we estimate can be written as

$$Ae_t = Bu_t \quad (20.12)$$

where e_t and u_t are vectors of length k . e_t is the observed (or reduced form) residuals, while u_t is the unobserved structural innovations. A and B are $k \times k$ matrices to be estimated. The structural innovations u_t are assumed to be orthonormal, *i.e.* its covariance matrix is an identity matrix $E[u_t u_t'] = I$. The assumption of orthonormal innovations u_t imposes the following identifying restrictions on A and B :

$$A\Sigma A' = BB' . \quad (20.13)$$

Noting that the expression on either side of (20.13) are symmetric, this imposes $k(k+1)/2$ restrictions on the $2k^2$ unknown elements in A and B . Therefore, in order to identify A and B , you need to supply at least $2k^2 - k(k+1)/2 = k(3k-1)/2$ additional restrictions.

Specifying the Identifying Restrictions

As explained above, in order to estimate the orthogonal factorization matrices A and B , you need to provide additional identifying restrictions. We distinguish two types of identi-

fying restrictions: *short-run* and *long-run*. For either type, the identifying restrictions can be specified either in text form or by pattern matrices.

Short-run Restrictions by Pattern Matrices

For many problems, the identifying restrictions on the A and B matrices are simple zero exclusion restrictions. In this case, you can specify the restrictions by creating a named “pattern” matrix for A and B . Any elements of the matrix that you want to be estimated should be assigned a missing value “NA”. All non-missing values in the pattern matrix will be held fixed at the specified values.

For example, suppose you want to restrict A to be a lower triangular matrix with ones on the main diagonal and B to be a diagonal matrix. Then the pattern matrices (for a $k = 3$ variable VAR) would be

$$A = \begin{pmatrix} 1 & 0 & 0 \\ \text{NA} & 1 & 0 \\ \text{NA} & \text{NA} & 1 \end{pmatrix}, \quad B = \begin{pmatrix} \text{NA} & 0 & 0 \\ 0 & \text{NA} & 0 \\ 0 & 0 & \text{NA} \end{pmatrix}. \quad (20.14)$$

You can create these matrices interactively. Simply use **Objects/NewObjects...** to create two new 3×3 matrices, A and B , and then use the spreadsheet view to edit the values. Alternatively, you can issue the following commands:

```
matrix(3,3) pata
' fill matrix in row major order
pata.fill(by=r) 1,0,0, na,1,0, na,na,1
matrix(3,3) patb = 0
patb(1,1) = na
patb(2,2) = na
patb(3,3) = na
```

Once you have created the pattern matrices, select **Procs/Estimate Structural Factorization...** from the VAR window menu. In the **SVAR Options** dialog, click the **Matrix** button and the **Short-Run Pattern** button and type in the name of the pattern matrices in the relevant edit boxes.

Short-run Restrictions in Text Form

For more general restrictions, you can specify the identifying restrictions in text form. In text form, you will write out the relation $Ae_t = Bu_t$ as a set of equations, identifying each element of the e_t and u_t vectors with special symbols. Elements of the A and B matrices to be estimated must be specified as elements of a coefficient vector.

To take an example, suppose again that you have a $k = 3$ variable VAR where you want to restrict A to be a lower triangular matrix with ones on the main diagonal and B to be a diagonal matrix. Under these restrictions, the relation $Ae_t = Bu_t$ can be written as

$$\begin{aligned}e_1 &= b_{11}u_1 \\e_2 &= -a_{21}e_1 + b_{22}u_2 \\e_3 &= -a_{31}e_1 - a_{32}e_2 + b_{33}u_3\end{aligned}\tag{20.15}$$

To specify these restrictions in text form, select **Procs/Estimate Structural Factorization...** from the VAR window and click the **Text** button. In the edit window, you should type the following:

```
@e1 = c(1)*@u1
@e2 = -c(2)*@e1 + c(3)*@u2
@e3 = -c(4)*@e1 - c(5)*@e2 + c(6)*@u3
```

The special key symbols “@e1”, “@e2”, “@e3,” represent the first, second, and third elements of the e_t vector, while “@u1”, “@u2”, “@u3” represent the first, second, and third elements of the u_t vector. In this example, all unknown elements of the A and B matrices are represented by elements of the C coefficient vector.

Long-run Restrictions

The identifying restrictions embodied in the relation $Ae = Bu$ are commonly referred to as short-run restrictions. Blanchard and Quah (1989) proposed an alternative identification method based on restrictions on the long-run properties of the impulse responses. The (accumulated) long-run response C to structural innovations takes the form

$$C = \hat{\Psi}_\infty A^{-1}B\tag{20.16}$$

where $\hat{\Psi}_\infty = (I - \hat{A}_1 - \dots - \hat{A}_p)^{-1}$ is the estimated accumulated responses to the reduced form (observed) shocks. Long-run identifying restrictions are specified in terms of the elements of this C matrix, typically in the form of zero restrictions. The restriction $C_{i,j} = 0$ means that the (accumulated) response of the i -th variable to the j -th structural shock is zero in the long-run.

It is important to note that the expression for the long-run response (20.16) involves the inverse of A . Since EViews currently requires all restrictions to be linear in the elements of A and B , if you specify a long-run restriction, the A matrix must be the identity matrix.

To specify long-run restrictions by a pattern matrix, create a named matrix that contains the pattern for the long-run response matrix C . Unrestricted elements in the C matrix should be assigned a missing value “NA”. For example, suppose you have a $k = 2$ variable VAR where you want to restrict the long-run response of the second endogenous vari-

able to the first structural shock to be zero $C_{2,1} = 0$. Then the long-run response matrix will have the following pattern:

$$C = \begin{pmatrix} \text{NA} & \text{NA} \\ 0 & \text{NA} \end{pmatrix} \quad (20.17)$$

You can create this matrix with the following commands:

```
matrix(2,2) patc = na
patc(2,1) = 0
```

Once you have created the pattern matrix, select **Procs/Estimate Structural Factorization...** from the VAR window menu. In the **SVAR Options** dialog, click the **Matrix** button and the **Long-Run Pattern** button and type in the name of the pattern matrix in the relevant edit box.

To specify the same long-run restriction in text form, select **Procs/Estimate Structural Factorization...** from the VAR window and click the **Text** button. In the edit window, you would type the following

```
@lr2(@u1)=0 ' zero LR response of 2nd variable to 1st shock
```

where everything on the line after the apostrophe is a comment. This restriction begins with the special keyword “@lr#”, with the number representing the response variable to restrict. Inside the parentheses, you must specify the impulse keyword @u and the innovation number, followed by an equal sign and the value of the response (typically 0). We caution you that while you can list multiple long-run restrictions, *you cannot mix short-run and long-run restrictions*.

Note that it is possible to specify long-run restrictions as short-run restrictions (by obtaining the infinite MA order representation). While the estimated A and B matrices should be the same, the impulse response standard errors from the short-run representation would be incorrect (since it does not take into account the uncertainty in the estimated infinite MA order coefficients).

Some Important Notes

Currently we have the following limitations for the specification of identifying restrictions:

- The A and B matrices must be square and non-singular. In text form, there must be exactly as many equations as there are endogenous variables in the VAR. For short-run restrictions in pattern form, you must provide the pattern matrices for both A and B matrices.

- The restrictions must be linear in the elements of A and B . Moreover, the restrictions on A and B must be independent (no restrictions across elements of A and B).
- You cannot impose both short-run and long-run restrictions.
- Structural decompositions are currently not available for VEC models.
- The identifying restriction assumes that the structural innovations u_t have unit variances. Therefore, you will almost always want to estimate the diagonal elements of the B matrix so that you obtain estimates of the standard deviations of the structural shocks.
- It is common in the literature to assume that the structural innovations have a diagonal covariance matrix rather than an identity matrix. To compare your results to those from these studies, you will have to divide each column of the B matrix with the diagonal element in that column (so that the resulting B matrix has ones on the main diagonal). To illustrate this transformation, consider a simple $k = 2$ variable model with $A = 1$:

$$\begin{aligned} e_{1,t} &= b_{11}u_{1,t} + b_{12}u_{2,t} \\ e_{2,t} &= b_{21}u_{1,t} + b_{22}u_{2,t} \end{aligned} \quad (20.18)$$

where $u_{1,t}$ and $u_{2,t}$ are independent structural shocks with unit variances as assumed in the EViews specification. To rewrite this specification with a B matrix containing ones on the main diagonal, define a new set of structural shocks by the transformations $v_{1,t} = b_{11}u_{1,t}$ and $v_{2,t} = b_{22}u_{2,t}$. Then the structural relation can be rewritten as

$$\begin{aligned} e_{1,t} &= v_{1,t} + (b_{12}/b_{22})v_{2,t} \\ e_{2,t} &= (b_{21}/b_{11})v_{1,t} + v_{2,t} \end{aligned} \quad (20.19)$$

where now

$$B = \begin{pmatrix} 1 & b_{12}/b_{22} \\ b_{21}/b_{11} & 1 \end{pmatrix}, \quad v_t = \begin{bmatrix} v_{1,t} \\ v_{2,t} \end{bmatrix} \sim \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} b_{11}^2 & 0 \\ 0 & b_{22}^2 \end{bmatrix} \right) \quad (20.20)$$

Note that the transformation involves only rescaling elements of the B matrix and not on the A matrix. For the case where B is a diagonal matrix, the elements in the main diagonal are simply the estimated standard deviations of the structural shocks.

Identification Conditions

As stated above, the assumption of orthonormal structural innovations imposes $k(k+1)/2$ restrictions on the $2k^2$ unknown elements in A and B , where k is the

number of endogenous variables in the VAR. In order to identify A and B , you need to provide at least $k(k+1)/2 - 2k^2 = k(3k-1)/2$ additional identifying restrictions. This is a necessary order condition for identification and is checked by counting the number of restrictions provided.

As discussed in Amisano and Giannini (1997), a sufficient condition for local identification can be checked by the invertibility of the “augmented” information matrix (see Amisano and Giannini, 1997). This local identification condition is evaluated numerically at the starting values. If EViews returns a singularity error message for different starting values, you should make certain that your restrictions identify the A and B matrices.

We also require the A and B matrices to be square and non-singular. The non-singularity condition is checked numerically at the starting values. If the A and B matrix is non-singular at the starting values, an error message will ask you to provide a different set of starting values.

Sign Indeterminacy

For some restrictions, the signs of the A and B matrices are not identified; see Christiano, Eichenbaum, and Evans (1999) for a discussion of this issue. When the sign is indeterminate, we choose a normalization so that the diagonal elements of the factorization matrix $A^{-1}B$ are all positive. This normalization ensures that all structural impulses have positive signs (as does the Cholesky factorization). The default is to always apply this normalization rules whenever applicable. If you do not want to switch the signs, deselect the **Normalize Sign** option from the **Optimization Control** tab of the **SVAR Options** dialog.

Estimation of A and B Matrices

Once you provide the identifying restrictions in any of the forms described above, you are ready to estimate the A and B matrices. Simply click the **OK** button in the **SVAR Options** dialog. You must first estimate these matrices in order to use the structural option in impulse responses and variance decompositions.

A and B are estimated by maximum likelihood, assuming the innovations are multivariate normal. We evaluate the likelihood in terms of unconstrained parameters by substituting out the constraints. The log likelihood is maximized by the method of scoring (with a Marquardt-type diagonal correction—See “Marquardt” on page 665), where the gradient and expected information matrix are evaluated analytically. See Amisano and Giannini (1997) for the analytic expression of these derivatives.

Optimization Control

Options for controlling the optimization process are provided in the **Optimization Control** tab of the **SVAR Options** dialog. You have the option to specify the starting values, maximum number of iterations, and the convergence criterion.

The starting values are those for the unconstrained parameters after substituting out the constraints. **Fixed** sets all free parameters to the value specified in the edit box. **User Specified** uses the values in the coefficient vector as specified in text form as starting values. For restrictions specified in pattern form, user specified starting values are taken from the first m elements of the default C coefficient vector, where m is the number of free parameters. **Draw from...** options randomly draw the starting values for the free parameters from the specified distributions.

Estimation Output

Once convergence is achieved, EViews displays the estimation output in the VAR window. The point estimates, standard errors, and z -statistics of the estimated free parameters are reported together with the maximized value of the log likelihood. The estimated standard errors are based on the inverse of the estimated information matrix (negative expected value of the Hessian) evaluated at the final estimates.

For overidentified models, we also report the LR test for over-identification. The LR test statistic is computed as

$$LR = 2(l_u - l_r) = T(\text{tr}P - \log|P| - k) \quad (20.21)$$

where $P = A'B^{-T}B^{-1}A\Sigma$. Under the null hypothesis that the restrictions are valid, the LR statistic is asymptotically distributed $\chi^2(q - k)$ where q is the number of identifying restrictions.

If you switch the view of the VAR window, you can come back to the previous results (without reestimating) by selecting **View/SVAR Output** from the VAR window. In addition, some of the SVAR estimation results can be retrieved as data members of the VAR; see [“Var Data Members” on page 50](#) of the *Command and Programming Reference* for a list of available VAR data members.

Cointegration Test

The finding that many macro time series may contain a unit root has spurred the development of the theory of non-stationary time series analysis. Engle and Granger (1987) pointed out that a linear combination of two or more non-stationary series may be stationary. If such a stationary linear combination exists, the non-stationary time series are said to be *cointegrated*. The stationary linear combination is called the *cointegrating equation* and may be interpreted as a long-run equilibrium relationship among the variables.

The purpose of the cointegration test is to determine whether a group of non-stationary series are cointegrated or not. As explained below, the presence of a cointegrating relation forms the basis of the VEC specification. EViews implements VAR-based cointegration tests using the methodology developed in Johansen (1991, 1995a). Consider a VAR of order p

$$y_t = A_1 y_{t-1} + \dots + A_p y_{t-p} + Bx_t + \epsilon_t \quad (20.22)$$

where y_t is a k -vector of non-stationary $I(1)$ variables, x_t is a d -vector of deterministic variables, and ϵ_t is a vector of innovations. We can rewrite this VAR as

$$\Delta y_t = \Pi y_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \Delta y_{t-i} + Bx_t + \epsilon_t \quad (20.23)$$

where

$$\Pi = \sum_{i=1}^p A_i - I, \quad \Gamma_i = - \sum_{j=i+1}^p A_j \quad (20.24)$$

Granger's representation theorem asserts that if the coefficient matrix Π has reduced rank $r < k$, then there exist $k \times r$ matrices α and β each with rank r such that $\Pi = \alpha\beta'$ and $\beta'y_t$ is $I(0)$. r is the number of cointegrating relations (the *cointegrating rank*) and each column of β is the cointegrating vector. As explained below, the elements of α are known as the adjustment parameters in the VEC model. Johansen's method is to estimate the Π matrix from an unrestricted VAR and to test whether we can reject the restrictions implied by the reduced rank of Π .

How to Perform a Cointegration Test

To carry out the Johansen cointegration test, select View/Cointegration Test... from the group or VAR window toolbar. Note that since this is a test for cointegration, this test is only valid when you are working with series that are known to be nonstationary. You may wish first to apply unit root tests to each series in the VAR. See [“Unit Root Test” on page 170](#) for details on carrying out unit root tests in EViews. The **Cointegration Test Specification** page prompts you for information about the test.

Deterministic Trend Specification

Your series may have nonzero means and deterministic trends as well as stochastic trends. Similarly, the cointegrating equations may have intercepts and deterministic trends. The asymptotic distribution of the LR test statistic for cointegration does not have the usual χ^2 distribution and depends on the assumptions made with respect to deterministic trends. Therefore,

Johansen Cointegration Test

Cointegration Test Specification: **VEC Restrictions**

$D(Y) = A(i,j) * COINTEQ + C(i,j) * OUTSIDE$

Deterministic trend assumption of test

Case	Intercept & Trend	Inclusion	Trend	Assumption
	COINTEQ	OUTSIDE		
<input type="radio"/> 1)	Neither	Neither	No Trend	Trend in Data
<input type="radio"/> 2)	C	Neither		
<input checked="" type="radio"/> 3)	C	C	Linear	Trend
<input type="radio"/> 4)	C & Trend	C	Quadratic	Trend
<input type="radio"/> 5)	C & Trend	C & Trend		
<input type="radio"/> 6)	Summary of all 5 assumptions			

Exog variables (OUTSIDE):

Do not include C or Trend

Critical values may not be valid with exogenous variables

Lag intervals:

14

Lag spec for differenced endogenous in OUTSIDE

OK Cancel

in order to carry out the test, you need to make an assumption regarding the trend underlying your data.

For each row case in the dialog, the COINTEQ column lists the deterministic variables that appear inside the cointegrating relations (error correction term), while the OUTSIDE column lists the deterministic variables that appear in the VEC equation outside the cointegrating relations. Cases 2 and 4 do not have the same set of deterministic terms in the two columns. For these two cases, some of the deterministic term is restricted to belong only in the cointegrating relation. For cases 3 and 5, the deterministic terms are common in the two columns and the decomposition of the deterministic effects inside and outside the cointegrating space is not uniquely identified; see the technical discussion below.

In practice, cases 1 and 5 are rarely used. You should use case 1 only if you know that all series have zero mean. Case 5 may provide a good fit in-sample but will produce implausible forecasts out-of-sample. As a rough guide, use case 2 if none of the series appear to have a trend. For trending series, use case 3 if you believe all trends are stochastic; if you believe some of the series are trend stationary, use case 4.

If you are not certain which trend assumption to use, you may choose the **Summary of all 5 trend assumptions** option (case 6) to help you determine the choice of the trend assumption. This option indicates the number of cointegrating relations under each of the 5 trend assumptions and you will be able to see how sensitive the results of the test are to the assumption of trend.

Technical Discussion

EViews considers the following five deterministic trend cases considered by Johansen (1995a, pp. 80–84):

1. The level data y_t have no deterministic trends and the cointegrating equations do not have intercepts:

$$H_2(r): \Pi y_{t-1} + Bx_t = \alpha \beta' y_{t-1}$$

2. The level data y_t have no deterministic trends and the cointegrating equations have intercepts:

$$H_1^*(r): \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0)$$

3. The level data y_t have linear trends but the cointegrating equations have only intercepts:

$$H_1(r): \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0) + \alpha_{\perp} \gamma_0$$

4. The level data y_t and the cointegrating equations have linear trends:

$$H^*(r): \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0 + \rho_1 t) + \alpha_{\perp} \gamma_0$$

5. The level data y_t have quadratic trends and the cointegrating equations have linear trends:

$$H(r): \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0 + \rho_1 t) + \alpha_{\perp}(\gamma_0 + \gamma_1 t)$$

The terms associated with α_{\perp} are the deterministic terms “outside” the cointegrating relations. When a deterministic term appears both inside and outside the cointegrating relation, the decomposition is not uniquely identified. Johansen (1995a) identifies the part that belongs inside the error correction term by orthogonally projecting the exogenous terms onto the α space so that α_{\perp} is the null space of α such that $\alpha' \alpha_{\perp} = 0$. EViews uses a different identification method so that the error correction term has a sample mean of zero. More specifically, we identify the part inside the error correction term by regressing the cointegrating relations $\beta' y_t$ on a constant (and linear trend).

Exogenous Variables

The test dialog allows you to specify additional exogenous variables x_t to include in the test VAR. *The constant and linear trend should not be listed in the edit box* since they are specified using the five **Trend Specification** options. If you choose to include exogenous variables, be aware that the critical values reported by EViews *do not account* for these variables.

The most commonly added deterministic terms are seasonal dummy variables. Note, however, that if you include standard 0–1 seasonal dummy variables in the test VAR, this will affect both the mean and the trend of the level series y_t . To handle this problem, Johansen (1995a, page 84) suggests using centered (orthogonalized) seasonal dummy variables, which shift the mean without contributing to the trend. Centered seasonal dummy variables for quarterly and monthly series can be generated by the commands

```
series d_q = @seas(q) - 1/4
series d_m = @seas(m) - 1/12
```

for quarter q and month m , respectively.

Lag Intervals

You should specify the lags of the test VAR as pairs of intervals. Note that the lags are specified as *lags of the first differenced terms* used in the auxiliary regression, not in terms of the levels. For example, if you type “1 2” in the edit field, the test VAR regresses Δy_t on Δy_{t-1} , Δy_{t-2} , and any other exogenous variables that you have specified. Note that in terms of the level series y_t the largest lag is 3. To run a cointegration test with one lag in the level series, type “0 0” in the edit field.

Interpreting Results of a Cointegration Test

As an example, the first part of the output for a four-variable system used in Johansen and Juselius (1990) for the Danish data is shown below.

Date: 10/17/00 Time: 09:32
 Sample(adjusted): 1974:3 1987:3
 Included observations: 53 after adjusting endpoints
 Trend assumption: No deterministic trend (restricted constant)
 Series: LRM LRY IBO IDE
 Lags interval (in first differences): 1 to 1

Unrestricted Cointegration Rank Test

Hypothesized No. of CE(s)	Eigenvalue	Trace Statistic	5 Percent Critical Value	1 Percent Critical Value
None	0.469677	52.71087	53.12	60.16
At most 1	0.174241	19.09464	34.91	41.07
At most 2	0.118083	8.947661	19.96	24.60
At most 3	0.042249	2.287849	9.24	12.97

*(**) denotes rejection of the hypothesis at the 5%(1%) level
 Trace test indicates no cointegration at both 5% and 1% levels

As indicated in the header of the output, the test assumes no trend in the series with a restricted intercept in the cointegration relation (second trend specification in the dialog), includes three orthogonalized seasonal dummy variables D1–D3, and uses one lag in differences (two lags in levels) which is specified as “1 1” in the edit field.

Number of Cointegrating Relations

The first part of the table reports results for testing the number of cointegrating relations. Two types of test statistics are reported. The first block reports the so-called *trace* statistics and the second block (not shown above) reports the *maximum eigenvalue* statistics. For each block, the first column is the number of cointegrating relations under the null hypothesis, the second column is the ordered eigenvalues of the Π matrix in (20.24), the third column is the test statistic, and the last two columns are the 5% and 1% critical values. The (nonstandard) critical values are taken from Osterwald-Lenum (1992), which differ slightly from those reported in Johansen and Juselius (1990).

To determine the number of cointegrating relations r conditional on the assumptions made about the trend, we can proceed sequentially from $r = 0$ to $r = k - 1$ until we fail to reject. The result of this sequential testing procedure is reported at the bottom of each table block.

The trace statistic reported in the first block tests the null hypothesis of r cointegrating relations against the alternative of k cointegrating relations, where k is the number of endogenous variables, for $r = 0, 1, \dots, k - 1$. The alternative of k cointegrating rela-

tions corresponds to the case where none of the series has a unit root and a stationary VAR may be specified in terms of the levels of all of the series. The trace statistic for the null hypothesis of r cointegrating relations is computed as

$$LR_{\text{tr}}(r|k) = -T \sum_{i=r+1}^k \log(1 - \lambda_i) \quad (20.25)$$

where λ_i is the i -th largest eigenvalue of the Π matrix in (20.24) which is reported in the second column of the output table.

The second block of the output reports the maximum eigenvalue statistic which tests the null hypothesis of r cointegrating relations against the alternative of $r + 1$ cointegrating relations. This test statistic is computed as

$$\begin{aligned} LR_{\text{max}}(r|r+1) &= -T \log(1 - \lambda_{r+1}) \\ &= LR_{\text{tr}}(r|k) - LR_{\text{tr}}(r+1|k) \end{aligned} \quad (20.26)$$

for $r = 0, 1, \dots, k - 1$.

There are a few other details to keep in mind:

- Critical values are available for up to $k = 10$ series. Also note that the critical values depend on the trend assumptions and may not be appropriate for models that contain other deterministic regressors. For example, a shift dummy variable in the test VAR implies a broken linear trend in the level series y_t .
- The trace statistic and the maximum eigenvalue statistic may yield conflicting results. For such cases, we recommend that you examine the estimated cointegrating vector and base your choice on the interpretability of the cointegrating relations; see Johansen and Juselius (1990) for an example.
- In some cases, the individual unit root tests will show that some of the series are integrated, but the cointegration test will indicate that the Π matrix has full rank ($r = k$). This apparent contradiction may be the result of low power of the cointegration tests, stemming perhaps from a small sample size or serving as an indication of specification error.

Cointegrating relations

The second part of the output provides estimates of the cointegrating relations β and the adjustment parameters α . As is well known, the cointegrating vector β is not identified unless we impose some arbitrary normalization. The first block reports estimates of β and α based on the normalization $\beta' S_{11} \beta = I$, where S_{11} is defined in Johansen (1995a). Note that the *transpose* of β is reported under **Unrestricted Cointegrating Coefficients** so that the first *row* is the first cointegrating vector, the second *row* is the second cointegrating vector, and so on.

The remaining blocks report estimates from a different normalization for each possible number of cointegrating relations $r = 0, 1, \dots, k - 1$. This alternative normalization expresses the first r variables as functions of the remaining $k - r$ variables in the system. Asymptotic standard errors are reported in parentheses for the parameters that are identified.

Imposing Restrictions

Since the cointegrating vector β is not identified, you may wish to impose your own identifying restrictions. Restrictions can be imposed on the cointegrating vector (elements of the β matrix) and/or on the adjustment coefficients (elements of the α matrix). To impose restrictions in a cointegration test, select **View/Cointegration Test...** and specify the options in the **Trend Specification** tab as explained above. Then bring up the **VEC Restrictions** tab and check the **Impose Restrictions** box. You will enter your restrictions in the edit box that appears.

Restrictions on the Cointegrating Vector

To impose restrictions on the cointegrating vector β , you must refer to the (i, j) -th element of the *transpose* of the β matrix by $B(i, j)$. The i -th cointegrating relation has the representation

$$B(i, 1) * y_1 + \\ B(i, 2) * y_2 + \dots + \\ B(i, k) * y_k$$

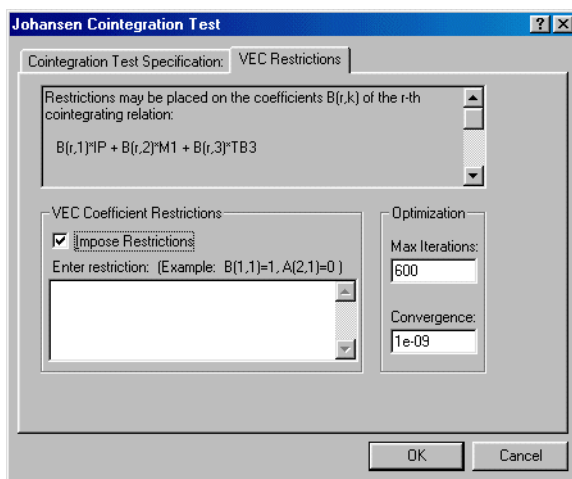
where y_1, y_2, \dots are the (lagged) endogenous variable. Then, if you want to impose the restriction that the coefficient on y_1 for the second cointegrating equation is 1, you would type the following in the edit box:

$$B(2, 1) = 1$$

You can impose multiple restrictions by separating each restriction with a comma on the same line or typing each restriction on a separate line. For example, if you want to impose the restriction that the coefficients on y_1 for the first and second cointegrating equations are 1, you would type

$$B(1, 1) = 1$$

$$B(2, 1) = 1$$



Currently *all restrictions must be linear (or more precisely affine) in the elements of the β matrix*. So for example

$$B(1,1) * B(2,1) = 1$$

will return a syntax error.

Restrictions on the Adjustment Coefficients

To impose restrictions on the adjustment coefficients, you must refer to the (i,j) -th elements of the α matrix by $A(i,j)$. The error correction terms in the i -th VEC equation will have the representation

$$A(i,1)*\text{CointEq1} + A(i,2)*\text{CointEq2} + \dots + A(i,r)*\text{CointEq}r$$

Restrictions on the adjustment coefficients are currently limited to linear homogeneous restrictions so that you must be able to write your restriction as $R\text{vec}(\alpha) = 0$, where R is a known $qk \times r$ matrix. This condition implies, for example, that the restriction

$$A(1,1) = A(2,1)$$

is valid but

$$A(1,1) = 1$$

will return a restriction syntax error.

One restriction of particular interest is whether the i -th row of the α matrix is all zero. If this is the case, then the i -th endogenous variable is said to be *weakly exogenous with respect to the β parameters*. See Johansen (1992b) for the definition and implications of weak exogeneity. For example, if we assume that there is only one cointegrating relation in the VEC, to test whether the second endogenous variable is weakly exogenous with respect to β you would enter

$$A(2,1) = 0$$

To impose multiple restrictions, you may either separate each restriction with a comma on the same line or type each restriction on a separate line. For example, to test whether the second endogenous variable is weakly exogenous with respect to β in a VEC with two cointegrating relations, you can type

$$A(2,1) = 0$$

$$A(2,2) = 0$$

You may also impose restrictions on both β and α . However, the restrictions on β and α must be *independent*. So for example,

$$A(1,1) = 0$$

$$B(1,1) = 1$$

is a valid restriction but

$$A(1, 1) = B(1, 1)$$

will return a restriction syntax error.

Identifying Restrictions and Binding Restrictions

EViews will check to see whether the restrictions you provided identify all cointegrating vectors for each possible rank. The identification condition is checked numerically by the rank of the appropriate Jacobian matrix; see Boswijk (1995) for the technical details. Asymptotic standard errors for the estimated cointegrating parameters will be reported only if the restrictions identify the cointegrating vectors.

If the restrictions are binding, EViews will report the LR statistic to test the binding restrictions. The LR statistic is reported if the degrees of freedom of the asymptotic χ^2 distribution is positive. Note that the restrictions can be binding even if they are not identifying, (e.g. when you impose restrictions on the adjustment coefficients but not on the cointegrating vector).

Options for Restricted Estimation

Estimation of the restricted cointegrating vectors β and adjustment coefficients α generally involves an iterative process. The **VEC Restrictions** tab provides iteration control for the maximum number of iterations and the convergence criterion. EViews estimates the restricted β and α using the switching algorithm as described in Boswijk (1995). Each step of the algorithm is guaranteed to increase the likelihood and the algorithm should eventually converge (though convergence may be to a local rather than a global optimum). You may need to increase the number of iterations in case you are having difficulty achieving convergence at the default settings.

Results of Restricted Cointegration Test

If you impose restrictions in the **Cointegration Test** view, the output will first display the test results without the restrictions as described above. The second part of the output begins by displaying the results of the LR test for binding restrictions.

Restrictions:

a(3,1)=0

Tests of cointegration restrictions:

Hypothesized No. of CE(s)	Restricted Log-likelihood	LR Statistic	Degrees of Freedom	Probability
1	668.6698	0.891088	1	0.345183
2	674.2964	NA	NA	NA
3	677.4677	NA	NA	NA

NA indicates restriction not binding.

If the restrictions are not binding for a particular rank, the corresponding rows will be filled with NAs. If the restrictions are binding but the algorithm did not converge, the corresponding row will be filled with an asterisk “*”. (You should redo the test by increasing the number of iterations or relaxing the convergence criterion.) For the example output displayed above, we see that the single restriction $\alpha_{31} = 0$ is binding only under the assumption that there is one cointegrating relation. *Conditional on there being only one cointegrating relation*, the LR test does not reject the imposed restriction at conventional levels.

The output also reports the estimated β and α imposing the restrictions. Since the cointegration test does not specify the number of cointegrating relations, results for all ranks that are consistent with the specified restrictions will be displayed. For example, suppose the restriction is

$$B(2, 1) = 1$$

Since this is a restriction on the second cointegrating vector, EViews will display results for ranks $r = 2, 3, \dots, k-1$ (if the VAR has only $k = 2$ variables, EViews will return an error message pointing out that the “implied rank from restrictions must be of reduced order”).

For each rank, the output reports whether convergence was achieved and the number of iterations. The output also reports whether the restrictions identify all cointegrating parameters under the assumed rank. If the cointegrating vectors are identified, asymptotic standard errors will be reported together with the parameters β .

Vector Error Correction (VEC) Models

A vector error correction (VEC) model is a restricted VAR designed for use with nonstationary series that are known to be cointegrated. The VEC has cointegration relations built into the specification so that it restricts the long-run behavior of the endogenous variables to converge to their cointegrating relationships while allowing for short-run adjustment dynamics. The cointegration term is known as the *error correction* term since the deviation from long-run equilibrium is corrected gradually through a series of partial short-run adjustments.

To take the simplest possible example, consider a two variable system with one cointegrating equation and no lagged difference terms. The cointegrating equation is

$$y_{2,t} = \beta y_{1,t} \quad (20.27)$$

and the VEC model is

$$\begin{aligned} \Delta y_{1,t} &= \alpha_1(y_{2,t-1} - \beta y_{1,t-1}) + \epsilon_{1,t} \\ \Delta y_{2,t} &= \alpha_2(y_{2,t-1} - \beta y_{1,t-1}) + \epsilon_{2,t} \end{aligned} \quad (20.28)$$

In this simple model, the only right-hand side variable is the error correction term. In long run equilibrium, this term is zero. However, if y_1 and y_2 deviate from the long run equilibrium, the error correction term will be nonzero and each variable adjusts to partially restore the equilibrium relation. The coefficient α_i measures the speed of adjustment of the i -th endogenous variable towards the equilibrium.

How to Estimate a VEC

As the VEC specification only applies to cointegrated series, you should first run the Johansen cointegration test as described above and determine the number of cointegrating relations. You will need to provide this information as part of the VEC specification.

To set up a VEC, click the **Estimate** button in the VAR toolbar and choose the **Vector Error Correction** specification from the **VAR/VEC Specification** tab. In the **VAR/VEC Specification** tab you should provide the same information as for an unrestricted VAR, except that

- The constant or linear trend term should *not* be included in the **Exogenous Series** edit box. The constant and trend specification for VECs should be specified in the **Cointegration** tab (see below).
- The lag interval specification refers to *lags of the first difference terms* in the VEC. For example, the lag specification “1 1” will include lagged first difference terms on the right-hand side of the VEC. Rewritten in levels, this VEC is a restricted VAR with two lags. To estimate a VEC with no lagged first difference terms, specify the lag as “0 0”.

- The constant and trend specification for VECs should be specified in the **Cointegration** tab. You must choose from one of the five trend specifications as explained in “[Deterministic Trend Specification](#)” on page 538. You must also specify the number of cointegrating relations in the appropriate edit field. This number should be a positive integer less than the number of endogenous variables in the VEC.
- If you want to impose restrictions on the cointegrating relations and/or the adjustment coefficients, use the **Restrictions** tab. “[Restrictions on the Cointegrating Vector](#)” on page 543 describes this restriction in greater detail. Note that this tab is grayed out unless you have clicked the **Vector Error Correction** specification in the **VAR/VEC Specification** tab.

Once you have filled the dialog, simply click **OK** to estimate the VEC. Estimation of a VEC model is carried out in two steps. In the first step, we estimate the cointegrating relations from the Johansen procedure as used in the cointegration test. We then construct the error correction terms from the estimated cointegrating relations and estimate a VAR in first differences including the error correction terms as regressors.

VEC Estimation Output

The VEC estimation output consists of two parts. The first part reports the results from the first step Johansen procedure. If you did not impose restrictions, EViews will use a default normalization that identifies all cointegrating relations. This default normalization expresses the first r variables in the VEC as functions of the remaining $k - r$ variables, where r is the number of cointegrating relations and k is the number of endogenous variables. Asymptotic standard errors (corrected for degrees of freedom) are reported for parameters that are identified under the restrictions. If you provided your own restrictions, standard errors will not be reported unless the restrictions identify all cointegrating vectors.

The second part of the output reports results from the second step VAR in first differences, including the error correction terms estimated from the first step. The error correction terms are denoted `CointEq1`, `CointEq2`, and so on in the output. This part of the output has the same format as the output from unrestricted VARs as explained in “[VAR Estimation Output](#)” on page 521, with one difference. At the bottom of the VEC output table, you will see two log likelihood values reported for the system. The first value, labeled **Log Likelihood (d.f. adjusted)** is computed using the determinant of the residual covariance matrix (reported as **Determinant Residual Covariance**), using small sample degrees of freedom correction as in (20.3). This is the log likelihood value reported for unrestricted VARs. The **Log Likelihood** value is computed using the residual covariance matrix without correcting for degrees of freedom. This log likelihood value is comparable to the one reported in the cointegration test output.

Views and Procs of a VEC

Views and procs available for VECs are mostly the same as those available for VARs as explained above. Here we only mention those that are specific to VECs.

Cointegrating Relations

View/Cointegration Graph displays a graph of the estimated cointegrating relations as used in the VEC. To store these estimated cointegrating relations as named series in the workfile, use **Procs/Make Cointegration Group**. This proc will create and display an untitled group object containing the estimated cointegrating relations as named series. These series are named COINTEQ01, COINTEQ02 and so on.

Forecasting

Currently forecasts from a VAR or VEC are not available from the VAR object. Forecasts can be obtained by solving a model created from the estimated VAR/VEC. Click on **Procs/Make Model** from the VAR window toolbar to create a model object from the estimated VAR/VEC. You may then make any changes to the model specification, including modifying the ASSIGN statement before solving the model to obtain the forecasts. See [Chapter 23, “Models”, on page 601](#), for further discussion on how to forecast from model objects in EViews.

Data Members

Various results from the estimated VAR/VEC can be retrieved through the command line data members. [“Var Data Members” on page 50](#) of the *Command and Programming Reference* provides a complete list of data members that are available for a VAR object. Here we focus on retrieving the estimated coefficients of a VAR/VEC.

Obtaining Coefficients of a VAR

Coefficients of (unrestricted) VARs can be accessed by referring to elements of a *two dimensional array* C. The first dimension of C refers to the equation number of the VAR, while the second dimension refers to the variable number in each equation. For example, C(2,3) is the coefficient of the third regressor in the second equation of the VAR. The C(2,3) coefficient of a VAR named VAR01 can then be accessed by the command

```
var01.c(2,3)
```

To examine the correspondence between each element of C and the estimated coefficients, select **View/Representations** from the VAR toolbar.

Obtaining Coefficients of a VEC

For VEC models, the estimated coefficients are stored in three different two dimensional arrays: A, B, and C. A contains the adjustment parameters α , B contains the cointegrating

vectors β' and C holds the short-run parameters (the coefficients on the lagged first difference terms).

- The first index of A is the equation number of the VEC, while the second index is the number of the cointegrating equation. For example, A(2,1) is the adjustment coefficient of the first cointegrating equation in the second equation of the VEC.
- The first index of B is the number of the cointegrating equation, while the second index is the variable number in the cointegrating equation. For example, B(2,1) is the coefficient of the first variable in the second cointegrating equation. Note that this indexing scheme corresponds to the *transpose* of β .
- The first index of C is the equation number of the VEC, while the second index is the variable number of the first differenced regressor of the VEC. For example, C(2, 1) is the coefficient of the first differenced regressor in the second equation of the VEC.

You can access each element of these coefficients by referring to the name of the VEC followed by a dot and coefficient element:

```
var01.a(2,1)
```

```
var01.b(2,1)
```

```
var01.c(2,1)
```

To see the correspondence between each element of A, B, and C and the estimated coefficients, select **View/Representations** from the VAR toolbar.

A Note on EViews Backward Compatibility

The following changes made in Version 4 may yield VAR results that do not match those reported from previous versions of EViews:

- The estimated residual covariance matrix is now computed using the finite sample adjustment so the sum-of-squares is divided by $T - p$ where p is the number of estimated coefficients in each VAR equation. Previous versions of EViews divided the sum-of-squares by T .
- The standard errors for the cointegrating vector are now computed using the more general formula in Boswijk (1995), which also covers the restricted case.

Chapter 21. Pooled Time Series, Cross-Section Data

Data often contain information on cross-sectional units observed over time. In many cases, a relatively small number of cross-sectional units are observed over a number of periods. For example, you may have time series data on GDP for a number of European nations. Or perhaps you have state level data on unemployment observed over time. We term such data *pooled time series, cross-section data*.

EViews provides a number of specialized tools to help you work with pooled data. EViews will help you manage your data, perform operations in either the time series or the cross-section dimension, and apply estimation methods that account for the pooled structure of your data.

The EViews object that manages time series/cross-section data is called a *pool*. The remainder of this chapter will describe how to set up your data to work with pools, and how to define and work with pool objects.

Creating a Workfile for Pooled Data

Your first step will be to create a workfile to hold your pooled data. There is nothing special about the creation of this workfile. Simply select **File/New/Workfile...** and enter the frequency and dates for your workfile.

The range of your workfile should represent the earliest and latest dates that you wish to consider for any of the cross-section units. For example, if you want to work with data for Japan from 1950 to 1993, and data for the United States from 1956 to 1997, you should create a workfile ranging from 1950 to 1997.

The Pool Object

At the heart of a pool object is a list of names that you will use to refer to cross-section members. For reasons that will become apparent, these names should be relatively short. For example, in a cross-country study, you might use “USA” to refer to the United States, “CAN” to refer to Canada, “KOR” to identify Korea, “JPN” for Japan, and “UK” for the United Kingdom.

Defining the cross-section identifiers in a pool tells EViews about the structure of your data. In the example above, EViews will understand that when using this pool, you wish to work with separate time series data for each of the countries.

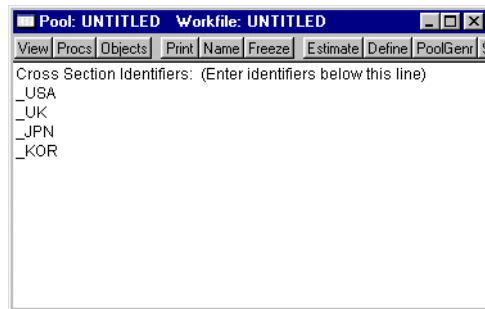
It is important to note that the pool object does not itself contain series or data. A pool object is simply a description of the underlying structure of your data. Consequently, delet-

ing the pool will not delete the series referenced by the pool, and modifying the series referenced by the pool will modify the data used by the pool.

Creating a Pool Object

To create a pool object, select **Objects/New Object/Pool...** and enter the identifiers in the edit window.

There are no special restrictions on the labels that you can use for cross-section identifiers, though you must be able to form legal EViews series names containing these identifiers. Here we have prepended the “_” character to each of the identifiers; this is not necessary, but we find that it makes it easier to spot the identifier when it is used as part of a series name.



Viewing or Editing a Pool Definition

To view the cross-section identifiers in your pool, push the **Define** button on the toolbar, or select **View/Cross-Section Identifiers**. If desired, you can edit the list of identifiers.

Copying a Pool Object

Typically, you will work with more than one pool object. Multiple pools are used to define various subsamples of cross-section identifiers, or to work with different pooled estimation specifications.

To copy a pool object, open the original pool, and select **Objects/Copy Object...** Alternatively, you can highlight the name of the pool in the workfile window, select **Objects/Copy Selected...** and enter the new name.

Working with Pools and Series

Even though you will be working with pooled data, all of your data will be held in ordinary EViews series. These series can be used in the usual ways: among other things, they can be tabulated, graphed, used to generate new series, or used in estimation. You can also use the pool object to work with the individual series.

Naming Your Series

The key to using series with pools is to name your series using a combination of a *base name* and the cross-section identifier. The cross-section identifier may be embedded at an arbitrary location in the series name, so long as this is done consistently across identifiers.

For example, suppose that you have a pool object that contains the identifiers `_JPN`, `_USA`, `_UK`, and that you have time series measurements on GDP for each of the cross-section units. We will use “GDP” as the base name for our series.

You can choose to place the identifier at the end of the base name, in which case, you should name your series `GDP_JPN`, `GDP_USA`, and `GDP_UK`. Alternatively, you can elect to put the section identifiers, `JPN_`, `USA_`, `UK_`, in front of the name, so the series will be named `JPN_GDP`, `USA_GDP`, `UK_GDP`. The identifiers may also be placed in the middle of series names—for example, `GDP_JPNINFO`, `GDP_USAINFO`, `GDP_UKINFO`.

It really doesn’t matter whether the identifiers are used at the beginning, end, or middle of your names; you should pick the naming style that you find easiest to work with. Consistency, however, is absolutely essential. You should not name your three series `JPNGDP`, `GDPUSA`, `UKGDP1`, as EViews would be unable to reference these series using a pool object.

Pool Series

Once the series names have been chosen to correspond with the identifiers in your pool, the pool object can be used to work with a set of series. The key to this processing is the concept of a *pool series*.

A pool series is actually a group of series defined by a base name and the entire list of cross-section identifiers. Pool series are specified using the base name, and a “?” character placeholder for the cross-section identifier. If your series are named `GDPJPN`, `GDPUSA`, `GDPUK`, the corresponding pool series will be referred to as `GDP?`. If the names of your series are `JPNGDP`, `USAGDP`, and `UKGDP`, the pool series will be `?GDP`.

When you use a pool series name, EViews understands that you wish to work with all of the series in the pool series. EViews loops through the list of cross-section identifiers and substitutes each identifier in place of the “?”. EViews will process your instructions using the constructed names.

A pool series may only be accessed through the pool object, since the placeholder “?” has no meaning without a list of cross-section identifiers. If you attempt to use a pool variable outside of a pool object, you will receive an error message saying that your variable is not defined, or EViews will interpret the “?” as a wildcard character (see [Appendix C, “Wildcards”](#), on page 657).

