Financial Econometrics

With Eviews

Roman Kozhan

bookboon
The eBook company

Roman Kozhan

Financial Econometrics

Financial Econometrics – with EViews

ISBN 978-87-7681-427-4

To my wife Nataly

Contents

Preface

The aim of this textbook is to provide a step-by-step guide to financial econometrics using EViews 6.0 statistical package. It contains brief overviews of econometric concepts, models and data analysis techniques followed by empirical examples of how they can be implemented in EViews.

This book is written as a compendium for undergraduate and graduate students in economics and finance. It also can serve as a guide for researchers and practitioners who desire to use EViews for analysing financial data. This book may be used as a textbook companion for graduate level courses in time series analysis, empirical finance and financial econometrics.

It is assumed that the reader has a basic background in probability theory and mathematical statistics

The material covered in the book includes concepts of linear regression, univariate and multivariate time series modelling and their implementation in EViews. Chapter 1 briefly introduces commands, structure and programming language of the EViews package. Chapter 2 provides an overview of the regression analysis and its inference. Chapters 3 to 5 cover some topics of univariate time series analysis including linear models, GARCH models of volatility, unit root tests. Chapter 6 introduces modelling of multivariate time series.

Chapter 1

Introduction to EViews 6.0

EViews is a simple, interactive econometrics package which proves many tools used in econometrics. It provides users with several convenient ways of performing analysis including a Windows and a command line interfaces. Many operations that can be implemented using menus may also be entered into the command window, or placed in programs for batch processing. The possibility of using interactive features like windows, buttons and menus makes EViews a user-friendly software.

In this chapter we briefly introduce you main features of the language, will show you the use of some important commands which will be used further in this textbook. We will start with the interactive Windows interface and then go into more detailed description about the EViews' batch processing language and advanced programming features.

1.1 Workfiles in EViews

EViews' design allows you to work with various types of data in an intuitive and convenient way. We start with the basic concepts of how to working with datasets using workfiles, and describing simple methods to get you started on creating and working with workfiles in EViews.

In the majority of cases you start your work in EViews with a workfile – a container for EViews objects. Before you perform any tasks with EViews' objects you first have to either create a new workfile or to load an existing workfile from the disc.

In order to create a new workfile you need to provide and information about its structure. Select *File/New/Workfile* from the main menu to open the *Workfile Create* dialog. On the left side of the dialog is a combo box for describing the underlying structure of your dataset. You have to choose between three options regarding the structure of your data – the *Dated - regular frequency*, the *Unstructured*, and the *Balanced Panel* settings. *Dated - regular frequency* is normally used to work with a simple time series data, *Balanced Panel* is used for a simple panel dataset and *Unstructured* options is used for all other cases.

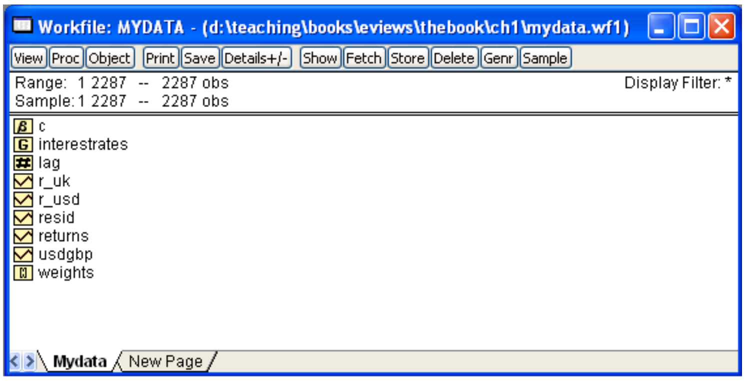
For the *Dated - regular frequency*, you may choose among the following options: Annual, Semi-annual, Quarterly, Monthly, Weekly, Daily - 5 day week, Daily - 7 day week and Integer date. EViews will also ask you to enter a Start date and End date for your workfile. When you click on OK, EViews will create a regular frequency workfile with the specified number of observations and the associated identifiers.

The *Unstructured data* simply uses integer identifiers instead of date identifiers. You would use this type of workfile while performing a crossectional analysis. Under this option you would only need to enter the number of observations.

The Balanced Panel entry provides a method of describing a regular frequency panel data structure. Panel data is the term that we use to refer to data containing observations with both a group (cross-section) and time series identifiers. This entry may be used when you wish to create a balanced structure in which every crosssection follows the same regular frequency with the same date observations. Under this option you should specify a desired Frequency, a Start and End date, and Number of cross sections.

Another method of creating an EViews workfile is to open a non-EViews data source and to read the data into an new EViews workfile. To open a foreign data source, first select *File/Open/Foreign Data as Workfile.* First, EViews will open a series of dialogs asking you to describe and select data to be read. The data will be read into the new workfile, which will be resized to fit. If there is a single date series in the data, EViews will attempt to restructure the workfile using the date series. A typical workfile view is given in Figure 1.1.

Figure 1.1: Workfile in EViews



Workfiles contain the EViews objects and provide you an access to your data and tools for working with this data.

Below the titlebar of a workfile is a button bar that provides you with easy access to some useful workfile operations. These buttons are simply shortcuts to items that may be accessed from the main EViews menu. Below the toolbar are two lines of status information where EViews displays the range of the workfile, the current sample (the range of observations that are to be used in calculations), and the display filter (rule used in choosing a subset of objects to display in the workfile window). You may change the range, sample, and filter by double clicking on these labels and entering the relevant information in the dialog boxes. The contents of your workfile page is provided in in the workfile directory. You can find there all named objects, sorted by name, with an icon showing the object type.

Push the *Save* button on the workfile toolbar to save a copy of the workfile on disk. You can also save a file using the *File/ Save As* or *File/Save* choices from the main menu. By default, EViews will save your data in the EViews workfile format, the extension ".wf1". You may also choose to save the data in your workfile in a foreign data format by selecting a different format in the combo box.

When you click on the *Save* button, EViews will display a dialog showing the current global default options for saving the data in your workfile. You should choose between saving your series data in either *Single precision* or *Double precision*. Single precision will create smaller files on disk, but saves the data with fewer digits of accuracy (7 versus 16). You may also choose to save your data in compressed or non-compressed form.

1.2 Objects

All 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. Together with the data information, EViews also associates procedures which can be used to process the data. For example, an equation object contains all of the information relevant to an estimated relationship, 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.

Each object contains various types of information. For example, series, matrix, vector, and scalar objects contain numeric data while equations and systems contain complete information about the specification of the equation or system, the estimation results. Graphs and tables contain numeric, text, and formatting information. Since objects contain various kinds of data, you will work with different objects in different ways.

EViews provides you with different tools for each object. These tools are views and procedures which often display tables or graphs in the object's window. Using procedures you can create new objects. For example, equation objects contain procedures for generating new series containing the residuals, fitted values, or forecasts from the estimated equation. You select procedures from the *Proc* menu and views from the *View* on the object's toolbar or from the EViews main menu.

There are 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 workfile window. The basic object icons are:

Figure 1.2: Object Icons

In order to create an object, create or loaded a workfile first and then select *Object/New Object* from the main menu. You will see the *New Object* dialog box where you can click on the type of object you want to create. For some object types, a second dialog box will open prompting you to describe your object in more detail. For example, if you select *Equation*, you will see a dialog box prompting you for additional information.

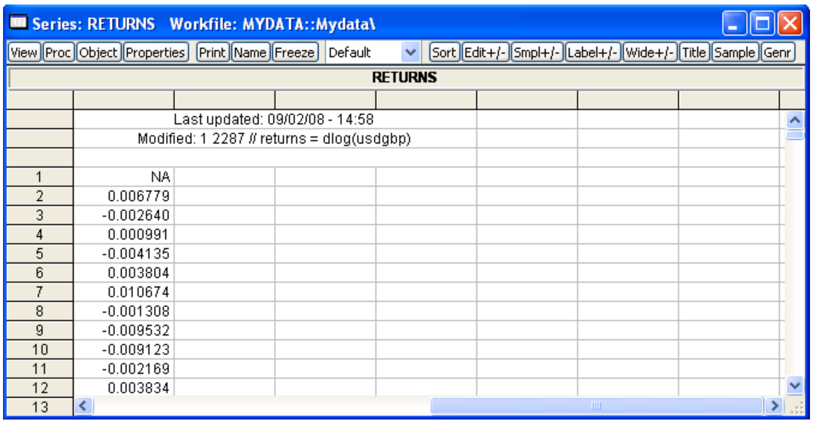
Once you have selected your object, you can open it by double clicking anywhere in the highlighted area. If you double click on a single selected object, you will open an object window. If you select multiple graphs or series and double click, a pop-up menu appears, giving you the option of creating 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 will be merged into a single graph and displayed in a single 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 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.

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.

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.

Let us look again at a typical object window:

Figure 1.3: Object Window in EViews



Here, we see the series window for RETURNS. 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. There are several buttons that are found on all object toolbars:

- *View* button lets you change the view that is displayed in the object window. The available choices will differ, depending upon the object type.

- *Proc* button provides access to a menu of procedures that are available for the object.

- *Object* button lets you manage your objects. You can store the object on disk, name, delete, copy, or print the object.

- *Print* button lets you print the current view of the object (the window contents).

- *Name* button allows you to name or rename the object.

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

There are two distinct methods of duplicating the information in an object: copying and freezing. If you select *Object/Copy* from the menu, EViews will create 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 may also copy an object from the workfile window. Simply highlight the object and click on *Object/Copy Selected* or right mouse click and select *Object/Copy*, then specify the destination name for the object.