Importing Pooled Data

There are several ways to import data so that they may be used in pooled data analysis. Before considering the various methods, we will need to understand the structure of pooled time series, cross-section data, and to distinguish between data that are in *unstacked* and *stacked* form.

A given piece of information in a pooled setting may be indexed along three dimensions: the variable, the time period, and the cross-section. For example, you may be interested in the value of GDP, in 1989, for Korea.

Working with your data in three dimensions is difficult, so typically you will employ a two-dimensional representation of your data. There is no unique way to organize three-dimensional data in two-dimensions, but several formats are commonly employed.

Unstacked Data

In this form, observations on a given variable for a given cross-section are grouped together, but are separated from observations for other variables and other cross sections. For example, suppose the top of our data file contains the following:

year	c_usa	c_kor	c_jpn	g_usa	g_jpn	g_kor
1954	61.6	77.4	66	17.8	18.7	17.6
1955	61.1	79.2	65.7	15.8	17.1	16.9
1956	61.7	80.2	66.1	15.7	15.9	17.5
1957	62.4	78.6	65.5	16.3	14.8	16.3
...

Here, the base name C represents consumption, while G represents government expenditure. Each country has its own separately identified column for consumption, and its own column for government expenditure.

EViews works naturally with data that are *unstacked* and will read unstacked data using the standard input procedures described in [Chapter 4, “Basic Data Handling”](#), page 64. Simply read each cross-section specific variable as an individual series, making certain that the names of the series follow the pool naming conventions described above.

Stacked Data

Pooled data can also be arranged in stacked form, where all of the data for a variable are grouped together, but separated from data for other variables. In the most common form, the data for different cross-sections are stacked on top of one another, with each column representing a variable.

id	year	c	g
_usa	1954	61.6	17.8
...
_usa	1992	68.1	13.2
_uk	1954	62.4	23.8
...
_uk	1992	67.9	17.3
_jpn	1954	66	18.7
...
_jpn	1992	54.2	7.6
_kor	1954	77.4	17.6
...
_kor	1992	na	na

We say that these data are *stacked by cross-section*. Alternatively, we may have data that are *stacked by date*.

per	id	c	g
1954	_usa	61.6	17.8
1954	_uk	62.4	23.8
1954	_jpn	66	18.7
1954	_kor	77.4	17.6
...
1992	_usa	68.1	13.2
1992	_uk	67.9	17.3
1992	_jpn	54.2	7.6
1992	_kor	na	na

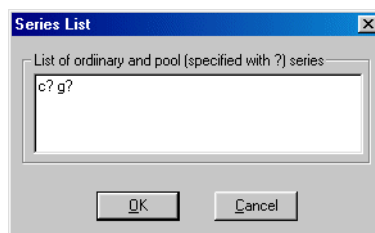
Each column again represents a variable, but within each column, the data are arranged by year. If data are stacked by year, you should make certain that the ordering of the cross-sectional identifiers within a year is consistent across years.

Manual Entry/Cut-and-Paste

One way to import data is by manually entering, or copying-and-pasting into a stacked representation of your pool series:

- First, specify which time series observations will be included in your stacked spreadsheet by setting the workfile sample.
- Next, open the pool, then select **View/Spreadsheet View...** EViews will prompt you for a list of series. You can enter ordinary series names or pool series names. If the series exist, then EViews will display the data in the series. If the series do not exist, then EViews will create the series or group of series, using the cross-section identifiers if you specify a pool series.
- EViews will open the stacked spreadsheet view of the pool series. If desired, click on the **Order** +/- button to toggle between stacking by cross-section and stacking by date.
- Click **Edit** +/- to turn on edit mode in the spreadsheet window, and enter your data, or cut-and-paste from another application.

For example, if we have a pool that contains the identifiers `_USA`, `_UK`, `_JPN`, and `_KOR`, we can instruct EViews to create the series `C_USA`, `C_UK`, `C_JPN`, `C_KOR`, and `G_USA`, `G_UK`, `G_JPN`, `G_KOR` by entering:



EViews will open a stacked spreadsheet view of the newly created series. Here we see the series stacked by cross-section, with the pool series names in the column header, and the cross-section/date identifiers labeling each row.

If desired, click on **Order** +/- to toggle between stacking methods. Click on **Edit** +/- to turn on edit mode, and enter or cut-and-paste into the window.

File Import

You can import stacked data from a file into individual series using a pool object. While the data in the file may be stacked either by cross-section or by period, EViews requires that:

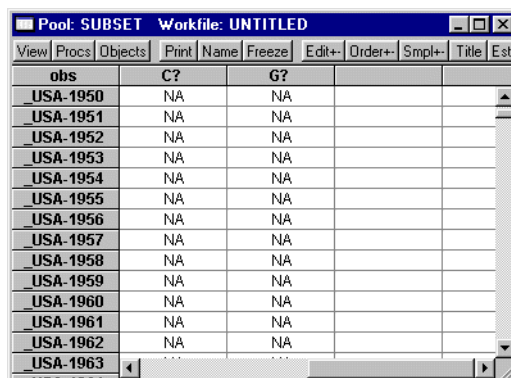
- The stacked data are *balanced*.
- The cross-sections are ordered in the file in the same way that the cross-sectional identifiers are listed in the pool.

By “balanced”, we mean that if the data are stacked by cross-section, each cross-section should contain exactly the same number of periods—if the data are stacked by date, each date should have exactly the same number of cross-sectional observations arranged in the same order.

We emphasize that *the underlying data need not be balanced, only the representation in the import file*. If you have missing values for some observations, you should make certain that there are lines in the file representing the missing values. In the two examples above, the underlying data are not balanced, since information is not available for Korea in 1992. The data in the file have been balanced by including an observation for the missing data.

To read data from a file using a pool object, first open the pool, then select **Procs/Import Pool data (ASCII, .XLS, .WK?)...** It is important that you use the import procedure associated with the pool object, and not the standard file import procedure.

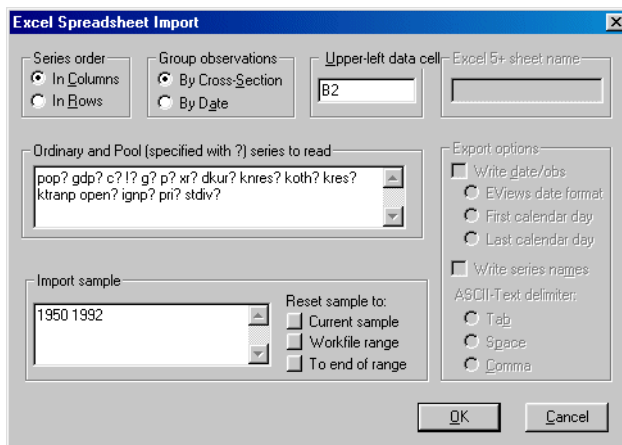
Select your input file in the usual fashion. If you select a spreadsheet file, EViews will open a spreadsheet import dialog prompting you for additional input.



obs	C?	G?
USA-1950	NA	NA
USA-1951	NA	NA
USA-1952	NA	NA
USA-1953	NA	NA
USA-1954	NA	NA
USA-1955	NA	NA
USA-1956	NA	NA
USA-1957	NA	NA
USA-1958	NA	NA
USA-1959	NA	NA
USA-1960	NA	NA
USA-1961	NA	NA
USA-1962	NA	NA
USA-1963	NA	NA

Much of this dialog should be familiar from the discussion in [Chapter 4, “Basic Data Handling”](#), on [page 55](#). Fill out the various parts of the dialog:

- Indicate whether the pool series are in rows or in columns, and whether the data are stacked by cross-section, or stacked by date.
- In the edit box, enter the names of the series you wish to import. These series names can be ordinary series names or pool names.
- Fill in the sample information, starting cell location, and optionally, the sheet name.



When you specify your series using pool series names, EViews will, if necessary, create and name the series using the cross-section identifiers. If you list an ordinary series name, EViews will, if needed, create the single series.

EViews will read from your file into the specified variables using the sample information. If you enter an ordinary series name, EViews will read multiple values into the series, so that, upon completion, the series will contain the last set of values read from the file.

The basics of importing from ASCII files are analogous, but the corresponding dialog contains many additional options to handle the complexity of ASCII files. For details, see [“Addendum: Reading ASCII Files”](#) on [page 76](#).

Exporting Pooled Data

You can export your data by reversing the procedures described above for data input. Since EViews allows you to read and write data that are unstacked, stacked by cross-section, or stacked by date, you can use EViews to restructure your data in accordance with your needs.

Working with Pooled Data

The underlying series for each cross-section member are ordinary series, so all of the EViews tools for working with the individual cross-section series are available. In addition, EViews provides you with a number of specialized tools which allow you to work with

your pool data. Using EViews, you can perform, in a single step, similar operations on all the series corresponding to a particular pooled variable.

Examining Data

As described above, you can view your data in stacked spreadsheet form. Select **View/Spreadsheet View...**, and list the series you wish to display. The names can include both ordinary and pool series names. Click on the **Order** +/- button to toggle between stacking your observations by cross-section and by date.

We emphasize that stacking your data only provides an alternative view of the data, and does not change the structure of the individual series in your workfile. Stacking data is not necessary for any of the data management or estimation procedures described below.

Describing Data

You may compute descriptive statistics for your series using a pool object. Select **View/Descriptive Statistics...** from the pool toolbar.

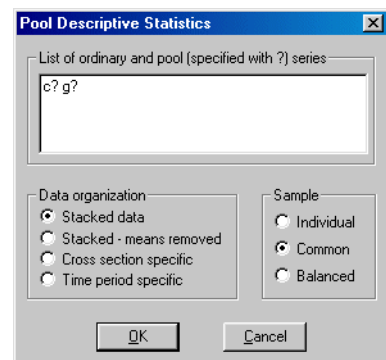
In the edit box, you should list the ordinary and pooled series for which you want to compute the descriptive statistics. EViews will compute the mean, median, minimum, maximum, standard deviation, skewness, kurtosis, and the Jarque-Bera statistic for these series.

Next you should choose between the three sample options:

- **Individual:** uses the maximum number of observations available. If an observation on a variable is available for a particular cross-section, it is used in computation.
- **Common:** uses an observation only if data on the variable are available for all cross-sections in the same period. This method is equivalent to performing listwise exclusion by variable, then cross-sectional casewise exclusion within each variable.
- **Balanced:** includes observations when data on all variables in the list are available for all cross-sections in the same period. The balanced option performs casewise exclusion by both variable and cross-section.

Lastly, you should choose the computational method corresponding to one of the four data structures:

- **Stacked data:** display statistics for each variable in the list, computed over all cross-sections and periods. These are the descriptive statistics that you would get if you



ignored the pooled nature of the data, stacked the data, and computed descriptive statistics.

- **Stacked – means removed:** compute statistics for each variable in the list after removing the cross-sectional means, taken over all cross-sections and periods.
- **Cross-section specific:** show the descriptive statistics for each cross-sectional variable, computed across all periods. These are the descriptive statistics derived by computing statistics for the individual series.
- **Time period specific:** compute period-specific statistics. For each period, compute the statistic using data on the variable from all the cross-sectional units in the pool.

Be aware that the latter two methods may generate a great deal of output. Cross-section specific computation generates a set of statistics for each variable/cross-section combination. If you ask for statistics for three pool variables and there are 20 cross-sections in your pool, EViews will compute descriptive statistics for 60 series. For time-period specific computation you will compute a set of statistics for each date/variable combination. If you have a sample with 100 periods and you list three pool variables, EViews will compute 300 sets of statistics.

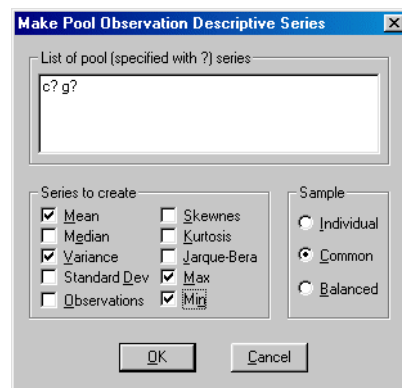
You can save the period-specific statistics in series objects. Select **Procs/Make Period Stat Series...** from the pool window, and fill out the dialog.

In the edit window, list the series for which you wish to calculate period-statistics. Next, select the particular statistics you wish to compute, and choose a sample option.

EViews will save your statistics in new series, and will open an untitled group window to display the results. The series will be named automatically using the base name followed by the name of the statistic (MEAN, MED, VAR, SD, OBS, SKEW, KURT, JARQ, MAX, MIN). In this example, EViews will save the statistics using the names CMEAN, GMEAN, CVAR, GVAR, CMAX, GMAX, CMIN, and GMIN.

Generating Data

You can generate or modify pool series using the **PoolGenr** procedure. Click on **PoolGenr** on the pool toolbar and enter your formula as you would for a regular **Genr**, using pool names as appropriate. Using our example from above, entering:



```
ratio? = g?/g_usa
```

is equivalent to entering the following four commands:

```
ratio_usa = g_usa/g_usa
ratio_uk = g_uk/g_usa
ratio_jpn = g_jpn/g_usa
ratio_kor = g_kor/g_usa
```

PoolGenr applies the formula you supply using an implicit loop across cross-section identifiers, creating or modifying series as appropriate.

You may use **PoolGenr** and **Genr** together to generate new variables. For example, to create a dummy variable that is equal to 1 for the US and 0 for all other countries, first select **PoolGenr** and enter:

```
dum? = 0
```

to initialize all of the dummy variable series to 0. Then, to set the US values to 1, select **Quick/Generate Series...** from the main menu, and enter:

```
dum_usa = 1
```

To modify a set of series using a pool, select **PoolGenr**, and enter the new pool series expression:

```
dum? = dum? * (g? > c?)
```

You can also use the implicit looping feature to perform calculations across cross-sectional units in a given period. For example, if we have an ordinary series SUM which is initialized to zero, then a **PoolGenr** using the expression:

```
sum = sum + c?
```

will calculate the ordinary series given by the ordinary **Genr**:

```
sum = c_usa + c_uk + c_jpn + c_kor
```

Note that this example is merely to illustrate the notion of implicit looping. As we saw above, EViews provides you with built-in features to compute period-specific statistics of this type.

Make Pool Group

You may wish to work with a set of pool series using standard EViews tools for group objects. Select **Procs/Make Group...** and enter the names of ordinary and pool series. EViews will create an untitled group object containing the series.

Deleting/Storing/Fetching Data

Pools may be used to delete, store, or fetch sets of series. Simply select **Procs/Delete pool series...**, **Procs/Store pool series (DB)...**, or **Procs/Fetch pool series (DB)...** as appropriate, and enter the ordinary and pool series names of interest.

If, for example, you instruct EViews to delete the pool series C?, EViews will loop through all of the cross-section identifiers and delete all series whose names begin with the letter “C” and end with the cross-section identifier.

Pooled Estimation

There are a number of ways that you can use information about the structure of your pooled data in estimating an equation. You might estimate a fixed or random intercept model, or perhaps a model with selected variables that have different coefficients across cross-sections, as well as separate AR(1) coefficients. Or you could estimate a separate equation for each cross-sectional unit.

EViews pool objects allow you to estimate your model using least squares, weighted least squares with estimated cross-section weights, or seemingly unrelated regressions, all without rearranging or reordering your data.

Below, we describe how you can use pools and systems to estimate more general and complex models, including two-stage least squares and nonlinear specifications, or models with complicated cross-section coefficient restrictions.

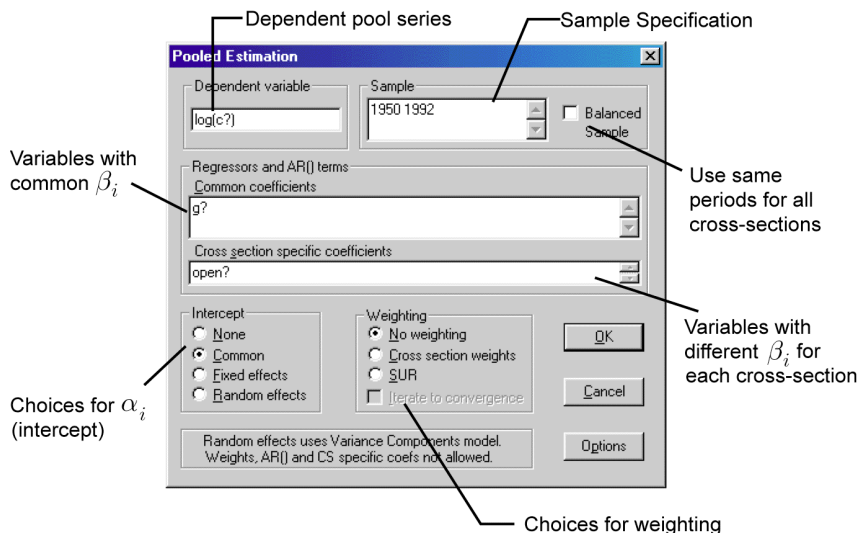
How to Estimate a Pool Equation

Generally speaking, pool objects can be used to estimate equations of the form:

$$y_{it} = \alpha_{it} + x_{it}'\beta_i + \epsilon_{it}, \quad (21.1)$$

for $i = 1, 2, \dots, N$ cross-section units and periods $t = 1, 2, \dots, T$. Further details and discussion of various pool equation specifications is provided in [“Pooled Estimation” on page 562](#).

Press the **Estimate** button on your pool toolbar, and the following dialog will open:



Dependent Variable

List a pool variable, or an EViews expression containing a pool variable, in the Dependent Variable edit box.

Sample

Enter your sample specification in the edit window at the upper right.

By default, EViews will use the largest sample possible in each cross-section. An observation will be excluded if any of the explanatory or dependent variables *for that cross-section* are unavailable in that period.

The checkbox for **Balanced Sample** instructs EViews to perform listwise exclusion over all cross-sections. EViews will eliminate an observation if data are unavailable *for any cross-section* in that period. This exclusion ensures that estimates for each cross-section will be based on a common set of dates.

If all of the observations for a cross-section unit are not available, that unit will temporarily be removed from the pool for purposes of estimation. The EViews output will inform you if any cross-section units were dropped.

Explanatory Variables

Next, you will list your regressors. There are two edit boxes where you will enter your explanatory variables:

- **Common coefficients:** — enter variables that are to have the same coefficient across all cross-section members of the pool. EViews will include a single coefficient for each variable, and will label the output using the ordinary or pool name, as appropriate.
- **Cross-section specific coefficients:** — list variables with different coefficients for each member of the pool. EViews will include a different coefficient for each cross-sectional unit, and will label the output using the cross-section identifier followed by “—” and then the ordinary series name.

For example, if you include the ordinary variable TIME and POP? in the common coefficient list, the output will include estimates for TIME and POP?. If you include these variables in the cross-section specific list, the output will include coefficients of the form `_USA—TIME`, `_UK—TIME`, and `_USA—POP_USA`, `_UK—POP_UK`, etc.

Be aware that estimating your model with cross-section specific variables may generate large numbers of coefficients—the number of these coefficients equals the product of the number of pool identifiers and the number of variables in the list.

You may include AR terms in your specification. If the terms are entered in the common coefficients list, EViews will estimate the model assuming a common AR error. If the AR terms are entered in the cross-section specific list, EViews will estimate separate AR terms for each member. See “[Estimating AR Models](#)” on page 307 for a description of AR specifications.

Note that EViews only allows specification by list for pool equations. If you wish to estimate a nonlinear specification, you must first create a system object, and then edit the system specification (see “[Creating Systems using Pools](#)” on page 570).

Intercept

In the area labeled **Intercept**: you can choose between alternative specifications for α_i :

None	no intercepts: $\alpha_{it} = 0$.
Common	identical intercept for all pool members: $\alpha_{it} = \alpha$.
Fixed effects	different intercepts estimated for each pool member: $\alpha_{it} = \alpha_i$.
Random effects	treats intercepts as random variables across pool members: $\alpha_{it} = \alpha_i$, $E(\alpha_i \epsilon_{it}) = 0$.

You cannot estimate random effects models with cross-section specific coefficients, AR terms, or weighting.

Weights

EViews does not weight observations in pooled estimation by default, but you have the option of estimating weighted versions of your specifications. There are three options for weights:

No weighting	all observations are given equal weight.
Cross section weights	GLS using estimated cross-section residual variances.
SUR	analogue to Seemingly Unrelated Regression—GLS using estimated cross-section residual covariance matrix.

If you select **Cross section weights**, EViews will estimate a feasible GLS specification assuming the presence of cross-section heteroskedasticity. If you select **SUR**, EViews estimates a feasible GLS specification correcting for both cross-section heteroskedasticity and contemporaneous correlation. This specification is sometimes referred to as the Parks estimator. More detailed descriptions of these methods are provided in the [“Technical Discussion” on page 571](#) below.

Bear in mind that there are a number of potential pitfalls associated with SUR/Parks estimation (see Beck and Katz (1995)). Furthermore, EViews may be unable to compute estimates for this model when you have large numbers of cross-sections or a small number of time periods. The average number of periods used in estimation must be at least as large as the number of cross-section units. Even if you have a sufficient number of observations, the estimated residual correlation matrix must also be nonsingular. If either condition is violated, EViews will display a “Near Singular Matrix” error.

There is a checkbox labeled **Iterate to convergence** which controls the feasible GLS procedures. If selected, EViews will continue to update the weights and coefficients until they converge. This option has no effect if your model includes AR terms since EViews will always iterate weights and coefficients to convergence in AR specifications.

Options

Iteration and Convergence Options. If you select weighted estimation and ask EViews to iterate to convergence, you can control the iterative process by specifying convergence criterion and the maximum number of iterations. Press the **Options** button in the equation dialog box and enter the desired values.

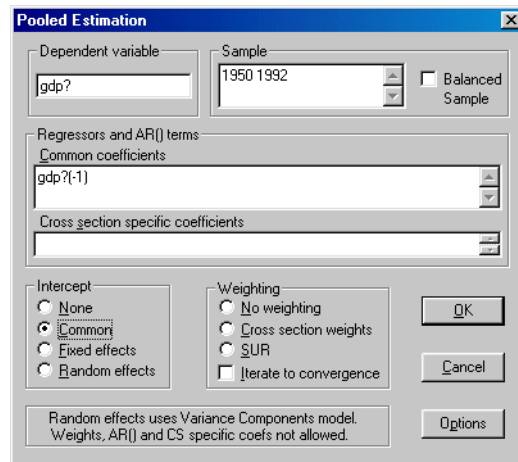
White Heteroskedasticity Covariance. EViews can estimate covariances that are robust to general heteroskedasticity. This form of heteroskedasticity is more general than the cross-section heteroskedasticity described above, since variances within a cross-section are allowed to differ across time.

To request White standard errors and covariances, press the **Options** button, and select White Heteroskedasticity Consistent Covariance. Note that this option is not available with SUR and random effects estimation.

Pool Equation Examples

As an illustration, we take data from the Penn World Table for GDP for seven countries: Canada, France, Germany, Italy, Japan, United Kingdom, and the US. The corresponding pool identifiers are `_CAN`, `_FRA`, `_GER`, `_ITA`, `_JPN`, `_UK`, `_US`.

First, we estimate a model regressing GDP_t^i on GDP_{t-1}^i (note that the '?' precedes the lag specification). All coefficients are restricted to be the same across all cross-sections, so this is equivalent to estimating a model on the stacked data, ignoring all cross-sectional information:



The output from this regression is given by:

Dependent Variable: GDP?
 Method: Pooled Least Squares
 Date: 10/20/97 Time: 15:45
 Sample(adjusted): 1951 1992
 Included observations: 42 after adjusting endpoints
 Total panel observations 294

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	245.3620	41.47131	5.916428	0.0000
GDP?(-1)	1.000448	0.004083	245.0305	0.0000
R-squared	0.995160	Mean dependent var		9681.432
Adjusted R-squared	0.995144	S.D. dependent var		3786.743
S.E. of regression	263.8918	Sum squared resid		20334560
Log likelihood	-1818.911	F-statistic		60039.95
Durbin-Watson stat	1.488062	Prob(F-statistic)		0.000000

Note that there is a single coefficient for both the constant C, and the pool series for lagged GDP.

Next, suppose we estimate the same specification, but select **Fixed effects** for the intercept specification, and select **Cross-section weights** for our weighting. This implies that each pool will have an unrestricted intercept, and that each pool equation is downweighted by an estimate of the cross-section residual standard deviation. The results are given by:

Dependent Variable: GDP?
 Method: GLS (Cross Section Weights)
 Date: 10/20/97 Time: 15:49
 Sample: 1951 1992
 Included observations: 42
 Total panel observations 294
 Convergence achieved after 1 iteration(s)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
GDP?(-1)	1.004838	0.004088	245.8188	0.0000
Fixed Effects				
_CAN--C	189.8252			
_FRA--C	197.0985			
_GER--C	239.3465			
_ITA--C	208.5929			
_JPN--C	298.8015			
_UK--C	136.0450			
_US--C	158.0119			
Weighted Statistics				
R-squared	0.996134	Mean dependent var		10507.92
Adjusted R-squared	0.996039	S.D. dependent var		4168.432
S.E. of regression	262.3466	Sum squared resid		19684159
Log likelihood	-1806.950	Durbin-Watson stat		1.504483
Unweighted Statistics				
R-squared	0.995313	Mean dependent var		9681.432
Adjusted R-squared	0.995199	S.D. dependent var		3786.743
S.E. of regression	262.3922	Sum squared resid		19691008
Durbin-Watson stat	1.543536			

Note that EViews displays both the estimates of the fixed effects, and summary statistics for the weighted and unweighted models. The estimates of the fixed effects do not have reported standard errors since EViews treats them as nuisance parameters for the purposes of estimation. If you wish to compute standard errors for the cross-section constants, you should estimate a model without a constant and explicitly enter the C in the **Cross section specific coefficients** edit field.

We can generalize this specification by estimating a separate coefficient for lagged GDP in each cross section. Simply click on the **Estimate** button to reestimate the pool, move GDP?(-1) to the **Cross section specific coefficients** edit field, and click OK. The top portion of the output is given below:

Dependent Variable: GDP?
 Method: GLS (Cross Section Weights)
 Date: 10/20/97 Time: 15:56
 Sample: 1951 1992
 Included observations: 42
 Total panel observations 294
 Convergence achieved after 1 iteration(s)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
_CAN--GDP_CAN(-1)	0.998332	0.016368	60.99230	0.0000
_FRA--GDP_FRA(-1)	0.997262	0.008985	110.9879	0.0000
_GER--GDP_GER(-1)	0.996295	0.010610	93.90234	0.0000
_ITA--GDP_ITA(-1)	1.006682	0.008819	114.1450	0.0000
_JPN--GDP_JPN(-1)	1.019567	0.008179	124.6552	0.0000
_UK--GDP_UK(-1)	1.003127	0.015082	66.51012	0.0000
_US--GDP_US(-1)	0.996048	0.018346	54.29327	0.0000
Fixed Effects				
_CAN--C	261.2705			
_FRA--C	266.0358			
_GER--C	318.7881			
_ITA--C	194.7066			
_JPN--C	192.1386			
_UK--C	151.0268			
_US--C	273.2745			

The coefficients on GDP? have lengthy labels formed by taking the cross-section specific variable (for example GDP_CAN(-1)), and prepending the cross-section identifier to denote the equation for which this coefficient is relevant. For example, the first coefficient is the coefficient of GDP_CAN(-1) in the _CAN equation. Note that this prepending is necessary for identification purposes since you could, for example, have included the GDP_UK(-1) variable as an explanatory variable in the _JPN equation.

Lastly, we can estimate a model with random effects

Dependent Variable: GDP?
 Method: GLS (Variance Components)
 Date: 10/20/97 Time: 16:08
 Sample: 1951 1992
 Included observations: 42
 Total panel observations 294

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	237.1665	43.99291	5.391016	0.0000
GDP?(-1)	1.001317	0.004233	236.5521	0.0000
Random Effects				
_CAN--C	-2.714267			
_FRA--C	-2.512245			
_GER--C	10.92250			
_ITA--C	-0.644577			
_JPN--C	27.25335			
_UK--C	-21.98726			
_US--C	-10.31750			

GLS Transformed Regression			
R-squared	0.995206	Mean dependent var	9681.432
Adjusted R-squared	0.995190	S.D. dependent var	3786.743
S.E. of regression	262.6320	Sum squared resid	20140863
Durbin-Watson stat	1.503691		

Unweighted Statistics including Random Effects			
R-squared	0.995238	Mean dependent var	9681.432
Adjusted R-squared	0.995222	S.D. dependent var	3786.743
S.E. of regression	261.7485	Sum squared resid	20005585
Durbin-Watson stat	1.513859		

We have simplified the specification so that GDP?(-1) has a common coefficient. Note that EViews provides estimates of the realization of the random effect components, as well as summary statistics for both the transformed and the untransformed data (where the latter is evaluated after removing the estimated random component). The details on these computations are presented in the [“Technical Discussion”](#) beginning on page 571.

Pool Equation Views and Procedures

Once you have estimated your pool equation, you can examine your output in the usual ways:

Representation

Select **View/Representations** to examine your specification. EViews estimates your pool as a system of equations, one for each cross-section unit.

Estimation Output

View/Estimation Output will change the display to show the results from the pooled estimation.

As with other estimation objects, you can examine the estimates of the coefficient covariance matrix by selecting **View/Coef Covariance Matrix**.

Testing

EViews allows you to perform coefficient tests on the estimated parameters of your pool equation. Select, **View/Wald Coefficient Tests...** and enter the restriction to be tested.

Residuals

You can view your residuals in spreadsheet or graphical format by selecting **View/Residuals/Table** or **View/Residuals/Graph**. EViews will display the residuals for each cross-sectional equation. Each residual will be named using the base name RES, followed by the cross-section identifier.

If you wish to save the residuals in series using these names, select **Procs/Make Resids**.

Residual Covariance/Correlation

You can examine the estimated residual contemporaneous covariance and correlation matrices. Select **View/Residual** and then either **Correlation Matrix** or **Covariance Matrix** to examine the appropriate matrix.

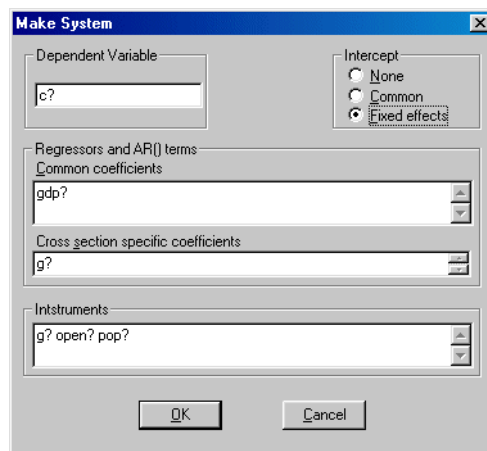
Forecasting

To perform forecasts using a pool equation you will first make a model. Select **Procs/Make Model** to create an untitled model object that incorporates all of the estimated coefficients. If desired, this model can be edited. Solving the model will generate forecasts for the dependent variable for each of the cross-section units. For further details, see [Chapter 23, “Models”](#), on page 601.

Creating Systems using Pools

You may have a complex pooled equation specification that cannot be estimated using the pool object. To apply more general estimation methods, such as two-stage least squares, three-stage least squares, and GMM, or to impose arbitrary coefficient restrictions, you should use the pool object to create a system object. You can create a system from an estimated pool, or you can provide information to generate a system from a pool. The system object can be further customized, and estimated using advanced techniques.

Select **Procs/Make System...** and fill out the dialog. You can enter the dependent variable, specify the intercept, common and cross-section specific variables as before. You can also enter a list of instrumental variables. Ordinary variables will appear as an instrument in each equation; pool variables will enter cross-section specific variables as instruments in the corresponding cross-section equation.



Commands

To create a new pool object, follow the `pool` command keyword with a name for the pool object:

```
pool g7
```

creates a new pool object named G7.

To define the cross-section identifiers of the pool, use the `define` command:

```
g7.define us uk ita ger fra can jpn
```

defines the cross section members of the pool object G7 as US, UK, ITA, GER, FRA, CAN, and JPN.

To estimate a pool with fixed effects, use the `f` option:

```
g7.ls(f) cs? @ gdp?
```

estimates a fixed effects model without constraining the coefficients on GDP to be the same for each pool member (which is the same as running OLS for each member separately).

See “[Pool](#)” on page 34 of the *Command and Programming Reference* for a complete list of commands and options available for pool objects.

Technical Discussion

The class of models that can be estimated using a pool object can be written as

$$y_{it} = \alpha_{it} + x_{it}'\beta_i + \epsilon_{it}, \quad (21.2)$$

where y_{it} is the dependent variable, and x_{it} and β_i are k -vectors of non-constant regressors and parameters for $i = 1, 2, \dots, N$ cross-sectional units. Each cross-section unit is observed for dated periods $t = 1, 2, \dots, T$.

While most of our discussion will be in terms of a balanced sample, EViews does not require that your data be balanced; missing values may be used to represent observations that are not available for analysis in a given period. We will describe the behavior of the estimator in the presence of missing values only where there is the potential for ambiguity.

We can view these data as a set of cross-section specific regressions so that we have N cross-sectional equations:

$$y_i = \alpha_i + x_i' \beta_i + \epsilon_i \quad (21.3)$$

each with T observations, stacked on top of one another. For purposes of discussion we will refer to the stacked representation:

$$Y = \alpha + X\beta + \epsilon \quad (21.4)$$

where α , β and X are set up to include any restrictions on the parameters between cross-sectional units.

The residual covariance matrix for this set of equations is given by:

$$\Omega = E(\epsilon\epsilon') = E \begin{pmatrix} \epsilon_1\epsilon_1' & \epsilon_2\epsilon_1' & \dots & \epsilon_N\epsilon_1' \\ \epsilon_2\epsilon_1' & \epsilon_2\epsilon_2' & & \vdots \\ & & \ddots & \\ \epsilon_N\epsilon_1' & \dots & & \epsilon_N\epsilon_N' \end{pmatrix} \quad (21.5)$$

The basic specification treats the pool specification as a system of equations and estimates the model using system OLS. This specification is appropriate when the residuals are contemporaneously uncorrelated, and time-period and cross-section homoskedastic:

$$\Omega = \sigma^2 I_N \otimes I_T. \quad (21.6)$$

The coefficients and their covariances are estimated using the usual OLS techniques applied to the stacked model.

Fixed Effects

The fixed effects estimator allows α_i to differ across cross-section units by estimating different constants for each cross-section. EViews computes the fixed effects by subtracting the “within” mean from each variable and estimating OLS using the transformed data:

$$y_i - \bar{y}_i = (\bar{x}_i - \bar{x}_i)' \beta + (\epsilon_i - \bar{\epsilon}_i) \quad (21.7)$$

where $\bar{y}_i = \sum_t y_{it}/T$, $\bar{x}_i = \sum_t x_{it}/T$, and $\bar{\epsilon}_i = \sum_t \epsilon_{it}/T$.

The coefficient covariance matrix estimates are given by the usual OLS covariance formula applied to the mean differenced model:

$$\text{var}(b_{FE}) = \hat{\sigma}_W^2 (\tilde{X}' \tilde{X})^{-1} \quad (21.8)$$

where \tilde{X} represents the mean differenced X , and

$$\hat{\sigma}_W^2 = \frac{e_{FE}' e_{FE}}{NT - N - K} = \frac{\sum_{it} (\tilde{y}_{it} - \tilde{x}_{it}' b_{FE})^2}{NT - N - K}, \quad (21.9)$$

where $e_{FE}' e_{FE}$ is the SSR from the fixed effects model. If the pool is unbalanced, NT is replaced by the total number of observations excluding missing values.

The fixed effects themselves are not estimated directly. We report estimated fixed effects computed from

$$\hat{\alpha}_i = \sum_t (\bar{y}_i - \bar{x}_i' b_{FE}) / N \quad (21.10)$$

Standard errors are not reported for the fixed effects coefficients. If you wish to obtain standard errors for the fixed effects, you should re-estimate a model with no intercept, including the constant term as a cross-section specific regressor.

You should note that estimating the cross-section specific constant regression model with a large number of cross-section units may be time-consuming, and may result in estimates that are less accurate than those obtained using the fixed-effects option.

Random Effects

The random effects model assumes that the term α_{it} is the sum of a common constant α and a time-invariant cross-section specific random variable u_i that is uncorrelated with the residual ϵ_{it} . EViews estimates the random effects model using the following steps:

- (1) Use the residuals e_{FE} from the fixed effects model to estimate the variance of ϵ_{it} using $\hat{\sigma}_W^2$ as described above.
- (2) Estimate the between-group (cross-sectional mean) model and compute:

$$\hat{\sigma}_B^2 = \frac{e_B' e_B}{N - K}, \quad \hat{\sigma}_u^2 = \hat{\sigma}_B^2 - \frac{\hat{\sigma}_W^2}{T} \quad (21.11)$$

where $e_B' e_B$ is the SSR from the between-group regression. If the estimated $\hat{\sigma}_u^2$ is negative, EViews will return an error message.

In the case where there are missing observations so that T_i varies among cross-sections, EViews will use the largest T_i in computing the variance estimates. This procedure is consistent, provided that the number of missing observations is asymptotically negligible.

(3) Apply OLS to the GLS transformed variables (X includes the constant term and the regressors x)

$$y_{it}^* = y_{it} - \hat{\lambda}\bar{y}_i, X_{it}^* = X_{it} - \hat{\lambda}\bar{X}_i \quad (21.12)$$

where $\hat{\lambda} = 1 - \hat{\sigma}_W/\hat{\sigma}_B$.

The EViews output displays the parameter estimates for the β derived from (3). The standard errors are computed using the standard estimator of the covariance matrix.

EViews also presents estimates of the values of the random effects. These values are computed using the formula:

$$\hat{u}_i = \frac{\hat{\sigma}_u^2}{\hat{\sigma}_B^2} \left(\sum_t (y_{it} - X_{it}b_{RE}) \right), \quad (21.13)$$

yielding the best linear unbiased predictor of u_i .

(4) Lastly, EViews displays weighted and unweighted summary statistics. The weighted statistics are from the GLS equation estimated in step (3). The unweighted statistics are derived using residuals from the original model based upon the parameters and the estimated random effect from step (3):

$$\epsilon_{it} = y_{it} - X_{it}b_{RE} - \hat{u}_i. \quad (21.14)$$

Cross-Section Weighting

Cross-section weighted regression is appropriate when the residuals are cross-section heteroskedastic and contemporaneously uncorrelated:

$$\Omega = E(\epsilon\epsilon') = E \begin{pmatrix} \sigma_1^2 I_{T_1} & 0 & \dots & 0 \\ 0 & \sigma_2^2 I_{T_2} & & \vdots \\ & & \ddots & \\ 0 & \dots & & \sigma_N^2 I_{T_N} \end{pmatrix} \quad (21.15)$$

EViews performs FGLS with $\hat{\sigma}_i^2$ estimated from a first-stage pooled OLS regression. The estimated variances are computed as:

$$\hat{\sigma}_i^2 = \frac{T_i}{\sum_{t=1}^{T_i} (y_{it} - \hat{y}_{it})^2} / T_i, \quad (21.16)$$

where \hat{y}_{it} are the OLS fitted values.

The estimated coefficients values and covariance matrix are given by the standard GLS estimator.

SUR Weighting

SUR weighted least squares (sometimes referred to as the Parks estimator) is the feasible GLS estimator when the residuals are both cross-section heteroskedastic and contemporaneously correlated (to simplify notation, we assume that $T_i = T$ for all i):

$$\Omega = E(\epsilon\epsilon)' = \begin{pmatrix} \sigma_{11}I_T & \sigma_{12}I_T & \dots & \sigma_{1N}I_T \\ \sigma_{21}I_T & \sigma_{22}I_T & & \vdots \\ & & \ddots & \\ \sigma_{N1}I_T & \dots & & \sigma_{NN}I_T \end{pmatrix} = \Sigma \otimes I_T, \quad (21.17)$$

where Σ is the symmetric matrix of contemporaneous correlations:

$$\Sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \dots & \sigma_{1N} \\ \sigma_{21} & \sigma_{22} & & \\ & & \ddots & \\ \sigma_{N1} & \dots & \dots & \sigma_{NN} \end{pmatrix}, \quad (21.18)$$

with typical element $\sigma_{ij} = E(\epsilon_{jt}\epsilon_{it})$, which is assumed constant across t .

EViews estimates the SUR model using $\hat{\sigma}_{ij}$ estimated from a first-stage pooled OLS regression:

$$\hat{\sigma}_{ij} = \sum_t (y_{it} - \hat{y}_{it})^2 / \max(T_i, T_j). \quad (21.19)$$

The use of the max function in the denominator handles the case of unbalanced data by down-weighting the covariance terms. Provided that the number of missing values is asymptotically negligible, this approach yields a consistent estimator of Σ that is generally invertible.

The parameter estimates and the covariance matrix of the parameters of the model are computed using the standard GLS formulae.

White Covariance Estimation

White's heteroskedasticity consistent covariance estimates may be computed for pooled specifications (except for SUR and random effects estimation). In computing the estimates, EViews calculates the White covariance matrix using the stacked model:

$$\text{var}(b) = \frac{(\sum N_i)}{(\sum N_i) - K} (X'X)^{-1} \left(\sum_{i,t} u_{it}^2 (x_{it}x_{it}') \right) (X'X)^{-1} \quad (21.20)$$

where K is the total number of estimated parameters. This variance estimator is robust to heteroskedasticity within each cross-section, but does not account for the possibility of contemporaneous correlation across cross-sections.

Chapter 22. State Space Models and the Kalman Filter

The EViews `sspace` object provides a straightforward, easy-to-use interface for specifying, estimating, and working with the results of your single or multiple equation dynamic system. EViews provides a wide range of specification, filtering, smoothing, and other forecasting tools which aid you in working with dynamic systems specified in state space form.

A wide range of time series models, including the classical linear regression model and ARIMA models, can be written and estimated as special cases of a state space specification. State space models have been applied in the econometrics literature to model unobserved variables: (rational) expectations, measurement errors, missing observations, permanent income, unobserved components (cycles and trends), and the non-accelerating rate of unemployment. Extensive surveys of applications of state space models in econometrics can be found in Hamilton (1994a, chapter 13; 1994b) and Harvey (1989, chapters 3, 4).

There are two main benefits to representing a dynamic system in state space form. First, the state space allows unobserved variables (known as the state variables) to be incorporated into, and estimated along with, the observable model. Second, state space models can be analyzed using a powerful recursive algorithm known as the Kalman (Bucy) filter. The Kalman filter algorithm has been used, among other things, to compute exact, finite sample forecasts for Gaussian ARMA models, multivariate (vector) ARMA models, MIMIC (multiple indicators and multiple causes), Markov switching models, and time varying (random) coefficient models.

Those of you who have used previous versions of the `sspace` object will note that much has changed with this release. We strongly recommend that you read [“Converting from Version 3 Sspace” on page 599](#) before loading existing workfiles and before beginning to work with the new state space routines.

Background

We present here a very brief discussion of the specification and estimation of a linear state space model. Those desiring greater detail are directed to Harvey (1989), Hamilton (1994a, Chapter 13, 1994b), and especially the excellent treatment of Koopman, Shephard and Doornik (1999).

Specification

A linear state space representation of the dynamics of the $n \times 1$ vector y_t is given by the system of equations:

$$y_t = c_t + Z_t \alpha_t + \epsilon_t \quad (22.1)$$

$$\alpha_{t+1} = d_t + T_t \alpha_t + v_t \quad (22.2)$$

where α_t is an $m \times 1$ vector of possibly unobserved state variables and where c_t , Z_t , d_t and T_t are conformable vectors and matrices, and ϵ_t and v_t are vectors of mean zero, Gaussian disturbances. Note that the unobserved state vector is assumed to move over time as a first-order vector autoregression.

We will refer to the first set of equations as the “signal” or “observation” equations and the second set as the “state” or “transition” equations. The disturbance vectors, ϵ_t and v_t are assumed to be serially independent, with contemporaneous variance structure:

$$\Omega_t = \text{var} \begin{bmatrix} \epsilon_t \\ v_t \end{bmatrix} = \begin{bmatrix} H_t & G_t \\ G_t' & Q_t \end{bmatrix} \quad (22.3)$$

where H_t is an $n \times n$ symmetric variance matrix, Q_t is an $m \times m$ symmetric variance matrix, and G_t is an $n \times m$ matrix of covariances.

In the discussion that follows, we will generalize the specification given in (22.1)–(22.3) by allowing the system matrices and vectors $\Xi_t \equiv \{c_t, d_t, Z_t, T_t, H_t, Q_t, G_t\}$ to depend upon observable explanatory variables X_t and unobservable parameters θ . Estimation of the parameters θ is discussed in “[Estimation](#)” [beginning on page 581](#).

Filtering

Consider the conditional distribution of the state vector α_t given information available at time s . We can define the mean and variance matrix of the conditional distribution as:

$$a_{t|s} \equiv E_s(\alpha_t) \quad (22.4)$$

$$P_{t|s} \equiv E_s[(\alpha_t - a_{t|s})(\alpha_t - a_{t|s})'] \quad (22.5)$$

where the subscript below the expectation operator indicates that expectations are taken using the conditional distribution for that period.

One important conditional distribution is obtained by setting $s = t - 1$, so that we obtain the *one-step ahead mean* $a_{t|t-1}$ and *one-step ahead variance* $P_{t|t-1}$ of the states α_t . Under the Gaussian error assumption, $a_{t|t-1}$ is also the minimum mean square error estimator of α_t and $P_{t|t-1}$ is the mean square error (MSE) of $a_{t|t-1}$. If the normality assumption is dropped, $a_{t|t-1}$ is still the minimum mean square *linear* estimator of α_t .

Given the one-step ahead state conditional mean, we can also form the (linear) minimum MSE *one-step ahead estimate of* y_t

$$\tilde{y}_t = y_{t|t-1} \equiv E_{t-1}(y_t) = E(y_t | a_{t|t-1}) = c_t + Z_t a_{t|t-1} \quad (22.6)$$

The *one-step ahead prediction error* is given by

$$\tilde{\epsilon}_t = \epsilon_{t|t-1} \equiv y_t - \tilde{y}_{t|t-1} \quad (22.7)$$

and the *prediction error variance* is defined as

$$\tilde{F}_t = F_{t|t-1} \equiv \text{var}(\epsilon_{t|t-1}) = Z_t P_{t|t-1} Z_t' + H_t \quad (22.8)$$

The Kalman (Bucy) filter is a recursive algorithm for sequentially updating the one-step ahead estimate of the state mean and variance given new information. Details on the recursion are provided in the references above. For our purposes, it is sufficient to note that given initial values for the state mean and covariance, values for the system matrices Ξ_t , and observations on y_t the Kalman filter may be used to compute estimates of the one-step ahead estimates of the state and the mean square error matrix of the estimates, $\{a_{t|t-1}, P_{t|t-1}\}$, the contemporaneous or *filtered* state mean and variance, $\{a_t, P_t\}$, and the one-step ahead prediction, prediction error, and prediction error variance: $\{y_{t|t-1}, \epsilon_{t|t-1}, F_{t|t-1}\}$. Note that we may also obtain the standardized prediction residual $e_{t|t-1}$ by dividing $\epsilon_{t|t-1}$ by the square-root of the corresponding diagonal element of $F_{t|t-1}$.

Fixed-Interval Smoothing

Suppose that we observe the sequence of data up to time period T . The process of using all this information to form expectations at any time period up to T is known as *fixed-interval smoothing*. Despite the fact that there a variety of other distinct forms of smoothing (e.g. - fixed-point, fixed-lag), we will use the term *smoothing* to refer to fixed-interval smoothing.

Additional details on the smoothing procedure are provided in the references given above. For now, note that smoothing uses all of the information in the sample to provide *smoothed estimates of the states* $\hat{\alpha}_t \equiv \alpha_{t|T} \equiv E_T(\alpha_t)$, and *smoothed estimates of the state variances*, $V_t \equiv \text{var}_T(\alpha_t)$. The matrix V_t may also be interpreted as the MSE of the smoothed state estimate $\hat{\alpha}_t$.

As with the one-step ahead states and variances above, we may use the smoothed values to form *smoothed estimates of the signal variables*,

$$\hat{y}_t \equiv E(y_t | \hat{\alpha}_t) = c_t + Z_t \hat{\alpha}_t \quad (22.9)$$

and to compute the *variance of the smoothed signal estimates*

$$S_t \equiv \text{var}(\hat{y}_{t|T}) = Z_t V_t Z_t' \quad (22.10)$$

Lastly, the smoothing procedure allows us to compute *smoothed disturbance estimates*, $\hat{\epsilon}_t \equiv \epsilon_{t|T} \equiv E_T(\epsilon_t)$ and $\hat{v}_t \equiv v_{t|T} \equiv E_T(v_t)$, and a corresponding *smoothed disturbance variance matrix*,

$$\hat{\Omega}_t = \text{var}_T \left(\begin{bmatrix} \epsilon_t \\ v_t \end{bmatrix} \right) \quad (22.11)$$

Dividing the smoothed disturbance estimates by the square roots of the corresponding diagonal elements of the smoothed variance matrix yields the *standardized smoothed disturbance estimates* \hat{e}_t and \hat{v}_t .

Forecasting

There are a variety of types of forecasting which may be performed with state space models. These methods differ primarily in what and how information is used. We will focus on the three methods that are supported by EViews built-in forecasting routines.

n-Step Ahead Forecasting

Earlier, we examined the notion of one-step ahead prediction. Consider now the notion of multi-step ahead prediction of observations, in which we take a fixed set of information available at a given period, and forecast several periods ahead. Modifying slightly the expressions in (22.4)—(22.8) yields the *n-step ahead state conditional mean and variance*

$$a_{t+n|t} \equiv E_t(\alpha_{t+n}), \quad (22.12)$$

$$P_{t+n|t} \equiv E_t[(\alpha_{t+n} - a_{t+n|t})(\alpha_{t+n} - a_{t+n|t})'] \quad (22.13)$$

the *n-step ahead forecast*

$$y_{t+n|t} \equiv E_t(y_{t+n}) = c_t + Z_t a_{t+n|t} \quad (22.14)$$

and the corresponding *n-step ahead forecast MSE matrix*

$$F_{t+n|t} \equiv \text{MSE}(\tilde{y}_{t+n|t}) = Z_{t+n} P_{t+n|t} Z_{t+n}' + H_t \quad (22.15)$$

for $n = 1, 2, \dots$. As before, $a_{t+n|t}$ may also be interpreted as the minimum MSE estimate of α_{t+n} based on the information set available at time t , and $P_{t+n|t}$ is the MSE of the estimate.

It is worth emphasizing that the definitions given above for the forecast MSE matrices $F_{t+n|t}$ do not account for extra variability introduced in the estimation of any unknown parameters θ . In this setting, the $F_{t+n|t}$ will understate the true variability of the forecast, and should be viewed as being computed conditional on the specific value of the estimated parameters.

It is also worth noting that the *n-step ahead forecasts* may be computed using a slightly modified version of the basic Kalman recursion (Harvey 1989). To forecast at period $s = t + n$, simply initialize a Kalman filter at time $t + 1$ with the values of the predicted states and state covariances using information at time t , and run the filter forward

$n - 1$ additional periods using no additional signal information. This procedure is repeated for each observation in the forecast sample, $s = t + 1, \dots, t + n^*$.

Dynamic Forecasting

The concept of *dynamic forecasting* should be familiar to you from other EViews estimation objects. In dynamic forecasting, we start at the beginning of the forecast sample t , and compute a complete set of n -period ahead forecasts for each period $n = 1, \dots, n^*$ in the forecast interval. Thus, if we wish to start at period t and forecast dynamically to $t + n^*$, we would compute a one-step ahead forecast for $t + 1$, a two-step ahead forecasts for $t + 2$, and so forth, up to an n^* -step ahead forecast for $t + n^*$. It may be useful to note that as with n -step ahead forecasting, we simply initialize a Kalman filter at time $t + 1$ and run the filter forward additional periods using no additional signal information. For dynamic forecasting, however, only one n -step ahead forecast is required to compute all of the forecast values since the information set is not updated from the beginning of the forecast period.

Smoothed Forecasting

Alternatively, we can compute *smoothed forecasts* which use all available signal data over the forecast sample (for example, $a_{t+n|t+n^*}$). These forward looking forecasts may be computed by initializing the states at the start of the forecast period, and performing a Kalman smooth over the entire forecast period using all relevant signal data. This technique is useful in settings where information on the entire path of the signals is used to interpolate values throughout the forecast sample.

We make one final comment about the forecasting methods described above. For traditional n -step ahead and dynamic forecasting, the states are typically initialized using the one-step ahead forecasts of the states and variances at the start of the forecast window. For smoothed forecasts, one would generally initialize the forecasts using the corresponding smoothed values of states and variances. There may, however, be situations where you wish to choose a different set of initial values for the forecast filter or smoother. The EViews forecasting routines (described in [“State Space Procedures” beginning on page 595](#)) provide you with considerable control over these initial settings. Be aware, however, that the interpretation of the forecasts in terms of the available information will change if you choose alternative settings.

Estimation

To implement the Kalman filter and the fixed-interval smoother, we must first replace any unknown elements of the system matrices by their estimates. Under the assumption that the ϵ_t and v_t are Gaussian, the sample log likelihood,

$$\log L(\theta) = -\frac{nT}{2} \log 2\pi - \frac{1}{2} \sum_t \log |\tilde{F}_t(\theta)| - \frac{1}{2} \sum_t \tilde{\epsilon}_t'(\theta) \tilde{F}_t(\theta)^{-1} \tilde{\epsilon}_t(\theta) \quad (22.16)$$

may be evaluated using the Kalman filter. Using numeric derivatives, standard iterative techniques may be employed to maximize the likelihood with respect to the unknown parameters θ , see [Appendix D, “Estimation Algorithms and Options”, on page 663](#).

Initial Conditions

Evaluation of the Kalman filter, smoother, and forecasting procedures all require that we provide the initial one-step ahead predicted values for the states $\alpha_{1|0}$ and variance matrix $P_{1|0}$. With some stationary models, steady-state conditions allow us to use the system matrices to solve for the values of $\alpha_{1|0}$ and $P_{1|0}$. In other cases, we may have preliminary estimates of $\alpha_{1|0}$, along with measures of uncertainty about those estimates. But in many cases, we may have no information, or *diffuse priors*, about the initial conditions.

Specifying a State Space Model in EViews

EViews handles a wide range of single and multiple-equation state space models, providing you with detailed control over the specification of your system equations, covariance matrices, and initial conditions.

The first step in specifying and estimating a state space model is to create a state space object. Select **Objects/New Object/Sspace** from the main toolbar or type `sspace` in the command window. EViews will create a state space object and open an empty state space specification window.

There are two ways to specify your state space model. The easiest is to use EViews special “auto-specification” features to guide you in creating some of the standard forms for these models. Simply press the **AutoSpec** button on the `sspace` toolbar. Specialized dialogs will open to guide you through the specification process. We will describe this method in greater detail in [“Auto-Specification” on page 590](#).

The more general method of describing your state space model uses keywords and text to describe the signal equations, state equations, error structure, initial conditions, and if desired, parameter starting values for estimation. The next section describes the general syntax for the state space object.

Specification Syntax

State Equations

A state equation contains the “@STATE” keyword followed by a valid state equation specification. Bear in mind that:

- Each equation must have a unique dependent variable name; expressions are not allowed. Since EViews does not automatically create workfile series for the states, you may use the name of an existing (non-series) EViews object.

- State equations may not contain signal equation dependent variables, or leads or lags of these variables.
- Each state equation must be linear in the one-period lag of the states. Nonlinearities in the states, or the presence of contemporaneous, lead, or multi-period lag states will generate an error message. We emphasize the point that the one-period lag restriction on states *is not restrictive* since higher order lags may be written as new state variables. An example of this technique is provided in the example “[ARMAX\(2, 3\) with a Random Coefficient](#)” on page 586.
- State equations may contain exogenous variables and unknown coefficients, and may be nonlinear in these elements.

In addition, state equations may contain an optional error or error variance specification. If there is no error or error variance, the state equation is assumed to be deterministic. Specification of the error structure of state space models is described in greater detail below in “[Errors and Variances](#)” on page 584.

Examples

The following two state equations define an unobserved error with an AR(2) process:

```
@state sv1 = c(2)*sv1(-1) + c(3)*sv2(-1) + [var = exp(c(5))]
@state sv2 = sv1(-1)
```

The first equation parameterizes the AR(2) for SV1 in terms of an AR(1) coefficient, C(2), and an AR(2) coefficient, C(3). The error variance specification is given in square brackets. Note that the state equation for SV2 defines the lag of SV1 so that SV2(-1) is the two period lag of SV1.

Similarly, the following are valid state equations:

```
@state sv1 = sv1(-1) + [var = exp(c(3))]
@state sv2 = c(1) + c(2)*sv2(-1) + [var = exp(c(3))]
@state sv3 = c(1) + exp(c(3)*x/z) + c(2)*sv3(-1) + [var =
exp(c(3))]
```

describing a random walk, and an AR(1) with drift (without/with exogenous variables).

The following are *not* valid state equations:

```
@state exp(sv1) = sv1(-1) + [var = exp(c(3))]
@state sv2 = log(sv2(-1)) + [var = exp(c(3))]
@state sv3 = c(1) + c(2)*sv3(-2) + [var=exp(c(3))]
```

since they violate at least one of the conditions described above (in order: expression for dependent state variable, nonlinear in state, multi-period lag of state variables).

Observation/Signal Equations

By default, if an equation specification is not specifically identified as a state equation using the “@STATE” keyword, it will be treated by EViews as an observation or signal equation. Signal equations may also be identified explicitly by the keyword “@SIGNAL”. There are some aspects of signal equation specification to keep in mind:

- Signal equation dependent variables may involve expressions.
- Signal equations may not contain current values or leads of signal variables. You should be aware that any lagged signals are treated as predetermined for purposes of multi-step ahead forecasting (for discussion and alternative specifications, see Harvey 1989, pp. 367-368).
- Signal equations must be linear in the contemporaneous states. Nonlinearities in the states, or the presence of leads or lags of states will generate an error message. Again, the restriction that there are no state lags is not restrictive since additional deterministic states may be created to represent the lagged values of the states.
- Signal equations may have exogenous variables and unknown coefficients, and may be nonlinear in these elements.

Signal equations may also contain an optional error or error variance specification. If there is no error or error variance, the equation is assumed to be deterministic. Specification of the error structure of state space models is described in greater detail in “[Errors and Variances](#)” on page 584.

Examples

The following are valid signal equation specifications:

```
log(passenger) = c(1) + c(3)*x + sv1 + c(4)*sv2
@signal y = sv1 + sv2*x1 + sv3*x2 + sv4*y(-1) + [var=exp(c(1))]
Z = sv1 + sv2*x1 + sv3*x2 + c(1) + [var=exp(c(2))]
```

The following are invalid equations:

```
log(passenger) = c(1) + c(3)*x + sv1(-1)
@signal y = sv1*sv2*x1 + [var = exp(c(1))]
z = sv1 + sv2*x1 + z(1) + c(1) + [var = exp(c(2))]
```

since they violate at least one of the conditions described above (in order: lag of state variable, nonlinear in a state variable, lead of signal variable).

Errors and Variances

While EViews always adds an implicit error term to each equation in an equation or system object, the handling of error terms differs in a sspace object. In a sspace object, the

equation specifications in a signal or state equation do not contain error terms unless specified explicitly.

The easiest way to add an error to a state space equation is to specify an implied error term using its variance. You can simply add an error variance expression, consisting of the keyword “VAR” followed by an assignment statement (all enclosed in square brackets), to the existing equation:

```
@signal y = c(1) + sv1 + sv2 + [var = 1]
@state sv1 = sv1(-1) + [var = exp(c(2))]
@state sv2 = c(3) + c(4)*sv2(-1) + [var = exp(c(2)*x)]
```

The specified variance may be a known constant value, or it can be an expression containing unknown parameters to be estimated. You may also build time-variation into the variances using a series expression. Variance expressions may not, however, contain state or signal variables.

While straightforward, this direct variance specification method does not admit correlation between errors in different equations (by default, EViews assumes that the covariance between error terms is 0). If you require a more flexible variance structure, you will need to use the “named error” approach to define named errors with variances and covariances, and then to use these named errors as parts of expressions in the signal and state equations.

The first step of this general approach is to define your named errors. You may declare a named error by including a line with the keyword “@ENAME” followed by the name of the error:

```
@ename e1
@ename e2
```

Once declared, a named error may enter linearly into state and signal equations. In this manner, one can build correlation between the equation errors. For example, the errors in the state and signal equations in

```
y = c(1) + sv1*x1 + e1
@state sv1 = sv1(-1) + e2 + c(2)*e1
@ename e1
@ename e2
```

are, in general, correlated since the named error E1 appears in both equations.

In the special case where a named error is the only error in a given equation, you can both declare and use the named residual by adding an error expression consisting of keyword “ENAME” followed by an assignment and a name identifier.

```
y = c(1) + sv1*x1 + [ename = e1]
@state sv1 = sv1(-1) + [ename = e2]
```

The final step in building a general error structure is to define the variances and covariances associated with your named errors. You should include a `sspace` line comprised of the keyword “`@EVAR`” followed by an assignment statement for the variance of the error or the covariance between two errors:

```
@evar cov(e1, e2) = c(2)
@evar var(e1) = exp(c(3))
@evar var(e2) = exp(c(4))*x
```

The syntax for the `@EVAR` assignment statements should be self-explanatory. Simply indicate whether the term is a variance or covariance, identify the error(s), and enter the specification for the variance or covariance. There should be a separate line for each named error covariance or variance that you wish to specify. If an error term is named, but there are no corresponding “`VAR=`” or `@EVAR` specifications, the missing variance or covariance specifications will remain at the default values of “`NA`” and “`0`”, respectively.

As you might expect, in the special case where an equation contains a single error term, you may combine the named error and direct variance assignment statements:

```
@state sv1 = sv1(-1) + [ename = e1, var = exp(c(3))]
@state sv2 = sv2(-1) + [ename = e2, var = exp(c(4))]
@evar cov(e1, e2) = c(5)
```

Specification Examples

ARMAX(2, 3) with a Random Coefficient

We can use the syntax described above to define an ARMAX(2,3) with a random coefficient for the regression variable X :

```
y = c(1) + sv5*x + sv1 + c(4)*sv2 + c(5)*sv3 + c(6)*sv4
@state sv1 = c(2)*sv1(-1) + c(3)*sv2(-1) + [var=exp(c(7))]
@state sv2 = sv1(-1)
@state sv3 = sv2(-1)
@state sv4 = sv3(-1)
@state sv5 = sv5(-1) + [var=3]
```

The AR coefficients are parameterized in terms of $C(2)$ and $C(3)$, while the MA coefficients are given by $C(4)$, $C(5)$ and $C(6)$. The variance of the innovation is restricted to be a positive function of $C(7)$. $SV5$ is the random coefficient on X , with variance restricted to be 3.

Recursive and Random Coefficients

The following example describes a model with one random coefficient (SV1), one recursive coefficient (SV2), and possible correlation between the errors for SV1 and Y:

```
y = c(1) + sv1*x1 + sv2*x2 + [ename = e1, var = exp(c(2))]
@state sv1 = sv1(-1) + [ename = e2, var = exp(c(3)*x)]
@state sv2 = sv2(-1)
@evar cov(e1,e2) = c(4)
```

The variances and covariances in the model are parameterized in terms of the coefficients C(2), C(3) and C(4), with the variances of the observed Y and the unobserved state SV1 restricted to be non-negative functions of the parameters.

Parameter Starting Values

Unless otherwise instructed, EViews will initialize all parameters to the current values in the corresponding coefficient vector or vectors. As in the system object, you may override this default behavior by specifying explicitly the desired values of the parameters using a PARAM or @PARAM statement. For additional details, see [“Starting Values” on page 503](#).

Specifying Initial Conditions

By default, EViews will handle the initial conditions for you. For some stationary models, steady-state conditions allow us to solve for the values of α_0 and P_0 . For cases where it is not possible to solve for the initial conditions, EViews will treat the initial values as diffuse, setting $\alpha_{1|0} = 0$, and $P_{1|0}$ to an arbitrarily high number to reflect our uncertainty about the values (see [“Technical Discussion” on page 600](#)).

You may, however have prior information about the values of $\alpha_{1|0}$ and $P_{1|0}$. In this case, you can create a vector or matrix that contains the appropriate values, and use the “@MPRIOR” or “@VPRIOR” keywords to perform the assignment.

To set the initial states, enter “@MPRIOR” followed by the name of a vector object. The length of the vector object must match the state dimension. The order of elements should follow the order in which the states were introduced in the specification screen.

```
@mprior v1
@vprior m1
```

To set the initial state variance matrix, enter “@VPRIOR” followed by the name of a sym object (note that it must be a sym object, and not an ordinary matrix object). The dimensions of the sym must match the state dimension, with the ordering following the order in which the states appear in the specification. If you wish to set a specific element to be diffuse, simply assign the element the “NA” missing value. EViews will reset all of the corresponding variances and covariances to be diffuse.

For example, suppose you have a two equation state space object named SS1 and you want to set the initial values of the state vector and the state variance matrix as

$$\begin{bmatrix} SV1 \\ SV2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \text{var} \begin{bmatrix} SV1 \\ SV2 \end{bmatrix} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix} \quad (22.17)$$

First create a named vector object, say SVEC0, to hold the initial values. Click **Objects/ New Object**, choose **Matrix-Vector-Coeff** and enter the name SVEC0. Click **OK**, and then choose the type **Vector** and specify the size of the vector (in this case 2 rows). When you click **OK**, EViews will display the spreadsheet view of the vector SVEC0. Click the **Edit + / -** button to toggle edit mode and type in the desired values. Then create a named matrix object, say SVAR0, in an analogous fashion.

Alternatively, you may find it easier to create and initialize the vector and matrix using commands. You can enter the following commands in the command window:

```
vector(2) svec0
svec0.fill 1, 0
matrix(2,2) svar0
svar0.fill(b=c) 1, 0.5, 0.5, 2
```

Then, simply add the lines

```
@mprior svec0
@vprior svar0
```

to your sspace object by editing the specification window. Alternatively, you can type the following commands in the command window:

```
ss1.append @mprior svec0
ss1.append @vprior svar0
```

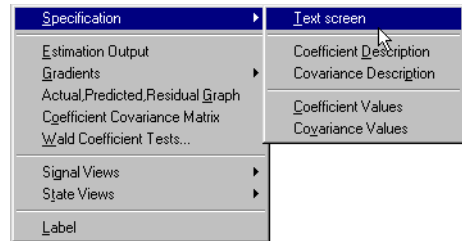
For more details on matrix objects and the `fill` and `append` commands, see [Chapter 4, “Matrix Language”](#), on page 55 of the *Command and Programming Reference*.

Specification Views

State space models may be very complex. To aid you in examining your specification, EViews provides views which allow you to view the text specification in a more compact form, and to examine the numerical values of your system matrices evaluated at current parameter values.

Click on the **View** menu and select **Specification...** The following Specification views are always available, regardless of whether the sspace has previously been estimated:

- Text Screen.** This is the familiar text view of the specification. You should use this view when you create or edit the state space specification. This view may also be accessed by clicking on the **Spec** button on the sspace toolbar.



- Coefficient Description.** Text description of the structure of your state space specification. The variables on the left-hand side, representing α_{t+1} and y_t , are expressed as linear functions of the state variables α_t , and a remainder term CONST. The elements of the matrix are the corresponding coefficients. For example, the ARMAX example has the following Coefficient Description view:

The screenshot shows the 'Structural Coefficient Description' window for the SS_ARMA23 model. The window title is 'Sspace: SS_ARMA23 Workfile: AIRLINE'. The menu bar includes View, Procs, Objects, Print, Name, Freeze, Estimate, Stats, Spec, AutoSpec, and MergeText. The table below shows the coefficients for the state variables and the output variable Y.

	CONST	SV1	SV2	SV3	SV4	SV5
SV1(1)	0	C(2)	C(3)	0	0	0
SV2(1)	0	1	0	0	0	0
SV3(1)	0	0	1	0	0	0
SV4(1)	0	0	0	1	0	0
SV5(1)	0	0	0	0	0	1
Y	C(1)	1	C(4)	C(5)	C(6)	X

- Covariance Description.** Text description of the covariance matrix of the state space specification. For example, the ARMAX example has the following Covariance Description view:

The screenshot shows the 'Structural Covariance Description' window for the SS_ARMA23 model. The window title is 'Sspace: SS_ARMA23 Workfile: AIRLINE'. The menu bar includes View, Procs, Objects, Print, Name, Freeze, Estimate, Stats, Spec, AutoSpec, and MergeText. The table below shows the covariance matrix for the state variables and the output variable Y.

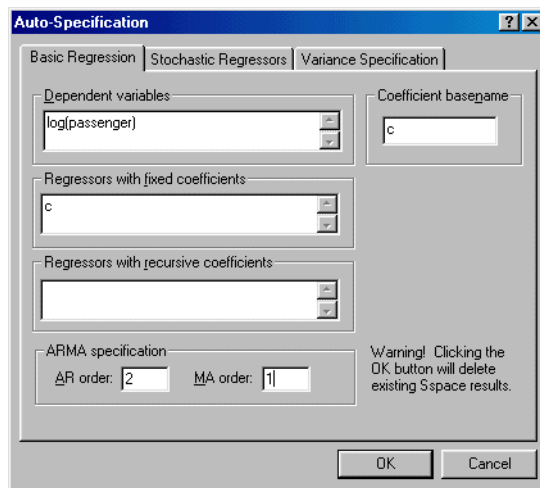
	SV1	SV2	SV3	SV4	SV5	Y
SV1	EXP(C(7))	0	0	0	0	0
SV2	0	0	0	0	0	0
SV3	0	0	0	0	0	0
SV4	0	0	0	0	0	0
SV5	0	0	0	0	EXP(C(8))	0
Y	0	0	0	0	0	0

- **Coefficient Values.** Numeric description of the structure of the signal and the state equations evaluated at current parameter values. If the system coefficient matrix is time-varying, EViews will prompt you for a date/observation at which to evaluate the matrix.
- **Covariance Values.** Numeric description of the structure of the state space specification evaluated at current parameter values. If the system covariance matrix is time-varying, EViews will prompt you for a date/observation at which to evaluate the matrix.

Auto-Specification

To aid you in creating a state space specification, EViews provides you with “auto-specification” tools which will create the text representation of a model that you specify from dialogs. This tool may be very useful if your model is a standard regression with fixed, recursive, and various random coefficient specifications, and/or your errors have a general ARMA structure.

Click on the **AutoSpec** button on the SSPACE toolbar, or select **Procs/Define State Space...** from the menu. EViews opens a three tab dialog. The first tab is used to describe the basic regression portion of your specification. Enter the dependent variable, and any regressors which have fixed or recursive coefficients. You can choose which COEF object EViews uses for indicating unknowns when setting up the specification. At the bottom, you can specify an ARMA structure for your errors. Here, we have specified a simple ARMA(2,1) from the example above.



The second tab of the dialog is used to add any regressors which have random coefficients. Simply enter the appropriate regressors in each of the four edit fields. EViews allows you to define regressors with any combination of constant mean, AR(1), random walk, or random walk (with drift) coefficients.

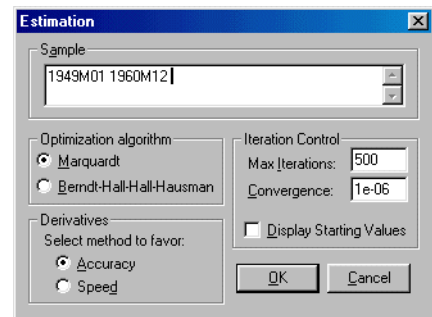
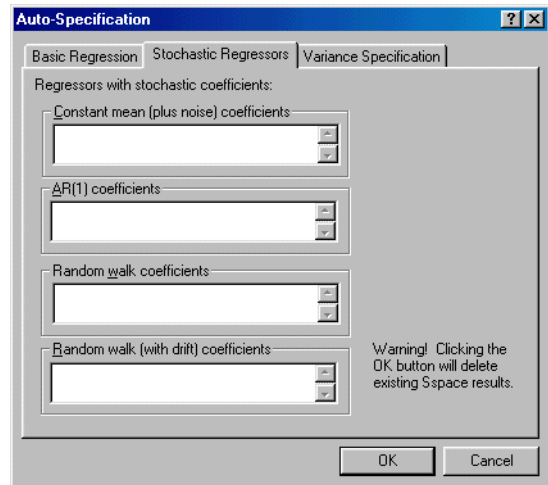
Lastly, the Auto-Specification dialog allows you to choose between basic variance structures for your state space model. Click on the **Variance Specification** tab, and choose between an identity matrix, common diagonal (diagonal with common variances), diagonal, or general (unrestricted) variance matrix for the signals and for the states. The dialog also allows you to allow the signal equation(s) and state equations(s) to have non-zero error covariances.

We emphasize the fact that your sspace object is not restricted to the choices provided in this dialog. If you find that the Auto-Specification is too restrictive, you can simply use it as a tool to build a basic specification, and then use the more general text tools to describe your model.

Estimating a State Space Model

Once you have specified a state space model and verified that your specification is correct, you are ready to estimate the model. To open the estimation dialog, simply click on the **Estimate** button on the toolbar or select **Procs/Estimate...**

As with other estimation objects, EViews allows you to set the estimation sample, the maximum number of iterations, convergence tolerance, the estimation algorithm, derivative settings and whether to display the starting values. The default settings should provide a good start for most problems, but if you choose to change the settings, be sure to read “[Setting Estimation Options](#)” on page 666, which provides a detailed discussion of these options. When you click on OK, EViews will begin estimation using the specified settings.



There are two additional things to keep in mind when estimating your model:

- Although the EViews Kalman filter routines will automatically handle any missing values in your sample, EViews does require that your estimation sample be contiguous, with no gaps between successive observations.
- If there are no unknown coefficients in your specification, you will still have to “estimate” your sspace to run the Kalman filter and initialize elements that EViews needs in order to perform further analysis.

Interpreting the estimation results

After you choose the variance options and click **OK**, EViews presents the estimation results in the state space window. For example, if we specify an ARMA(2,1) for the log of the monthly international airline passenger totals from January 1949 to December 1960 (from Box and Jenkins, 1976, series G, p. 531):

```
log(passenger) = c(1) + sv1 + c(4)*sv2
@state sv1 = c(2)*sv1(-1) + c(3)*sv2(-1) + [var=exp(c(5))]
@state sv2 = sv1(-1)
```

and estimate the model, EViews will open the estimation output view.

```
Sspace: SS_ARMA21
Estimation Method: Maximum Likelihood (Marquardt)
Date: 11/12/99 Time: 11:58
Sample: 1949M01 1960M12
Included observations: 144
Convergence achieved after 55 iterations
```

	Coefficient	Std. Error	z-Statistic	Prob.
C(1)	5.499767	0.257517	21.35687	0.0000
C(2)	0.409013	0.167201	2.446239	0.0144
C(3)	0.547165	0.164608	3.324055	0.0009
C(4)	0.841481	0.100167	8.400800	0.0000
C(5)	-4.589401	0.172696	-26.57501	0.0000

	Final State	Root MSE	z-Statistic	Prob.
SV1	0.267125	0.100792	2.650274	0.0080
SV2	0.425488	0.000000	NA	1.0000

Log likelihood	124.3366	Parameters	5
Akaike info criterion	-1.629674	Likelihood observations	144
Schwarz criterion	-1.485308	Missing observations	0
Hannan-Quinn criter.	-1.571012	Partial observations	0
		Diffuse priors	0

The bulk of the output view should be familiar from other EViews estimation objects. The information at the top describes the basics of the estimation: the name of the sspace object, estimation method, the date and time of estimation, sample, and number of objects in the sample, convergence information, and the coefficient estimates. The bottom part of

the view reports the maximized log likelihood value, the number of estimated parameters, and the associated information criteria.

Some parts of the output, however, are new and may require discussion. The bottom section provides additional information about the handling of missing values in estimation. “Likelihood observations” reports the actual number of observations that are used in forming the likelihood. This number (which is the one used in computing the information criteria) will differ from the “Included observations” reported at the top of the view when EViews drops an observation from the likelihood calculation because all of the signal equations have missing values. The number of omitted observations is reported in “Missing observations”. “Partial observations” reports the number of observations that are included in the likelihood, but for which some equations have been dropped. “Diffuse priors” indicates the number of initial state covariances for which EViews is unable to solve and for which there is no user initialization. EViews’ handling of initial states and covariances is described in greater detail in [“Initial Conditions” on page 600](#).

EViews also displays the final one-step ahead values of the state vector, $\alpha_{T+1|T}$ and the corresponding RMSE values (square roots of the diagonal elements of $P_{T+1|T}$). For settings where you may care about the entire path of the state vector and covariance matrix, EViews provides you with a variety of views and procedures for examining the state results in greater detail.

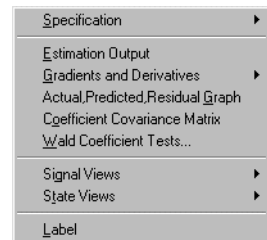
Working with the State Space

EViews provides a variety of specialized tools for specifying and examining your state space specification. As with other estimation objects, the sspace object provides additional views and procedures for examining the estimation results, performing inference and specification testing, and extracting results into other EViews objects.

State Space Views

Many of the state space views should be familiar from previous discussion:

- We have already discussed the **Specification...** views in our analysis of [“Specification Views” on page 588](#) above.
- The **Estimation Output** view displays the coefficient estimates and summary statistics as described above in [“Interpreting the estimation results” on page 592](#). You may also access this view by pressing **Stats** on the system toolbar.
- The **Gradients and Derivatives...** view should be familiar from other estimation objects. If the sspace contains parameters to be estimated, this view provides sum-



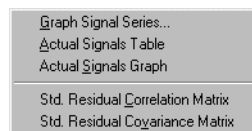
mary and visual information about the gradients of the log likelihood at estimated parameters (if the sspace is estimated) or at current parameter vales.

- **Actual, Predicted, Residual Graph** displays, in graphical form, the actual and one-step ahead fitted values of the signal dependent variable(s), $y_{t|t-1}$, and the one-step ahead standardized residuals, $e_{t|t-1}$.
- Select **Coefficient Covariance Matrix** to view the estimated coefficient covariance.
- **Wald Coefficient Tests...** allows you to perform hypothesis tests on the estimated coefficients. For details, see “[Wald Test \(Coefficient Restrictions\)](#)” on page 368.
- **Label** allows you to annotate your object. See “[Labeling Objects](#)” on page 50.

Note that with the exception of the **Label** and **Specification...** views, these views are available only following successful estimation of your state space model.

Signal Views

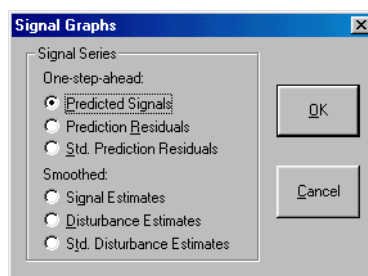
When you click on **View/Signal Views**, EViews displays a sub-menu containing additional view selections. Two of these selections are always available, even if the state space model has not yet been estimated:



- **Actual Signal Table** and **Actual Signal Graph** display the dependent signal variables in spreadsheet and graphical forms. If there are multiple signal equations, EViews will display a each series with its own axes.

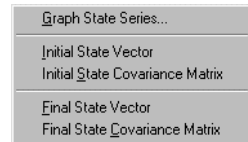
The remaining views are only available following estimation:

- **Graph Signal Series...** opens a dialog with choices for the results to be displayed. The dialog allows you to choose between the one-step ahead predicted signals, $y_{t|t-1}$, the corresponding one-step residuals, $e_{t|t-1}$ or standardized one-step residuals, $e_{t|t-1}$, the smoothed signals, \hat{y}_t , smoothed signal disturbances, \hat{e}_t , or the standardized smoothed signal disturbances, $\hat{e}_t \cdot \pm 2$ (root mean square) standard error bands are plotted where appropriate.
- **Std. Residual Correlation Matrix** and **Std. Residual Covariance Matrix** display the correlation and covariance matrix of the standardized one-step ahead signal residual, $e_{t|t-1}$.



State Views

To examine the unobserved state components, click on **View/State Views** to display the state submenu. EViews allows you to examine the initial or final values of the state components, or to graph the full time-path of various filtered or smoothed state data.



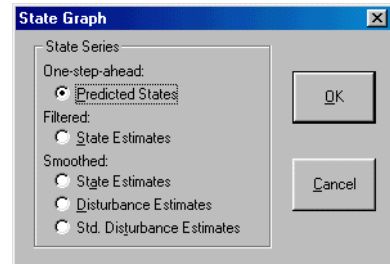
Two of the views are available either before or after estimation:

- **Initial State Vector** and **Initial State Covariance Matrix** display the values of the initial state vector, α_0 , and covariance matrix, P_0 . If the unknown parameters have previously been estimated, EViews will evaluate the initial conditions using the estimated values. If the sspace has not been estimated, the current coefficient values will be used in evaluating the initial conditions.

This information is especially relevant in models where EViews is using the current values of the system matrices to solve for the initial conditions. In cases where you are having difficulty starting your estimation, you may wish to examine the values of the initial conditions at the starting parameter values for any sign of problems.

The remainder of the views are only available following successful estimation:

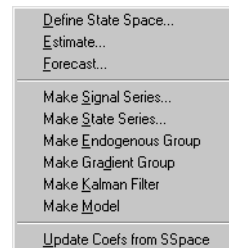
- **Final State Vector** and **Final State Covariance Matrix** display the values of the final state vector, α_T , and covariance matrix, P_T , evaluated at the estimated parameters.
- Select **Graph State Series...** to display a dialog containing several choices for the state information. You can graph the one-step ahead predicted states, $a_{t|t-1}$, the filtered (contemporaneous) states, a_t , the smoothed state estimates, $\hat{\alpha}_t$, smoothed state disturbance estimates, \hat{v}_t , or the standardized smoothed state disturbances, $\hat{\eta}_t$. In each case, the data are displayed along with corresponding ± 2 standard error bands.



State Space Procedures

You can use the EViews procedures to create, estimate, forecast, and generate data from your state space specification.

- **Define State Space...** calls up the Auto-Specification dialog (see “[Auto-Specification](#)” on page 590). This feature provides a method of specifying a variety of common state space specifications using interactive menus.
- Select **Estimate...** to estimate the parameters of the specification (see “[Estimating a State Space Model](#)” on page 591).



These above items are available both before and after estimation.

The Auto-Specification tool will replace both the existing state space specification and clear any results. Estimation will replace existing results.

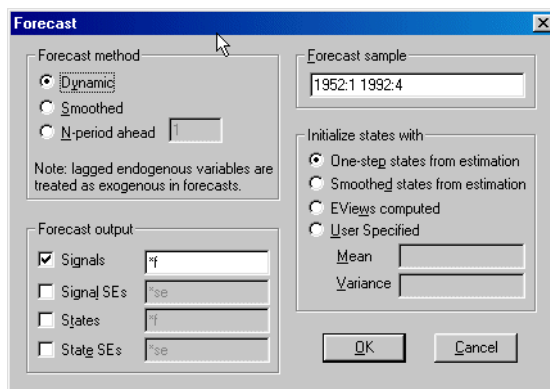
Once you have estimated your sspace, EViews provides additional tools for generating data:

- The **Forecast...** dialog allows you to generate forecasts of the states, signals, and the associated standard errors using alternative methods and initialization approaches.

First, select the forecast method. You can select between dynamic, smoothed, and n -period ahead forecasting, as described in “[Forecasting](#)” on page 580. Note that any lagged endogenous variables on the right-hand side of your signal equations will be treated as predetermined for purposes of forecasting.

EViews allows you to save various types of forecast output in series in your workfile. Simply check any of the output boxes, and specify the names for the series in the corresponding edit field.

You may specify the names either as a list, or using a wildcard expression. If you choose to list the names, the number of identifiers must



match the number of signals in your specification. You should be aware that if an output series with a specified name already exists in the workfile, EViews will overwrite the entire contents of the series.

If you use a wildcard expression, EViews will substitute the name of each signal in the appropriate position in the wildcard expression. For example, if you have a model with signals Y1 and Y2, and elect to save the one-step predictions in “`PRED*`”, EViews will use the series `PREDY1` and `PREDY2` for output. There are two limitations to this feature: (1) you may not use the wildcard expression “`*`” to save

signal results since this will overwrite the original signal data, and (2) you may not use a wildcard when any signal dependent variables are specified by expression, or when there are multiple equations for a signal variable. In both cases, EViews will be unable to create the new series and will generate an error message.

Keep in mind that if your signal dependent variable is an expression, EViews will only provide forecasts of the expression. Thus, if your signal variable is $\text{LOG}(Y)$, EViews will forecast the logarithm of Y .

Now enter a sample and specify the treatment of the initial states, and then click **OK**. EViews will compute the forecast and will place the results in the specified series. No output window will open.

There are several options available for setting the initial conditions. If you wish, you can instruct the `sspace` object to use the **One-step ahead** or **Smoothed** estimates of the state and state covariance as initial values for the forecast period. The two initialization methods differ in the amount of information used from the estimation sample; one-step ahead uses information up to the beginning of the forecast period, while smoothed uses the entire estimation period.

Alternatively, you may use **EViews computed** initial conditions. As in estimation, if possible, EViews will solve the Algebraic Riccati equations to obtain values for the initial state and state covariance at the start of each forecast interval. If solution of these conditions is not possible, EViews will use diffuse priors for the initial values.

Lastly, you may choose to provide a vector and sym object which contain the values for the forecast initialization. Simply select **User** and enter the name of valid EViews objects in the appropriate edit fields.

Note that when performing either dynamic or smoothed forecasting, EViews requires that one-step ahead and smoothed initial conditions be computed from the estimation sample. If you choose one of these two forecasting methods and your forecast period begins either before or after the estimation sample, EViews will issue an error and instruct you to select a different initialization method.

When computing n -step ahead forecasting, EViews will adjust the start of the forecast period so that it is possible to obtain initial conditions for each period using the specified method. For the one-step ahead and smoothed methods, this means that at the earliest, the forecast period will begin $n - 1$ observations into the estimation sample, with earlier forecasted values set to NA. For the other initialization methods, forecast sample endpoint adjustment is not required.

- **Make Signal Series...** allows you to create series containing various signal results computed over the estimation sample. Simply click on the menu entry to display the results dialog.

You may select the one-step ahead predicted signals, $\hat{y}_{t|t-1}$, one-step prediction residuals, $\epsilon_{t|t-1}$, smoothed signal, or signal disturbance estimates, \hat{y}_t or $\hat{\epsilon}_t$. EViews also allows you to save the corresponding standard errors for each of these components (square roots of the diagonal elements of $F_{t|t-1}$, S_t , and $\hat{\Omega}_t$), or the standardized values of the one-step residuals and smoothed disturbances, $e_{t|t-1}$ or \hat{e}_t .

Next, specify the names of your series in the edit field using a list or wildcards as described above. Click **OK** to generate a group containing the desired signal series.

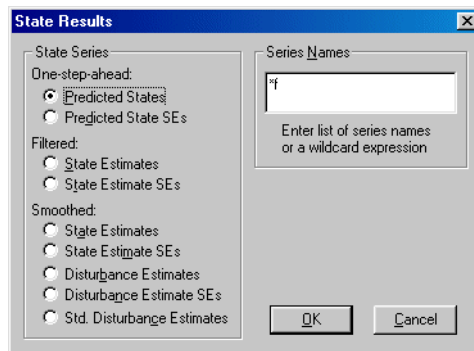
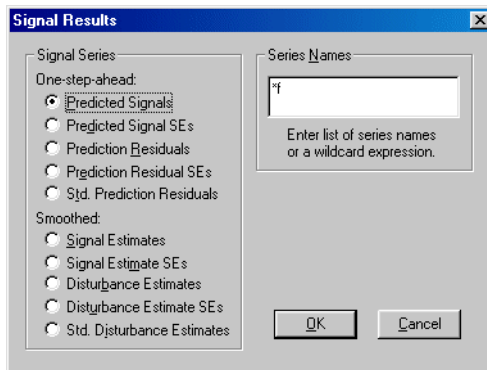
As above, if your signal dependent variable is an expression, EViews will only export results based upon the entire expression.

- **Make State Series...** opens a dialog so you can create series containing results for the state variables computed over the estimation sample. You can choose to save either the one-step ahead state estimate $a_{t|t-1}$, the filtered state mean a_t , the smoothed states \hat{a}_t , state disturbances, \hat{v}_t , standardized state disturbances $\hat{\eta}_t$, or the corresponding standard error series (square roots of the diagonal elements of $P_{t|t-1}$, P_t , V_t and $\hat{\Omega}_t$).

Simply select one of the output types, and enter the names of the output series in the edit field. The rules for specifying the output names are the same as for the **Forecast...** procedure described above. Note that the wildcard expression “*” is permitted when saving state results. EViews will simply use the state names defined in your specification.

We again caution you that if an output series exists in the workfile, EViews will overwrite the entire contents of the series.

- Click on **Make Endogenous Group** to create a group object containing the signal dependent variable series.



- **Make Gradient Group** creates a group object with series containing the gradients of the log likelihood. These series are named “GRAD##” where ## is a unique number in the workfile.
- **Make Kalman Filter** creates a new state space object containing the current specification, but with all parameters replaced by their estimated values. In this way you can “freeze” the current state space for additional analysis. This proc is similar to the **Make Model** proc found in other estimation objects.
- **Make Model** creates a model object containing the state space equations.
- **Update Coefs from Sspace** will place the estimated parameters in the appropriate coefficient vectors.

Converting from Version 3 Sspace

Those of you who have worked with the EViews Version 3 sspace object will undoubtedly be struck by the large number of changes and additional features. In addition to new estimation options, views and procedures, we have changed the underlying specification syntax to provide you with considerable additional flexibility. There are a wide variety of models that may be estimated with the current sspace that were not allowed in the earlier version.