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. 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 shows a snapshot of the object at the moment you pushed the button. The primary feature of freezing an object is that the tables and graphs created by freezing may be edited for presentations or reports. Frozen views do not change when the workfile sample or data change.

To delete an object or objects from your workfile, select the object or objects in the workfile directory and click *Delete* or *Object/Delete Selected* on the workfile toolbar.

Series

An series object contains a set of observations on a numeric variable. Associated with each observation in the series is a date or observation label. Note that the series object may only be used to hold numeric data. If you wish to work with alphanumeric data, you should employ *alpha* series.

You can create a numeric series by selecting *Object/New Object* from the menu, and then to select *Series*. EViews will open a spreadsheet view of the new series object with all of the observations containing "NA" (the missing value). You may then edit or use expressions to assign values for the series. A second method of creating a series is to generate the series using mathematical expressions. Click on *Quick/Generate Series* in the main EViews menu, and enter an expression defining the series.

Lastly, you may create the series by entering a series command in the command window. Entering an expression of the form:

```
series returns=expression
```

creates a series with the name returns and assigns the expression to each observation.

You can edit individual values of the data in a series. First, open the spreadsheet view of the series. Next, make certain that the spreadsheet window is in edit mode (you can use the *Edit +/–* button on the toolbar to toggle between edit mode and protected mode). To change the value for an observation, select the cell, type in the value, and press ENTER.

You can also insert and delete observations in the series. First, click on the cell where you want the new observation to appear. Next, right click and select *Insert Obs* or *Delete Obs* from the menu. You will see a dialog asking how many observations you wish to insert or delete at the current position and whether you wish to insert observations in the selected series or in all of the series in the group. If you choose to insert a single 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

A group is a list of series names 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 using the group object, rather than each one of the individual series. 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 would normally 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.

There are several ways to create a group. Perhaps the easiest method is to select *Object/New Object* from the main menu or workfile toolbar, click on *Group*. You should enter the names of the series to be included in the group, separated by spaces, and then click OK. A group window will open showing a spreadsheet view of the group.

If you apply an operation to a group, EViews will automatically evaluate the expressions for each observation and display the results as if they were an ordinary series.

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 list of series, groups and series expressions to be included in the group. 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.

Samples

One of the most important concepts in EViews is the sample of observations. The sample is the set of observations in the workfile used for performing statistical procedures. Samples may be specified using ranges of observations and "if conditions" that observations must satisfy to be included. For example, you can tell EViews that you want to work with observations from 1973M1 to 1990M12 and 1995M1 to 20066M12. Or you may want to work with data from 1973M1 to 1978M12 where observations in the Returns series are positive. When you create a workfile, the 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. You can always determine the current workfile sample of observations by looking at the top of your workfile window. Here the MYDATA workfile consists of 408 observations from January 1973 to December 2006. The current workfile sample uses a subset of those 72 observations between 1973M01 and 1978M12 for which the value of the Returns series is positive.

There are four ways to set the workfile sample: you may click on the *Sample* button in the workfile toolbar, you may double click on the sample string display in the workfile window, you can select *Proc/Set Sample* from the main workfile menu, or you may enter a smpl command in the command window.

EViews provides special keywords that may make entering sample date 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 typing "1973M12006M12". Furthermore, you may use @first and @last to refer to the first and last observation in the workfile. Thus, the three sample specifications for the above workfile:

@all

@first 2006m12

19733m1 @last

are identical. [1]

[1] You may use the IEEE standard format, "YYYY-MM-DD", which uses a four-digit year, followed by a dash, a two-digit month, a second dash, and a two-digit day. The presence of a dash in the format means that you must enclose the date in quotes for EViews to accept this format. For example: "1991-01-03" "1995-07-05" will always be interpreted as January 3, 1991 and July

Sample Commands

EViews allows you to add conditions to the sample specification. In this case 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. This can be done by typing the expression:

```
smpl 1973m1 1978m12 if returns>0
```

in the command window. You should see the sample change in the workfile window.

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 regular frequency monthly workfile above, the sample string: 1973m1 1973m1+11 defines a sample that includes the 12 observations in the calendar year beginning in 1973M1. The offsets are perhaps most useful when combined with the special keywords to trim observations from the beginning or end of the sample. For example, to drop the first observation in your sample, you may use the sample statement:

```
smpl @first+1 @last
```

Accordingly, the following commands generate a cumulative returns series from the price levels one:

```
smpl @first @first
series returns = 0
smpl @first+1 @last
returns = returns(-1) + log(price) - log(price(-1))
```

The first two commands initialize the cumulative returns series at 0, the last two commands compute them recursively all remaining dates. Later we will see how sample offsets can be used to perform the rolling window estimation.

EViews provides you with a method of saving sample information in an object which can then be referred to by name. To create a sample object, select *Object/New Object* from the main menu or the workfile toolbar. When the *New Object* dialog appears, select *Sample* and, optionally provide a name. Click on OK and EViews will open the sample object specification dialog which you should fill out. The sample object now appears in the workfile directory with a double-arrow icon. To declare a sample object using a command, simply issue the sample declaration, followed by the name to be given to the sample object, and then the sample string:

5, 1995.

```
sample mysample 1973m1 1978m12 if returns>0
```

EViews will create the sample object MYSAMPLE which will use observations between 1973:01 and 1978:12, where the cumulative returns are positive.

You may 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. You can set the workfile sample using the sample object, by entering the smpl command, followed by the sample object name. For example, the command:

```
smpl mysample
```

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

1.3 Eviews Functions

1.3.1 Operators

All of the operators described below may be used in expressions involving series and scalar values. When applied to a series expression, the operation is performed for each observation in the current sample.

Table 1.1: Operators

Expression	Operator Description
+	add, x+y, adds the contents of X and Y
–	subtract, x–y, subtracts the contents of Y from X
*	multiply, x*y, multiplies the contents of X by Y
/	divide, x/y, divides the contents of X by Y
^	raise to the power, x^y, raises X to the power of Y
>	greater than, x>y, takes the value 1 if X exceeds Y, and 0 otherwise
<	less than, x<y, takes the value 1 if Y exceeds X, and 0 otherwise
=	equal to, x=y, takes the value 1 if X and Y are equal, and 0 otherwise
<>	not equal to, x<>y, takes the value 1 if X and Y are not equal, and 0 if they are equal
<=	less than or equal to, x<=y, takes the value 1 if X does not exceed Y, and 0 otherwise
>=	greater than or equal to, x>=y, takes the value 1 if Y does not exceed X, and 0 otherwise
and	logical and, x and y, takes the value 1 if both X and Y are nonzero, and 0 otherwise
or	logical or, x or y, takes the value 1 if either X or Y is nonzero, and 0 otherwise

1.3.2 Basic Mathematical Functions

The following functions perform basic mathematical operations. When applied to a series, they return a value for every observation in the current sample. When applied to a matrix object, they return a value for every element of the matrix object.

Table 1.2: Mathematical Functions

Function	Function Description
@abs(x)	absolute value @abs(-3)=3
@ceiling(x)	smallest integer not less than X, @ceiling(2.34)=3
@exp(x)	exponential, @exp(1)=2.71813
@floor(x)	largest integer not greater than X, @floor(1.23)=1
@iff(s,x,y)	returns X if condition S is true; otherwise returns Y
@inv(x)	reciprocal, @inv(2)=0.5 (For series or scalars only)
@log(x)	natural logarithm, @log(2)=0.693...
@log10(x)	base-10 logarithm
@logx(x,b)	base-b logarithm
@nan(x,y)	returns X if X<> NA, and Y if X=NA
@round(x)	rounds to the nearest integer @round(-97.5)=-98, @round(3.5)=4
@sqrt(x)	square root, @sqrt(9)=3

Time Series Functions The following functions facilitate working with time series data.

1.3.3 Statistical functions

These functions compute descriptive statistics for a specified sample, excluding missing values if necessary. The default sample is the current workfile sample. If you are performing these computations on a series and placing the results into a series,

Table 1.3: Time Series Functions

Function	Function Description
(-k)	k-lag operator
(+k)	k-lead operator
d(x)	first difference
d(x,n)	n-th order difference
d(x,n,s)	n-th order difference with a seasonal difference at S
dlog(x)	first difference of the logarithm
dlog(x,n)	n-th order difference of the logarithm
dlog(x,n,s)	n-th order difference of the logarithm with a seasonal difference at S

you can specify a sample as the last argument of the descriptive statistic function, either as a string (in double quotes) or using the name of a sample object.

Statistical Functions

Function	Function Description
@cor(x,y[,s])	correlation between X and Y
@cov(x,y[,s])	covariance between X and Y
@inner(x,y[,s])	inner product of X and Y
@obs(x[,s])	number of non-missing observations for X in the current sample
@nas(x[,s])	number of missing observations for X in the current sample
@mean(x[,s])	average of the values in X
@median(x[,s])	computes the median of the X
@min(x[,s])	minimum of the values in X
@max(x[,s])	maximum of the values in X
@quantile(x,q[,s])	q-th quantile of the series X
@ranks(x[,o,t,s])	rank the ranking of each observation in X. The order of ranking is set using o: "a" (ascending - default) or "d" (descending). Ties are broken according to the setting of t: "i" (ignore), "f" (first), "l" (last), "a" (average - default), "r" randomize
@stdev(x[,s])	standard deviation of the values in X
@var(x[,s])	variance of the values in X
@skew(x[,s])	skewness of values in X
@kurt(x[,s])	kurtosis of values in X
@sum(x[,s])	sum of the values in X
@prod(x[,s])	product of the values in X
@sumsq(x[,s])	sum of the squares of the values in X
@cumsum(x[,s])	sum of the values in X from the start of the sample to the current observation
@cumprod(x[,s])	product of the values in X from the start of the sample to the current observation
@cummean(x[,s])	mean of the values in X from the start of the sample to the current observation
@cumstdev(x[,s])	standard deviation of the values in X from the start of the sample to the current observation
@cumvar(x[,s])	variance of the values in X from the start of the sample to the current observation
@cumsumsq(x[,s])	sum-of-squares of the values in X from the start of the sample to the current observation
@movsum(x,n)	n-period backward moving sum of X for the current and previous n-1 observations
@movav(x,n)	n-period backward moving average of X for the current and previous n-1 observations
@movstdev(x,n)	n-period backward moving standard deviation of X for the current and previous n-1 observations
@movvar(x,n)	n-period backward moving variance of X for the current and previous n-1 observations
@movcov(x,y,n)	n-period backwards moving covariance between X and Y of the current and previous n-1 observations
@movcor(x,y,n)	n-period backwards moving correlation between X and Y of the current and previous n-1 observations
@movsumsq(x,n)	n-period backwards sum-of-squares of X for the current and previous observations

1.3.4 Statistical Distribution Functions

The following set of functions gives you a possibility to compute and use within your analysis values of density functions, cumulative distribution, quantile functions, and random number generators for a variety of statistical distributions.

Table 1.4: Statistical Distribution Functions

This tables provides cumulative, density, quantile functions and the random number generator functions respectively for the following distributions

Distribution	Function Description
Beta $\beta(a,b)$	@cbeta(x,a,b), @dbeta(x,a,b), @qbeta(p,a,b), @rbeta(a,b)
Binomial $B(n,p)$	@cbinom(x,n,p), @dbinom(x,n,p), @qbinom(s,n,p), @rbinom(n,p)
Chi-square $\chi^2(v)$	@cchisq(x,v), @dchisq(x,v), @qchisq(p,v), @rchisq(v)
Exponential $E(m)$	@cexp(x,m), @dexp(x,m), @qexp(p,m), @rexp(m)
F-distribution $F(v1,v2)$	@cfdist(x,v1,v2), @dfdist(x,v1,v2), @qfdist(p,v1,v2), @rfdist(v1,v1)
Gamma $\Gamma(b,r)$	@cgamma(x,b,r), @dgamma(x,b,r), @qgamma(p,b,r), @rgamma(b,r)
Laplace	@claplace(x), @dlaplace(x), @qlaplace(x), @rlaplace
Log-normal $LN(m,s)$	@clognorm(x,m,s), @dlognorm(x,m,s), @qlognorm(p,m,s), @rlognorm(m,s)
Negative Binomial $NB(n,p)$	@cnegbin(x,n,p), @dnegbin(x,n,p), @qnegbin(s,n,p), @rnegbin(n,p)
Normal $N(0,1)$	@cnorm(x), @dnorm(x), @qnorm(p), @rnorm, nrnd
Poisson $P(m)$	@cpoisson(x,m), @dpoisson(x,m), @qpoisson(p,m), @rpoisson(m)
Pareto	@cpareto(x,k,a), @dpareto(x,k,a), @qpareto(p,k,a), @rpareto(k,a)
Student t-distribution $t(v)$	@ctdist(x,v), @dtdist(x,v), @qtdist(p,v), @rtdist(v)
Uniform $U(a,b)$	@cunif(x,a,b), @dunif(x,a,b), @qunif(p,a,b), @runif(a,b), rnd
Weibull $W(m,a)$	@cweib(x,m,a), @dweib(x,m,a), @qweib(p,m,a), @rweib(m,a)

1.4 Programming in Eviews

On addition to the interactive part of the EViews, where you use the menu commands, windows and graphical interface, you can use programming language to perform your analysis. There are two ways of using the EViews batch language – either enter and edit commands in the command window, or create programs. A program is simply a text file containing EViews commands. Each command in the program will be executed in the order that it appears in the program. Using programs allows you to use looping, conditioning and subroutine processing.

In order to create a program file in EViews, select *File/New/Program* from the main menu. EViews will open an untitled program window where you can enter your commands. You can save the program by clicking on the *Save* or *Save As* button. EViews will add the extension ".PRG" to the name you provide.

To load a program previously saved on disk, click on *File/Open/Program*, navigate to the appropriate directory, and click on the desired name. Alternatively, from the command line, you may type `open` followed by the full program name, including the file extension ".prg". If necessary, include the full path to the file. The entire name should be enclosed in quotations if necessary.

A program consists of a one or more lines of text. Since each line of a program corresponds to a single EViews command, simply enter the text for each command and terminate the line by pressing the **Enter** key.

There are several ways to execute a program. The easiest method is to execute your program by pushing the *Run* button on a program window. The *Run* dialog opens, where you can enter the program name and supply arguments. You may use the radio buttons to choose between Verbose and Quiet modes. In verbose mode, EViews sends messages to the status line and continuously updates the workfile window as objects are created and deleted. Quiet mode suppresses these updates, reducing the time spent writing to the screen.

By default, when EViews encounters an error, it will immediately terminate the program and display a message. If you enter a number into the Maximum errors before halting field, EViews will continue to execute the program until the maximum number of errors is reached (unless there is a serious error occurred).

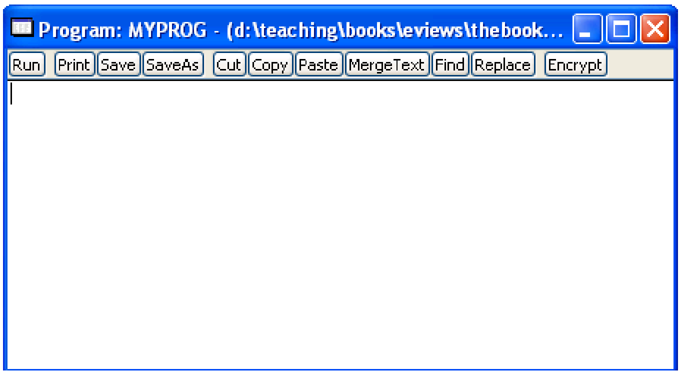
You may also execute a program by entering the `run` command, followed by the name of the program file:

```
run mysp500 or run c:\eviews\myprog
```

Simple Programs

The simplest program is just a list of commands. Execution of the program is equivalent to typing the commands one by one into the command window. Entering commands in the program file has the advantage that you can save the set of commands for later use, and execute the program repeatedly, making minor modifications each time. Let us look at a simple example. Create a new program by typing program MYPROG in the command window. In the program window that opens for MYPROG, we are going to enter the commands to create a workfile, run a regression, compute residuals and a forecast, make a plot of the forecast, and save the results.

Figure 1.4: Program Window



1.4.1 Program Variables

Control variables are variables that you can use in place of numerical values in your EViews programs. Once a control variable is assigned a value, you can use it anywhere in a program that you would normally use a number. The name of a control variable starts with an "!" mark. After the "!", the name should be a legal EViews name of 15 characters or fewer. Examples of control variable names are: !q !1 !time

You do not need to declare control variables before your refer to them, though you must assign them a value before use. Control variables are assigned in the usual way, with the control variable name on the left of an "=" sign and a numerical value or expression on the right. For example:

```
!x = 7
!time = 12
```

Once assigned a value, a control variable may appear in an expression. For example:

```
!time = !time + 1
```

```
series returns = log(price) - log(price(-!q))
smpl 1950q1+!i 1960q4+!i
```

Control variables are automatically deleted after a program finishes. As a result, control variables are not saved when you save the workfile. You can save the values of control variables by creating new EViews objects which contain the values of the control variable. For example, the following command:

```
scalar numberx=!q
```

saves the numeric value assigned to the control variables !q into a scalar object numberx.

A **string variable** is a variable whose value is a string of text. A string expression or string is text enclosed in double quotes:

```
"cumulative returns"
"3.14159"
"ar(1) ar(2) ma(1) ma(2)"
```

String variables, which only exist during the time that your program is executing, have names that begin with a "%" symbol. The following lines assign values to string variables:

%mtvar = "cumulative returns"

%armas = "ar(1) ar(2) ma(1) ma(2)"

%pi = " 3.14159"

You may use strings variables to build up command text, variable names, or other string values. EViews provides a number of operators and functions for manipulating strings. Once assigned a value, a string variable may appear in any expression in place of the underlying string. Here is a quick example where we use string operations to concatenate the contents of three string variables.

%str1 = "USD/GBP "

%str2 = "cumulative returns"

%st3 = %st1 + %st2

In this example $\%ST3$ is set to the value "USD/GBP cumulative returns". String variables can be assigned to the table object for the output:

table1(1,1) = %st3

which is equivalent to entering the command

table(1,1) = "USD/GBP cumulative returns"

You can use a string variable to refer to a command, or a name, or portion of names indirectly. Suppose, for example, that we assign the string variable

%x = "usdgbp"

If you enclose a string variable in curly braces ("" and "") EViews will replace the expression with the name or name fragment given by the string value. In this context we refer to the expression "%x" as a replacement variable since the string variable %x is replaced in the command line by the name or names of objects to which the string refers. For example, the program line

plot %x

would be interpreted by EViews as

plot usdgbp

Changing the contents of %x to "usdjpy" changes the interpretation of the original line to

```
plot usdjpy
```

since the replacement variable uses the name obtained from the new %x.