The cost of these additional features and added flexibility is that Version 3 sspace objects are not fully compatible with those in the current version. This has two important practical effects:

- If you load in a workfile which contains a Version 3 sspace object, *all previous estimation results will be cleared* and the text of the specification will be translated to the current syntax. The original text will be retained as comments at the bottom of your sspace specification.
- If you take a workfile which contains a new sspace object and attempt to read it into an earlier version of EViews, the object will not be read, and EViews will warn you that a partial load of the workfile was performed. If you subsequently save the workfile, *the original sspace object will not be saved with the workfile*.

Commands

To declare a sspace object, use the keyword sspace followed by a valid EViews name. You may then use the append proc to add text lines for each line of the sspace.

```
sspace tvp
tvp.append cs = c(1) + sv1*inc
tvp.append @state sv1 = sv1(-1) + [var=c(2)]
```

To estimate the unknown parameters of the model with convergence criterion $1e-8$, maximum iterations 500, and displaying the starting values and other estimation options, you may enter

```
tvpm1(m=500, c=1e-8, showopts)
```

To display the smoothed signals and the one-step ahead state estimates, enter

```
tvpsignalgraphs(t=smooth)
tvpsstategraphs(t=pred)
```

Here we save the four-period ahead state forecasts in SV1, and we save the signal forecasts in the series CSF:

```
tvpm1.forecast(method=n,n=4) @state * @signal csf
```

See “[Sspace](#)” on [page 40](#) of the *Command and Programming Reference* for additional details.

Technical Discussion

Initial Conditions

If there are no @MPRIOR or @VPRIOR statements in the specification, EViews will either: (1) solve for the initial state mean and variance, or (2) initialize the states and variances using diffuse priors.

Solving for the initial conditions is only possible if the state transition matrices T , and variance matrices P and Q are non time-varying and satisfy certain stability conditions (see Harvey, p. 121). If possible, EViews will solve for the conditions $P_{1|0}$ using the familiar relationship: $(I - T \otimes T) \times \text{vec}(P) = \text{vec}(Q)$. If this is not possible, the states will be treated as diffuse unless otherwise specified.

When using diffuse priors, EViews follows the method adopted by Koopman, Shephard and Doornik (1999) in setting $\alpha_{1|0} = 0$, and $P_{1|0} = \kappa I_M$, where the κ is an arbitrarily chosen large number. EViews uses the authors' recommendation that one first set $\kappa = 10^6$ and then adjust it for scale by multiplying by the largest diagonal element of the residual covariances.

Chapter 23. Models

A model in EViews is a set of one or more equations that jointly describe the relationship between a set of variables. The model equations can come from many sources: they can be simple identities, they can be the result of estimation of single equations, or they can be the result of estimation using any one of EViews' multiple equation estimators.

EViews models allow you to combine equations from all these sources inside a single object, which may be used to create a deterministic or stochastic joint forecast or simulation for all of the variables in the model. In a deterministic setting, the inputs to the model are fixed at known values, and a single path is calculated for the output variables. In a stochastic environment, uncertainty is incorporated into the model by adding a random element to the coefficients, the equation residuals or the exogenous variables.

Models also allow you to examine simulation results under different assumptions concerning the variables that are determined outside the model. In EViews, we refer to these sets of assumptions as *scenarios*, and provide a variety of tools for working with multiple model scenarios.

Even if you are working with only a single equation, you may find that it is worth creating a model from that equation so that you may use the features provided by the EViews Model object.

Overview

The following section provides a brief introduction to the purpose and structure of the EViews model object, and introduces terminology that will be used throughout the rest of the chapter.

A model consists of a set of *equations* that describe the relationships between a set of *variables*.

The variables in a model can be divided into two categories: those determined inside the model, which we refer to as the *endogenous variables*, and those determined outside the model, which we refer to as the *exogenous variables*. A third category of variables, the *add factors*, are a special case of exogenous variables.

In its most general form, a model can be written in mathematical notation as:

$$F(y, x) = 0 \tag{23.1}$$

where y is the vector of endogenous variables, x is the vector of exogenous variables, and F is a vector of real-valued functions $f_i(y, x)$. For the model to have a unique solution, there should typically be as many equations as there are endogenous variables.

In EViews, each equation in the model must have a unique endogenous variable assigned to it. That is, each equation in the model must be able to be written in the form

$$y_i = f_i(y, x) \quad (23.2)$$

where y_i is the endogenous variable assigned to equation i . EViews has the ability to *normalize* equations involving simple transformations of the endogenous variable, rewriting them automatically into explicit form when necessary. Any variable that is not assigned as the endogenous variable for any equation is considered exogenous to the model.

Equations in an EViews model can either be *inline* or *linked*. An inline equation contains the specification for the equation as text within the model. A linked equation is one that brings its specification into the model from an external EViews object such as a single or multiple equation estimation object, or even another model. Linking allows you to couple a model more closely with the estimation procedure underlying the equations, or with another model on which it depends. For example, a model for industry supply and demand might link to another model and to estimated equations:

INDUSTRY SUPPLY AND DEMAND MODEL	
←	link to macro model object for forecasts of total consumption
←	link to equation object containing industry supply equation
←	link to equation object containing industry demand equation
←	inline identity: supply = demand

Equations can also be divided into *stochastic equations* and *identities*. Roughly speaking, an identity is an equation that we would expect to hold exactly when applied to real world data, while a stochastic equation is one that we would expect to hold only with random error. Stochastic equations typically result from statistical estimation procedures while identities are drawn from accounting relationships between the variables.

The most important operation performed on a model is to *solve* the model. By solving the model, we mean that for a given set of values of the exogenous variables, X , we will try to find a set of values for the endogenous variables, Y , so that the equations in the model are satisfied within some numerical tolerance. Often, we will be interested in solving the model over a sequence of periods, in which case, for a simple model, we will iterate

through the periods one by one. If the equations of the model contain future endogenous variables we may require a more complicated procedure to solve for the entire set of periods simultaneously.

In EViews, when solving a model, we must first associate data with each variable in the model by *binding* each of the model variables to a series in the workfile. We then solve the model for each observation in the selected sample and place the results in the corresponding series.

When binding the variables of the model to specific series in the workfile, EViews will often modify the name of the variable to generate the name of the series. Typically, this will involve adding an extension of a few characters to the end of the name. For example, an endogenous variable in the model may be called “Y”, but when EViews solves the model, it may assign the result into an observation of a series in the workfile called “Y_0”. We refer to this mapping of names as *aliasing*. Aliasing is an important feature of an EViews model, as it allows the variables in the model to be mapped into different sets of workfile series, without having to alter the equations of the model.

When a model is solved, aliasing is typically applied to the endogenous variables so that historical data is not overwritten. Furthermore, for models which contain lagged endogenous variables, aliasing allows us to bind the lagged variables to either the actual historical data, which we refer to as a *static forecast*, or to the values solved for in previous periods, which we refer to as a *dynamic forecast*. In both cases, the lagged endogenous variables are effectively treated as exogenous variables in the model when solving the model for a single period.

Aliasing is also frequently applied to exogenous variables when using *model scenarios*. Model scenarios allow you to investigate how the predictions of your model vary under different assumptions concerning the path of exogenous variables or add factors. In a scenario, you can change the path of an exogenous variable by overriding the variable. When a variable is *overridden*, the values for that variable will be fetched from a workfile series specific to that scenario. The name of the series is formed by adding a suffix associated with the scenario to the variable name. This same suffix is also used when storing the solutions of the model for the scenario. By using scenarios it is easy to compare the outcomes predicted by your model under a variety of different assumptions without having to edit the structure of your model.

The following table gives a typical example of how model aliasing might map variable names in a model into series names in the workfile:

Model Variable		Workfile Series
endogenous Y	→	Y historical data
	→	Y_0 baseline solution
	→	Y_1 scenario 1
exogenous X	→	X historical data followed by baseline forecast
	→	X_1 overridden forecast for scenario 1

Earlier, we mentioned a third category of variables called *add factors*. An add factor is a special type of exogenous variable that is used to shift the results of a stochastic equation to provide a better fit to historical data or to fine-tune the forecasting results of the model. While there is nothing that you can do with an add factor that could not be done using exogenous variables, EViews provides a separate interface for add factors to facilitate a number of common tasks.

An Example Model

In this section we demonstrate how we can use the EViews model object to implement a simple macroeconomic model of the U.S. economy. The specification of the model is taken from Pindyck and Rubinfeld (1998, p. 390). We have provided the data and other objects relating to the model in the sample workfile MACROMOD.WF1. You may find it useful to follow along with the steps in the example, and you can use the workfile to experiment further with the model object.

The macro model contains three stochastic equations and one identity. In EViews notation, these can be written:

$$\begin{aligned}
 cn &= c(1) + c(2)*y + c(3)*cn(-1) \\
 i &= c(4) + c(5)*(y(-1)-y(-2)) + c(6)*y + c(7)*r(-4) \\
 r &= c(8) + c(9)*y + c(10)*(y-y(-1)) + c(11)*(m-m(-1)) + c(12)* \\
 &\quad (r(-1)+r(-2)) \\
 y &= cn + i + g
 \end{aligned}$$

where

- CN is real personal consumption
- I is real private investment

- G is real government expenditure
- Y is real GDP less net exports
- R is the interest rate on three-month treasury bills
- M is the real money supply, narrowly defined (M1)

and the $C(i)$ are the unknown coefficients.

The model follows the structure of a simple textbook ISLM macroeconomic model, with expenditure equations relating consumption and investment to GDP and interest rates, and a money market equation relating interest rates to GDP and the money supply. The fourth equation is the national accounts expenditure identity which ensures that the components of GDP add to total GDP. The model differs from a typical textbook model in its more dynamic structure, with many of the variables appearing in lagged or differenced form.

To begin, we must first estimate the unknown coefficients in the stochastic equations. For simplicity, we estimate the coefficients by simple single equation OLS. Note that this approach is not strictly valid, since Y appears on the right-hand side of several of the equations as an independent variable but is endogenous to the system as a whole. Because of this, we would expect Y to be correlated with the residuals of the equations, which violates the assumptions of OLS estimation. To adjust for this, we would need to use some form of instrumental variables or system estimation (for details, see the discussion of single equation “Two-stage Least Squares” beginning on page 283 and system “Two-Stage Least Squares” and related sections beginning on page 497).

To estimate the equations in EViews, we create three new equation objects in the workfile (using **Objects/New Object/Equation**), and then enter the appropriate specifications. Since all three equations are linear, we can specify them using list form. To minimize confusion, we will name the three equations according to their endogenous variables. The resulting names and specifications are:

Equation eqcn:	$cn \ c \ y \ cn(-1)$
Equation eqi:	$i \ c \ y(-1)-y(-2) \ y \ r(-4)$
Equation eqr:	$r \ c \ y \ y-y(-1) \ m-m(-1) \ r(-1) + r(-2)$

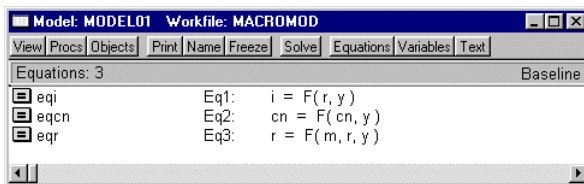
The three equations estimate satisfactorily and provide a reasonably close fit to the data, although much of the fit probably comes from the lagged endogenous variables. The consumption and investment equations show signs of heteroskedasticity, possibly indicating that we should be modeling the relationships in log form. All three equations show signs of serial correlation. We will ignore these problems for the purpose of this example, although you may like to experiment with alternative specifications and compare their performance.

Now that we have estimated the three equations, we can proceed to the model itself. To create the model, we simply select **Objects/New Object/Model** from the menus. To keep the model permanently in the workfile, we name the model by clicking on the **Name** button, and clicking on **OK** to accept the suggested name of “MODEL01”.

When first created, the model object defaults to *equation view*. Equation view allows us to browse through the specifications and properties of the equations contained in the model. Since we have not yet added any equations to the model, this window will appear empty.

To add our estimated stochastic equations to the model, we can simply copy-and-paste them across from the workfile window. To copy-and-paste, first select the objects in the workfile window, and then use **Edit/Copy** or the right mouse button menu to copy the objects to the clipboard. Click anywhere in the model object window, and use **Edit/Paste** or the right mouse button menu to paste the objects into the model object window.

The three estimated equations should now appear in the equation window. Each equation appears on a line with an icon showing the type of object, its name, its equation number, and a symbolic representation of the equation in terms of the variables that it contains. Double clicking on any equation will bring up a dialog of properties of that equation. For the moment, we do not need to alter any of these properties.

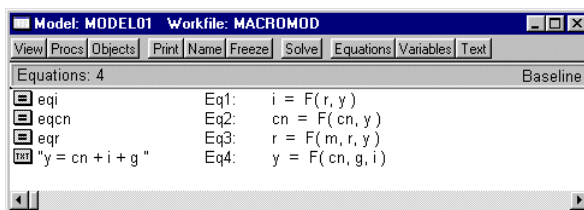


We have added our three equations as linked equations. This means if we go back and reestimate one or more of the equations, we can automatically update the equations in the model to the new estimates by using the procedure **Procs/Links/Update All Links**.

To complete the model, we must add our final equation, the national accounts expenditure identity. There is no estimation involved in this equation, so instead of including the equation via a link to an external object, we merely add the equation as inline text.

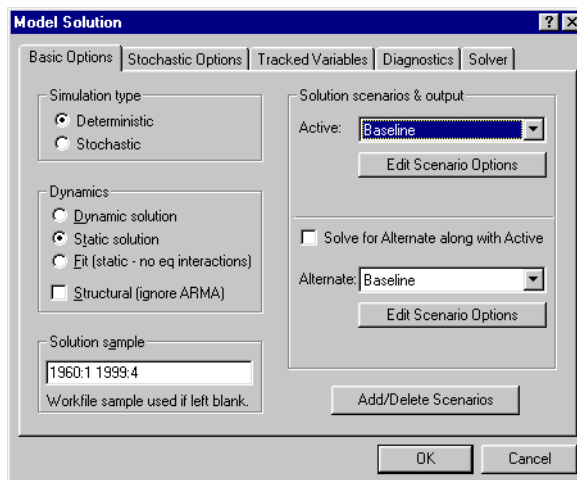
To add the identity, we click with the right mouse button anywhere in the equation window, and select **Insert...** A dialog box will appear titled **Model Source Edit** which contains a text box with the heading **Enter one or more lines**. Simply type the identity, “ $Y = CN + I + G$ ”, into the text box, then click on **OK** to add it to the model.

The equation should now appear in the model window. The appearance differs slightly from the other equations, which is an indicator that the new equation is an inline text equation rather than a link.



Our model specification is now complete. At this point, we can proceed straight to solving the model. To solve the model, simply click on the **Solve** button in the model window button bar.

There are many options available from the dialog, but for the moment we will consider only the basic settings. As our first exercise in assessing our model, we would like to examine the ability of our model to provide one-period ahead forecasts of our endogenous variables. To do this, we can look at the predictions of our model against our historical data, using actual values for both the exogenous and the lagged endogenous variables of the model. In EViews, we refer to this as a *static* simulation. We can easily perform this type of simulation by choosing **Static solution** in the **Dynamics** box of the dialog.



We must also adjust the sample over which to solve the model, so as to avoid initializing our solution with missing values from our data. Most of our series are defined over the range of 1947:1 to 1999:4, but our money supply series is available only from 1959:1. Because of this, we set the sample to 1960:1 to 1999:4, allowing a few extra periods prior to the sample for any lagged variables.

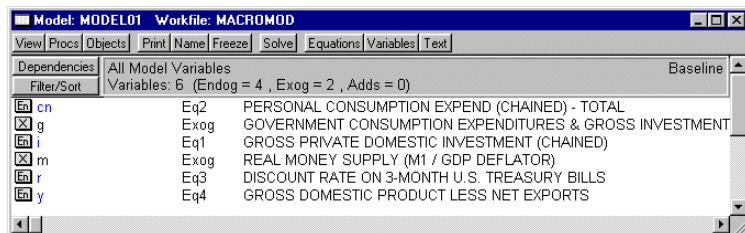
We are now ready to solve the model. Simply click on **OK** to start the calculations. The model window will switch to the Solution Messages view.

The output should be fairly self-explanatory. In this case, the solution took less than a second and there were no errors while performing the calculations.

Now that we have solved the model, we would like to look at the results. When we solved the model, the results for the endogenous variables were placed into series in the workfile with names determined by the name aliasing rules of the model. Since these series are ordinary EViews objects, we could use the workfile window to open the series and examine them directly. However, the model object provides a much more convenient way to work with the series through a view called the Variable View.

The easiest way to switch to the variable view is to click on the button labeled **Variables** on the model window button bar.

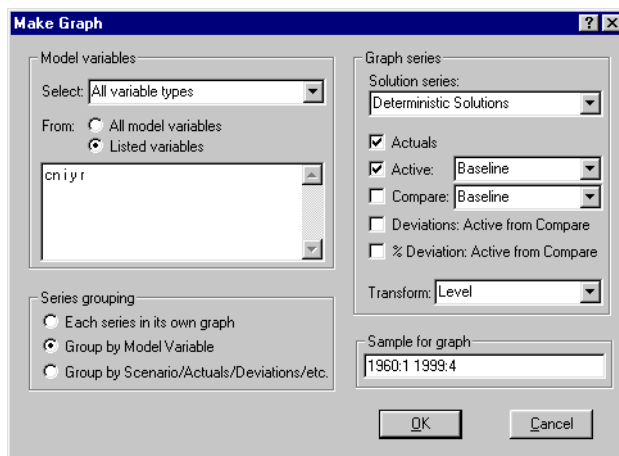
In the variable view, each line in the window is used to represent a variable. The line contains an icon indicating the variable



type (endogenous, exogenous or add factor), the name of the variable, the equation with which the variable is associated (if any), and the description field from the label of the underlying series (if available). The name of the variable may be colored according to its status, indicating whether it is being traced (blue) or whether it has been overridden (red). In our model, we can see from the variable view that CN, I, R and Y are endogenous variables in the model, while G and M are exogenous.

Much of the convenience of the variable view comes from the fact that it allows you to work directly with the names of the variables in the model, rather than the names of series in the workfile. This is useful because when working with a model, there will often be many different series associated with each variable. For endogenous variables there will be the actual historical values, and one or more series of solution values. For exogenous variables, there may be several alternative scenarios for the variable. The variable view and its associated procedures help you move between these different sets of series without having to worry about the many different names involved.

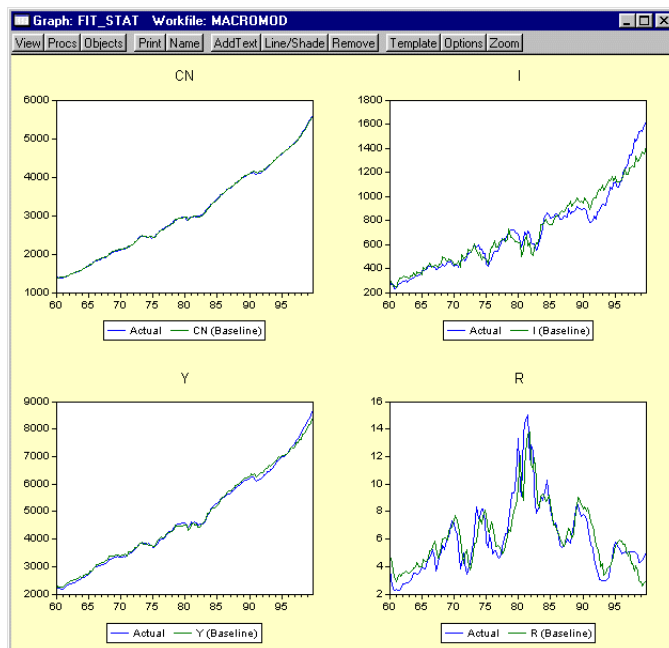
For example, to look at graphs containing the actual and fitted values for the endogenous variables in our model, we simply select the four variables (by holding down the control key and clicking on the variable names), then use **Procs/Make Graph...** to enter the dialog. Again, the dialog has many options, but for our current purposes, we can leave most settings at their default values. Simply make sure that the **Actuals** and **Active** checkboxes are checked, set the sample for the graph to 1960:1 to 1999:4, then click on **OK**.



The graphs show that as a one-step ahead predictor, the model performs quite well, although the ability of the model to predict investment deteriorates during the second half of the sample.

An alternative way of evaluating the model is to examine how the model performs when used to forecast many periods into the future. To do this, we must use our forecasts from previous periods, not actual historical data, when assigning values to the lagged endogenous terms in our model. In EViews, we refer to such a forecast as a *dynamic* forecast.

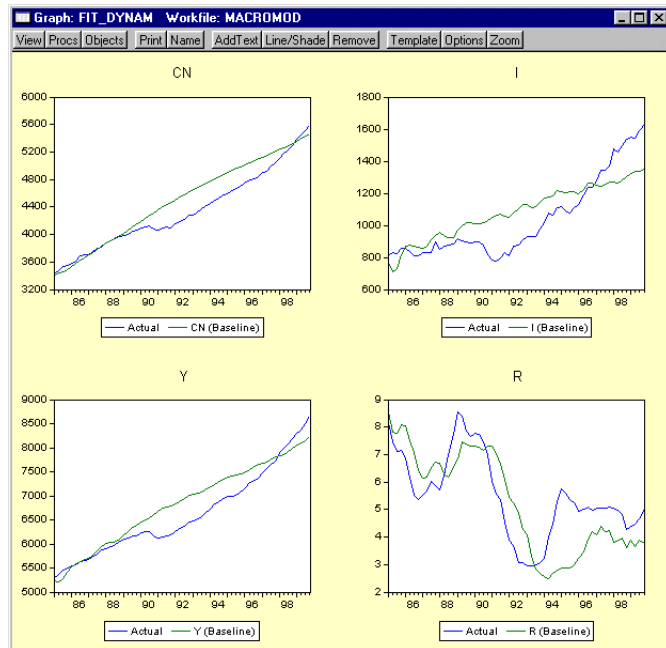
To perform a dynamic forecast, we resolve the model with a slightly



different set of options. Return to the model window and again click on the **Solve** button. In the model solution dialog, choose **Dynamic solution** in the **Dynamics** section of the dialog, and set the solution sample to 1985:1 to 1999:4.

Click on **OK** to solve the model. To examine the results, return to the variable view, select the endogenous series again, and use **Procs/Make Graph...** exactly as above. Make sure the sample is set to 1985:1 to 1999:4 then click on **OK**. The results illustrate how our model would have performed if we had used it back in 1985 to make a forecast for the economy over the next fifteen years, assuming that we had used the correct paths for the exogenous variables. (In reality, we would not have known these values at the time the forecasts were generated). Not surprisingly, the results show substantial deviations from the actual outcomes, although they do seem to follow the general trends in the data.

Once we are satisfied with the performance of our model against historical data, we can use the model to forecast future values of our endogenous variables. The first step in producing such a forecast is to decide on values for our exogenous variables during the forecast period. These may be based on our best guess as to what will actually happen, or they may be simply one particular possibility that we are interested in considering. Often we will be interested in constructing several different paths and then comparing the results.



In our model, we must provide future values for our two exogenous variables: government expenditure (G), and the real money supply (M). For our example we will try and construct a set of paths that broadly follow the trends of the historical data.

A quick look at our historical series for G suggests that the growth rate of G has been fairly constant since 1960, so that the log of G roughly follows a linear trend. Where G deviates from the trend, the deviations seem to follow a cyclical pattern.

As a simple model of this behavior, we can regress the log of G against a constant and a time trend, using an AR(4) error structure to model the cyclical deviations. This gives the following equation, which we save in the workfile as EQG:

$$\log(g) = 6.252335365 + 0.004716422176*\text{@trend} +$$

$$[\text{ar}(1)=1.169491542, \text{ar}(2)=-0.1986105967, \text{ar}(3)=0.2399131262,$$

$$\text{ar}(4)=-0.245360709]$$

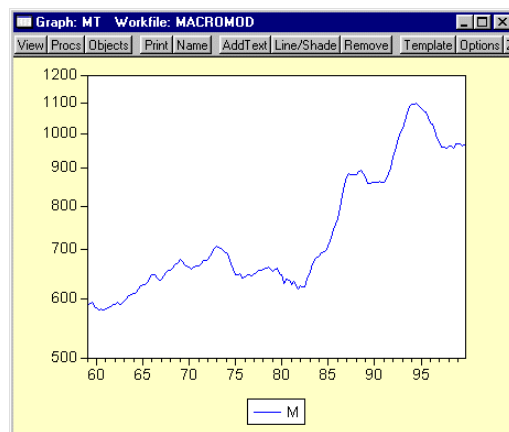
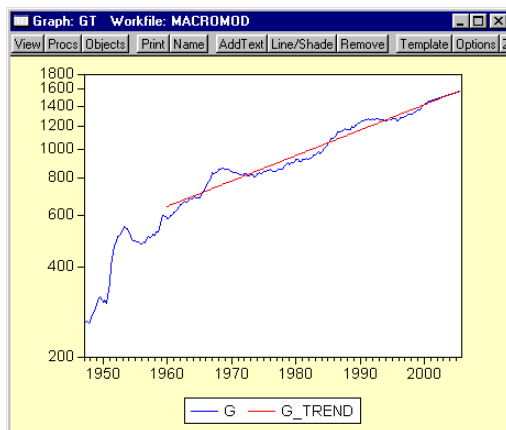
To produce a set of future values for G , we can use this equation to perform a dynamic forecast from 2000:1 to 2005:4, saving the results back into G itself (see [page 621](#) for details).

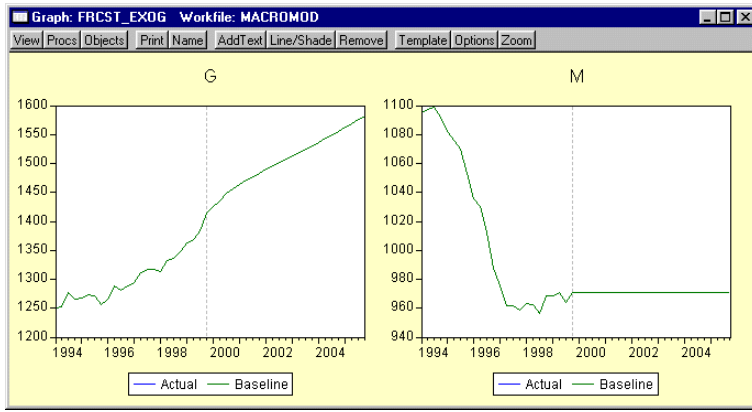
The historical path of the real M1 money supply, M , is quite different from G , showing spurts of growth followed by periods of stability. For now, we will assume that the real money supply simply remains at its last observed historical value over the entire forecast period.

We can use an EViews series statement to fill in this path. The following lines will fill the series M from 2000:1 to the last observation in the sample with the last observed historical value for M :

```
smp1 2000:1 @last
series m = m(-1)
smp1 @all
```

We now have a set of possible values for our exogenous variables over the forecast period. After plotting, we have





To produce forecasts for our *endogenous* variables, we return to the model window, click on **Solve**, choose **Dynamic Solution**, set the forecast sample for 2000:1 to 2005:4, and then click on **OK**. The Solution Messages screen should appear, indicating that the model was successfully solved.

To examine the results in a graph, we again use **Procs/Make Graph...** from the variables view, set the sample to 1995:1 to 2005:4 (so that we include five years of historical data), then click on **OK**. After adding a line in 1999:4 to separate historical and actual results, we get the graph:

There is some strange behavior in the results. At the beginning of the forecast period we see a heavy dip in investment, gdp and interest rates. This is followed by a series of oscillations in these series with a period of about a year, which die out slowly during the forecast period. This is not a particularly convincing forecast.

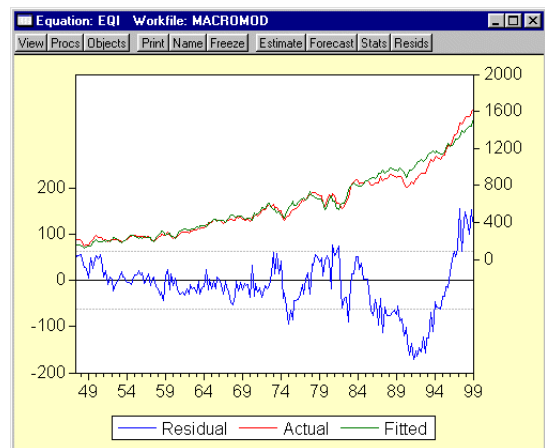
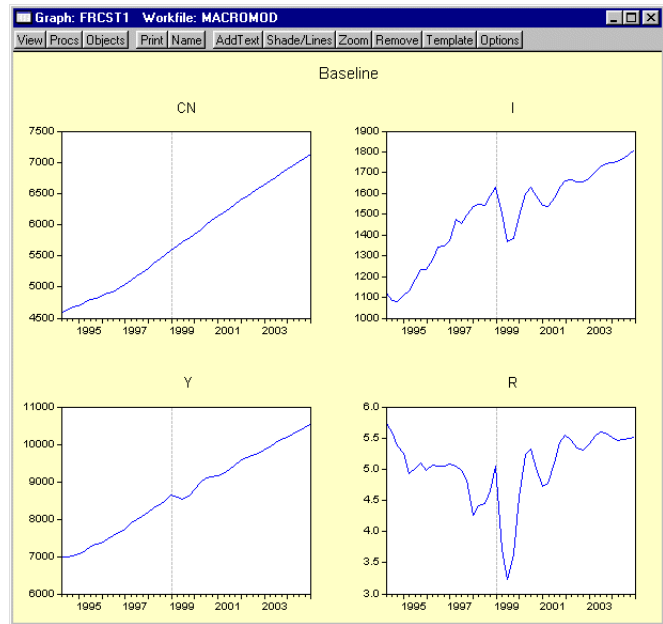
There is little in the paths of our exogenous variables or the history of our endogenous variables that would lead to this sharp dip, suggesting that the

problem may lie with the residuals of our equations. Our investment equation is the most likely candidate, as it has a large, persistent positive residual near the end of the historical data (see figure below). This residual will be set to zero over the forecast period when solving the model, which might be the cause of the sudden drop in investment at the beginning of the forecast.

One way of dealing with this problem would be to change the specification of the investment equation. The simplest modification would be to add an autoregressive component to the equation, which would help reduce the persistence of the error. A better alternative would be to try to modify the variables in the equation so that the equation can provide some explanation for the sharp rise in investment during the 1990s.

An alternative approach to the problem is to leave the equation as it is,

but to include an add factor in the equation so that we can model the path of the residual



by hand. To include the add factor, we switch to the equation view of the model, double click on the investment equation, EQI, select the **Add factor** tab. Under **Factor type**, choose **Equation intercept (residual shift)**. A prompt will appear asking if we would like to create the add factor series. Click on **Yes** to create the series. When you return to the variable view, you should see that a new variable, I_A, has been added to the list of variables in the model.

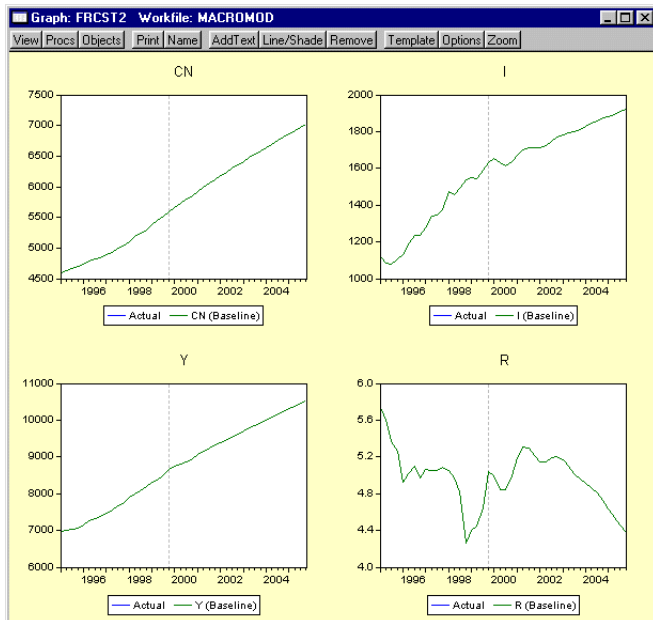
Using the add factor, we can specify any path we choose for the residual of the investment equation during the forecast period. By examining the **Actual/Fitted/Residual Graph** view from the equation object, we see that near the end of the historical data, the residual appears to be hovering around a value of about 160. We will assume that this value holds throughout the forecast period. We can set the add factor using a few simple EViews commands:

```
smp1 2000:1 @last
i_a = 161
smp1 @all
```

With the add factor in place, we can follow exactly the same procedure that we followed above to produce a new set of solutions for the model and a new graph for the results.

Including the add factor in the model has made the results far more appealing. The sudden dip in the first period of the forecast that we saw above has been removed. The oscillations are still apparent, but are much less pronounced.

So far, we have been working under the assumption that our stochastic equations hold exactly over the forecast period. In reality, we would expect to see the same sort of errors occurring in the future as we have seen in history. We have also been ignoring the fact that some of the coefficients



in our equations are estimated, rather than fixed at known values. We may like to reflect this uncertainty about our coefficients in some way in the results from our model.

We can incorporate these features into our EViews model using stochastic simulation.

Up until now, we have thought of our model as forecasting a single point for each of our endogenous variables at each observation. As soon as we add uncertainty to the model, we should think instead of our model as predicting a whole distribution of outcomes for each variable at each observation. Our goal is to summarize these distributions using appropriate statistics.

If the model is linear (as in our example) and the errors are normal, then the endogenous variables will follow a normal distribution, and the mean and standard deviation of each distribution should be sufficient to describe the distribution completely. In this case, the mean will actually be equal to the deterministic solution to the model. If the model is not linear, then the distributions of the endogenous variables need not be normal. In this case, the quantiles of the distribution may be more informative than the first two moments, since the distributions may have tails which are very different from the normal case. In a non-linear model, the mean of the distribution need not match up to the deterministic solution of the model.

EViews makes it easy to calculate statistics to describe the distributions of your endogenous variables in an uncertain environment. To simulate the distributions, the model object uses a Monte Carlo approach, where the model is solved many times with pseudo-random numbers substituted for the unknown errors at each repetition. This method provides only approximate results. However, as the number of repetitions is increased, we would expect the results to approach their true values.

To return to our simple macroeconomic model, we can use a stochastic simulation to provide some measure of the uncertainty in our results by adding error bounds to our predictions. From the model window, click on the **Solve** button. When the model solution dialog appears, choose **Stochastic** for the simulation type. In the **Solution scenarios & output** box, make sure that the **Std. Dev.** checkbox in the **Active** section is checked. Click on **OK** to begin the simulation.

The simulation should take about half a minute. Status messages will appear to indicate progress through the repetitions. When the simulation is complete, you may return to the variable view, use the mouse to select the variables as discussed above, and then select **Procs/Make Graph...** When the **Make Graph** dialog appears, select the option **Mean + - 2 standard deviations** in the **Solution Series** list box in the **Graph Series** area on the right of the dialog. Set the sample to 1995:1 to 2005:4 and click on **OK**.

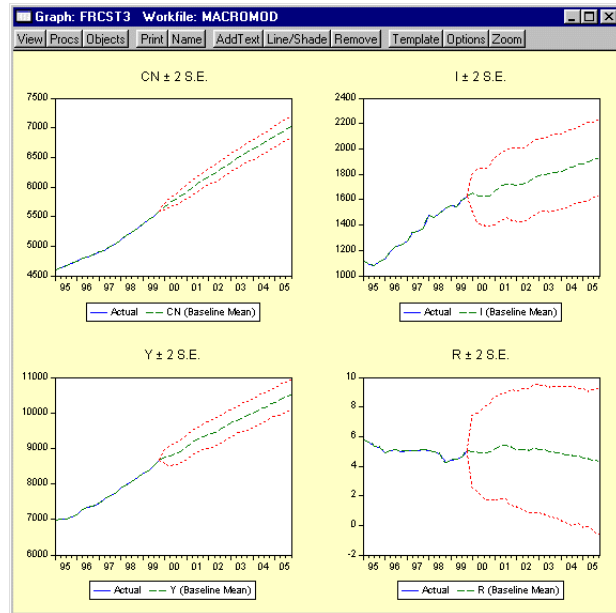
The error bounds in the resulting output graph show that we should be reluctant to place too much weight on the point forecasts of our model, since the bounds are quite wide on several of the variables. Much of the uncertainty is probably due to the large residual in the investment equation, which is creating a lot of variation in investment and interest rates in the stochastic simulation.

Another exercise we might like to consider when working with our model is to examine how the model behaves under alternative

assumptions with respect to the exogenous variables. One approach to this would be to directly edit the exogenous series so that they contain the new values, and then resolve the model, overwriting any existing results. The problem with this approach is that it makes it awkward to manage the data and to compare the different sets of outcomes.

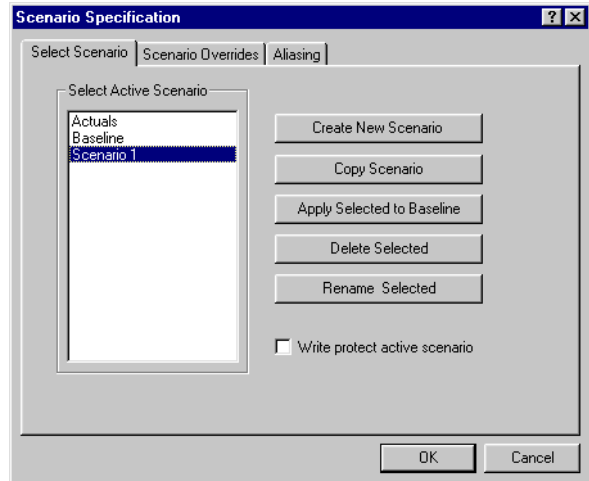
EViews provides a better way of carrying out exercises such as this through the use of model scenarios. Using a model scenario, you can override a subset of the exogenous variables in a model to give them new values, while using the values stored in the actual series for the remainder of the variables. When you solve for a scenario, the values of the endogenous variables are assigned into workfile series with an extension specific to that scenario, making it easy to keep multiple solutions for the model within a single workfile.

To create a scenario, we begin by selecting **View/Scenarios...** from the model object menus. The scenario specification dialog will appear with a list of the scenarios currently included in the model. There are two special scenarios that are always present in the model: **Actuals** and **Baseline**. These two scenarios are special in that they cannot contain any overridden variables.



They differ in that the actuals scenario writes its solution values directly into the workfile series with the same names as the endogenous variables, while the baseline scenario writes its solution values back into workfile series with the extension “_0”.

To add a new scenario to the model, simply click on the button labeled **Create new scenario**. A new scenario will be created immediately. You can use this dialog to select which scenario is currently active, or to rename and delete scenarios.



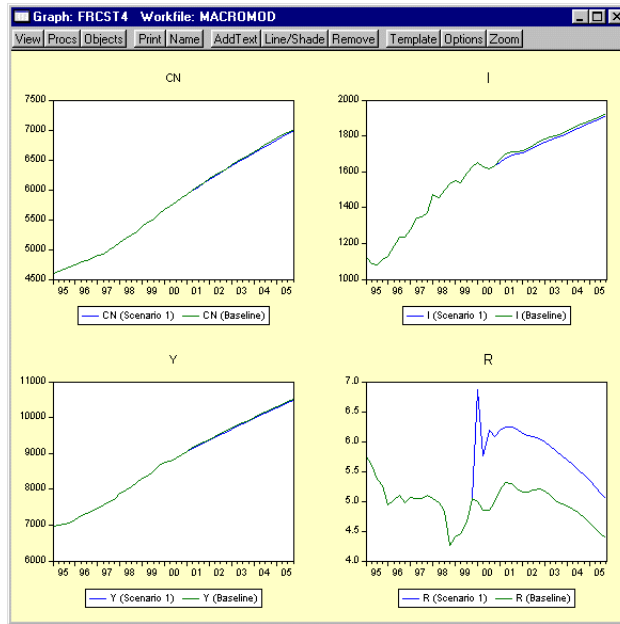
Once we have created the scenario, we can modify the scenario from the baseline case by overriding one of our exogenous variables. To do this, return to the variable window of the model, click on the variable *M*, use the right mouse button to call up the **Properties** dialog for the variable, and then in the **Scenario** box, click on the checkbox for **Use override series in scenario**. A message will appear asking if you would like to create the new series. Click on **Yes** to create the series, then **OK** to return to the variable window.

In the variable window, the variable name “*M*” should now appear in red, indicating that it has been overridden in the active scenario. This means that the variable *M* will now be bound to the series *M_1* instead of the series *M* when solving the model. In our previous forecast for *M*, we assumed that the real money supply would be kept at a constant level during the forecast period. For our alternative scenario, we are going to assume that the real money supply is contracted sharply at the beginning of the forecast period, and held at this lower value throughout the forecast. We can set the new values using a few simple commands:

```
smp1 2000:1 2005:4
series m_1 = 900
smp1 @all
```

As before, we can solve the model by clicking on the **Solve** button. Restore the **Simulation type** to deterministic, make sure that Scenario 1 is the active scenario, then click on **OK**. Once the solution is complete, we can use **Procs/Make Graph...** to display the results following the same procedure as above. Restore the **Solution series** list box to the setting **Deterministic solutions**, then check both the **Active** and **Compare** solution checkboxes

below, making sure that the active scenario is set to Scenario 1, and the comparison scenario is set to Baseline. Again set the sample to 1995:1 to 2005:4. The following graph should be displayed:



The simulation results suggest that the cut in the money supply causes a substantial increase in interest rates, which creates a small reduction in investment and a relatively minor drop in income and consumption. Overall, the predicted effects of changes in the money supply on the real economy are relatively minor in this model.

This concludes the discussion of our example model. The remainder of this chapter provides detailed information about working with particular features of the EViews model object.

Building a Model

Creating a Model

The first step in working with a model is to create the model object itself. There are several different ways of creating a model:

- You can create an empty model by using **Objects/New Object...** and then choosing **Model**, or by performing the same operation using the right mouse button menu from inside the workfile window.

- You can select a list of estimation objects in the workfile window (equations, VARs, systems), and then select **Open as Model** from the right mouse button menu. This item will create a model which contains the equations from the selected objects as links.
- You can use the **Make model** procedure from an estimation object to create a model containing the equation or equations in that object.

Adding Equations to the Model

The equations in a model can be classified into two types: linked equations and inline equations. Linked equations are equations that import their specification from other objects in the workfile. Inline equations are contained inside the model as text.

There are a number of ways to add equations to your model:

- To add a linked equation: from the workfile window, select the object which contains the equation or equations you would like to add to the model, then copy-and-paste the object into the model equation view window.
- To add an equation using text: select **Insert...** from the right mouse button menu. In the text box titled: **Enter one or more lines...**, type in one or more equations in standard EViews format. You can also add linked equations from this dialog by typing a colon followed by the name of the object you would like to link to, for example “:EQ1”, because this is the text form of a linked object.

In an EViews model, the first variable that appears in an equation will be considered the endogenous variable for that equation. Since each endogenous variable can be associated with only one equation, you may need to rewrite your equations to ensure that each equation begins with a different variable. For example, say we have an equation in the model

$$x / y = z$$

EViews will associate the equation with the variable X. If we would like the equation to be associated with the variable Y, we would have to rewrite the equation:

$$1 / y * x = z$$

Note that EViews has the ability to handle simple expressions involving the endogenous variable. You may use functions like $\log()$, $D()$ and $Dlog()$ on the left-hand side of your equation. EViews will normalize the equation into explicit form if the Gauss-Seidel method is selected for solving the model.

Removing equations from the model

To remove equations from the model, simply select the equations using the mouse in Equation view, then use **Delete** from the right mouse button menu to remove the equations.

Both adding and removing equations from the model will change which variables are considered endogenous to the model.

Updating Links in the Model

If a model contains linked equations, changes to the specification of the equations made outside the model can cause the equations contained in the model to become out of date. You can incorporate these changes in the model by using the procedure **Update All Links**. Alternatively, you can update just a single equation using the **Update Link** item from the right mouse button menu. Links are also updated when a workfile is reloaded from disk.

Sometimes, you may want to sever equations in the model from their linked objects. For example, you may wish to see the entire model in text form, with all equations written in place. To do this, you can use the **Break All Links** procedure to convert all linked equations in the model into inline text. You can convert just a single equation by selecting the equation, then using **Break Link** from the right mouse button menu.

When a link is broken, the equation is written in text form with the unknown coefficients replaced by their point estimates. Any information relating to uncertainty of the coefficients will be lost. This will have no effect on deterministic solutions to the model, but may alter the results of stochastic simulations if the **Include coefficient uncertainty** option has been selected.

Working with the Model Structure

As with other objects in EViews, we can look at the information contained in the model object in several ways. Since a model is a set of equations that describe the relationship between a set of variables, the two primary views of a model are the equation view and the variable view. EViews also provides two additional views of the structure of the model: the block view and the text view.

Equation View

The equation view is used for displaying, selecting, and modifying the equations contained in the model. An example of the equation view can be seen on [page 607](#).

Each line of the window is used to represent either a linked object or an inline text equation. Linked objects will appear similarly to how they do in the workfile, with an icon representing their type, followed by the name of the object. Even if the linked object contains

many equations, it will use only one line in the view. Inline equations will appear with a “TXT” icon, followed by the beginning of the equation text in quotation marks.