Program arguments are special string variables that are passed to your program when you run the program. Arguments allow you to change the value of string variables every time you run the program. You may use them in any context where a string variable is appropriate. Program arguments will be named %0, %1, %2, and so on. When you run a program that takes arguments, you will also supply the values for the arguments. If you use the Run button or **File/Run**, you will see a dialog box where you can type in the values of the arguments. If you use the run command, you should list the arguments consecutively after the name of the program. For example, suppose we have a program named RETS containing a command

```
series returns=log(%0)-log(%0(-1))
```

To run RETS from the command line with

$\%0 = "USDGBP"$, enter

```
run rets usdgbp
```

This program creates a time series returns using the usdgbp exchange rate defined or loaded previously in your workfile.

Alternatively, you can run this program by clicking on the *Run* button on the program window, or selecting *File/Run*. In the Run Program dialog box that appears, type the name of the program in the Program name or path field and enter the values of the arguments in the Program arguments field. Any arguments in your program that are not initialized in the run command or Run Program dialog are treated as blanks.

IF Statements

There are many situations where you want to execute commands only if some condition is satisfied. EViews uses IF and ENDIF, or IF, ELSE, and ENDIF statements to indicate the condition to be met and the commands to be executed. An IF statement starts with the if keyword, followed by an expression for the condition, and then the word then. You may use AND/OR statements in the condition, using parentheses to group parts of the statement as necessary. If the expression is TRUE, all of the commands until the matching endif are executed. If the expression is FALSE, all of these commands are skipped. For example:

```
if !q = 3 then series returns = dlog(USDGBP) endif

if !time > 100 and !time < 200 then !age = 1/!time else !age = 0 endif
```

The FOR Loop

The for loop allows you to repeat a set of commands for different values of a control or string variable. The FOR loop begins with a for statement and ends with a next statement. Any number of commands may appear between these two statements. The syntax of the FOR statement differs depending upon whether it uses control variables or string variables.

FOR Loops with Control Variables To repeat statements for different values of a control variable, the for statement involves setting a control variable equal to an initial value, followed by the word to, and then an ending value. After the ending value you may include the word step followed by a number indicating by how much to change the control variable each time the loop is executed. If you do not include step, the step is assumed to be 1. For example,

```
for !j=1 to 10
vector(10) weights(!j)=returns(!j)/stddev(!j)
next
```

The for loop is executed first for the initial value, unless that value is already beyond the terminal value. After it is executed for the initial value, the control variable is incremented by step and EViews compares the variable to the limit. If the limit is passed, execution stops.

One important use of FOR loops with control variables is to change the sample. If you add a control variable to a date in a smpl command, you will get a new date as many observations forward as the current value of the control variable. Here is a FOR loop that gradually increases the size of the sample and computes an average returns:

```
for !i=1 to 60

smpl 1973m1 1974m1+!i

scalar avret!i = @mean(returns) next
```

One other important case where you will use loops with control variables is in accessing elements of a series or matrix objects. For example,

```
!rows=@rows(vec1)

vector cumsum1=vec1

for !i=2 to !rows cumsum1(!i)=cumsum1(!i-1)+vec1(!i) next
```

computes the cumulative sum of the elements in the vector vec1 and saves it in the vector cusum1. To access an individual element of a series, you will need to use the @elem function and @otod to get the desired element

```
for !i=2 to !rows

cumsum1(!i) = @elem(ser1, @otod(!i)) next
```

The @otod function returns the date associated with the observation index (counting from the beginning of the workfile), and the @elem function extracts the series element associated with a given date.

You can nest for loops to contain loops within loops. The entire inner for loop is executed for each successive value of the outer for loop. For example:

```
matrix(25,10) xx

for !i=1 to 25

for !j=1 to 10

xx(!i,!j)=(!i-1)*10+!j

next

next
```

FOR Loops with String Variables When you wish to repeat statements for different values of a string variable, you can use the FOR loop to let a string variable range over a list of string values. Give the name of the string variable followed by the list of values. For example,

```
for %y usdgbp usdjpy

series %yrets = dlog(%y) next
```

creates the returns series of two exchange rates

```
series usdgbpret = dlog(usdgbp)

series usdjpyret = dlog(usdjpy)
```

You can put multiple string variables in the same for statement – EViews will process the strings in sets.

For example:

```
for %y %z usdgbp usdjpy nzdusd audusd

equation e%y.ls %y c %z

next
```

In this case, the elements of the list are taken in groups of three. The loop is executed two times for the different sample pairs:

```
equation eusdgbp.ls usdgbp c usdjpy

equation eusdgbp.ls nzdusd c audusd
```

The WHILE Loop In some cases, we wish to repeat a series of commands several times, but only while one or more conditions are satisfied. Like the FOR loop, the WHILE loop allows you to repeat commands, but the WHILE loop provides greater flexibility in specifying the required conditions. The WHILE loop begins with a while statement and ends with a wend statement. Any number of commands may appear between the two statements. WHILE loops can be nested. The WHILE statement consists of the while keyword followed by an expression involving a control variable. The expression should have a logical (true or false) value or a numerical value. In the latter case, zero is considered false and any non-zero value is considered true. If the expression is true, the subsequent statements, up to the matching wend, will be executed, and then the procedure is repeated. If the condition is false, EViews will skip the following commands and continue on with the rest of the program following the wend statement. For example:

```
!val = 1

!a = 1
```

```
while !val<10000 and !a<10
smpl 1950q1 1970q1+!a
series inc!val = income/!val
!val = !val*10 !a = !a+1 wend
```

Unlike a FOR statement, the WHILE statement does not update the control variable used in the test condition. You need to explicitly include a statement inside the loop that changes the control variable, or your loop will never terminate. Use the F1 key to break out of a program which is in an infinite loop.

Subroutines A subroutine is a collection of commands that allows you to perform a given task repeatedly, with minor variations, without actually duplicating the commands. You can also use subroutines from one program to perform the same task in other programs. A subroutine starts with the keyword subroutine followed by the name of the routine and any arguments, and ends with the keyword `endsub`. Any number of commands can appear in between. The simplest type of subroutine has the following form:

```
subroutine rets

series returns = dlog(price)

endsub
```

where the keyword subroutine is followed only by the name of the routine. This subroutine has no arguments so that it will behave identically every time it is used. It creates the log-return time series from the existing price levels price.

You can use the return command to force EViews to exit from the subroutine at any time. A common use of return is to exit from the subroutine if an unanticipated error is detected.

To define a subroutine with arguments, you start with subroutine, followed by the subroutine name, a left parenthesis, the arguments separated by commas, and finally a right parenthesis. Each argument is specified by listing a type of EViews object, followed by the name of the argument. Control variables may be passed by the scalar type and string variables by the string type. For example:

```
subroutine rets1(series r, series p, scalar lg)

series r = dlog(p, lg)

endsub
```

This subroutine generalizes the example subroutine RETS. Calling RETS1 will fill the series given by the argument R with the log-returns of frequency LG from the series P. So if you set R equal to RETURNS, P equal to PRICES, and LG equal to 1, you will get the equivalent of the subroutine RETS above.

Subroutine call Your subroutine definitions should be placed, in any order, at the beginning of your program. The subroutines are executed by the program using a call statement. For example:

```
subroutine rets

series returns = dlog(price)

endsub

' program execution

load mywork

fetch z

call rets
```

Execution of this program begins with the load statement. The subroutine definition is executed only at the last line when it is "called". Subroutines may call each other, or even call themselves. Alternatively, you may wish to place frequently used subroutines in a separate program file and use an include statement to insert them at the beginning of your program. If, for example, you put the subroutine lines in the file RETURNS.PRG, then you may put the line:

```
include returns
```

at the top of any other program that needs to call RETS or RETS1. You can use the subroutines in these programs as though they were built-in parts of the EViews programming language.

If a subroutine has got arguments, it is executed by using the call keyword call which follows by the name of the subroutine and a list of any argument values you wish to use, enclosed in parentheses and separated by commas. All arguments must be provided in the same order as in the declaration statement. For example:

```
include rets1

load mywork

fetch z price

series returns

call rets1(returns, price, 3)
```

Subroutines work with variables and objects that are either global or local. Global variables refer either to objects which exist in the workfile when the subroutine is called, or to the objects that are created in the workfile by a subroutine. Global variables remain in the workfile when the subroutine finishes. A local variable is one that has meaning only within the subroutine. Local variables are deleted from the workfile once a subroutine finishes.

Global objects may be used and updated directly from within the subroutine. If, however, a global object has the same name as an argument in a subroutine, the variable name will refer to the argument and not to the global variable.

Local Subroutines All objects created by a global subroutine will be global and will remain in the workfile upon exit from the subroutine. If you include the word local in the definition of the subroutine, you create a local subroutine. All objects created by a local subroutine will be local and will be removed from the workfile upon exit from the subroutine. Local subroutines are most useful when you wish to write a subroutine which creates many temporary objects that you do not want to keep. You may not use or update global objects directly from within the subroutine. The global objects corresponding to arguments may be used and updated by referring to the arguments. All other objects in the subroutine are local and will be deleted when the subroutine finishes. If you want to save results from a local subroutine, you have to explicitly include them in the arguments.

Local subroutines can call global subroutines and vice versa. The global subroutine will only have access to the global variables, and the local subroutine will only have access to the local variables, unless information is passed between the routines via arguments.