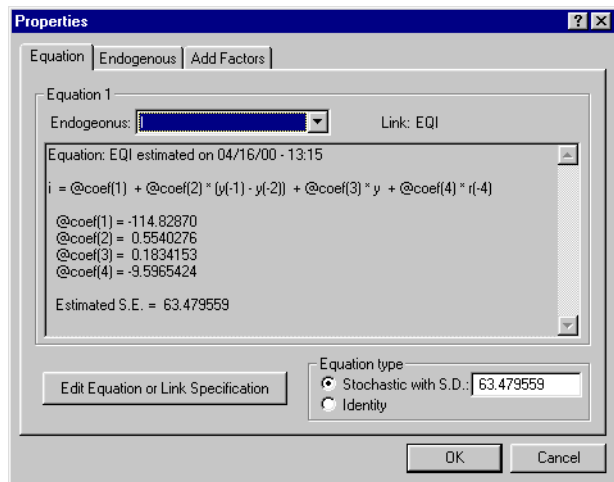
The remainder of the line contains the equation number, followed by a symbolic representation of the equation, indicating which variables appear in the equation.

Any errors in the model will appear as red lines containing an error message describing the cause of the problem.

You can open any linked objects directly from the equation view. Simply select the line representing the object using the mouse, then choose **Open Link** from the right mouse button menu.

The contents of a line can be examined in more detail using the equation properties dialog. Simply select the line with the mouse, then choose **Properties...** from the right mouse button menu. Alternatively, simply double click on the object to call up the dialog

For a link to a single equation, the dialog shows the functional form of the equation, the values of any estimated coefficients, and the standard error of the equation residual from the estimation. If the link is to an object containing many equations, you can move between the different equations imported from the object using the **Endogenous** list box at the top of the dialog. For an inline equation, the dialog simply shows the text of the equation.



The **Edit Equation or Link Specification** button allows you to edit the text of an inline equation or to modify a link to point to an object with a different name. A link is represented in text form as a colon followed by the name of the object. Note that you cannot modify the specification of a linked object from within the model object, you must work directly with the linked object itself.

In the bottom right of the dialog there are a set of fields that allow you to set the stochastic properties of the residual of the equation. If you are only performing deterministic simulations, then these settings will not affect your results in any way. If you are performing sto-

chastic simulations, then these settings are used in conjunction with the solution options to determine the size of the random innovations applied to this equation.

The **Stochastic with S.D.** option for **Equation type** lets you set a standard deviation for any random innovations applied to the equation. If the standard deviation field is blank or is set to “NA”, then the standard deviation will be estimated from the historical data. The **Identity** option specifies that the selected equation is an identity, and should hold without error even in a stochastic simulation. See “[Stochastic Options](#)” on page 634 below for further details.

The equation properties dialog also gives you access to the property dialogs for the endogenous variable and add factor associated with the equation. Simply click on the appropriate tab. These will be discussed in greater detail below.

Variable View

The variable view is used for adjusting options related to variables and for displaying and editing the series associated with the model (see the discussion in “[An Example Model](#)” (p. 608)). The variable view lists all the variables contained in the model, with each line representing one variable. Each line begins with an icon classifying the variable as endogenous, exogenous or an add factor. This is followed by the name of the variable, the equation number associated with the variable, and the description of the variable. The description is read from the associated series in the workfile.

Note that the names and types of the variables in the model are determined fully by the equations of the model. The only way to add a variable or to change the type of a variable in the model is to modify the model equations.

You can adjust what is displayed in the variable view in a number of ways. By clicking on the **Filter/Sort** button just above the variable list, you can choose to display only variables that match a certain name pattern, or to display the variables in a particular order. For example, sorting by type of variable makes the division into endogenous and exogenous variables clearer, while sorting by override highlights which variables have been overridden in the currently active scenario.

The variable view also allows you to browse through the dependencies between variables in the model by clicking on the **Dependencies** button. Each equation in the model can be thought of as a set of links that connect other variables in the model to the endogenous variable of the equation. Starting from any variable, we can travel up the links, showing all the endogenous variables that this variable directly feeds into, or we can travel down the links, showing all the variables upon which this variable directly depends. This may sometimes be useful when trying to find the cause of unexpected behavior. Note, however, that in a simultaneous model, every endogenous variable is indirectly connected to every other

variable in the same block, so that it may be hard to understand the model as a whole by looking at any particular part.

You can quickly view or edit one or more of the series associated with a variable by double clicking on the variable. For several variables, simply select each of them with the mouse then double click inside the selected area.

Block Structure View

The block structure view of the model analyzes and displays any block structure in the dependencies of the model.

Block structure refers to whether the model can be split into a number of smaller parts, each of which can be solved for in sequence. For example, consider the system:

block 1	$x = y + 4$
	$y = 2 * x - 3$
block 2	$z = x + y$

Because the variable Z does not appear in either of the first two equations, we can split this equation system into two blocks: a block containing the first two equations, and a block containing the third equation. We can use the first block to solve for the variables X and Y, then use the second block to solve for the variable Z. By using the block structure of the system, we can reduce the number of variables we must solve for at any one time. This typically improves performance when calculating solutions.

Blocks can be classified further into *recursive* and *simultaneous* blocks. A recursive block is one which can be written so that each equation contains only variables whose values have already been determined. A recursive block can be solved by a single evaluation of all the equations in the block. A simultaneous block cannot be written in a way that removes feedback between the variables, so it must be solved as a simultaneous system. In our example above, the first block is simultaneous, since X and Y must be solved for jointly, while the second block is recursive, since Z depends only on X and Y, which have already been determined in solving the first block.

The block structure view displays the structure of the model, labeling each of the blocks as recursive or simultaneous. EViews uses this block structure whenever the model is solved. The block structure of a model may also be interesting in its own right, since reducing the system to a set of smaller blocks can make the dependencies in the system easier to understand.

Text View

The text view of a model allows you to see the entire structure of the model in a single screen of text. This provides a quick way to input small models, or a way to edit larger models using copy-and-paste.

The text view consists of a series of lines. In a simple model, each line simply contains the text of one of the inline equations of the model. More complicated models may contain one of more of the following:

- A line beginning with a colon “:” represents a link to an external object. The colon must be followed by the name of an object in the workfile. Equations contained in the external object will be imported into the model whenever the model is opened, or when links are updated.
- A line beginning with “@ADD” specifies an add factor. The add factor command has the form:

```
@add(v) endogenous_name add_name
```

where `endogenous_name` is the name of the endogenous variable of the equation to which the add factor will be applied, and `add_name` is the name of the series. The option `(v)` is used to specify that the add factor should be applied to the endogenous variable. The default is to apply the add factor to the residual of the equation. See [“Using Add Factors” on page 626](#) for details.

- A line beginning with “@INNOV” specifies an innovation variance. The innovation variance has two forms. When applied to an endogenous variable it has the form

```
@innov endogenous_name number
```

where `endogenous name` is the name of the endogenous variable and `number` is the standard deviation of the innovation to be applied during stochastic simulation. When applied to an exogenous variable, it has the form

```
@innov exogenous_name number_or_series
```

where `exogenous name` is the name of the exogenous variable and `number_or_series` is either a number or the name of the series that contains the standard deviation to be applied to the variable during stochastic simulation. Note that when an equation in a model is linked to an external estimation object, the variance from the estimated equation will be brought into the model automatically and does not require an `@innov` specification unless you would like to modify its value.

- The keyword “@TRACE”, followed by the names of the endogenous variables that you wish to trace may be used to request model solution diagnostics. See [“Diagnostics” on page 637](#).

Users of earlier versions of EViews should note that two commands that were previously available, `@assign` and `@exclude`, are no longer part of the text form of the model. These commands have been removed because they now address options that apply only to specific model scenarios rather than to the model as a whole. When loading in models created by earlier versions of EViews, these commands will be converted automatically into scenario options in the new model object.

Specifying Scenarios

When working with a model, you will often want to compare model predictions under a variety of different assumptions regarding the paths of your exogenous variables, or with one or more of your equations excluded from the model. Model scenarios allow you to do this without overwriting previous data or changing the structure of your model.

The most important function of a scenario is to specify which series will be used to hold the data associated with a particular solution of the model. To distinguish the data associated with different scenarios, each scenario modifies the names of the model variables according to an aliasing rule. Typically, aliasing will involve adding an underline followed by a number, such as “_0” or “_1” to the variable names of the model. The data for each scenario will be contained in series in the workfile with the aliased names.

Model scenarios support the analysis of different assumptions for exogenous variables by allowing you to override a set of variables you would like to alter. Exogenous variables which are overridden will draw their values from series with names aliased for that scenario, while exogenous variables which are not overridden will draw their values from series with the same name as the variable.

Scenarios also allow you to exclude one or more endogenous variables from the model. When an endogenous variable is excluded, the equation associated with that variable is dropped from the model and the value of the variable is taken directly from the workfile series with the same name. Excluding an endogenous variable effectively treats the variable as an exogenous variable for the purposes of solving the model.

When excluding an endogenous variable, you can specify a sample range over which the variable should be excluded. One use of this is to handle the case where more recent historical data is available for some of your endogenous variables than others. By excluding the variables for which you have data, your forecast can use actual data where possible, and results from the model where data are not yet available.

Each model can contain many scenarios. You can view the scenarios associated with the current model by choosing **View/Scenario Specification...** as shown above on [page 617](#).

There are two special scenarios associated with every model: actuals and baseline. These two scenarios have in common the special property that they cannot contain any overrides

or excludes. They differ in that the actuals scenario writes the values for endogenous variables back into the series with the same name as the variables in the model, while the baseline scenario modifies the names. When solving the model using actuals as your active scenario, you should be careful not to accidentally overwrite your historical data.

The baseline scenario gets its name from the fact that it provides the base case from which other scenarios are constructed. Scenarios differ from the baseline by having one or more variables overridden or excluded. By comparing the results from another scenario against those of the baseline case, we can separate out the movements in the endogenous variables that are due to the changes made in that particular scenario from movements which are present in the baseline itself.

The **Select Scenario** page of the dialog allows you to select, create, copy, delete and rename the scenarios associated with the model. You may also apply the selected scenario to the baseline data, which involves copying the series associated with any overridden variables in the selected scenario on top of the baseline values. Applying a scenario to the baseline is a way of committing to the edited values of the selected scenario making them a permanent part of the baseline case.

The **Scenario overrides** page provides a summary of variables which have been overridden in the selected scenario and equations which have been excluded. This is a useful way of seeing a complete list of all the changes which have been made to the scenario from the baseline case.

The **Aliasing** page allows you to examine the name aliasing rules associated with any scenario. The page displays the complete set of aliases that will be applied to the different types of variables in the model.

Although the scenario dialog lets you see all the settings for a scenario in one place, you will probably alter most scenario settings directly from the variable view instead. For both exogenous variables and add factors, you can select the variable from the variable view window, then use the right mouse button menu to call up the properties page for the variable. The override status of the variable can be adjusted using the **Use override** checkbox. Once a variable has been overridden, it will appear in red in the variable view.

Using Add Factors

Normally, when a model is solved deterministically, the equations of the model are solved so that each of the equations of the model is exactly satisfied. When a model is solved stochastically, random errors are added to each equation, but the random errors are still chosen so that their average value is zero.

If we have no information as to the errors in our stochastic equations that are likely to occur during the forecast period, then this behavior is appropriate. If, however, we have

additional information as to the sort of errors that are likely during our forecast period, then we may incorporate that information into the model using add factors.

The most common use for add factors is to provide a smoother transition from historical data into the forecast period. Typically, add factors will be used to compensate for a poor fit of one or more equations of the model near the end of the historical data, when we suspect this will persist into the forecast period. Add factors provide an ad hoc way of trying to adjust the results of the model without respecifying or reestimating the equations of the model.

In reality, an add factor is just an extra exogenous variable which is included in the selected equation in a particular way. EViews allows an add factor to take one of two forms. If our equation has the form

$$f(y_i) = f_i(y, x) \quad (23.3)$$

then we can provide an add factor for the equation intercept or residual by simply including the add factor at the end of the equation:

$$f(y_i) = f_i(y, x) + a \quad (23.4)$$

Alternatively we can provide an add factor for the endogenous variable of the model by including the add factor next to the endogenous variable:

$$f(y_i - a) = f_i(y, x) \quad (23.5)$$

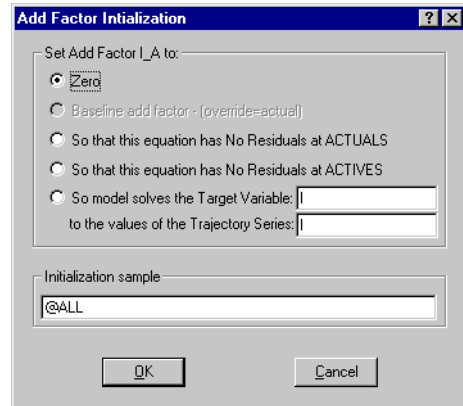
where the sign of the add factor is reversed so that it acts in the same direction as for the previous case.

If the endogenous variable appears by itself on the left hand side of the equal sign, then the two types of add factor are equivalent. If the endogenous variable is contained in an expression, for example, a log transformation, then this is no longer the case. Although the two add factors will have a similar effect, they will be expressed in different units with the former in the units of the residual of the equation, and the latter in the units of the endogenous variable of the equation.

There are two ways to include add factors. The easiest way is to go to the equation view of the model, then double click on the equation in which you would like to include an add factor. When the equation properties dialog appears, switch to the **Add Factors** tab. In the **Factor type** box, select whether you would like an intercept or an endogenous variable shift add factor. A message box will prompt for whether you would like to create a series in the workfile to hold the add factor values. Click on **Yes** to create the series.

The series will initially be filled with NAs. You can initialize the add factor using one of several methods by clicking on the **Initialize Add Factor** button. A dialog box will come up offering the following options:

- **Zero:** set the add factor to zero for every period.
- **So that this equation has no residuals at actuals:** set the values of the add factor so that the equation is exactly satisfied without error when the variables of the model are set to the values contained in the actual series (typically the historical data).
- **So that this equation has no residuals at actives:** set the values of the add factor so that the equation is exactly satisfied without error when the variables of the model are set to the values contained in the endogenous and exogenous series associated with the active scenario.
- **So model solves the target variable to the values of the trajectory series:** set the values of the add factor so that an endogenous variable of the model follows a particular target path when the model is solved.



You can also change the sample over which you would like the add factor to be initialized by modifying the **Initialization sample** box. Click on OK to accept the settings.

Once an add factor has been added to an equation, it will appear in the variable view of the model as an additional variable. If an add factor is present in any scenario, then it must be present in every scenario, although the values of the add factor can be overridden for a particular scenario in the same way as for an exogenous variable.

The second way to handle add factors is to assign, initialize or override them for all the equations in the model at the same time using the **Procs/Add Factors** menu from the model window. For example, to create a complete set of add factors that make the model solve to actual values over history, we can use **Add Factors/Equation Assignment...** to create add factors for every equation, then use **Add Factors/Set Values...** to set the add factors so that all the equations have no residuals at the actual values.

When solving a model with an add factor, any missing values in the add factor will be treated as zeros.

Solving the Model

Once the model specification is complete, you can solve the model. EViews can perform both deterministic and stochastic simulations.

A deterministic simulation consists of the following steps:

- The block structure of the model is analyzed.
- The variables in the model are bound to series in the workfile, according to the override settings and name aliasing rules of the scenario that is being solved. If an endogenous variable is being tracked and a series does not already exist in the workfile, a new series will be created. If an endogenous variable is not being tracked, a temporary series will be created to hold the results.
- The equations of the model are solved for each observation in the solution sample, using an iterative algorithm to compute values for the endogenous variables.
- Any temporary series which were created are deleted.
- The results are rounded to their final values.

A stochastic simulation follows a similar sequence, with the following differences:

- When binding the variables, a temporary series is created for every endogenous variable in the model. Additional series in the workfile are used to hold the statistics for the tracked endogenous variables. If bounds are being calculated, extra memory is allocated as working space for intermediate results.
- The model is solved repeatedly for different draws of the stochastic components of the model. If coefficient uncertainty is included in the model, then a new set of coefficients is drawn before each repetition. During the repetition, errors are generated for each observation in accordance with the residual uncertainty and the exogenous variable uncertainty in the model. At the end of each repetition, the statistics for the tracked endogenous variables are updated to reflect the additional results.
- If a comparison is being performed with an alternate scenario, then the same set of random residuals and exogenous variable shocks are applied to both scenarios during each repetition. This is done so that the deviation between the two is based only on differences in the exogenous and excluded variables, not on differences in random errors.

Models Containing Future Values

So far, we have assumed that the structure of the model allows us to solve each period of the model in sequence. This will not be true in the case where the equations of the model contain future (as well as past) values of the endogenous variables.

Consider a model where the equations have the form:

$$F(y(-\text{maxlag}), \dots, y(-1), y, y(1), \dots, y(\text{maxlead}), x) = 0 \quad (23.6)$$

where F is the complete set of equations of the model, y is a vector of all the endogenous variables, x is a vector of all the exogenous variables, and the parentheses follow the usual EViews syntax to indicate leads and lags.

Since solving the model for any particular period requires both past and future values of the endogenous variables, it is not possible to solve the model recursively in one pass. Instead, the equations from all the periods across which the model will be solved must be treated as a simultaneous system, and we will require terminal as well as initial conditions. For example, in the case with a single lead and a single lag and a sample that runs from s to t , we must effectively solve the entire stacked system:

$$\begin{aligned} F(y_{s-1}, y_s, y_{s+1}, x) &= 0 \\ F(y_s, y_{s+1}, y_{s+2}, x) &= 0 \\ F(y_{s+1}, y_{s+2}, y_{s+3}, x) &= 0 \\ &\dots \\ F(y_{t-2}, y_{t-1}, y_t, x) &= 0 \\ F(y_{t-1}, y_t, y_{t+1}, x) &= 0 \end{aligned} \quad (23.7)$$

where the unknowns are y_s, y_{s+1}, \dots, y_t the initial conditions are given by y_{s-1} and the terminal conditions are used to determine y_{t+1} . Note that if the leads or lags extend more than one period, we will require multiple periods of initial or terminal conditions.

To solve models such as these, EViews applies a Gauss-Seidel iterative scheme across all the observations of the sample. Roughly speaking, this involves looping repeatedly through every observation in the forecast sample, at each observation solving the model while treating the past and future values as fixed, where the loop is repeated until changes in the values of the endogenous variables between successive iterations become less than a specified tolerance.

This method is often referred to as the Fair-Taylor method, although the Fair-Taylor algorithm includes a particular handling of terminal conditions (the extended path method) that is slightly different from the options provided by EViews. When solving the model, EViews allows the user to specify fixed end conditions by providing values for the endogenous variables beyond the end of the forecast sample, or to determine the terminal conditions endogenously by adding extra equations for the terminal periods which impose either a constant level, a linear trend, or a constant growth rate on the endogenous variables for values beyond the end of the forecast period.

Although this method is not guaranteed to converge, failure to converge is often a sign of the instability which results when the influence of the past or the future on the present does not die out as the length of time considered is increased. Such instability is often undesirable for other reasons and may indicate a poorly specified model.

Model Consistent Expectations

One source of models in which future values of endogenous variables may appear in equations are models of economic behavior in which expectations of future periods influence the decisions made in the current period. For example, when negotiating long term wage contracts, employers and employees must consider expected changes in prices over the duration of the contract. Similarly, when choosing to hold a security denominated in foreign currency, an individual must consider how the exchange rate is expected to change over the time that they hold the security.

Although the way that individuals form expectations is obviously complex, if the model being considered accurately captures the structure of the problem, we might expect the expectations of individuals to be broadly consistent with the outcomes predicted by the model. In the absence of any other information, we may choose to make this relationship hold exactly. Expectations of this form are often referred to as *model consistent expectations*.

If we assume that there is no uncertainty in the model, imposing model consistent expectations simply involves replacing any expectations that appear in the model with the future values predicted by the model. In EViews, we can simply write out the expectation terms that appear in equations using the lead operator. A deterministic simulation of the model can then be run using EViews ability to solve models with equations which contain future values of the endogenous variables.

When we add uncertainty to the model, the situation becomes more complex. In this case, instead of the expectations of agents being set equal to the single deterministic outcome predicted by the model, the expectations of agents should be calculated based on the entire distribution of stochastic outcomes predicted by the model. To run a stochastic simulation of a model involving expectations would require a procedure like the following:

1. Take an initial guess as to a path for expectations over the forecast period (for example, by calculating a solution for the expectations in the deterministic case)
2. Run a large number of stochastic repetitions of the model holding these expectations constant, calculating the mean paths of the endogenous variables over the entire set of outcomes.
3. Test if the mean paths of the endogenous variables are equal to the current guess of expectations within some tolerance. If not, replace the current guess of expectations with the mean of the endogenous variables obtained in step 2, and return to step 2.

At present, EViews does not have built in functionality for automatically carrying out this procedure. Because of this, EViews will not perform stochastic simulations if your model contains equations involving future values of endogenous variables. We hope to add this functionality to future revisions of EViews.

Basic Options

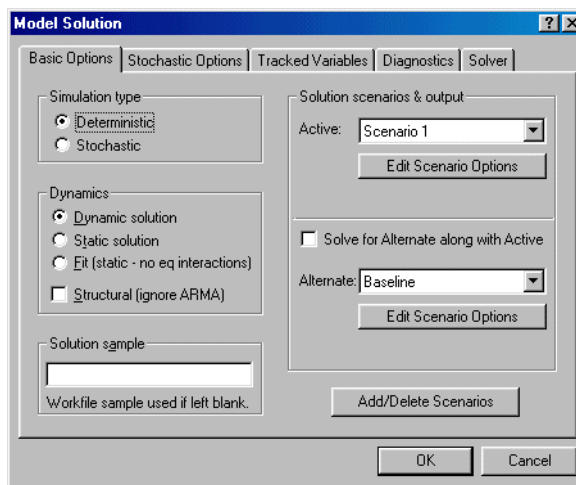
To begin solving a model, you can use **Procs/Solve Model...** or you can simply click on the **Solve** button on the model toolbar. EViews will display a tabbed dialog containing the solution options.

The basic options page contains the most important options for the simulation. While the options on other pages can often be left at their default values, the options on this page will need to be set appropriately for the task you are trying to perform.

At the top left, the **Simulation type** box allows you to determine whether the model should be simulated deterministically or stochastically. In a deterministic simulation, all equations in the model are solved so that they hold without error during the simulation period, all coefficients are held fixed at their point estimates, and all exogenous variables are held constant. This results in a single path for the endogenous variables which can be evaluated by solving the model once.

In a stochastic simulation, the equations of the model are solved so that they have residuals which match to randomly drawn errors, and, optionally, the coefficients and exogenous variables of the model are also varied randomly. In this case, the model generates a distribution of outcomes for the endogenous variables in every period. We approximate this distribution by solving the model many times using different draws for the random components in the model then calculating statistics over all the different outcomes.

Typically, you will first analyze a model using deterministic simulation, and then later proceed to stochastic simulation to get an idea of the sensitivity of the results to various sorts of error. You should generally make sure that the model can be solved deterministically and is behaving as expected before trying a stochastic simulation, since stochastic simulation can be very time consuming.



The next option is the **Dynamics** box. This option determines how EViews uses historical data for the endogenous variables when solving the model:

- When **Dynamic solution** is chosen, only values of the endogenous variables from before the solution sample are used when forming the forecast. Lagged endogenous variables and ARMA terms in the model are calculated using the solutions calculated in previous periods, not from actual historical values. A dynamic solution is typically the correct method to use when forecasting values several periods into the future (a multi-step forecast), or evaluating how a multi-step forecast would have performed historically.
- When **Static solution** is chosen, values of the endogenous variables up to the previous period are used each time the model is solved. Lagged endogenous variables and ARMA terms in the model are based on actual values of the endogenous variables. A static solution is typically used to produce a set of one-step ahead forecasts over the historical data so as to examine the historical fit of the model. A static solution cannot be used to predict more than one observation into the future.
- When the **Fit** option is selected, values of the endogenous variables for the current period are used when the model is solved. All endogenous variables except the one variable for the equation being evaluated are replaced by their actual values. The fit option can be used to examine the fit of each of the equations in the model when considered separately, ignoring their interdependence in the model. The fit option can only be used for periods when historical values are available for all the endogenous variables.

In addition to these options, the **Structural** checkbox gives you the option of ignoring any ARMA specifications that appear in the equations of the model.

At the bottom left of the dialog is a box for the solution sample. The solution sample is the set of observations over which the model will be solved. Unlike in some other EViews procedures, the solution sample will not be contracted automatically to exclude missing data. For the solution to produce results, data must be available for all exogenous variables over the course of the solution sample. If you are carrying out a static solution or a fit, data must also be available for all endogenous variables during the solution sample. If you are performing a dynamic solution, only pre-sample values are needed to initialize any lagged endogenous or ARMA terms in the model.

On the right-hand side of the dialog are controls for selecting which scenarios we would like to solve. By clicking on one of the **Edit Scenario Options** buttons, you can quickly examine the settings of the selected scenario. The option **Solve for Alternate along with Active** should be used mainly in a stochastic setting, where the two scenarios must be solved together to ensure that the same set of random shocks is used in both cases. Whenever two models are solved together stochastically, a set of series will also be created con-

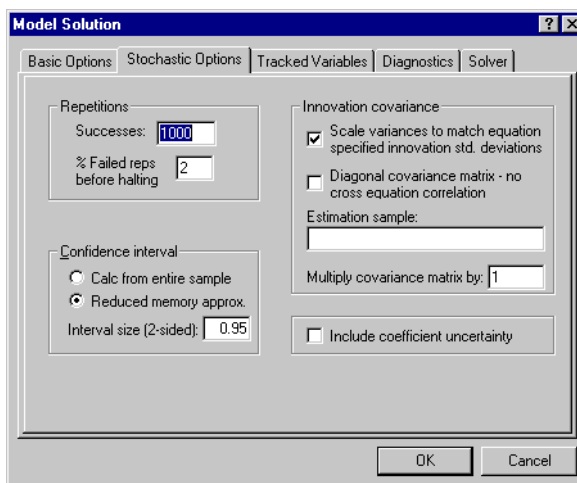
taining the deviations between the scenarios (this is necessary because in a non-linear model, the difference of the means need not equal the mean of the differences).

When stochastic simulation has been selected, additional checkboxes are available for selecting which statistics you would like to calculate for your tracked endogenous variables. A series for the mean will always be calculated. You can also optionally collect series for the standard deviation or quantile bounds. Quantile bounds require considerable working memory, but are useful if you suspect that your endogenous variables may have skewed distributions or fat tails. If standard deviations or quantile bounds are chosen for either the active or alternate scenario, they will also be calculated for the deviations series.

Stochastic Options

The stochastic options page contains settings used during stochastic simulation. In many cases, you can leave these options at their default settings.

The **Repetitions** box, in the top left corner of the dialog, allows you to set the number of repetitions that will be performed during the stochastic simulation. A higher number of repetitions will reduce the sampling variation in the statistics being calculated, but will take more time. The default value of one thousand repetitions is generally adequate to get a good idea of the underlying values, although there may still be some random variation visible between adjacent observations.



Also in the repetitions box is a field labeled **% Failed reps before halting**. Failed repetitions typically result from random errors driving the model into a region in which it is not defined, for example where the model is forced to take the log or square root of a negative number. When a repetition fails, EViews will discard any partial results from that repetition, then check whether the total number of failures exceeds the threshold set in the **% Failed reps before halting** box. The simulation continues until either this threshold is exceeded, or the target number of successful repetitions is met.

Note, however, that even one failed repetition indicates that care should be taken when interpreting the simulation results, since it indicates that the model is ill-defined for some possible draws of the random components. Simply discarding these extreme values may

create misleading results, particularly when the tails of the distribution are used to measure the error bounds of the system.

The **Confidence interval** box sets options for how confidence intervals should be calculated, assuming they have been selected. The **Calc from entire sample** option uses the sample quantile as an estimate of the quantile of the underlying distribution. This involves storing complete tails for the observed outcomes. This can be very memory intensive since the memory used increases linearly in the number of repetitions. The **Reduced memory approx** option uses an updating algorithm due to Jain and Chlamtac (1985). This requires much less memory overall, and the amount used is independent of the number of repetitions. The updating algorithm should provide a reasonable estimate of the tails of the underlying distribution as long as the number of repetitions is not too small.

The **Interval size (2 sided)** box lets you select the size of the confidence interval given by the upper and lower bounds. The default size of 0.95 provides a 95% confidence interval with a weight of 2.5% in each tail. If, instead, you would like to calculate the interquartile range for the simulation results, you should input 0.5 to obtain a confidence interval with bounds at the 25% and 75% quantiles.

The **Innovation covariance** box on the right side of the dialog determines how the innovations to stochastic equations will be generated. At each observation of a stochastic simulation, a set of independent random numbers are drawn from the standard normal distribution, then these numbers are scaled to match the desired variance-covariance matrix of the system. In the general case, this involves multiplying the vector of random numbers by the Cholesky factor of the covariance matrix. If the matrix is diagonal, this reduces to multiplying each random number by its desired standard deviation.

The **Scale variances to match equation specified standard deviations** box lets you determine how the variances of the residuals in the equations are determined. If the box is not checked, the variances are calculated from the model equation residuals. If the box is checked, then any equation that contains a specified standard deviation will use that number instead (see [page 622](#) for details on how to specify a standard deviation from the equation properties page). Note that the sample used for estimation in a linked equation may differ from the sample used when estimating the variances of the model residuals.

The **Diagonal covariance matrix** box lets you determine how the off diagonal elements of the covariance matrix are determined. If the box is checked, the off diagonal elements are set to zero. If the box is not checked, the off diagonal elements are set so that the correlation of the random draws matches the correlation of the observed equation residuals. If the variances are being scaled, this will involve rescaling the estimated covariances so that the correlations are maintained.

The **Estimation sample** box allows you to specify the set of observations that will be used when estimating the variance-covariance matrix of the model residuals. By default, EViews will use the default workfile sample.

The **Multiply covariance matrix** field allows you to set an overall scale factor to be applied to the entire covariance matrix. This can be useful for seeing how the stochastic behavior of the model changes as levels of random variation are applied which are different from those that were observed historically, or as a means of trouble-shooting the model by reducing the overall level of random variation if the model behaves badly.

The **Include coefficient uncertainty** field at the bottom right of the dialog specifies whether estimated coefficients in linked equations should be varied randomly during a stochastic simulation. If this option is selected, coefficients are redrawn randomly once at the beginning of each repetition. This provides a way of incorporating uncertainty surrounding the true values of the coefficients into variation in our forecast results.

Note that the dynamic behavior of a model may be changed considerably when the coefficients in the model are varied random. A model which is stable may become unstable, or a model which converges exponentially may develop cyclical oscillations. One consequence of this is that the standard errors from a stochastic simulation of a single equation may vary from the standard errors obtained when the same equation is forecast using the EViews equation object. This is because the equation object uses an analytic approach to calculating standard errors based on a local linear approximation that effectively imposes stationarity on the equation.

Tracked Variables

The Tracked Variables page of the dialog lets you examine and modify which endogenous variables are being tracked by the model. When a variable is tracked, the results for that variable are saved in a series in the workfile after the simulation is complete. No results are saved for variables that are not tracked.

Tracking is most useful when working with large models, where keeping the results for every endogenous variable in the model would clutter the workfile and use up too much memory.

By default, all variables are tracked. You can switch on selective tracking using the radio button at the top of the dialog. Once selective tracking is selected, you can type in variable names in the dialog below, or use the properties dialog for the endogenous variable to switch tracking on and off.

You can also see which variables are currently being tracked using the variable view, since the names of tracked variables appear in blue.

Diagnostics

The Diagnostics dialog page lets you set options to control the display of intermediate output. This can be useful if you are having problems getting your model to solve.

When the **Display detailed messages** box is checked, extra output will be produced in the solution messages window as the model is solved.

The traced variables list lets you specify a list of variables for which intermediate values will be stored during the iterations of the solution process. These results can be examined by switching to the **Trace Output** view after the model is complete. Tracing intermediate values may give you some idea of where to look for problems when a model is generating errors or failing to converge.

Solver

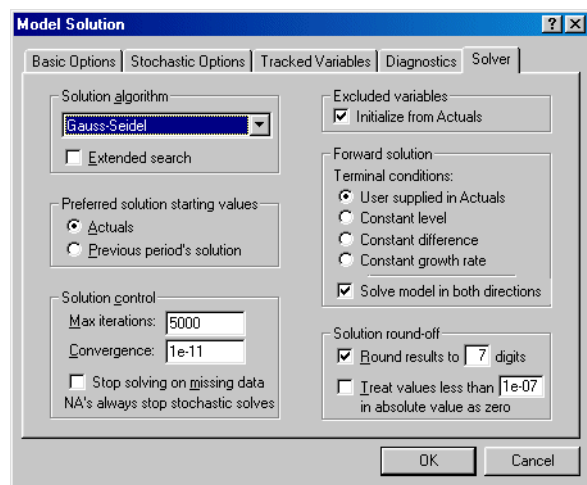
The Solver dialog page sets options relating to the non-linear equation solver which is applied to the model.

The **Solution algorithm** box lets you select the algorithm that will be used to solve the model for a single period. The following choices are available:

- **Gauss-Seidel:** the Gauss-Seidel algorithm is an iterative algorithm, where at each iteration we solve each equation in the model for the value of its associated endogenous variable, treating all other endogenous variables as fixed.

This algorithm requires little working memory and has fairly low computational costs, but requires the equation system to have certain stability properties for it to converge. Although it is easy to construct models that do not satisfy these properties, in practice, the algorithm generally performs well on most econometric models. If you are having difficulties with the algorithm, you might like to try reordering the equations, or rewriting the equations to change the assignment of endogenous variables, since these changes can affect the stability of the Gauss-Seidel iterations.

- **Newton:** Newton's method is also an iterative method, where at each iteration we take a linear approximation to the model, then solve the linear system to find a root



of the model. This algorithm can handle a wider class of problems than Gauss-Seidel, but requires considerably more working memory and has a much greater computational cost when applied to large models. Newton's method is invariant to equation reordering or rewriting.

Note that even if Newton's method is selected for solving within each period of the model, a Gauss-Seidel type method is used between all the periods if the model requires iterative forward solution. See [“Models Containing Future Values” on page 629](#).

The **Excluded variables/Initialize from Actuals** checkbox controls where EViews takes values for excluded variables. By default, this box is checked and all excluded observations for solved endogenous variables (both in the solution sample and pre-solution observations) are initialized to the actual values of the endogenous variables prior to the start of a model solution. If this box is unchecked, EViews will initialize the excluded variables with values from the solution series (aliased series), so that you may set the values manually without editing the original series.

The **Extended search** checkbox tells the solver to try alternative step sizes when searching for new values for the endogenous variables during an iteration. This improves the chances of convergence, but will generally increase the time taken to solve the model. If your model is having difficulty converging, you may like to try this option.

The **Preferred solution starting values** section lets you select the values to be used as starting values in the iterative procedure. When **Actuals** is selected, EViews will first try to use values contained in the actuals series as starting values. If these are not available, EViews will try to use the values solved for in the previous period. If these are not available, EViews will default to using arbitrary starting values of 0.1. When **Previous period's solution** is selected, the order is changed so that the previous periods values are tried first, and only if they are not available, are the actuals used.

The **Solution control** section allows you to set termination options for the solver. **Max iterations** sets the maximum number of iterations that the solver will carry out before aborting. **Convergence** sets the threshold for the convergence test. If the largest relative change between iterations of any endogenous variable has an absolute value less than this threshold, then the solution is considered to have converged. **Stop on missing data** means that the solver should stop as soon as one or more exogenous (or lagged endogenous) variables is not available. If this option is not checked, the solver will proceed to subsequent periods, storing NAs for this period's results.

The **Forward solution** section allows you to adjust options that affect how the model is solved when one or more equations in the model contain future (forward) values of the endogenous variables. The **Terminal conditions** section lets you specify how the values of the endogenous variables are determined for leads that extend past the end of the forecast period. If **User supplied in Actuals** is selected, the values contained in the Actuals series

after the end of the forecast sample will be used as fixed terminal values. If no values are available, the solver will be unable to proceed. If **Constant level** is selected, the terminal values are determined endogenously by adding the condition to the model that the values of the endogenous variables are constant over the post-forecast period at the same level as the final forecasted values ($y_t = y_{t-1}$ for $t = T, T + 1, \dots, T + k - 1$), where T is the first observation past the end of the forecast sample, and k is the maximum lead in the model). This option may be a good choice if the model converges to a stationary state. If **Constant difference** is selected, the terminal values are determined endogenously by adding the condition that the values of the endogenous variables follow a linear trend over the post forecast period, with a slope given by the difference between the last two forecasted values

$$y_t - y_{t-1} = y_{t-1} - y_{t-2} \quad (23.8)$$

for $t = T, T + 1, \dots, T + k - 1$). This option may be a good choice if the model is in log form and tends to converge to a steady state. If **Constant growth rate** is selected, the terminal values are determined endogenously by adding the condition to the model that the endogenous variables grow exponentially over the post-forecast period, with the growth rate given by the growth between the final two forecasted values

$$(y_t - y_{t-1})/y_{t-1} = (y_{t-1} - y_{t-2})/y_{t-2} \quad (23.9)$$

for $t = T, T + 1, \dots, T + k - 1$). This latter option may be a good choice if the model tends to produce forecasts for the endogenous variables which converge to constant growth paths.

The **Solve in both directions** option affects how the solver loops over periods when calculating forward solutions. When the box is not checked, the solver always proceeds from the beginning to the end of the forecast period during the Gauss-Seidel iterations. When the box is checked, the solver alternates between moving forwards and moving backwards through the forecast period. The two approaches will generally converge at slightly different rates depending on the level of forward or backward persistence in the model. You should choose whichever setting results in a lower iteration count for your particular model.

The **Solution round-off** section of the dialog controls how the results are rounded after convergence has been achieved. Because the solution algorithms are iterative and provide only approximate results to a specified tolerance, small variations can occur when comparing solutions from models, even when the results should be identical in theory. Rounding can be used to remove some of this minor variation so that results will be more consistent. The default settings will normally be adequate, but if your model has one or more endogenous variables of very small magnitude, you will need to switch off the rounding to zero or rescale the variables so that their solutions are farther from zero.

Solve Control for Target

Normally, when solving a model, we start with a set of known values for our exogenous variables, then solve for the unknown values of the endogenous variables of the model. If we would like an endogenous variable in our model to follow a particular path, we can solve the model repeatedly for different values of the exogenous variables, changing the values until the path we want for the endogenous variable is produced. For example, in a macroeconomic model, we may be interested in examining what value of the personal tax rate would be needed in each period to produce a balanced budget over the forecast horizon.

The problem with carrying out this procedure by hand is that the interactions between variables in the model make it difficult to guess the correct values for the exogenous variables. It will often require many attempts to find the values that solve the model to give the desired results.

To make this process easier, EViews provides a special procedure for solving a model which automatically searches for the unknown values. Simply create a series in the workfile which contains the values you would like the endogenous variable to achieve, then select **Procs/Solve Control for Target...** from the menus. Enter the name of the exogenous variable you would like to modify in the **Control Variable** box, the name of the endogenous variable which you are targeting in the **Target Variable** box, and the name of the workfile series which contains the target values in the **Trajectory Variable** box. Set the sample to the range for you would like to solve, then click on **OK**.

The procedure may take some time to complete, since it involves repeatedly solving the model to search for the desired solution. It is also possible for the procedure to fail if it cannot find a value of the exogenous variable for which the endogenous variable solves to the target value. If the procedure fails, you may like to try moving the trajectory series closer to values that you are sure the model can achieve.

Working with the Model Data

When working with a model, much of your time will be spent viewing and modifying the data associated with the model. Before solving the model, you will edit the paths of your exogenous variables or add factors during the forecast period. After solving the model, you will use graphs or tables of the endogenous variables to evaluate the results. Because there is a large amount of data associated with a model, you will also spend time simply managing the data.

Since all the data associated with a model is stored inside standard series in the workfile, you can use all of the usual tools in EViews to work with the data of your model. However, it is often more convenient to work directly from the model window.

Although there are some differences in details, working with the model data generally involves following the same basic steps. You will typically first use the variable view to select the set of variables you would like to work with, then use either the right mouse button menu or the model procedure menu to select the operation to perform.

Because there may be several series in the workfile associated with each variable in the model, you will then need to select the types of series with which you wish to work. The following types will generally be available:

- **Actuals:** the workfile series with the same name as the variable name. This will typically hold the historical data for the endogenous variables, and the historical data and baseline forecast for the exogenous variables.
- **Active:** the workfile series that is used when solving the active scenario. For endogenous variables, this will be the series with a name consisting of the variable name followed by the scenario extension. For exogenous variables, the actual series will be used unless it has been overridden. In this case, the exogenous variable will also be the workfile series formed by appending the scenario extension to the variable name.
- **Alternate:** the workfile series that is used when solving the alternate scenario. The rules are the same as for active.

In the following sections, we discuss how different operations can be performed on the model data from within the variable view.

Editing Data

The easiest way to make simple changes to the data associated with a model is to open a series or group spreadsheet window containing the data, then edit the data by hand.

To open a series window from within the model, simply select the variable using the mouse in the variable view, then use the right mouse button menu to choose **Open selected series...**, followed by **Actuals**, **Active Scenario** or **Alternate Scenario**. If you select several series before using the option, an unnamed group object will be created to hold all the series.

To edit the data, click the **Edit +/-** button to make sure the spreadsheet is in edit mode. You can either edit the data directly in levels or use the **Units** button to work with a transformed form of the data, such as the differences or percentage changes.

To create a group which allows you to edit more than one of the series associated with a variable at the same time, you can use the **Make Group/Table** procedure discussed below to create a dated data table, then switch the group to spreadsheet view to edit the data.

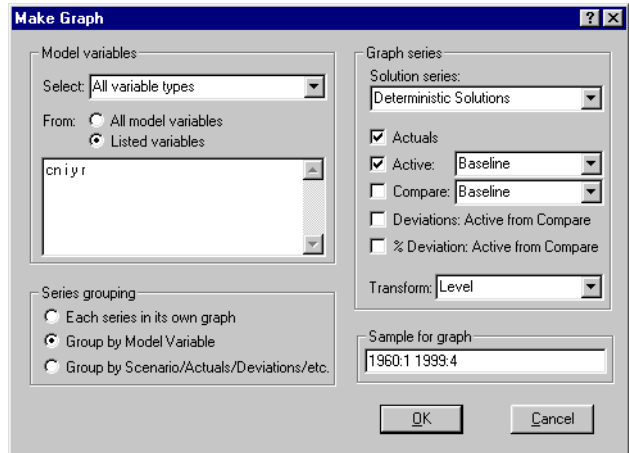
More complicated changes to the data may require using a `genr` command to calculate the series by specifying an expression. Click the **Genr** button from the series window toolbar to call up the dialog, then type in the expression to generate values for the series and set the workfile sample to the range of values you would like to modify.

Displaying Data

The EViews model object provides two main forms in which to display data: as a graph or as a table. Both of these can be generated easily from the model window.

From the variable view, select the variables you wish to display, then use the right mouse button menu or the main menu to select **Procs** and then **Make Group/Table** or **Make Graph**.

The dialogs for the two procs are almost identical. Here we see the **Make Graph** dialog. We saw this dialog earlier in our macro model example. The majority of fields in the dialog control which series



you would like the table or graph to contain. At the top left of the graph is the **Model Variables** box, which is used to select the set of variables to place in the graph. By default, the table or graph will contain the variables that are currently selected in the variable view. You can expand this to include all model variables, or add or remove particular variables from the list of selected variables using the radio buttons and text box labeled **From**. You can also restrict the set of variables chosen according to variable type using the list box next to **Select**. By combining these fields, it is easy to select sets of variables such as all of the endogenous variables of the model, or all of the overridden variables.

Once the set of variables has been determined, it is necessary to map the variable names into the names of series in the workfile. This typically involves adding an extension to each name according to which scenario the data is from and the type of data contained in the series. The options affecting this are contained in the **Graph series** (if you are making a graph) or **Series types** (if you are making a group/table) box at the right of the dialog.

The **Solution series** box lets you choose which solution results you would like to examine when working with endogenous variables. You can choose from a variety of series generated during deterministic or stochastic simulations.

The series of checkboxes below determine which scenarios you would like to display in the graphs, as well as whether you would like to calculate deviations between various scenarios. You can choose to display the actual series, the series from the active scenario, or the series from an alternate scenario (labeled “Compare”). You can also display either the difference between the active and alternate scenario (labeled “Deviations: Active from Compare”), or the ratio between the active and alternate scenario in percentage terms (labeled “% Deviation: Active from Compare”).

The final field in the **Graph series** or **Series types** box is the **Transform** listbox. This lets you apply a transformation to the data similar to the **Transform** button in the series spreadsheet.