Chapter 2

Regression Model

2.1 Introduction

This chapter starts with the introduction to a linear regression analysis, estimation and inference methods. Regression analysis is widely used tool in financial econometrics. They are used to describe and evaluate the relationship between financial variables, perform forecasting tasks.

This chapter provides only a short and brief description of main tools used in the regression analysis. More detailed discussion and deeper theoretical background can be found in Greene (2000), Hamilton (1994), Hayashi (2000), Verbeek (2008), Mills (1999), Zivot and Wang (2006).

2.2 Linear Regression Model

Consider the linear regression model

$$Y_i = \beta_1 + \beta_2 X_{2i} + ... + \beta_k X_{ki} + u_i = \mathbf{X}_i'\beta + u_i, \quad i = 1, ..., n, \tag{2.2.1}$$

where $\mathbf{X}_i = [1, X_{2i}, ..., X_{ki}]'$ is a $k \times 1$ vector of explanatory variables, $\beta = (\beta_1, ..., \beta_k)'$ is a $k \times 1$ vector of coefficients, and u_i is a random error term. In matrix form the model is expressed as

$$\mathbf{Y} = \mathbf{X}\beta + \mathbf{u}, \tag{2.2.2}$$

where $\mathbf{Y}$ and β are $n \times 1$ vectors and $\mathbf{X}$ is a $n \times k$ matrix.

The standard assumptions of the linear regression model are:

1. the linear model (2.2.2) is correctly specified;
2. the regressors $\mathbf{X}_i$ are uncorrelated with the error term $\mathbf{u}$: $E[\mathbf{X}_i u_i] = 0$ for all $i = 1, ..., n$;

3. $E\left[\mathbf{X}_i\mathbf{X}_i'\right] = \sigma_{XX}$ is of full rank k;
4. u_i are independently identically distributed (iid) with mean zero and constant variance σ^2.

Ordinary Least Squares (OLS) estimation is based on minimizing the residual sum of squares RSS. The fitted model is

$$Y_i = \mathbf{X}_i'\hat{\beta} + \hat{u}_i, \quad i = 1, ..., n,$$

where

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$$

and $\hat{u}_i = \mathbf{Y}_i - \hat{\mathbf{Y}}_i = Y_i - \mathbf{X}_i'\hat{\beta}$. An unbiased estimator of the regression variance is $\hat{\sigma}^2 = \frac{\hat{\mathbf{u}}'\hat{\mathbf{u}}}{n-k}$.

Under the assumptions described above, the OLS estimates $\hat{\beta}$ are consistent and asymptotically normally distributed. A consistent estimator of the asymptotic variance of the parameters estimator is

$$var\left[\hat{\beta}\right] = \hat{\sigma}^2\left(\mathbf{X}'\mathbf{X}\right). \tag{2.2.3}$$

Estimated standard errors $se\left[\hat{\beta}_i\right]$ for individual parameter estimators $\hat{\beta}_i$ are given by the square root of the diagonal elements of (2.2.3).

Goodness of fit is summarized by the R^2 of the regression $R^2 = 1 - \frac{RSS}{TSS}$, where $TSS = \sum_{i=1}^{n}\left(Y_i - \bar{Y}\right)^2$. The coefficient R^2 measures the percentage of the variation of the dependent variable $\mathbf{Y}$ that is explained by the variation of the regressors $\mathbf{X}$. The usual R^2 has the undesirable feature of never decreasing as more variables are added to the regression, even if the extra variables are irrelevant. A common way to solve the problem is to adjust R^2 for degrees of freedom; this gives $\bar{R}^2 = 1 - (1 - R^2)\frac{n-1}{n-k}$. The adjusted $\bar{R}^2$ may decrease with the addition of variables with low explanatory power.

2.2.1 Hypothesis testing

Suppose that we need to test the null hypothesis

$$H_0\colon \beta_j = \beta_j^0.$$

The OLS test statistic for testing this hypothesis (also called t-statistic) is

$$t = \frac{\hat{\beta}_j - \beta_j^0}{se\left[\hat{\beta}_j\right]},$$

which is asymptotically distributed $N(0,1)$ under the null hypothesis. With the additional assumption of iid Gaussian error term, $\hat{\beta}_j$ is normally distributed and the t-statistic follows Student's t distribution with $n-k$ degrees of freedom.

More general linear restrictions hypotheses of the form $H_0\colon R\beta = r$, where R is a fixed $m \times k$ matrix of rank m and r is a $m \times 1$ vector, are tested using the Wald statistic

$$F = \frac{\left(R\hat{\beta} - r\right)' \left(R \left(\mathbf{X}'\mathbf{X}\right)^{-1} R'\right) \left(R\hat{\beta} - r\right)}{var\left[\hat{\sigma}^2\right]}.$$

Under the null, the Wald statistic is asymptotically distributed χ^2_m. Under the Gaussian assumptions of residuals, $F/m \sim F_{m,n-k}$.

The statistical significance of all of the regressors excluding the intercept is captured by the F-statistic

$$F = \frac{ESS/(k-1)}{RSS/n-k} \frac{R^2/k}{(1-R^2)/(n-k)},$$

which is distributed $F_{k-1,n-k}$ under the null hypothesis that all slope coefficients are zero.

2.2.2 Residual diagnostics

If the classical assumptions of the linear regression model do not hold it may lead to inconsistent, inefficient estimates. There are several residual diagnostic statistics are usually reported along with the regression results to check the validity of the model predictions.

Two common problems with the regression assumptions are heteroscedasticity and autocorrelation of the error terms. Heteroscedasticity means that variances of error terms are not constant from observation to observation. Autocorrelation means presence of series correlation between error terms. In both cases, the OLS estimator is unbiased and consistent but not efficient anymore. Moreover, the standard formula for computing the variance of the parameters estimators (2.2.3) is not valid any more which may lead to wrong conclusions. If the variance-covariance matrix of error terms $var\,[\mathbf{u}] = \sigma^2\Omega$ then

$$var\left[\hat{\beta}\right] = \hat{\sigma}^2\left(\mathbf{X}'\mathbf{X}\right)\mathbf{X}'\Omega\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right).$$

One way of obtaining an efficient estimate of the regression parameters is to use Generalized Least Squares (GLS) method. The GLS estimator is given by $\hat{\beta}_{GLS} = (\mathbf{X}'\Omega^{-1}\mathbf{X})^{-1}\mathbf{X}'\Omega^{-1}\mathbf{Y}$ with variance $var\left[\hat{\beta}\right] = \hat{\sigma}^2\left(\mathbf{X}'\Omega^{-1}\mathbf{X}\right)$.

If the matrix Ω is not known, one can use White's heteroscedasticity consistent estimator of standard errors of the OLS estimators. The matrix

$$v\hat{a}r\left[\hat{\beta}\right] = \left(\sum_{i=1}^{n}\mathbf{X}_i\mathbf{X}_i'\right)^{-1}\left(\sum_{i=1}^{n}\hat{u}_i^2\mathbf{X}_i\mathbf{X}_i'\right)\left(\sum_{i=1}^{n}\mathbf{X}_i\mathbf{X}_i'\right)^{-1}$$

can be used as an estimate of the true variance of the OLS estimator.

There are several testing procedures to detect heteroscedasticity. The White test suggest to estimate an auxiliary regression of the squared OLS residuals on a constant and all regressors, their squares and cross products. Under the null hypothesis of homoscedasticity nR^2 statistic is asymptotically distributed $\chi^2(q)$, where q is the number of variables in the auxiliary regression minus one. If the value of statistic is large, the null hypothesis of homoscedasticity in residuals is rejected.

Another test for heteroscedasticity is Breusch-Godfrey-Pagan test. It suggests to regress squared residuals from the initial regression scaled by $\tilde{\sigma}^2 = \sum\hat{u}_i^2/n$ on a set of known variables $\mathbf{Z}_t$ (they also could be regressors but not restricted to). Under the null of homoscedasticity, the scaled $\frac{1}{2}ESS$ from the auxiliary regression follows asymptotically $\chi^2(p-1)$ distribution, where p is a number of auxiliary variables $\mathbf{Z}_t$.

The most common diagnostic statistics for presence of autocorrelation based on the estimated residuals $\hat{\mathbf{u}}_i$ is the Durbin-Watson statistic DW. It is defined as

$$DW = \frac{\sum_{i=2}^{n} \left(\hat{u}_i - \hat{u}_{i-1}\right)^2}{\sum_{i=1}^{n} \hat{u}_i^2} \tag{2.2.4}$$

For large n the Durbin-Watson statistics can be approximated $DW = 2\left(1 - \hat{\rho}\right)$, where $\hat{\rho}$ is the estimated correlation between and $\hat{u}_i$ and $\hat{u}_{i-1}$. Thus, the range of values of DW is from 0 to 4. Values of DW around 2 indicate no serial correlation in the error terms, values less than 2 suggest positive serial correlation, and values greater than 2 suggest negative serial correlation.

Exact critical values for a general case cannot be tabulated; however, Durbin and Watson (1950) established upper and lower bounds (d_U and d_L respectively) for the critical values. The testing procedure is as follows:

- if $DW < d_L$ we reject the null hypothesis of no autocorrelation in favour of positive first-order autocorrelation;
- if $DW > d_U$ we do not reject the null hypothesis

The bounds for critical values in the case of negative autocorrelation alternative are $4 - d_U$ and $4 - d_L$. The values of the bounds can be found in Savin and White (1977); some of the are tabulated in Table 2.1.

Table 2.1: Lower and Upper bounds for 5% critical values of the Durbin-Watson test

n	Number of regressors							
	$k=3$		$k=5$		$k=7$		$k=9$	
	d_L	d_U	d_L	d_U	d_L	d_U	d_L	d_U
25	1.206	1.550	1.038	1.767	0.868	2.012	0.702	2.280
50	1.462	1.628	1.378	1.721	1.291	1.822	1.201	1.930
75	1.571	1.680	1.515	1.739	1.458	1.801	1.399	1.867
100	1.634	1.715	1.592	1.758	1.550	1.803	1.506	1.850
200	1.748	1.789	1.728	1.810	1.707	1.831	1.686	1.852

The Breusch-Godfrey test for autocorrelation considers the regression of the OLS residuals $\hat{u}_i$ upon its lad $\hat{u}_{i-1}$. This auxiliary regression produces an estimate for the first-order autocorrelation coefficient $\hat{\rho}$ and provides a standard error to this estimate. In general case the test is easily extended to higher orders of autocorrelation by including additional lags of the residual. Testing the null hypothesis of no autocorrelation is equivalent to testing the significance of the auxiliary regression.

Another common diagnostic for serial correlation is the Ljung-Box modified Q statistic. The Q-statistic at lag q is a test statistic for the null hypothesis of no autocorrelation up to order q and is computed as:

$$Q = n(n+2)\sum_{j=1}^{q} \frac{\hat{\rho}_j^2}{n-j} \sim \chi_q^2,$$

where $\hat{\rho}_j$ is the j-th autocorrelation.

The most often used diagnostic statistic to test for normality of the residuals is the Jarque-Bera test statistics. It measures the difference of the skewness and kurtosis of the series with those from the normal distribution. The statistic is computed as:

$$JB = \frac{n}{6}\left(S^2 + \frac{(K-3)^2}{4}\right) \sim \chi_2^2,$$

where S is the skewness, and K is the kurtosis. We reject the null hypothesis of normality if a Jarque-Bera statistic exceeds the corresponding critical value.

2.2.3 Example: Factor Model

Fama and French (1993) suggested a three-factor model to explain the expected stock return premium required by investors. The three factors are

- The excess return of the market portfolio ($R_{mt}\breve{}r_{ft}$);
- The difference between the expected returns on portfolios of small and large firms (SMB_t); the small and large stock portfolios include all stocks with market capitalization in the lower and upper deciles of the sample median;
- The difference between the expected returns on portfolios of stocks with high and low book-to-market ratios (HML_t).

Thus, the expected excess return of stock i can be represented as

$$R_{it} - r_{ft} = \beta_0 + \beta_1(R_{it} - r_{ft}) + \beta_2 SMB_t + \beta_3 HML_t + u_t \tag{2.2.5}$$

As an example we consider monthly returns on IBM stocks for the period January 1990 to September 2007. This data with Fama-French factors is available as IBM1.xls. Variables in the data sets are

- ibm – monthly returns on IBM stocks;
- Mkt – monthly returns on the market index;
- rf – monthly rate of the risk-free rate;
- SMB and HML – Fama-French size and book-to-market risk factors, respectively.

In order to estimate the relation (2.2.5) we have to construct excess returns on IBM stocks and market portfolio. In EViews they can be created using

```
series ibm_ex=ibm-rf
series Mkt_ex=Mkt-rf
```

There are two ways of estimating linear regression in EViews. The first one, and more powerful, is through the main menu **Quick/Estimate Equation**. In the **Equation specification** window type the equation to be estimated. Using arithmetic operation we can specify the equation as

```
ibm_ex=C(1)+C(2)*Mkt_ex+C(3)*SMB+C(4)*HML
```

Note that coefficients of the equation should always be in the form C(1), C(2), etc. However, if the model is linear, it is more common to omit operation and coefficients signs and write ibm_ex C Mkt_ex SMB HML

Note, that in the latter specification the dependent variable should be on the first place. The term C indicates that we are estimating the model with intercept; if it is omitted, the regression will be estimated without the intercept term.

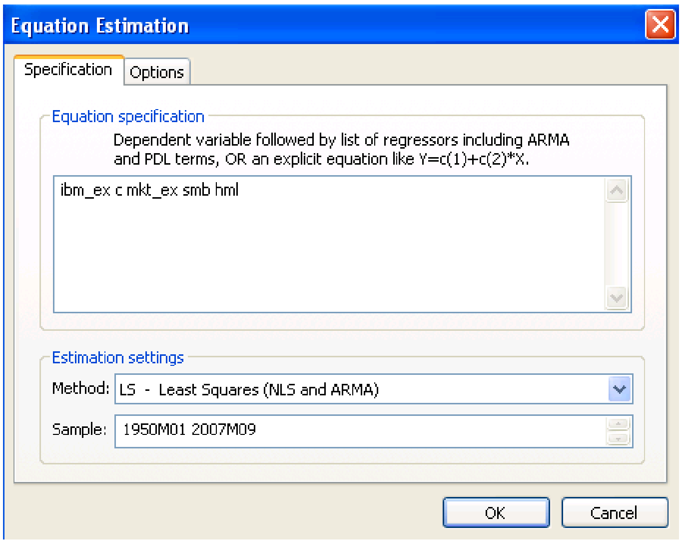


Figure 2.1: Regression estimation dialog window

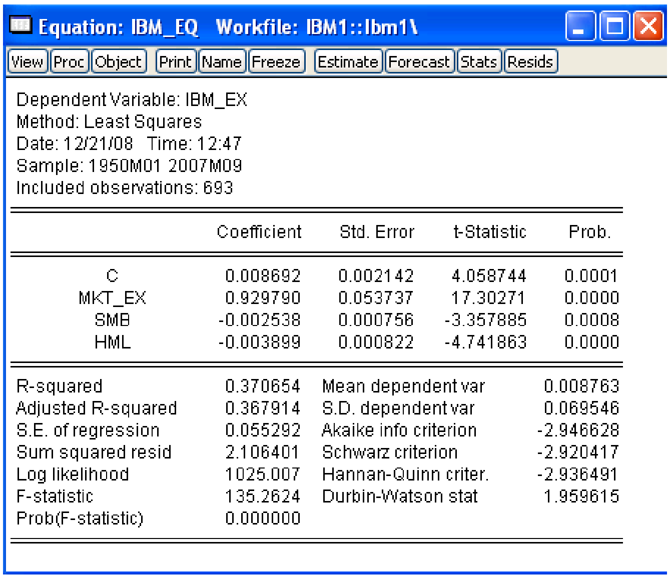
In the **Estimated setting window** make sure **LS – Least Squares (NLS and ARMA)** is chosen. The **Sample** window allows to estimate the model for different subsamples. This option subsample is specified in the same way as in the **Sample** object. Press **OK** and the regression output appears on the screen.

Another way of estimating a linear regression model is through the command line. To create an equation object use the declaration command **equation** following by a name of the object and the estimation type command (ls in our case stands for least squares) separated by the dot. Finally one should specify the model in the same way as above

equation ibmeq.ls ibm_ex C Mkt_ex SMB HML

Estimation Output The regression estimation output looks as follows

The estimated coefficients of the model are given in the column **Coefficients** (the coefficient in front of C denote estimate of the intercept term). Slope coefficients denote the sensitivities of the returns on the stock to the three factors and show the impact of systematic factors on returns. In column **t-statistic**, the value of the test statistic is provided to test that the hypothesis $\beta_i = 0$. All the coefficients are highly statistically significant as indicated by low p-values (column **Prob**). The



Equation: IBM_EQ Workfile: IBM1::Ibm1\

View | Proc | Object | Print | Name | Freeze | Estimate | Forecast | Stats | Resids

Dependent Variable: IBM_EX
Method: Least Squares
Date: 12/21/08 Time: 12:47
Sample: 1950M01 2007M09
Included observations: 693

	Coefficient	Std. Error	t-Statistic	Prob.
C	0.008692	0.002142	4.058744	0.0001
MKT_EX	0.929790	0.053737	17.30271	0.0000
SMB	-0.002538	0.000756	-3.357885	0.0008
HML	-0.003899	0.000822	-4.741863	0.0000

R-squared	0.370654	Mean dependent var	0.008763
Adjusted R-squared	0.367914	S.D. dependent var	0.069546
S.E. of regression	0.055292	Akaike info criterion	-2.946628
Sum squared resid	2.106401	Schwarz criterion	-2.920417
Log likelihood	1025.007	Hannan-Quinn criter.	-2.936491
F-statistic	135.2624	Durbin-Watson stat	1.959615
Prob(F-statistic)	0.000000		

Figure 2.2: Regression estimation output

overall significance of the regression is reflected in the value of **F-statistic** which is high enough to reject the null hypothesis of insignificance of all slope coefficients (p-value is given in **Prob (F-statistic)**.

The proportion of the variance R_{it} explained by the variability in the market index is the usual regression R^2 statistic and $1-R^2$ is the proportion of the variability of R_{it} that is due to firm specific factors. The proportion of market specific risk is $R^2 = 0.37$ and the proportion of firm specific risk is $1 - R^2 = 0.63$.