While the deviations and units options allow you to present a variety of transformations of your data, in some cases you may be interested in other transformations that are not directly available. Similarly, in a stochastic simulation, you may be interested in examining standard errors or confidence bounds on the transformed series, which will not be available when you apply transformations to the data after the simulation is complete. In either of these cases, it may be worth adding an identity to the model that generates the series you are interested in examining as part of the model solution.

For example, if your model contains a variable GDP, you may like to add a new equation to the model to calculate the percentage change of GDP:

$$pgdp = @pch(gdp)$$

After you have solved the model, you can use the variable PGDP to examine the percentage change in GDP, including examining the error bounds from a stochastic simulation. Note that the cost of adding such identities is relatively low, since EViews will place all such identities in a final recursive block which is evaluated only once after the main endogenous variables have already been solved.

The remaining option, at the bottom left of the dialog, lets you determine how the series will be grouped in the output. The options are slightly different for tables and graphs. For tables, you can choose to either place all series associated with the same model variable together, or to place each series of the same series type together. For graphs, you have the same two choices, and one additional choice, which is to place every series in its own graph.

In the graph dialog, you also have the option of setting a sample for the graph. This is often useful when you are plotting forecast results since it allows you to choose the amount of historical data to display in the graph prior to the forecast results. By default, the sample is set to the workfile sample.

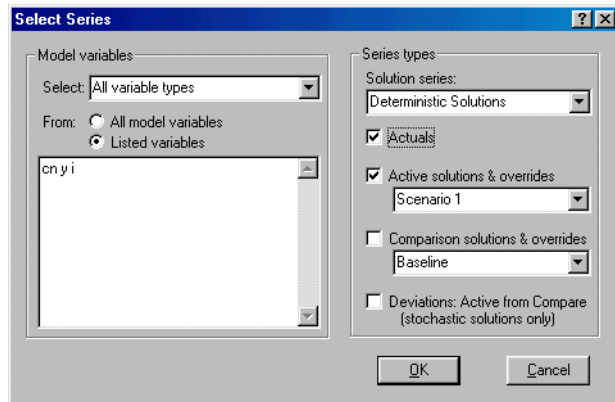
When you have finished setting the options, simply click on **OK** to create the new table or graph. All of EViews usual editing features are available to modify the table or graph for final presentation.

Managing Data

When working with a model, you will often create many series in the workfile for each variable, each containing different types of results or the data from different scenarios. The model object provides a number of tools to help you manage these series, allowing you to perform copy, fetch, store and delete operations directly from within the model.

Because the series names are related to the variable names in a consistent way, management tasks can often also be performed from outside the model by using the pattern matching features available in EViews commands (see [Appendix C, “Wildcards”, on page 657](#)).

The data management operations from within the model window proceed very similarly to the data display operations. First, select the variables you would like to work with from the variable view, then choose **Copy**, **Store series...**, **Fetch series...** or **Delete series...** from the right mouse button menu or the object procedures menu. A dialog will appear, similar to the one used when making a table or graph.



In the same way as for the table and graph dialogs, the left side of the dialog is used to choose which of the model variables to work with, while the right side of the dialog is used to select one or more series associated with each variable. Most of the choices are exactly the same as for graphs and tables. One significant difference is that the checkboxes for active and comparison scenarios include exogenous variables only if they have been overridden in the scenario. Unlike when displaying or editing the data, if an exogenous variable has not been overridden, the actual series will not be included in its place. The only way to store, fetch or delete any actual series is to use the **Actuals** checkbox.

After clicking on **OK**, you will receive the usual prompts for the store, fetch and delete operations. You can proceed as usual.

Commands

To create a model using commands, you should use a `model` declaration statement, and the `append` command to add equation lines:

```
model macro
macro.append cs=10+0.8*y(-1)
macro.append i=0.7*(y(-1)-y(-2))
macro.append y=cs+i+g
```

You can assign and initialize intercept shift add factors to all equations in the model using the commands

```
macro.addassign(i) @all
macro.addinit(v=z) @all
```

Here, the add factors are all initialized to zero.

To solve the model, use the `solve` command

```
macro.solve(m=500,e)
```

The solution options set the maximum number of iterations to 500, and instruct the solver to use Gauss-Seidel with extended search.

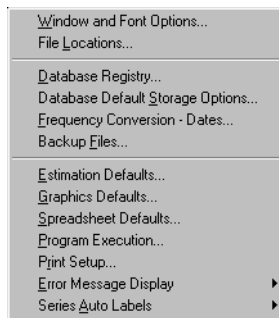
For additional details, see [“Model” on page 32](#) of the *Command and Programming Reference*.

Appendix A. Global Options

EViews employs default settings in most operations. In a wide variety of settings, the default settings may be user-specified. For example, you can set global defaults for everything from which font to use in table output to how graphics are exported to other programs to how to compute derivatives in estimation routines.

Setting Options

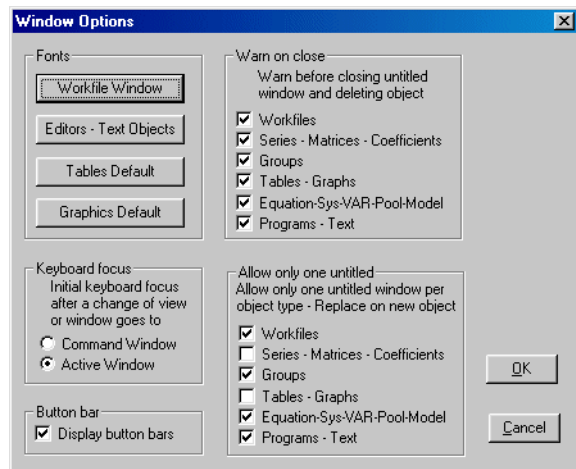
The **Options** menu in the main toolbar allows you to define the default behavior for many of the operations in EViews.



Window and Font Options

The window and font options control the display characteristics of various types of EViews output.

- The **Fonts** options allow you to change the default font styles and sizes for each type of window or object. Press the button of the type of object for which you want to change the font and select the new font from the Font dialog.
- **Keyboard Focus** controls where the keyboard cursor is placed when you change views or windows. The two choices make a difference



only when the window is an editable window such as the specification view of system objects.

- The **Button Bar** toggles on and off the display of the button bar in the object windows. The button bar provides shortcuts to frequently used items on the main menu. The main menu is always displayed, so that if you turn off the button bar, you can still perform all operations from the main menu.
- **Warn On Close** instructs EViews to provide a warning message when you close an untitled object. You may choose to set warnings for various object types. By default, EViews warns you that closing an unnamed object without naming it will cause the object to be deleted. Along with the warning, you will be given an opportunity to name the object. If you turn the warning off, closing an untitled window will automatically delete the object.
- **Allow Only One Untitled** specifies, for each object type, whether to allow multiple untitled objects to be opened at the same time. If only one is allowed, creating a new untitled object will cause any existing untitled object of that type to be deleted automatically.

Setting EViews to allow only one untitled object reduces the number of windows that you will see on your desktop. By default, only one untitled workfile, group, equation, system, model, program or text object can be open, but any number of untitled series, matrices, coefficients, tables, or graphs may be open.

If you allow only one untitled object, we strongly recommend that you select **Warn on Close**. With this option set, you will be given the opportunity to name and save an existing untitled object, otherwise the object will simply be deleted.

File Locations

This dialog allows you to set the default working directory, and the locations of the .INI file, database registry and alias map, and the temp directory. The dialog also reports the location of your EViews executable.

Data Registry / Database Default Storage Options

These entries control the default behavior of EViews when working with databases and moving data into and out of databases. You can also control whether data in databases are stored in single or double precision. For additional details see [“The Database Registry” on page 121](#) and [“Store, Fetch, and Copy of Group Objects” on page 117](#).

Frequency Conversion - Dates

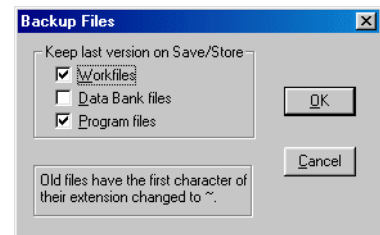
This dialog allows you to set the default frequency conversion method and the method of displaying dates.

The default frequency conversion method tells EViews how to convert data when you move data to lower or higher frequency workfiles. The frequency conversion methods are discussed in detail in [Chapter 6, “EViews Databases”](#), on page 107.

The default date display controls the text format for dates in tables and other output views. For daily and weekly data, you can set the default to American (Month/Day/Year) or you can switch to the European notation, where the day precedes the month.

Backup Files

You can choose whether to keep backup copies of workfiles, data bank files, and program files. The backup copy will have the same name as the file, but with the first character in the extension changed to `~`. For example, if you have a workfile named MYDATA.WF1, the backup file will be named MYDATA.~F1.



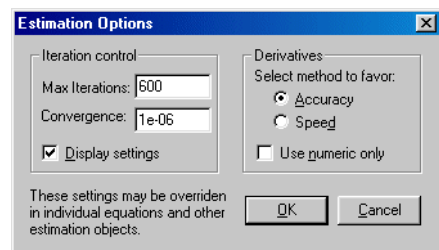
Estimation Defaults

You can set the global defaults for maximum number of iterations and convergence criterion. These settings will be used as the default settings in iterative estimation methods for equation, log likelihood, state space, and system objects. Note that previously estimated EViews objects that were estimated with the previous default settings will be unchanged, unless reestimated.

See [“Setting Estimation Options”](#) on page 666 for additional details.

Graphics Default

These options control how the graphs appear when you first create a graph. The options are explained in detail in [Chapter 10, “Graphs, Tables, and Text Objects”](#), beginning on page 243.



Spreadsheet Defaults

These options control the default spreadsheet view of series and group objects. This is the view that is displayed when the object is created.

For series spreadsheet views, the default is to display the series for the entire workfile range in a single column. You may change to display the series only for the current sample or to display the series in multiple columns. You also have the option whether to include the series label in the spreadsheet view or not and whether to display the view with edit mode enabled.

For group spreadsheet views, you have the option to display the spreadsheet in transposed form, with series arranged in rows instead of columns. You can also choose to open the view with edit mode enabled by default.

Note that these options only apply to newly created series or group objects. Once you modify the view using the **Edit**, **Smpl**, **Label**, **Wide**, or **Transpose** options, the object will be displayed accordingly.

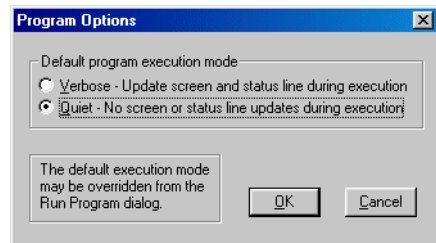
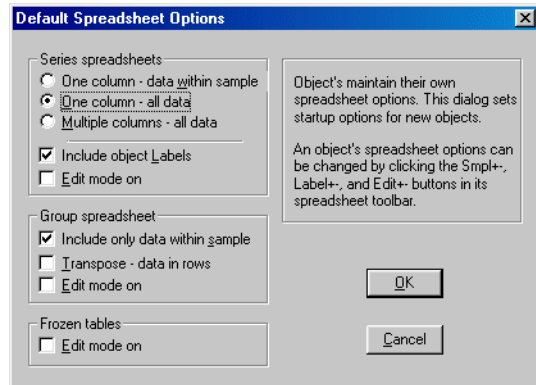
Program Execution

By default, EViews runs programs in verbose mode, listing commands in the status line as they are executed. Alternatively, you can switch to quiet mode, which suppresses this information. EViews will run faster in quiet mode since the status line display does not need to be updated.

The default may always be overridden from the Run Program dialog, or by using the option “Q” in the run statement, as in:

```
run(q) myprogram
```

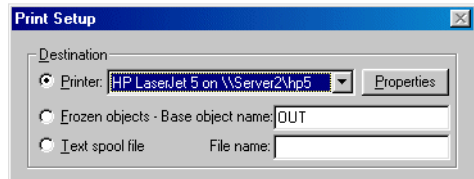
For details see [Chapter 6, “EViews Programming”](#), on page 85 of the *Command and Programming Reference*.



Print Setup

These options determine the default print behavior when you push the **Print** button or select **View/Print Selected** from the window toolbar.

The top of the Print Setup Options dialog gives you three choices for the destination of printed output. The default is to send the view or procedure output to the current Windows printer. The other two options redirect output into graph or table objects, or into text files.



Output Redirection

The **Frozen Output** option saves the views or output from procedures as frozen graphs, tables, or text objects in the workfile. No output is sent to the printer. Under this option, the **Print** button behaves as if it were the **Freeze** button.

If you choose **Frozen Output**, you must also supply a base name. Each print command will then create a frozen object within the active workfile, naming it with the base name followed by a number. For example, if you supply the base name of OUT, the first print command will generate a table or graph named OUT01, the second print command will generate OUT02, and so on. You can select some or all of the frozen objects in the workfile window, open them and copy-and-paste to your word processor, print them (after turning off output redirection), store them, or delete them.

The **Text File** option redirects printing of output to a file. You may use this option to create a log of the results from your EViews session. Note that all print commands of graph views or graph objects are still sent to the printer.

If you choose the **Text File** option, you must supply a file name. The file will be created in the default path with a .TXT extension, unless you explicitly specify the full path name. If a file with the same name already exists, EViews will warn you that the file will be overwritten. Any instruction to print tables or text will append the output to the end of the file.

Error Message Display

You can choose to have error messages displayed prominently in a box or, less conspicuously, on the status line at the bottom of the EViews window.

Series Auto Labels

You can elect to have EViews keep a history of the commands that created or modified a series as part of the series label. You can choose to turn this option off.

Appendix B. Date Formats

One of the more important features of EViews is the ability to process dated data. Once you determine the frequency of your data, EViews will use all available calendar information to organize and manage your data.

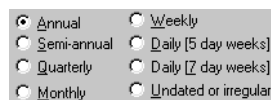
When using EViews, you must often specify dates. Dates are required when creating a workfile, specifying a sample, or specifying periods in hypothesis tests. This chapter discusses the format for dates in EViews.

Date Specification

The rules for specifying dates depend on the workfile frequency. Currently, there are eight frequencies available in EViews:

Annual

Years from 1930–2029 may be identified using either 2- or 4-digit identifiers (e.g. “97” or “1997”). All other years must be identified with full year identifiers (e.g. “1776”, “2040”, “9789” or “50234”). Note that since 2-digit identifiers are assumed to be in either the 20th or 21st century, EViews cannot handle dates prior to A.D. 100.



Semi-annual

Follow the year number with a colon and the number 1 for the first half of the year and the number 2 for the second half of the year. Examples: “1965:1”, “2010:2”.

Quarterly

Follow the year number with a colon and the quarter number. The quarter number should be 1, 2, 3 or 4. Examples: “1992:1”, “1965:4”, “2002:3”.

Monthly

Follow the year number with a colon and the month number. You may prefix the single digit month numbers with a zero, e.g. 01 or 04. Examples: “1956:04”, “1965:10”, “2010:12”.

Weekly and Daily

By default, the dates are specified in American notation as

mm/dd/yyyy

where mm, dd, yyyy stand for the month, date, and year numbers, respectively. The month and date numbers can be a single digit and the year numbers can be less than four digits. Examples: “3/1/1995”, “03/01/1995”, “10/01/2010”, “1/01/1965”.

With weekly data, the start of the week is identified by the first day specified in the workfile. All weeks will follow this convention. Thus, if you specify the start of your workfile as “10/15/1997”, a Wednesday, all of your weeks will begin on Wednesdays.

In any sample statement, if you specify a date in the middle of a week, EViews will view that date as the week containing the specified date. For example, for the Wednesday-based workfile above, if you enter:

```
smp1 10/17/1997 10/30/1997
```

EViews will display the sample as

```
10/15/1997 10/29/1997
```

since October 17 is the Friday of the week beginning on October 15 and October 30 is the Thursday of the week beginning on October 29.

You can use European notation by choosing **Options/Frequency Conversion–Dates...** from the main menu. Dates in European notation have the form

dd/mm/yyyy

For example, “8/10/97” in American notation indicates August 10, 1997, while in European notation it indicates October 8, 1997.

Undated or Irregular

Simply specify the observation number.

Implicit Dating

When you first create a workfile, you can specify the workfile range by indicating only the year number for any frequency (except undated). EViews will create a workfile that starts at the beginning of the specified year and that ends at the end of the specified year. For example, if you specify the workfile range as

```
60 90
```

EViews will create a workfile with range

1960–1990	for annual
1960:1–1990:2	for semi-annual
1960:1–1990:4	for quarterly
1960:01–1990:12	for monthly
1/01/1960–12/28/1990	for weekly
1/01/1960–12/31/1990	for daily
60–90	for undated or irregular

If you want to create a workfile starting or ending in the middle of a year, you should specify the full date.

Special Date Functions

There are three special functions that provide shortcuts for changing the workfile sample: “@all”, “@first”, and “@last”.

The keyword “@all” refers to the entire workfile range, while the keywords “@first” and “@last” refer to the first and last observation in the workfile, respectively. For example, suppose you have a workfile with range 1953:01–1996:12. Then to reset the sample to workfile range, you can use any of the following three commands:

```
sml @all
sml @first 1996:12
sml 53:1 @last
```

Sample range elements may contain mathematical expressions to create date offsets. This feature can be particularly useful in setting up a fixed width window of observations. For example, to set the sample to the last ten observations of the workfile range, you can use the command

```
sml @last-9 @last
```

Examples

```
1989:2
```

is the second half of 1989 (semi-annual) or the second quarter of 1989 (quarterly) or February 1989 (monthly).

```
1989:02
```

is the same as “1989:2”.

1992:4

is the fourth quarter of 1992 (quarterly) or April 1992 (monthly).

82

is the year 1982 (annual) or observation number 82 (undated).

1979:02

is the second half of 1979 (semi-annual) or the second quarter of 1979 (quarterly) or February 1979 (monthly).

1882

is the year 1882 (annual) or observation number 1882 (undated).

1968:5

is May 1968 (monthly).

1976:56

is an error since there is no month 56.

9/30/1996

is September 30, 1986 (daily) or the week starting on that day (weekly), if dates are in American notation. If dates are set to European, you should use “30/9/1996”.

2/30/1993

is an error since there is no February 30.

Appendix C. Wildcards

EViews supports the use of wildcard characters in a variety of situations where you need to enter a list of objects or a list of series. For example, you can use wildcards to:

- fetch, store, copy, rename or delete a list of objects
- specify a group object
- query a database by name or filter the workfile display

The following discussion describes some of the issues involved in the use of wildcard characters and expressions.

Wildcard Expressions

There are two wildcard characters: “*” and “?”. The wildcard character “*” matches zero or more characters in a name, and the wildcard “?” matches any single character in a name.

For example, you can use the wildcard expression “GD*” to refer to all objects whose names begin with the characters “GD”. The series GD, GDP, GD_F will be included in this list GGD, GPD will not. If you use the expression GD?, EViews will interpret this as a list of all objects with three character names beginning with the string “GD”: GDP and GD2 will be included, but GD, GD_2 will not.

You can instruct EViews to match a fixed number of characters by using as many “?” wildcard characters as necessary. For example, EViews will interpret “??GDP” as matching all objects with names that begin with any two characters followed by the string “GDP”. USGDP and F_GDP will be included but GDP, GGDP, GDPUS will not.

You can also mix the different wildcard characters in an expression. For example, you can use the expression “*GDP?” to refer to any object that ends with the string “GDP” and an arbitrary character. Both GDP_1, USGDP_F will be included.

Using Wildcard Expressions

Wildcard expressions may be used in filtering the workfile display (see [“Display Filter” on page 40](#)), in selected EViews commands, and in creating a group object.

The following commands support the use of wildcards: `show`, `store`, `fetch`, `copy`, `rename` and `delete`.

To create a group using wildcards, simply select **Object/New Objects/Group**, and enter the expression, EViews will first expand the expression, and then attempt to create a group using the corresponding list of series. For example, entering the list

```
y x*
```

will create a group comprised of Y and all series beginning with the letter X. Alternatively, you can enter the command

```
group g1 x* y?? c
```

defines a group G1, consisting of all of the series matching X*, and all series beginning with the letter Y followed by two arbitrary characters.

When making a group, EViews will only select series objects which match the given name pattern and will place these objects in the group.

Once created, these groups may be used anywhere that EViews takes a group as input. For example, if you have a series of dummy variables, DUM1, DUM2, DUM3, ..., DUM9, that you wish to enter in a regression, you can create a group containing the dummy variables, and then enter the group in the regression:

```
group gdum dum?  
equation eq1.ls y x z gdum
```

will run the appropriate regression. Note that we are assuming that the dummy variables are the only series objects which match the wildcard expression DUM?.

Source and Destination Patterns

When wildcards are used during copy and rename operations, a pattern must be provided for both the source and the destination. The destination pattern must always conform to the source pattern in that the number and order of wildcard characters must be exactly the same between the two. For example, the following patterns

Source Pattern	Destination Pattern
x*	y*
c	b
x*12?	yz*f?abc

conform to each other, while these patterns do not

Source Pattern	Destination Pattern
a*	b
*x	?y
x*y*	*x*y*

When using wildcards, the new destination name is formed by replacing each wildcard in the destination pattern by the characters from the source name that matched the corresponding wildcard in the source pattern. This allows you to both add and remove characters from the source name during the copy or rename process. Some examples should make this clear:

Source Pattern	Destination Pattern	Source Name	Destination Name
*_base	*_jan	x_base	x_jan
us_*	*	us_gdp	gdp
x?	x?f	x1	x1f
_	**f	us_gdp	usgdpf
??*f	??_*	usgdpf	us_gdp

Note, as shown in the second example, that a simple asterisk for the destination pattern will result in characters being removed from the source name when forming the destination name. To copy objects between containers preserving the existing name, either repeat the source pattern as the destination pattern

```
copy x* db1::x*
```

or omit the destination pattern entirely

```
copy x* db1::
```

Resolving Ambiguities

Note that an ambiguity can arise with wildcard characters since both “*” and “?” have multiple uses. The “*” character may be interpreted as either a multiplication operator or a wildcard character. The “?” character serves as both the single character wildcard and the pool cross section identifier.

Wildcard versus Multiplication

There is a potential for ambiguity in the use of the wildcard character “*”.

Suppose you have a workfile with the series X, X2, Y, XYA, XY2. There are then two interpretations of the wildcard expression “X*2”. The expression may be interpreted as an auto-

series representing X multiplied by 2. Alternatively, the expression may be used as a wildcard expression, referring to the series X^2 and XY^2 .

Note that there is only an ambiguity when the character is used in the middle of an expression, not when the wildcard character “*” is used at the beginning or end of an expression. EViews uses the following rules to determine the interpretation of ambiguous expressions:

- EViews first tests to see whether the expression represents a valid series expression. If so, the expression is treated as an auto-series. If it is not a valid series expression, then EViews will treat the “*” as a wildcard character. For example,

$y*x$
 $2*x$

are interpreted as auto-series, while

$*x$
 $x*a$

are interpreted as wildcard expressions.

- You can force EViews to treat “*” as a wildcard by preceding the character with another “*”. Thus, expressions containing “**” will always be treated as wildcard expressions. For example, the expression

$x**2$

unambiguously refers to all objects with names beginning with “X” and ending with “2”. Note that the use of “**” does *not* conflict with the EViews exponentiation operator “^”.

- You can instruct EViews to treat “*” as a series expression operator by enclosing the expression (or any subexpression) in parentheses. For example,

$(y*x)$

always refers to X times Y .

We *strongly* encourage you to resolve the ambiguity by using parentheses to denote series expressions, and double asterisks to denote wildcards (in the middle of expressions), whenever you create a group. This is especially true when group creation occurs in a program; otherwise the behavior of the program will be difficult to predict since it will change as the names of other objects in the workfile change.

Wildcard versus Pool Identifier

The “?” wildcard character is used both to match any single character in a pattern and as a place-holder for the cross-section identifier in pool objects.

EViews resolves this ambiguity by not allowing the wildcard interpretation of “?” in any expression involving a pool object or entered into a pool dialog. “?” is used exclusively as a cross-section identifier. For example, suppose that you have the pool object POOL1.

Then, the expression

```
pool1.est y? x? c
```

is a regression of the pool variable Y? on the pool variable X?, and

```
pool1.delete x?
```

deletes all of the series in the pool series X?. There is no ambiguity in the interpretation of these expressions since they both involve POOL1.

Similarly, when used apart from a pool object, the “?” is interpreted as a wildcard character. Thus,

```
delete x?
```

unambiguously deletes all of the series matching X?.

Appendix D. Estimation Algorithms and Options

EViews estimates the parameters of a wide variety of nonlinear models, from nonlinear least squares equations, to maximum likelihood models, to GMM specifications. These types of nonlinear estimation problems do not have closed form solutions and must be estimated using iterative methods. EViews also solves systems of non-linear equations. Again, there are no closed form solutions to these problems, and EViews must use an iterative method to obtain a solution.

Here, we provide details on the algorithms used by EViews in dealing with these problems, and the optional settings that we provide to allow you to control the estimation procedures.

Our discussion here is necessarily brief. For additional details we direct you to the quite readable discussions in Press, *et al.* (1992), Quandt (1983), Thisted (1988), and Amemiya (1983).

Optimization Algorithms

Before discussing EViews estimation options, it is useful to review briefly some basic optimization algorithms. Recall that the problem faced in non-linear estimation is to find the values of parameters θ that optimize (maximize or minimize) an objective function $F(\theta)$.

Iterative optimization algorithms work by taking an initial set of values for the parameters, say $\theta_{(0)}$, then performing calculations based on these values to obtain a better set of parameter values, $\theta_{(1)}$. This process is repeated for $\theta_{(2)}$, $\theta_{(3)}$ and so on until the objective function F no longer improves between iterations.

There are three main parts to the optimization process: (1) obtaining the initial parameter values, (2) updating the candidate parameter vector θ at each iteration, and (3) determining when we have reached the optimum.

If the objective function is globally concave so that there is a single maximum, any algorithm which improves the parameter vector at each iteration will eventually find this maximum (assuming that the size of the steps taken does not become negligible). If the objective function is not globally concave, different algorithms may find different local maxima, but all iterative algorithms will suffer from the same problem of being unable to tell apart a local and a global maximum.

The main thing that distinguishes different algorithms is how quickly they find the maximum. Unfortunately, there are no hard and fast rules. For some problems, one method may be faster, for other problems it may not. EViews provides different algorithms, and will often let you choose which method you would like to use.

The following sections outline these methods. The algorithms used in EViews may be broadly classified into three types: *second derivative* methods, *first derivative* methods, and *derivative free* methods. EViews' second derivative methods evaluate current parameter values and the first and second derivatives of the objective function for every observation. First derivative methods use only the first derivatives of the objective function during the iteration process. As the name suggests, derivative free methods do not compute derivatives.

Second Derivative Methods

For binary, ordered, censored, and count models, EViews can estimate the model using Newton-Raphson or *quadratic hill-climbing*.

Newton-Raphson

Candidate values for the parameters $\theta_{(1)}$ may be obtained using the method of Newton-Raphson by linearizing the first order conditions $\partial F/\partial\theta$ at the current parameter values, $\theta_{(i)}$:

$$\begin{aligned} g_{(i)} + H_{(i)}(\theta_{(i+1)} - \theta_{(i)}) &= 0 \\ \theta_{(i+1)} &= \theta_{(i)} - H_{(i)}^{-1}g_{(i)} \end{aligned} \tag{D.1}$$

where g is the gradient vector $\partial F/\partial\theta$, and H is the Hessian matrix $\partial^2 F/\partial\theta^2$.

If the function is quadratic, Newton-Raphson will find the maximum in a single iteration. If the function is not quadratic, the success of the algorithm will depend on how well a local quadratic approximation captures the shape of the function.

Quadratic hill-climbing (Goldfeld-Quandt)

This method, which is a straightforward variation on Newton-Raphson, is sometimes attributed to Goldfeld and Quandt. Quadratic hill-climbing modifies the Newton-Raphson algorithm by adding a correction matrix (or ridge factor) to the Hessian. The quadratic hill-climbing updating algorithm is given by

$$\theta_{(i+1)} = \theta_{(i)} - \tilde{H}_{(i)}^{-1}g_{(i)} \quad \text{where } -\tilde{H}_{(i)} = -H_{(i)} + \alpha I \tag{D.2}$$

where I is the identity matrix and α is a positive number that is chosen by the algorithm.

The effect of this modification is to push the parameter estimates in the direction of the gradient vector. The idea is that when we are far from the maximum, the local quadratic approximation to the function may be a poor guide to its overall shape, so we may be better off simply following the gradient. The correction may provide better performance at locations far from the optimum, and allows for computation of the direction vector in cases where the Hessian is near singular.

For models which may be estimated using second derivative methods, EViews uses quadratic hill-climbing as its default method. You may elect to use traditional Newton-Raphson, or the first derivative methods described below, by selecting the desired algorithm in the Options menu.

Note that asymptotic standard errors are always computed from the unmodified Hessian once convergence is achieved.

First Derivative Methods

Second derivative methods may be computationally costly since we need to evaluate the $k(k + 1)/2$ elements of the second derivative matrix at every iteration. Moreover, second derivatives calculated may be difficult to compute accurately. An alternative is to employ methods which require only the first derivatives of the objective function at the parameter values.

For general nonlinear models (nonlinear least squares, ARCH and GARCH, nonlinear system estimators, GMM, State Space), EViews provides two first derivative methods: Gauss-Newton/BHHH or Marquardt.

Gauss-Newton/BHHH

This algorithm follows Newton-Raphson, but replaces the negative of the Hessian by an approximation formed from the sum of the outer product of the gradient vectors for each observation's contribution to the objective function. For least squares and log likelihood functions, this approximation is asymptotically equivalent to the actual Hessian when evaluated at the parameter values which maximize the function. When evaluated away from the maximum, this approximation may be quite poor.

The algorithm is referred to as *Gauss-Newton* for general nonlinear least squares problems, and often attributed to Berndt, Hall, Hall, and Hausman (*BHHH*) for maximum likelihood problems.

The advantages of approximating the negative Hessian by the outer product of the gradient are that (1) we need to evaluate only the first derivatives, and (2) the outer product is necessarily positive semi-definite. The disadvantage is that, away from the maximum, this approximation may provide a poor guide to the overall shape of the function, so that more iterations may be needed for convergence.

Marquardt

The Marquardt algorithm modifies the Gauss-Newton algorithm in exactly the same manner as quadratic hill climbing modifies the Newton-Raphson method (by adding a correction matrix (or ridge factor) to the Hessian approximation).

The ridge correction handles numerical problems when the outer product is near singular and may improve the convergence rate. As above, the algorithm pushes the updated parameter values in the direction of the gradient.

For models which may be estimated using first derivative methods, EViews uses Marquardt as its default method. You may elect to use traditional Gauss-Newton via the Options menu.

Note that asymptotic standard errors are always computed from the unmodified (Gauss-Newton) Hessian approximation once convergence is achieved.

Choosing the step size

At each iteration we can search along the given direction for the optimal step size. EViews performs a simple trial-and-error search at each iteration to determine a step size λ that improves the objective function. This procedure is sometimes referred to as *squeezing* or *stretching*.

Note that while EViews will make a crude attempt to find a good step, λ is not actually optimized at each iteration since the computation of the direction vector is often more important than the choice of the step size. It is possible, however, that EViews will be unable to find a step size that improves the objective function. In this case, EViews will issue an error message.

EViews also performs a crude trial-and-error search to determine the scale factor α for Marquardt and quadratic hill-climbing methods.

Derivative free methods

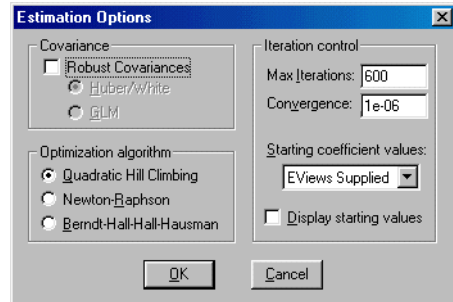
Other optimization routines do not require the computation of derivatives. The *grid search* is a leading example. Grid search simply computes the objective function on a grid of parameter values and chooses the parameters with the highest values. Grid search is computationally costly, especially for multi-parameter models.

EViews uses (a version of) grid search for the exponential smoothing routine.

Setting Estimation Options

When you estimate an equation in EViews, you enter specification information into the Equation Specification dialog. The dialog will differ depending upon the estimation method chosen.

Clicking on the **Options** button in the dialog opens the Estimation Options dialog, allowing you to set various options to control the estimation procedure. The contents of the Estimation Options dialog will differ depending upon the options available for a particular estimation procedure. The Estimation Options dialog for binary models is depicted here. For other estimator and estimation techniques (e.g. systems) the dialog will differ as different available estimation options are available.



Starting Coefficient Values

Iterative estimation procedures require starting values for the coefficients of the model. There are no general rules for selecting starting values for parameters. Obviously, the closer to the true values, the better, so if you have reasonable guesses for parameter values, these can be useful. In some cases, you can obtain starting values by estimating a restricted version of the model. In general, however, you may have to experiment to find good starting values.

EViews follows three basic rules for selecting starting values:

- For nonlinear least squares type problems, EViews uses the values in the coefficient vector at the time you begin the estimation procedure as starting values.
- For system estimators and ARCH, EViews uses starting values based upon preliminary single equation OLS or TSLS estimation. In the dialogs for these estimators, the drop-down menu for setting starting values will not appear.
- For selected estimation techniques (binary, ordered, count, censored and truncated) EViews has built-in algorithms for determining the starting values using specific information about the objective function. These will be labeled in the Starting Values combo box as **EViews Supplied**.

In the latter two cases, you may change this default behavior by selecting an item from the **Starting coefficient values** drop down menu. You may choose fractions of the default starting values, zero, or arbitrary **User Supplied**.

If you select **User Supplied**, EViews will use the values stored in the C coefficient vector at the time of estimation as starting values. To see the starting values, double click on the coefficient vector in the workfile directory. If the values appear to be reasonable, you can close the window and proceed with estimating your model.

If you wish to change the starting values, first make certain that the spreadsheet view of the coefficient vector is in edit mode, then enter the coefficient values. When you are finished setting the initial values, close the coefficient vector window and estimate your model.

You may also set starting coefficient values from the command window using the PARAM command. Simply enter the PARAM keyword, followed by pairs of coefficients and their desired values:

```
param c(1) 153 c(2) .68 c(3) .15
```

sets $C(1) = 153$, $C(2) = .68$, and $C(3) = .15$. All of the other elements of the coefficient vector are left unchanged.

Lastly, if you want to use estimated coefficients from another equation, select **Procs/Update Coefs from Equation** from the equation window toolbar.

For nonlinear least squares problems or situations where you specify the starting values, bear in mind that:

- The objective function must be defined at the starting values. For example, if your objective function contains the expression $1/C(1)$, then you cannot set $C(1)$ to zero. Similarly, if the objective function contains $\text{LOG}(C(2))$, then $C(2)$ must be greater than zero.
- A poor choice of starting values may cause the nonlinear least squares algorithm to fail. EViews begins nonlinear estimation by taking derivatives of the objective function with respect to the parameters, evaluated at these values. If these derivatives are not well behaved, the algorithm may be unable to proceed.

If, for example, the starting values are such that the derivatives are all zero, you will immediately see an error message indicating that EViews has encountered a “Near Singular Matrix”, and the estimation procedure will stop.

- Unless the objective function is globally concave, iterative algorithms may stop at a local optimum. There will generally be no evidence of this fact in any of the output from estimation.

If you are concerned with the possibility of local optima, you may wish to select various starting values and see whether the estimates converge to the same values. One common suggestion is to estimate the model and then randomly alter each of the estimated coefficients by some percentage, then use these new coefficients as starting values in estimation.

Iteration and Convergence Options

There are two common iteration stopping rules: based on the change in the objective function, or based on the change in parameters. The convergence rule used in EViews is based upon changes in the parameter values. This rule is generally conservative, since the change in the objective function may be quite small as we approach the optimum (this is how we choose the direction), while the parameters may still be changing.

The exact rule in EViews is based on comparing the norm of the change in the parameters with the norm of the current parameter values. More specifically, the convergence test is:

$$\frac{\|\theta_{(i+1)} - \theta_{(i)}\|_2}{\|\theta_{(i)}\|_2} \leq tol \quad (D.3)$$

where θ is the vector of parameters, $\|x\|_2$ is the 2-norm of x , and tol is the specified tolerance. However, before taking the norms, each parameter is scaled based on the largest observed norm across iterations of the derivative of the least squares residuals with respect to that parameter. This automatic scaling system makes the convergence criteria more robust to changes in the scale of the data, but does mean that restarting the optimization from the final converged values may cause additional iterations to take place, due to slight changes in the automatic scaling value when started from the new parameter values.

The estimation process achieves convergence if the stopping rule is reached using the tolerance specified in the **Convergence** edit box of the Estimation Dialog or the Estimation Options Dialog. By default, the box will be filled with the tolerance value specified in the global estimation options, or if the estimation object has previously been estimated, it will be filled with the convergence value specified for the last set of estimates.

EViews may stop iterating even when convergence is not achieved. This can happen for two reasons. First, the number of iterations may have reached the prespecified upper bound. In this case, you should reset the maximum number of iterations to a larger number and try iterating until convergence is achieved.

Second, EViews may issue an error message indicating a “Failure to improve” after a number of iterations. This means that even though the parameters continue to change, EViews could not find a direction or step size that improves the objective function. This can happen when the objective function is ill-behaved; you should make certain that your model is identified. You might also try other starting values to see if you can approach the optimum from other directions.

Lastly, EViews may converge, but warn you that there is a singularity and that the coefficients are not unique. In this case, EViews will not report standard errors or t -statistics for the coefficient estimates.

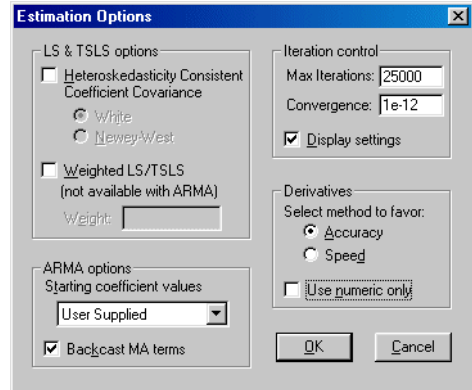
Derivative Computation Options

In many EViews estimation procedures, you can specify the form of the function for the mean equation. For example, when estimating a regression model, you may specify an arbitrary nonlinear expression in the coefficients. In these cases, when estimating the model, EViews will compute derivatives of the user-specified function.

EViews uses two techniques for evaluating derivatives: numeric (finite difference) and analytic. The approach that is used depends upon the nature of the optimization problem and any user-defined settings:

- In most cases, EViews offers the user the choice of computing either analytic or numeric derivatives. By default, EViews will fill the options dialog with the global estimation settings. If the **Use numeric only** setting is chosen, EViews will only compute the derivatives using finite difference methods. If this setting is not checked, EViews will attempt to compute analytic derivatives, and will use numeric derivatives only where necessary.
- EViews will ignore the numeric derivative setting and use an analytic derivative whenever a coefficient derivative is a constant value.
- For some procedures where the range of specifications allowed is limited, EViews always uses analytic first and/or second derivatives. VARs, Pools, binary models (probit, logit, etc.), count models, censored (tobit) models, and ordered models all fall into this category.
- The derivatives with respect to the AR coefficients in an ARMA specification are always computed analytically while those with respect to the MA coefficients are computed numerically.
- In a limited number of cases, EViews will always use numeric derivatives. For the moment, GARCH and State Space models always use numeric derivatives. As noted above, MA coefficient derivatives are always computed numerically.
- Logl objects always use numeric derivatives unless you provide the analytic derivatives in the specification.

- Where relevant, the estimation options dialog allows you to control the method of taking derivatives. For example, the options dialog for standard regression allows you to override the use of EViews analytic derivatives, and to choose between favoring speed or accuracy in the computation of any numeric derivatives.



Computing the more accurate numeric derivatives requires additional objective function evaluations. While the algorithms may change in future versions, at present, EViews computes numeric derivatives using either a one-sided finite difference (favor speed), or using a four-point routine using Richardson extrapolation (favor precision). Additional details are provided in Kincaid and Cheney (1996).

Analytic derivatives will often be faster and more accurate than numeric derivatives, especially if the analytic derivatives have been simplified and carefully optimized to remove common subexpressions. Numeric derivatives will sometimes involve fewer floating point operations than analytic, and in these circumstances, may be faster.

EViews provides tools for examining the effect of your derivative choices. See [Appendix E, “Gradients and Derivatives”](#), on page 675 for additional details.

Nonlinear Equation Solution Methods

When solving a nonlinear equation system, EViews first analyzes the system to determine if the system can be separated into two or more blocks of equations which can be solved sequentially rather than simultaneously. Technically, this is done by using a graph representation of the equation system where each variable is a vertex and each equation provides a set of edges. A well known algorithm from graph theory is then used to find the strongly connected components of the directed graph.

Once the blocks have been determined, each block is solved for in turn. If the block contains no simultaneity, each equation in the block is simply evaluated once to obtain values for each of the variables.

If a block contains simultaneity, the equations in that block are solved by either a Gauss-Seidel or Newton method, depending on how the solver options have been set.

Gauss-Seidel

By default, EViews uses the Gauss-Seidel method when solving systems of nonlinear equations. Suppose the system of equations is given by:

$$\begin{aligned}x_1 &= f_1(x_1, x_2, \dots, x_N, z) \\x_2 &= f_2(x_1, x_2, \dots, x_N, z) \\&\vdots \\x_N &= f_N(x_1, x_2, \dots, x_N, z)\end{aligned}\tag{D.4}$$

where x are the endogenous variables and z are the exogenous variables.

The problem is to find a fixed point such that $x = f(x, z)$. Gauss-Seidel uses an iterative updating rule of the form:

$$x^{(i+1)} = f(x^{(i)}, z).\tag{D.5}$$

to find the solution. At each iteration, EViews solves the equations in the order that they appear in the model. If an endogenous variable that has already been solved for in that iteration appears later in some other equation, EViews uses the value as solved in that iteration. For example, the k -th variable in the i -th iteration is solved by

$$x_k^{(i)} = f_k(x_1^{(i)}, x_2^{(i)}, \dots, x_{k-1}^{(i)}, x_k^{(i-1)}, x_{k+1}^{(i-1)}, \dots, x_N^{(i-1)}, z).\tag{D.6}$$

The performance of the Gauss-Seidel method can be affected by reordering of the equations. If the Gauss-Seidel method converges slowly or fails to converge, you should try moving the equations with relatively few and unimportant right-hand side endogenous variables so that they appear early in the model.

Newton's Method

Newton's method for solving a system of nonlinear equations consists of repeatedly solving a local linear approximation to the system.

Consider the system of equations written in implicit form:

$$F(x, z) = 0\tag{D.7}$$

where F is the set of equations, x is the vector of endogenous variables and z is the vector of exogenous variables.

In Newton's method, we take a linear approximation to the system around some values x^* and z^* :

$$F(x, z) = F(x^*, z^*) + \frac{\partial}{\partial x} F(x^*, z^*) \Delta x = 0\tag{D.8}$$

and then use this approximation to construct an iterative procedure for updating our current guess for x :

$$x_{t+1} = x_t - \left[\frac{\partial}{\partial x} F(x_t, z^*) \right]^{-1} F(x_t, z^*) \quad (\text{D.9})$$

where raising to the power of -1 denotes matrix inversion.

The procedure is repeated until the changes in x_t between periods are smaller than a specified tolerance.

Note that in contrast to Gauss-Seidel, the ordering of equations under Newton does not affect the rate of convergence of the algorithm.

Appendix E. Gradients and Derivatives

Many EViews estimation objects provide built-in routines for examining the gradients and derivatives of your specifications. You can, for example, use these tools to examine the analytic derivatives of your nonlinear regression specification in numeric or graphical form, or you can save the gradients from your estimation routine for specification tests.

The gradient and derivative views may be accessed from most estimation objects by selecting **View/Gradients and Derivatives** or, in some cases, **View/Gradients**, and then selecting the appropriate view.

If you wish to save the numeric values of your gradients and derivatives you will need to use the gradient and derivative procedures. These procs may be accessed from the main **Procs** menu.

Note that all views and procs are not available for every estimation object or every estimation technique.

Gradients

EViews provides you with the ability to examine and work with the gradients of the objective function for a variety of estimation objects. Examining these gradients can provide useful information for evaluating the behavior of your nonlinear estimation routine, or can be used as the basis of various tests of specification.

Since EViews provides a variety of estimation methods and techniques, the notion of a gradient is a bit difficult to describe in casual terms. EViews will generally report the values of the first-order conditions used in estimation. To take the simplest example, ordinary least squares minimizes the sum-of-squared residuals:

$$S(\beta) = \sum_t (y_t - X_t' \beta)^2 \quad (\text{E.1})$$

The first-order conditions for this objective function are obtained by differentiating with respect to β , yielding

$$\sum_t -2(y_t - X_t' \beta) X_t \quad (\text{E.2})$$

EViews allows you to examine both the sum and the corresponding average, as well as the value for each of the individual observations. Furthermore, you can save the individual values in series for subsequent analysis.