By estimating the regression model, EViews produces an object **Equation**, which can be saved and used later on (press **Name** button in the top of the equation window). As each object in EViews, **Equation** can be represented in different views. **View/Representation** view contains the equation specification of the model, **View/Estimation Output** provides the familiar model output. **View/Actual, Fitted, Residuals** creates various plots of the estimated residual series, as well as fitted values of the dependent variable. Residual series is automatically stored in the series object **resid** which created by EViews in each workfile. Note, that **resid** contains residuals of the last estimated model and will be lost once the model is reestimated. Thus, residual series has to be saved for further use, if necessary. This can done by copying the residual series into a new object

```
series resid_ibm=resid
```

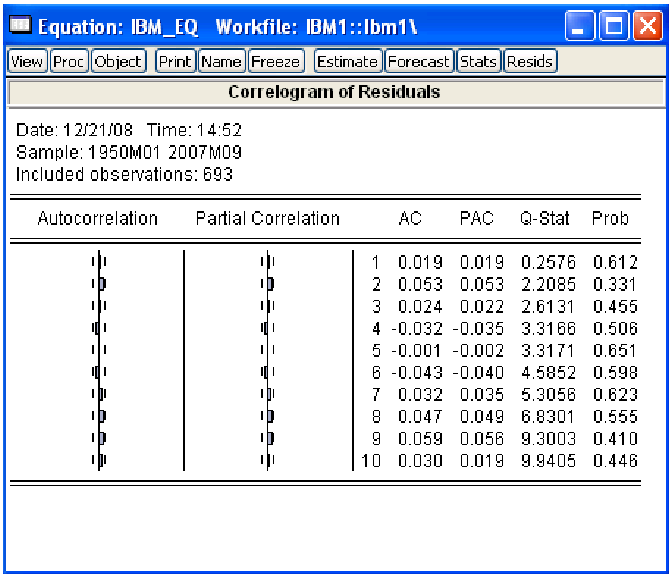
Now, the residuals from the CAPM regression for IBM stock returns are stored in the new series object **resid_ibm**.

Besides the standard errors of the coefficient estimators, given in the output window, one can retrieve the whole variance-covariance matrix by clicking on **View/Covariance Matrix**.

Residuals Diagnostic Before drawing any conclusions from the estimated regression, it is necessary to perform residual diagnostic to make sure that the assumptions of the classic linear regression model are satisfied. This can be done in the section **View/Residual Tests**. **Correlogram - Q-statistic** provides values of the Box-Ljung statistics to test the significance of of autocorrelations of residuals. The correlogram of the residuals from the factor model is given in Figure **??**.

Low p-values indicate absence of serial autocorrelations up to lag 10. Another way to test for series correlation is to perform Breusch-Godfrey Test – in EViews this can be done through **Serial Correlation LM Test**. In the upper panel of the Breusch-Godfrey test output there are two versions of the test statistic which are asymptotically equivalent. Their p-values both confirm the absence of series autocorrelation up to the second order. The no-autocorrelation null hypothesis is also not rejected by the Durbin-Watson test; test statistic is given in the regression output is equal to 1.959 which is in the acceptance region.

The option **Histogram - Normality Test** builds the histogram of the residuals, their descriptive statistics as well as the value of the Jarque-Bera statistic.



Equation: IBM_EQ Workfile: IBM1::Ibm1\

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Correlogram of Residuals

Date: 12/21/08 Time: 14:52
Sample: 1950M01 2007M09
Included observations: 693

Autocorrelation	Partial Correlation		AC	PAC	Q-Stat	Prob
		1	0.019	0.019	0.2576	0.612
		2	0.053	0.053	2.2085	0.331
		3	0.024	0.022	2.6131	0.455
		4	-0.032	-0.035	3.3166	0.506
		5	-0.001	-0.002	3.3171	0.651
		6	-0.043	-0.040	4.5852	0.598
		7	0.032	0.035	5.3056	0.623
		8	0.047	0.049	6.8301	0.555
		9	0.059	0.056	9.3003	0.410
		10	0.030	0.019	9.9405	0.446

Figure 2.3: Correlogram of residuals from the factor model for IBM stock returns

Notice that the Jarque-Bera statistic indicates that the residuals from the CAPM regression are not normally distributed. Note that even the residuals are not normally distributed, the inference is still correct asymptotically.

EViews also provides a number of test to test the hypothesis of homoscedasticity on the regression. Under the option **Heteroscedasticity Tests...** one can choose among Breush-Godfrey-Pagan, Harvey, Glejser, ARCH and White tests.

Three of them – Breush-Godfrey-Pagan, Glejser and White tests reject the hypothesis of homoscedasticity while Harvey and ARCH test do not reject the null. The reason for using several tests is that there are many different possible alternatives for the form of heteroscedasticity.

All the tests for autocorrelations and heteroscedasticity can be performed through the command line as well.

For the Breusch-Godfrey test for serial correlation we should specify the name of the regression equation we need to test and then the command auto(*lags*) where *lags* corresponds to the order of autocorrelation being tested. For example

```
ibm_eq.auto(2)
```

will perform the test for second order autocorrelation in the factor model for IBM stock.

To perform heteroscedasticity tests we should specify the equation name followed by the command hettest(*options*). In the *options* field we can specify the test being performed in the following way: type=*keyword*, where *keyword* is either "BPG"

(Breusch-Pagan-Godfrey - default), "Harvey", "Glejser", "ARCH", or "White". Inclusion of the command c in the options will lead to inclusion of cross-product terms in the auxiliary regression specification. Optionally, a list of variables may follow the command to include them into auxiliary regression as well.

For example,

```
ibm_eq.hettest(type="White",c)
```

will perform the WHite's test for heteroscedasticity for the ibm_eq equation.

Since the exact form of heteroscedasticity is not known, it is not clear how to perform GLS estimator is this case. EViews allows to compute heteroskedasticity consistent as well as heteroskedasticity and autocorrelation consistent coefficient covariance matrices. In order to compute them, click on **Estimate** button in the object menu and choose the **Options** tab. Tick the box in front of **Heteroscedasticity consistent coefficient matrix** to activate the option. Click **OK** to reestimate the model.

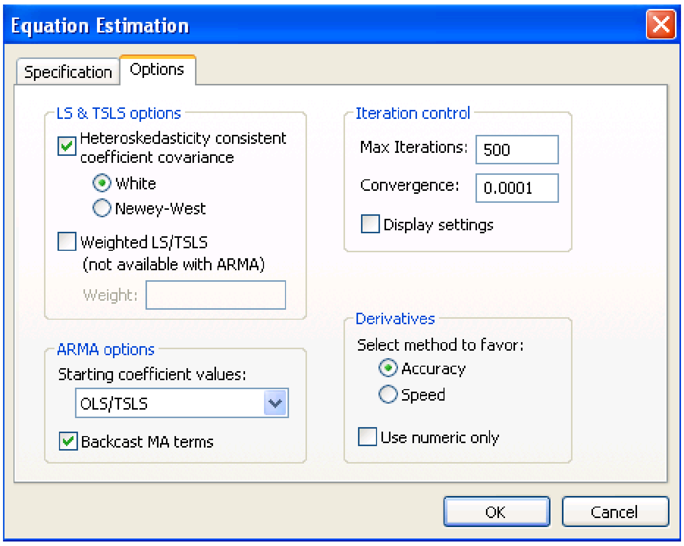


Figure 2.4: Regression output with White's heteroscedasticity adjusted standard errors

All coefficients remain still statistically significant using White's heteroscedasticity consistent standard errors.

Stability tests Finally, we can test the model for coefficients stability and structural breaks. In EViews this can be performed under the option **Views/Stability tests**. With Ramsey RESET test for model misspecification we cannot reject the null hypothesis of the correct specification (p-value 0.3754).

We start stability tests with the recursive residuals tests as they can help us to detect visually potential breakpoints. Click on **Recursive Estimates (OLS only)** and choose **Recursive residuals**. EViews will produce the plot of recursively estimated residuals from the model together with their confidence intervals.

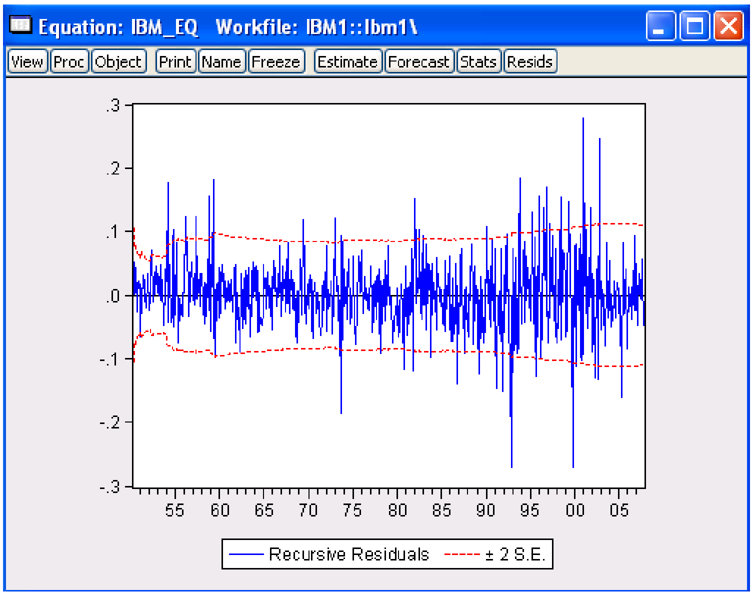


Figure 2.5: Recursively estimated residuals and their confidence bounds

Majority of the recursive residuals are within their confidence intervals however there are several outliers spraying out their bounds. These are potential points for the structural breaks in the models. The CUSUM test does not indicate any potential breakpoints, however the CUSUM squared test suggests that there may be some breaks in sixties and at the beginning of 2000.