The individual gradient computations are summarized in the following table:

Least squares	$g_t = -2(y_t - f_t(X_t, \beta)) \left(\frac{\partial f_t(X_t, \beta)}{\partial \beta} \right)$
Weighted least squares	$g_t = -2(y_t - f_t(X_t, \beta)) w_t^2 \left(\frac{\partial f_t(X_t, \beta)}{\partial \beta} \right)$
Two-stage least squares	$g_t = -2(y_t - f_t(X_t, \beta)) P_t \left(\frac{\partial f_t(X_t, \beta)}{\partial \beta} \right)$
Weighted two-stage least squares	$g_t = -2(y_t - f_t(X_t, \beta)) w_t \tilde{P}_t w_t \left(\frac{\partial f_t(X_t, \beta)}{\partial \beta} \right)$
Maximum likelihood	$g_t = \frac{\partial l_t(X_t, \beta)}{\partial \beta}$

where P and \tilde{P} are the projection matrices corresponding to the expressions for the estimators in [Chapter 12, “Additional Regression Methods”, beginning on page 279](#), and where l is the log likelihood contribution function.

Note that the expressions for the regression gradients are adjusted accordingly in the presence of ARMA error terms.

Gradient Summary

To view the summary of the gradients, select **View/Gradients and Derivatives/Gradient Summary**, or **View/Gradients/Summary**. EViews will display a summary table showing the sum, mean and Newton direction associated with the gradients. Here is an example table from a nonlinear least squares estimation equation:

```
Gradients of the objective function at estimated
parameters
Equation: EQ1
Method: Least Squares
Specification: Y = C(1)*EXP(-C(2)*X) + C(3)*EXP(-(X
-C(4))^2) / C(5)^2 + C(6)*EXP(-(X-C(7))^2) /
C(8)^2
Computed using analytic derivatives
```

Coefficient	Sum	Mean	Newton Dir.
C(1)	-3.49E-09	-1.40E-11	-2.43E-12
C(2)	-2.72E-06	-1.09E-08	-7.74E-16
C(3)	-7.76E-09	-3.11E-11	-9.93E-12
C(4)	3.85E-09	1.54E-11	1.04E-14
C(5)	8.21E-09	3.29E-11	1.97E-13
C(6)	1.21E-09	4.84E-12	-2.20E-12
C(7)	-9.16E-10	-3.67E-12	3.53E-14
C(8)	2.85E-08	1.14E-10	3.95E-13

There are several things to note about this table. The first line of the table indicates that the gradients have been computed at estimated parameters. If you ask for a gradient view for an estimation object that has not been successfully estimated, EViews will compute the gradients at the current parameter values and will note this in the table. This behavior allows you to diagnose unsuccessful estimation problems using the gradient values.

Second, you will note that EViews informs you that the gradients were computed using analytic derivatives. EViews will also inform you if the specification is linear, if the derivatives were computed numerically, or if EViews used a mixture of analytic and numeric techniques. We remind you that all MA coefficient derivatives are computed numerically.

Lastly, there is a table showing the sum and mean of the gradients as well as a column labeled “Newton Dir.”. The column reports the non-Marquardt adjusted Newton direction used in first-derivative iterative estimation procedures (see [“First Derivative Methods” on page 665](#)).

In the example above, all of the values are “close” to zero. While one might expect these values always to be close to zero when evaluated at the estimated parameters, there are a number of reasons why this will not always be the case. First, note that the sum and mean values are highly scale variant so that changes in the scale of the dependent and independent variables may lead to marked changes in these values. Second, you should bear in mind that while the Newton direction is *related* to the terms used in the optimization procedures, EViews’ test for convergence does not directly use the Newton direction. Third, some of the iteration options for system estimation do not iterate coefficients or weights fully to convergence. Lastly, you should note that the values of these gradients are sensitive to the accuracy of any numeric differentiation.

Gradient Table and Graph

There are a number of situations in which you may wish to examine the individual contributions to the gradient vector. For example, one source of singularity in nonlinear estimation, or poor starting values is the presence of very small combined with very large gradients at a given set of coefficient values.

EViews allows you to examine your gradients in two ways: as a spreadsheet of values, or as line graphs, which each set of coefficient gradients plotted in a separate graph. Using these tools you can examine your data for observations with outlier values for the gradients.

Gradient Series

You can save the individual gradient values in series using the **Make Gradient Group** procedure. EViews will create a new group containing series with names of the form GRAD## where ## is the next available name.

Note that when you store the gradients, EViews will fill the series for the full workfile range. If you view the series, make sure to set the workfile sample to the sample used in estimation if you want to reproduce the table displayed in the gradient views.

Application to LM Tests

The gradient series are perhaps most useful for carrying out Lagrange multiplier tests for nonlinear models by running what is known as artificial regressions (Davidson and MacKinnon 1993, Chapter 6). A generic artificial regression for hypothesis testing takes the form of regressing

$$\tilde{u}_t \text{ on } \left(\frac{\partial f_t(X_t, \beta)}{\partial \beta} \right) \text{ and } Z_t \quad (\text{E.3})$$

where \tilde{u} are the estimated residuals under the restricted (null) model, and β are the estimated coefficients. The Z are a set of “misspecification indicators” which correspond to departures from the null hypothesis.

An example program (“GALLANT2.PRG”) for performing an LM auxiliary regression test is provided in your EViews installation directory.

Gradient Availability

The gradient views are currently available for the equation, logl, sspace and system objects. The views are not, however, currently available for equations estimated by GMM or ARMA equations specified by expression.

Derivatives

EViews employs a variety of rules for computing the derivatives used by iterative estimation procedures. These rules, and the user-defined settings that control derivative taking, are described in detail in [“Derivative Computation Options” on page 670](#).

In addition, EViews provides both object views and object procedures which allow you to examine the effects of those choices, and the results of derivative taking. These views and procedures provide you with quick and easy access to derivatives of your user-specified functions.

It is worth noting that these views and procedures are not available for all estimation techniques. For example, the derivative views are currently not available for binary models since only a limited set of specifications are allowed.

Derivative Description

The Derivative Description view provides a quick summary of the derivatives used in estimation.

For example, consider the simple nonlinear regression model

$$y_t = c(1)(1 - \exp(-c(2)x_t)) + \epsilon_t \quad (\text{E.4})$$

Following estimation of this single equation, we can display the description view by selecting **View/Gradients and Derivatives.../Derivative Description**.

```
Derivatives of the equation specification
Equation: EQ1
Method: Least Squares
Specification: RESID = Y - (C(1)*(1 - EXP(-C(2)*X)))
Computed using analytic derivatives
```

Coefficient	Derivative of Specification
C(1)	-1 + exp(-c(2) * x)
C(2)	-c(1) * x * exp(-c(2) * x)

There are three parts to the output from this view. First, the line labeled “Specification:” describes the equation specification that we are estimating. You will note that we have written the specification in terms of the implied residual from our specification.

The next line describes the method used to compute the derivatives used in estimation. Here, EViews reports that the derivatives were computed analytically.

Lastly, the bottom portion of the table displays the expressions for the derivatives of the regression function with respect to each coefficient. Note that the derivatives are in terms of the implied residual so that the signs of the expressions have been adjusted accordingly.

In this example, all of the derivatives were computed analytically. In some cases, however, EViews will not know how to take analytic derivatives of your expression with respect to one or more of the coefficient. In this situation, EViews will use analytic expressions where possible, and numeric where necessary, and will report which type of derivative was used for each coefficient.

Suppose, for example, that we estimate

$$y_t = c(1)(1 - \exp(-\phi(c(2)x_t))) + \epsilon_t \quad (\text{E.5})$$

where ϕ is the standard normal density function. The derivative view of this equation is

Derivatives of the equation specification
 Equation: EQ1
 Method: Least Squares
 Specification: $\text{RESID} = Y - (C(1)*(1 - \text{EXP}(-@DNORM(C(2)*X))))$
 Computed using analytic derivatives
 Use accurate numeric derivatives where necessary

Coefficient	Derivative of Specification
C(1)	$-1 + \exp(-@dnorm(c(2) * x))$
C(2)	--- accurate numeric ---

Here, EViews reports that it attempted to use analytic derivatives, but that it was forced to use a numeric derivative for C(2) (since it has not yet been taught the derivative of the @DNORM function).

If we set the estimation option so that we only compute fast numeric derivatives, the view would change to

Derivatives of the equation specification
 Equation: EQ1
 Method: Least Squares
 Specification: $\text{RESID} = Y - (C(1)*(1 - \text{EXP}(-C(2)*X)))$
 Computed using fast numeric derivatives

Coefficient	Derivative of Specification
C(1)	--- fast numeric ---
C(2)	--- fast numeric ---

to reflect the different method of taking derivatives.

If your specification contains autoregressive terms, EViews will only compute the derivatives with respect to the regression part of the equation. The presence of the AR components is, however, noted in the description view.

Derivatives of the equation specification
 Equation: EQ1
 Method: Least Squares
 Specification: $[\text{AR}(1)=C(3)] = Y - (C(1)*(1 - \text{EXP}(-C(2)*X)))$
 Computed using analytic derivatives

Coefficient	Derivative of Specification*
C(1)	$-1 + \exp(-c(2) * x)$
C(2)	$-c(1) * x * \exp(-c(2) * x)$

*Note: derivative expressions do not account for AR components

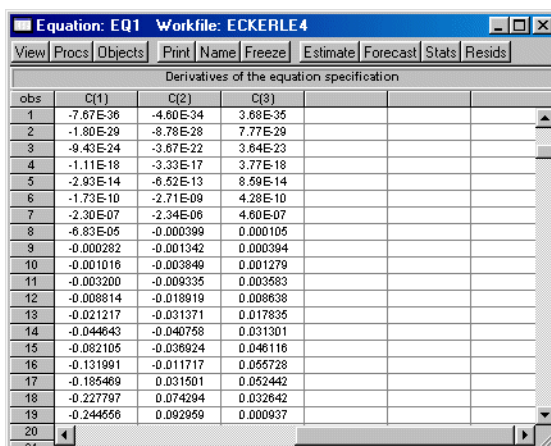
Recall that the derivatives of the objective function with respect to the AR components are always computed analytically using the derivatives of the regression specification, and the lags of these values.

One word of caution about derivative expressions. For many equation specifications, analytic derivative expressions will be quite long. In some cases, the analytic derivatives will be longer than the space allotted to them in the table output. You will be able to identify these cases by the trailing “...” in the expression.

To see the entire expression, you will have to create a table object and then resize the appropriate column. Simply click on the **Freeze** button on the toolbar to create a table object, and then highlight the column of interest. Click on **Width** on the table toolbar and enter in a larger number.

Derivative Table and Graph

Once we obtain estimates of the parameters of our nonlinear regression model, we can examine the values of the derivatives at the estimated parameter values. Simply select **View/Gradients and Derivatives...** to see a spreadsheet view or line graph of the values of the derivatives for each coefficient.



obs	C(1)	C(2)	C(3)
1	-7.87E-36	-4.60E-34	3.68E-36
2	-1.80E-29	-8.78E-28	7.77E-29
3	-9.43E-24	-3.67E-22	3.64E-23
4	-1.11E-18	-3.33E-17	3.77E-18
5	-2.93E-14	-6.52E-13	8.59E-14
6	-1.73E-10	-2.71E-09	4.28E-10
7	-2.30E-07	-2.34E-06	4.60E-07
8	-6.83E-05	-0.000399	0.000105
9	-0.000282	-0.001342	0.000394
10	-0.001016	-0.003849	0.001279
11	-0.003200	-0.009335	0.003583
12	-0.008814	-0.018919	0.008638
13	-0.021217	-0.031371	0.017835
14	-0.044643	-0.040758	0.031301
15	-0.082105	-0.036924	0.046116
16	-0.131991	-0.011717	0.055728
17	-0.185469	0.031501	0.052442
18	-0.227797	0.074294	0.032642
19	-0.244556	0.092959	0.000937
20			

This spreadsheet view displays the value of the derivatives for each observation in the standard spreadsheet form. The graph view, plots the value of each of these derivatives for each coefficient.

Derivative Series

You can save the derivative values in series for later use. Simply select **Procs/Make Derivative Group** and EViews will create an untitled group object containing the new series. The series will be named DERIV##, where ## is a number associated with the next available free name. For example if you have the objects DERIV01 and DERIV02, but not DERIV03 in the workfile, EViews will save the next derivative in the series DERIV03.

Appendix F. Information Criteria

As part of the output for most regression procedures, EViews reports various information criteria. The information criteria are often used as a guide in model selection (see for example, Grasa 1989).

The Kullback-Leibler quantity of information contained in a model is the distance from the “true” model and is measured by the log likelihood function. The notion of an information criterion is to provide a measure of information that strikes a balance between this measure of goodness of fit and parsimonious specification of the model. The various information criteria differ in how to strike this balance.

Definitions

The basic information criteria are given by

Akaike info criterion (AIC)	$-2(l/T) + 2(k/T)$
Schwarz criterion (SC)	$-2(l/T) + k\log(T)/T$
Hannan-Quinn criterion (HQ)	$-2(l/T) + 2k\log(\log(T))/T$

Let l be the value of the log of the likelihood function with the k parameters estimated using T observations. The various information criteria are all based on -2 times the average log likelihood function, adjusted by a penalty function.

In addition to the information criteria described above, there are specialized information criteria that are used in by EViews when computing unit root tests:

Modified AIC (MAIC)	$-2(l/T) + 2((k + \tau)/T)$
Modified SIC (MSIC)	$-2(l/T) + (k + \tau)\log(T)/T$
Modified Hannan-Quinn (MHQ)	$-2(l/T) + 2(k + \tau)\log(\log(T))/T$

where the modification factor τ is computed as

$$\tau = \alpha^2 \sum_t \tilde{y}_{t-1}^2 / \sigma^2 \quad (\text{F.1})$$

for $\tilde{y}_t \equiv y_t$ when computing the ADF test equation (13.50), and for \tilde{y}_t as defined in (“Autoregressive Spectral Density Estimator” beginning on page 339) when estimating the frequency zero spectrum (see Ng and Perron, 2002, for a discussion of the modified information criteria).

Note also that:

- The Hannan-Quinn criterion is reported only for binary, ordered, censored, and count models.
- The definitions used by EViews may differ slightly from those used by some authors. For example, Grasa (1989, equation 3.21) does not divide the AIC by n . Other authors omit inessential constants of the Gaussian log likelihood (generally, the terms involving 2π).

While very early versions of EViews reported information criteria that omitted inessential constant terms, the current version of EViews always uses the value of the full likelihood function. All of your equation objects estimated in earlier versions of EViews will automatically be updated to reflect this change. You should, however, keep this fact in mind when comparing results from frozen table objects or printed output from previous versions.

- For systems of equations, where applicable, the information criteria are computed using the full system log likelihood. The log likelihood value is computed assuming a multivariate normal (Gaussian) distribution as:

$$l = -\frac{TM}{2}(1 + \log 2\pi) - \frac{T}{2} \log |\hat{\Omega}| \quad (\text{F.2})$$

where

$$|\hat{\Omega}| = \det\left(\sum_i \hat{\epsilon}\hat{\epsilon}'/T\right) \quad (\text{F.3})$$

M is the number of equations. Note that these expressions are only strictly valid when you there are equal numbers of observations for each equation. When your system is unbalanced, EViews replaces these expressions with the appropriate summations.

Using Information Criteria as a Guide to Model Selection

As a user of these information criteria as a model selection guide, you select the model with the smallest information criterion.

The information criterion has been widely used in time series analysis to determine the appropriate length of the distributed lag. Lütkepohl (1991, Chapter 4) presents a number of results regarding consistent lag order selection in VAR models.

You should note, however, that the criteria depend on the unit of measurement of the dependent variable y . For example, you cannot use information criteria to select between a model with dependent variable y and one with $\log(y)$.

References

- Abramowitz, M. and I. A. Stegun (1964). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover Publications.
- Aitchison, J. and S.D. Silvey (1957). "The Generalization of Probit Analysis to the Case of Multiple Responses," *Biometrika*, 44, 131-140.
- Agresti, Alan (1996). *An Introduction to Categorical Data Analysis*, New York: John Wiley & Sons.
- Amemiya, Takeshi (1983) "Nonlinear Regression Models," Chapter 6 in Z. Griliches and M. D. Intriligator (eds.), *Handbook of Econometrics*, Volume 1, North-Holland.
- Amisano, Gianni and Carlo Giannini (1997). *Topics in Structural VAR Econometrics*, 2nd ed, Springer.
- Anderson, T. W. and D. A. Darling (1952). "Asymptotic Theory of Certain Goodness of Fit Criteria Based on Stochastic Processes," *Annals of Mathematical Statistics*, 23, 193-212.
- Anderson, T. W. and D. A. Darling (1954), "A Test of Goodness of Fit," *Journal of the American Statistical Association*, 49, 765-769.
- Andrews, Donald W. K. (1988a). "Chi-Square Diagnostic Tests for Econometric Models: Theory," *Econometrica*, 56, 1419-1453.
- Andrews, Donald W. K. (1988b). "Chi-Square Diagnostic Tests for Econometric Models: Introduction and Applications," *Journal of Econometrics*, 37, 135-156.
- Andrews, Donald W. K. (1991). "Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation," *Econometrica*, 59, 817-858.
- Andrews, Donald W. K. and J. Christopher Monahan (1992). "An Improved Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimator," *Econometrica*, 60, 953-966.
- Beck, Nathaniel and Jonathan N. Katz (1995). "What to Do (and Not to Do) With Time-series Cross-section Data," *American Political Science Review*, 89(3), September, 634-647.
- Bergmann, Reinhard, John Ludbrook, and Will P. J. M. Spoonen (2000). "Different Outcomes of the Wilcoxon-Mann-Whitney Test From Different Statistical Packages," *The American Statistician*, 45(1), 72-77.
- Berndt, Ernst R. and David O. Wood (1975). "Technology, Prices and the Derived Demand for Energy," *Review of Economics and Statistics*, 57(3), August, 259-268.
- Bhargava, A. (1986). "On the Theory of Testing for Unit Roots in Observed Time Series," *Review of Economic Studies*, 53, 369-384.
- Blanchard, Olivier (1989). "A Traditional Interpretation of Macroeconomic Fluctuations," *American Economic Review*, 79, 1146-1164.
- Blanchard, Olivier and Danny Quah (1989). "The Dynamic Effects of Aggregate Demand and Aggregate Supply Disturbances," *American Economic Review*, 79, 655-673.
- Bollerslev, Tim (1986). "Generalized Autoregressive Conditional Heteroskedasticity," *Journal of Econometrics*, 31, 307-327.
- Bollerslev, Tim, Ray Y. Chou, and Kenneth F. Kroner (1992). "ARCH Modeling in Finance: A Review

- of the Theory and Empirical Evidence,” *Journal of Econometrics*, 52, 5–59.
- Bollerslev, Tim and Jeffrey M. Wooldridge (1992). “Quasi-Maximum Likelihood Estimation and Inference in Dynamic Models with Time Varying Covariances,” *Econometric Reviews*, 11, 143–172.
- Bollerslev, Tim, Robert F. Engle and Daniel B. Nelson (1994). “ARCH Models,” Chapter 49 in Robert F. Engle and Daniel L. McFadden (eds.), *Handbook of Econometrics*, Volume 4, North-Holland.
- Boswijk, H. Peter (1995). “Identifiability of Cointegrated Systems,” Technical Report, Tinbergen Institute.
- Bowerman, Bruce L. and Richard T. O’Connell (1979). *Time Series and Forecasting: An Applied Approach*, Duxbury Press.
- Box, George E. P. and Gwilym M. Jenkins (1976). *Time Series Analysis: Forecasting and Control*, Revised Edition, Holden-Day.
- Box, George E. P. and D. A. Pierce (1970). “Distribution of Residual Autocorrelations in Autoregressive Integrated Moving Average Time Series Models,” *Journal of the American Statistical Association*, 65, 1509–1526.
- Brock, William, Davis Dechert, Jose Sheinkman & Blake LeBaron (1996). “A Test for Independence Based on the Correlation Dimension,” *Econometric Reviews*, August, 15(3), 197–235.
- Brown, R. L., J. Durbin, and J. M. Evans (1975). “Techniques for Testing the Constancy of Regression Relationships Over Time,” *Journal of the Royal Statistical Society, Series B*, 37, 149–192.
- Brown, M. B. and A. B. Forsythe (1974a). “Robust Tests for the Equality of Variances,” *Journal of the American Statistical Association*, 69, 364–367.
- Brown, M. B. and A. B. Forsythe (1974b). “The Small Sample Behavior of Some Test Statistics which Test the Equality of Several Means,” *Technometrics*, 16, 129–132.
- Cameron, A. Colin and Pravin K. Trivedi (1990). “Regression-based Tests for Overdispersion in the Poisson Model,” *Journal of Econometrics*, 46, 347–364.
- Campbell, John Y. and Pierre Perron (1991). “Pitfalls and Opportunities: What Macroeconomists Should Know about Unit Roots,” *NBER Macroeconomics Annual*, 141–201.
- Chambers, John M., William S. Cleveland, Beat Kleiner, Paul A. Tukey (1983). *Graphical Methods for Data Analysis*, Wadsworth & Brooks/Cole Publishing Company.
- Chesher, A., T. Lancaster, and M. Irish (1985). “On Detecting the Failure of Distributional Assumptions,” *Annales de L’Insee*, 59/60, 7–44.
- Chesher, A. and M. Irish (1987). “Residual Analysis in the Grouped Data and Censored Normal Linear Model,” *Journal of Econometrics*, 34, 33–62.
- Christiano, L. J., M. Eichenbaum, C. L. Evans (1999). “Monetary policy shocks: what have we learned and to what end?” Chapter 2 in J. B. Taylor and M. Woodford, eds., *Handbook of Macroeconomics*, Volume 1A, Elsevier.
- Cleveland, William S. (1993). *Visualizing Data*, Hobart Press.
- Cleveland, William S. (1994). *The Elements of Graphing Data*, Hobart Press.
- Conover, W. J. (1980). *Practical Nonparametric Statistics*, 2nd edition, John Wiley & Sons.

- Conover, W. J., M. E. Johnson and M. M. Johnson (1981). "A Comparative Study of Tests for Homogeneity of Variance with Applications to the Outer Continental Shelf Bidding Data," *Technometrics*, 23, 351–361.
- Csörgö, Sandor and Julian Faraway (1996). "The Exact and Asymptotic Distributions of Cramer-von Mises Statistics", *Journal of the Royal Statistical Society, Series B*, 58, 221-234.
- D'Agostino and Michael A. Stephens, eds. (1986). *Goodness-of-Fit Techniques*. New York: Marcel A. Dekker.
- Dallal, Gerard E. and Leland Wilkinson (1986). "An Analytic Approximation to the Distribution of Lilliefors's Test Statistic For Normality", *The American Statistician*, 40(4), 294-296.
- Davidson, Russell and James G. MacKinnon (1989). "Testing for Consistency using Artificial Regressions," *Econometric Theory*, 5, 363–384.
- Davidson, Russell and James G. MacKinnon (1993). *Estimation and Inference in Econometrics*, Oxford University Press.
- Davis, Charles S., and Michael A. Stephens (1989). "Empirical Distribution Function Goodness-of-Fit Tests," *Applied Statistics*, 38(3), 535-582.
- Dezhbaksh, Hashem (1990). "The Inappropriate Use of Serial Correlation Tests in Dynamic Linear Models," *Review of Economics and Statistics*, 126–132.
- Dickey, D.A. and W.A. Fuller (1979). "Distribution of the Estimators for Autoregressive Time Series with a Unit Root," *Journal of the American Statistical Association*, 74, 427–431.
- Doan, Thomas, Robert B. Litterman, and Christopher A. Sims (1984). "Forecasting and Conditional Projection using Realistic Prior Distributions," *Econometric Reviews*, 3, 1–100.
- Doornik, Jurgen A. and Henrik Hansen (1994). "An Omnibus Test for Univariate and Multivariate Normality," manuscript.
- Doornik, Jurgen A. (1995). "Testing General Restrictions on the Cointegrating Space", manuscript.
- Durbin, J. (1970). *Distribution Theory for Tests Based on the Sample Distribution Function*. SIAM: Philadelphia.
- Dyer, D. D. and J. P. Keating (1980). "On the Determination of Critical Values for Bartlett's Test," *Journal of the American Statistical Association*, 75, 313–319.
- Elliott, Graham, Thomas J. Rothenberg and James H. Stock (1996). "Efficient Tests for an Autoregressive Unit Root," *Econometrica* 64, 813-836.
- Engle, Robert F. (1982). "Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of U.K. Inflation," *Econometrica*, 50, 987–1008.
- Engle, Robert F. (1984). "Wald, Likelihood Ratio, and Lagrange Multiplier Tests in Econometrics," Chapter 13 in Z. Griliches and M. D. Intriligator (eds.), *Handbook of Econometrics*, Volume 2, North-Holland.
- Engle, Robert F. and C. W. J. Granger (1987). "Co-integration and Error Correction: Representation, Estimation, and Testing," *Econometrica*, 55, 251–276.
- Engle, Robert F. and K. F. Kroner (1995). "Multivariate Simultaneous Generalized ARCH," *Econometric Theory*, 11, 122-150.

- Engle, R. and G. J. Lee (1999). "A Permanent and Transitory Component Model of Stock Return Volatility," in R. Engle and H. White, eds., *Cointegration, Causality, and Forecasting: A Festschrift in Honor of Clive W. J. Granger*, Oxford University Press, 475-497.
- Engle, Robert F., David M. Lilien, and Russell P. Robins (1987). "Estimating Time Varying Risk Premia in the Term Structure: The ARCH-M Model," *Econometrica*, 55, 391-407.
- Engle, Robert F. and Victor K. Ng (1993). "Measuring and Testing the Impact of News on Volatility," *Journal of Finance*, 48, 1022-1082.
- Engle, Robert F. and Mark W. Watson (1987). "The Kalman filter: Applications to Forecasting and Rational-Expectations Models" Chapter 7 in Truman F. Bewley (ed.), *Advances in Econometrics—Fifth World Congress*, Volume 1, Cambridge University Press.
- Evans, M., N. Hastings, and B. Peacock (1993). *Statistical Distributions*, 2nd edition, John Wiley & Sons.
- Fahrmeir, Ludwig, and Gerhard Tutz (1994). *Multivariate Statistical Modelling Based on Generalized Linear Models*, Springer.
- Fair, Ray C. (1970). "The Estimation of Simultaneous Equation Models With Lagged Endogenous Variables and First Order Serially Correlated Errors," *Econometrica*, 38, 507-516.
- Fair, Ray C. (1978). "A Theory of Extramarital Affairs," *Journal of Political Economy*, 86, 45-61.
- Fair, Ray C. (1984). *Specification, Estimation, and Analysis of Macroeconometric Models*, Harvard University Press.
- Fama, Eugene F. and Michael R. Gibbons (1982). "Inflation, Real Returns, and Capital Investment", *Journal of Monetary Economics*, 9, 297-323.
- Fan, J. and Gijbels, I. (1996). *Local Polynomial Modelling and Its Applications*, Chapman & Hall.
- Fan, J. and J. S. Marron (1994). "Fast Implementations of Nonparametric Curve Estimators," *Journal of Computational and Graphical Statistics*, 3, 35-56.
- Glosten, L. R., R. Jaganathan, and D. Runkle (1993). "On the Relation between the Expected Value and the Volatility of the Normal Excess Return on Stocks," *Journal of Finance*, 48, 1779-1801.
- Godfrey, L. G. (1988). *Specification Tests in Econometrics*, Cambridge University Press.
- Goldberger, Arthur S. (1991). *A Course in Econometrics*, Harvard University Press.
- Gourieroux, C., A. Monfort, and C. Trognon (1984a). "Pseudo-Maximum Likelihood Methods: Theory," *Econometrica*, 52, 681-700.
- Gourieroux, C., A. Monfort, and C. Trognon (1984b). "Pseudo-Maximum Likelihood Methods: Applications to Poisson Models," *Econometrica*, 52, 701-720.
- Gourieroux, C., A. Monfort, E. Renault, and A. Trognon (1987). "Generalized Residuals," *Journal of Econometrics*, 34, 5-32.
- Granger, C. W. J. (1969). "Investigating Causal Relations by Econometric Models and Cross-Spectral Methods," *Econometrica*, 37, 424-438.
- Grasa, Antonio Aznar (1989). *Econometric Model Selection: A New Approach*, Kluwer.
- Greene, William H. (1997). *Econometric Analysis*, 3rd Edition, Prentice Hall.

- Gujarati, Damodar N. (1995). *Basic Econometrics*, 3rd Edition, McGraw-Hill.
- Hamilton, James D. (1994a). *Time Series Analysis*, Princeton University Press.
- Hamilton, James D. (1994b). "State Space Models", Chapter 50 in Robert F. Engle and Daniel L. McFadden (eds.), *Handbook of Econometrics*, Volume 4, North-Holland.
- Härdle, Wolfgang (1991). *Smoothing Techniques with Implementation in S*, Springer-Verlag.
- Harvey, Andrew C. (1987). "Applications of the Kalman Filter in Econometrics", Chapter 8 in Truman F. Bewley (ed.), *Advances in Econometrics—Fifth World Congress*, Volume 1, Cambridge University Press.
- Harvey, Andrew C. (1989). *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press.
- Harvey, Andrew C. (1990). *The Econometric Analysis of Time Series*, 2nd edition, MIT Press.
- Harvey, Andrew C. (1993). *Time Series Models*, 2nd edition, MIT Press.
- Hayashi, Fumio. (2000). *Econometrics*, Princeton University Press.
- Hausman, Jerry A. (1978). "Specification Tests in Econometrics," *Econometrica*, 46, 1251–1272.
- Hodrick, R. J. and E. C. Prescott (1997). "Postwar U.S. Business Cycles: An Empirical Investigation," *Journal of Money, Credit, and Banking*, 29, 1–16.
- Hosmer, David W. Jr. and Stanley Lemeshow (1989). *Applied Logistic Regression*, John Wiley & Sons.
- Hyndman, Rob J. and Yanan Fan (1996), "Sample Quantiles in Statistical Packages", *The American Statistician*, 50(4), 361-365.
- Jain, Raj and Imrich Chlamtac (1985). "The P2 Algorithm for Dynamic Calculation of Quantiles and Histograms Without Storing Observations," *Communications of the ACM*, 28(10), p. 1076–1085.
- Johansen, Søren and Katarina Juselius (1990). "Maximum Likelihood Estimation and Inferences on Cointegration—with applications to the demand for money," *Oxford Bulletin of Economics and Statistics*, 52, 169–210.
- Johansen, Søren (1991). "Estimation and Hypothesis Testing of Cointegration Vectors in Gaussian Vector Autoregressive Models," *Econometrica*, 59, 1551–1580.
- Johansen, Søren (1995). *Likelihood-based Inference in Cointegrated Vector Autoregressive Models*, Oxford University Press.
- Johnson, Norman L. and Samuel Kotz (1969). *Discrete Distributions*, Houghton Mifflin.
- Johnson, Norman L. and Samuel Kotz (1970). *Continuous Univariate Distributions 1 & 2*, Houghton Mifflin.
- Johnston, Jack and John Enrico DiNardo (1997). *Econometric Methods*, 4th Edition, McGraw-Hill.
- Judge, George G., W. E. Griffiths, R. Carter Hill, Helmut Lütkepohl, and Tsoung-Chao Lee (1985). *The Theory and Practice of Econometrics*, 2nd edition, John Wiley & Sons.
- Kelejian, H. H. (1982). "An Extension of a Standard Test for Heteroskedasticity to a Systems Framework," *Journal of Econometrics*, 20, 325-333.
- Kennan, John (1985). "The Duration of Contract Strikes in U.S. Manufacturing," *Journal of Economet-*

rics, 28, 5–28.

- Kincaid, David, and Ward Cheney (1996). *Numerical Analysis*. 2nd edition, Brooks/Cole Publishing Company.
- Knuth, D. E. (1997). *The Art of Computer Programming, Volume 2, Semi-numerical Algorithms*, 3rd edition. Note: the C implementation of the lagged Fibonacci generator is described in the errata to the 2nd edition, downloadable from Knuth's web site.
- Koopman, Siem Jan, Neil Shephard, and Jurgen A. Doornik (1999). "Statistical algorithms for models in state space using SsfPack 2.2", *Econometrics Journal*, 2(1), 107-160.
- Kwiatkowski, Denis, Peter C. B. Phillips, Peter Schmidt & Yongcheol Shin (1992). "Testing the Null Hypothesis of Stationary against the Alternative of a Unit Root," *Journal of Econometrics*, 54, 159-178.
- LeBaron, Blake (1997). "A Fast Algorithm for the BDS statistic," *Studies in Nonlinear Dynamics and Econometrics*, July, 2(2), 53–59.
- L'Ecuyer, P. (1999). "Good Parameters and Implementations for Combined Multiple Recursive Random Number Generators," *Operations Research*, 47(1), 159-164
- Levene, H. (1960). "Robust Tests for the Equality of Variances," in I. Olkin, S. G. Ghurye, W. Hoefding, W. G. Madow, and H. B. Mann (eds.), *Contribution to Probability and Statistics*, Stanford University Press.
- Lewis, Peter A. W. (1961). "Distribution of the Anderson-Darling Statistic," *Annals of Mathematical Statistics*, 32, 1118-1124.
- Ljung, G. and G. Box (1979). "On a Measure of Lack of Fit in Time Series Models," *Biometrika*, 66, 265–270.
- Lütkepohl, Helmut (1991). *Introduction to Multiple Time Series Analysis*, Springer-Verlag.
- MacKinnon, J. G. (1991). "Critical Values for Cointegration Tests," Chapter 13 in R. F. Engle and C. W. J. Granger (eds.), *Long-run Economic Relationships: Readings in Cointegration*, Oxford University Press.
- MacKinnon, James G. (1996). "Numerical Distribution Functions for Unit Root and Cointegration Tests," *Journal of Applied Econometrics*, 11, 601-618.
- Marron, J. S. and D. Nolan (1989). "Canonical Kernels for Density Estimation," *Statistics and Probability Letters*, 7, 191–195.
- Marsaglia, G. (1993). "Monkey Tests for Random Number Generators," *Computers and Mathematics with Applications*, 9, 1-10.
- Matsumoto, M. and T. Nishimura (1998). "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator," *ACM Transactions on Modeling and Computer Simulation*, 8(1), January, 3-30.
- McCallum, Bennett T. (1989). "Real Business Cycle Models," Chapter 1 in Robert J. Barro (ed.), *Modern Business Cycle Theory*, Harvard University Press.
- McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models*, 2nd Edition, Chapman & Hall.
- McDonald, J. and R. Moffitt (1980). "The Uses of Tobit Analysis," *Review of Economic and Statistics*,

- 62, 318–321.
- Nelson, Daniel B. (1991). “Conditional Heteroskedasticity in Asset Returns: A New Approach,” *Econometrica*, 59, 347–370.
- Neter, John, Michael H. Kutner, Christopher J. Nachtsheim, and William Wasserman (1996). *Applied Linear Statistical Models*, 4th Edition. Chicago: Times Mirror Higher Education Group, Inc. and Richard D. Irwin, Inc.
- Newey, Whitney and Kenneth West (1987a). “Hypothesis Testing with Efficient Method of Moments Estimation,” *International Economic Review*, 28, 777–787.
- Newey, Whitney and Kenneth West (1987b). “A Simple Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix,” *Econometrica*, 55, 703–708.
- Newey, Whitney and Kenneth West (1994). “Automatic Lag Selection in Covariance Matrix Estimation,” *Review of Economic Studies*, 61, 631–653.
- Ng, Serena and Pierre Perron. 2001. “Lag Length Selection and the Construction of Unit Root Tests with Good Size and Power,” *Econometrica*, 69(6), 1519–1554.
- Osterwald-Lenum, Michael (1992). “A Note with Quantiles of the Asymptotic Distribution of the Maximum Likelihood Cointegration Rank Test Statistics,” *Oxford Bulletin of Economics and Statistics*, 54, 461–472.
- Pagan, A. and F. Vella (1989). “Diagnostic Tests for Models Based on Individual Data: A Survey,” *Journal of Applied Econometrics*, 4, S29–S59.
- Pesaran, M. Hashem and Yongcheol Shin (1998). “Impulse Response Analysis in Linear Multivariate Models,” *Economics Letters*, 58, 17–29.
- Phillips, Peter C. B. and S. Ouliaris (1990). “Asymptotic properties of residual based tests for cointegration,” *Econometrica*, 58, 165–193.
- Phillips, P.C.B. and P. Perron (1988). “Testing for a Unit Root in Time Series Regression,” *Biometrika*, 75, 335–346.
- Pindyck, Robert S. and Daniel L. Rubinfeld (1991). *Econometric Models and Economic Forecasts*, 3rd edition, McGraw-Hill.
- Powell, J. L. (1986). “Symmetrically Trimmed Least Squares Estimation for Tobit Models,” *Econometrica*, 54, 1435–1460.
- Prescott, Edward C. (1986). “Theory Ahead of Business-Cycle Measurement,” *Carnegie-Rochester Conference Series on Public Policy*, 25, 11–44.
- Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C*, 2nd edition, Cambridge University Press.
- Quandt, Richard E. (1983). “Computational Problems and Methods,” Chapter 12 in Z. Griliches and M. D. Intriligator (eds.), *Handbook of Econometrics*, Volume 1, North-Holland.
- Ramsey, J. B. (1969). “Tests for Specification Errors in Classical Linear Least Squares Regression Analysis,” *Journal of the Royal Statistical Society, Series B*, 31, 350–371.
- Ramsey, J. B. and A. Alexander (1984). “The Econometric Approach to Business-Cycle Analysis Reconsidered,” *Journal of Macroeconomics*, 6, 347–356.

- Rao, P. and Z. Griliches (1969). "Small Sample Properties of Several Two-Stage Regression Methods in the Context of Auto-Correlated Errors," *Journal of the American Statistical Association*, 64, 253–272.
- Said, Said E. and David A. Dickey (1984). "Testing for Unit Roots in Autoregressive Moving Average Models of Unknown Order," *Biometrika*, 71, 599–607.
- Sheskin, David J. (1997). *Parametric and Nonparametric Statistical Procedures*, CRC Press.
- Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*, Chapman & Hall.
- Simonoff, Jeffrey S. (1996). *Smoothing Methods in Statistics*, Springer-Verlag.
- Sims, Chris (1980). "Macroeconomics and Reality," *Econometrica*, 48, 1-48.
- Sims, Chris (1986). "Are Forecasting Models Usable for Policy Analysis?" *Quarterly Review of the Federal Reserve Bank of Minneapolis*, 2-16.
- Sokal, Robert R. and F. James Rohlf (1995). *Biometry*. W. H. Freeman.
- Stephens, Michael A. (1986). "Tests Based on EDF Statistics," in *Goodness-of-Fit Techniques*, Ralph B. D'Agostino and Michael A. Stephens, eds.. New York: Marcel A. Dekker, 97-193.
- Tauchen, George (1986). "Statistical Properties of Generalized Method-of-Moments Estimators of Structural Parameters Obtained From Financial Market Data," *Journal of Business & Economic Statistics*, 4, 397–416.
- Temme, Nico M. (1996). *Special Functions: An Introduction to the Classical Functions of Mathematical Physics*, John Wiley & Sons.
- Thisted, Ronald A. (1988). *Elements of Statistical Computing*. Chapman and Hall.
- Urzua, Carlos M. (1997). "Omnibus Tests for Multivariate Normality Based on a Class of Maximum Entropy Distributions," in *Advances in Econometrics*, Volume 12, JAI Press, 341-358.
- Watson, Mark W. and Robert F. Engle (1983). "Alternative Algorithms for the Estimation of Dynamic Factor, MIMIC and Varying Coefficient Regression Models," *Journal of Econometrics*, 23, 385–400.
- White, Halbert (1980). "A Heteroskedasticity-Consistent Covariance Matrix and a Direct Test for Heteroskedasticity," *Econometrica*, 48, 817–838.
- White, Halbert (1982). "Maximum Likelihood Estimation of Misspecified Models," *Econometrica*, 50, 1–26.
- Wooldridge, Jeffrey M. (1990). "Quasi-Likelihood Methods for Count Data," Chapter 8 in M. Hashem Pesaran and P. Schmidt (eds.) *Handbook of Applied Econometrics*, Volume 2, Blackwell, 352–406.
- Wooldridge, Jeffrey M. (1997). "A Note on the Lagrange Multiplier and F-statistics for Two Stage Least Squares Regressions," *Economics Letters*, 34, 151-155.
- Zakoian, J. M. (1994). "Threshold Heteroskedastic Models," *Journal of Economic Dynamics and Control*, 18, 931-944.

Index

Symbols

- .DB file [112](#)
- .DB? file [112](#)
 - automatic backup [649](#)
- .EDB file [108](#)
- .RTF file [254](#)
- .WF1 file [37](#)
- ?
 - pool cross section identifier [553](#)
 - wildcard versus pool identifier [660](#)
- @all
 - with smpl [61](#), [655](#)
- @first [655](#)
 - with smpl [61](#), [655](#)
- @last [655](#)
 - with smpl [61](#), [655](#)
- ~, in backup file name [649](#)

Numerics

- 2sls (two-stage least squares) [283](#)
- 3sls (three-stage least squares) [497](#), [514](#)

A

- Active window [6](#), [47](#)
- Add factor [601](#)
- Adjusted R-squared
 - for regression [267](#)
- Advanced database query [126](#)
- AIC [683](#)
- Akaike criterion [269](#), [683](#)
 - for equation [269](#)
- Alias [603](#)
 - database [122](#)
 - OBALIAS.INI file [133](#)
 - object [132](#)
- All (database) [110](#)
- Almon lag [323](#)
- Analysis of variance [159](#)
 - by ranks [162](#)
- Analytic derivatives [678](#)
- And operator [62](#), [90](#)

- Anderson-Darling test [164](#)
- Andrews test [431](#), [469](#)
- ANOVA [159](#)
 - by ranks [162](#)
- AR specification
 - estimation [307](#)
 - forecast [357](#)
 - in 2SLS [286](#)
 - in ARIMA models [311](#)
 - in nonlinear 2SLS [296](#)
 - in nonlinear least squares [294](#)
 - in pool [564](#)
 - in systems [499](#)
- AR(1)
 - coefficient [304](#)
 - Durbin-Watson statistic [304](#)
 - estimation [308](#)
- AR(p) [304](#)
 - estimation [308](#)
- ARCH
 - correlogram test [375](#)
 - LM test [377](#)
 - threshold (TARCH) [408](#)
- AREMOS data [139](#)
- ARIMA models [311](#)
 - Box-Jenkins approach [312](#)
 - diagnostic checking [322](#)
 - difference operator [314](#)
 - identification [312](#)
 - specification [313](#)
 - starting values [319](#)
- ARMA terms [315](#)
 - in models [633](#)
 - seasonal [316](#)
 - testing [376](#)
- Artificial regression [378](#), [390](#)
- ASCII
 - file export [72](#)
 - file import [70](#), [83](#)
- Asymmetric GARCH models [407](#)
 - component model [413](#)
- Asymptotic test [367](#)
- Augmented Dickey-Fuller test [333](#)
- Augmented regression [383](#)

Autocorrelation [268](#)
Autocorrelation function [167](#)
Auto-search
 database [123](#)
Auto-series
 generate new series [98](#), [120](#)
 in estimation [102](#)
 in groups [101](#)
 in regression [120](#)
 with database [119](#)
Autospec [582](#)
Auxiliary regression [376](#), [377](#)
Average log likelihood [426](#)

B

Backcast [320](#)
Backup file option [649](#)
Balanced data [557](#)
Balanced sample [563](#)
Bandwidth [230](#)
 Andrews [517](#)
 bracket option [231](#), [237](#)
 in kernel fit [239](#)
 in nearest neighbor fit [236](#)
 Newey-West (fixed) [517](#)
 Newey-West (variable) [518](#)
 selection in GMM [504](#), [517](#)
Bar graph [151](#)
Bartlett kernel [516](#)
Bartlett test [163](#)
BDS test [170](#)
Berndt-Hall-Hall-Hausman. See also Optimization algorithms.
Bias proportion [353](#)
Binary dependent variable [421](#)
 error messages [428](#)
 fitted index [434](#)
 interpretation of coefficient [425](#)
 log likelihood [422](#)
Binary Estimation
 perfect predictor [428](#)
Binning option [155](#), [216](#)
 exact [231](#), [240](#)
 linear [231](#), [240](#)
Binomial sign test [158](#)
Box-Cox transformation [234](#)

Box-Jenkins [312](#)
Bracket bandwidth option
 in kernel density [231](#)
 in kernel fit [240](#)
 in nearest neighbor fit [237](#)
Breusch-Godfrey test [305](#), [377](#)
Brown-Forsythe test [163](#)
Button bar display option [648](#)

C

C
 coef vector [260](#)
 constant in regression [260](#)
Cache [143](#)
Categorical regressor stats [429](#), [449](#)
Causality
 Granger's test [223](#)
Censored dependent variable [444](#)
 fitted index [450](#)
 goodness-of-fit tests [451](#)
 interpretation of coefficient [449](#)
 log likelihood [445](#)
 scale factor [449](#)
 specifying the censoring point [446](#)
Census X11
 historical [184](#)
 limitations [184](#)
 using X12 [178](#)
Census X12 [177](#)
 seasonal adjustment options [178](#)
Change default directory [38](#)
Chi-square
 independence test in tabulation [217](#)
 statistic for Wald test [369](#)
 test for independence in n-way table [218](#)
 test for the median [162](#)
Cholesky factor
 in VAR impulse responses [528](#)
 in VAR normality test [525](#)
Chow test
 breakpoint [380](#)
 forecast [381](#)
 n-step forecast [387](#)
 one-step forecast [387](#)
Classification table
 binary models [429](#)
 ordered models [442](#)

- sensitivity [430](#)
- specificity [430](#)
- Cleveland subsampling [236](#)
- Click(ing) [6](#)
- Clipboard [252](#), [255](#)
- Close
 - EViews [12](#)
 - object [648](#)
 - window [7](#)
- Cochrane-Orcutt [287](#), [310](#)
- Coef (coefficient vector)
 - default [260](#)
 - update from equation [273](#)
- Coefficient
 - common (pool) [564](#)
 - covariance matrix [271](#)
 - covariance matrix of estimates [272](#)
 - cross-section specific (pool) [564](#)
 - estimated from regression [265](#)
 - maximum number in default [442](#)
 - recursive estimates [388](#)
 - restrictions [262](#)
 - setting initial values [292](#), [667](#)
 - standard error [266](#)
- Coefficient restriction test [368](#)
- Cointegration test [537](#)
- Collinearity [323](#)
- Column width [254](#)
- Command window [10](#)
- Comments [41](#), [50](#)
- Common coefficients (pool) [564](#)
- Common sample [93](#)
- Component GARCH [412](#)
 - asymmetric component [413](#)
 - permanent component [413](#)
 - transitory component [413](#)
- Conditional standard deviation
 - graph [404](#)
- Conditional variance [395](#), [397](#)
 - forecast [406](#)
 - in the mean equation [399](#)
 - make series [406](#)
- Confidence interval
 - for forecast [352](#)
 - for stochastic model solution [635](#)
- Constant
 - in equation [260](#), [266](#)
 - in ordered models [440](#)
 - in pool estimation [564](#)
- Contingency coefficient [218](#)
- Contract range of workfile [38](#)
- Convergence criterion [669](#)
 - in nonlinear least squares [293](#), [297](#)
 - in pool estimation [565](#)
- Copy
 - and paste [51](#)
 - data [71](#)
 - data cut-and-paste [65](#)
 - database [133](#)
 - graph [252](#)
 - object [51](#)
 - table [254](#)
 - to and from database [115](#)
- Copy and paste [252](#), [254](#)
- Correlation [221](#)
- Correlogram [170](#), [221](#), [305](#)
 - autocorrelation function [167](#)
 - cross [221](#)
 - partial autocorrelation function [168](#)
 - Q-statistic [169](#)
 - squared residuals [375](#), [405](#)
- Count models [458](#)
 - negative binomial (ML) [459](#)
 - Poisson [459](#)
- Covariance [221](#)
- Covariance matrix
 - HAC (Newey-West) [282](#)
 - heteroskedasticity consistent of estimated coefficients [281](#)
 - of estimated coefficients [271](#)
- Covariance proportion [353](#)
- Cramer's V [218](#)
- Cramer-von Mises test [164](#)
- Create
 - database [109](#)
 - group [59](#), [102](#)
 - object [45](#)
 - series [55](#), [94](#)
 - text object [255](#)
 - workfile [34](#)
- Cross correlation [221](#)
 - test asymmetry [412](#)
- Cross correlogram [221](#)
- Cross section

- identifiers in pool [552](#)
- Cross-equation
 - coefficient restriction [496](#), [499](#)
 - correlation [497](#), [498](#)
 - weighting [496](#)
- Cross-section
 - cross-section specific coefficients in pool [564](#)
- Cumulative distribution [225](#)
 - Blom option [227](#)
 - Ordinary option [227](#)
 - Rankit option [227](#)
 - Tukey option [227](#)
 - Van der Waerden option [227](#)
- CUSUM of squares test [386](#)
- CUSUM test [385](#)
- D**
- Data
 - cut and paste [67](#), [71](#)
 - export [71](#)
 - import [68](#)
 - keyboard entry [65](#)
 - missing [94](#)
 - pool [553](#)
- Database
 - alias [122](#)
 - auto-search [123](#)
 - auto-series [121](#)
 - cache [143](#)
 - copy [133](#)
 - copy objects [115](#)
 - create [109](#)
 - default [112](#)
 - default in search order [123](#)
 - delete [133](#)
 - delete objects [117](#)
 - export [115](#)
 - fetch objects [114](#)
 - field [126](#)
 - frequency in query [128](#)
 - link [141](#)
 - link options [144](#)
 - match operator in query [128](#)
 - open [109](#)
 - packing [134](#)
 - queries [123](#)
 - rebuild [135](#)
 - rename [133](#)
 - rename object [117](#)
 - repair [135](#)
 - sharing violation [111](#)
 - statistics [134](#)
 - store objects [113](#)
 - test integrity [134](#)
 - window [109](#)
- Database registry [121](#)
- Date
 - format [653](#)
 - notation options [648](#)
- Dated data table [200](#)
 - formatting options [206](#)
 - frequency conversion [204](#)
 - row options [206](#)
 - table options [201](#)
 - transformation methods [204](#)
- Default
 - database [12](#), [112](#)
 - database in search order [123](#)
 - directory [12](#), [648](#)
 - equation [36](#)
 - set directory [38](#)
 - setting global options [647](#)
 - update directory [38](#)
- Default equation [273](#)
- Delete [51](#)
 - cells from table [253](#)
 - database [133](#)
 - objects from database [117](#)
 - observation in series [58](#)
 - series using pool [562](#)
- Demonstration [31](#)
 - creating a workfile [15](#)
 - estimation [22](#)
 - forecasting [28](#)
 - import data [18](#)
 - specification test [24](#)
- Dependent variable
 - no variance in binary models [428](#)
- Derivatives [670](#), [678](#)
 - description [678](#)
 - saving in series [681](#)
- Description
 - field in database query [130](#)
- Descriptive statistics

- balanced sample (pool) [559](#)
 - by classification [154](#)
 - common sample (group) [214](#)
 - common sample (pool) [559](#)
 - cross-section specific [560](#)
 - for a series [156](#)
 - group [214](#)
 - individual samples (group) [214](#)
 - individual samples (pool) [559](#)
 - stacked data [559](#)
 - time period specific [560](#)
 - Deselect all [45](#)
 - Dialog box [9](#)
 - Dickey-Fuller test [333](#)
 - Difference from moving-average [185](#)
 - Difference operator [92](#), [314](#)
 - seasonal [92](#), [314](#)
 - Display filter [36](#), [41](#)
 - Display name
 - field in database query [130](#)
 - Distribution
 - empirical distribution function tests [164](#)
 - tests of [164](#)
 - Distribution graphs [164](#), [225](#)
 - cumulative distribution [225](#)
 - kernel density [164](#), [229](#)
 - QQ-plot [164](#), [227](#)
 - quantile [226](#)
 - survivor [226](#)
 - Doornik and Hansen factorization matrix [525](#)
 - Drag(ging) [6](#)
 - text in graph [249](#)
 - DRI
 - frequency [145](#)
 - queries [145](#)
 - DRI database
 - DRipro [141](#)
 - illegal names [144](#)
 - object alias [132](#)
 - shadowing of object names [132](#)
 - troubleshooting [146](#)
 - Dummy variable [98](#)
 - censoring point [447](#)
 - dependent [421](#)
 - pool series [561](#)
 - Durbin-Watson statistic [304](#)
 - for regression [268](#)
 - lagged dependent variable [305](#)
 - Dynamic forecasting [355](#)
- ## E
- Easy query [124](#)
 - Edit
 - group [60](#)
 - series [56](#), [200](#)
 - table [254](#)
 - Elliot, Rothenberg, and Stock point optimal test [337](#)
 - Empirical distribution function tests [164](#)
 - End field [129](#)
 - Endogeneity [390](#)
 - Endogenous variable [283](#)
 - Epanechnikov kernel [230](#), [239](#)
 - Equality tests [159](#)
 - mean [159](#)
 - median [161](#)
 - variance [163](#)
 - Equation
 - coefficient covariance matrix [271](#)
 - coefficient covariance scalar [270](#)
 - coefficient standard error vector [271](#)
 - coefficient t-statistic scalar [271](#)
 - coefficient t-statistic vector [271](#)
 - coefficient vector [271](#)
 - create [259](#)
 - default [274](#)
 - regression coefficients [265](#)
 - regression summary statistics [267](#)
 - results [265](#)
 - retrieve previously estimated [275](#)
 - saved results [270](#), [274](#)
 - scalar results [270](#)
 - specification [260](#)
 - specify by formula [261](#)
 - specify by list [260](#)
 - specify with non-default coefs [263](#)
 - specify with restrictions [262](#)
 - specify without dependent variable [262](#)
 - specifying a constant [260](#)
 - store [275](#)
 - text representation [272](#)
 - t-statistic [266](#)
 - vector and matrix results [271](#)
 - Error bar graph [210](#)

- Error bar graphs [245](#)
 - Error message display option [651](#)
 - Estimation
 - AR specification [307](#)
 - auto-series [102](#)
 - binary dependent variable [423](#)
 - censored models [445](#)
 - collinearity [276](#)
 - convergence problems [669](#)
 - count models [458](#)
 - derivative computation options [670](#)
 - failure to improve [669](#)
 - for pool [562](#)
 - missing data [264](#)
 - near singular matrix problems [668](#)
 - nonlinear least squares [289](#)
 - options [666](#)
 - ordered models [439](#)
 - output [265](#)
 - problems in convergence [668](#)
 - residuals from equation [274](#)
 - sample [263](#)
 - sample adjustment [264](#)
 - single equation methods [263](#)
 - state space [591](#)
 - systems [503](#)
 - truncated models [454](#)
 - two-stage least squares [283](#)
 - weighted least squares [281](#)
 - Evaluate
 - precedence [88](#)
 - EViews
 - Enterprise Edition [137](#), [138](#)
 - EViews Databases [107](#)
 - Excel files
 - reading data from [68](#)
 - Exogenous variable [283](#)
 - Expand range [38](#)
 - Expectation-prediction table
 - binary models [429](#)
 - ordered models [442](#)
 - Expectations consistency in models [631](#)
 - Expected dependent variable
 - censored models [450](#)
 - truncated models [456](#)
 - Expected latent variable
 - censored models [450](#)
 - truncated models [456](#)
 - Exponential GARCH (EGARCH) [409](#)
 - Exponential smoothing [195](#)
 - double [192](#)
 - Holt-Winters additive [193](#)
 - Holt-Winters multiplicative [192](#)
 - Holt-Winters no-seasonal [193](#)
 - single [191](#)
 - Export data [71](#)
 - from pool objects [558](#)
 - to ASCII files [72](#)
 - to spreadsheet files [71](#)
 - Export database [115](#)
 - Expression [87](#)
 - for database fields [127](#)
 - logical [90](#)
 - parentheses [88](#)
 - Extended search in models [638](#)
 - Extract from workfile [39](#)
 - Extreme value
 - binary model [424](#)
- ## F
- Fair-Taylor model solution [630](#)
 - Fetch
 - from database [114](#)
 - from pool [562](#)
 - Fields in database [126](#)
 - description [130](#)
 - display_name [130](#)
 - end [129](#)
 - expressions [127](#)
 - freq [128](#)
 - history [130](#)
 - last_update [129](#)
 - last_write [129](#)
 - name [127](#)
 - remarks [130](#)
 - source [130](#)
 - start [129](#)
 - type [128](#)
 - units [130](#)
 - Files
 - default locations [648](#)
 - First derivative methods [665](#)
 - Fitted index
 - binary models [434](#)

- censored models [450](#)
- truncated models [456](#)
- Fitted probability
 - binary models [434](#)
- Fitted values
 - of equation [272](#)
- Fixed
 - decimal [254](#)
 - digits [254](#)
- Fixed effects [564](#), [572](#)
 - standard errors [573](#)
- Fixed variance parameter
 - negative binomial QML [462](#)
 - normal QML [461](#)
- Font options [647](#)
 - table [253](#)
 - text in graph [248](#)
- Forecast
 - AR specification [357](#)
 - binary models [434](#)
 - by exponential smoothing [195](#)
 - censored models [450](#)
 - Chow test [381](#)
 - conditional variance [406](#)
 - count models [462](#)
 - dynamic [355](#), [581](#)
 - equations with formula [359](#)
 - error [350](#)
 - evaluation [352](#)
 - fitted values [349](#)
 - from estimated equation [343](#)
 - interval [352](#)
 - MA specification [358](#)
 - n-step ahead [580](#)
 - n-step test [387](#)
 - one-step test [387](#)
 - ordered models [443](#)
 - out-of-sample [349](#)
 - smoothed [581](#)
 - standard error [351](#), [362](#)
 - static [356](#)
 - structural [357](#)
 - truncated models [456](#)
 - variance [350](#)
 - with AR errors [357](#)
- Formula
 - forecast [359](#)
 - implicit assignment [96](#)
 - normalize [97](#)
 - specify equation by [261](#)
- Forward solution for models [629](#)
- Freeze [51](#)
 - create graph from view [243](#)
 - output [651](#)
- Freq
 - field in database query [128](#)
- Frequency [653](#)
- Frequency conversion [51](#), [72](#), [173](#)
 - DRI database [145](#)
 - in dated data table [204](#)
 - propagate NAs [73](#)
 - undated series [73](#)
- F-statistic [369](#), [373](#)
 - for regression [269](#)
- F-test
 - for variance equality [163](#)
- Full information maximum likelihood [498](#)
- G**
- GARCH
 - ARCH-M model [399](#)
 - asymmetric models [407](#)
 - component models [412](#)
 - exponential GARCH (EGARCH) [409](#)
 - GARCH(1,1) model [397](#)
 - GARCH(p,q) model [399](#)
 - initialization [401](#)
 - mean equation [400](#)
 - multivariate [492](#)
 - news impact curve [410](#)
 - QML standard errors [401](#)
 - standardized residual [407](#)
- Gauss-Newton [665](#)
- Gauss-Seidel algorithm [637](#), [672](#)
- Generalized ARCH [397](#)
- Generalized error distribution [409](#)
- Generalized least squares [565](#)
- Generalized linear models [468](#)
 - quasi-likelihood ratio test [462](#)
 - robust standard errors [468](#)
 - variance factor [468](#)
- Generalized method of moments. See GMM.
- Generalized residual
 - binary models [434](#)
 - censored models [450](#)

- count models [463](#)
 - ordered models [444](#)
 - score vector [435](#)
 - truncated models [456](#)
 - Generate series [94](#)
 - by command [97](#)
 - dynamic assignment [96](#)
 - for pool [560](#)
 - implicit formula [96](#)
 - using samples [95](#)
 - Geometric moving average [101](#)
 - GiveWin data [139](#)
 - GLM (generalized linear model) [468](#)
 - standard errors [468](#)
 - Global optimum [668](#)
 - GLS detrending [335](#)
 - GMM
 - bandwidth selection [504](#)
 - for systems [498](#)
 - HAC weighting matrix [516](#)
 - J-statistic [300](#)
 - kernel options [504](#)
 - orthogonality condition [298](#)
 - overidentifying restrictions [300](#)
 - prewhitening option [504](#), [518](#)
 - single equation [297](#)
 - system [515](#)
 - White weighting matrix [516](#)
 - Goldfeld-Quandt [664](#)
 - Gompit models [424](#)
 - Goodness-of-fit
 - adjusted R-squared [267](#)
 - Andrews test [431](#), [469](#)
 - forecast [352](#)
 - Hosmer-Lemeshow test [431](#), [468](#)
 - R-squared [267](#)
 - Gradients [675](#)
 - saving in series [677](#)
 - summary [676](#)
 - Granger causality [222](#)
 - test in VAR [522](#)
 - Graph
 - adding lines and shading [249](#)
 - aspect ratio [246](#)
 - axes control [246](#)
 - color settings [246](#)
 - combining [244](#)
 - coordinates for positioning elements [248](#)
 - creating [243](#)
 - customizing lines and symbols [247](#)
 - error bar [245](#)
 - high-low-open-close [245](#)
 - merging multiple [45](#)
 - modifying [244](#)
 - multiple graph options [250](#)
 - multiple graph positioning [251](#)
 - print in color [252](#)
 - printing [252](#)
 - save as postscript file [252](#)
 - spike [245](#)
 - stacked lines and bars [245](#)
 - templates [249](#)
 - text justification [248](#)
 - Grid
 - in table [254](#)
 - Grid search [666](#)
 - Group [104](#)
 - add member [199](#)
 - auto-series [101](#)
 - create [59](#), [102](#)
 - creating using wildcards [658](#)
 - edit series [200](#)
 - editing [60](#)
 - element [103](#)
 - spreadsheet view [199](#), [650](#)
 - Group into bins option [155](#), [216](#)
 - Groupwise heteroskedasticity [215](#)
- ## H
- Hannan-Quinn criterion [683](#)
 - Hatanaka two step estimator [310](#)
 - Hausman test [390](#)
 - Haver Analytics Database [138](#)
 - Help
 - help system [13](#)
 - World Wide Web [13](#)
 - Heteroskedasticity
 - binary models [437](#)
 - groupwise [215](#)
 - of known form [279](#)
 - of unknown form [281](#)
 - White's test [378](#)
 - Heteroskedasticity and autocorrelation consistent covariance (HAC) [282](#)

- Heteroskedasticity consistent covariance matrix
281
- High-Low (Open-Close) graphs 210, 245
- Hildreth-Lu 310
- Histogram 152, 376
- History
field in database query 130
- Hodrick-Prescott filter 196
- Holt-Winters
additive 193
multiplicative 192
no-seasonal 193
- Horizontal line in table 254
- Hosmer-Lemeshow test 431, 468
- Huber/White standard errors 467
- Hypothesis tests
ARCH 377
Bartlett test 163
BDS independence 170
binomial sign test 158
Brown-Forsythe 163
chi-square test 162
Chow breakpoint 380
coefficient p-value 266
CUSUM 385
CUSUM of squares 386
distribution 164
F-test 163
Hausman test 390
heteroskedasticity 378
irrelevant or redundant variable 374
Kruskal-Wallis test 162
Levene test 163
multi-sample equality 159
nonnested 391
normality 376
omitted variables 373
Ramsey RESET 382
Siegel-Tukey test 163
single sample 156
stability test 379
unit root 170, 329
Van der Waerden test 158, 162
Wald coefficient restriction test 368
Wilcoxon signed ranks test 158
- I**
- Icon 43
- Identification
Box-Jenkins 312
nonlinear models 297
- Identity
in system 500
- If condition in samples 62
- Import data
for pool objects 553, 557
from ASCII 70, 76
from spreadsheet 68
- Impulse response 527
generalized impulses 529
See also VAR.
standard errors 528
structural decomposition 529
transformation of impulses 528
user specified impulses 529
- Incorrect functional form 379, 382
- Independence test 170
- Index
fitted from binary models 434
fitted from censored models 450
fitted from truncated models 456
- Individual sample 93
- Information criterion
Akaike 269, 683
Hannan-Quinn 683
Schwarz 269, 683
- Innovation 303
- Insert
cells to table 253
observation 58
- Insertion point 11
- Instrumental variable 283
for 2SLS with AR specification 287
for GMM 298
for nonlinear 2SLS 295
identification 503
in systems 500
order condition 284
- Instruments 283
using PDL specifications 325
- Integer dependent variable 458
- Integrated series 328

- Integrity (database) [134](#)
- Intercept
 - in pool estimation [564](#)
- Intercept in equation [260](#), [266](#)
- Inverted AR roots [310](#), [317](#)
- Inverted MA roots [317](#)
- Irrelevant variable test [374](#)
- Iteration
 - failure to improve message [669](#)
 - in models [638](#)
- Iteration option [669](#)
 - for pool estimation [565](#)
 - for system estimation [504](#)
 - in nonlinear least squares [293](#)
- J**
- Jarque-Bera statistic [376](#), [405](#)
 - in VAR [524](#)
- J-statistic [300](#)
- J-test [392](#)
- Justification [254](#)
- K**
- Kalman filter [579](#)
- Kernel
 - bivariate fit [238](#)
 - choice in HAC weighting [504](#), [516](#)
 - density estimation [164](#), [229](#)
- Kernel function
 - in density estimation [230](#)
 - in kernel regression [239](#)
- Keyboard
 - data entry [64](#)
 - focus option [647](#)
 - shortcuts [9](#)
- Kolmogorov-Smirnov test [164](#)
- KPSS test [336](#)
- Kruskal-Wallis test [162](#)
- Kullback-Leibler [683](#)
- Kurtosis [153](#)
- Kwiatkowski, Phillips, Schmidt, and Shin test [336](#)
- L**
- Label [41](#), [50](#)
 - automatic update option [651](#)
 - capitalization [49](#)
 - pie graph [211](#)
 - series [174](#)
- Lag
 - dynamic assignment [96](#)
 - series [91](#)
- Lagged dependent variable
 - and serial correlation [303](#)
 - Durbin-Watson statistic [305](#)
- Lagged series in equation [261](#)
- Lagrange multiplier test [305](#)
- Large sample test [367](#)
- Last_update
 - field in database query [129](#)
- Last_write
 - field in database query [129](#)
- Latent variable
 - binary model [422](#)
 - censored models [444](#)
 - ordered models [438](#)
- Lead
 - series [91](#)
- Levene test [163](#)
- Leverage effect [408](#)
- Likelihood [268](#)
- Lilliefors test [164](#)
- Limit points [441](#)
 - make covariance matrix [443](#)
 - make vector [443](#)
 - non-ascending [442](#)
- Limited dependent variable [421](#)
- Line graph [151](#)
- Linked equations in models [619](#)
- List
 - specifying equation by [260](#)
- Ljung-Box Q-statistic [169](#)
 - serial correlation test [305](#)
- LM test
 - ARCH [377](#)
 - artificial regression [678](#)
 - auxiliary regression [376](#)
 - serial correlation [305](#), [376](#)
- Load
 - workfile [37](#)
- Local optimum [668](#)
- Local regression [236](#)

Local weighting option [237](#)

LOESS [236](#), [238](#)

Log likelihood

average [426](#)

censored models [445](#)

exponential [461](#)

for binary models [422](#)

for regression (normal errors) [268](#)

negative binomial [459](#)

normal [461](#)

ordered models [439](#)

Poisson model [459](#)

restricted [426](#)

truncated models [454](#)

Logical expression [90](#)

in easy query [125](#)

Logit models [424](#)

Long name [49](#)

for series [174](#)

LOWESS. See also LOESS

LR statistic [373](#), [426](#), [452](#), [453](#), [462](#)

QLR [466](#)

M

MA specification

backcasting [320](#)

forecast [358](#)

in ARIMA models [311](#)

in two stage least squares [289](#)

Marginal significance level [266](#), [367](#)

Marquardt [665](#)

Match operator in database query [128](#)

Maximize window [7](#)

Maximum likelihood

full information [498](#)

quasi-generalized pseudo-maximum likelihood
[465](#)

quasi-maximum likelihood [460](#)

user specified [471](#)

McFadden R-squared [426](#)

Mean [152](#)

Mean absolute error [353](#)

Mean absolute percentage error [353](#)

Measurement equation [578](#)

Measurement error [283](#), [382](#)

Median [152](#)

Menu [8](#), [48](#)

main [8](#)

objects [49](#)

Merge

graphs [45](#)

store option [114](#)

Micro TSP

opening workfiles [38](#)

Minimize window [7](#)

Missing values [92](#)

handling in estimation [264](#)

in frequency conversion [73](#)

in models [638](#)

recoding [94](#)

Model consistent expectations [631](#)

Models

add factors [601](#), [626](#)

aliasing [603](#), [626](#)

binding variables [603](#)

block structure [623](#)

convergence test [638](#)

creating [618](#)

definition [495](#)

diagnostic messages and iteration history [637](#)

dynamic solution [633](#)

equation view [620](#)

excluding variables [625](#)

extended search [638](#)

Fair-Taylor solution [630](#)

fit option for solution [633](#)

future values [629](#)

Gauss-Seidel solution [672](#)

handling of ARMA terms [633](#)

initialize excluded variables [638](#)

inline equations [619](#)

intercept shift add factor [627](#)

linked equations [619](#)

missing value handling [638](#)

Newton solution [672](#)

overriding variables [603](#), [626](#), [629](#)

properties of equations [621](#)

roundoff of solution [639](#)

scenarios [616](#), [625](#)

simultaneous and recursive blocks [623](#)

solution methods [637](#)

solving [629](#)

solving to match target [640](#)

starting values [638](#)

- static solution [633](#)
- stochastic solution [634](#)
- text description of [624](#)
- text keywords [624](#)
- tracking variables [636](#)
- updating links [620](#)
- variable dependencies [622](#)
- variable shift add factor [627](#)
- variable view [622](#)

Mouse [6](#)

Move window [8](#)

Moving average

- geometric [101](#)

Multicollinearity [276](#)

N

NA [92](#)

NA. See also Missing data.

Nadaraya-Watson [238](#)

Name

- display option [41](#)
- equation [273](#)
- object [49](#)
- reserved [49](#)

Name field in database query [127](#)

Near singular matrix [276](#)

- binary models [428](#)
- nonlinear models [297](#), [668](#)
- polynomial distributed lag [324](#)
- pool estimation [565](#)
- RESET test [384](#)

Nearest neighbor fit [235](#)

Newey-West

- HAC covariance [282](#)
- truncation lag [282](#)

News impact curve [410](#)

Newton's method [637](#), [672](#)

Newton-Raphson [664](#)

Noninvertible MA process [317](#), [321](#)

Nonlinear coefficient restriction

- Wald test [371](#)

Nonlinear least squares [289](#)

- convergence criterion [293](#)
- forecast standard errors [352](#)
- iteration option [293](#)
- specification [291](#)

starting values [292](#)

two stage [295](#)

two stage with AR specification [296](#)

weighted [294](#)

weighted two stage [296](#)

with AR specification [294](#), [308](#)

Nonnested tests [391](#)

Nonstationary time series [328](#)

Normal distribution

- test for [164](#), [376](#)

Normalize formula [97](#)

N-step forecast test [387](#)

Null hypothesis [367](#)

Number format [254](#)

N-way table [219](#)

- chi-square tests [217](#)

O

Object [33](#)

- basics [42](#)
- copy [51](#)
- create [45](#)
- data [42](#)
- delete [51](#)
- freeze [51](#)
- icon [43](#)
- label [50](#)
- naming [50](#)
- open [45](#)
- print [52](#)
- procedure [43](#)
- sample [64](#)
- show [46](#)
- store [52](#)
- type [44](#)
- window [46](#)

Objects menu [49](#)

Observation equation [578](#)

OLS (ordinary least squares)

- adjusted R-squared [267](#)
- coefficient standard error [266](#)
- coefficient t-statistic [266](#)
- coefficients [265](#)
- standard error of regression [268](#)
- sum of squared residuals [268](#)
- system estimation [496](#), [512](#)

Omitted variables test [373](#), [382](#)

- One-step forecast test [387](#)
 - One-way tabulation [166](#)
 - Open
 - database [109](#)
 - multiple objects [45](#)
 - object [45](#)
 - options [38](#)
 - workfile [38](#)
 - Operator [87](#)
 - arithmetic [87](#)
 - comparison [90](#)
 - conjunction (and, or) [90](#)
 - difference [92](#)
 - lag [91](#)
 - lead [91](#)
 - parentheses [88](#)
 - Optimization algorithms
 - BHHH [665](#)
 - convergence criterion [669](#)
 - first derivative methods [665](#)
 - Gauss-Newton [665](#)
 - Goldfeld-Quandt [664](#)
 - grid search [666](#)
 - iteration control [669](#)
 - Marquardt [665](#)
 - Newton-Raphson [664](#)
 - second derivative methods [664](#)
 - starting values [667](#)
 - step size [666](#)
 - Option settings
 - allow only one untitled [648](#)
 - backup files [649](#)
 - button bar [648](#)
 - date notation [648](#)
 - error message display [651](#)
 - fonts [647](#)
 - keyboard focus [647](#)
 - output redirection [651](#)
 - print setup [651](#)
 - program execution mode [650](#)
 - series auto label [651](#)
 - spreadsheet view option [650](#)
 - warn on close [648](#)
 - Or operator [62, 90](#)
 - Order condition for identification [284](#)
 - Order of stacked data [556](#)
 - Ordered dependent variable [438](#)
 - error messages [442](#)
 - log likelihood [439](#)
 - Ordinary residual
 - binary models [434](#)
 - censored models [450](#)
 - count models [463](#)
 - truncated models [456](#)
 - Orthogonality condition [298, 515](#)
 - Output redirection [651](#)
 - Overdispersion [460, 468](#)
 - specification test [463](#)
 - Overidentifying restrictions [300](#)
- P**
- Pack database [134](#)
 - Packable space [110, 134](#)
 - Panel data [551](#)
 - Param (command) [292, 503, 668](#)
 - Parks estimator [565, 575](#)
 - Partial autocorrelation [312](#)
 - function [168](#)
 - Paste [51](#)
 - existing series [67](#)
 - new series [66](#)
 - PcGive data [139](#)
 - PDL (polynomial distributed lag)
 - forecast standard errors [352](#)
 - instruments [325](#)
 - near end restriction [324](#)
 - specification [324](#)
 - Phi coefficient [218](#)
 - Phillips-Perron test [335](#)
 - Pie graph [211](#)
 - Polynomial distributed lag
 - far end restriction [324](#)
 - Polynomial distributed lags [352](#)
 - Pool
 - AR specification [564](#)
 - balanced data [557, 559](#)
 - balanced sample [563](#)
 - coefficient test [570](#)
 - common coefficients [564](#)
 - cross section identifiers [552](#)
 - cross-section specific coefficients [564](#)
 - cross-section weighting [574](#)
 - dummy variable [561](#)
 - export data [558](#)

- fixed effects [564](#), [572](#)
- generate series [560](#)
- identifier comparison with “?” wildcard [660](#)
- import data [553](#)
- iteration option [565](#)
- make group [561](#)
- make system [570](#)
- naming series [552](#)
- order [556](#)
- random effects [564](#), [573](#)
- residuals [570](#)
- series [553](#)
- stacked data [554](#)
- SUR weighting [575](#)
- unstacked data [554](#)
- White heteroskedasticity covariance [565](#), [575](#)
- workfile [551](#)

- Pool (object)
 - ? placeholder [553](#)
 - base name [552](#)
 - convergence criterion [565](#)
 - copy [552](#)
 - create [552](#)

- Postscript file [252](#)

- Prais-Winsten [310](#)

- Precedence of evaluation [88](#)

- Predetermined variable [283](#)

- Prediction table
 - binary models [429](#)
 - ordered models [442](#)

- Prewhitening [504](#), [518](#)

- Principal components [219](#)

- Print [52](#)
 - redirection [651](#)
 - selected [52](#)
 - setup options [651](#)

- Probability response curve [435](#)

- Probit models [424](#)

- Procedure [43](#)

- Program
 - execution option [650](#)

- p-value [367](#)
 - for coefficient t-statistic [266](#)

Q

- QML [460](#)

- QQ-plot (quantile-quantile) [164](#), [227](#)

- multiple [213](#)

- Q-statistic

- Ljung-Box [169](#)

- residual serial correlation test [524](#)

- serial correlation test [305](#)

- Quadratic hill-climbing [664](#)

- Quadratic spectral kernel [517](#)

- Qualitative dependent variable [421](#)

- Quantile [226](#)

- Quasi-generalized pseudo-maximum likelihood [465](#)

- Quasi-likelihood ratio test [462](#), [466](#)

- Quasi-maximum likelihood [460](#)

- robust standard errors [467](#)

- Queries on database [123](#)

- advanced query [126](#)

- DRI [145](#)

- easy query [124](#)

- examples [130](#)

- logical expressions [125](#)

- wildcard characters [124](#)

- Quiet mode [650](#)

R

- Ramsey RESET test [382](#)

- Random effects [564](#), [573](#)

- Random walk [328](#)

- Randomize ties [432](#)

- Rank condition for identification [284](#)

- Ratio to moving-average [184](#)

- RATS data

- 4.x native format [139](#)

- portable format [140](#)

- Read [553](#)

- Rebuild database [135](#)

- Recursive coefficient [388](#)

- save as series [388](#)

- Recursive estimation

- least squares [384](#)

- Recursive residual [384](#), [385](#)

- CUSUM [385](#)

- CUSUM of squares [386](#)

- n-step forecast test [387](#)

- one-step forecast test [387](#)

- save as series [388](#)

- Redirect output to file [651](#)

- Redundant variables test [374](#)
- Registry [121](#)
- Regression
 - adjusted R-squared [267](#)
 - coefficient standard error [266](#)
 - collinearity [276](#)
 - forecast [343](#)
 - F-statistic [269](#)
 - log likelihood [268](#)
 - residuals from [274](#)
 - standard error of [268](#)
 - sum of squared residuals [268](#)
 - t-statistic for coefficient [266](#)
- Remarks
 - field in database query [130](#)
- Remove [249](#)
- Rename [49](#)
 - database [133](#)
 - objects in database [117](#)
- Repair database [135](#)
- Representation view
 - of equation [272](#)
- Reserved names [49](#)
- RESET test [382](#)
- Residuals
 - default series RESID [274](#)
 - from estimated equation [274](#)
 - from two stage least squares [285](#)
 - generalized [434](#), [450](#), [456](#), [463](#)
 - make series [273](#)
 - of equation [272](#)
 - ordinary [434](#), [450](#), [456](#), [463](#)
 - recursive [384](#), [385](#)
 - standardized [272](#), [434](#), [450](#), [456](#), [463](#)
 - sum of squares [268](#)
 - symmetrically trimmed [452](#)
 - tests of [375](#)
 - unconditional [304](#), [309](#)
- Resize window [8](#)
- Restore window [7](#)
- Restricted estimation [262](#)
- Restricted log likelihood [426](#)
- Results
 - accessing from estimated equation [270](#)
- Rich text format [254](#)
- Robust standard errors
 - GLM [468](#)
 - Huber/White (QML) [467](#)
- Robustness iterations
 - with nearest neighbor fit [237](#)
 - with regression [235](#)
- Root mean squared error [353](#)
- R-squared
 - adjusted [267](#)
 - for regression [267](#)
 - from two stage least squares [286](#)
 - McFadden [426](#)
 - negative [403](#)
 - uncentered [377](#), [378](#)
 - with AR specification [310](#)
- RTF [254](#)
- S**
- Sample
 - @all [61](#), [655](#)
 - @first [61](#), [655](#)
 - @last [61](#), [655](#)
 - adjustment in estimation [264](#)
 - balanced [563](#)
 - change [61](#)
 - command [64](#)
 - common [93](#)
 - current [36](#)
 - if condition [62](#)
 - individual [93](#)
 - selection and missing values [63](#)
 - specifying sample object [64](#)
 - used in estimation [263](#)
 - with expressions [61](#), [655](#)
 - workfile [60](#)
- SAR specification [316](#)
- Save
 - options [38](#)
 - save as new workfile [37](#)
 - workfile [36](#)
- Scale factor [449](#)
- Scatter diagrams [209](#), [233](#)
 - matrix [212](#)
 - multiple [211](#)
 - simple [209](#), [233](#)
 - with kernel fit [238](#)
 - with nearest neighbor fit [235](#)
 - with regression [209](#), [233](#), [234](#)
- Scenarios in models [616](#)

- Schwarz criterion [683](#)
 - for equation [269](#)
- Score vector [435](#)
- Scroll window [6](#)
- Seasonal
 - ARMA terms [316](#)
 - difference [92](#), [314](#)
- Seasonal adjustment
 - additive [185](#)
 - Census X11 (historical) [184](#)
 - Census X12 [177](#)
 - multiplicative [184](#)
 - Tramo/Seats [185](#)
- Seasonal graphs [152](#)
- Second derivative methods [664](#)
- Seemingly unrelated regression [497](#)
- Select
 - all [45](#)
 - multiple items [8](#)
 - object [45](#)
 - single item [8](#)
- sensitivity [430](#)
- Serial correlation
 - ARIMA models [311](#)
 - Durbin-Watson statistic [268](#), [304](#)
 - first order [304](#)
 - higher order [304](#)
 - nonlinear models [308](#)
 - tests [304](#), [376](#)
 - theory [303](#)
 - two stage regression [309](#)
- Series
 - bar graph view [151](#)
 - create [55](#), [94](#)
 - delete observation [58](#)
 - descriptive statistics [152](#)
 - difference [92](#)
 - edit in differences [200](#)
 - editing [56](#)
 - in pool objects [553](#)
 - insert observation [58](#)
 - label [174](#)
 - lag [91](#)
 - lead [91](#)
 - line graph view [151](#)
 - seasonal graph view [152](#)
 - smpl + /- [57](#)
 - spike graph view [152](#)
 - spreadsheet view [151](#), [650](#)
- Shadowing of object names [132](#)
- Sharing violation [111](#)
- Show [46](#)
- Siegel-Tukey test [163](#)
- Sign test [158](#)
- Simple hypothesis tests [156](#)
 - mean [156](#)
 - median [158](#)
 - variance [157](#)
- Simple scatter plot [209](#), [233](#)
- Skewness [153](#)
- SMA specification [316](#)
- Smoothing
 - methods [190](#)
 - parameters [190](#)
- Smpl command [64](#)
- Smpl + /- [57](#)
- Solve
 - Gauss-Seidel [672](#)
- Sort workfile [39](#)
- Source
 - field in database query [130](#)
- Sparse label option [154](#), [217](#)
- Specification test
 - for binary models [437](#)
 - for overdispersion [463](#)
 - for tobit [452](#)
 - RESET (Ramsey) [382](#)
 - White [379](#)
- Specificity [430](#)
- Specify an equation [260](#)
- Specify equation [367](#)
 - in systems [499](#)
 - nonlinear [291](#)
- Spike graph [152](#)
- Spreadsheet
 - file export [71](#)
 - file import [68](#)
 - view option [650](#)
- Stability test [379](#)
 - Chow breakpoint [380](#)
 - Chow forecast [381](#)
 - RESET [382](#)
 - with unequal variance [389](#)

- Stacked data [554](#)
 - balanced [557](#)
 - descriptive statistics [559](#)
 - order [556](#)
 - Standard deviation [153](#)
 - Standard error
 - for estimated coefficient [266](#)
 - for fixed effects [573](#)
 - forecast [351](#), [362](#)
 - of the regression [268](#)
 - Standardized residual [272](#), [407](#)
 - binary models [434](#)
 - censored models [450](#)
 - count models [463](#)
 - truncated models [456](#)
 - Start
 - field in database query [129](#)
 - Starting values
 - (G)ARCH models [401](#)
 - binary models [427](#)
 - for ARMA estimation [319](#)
 - for nonlinear least squares [292](#)
 - for systems [503](#)
 - param statement [292](#)
 - param statements [668](#)
 - user supplied [320](#)
 - Starting values for coefficients [292](#), [667](#)
 - State equation [578](#)
 - State space [577](#)
 - @mprior [587](#)
 - @vprior [587](#)
 - estimation [591](#)
 - observation equation [578](#)
 - representation [577](#)
 - state equation [578](#)
 - State variables [577](#)
 - Static forecast [356](#)
 - Stationary time series [328](#)
 - Status line [11](#)
 - Step size [666](#)
 - Store [52](#)
 - as .DB? file [113](#)
 - from pool [562](#)
 - in database [113](#)
 - merge objects [114](#)
 - Structural change [379](#)
 - Structural forecast [357](#)
 - Structural solution of models [633](#)
 - Structural VAR [531](#)
 - estimation [536](#)
 - factorization matrix [525](#)
 - identification [535](#)
 - long-run restrictions [533](#)
 - short-run restrictions [532](#)
 - Subset a workfile [39](#)
 - Sum of squared residuals
 - for regression [268](#)
 - Summary statistics
 - for regression variables [267](#)
 - SUR [497](#), [513](#)
 - in pool estimation [565](#), [575](#)
 - Survivor function [226](#)
 - Symmetrically trimmed residuals [452](#)
 - System
 - create [498](#)
 - definition [495](#)
 - estimation [496](#)
 - estimation methods (technical) [511](#)
 - specification [499](#)
- ## T
- Table [253](#)
 - as unformatted text [255](#)
 - column width [254](#)
 - copy to other windows programs [254](#)
 - delete cells [253](#)
 - edit [254](#)
 - font [253](#)
 - horizontal line [254](#)
 - insert cells [253](#)
 - number format [254](#)
 - objects [253](#)
 - title [254](#)
 - views [253](#)
 - Tabulation
 - n-way [216](#)
 - one-way [166](#)
 - Template [249](#)
 - Tests. See also Hypothesis tests, Specification test and Goodness of fit.
 - Text (object) [255](#)
 - Theil inequality coefficient [353](#)
 - Three stage least squares [497](#), [514](#)
 - Threshold ARCH (TARCH) [408](#)

- component model [413](#)
- forecast [409](#)
- Title [254](#)
- Title bar [10](#), [36](#), [47](#)
- to (lag range) [261](#)
- Tobit [445](#)
- Toggling [9](#)
- Toolbar [36](#), [47](#)
- Tracking model variables [636](#)
- Tramo/Seats [185](#)
- Transition equation [578](#)
- Transpose [199](#)
- Trend series [98](#)
- Truncated dependent variable [454](#)
 - fitted index [456](#)
 - log likelihood [454](#)
- Truncation point [455](#)
- TSD data format [139](#)
- TSP portable data format [140](#)
- Two stage least squares [289](#)
 - in systems [497](#)
 - nonlinear [295](#)
 - nonlinear with AR specification [296](#)
 - order condition [284](#)
 - rank condition [284](#)
 - residuals [285](#)
 - system estimation [514](#)
 - weighted [286](#)
 - weighted in systems [497](#), [514](#)
 - weighted nonlinear [296](#)
 - with AR specification [286](#), [309](#)
 - with MA specification [289](#)
- Type
 - field in database query [128](#)

U

- Unconditional residual [309](#)
- Unit root test [170](#), [329](#)
 - augmented Dickey-Fuller [333](#)
 - Dickey-Fuller [333](#)
 - Dickey-Fuller GLS detrended [334](#)
 - Elliot, Rothenberg, and Stock [337](#)
 - KPSS [336](#)
 - Phillips-Perron [335](#), [336](#)
 - trend assumption [334](#)
- Units

- field in database query [130](#)
- Unstacked data [554](#)
- Untitled [49](#), [50](#)
 - allow only one option [648](#)
- Update
 - coefficient vector [273](#), [668](#)
 - group [199](#)
- Urzua factorization matrix [525](#)
- User supplied starting values [320](#)

V

- Van der Waerden test [158](#), [162](#)
- VAR
 - AR roots [522](#)
 - autocorrelation LM test [524](#)
 - decomposition [529](#)
 - estimation output [521](#)
 - factorization matrix in normality test [524](#)
 - Granger causality test [522](#)
 - impulse response [527](#)
 - Jarque-Bera normality test [524](#)
 - lag exclusion test [523](#)
 - lag length choice [523](#)
 - mathematical model [519](#)
 - See also Impulse response.
 - See also Structural VAR.
- Variance decomposition [529](#)
- Variance factor [468](#)
- Variance proportion [353](#)
- VEC [547](#)
 - estimating [547](#)
- Vector autoregression [519](#)
- Verbose mode [650](#)
- View
 - default [45](#)
- Volatility [398](#)

W

- Wald test [368](#)
 - coefficient restriction [368](#)
 - formula [371](#)
 - F-statistic [372](#)
 - joint restriction [369](#)
 - nonlinear restriction [371](#)
 - structural change with unequal variance [389](#)
- Warn on close option [648](#)

Watson test [164](#)

Weighted least squares [281](#)
cross-equation weighting [496](#)
nonlinear [294](#)
nonlinear two stage [296](#)
pool [565](#)
system estimation [512](#)
two stage [286](#)
two stage in systems [497](#), [514](#)

Weighting matrix
cross-section [574](#)
for GMM [298](#), [516](#)
heteroskedasticity and autocorrelation consistent [516](#)
kernel options [516](#)
White heteroskedasticity consistent [516](#)

White
heteroskedasticity consistent covariance [281](#),
[565](#), [575](#)

Width of table column [254](#)

Wilcoxon signed ranks test [158](#), [162](#)

Wildcard characters [40](#), [657](#)
in easy query [124](#)

Window
active [6](#), [47](#)
database [109](#)
EViews main [9](#)
object [49](#)
scrolling [6](#)
size control [8](#)

Work area [12](#)

Workfile
automatic backup [649](#)
basics [33](#)
create [35](#)
directory [36](#)
extract subset [39](#)
frequency [34](#), [653](#)
range [34](#)
resize [38](#)
sample [60](#)
save [36](#)
sort [39](#)
window [35](#)

Write [558](#)

X

X11 (historical) [184](#)
limitations [184](#)
X11 using X12 [178](#)
X12 [177](#)
XY-line
view [210](#), [213](#)

Y

Yates' continuity correction [162](#)