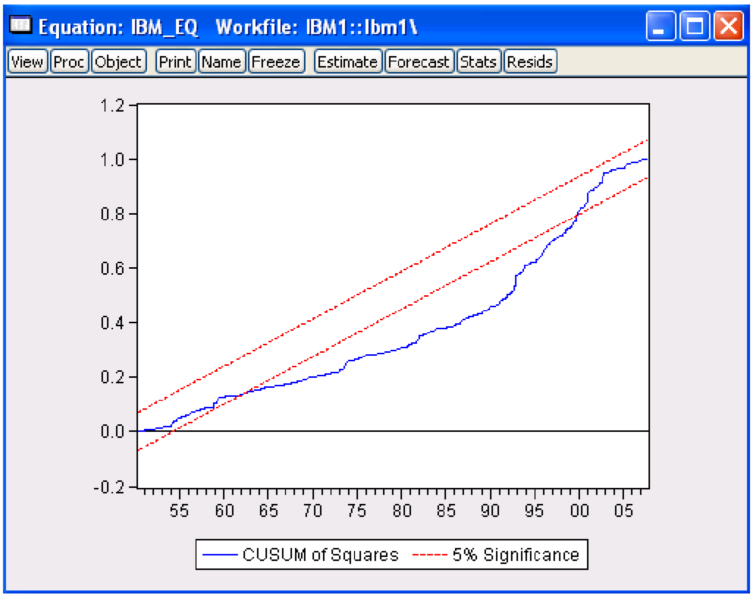


Figure 2.6: CUSUM squared statistics and its confidence bounds

We can go further and test whether there is a structural break at the specified dates using the Chow test. The p-value of the F-statistic for the Chow test is 0.3090 at the breakpoint January 1961 indicating no structural break at that date. However, if the breakpoint is specified at January 2000, we reject the null hypothesis of the parameters constancy at 1% significance level. Structural breaks may occur in the model due to some misspecifications. For example, from January 2000 there is one missing factor in the model which plays important role in explaining stock returns. The breakpoint data also corresponds to the dot.com bubble period where the classic factors model structure may change. In order to verify our hypothesis, we can include dummy variable corresponding to the bubble period to eliminate the effect from the model. To create the dummy which is equal to 1 for the period from January 2000 to December 2001, we write

```
series dummy=0
smpl 2000M01 2001M12
series dummy=1
```

smpl @all

Since structural breaks may occur in all the parameters, we include the dummy variable interacting with all regressors. Thus, in the **Estimate equation** box we have to specify a new model

ibm_ex C Mkt_ex SMB HML dummy*Mkt_ex dummy*SMB dummy*HML

As a result, the coefficient for interacting term with market portfolio returns and SMB factor is insignificant, however the interacting term with HML factor is statically significant at 10% level.

Equation: UNTITLED Workfile: IBM1::Ibm1\

View | Proc | Object | Print | Name | Freeze | Estimate | Forecast | Stats | Resids

Dependent Variable: IBM_EX
Method: Least Squares
Date: 12/23/08 Time: 19:42
Sample: 2000M01 2007M09
Included observations: 93

	Coefficient	Std. Error	t-Statistic	Prob.
C	0.013473	0.006911	1.949517	0.0545
MKT_EX	1.508672	0.231477	6.517579	0.0000
SMB	-0.007286	0.003156	-2.308490	0.0234
HML	-0.013625	0.003824	-3.562766	0.0006
DUMMY*MKT_EX	-0.434852	0.390015	-1.114963	0.2680
DUMMY*SMB	0.003600	0.004081	0.882275	0.3801
DUMMY*HML	0.009399	0.005163	1.820551	0.0722

R-squared	0.555885	Mean dependent var	0.002823
Adjusted R-squared	0.524900	S.D. dependent var	0.089562
S.E. of regression	0.061733	Akaike info criterion	-2.659709
Sum squared resid	0.327743	Schwarz criterion	-2.469083
Log likelihood	130.6765	Hannan-Quinn criter.	-2.582740
F-statistic	17.94059	Durbin-Watson stat	2.264644
Prob(F-statistic)	0.000000		

Figure 2.7: Output of the regression estimation with dummy variables

Correcting of misspecification also helps to improve properties of residuals. After introducing dummy variable all tests for heteroscedasticity indicate either no heteroscedasticity or produce some marginally significant p-values.

Testing linear restrictions EViews makes it possible to test hypothesis on the coefficient restrictions by means of Wald test. Consider testing the joint null hypothesis $\beta_1 = 1$ and $\beta_2 = \beta_3$. This hypothesis imposes two linear restrictions on the parameter vector. In the **View** option of the object menu choose **Coefficient Tests/Wald – Coefficient Restrictions...**. Type the restrictions to be tested in the box. Note that coefficients of the model are denoted by C(1), C(2), etc. In order to find out the exact notations of the parameters, go to **View/Representation**. P-value of the Wald test statistic is higher than any reasonable significance levels so we do not reject the null hypothesis of the validity of the restrictions.

The Wald test can also be performed in the command line. One should first specify the name of equation being tested followed by dot and command wald. Specifications of the restrictions follows separated by commas.

```
eq1.wald c(2)=0, c(3)=0
```

Predictions After having estimated the regression, often our aim is to construct forecast of the dependent variable. EViews' forecast function can be invoked through **Forecast** option in the menu of the equation object. In the box **Forecast name** type the name of the variable where the regression forecasts will be stored. EViews will automatically create a new series object with the specified name and plot the predicted series with two confidence region bounds.

Alternatively, one can view the forecast by double-click on the forecast variable. In the menu option **View** choose **Graph** where the required graph type can be generated.

In order to generate forecasts through the command line, use the command fit followed by a name of a series variable where the forecast values should be stored

```
ibmeq.fit ibm_exf
```

EViews also allows to generate standard errors of the forecasts along with the predictions themselves. To do this, simply include a name of another variable at the end of the line.

```
ibmeq.fit ibm_exf ibm_exfstd
```

2.2.4 Programming Example

Note that the factor sensitivities of stock (portfolio) returns represented by the estimated coefficients vary through time. As the model is estimated for alternative sample periods, the estimated coefficients will change. A useful analogy is the value of a stock's beta that varies through time based on the sample period data used to estimate the security market line.

The estimated factor model (2.2.5) for IBM uses all of the data over the 57 year period from January 1950 to September 2007. It is generally thought that coefficients do not stay constant over such a long time period. To take into account this fact while building returns forecast based on the factor model, we can perform rolling window regression. We start with initializing necessary variables (e.g., number of observation in the workfile, length of the window). For this purpose we make sure that the current sample is set to the whole range of the data. Type the following commands in a new program window:

```
smpl @all
scalar n=@obs(ibm_ex)
scalar window=60
```

Next, we create new object we will be using in the program – series of the forecasts and an equation object.

```
series ibm_exf
equation e
```

In the next lines we specify a loop where we reset the current sample to the estimation window and roll it across the data range. For each of the subsamples we estimate the factor model.

```
for !i=0 to n-window-1
smpl @first+!i @first+!i+window-1
e.ls ibm_ex c Mkt_ex SMB HML
```

Once the model is estimated we reset the sample to a subsample where we want to forecast returns. Since we build one-step-ahead forecast, our new subsample will be just one observation ahead the estimation window.

```
smpl @first+!i+window @first+!i+window
```

We generate the forecast using the estimated model. To access the values of the estimated parameters we use EViews function `@coefs`. In parentheses we specify the order of the parameter – it corresponds to the order of respective variable in the regression model. Note that `@coefs` contains the values of the last estimated model. Once the equation is re-estimated, the new values of parameters are stored in `@coefs`.

```
ibm_exf=@coefs(1)+@coefs(2)*Mkt_ex+@coefs(3)*smb+@coefs(4)*hml
next
```

Just to tidy up the workfile we delete auxiliary variables `window` and `n`.

```
delete window n
```

The series `ibm_exf` contains the generated forecast from the rolling window model.

Similarly to the use of `@coefs` function, one can access other OLS statistics. The specifications are given in Table 2.2.

Table 2.2: Equation Data Members

Data Member	Description
@aic	Akaike information criterion
@coefcov(i,j)	covariance of coefficient estimates β_i and β_j
@coefs(i)	i-th coefficient value
@dw	Durbin-Watson statistic
@f	F-statistic
@hq	Hannan-Quinn information criterion
@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
@tstats(i)	t-statistic value for coefficient
@coefcov	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

2.3 Nonlinear Regression

In many cases the relation between variables can happen to be nonlinear. If such model cannot be transformed into a linear one, we call such model intrinsically nonlinear regression model.

We can represent such model in the following way

$$bfY = F(\mathbf{X}, \theta) + \mathbf{u},$$

where F is a non-linear function, where $\mathbf{X}_i = [1, X_{2i}, ..., X_{ki}]'$ is a $k \times 1$ vector of explanatory variables, and u_i is a random error term.

The least squares estimation problem to minimize

$$S(\theta) = \sum_{i-1}^{n} (Y_i - F(X_i, \theta)^2 \qquad (2.3.1)$$

becomes non-linear. The first order conditions are given by

$$\frac{\partial S(\theta)}{\partial \theta_j} = -2\sum_{i-1}^{n} (Y_i - F(X_i, \theta)) \frac{\partial F}{\partial \theta_j}.$$

This gives a set of non-linear normal equations in θ. The non-linear least squares (NLS) estimator $\hat{\theta}_{NLS}$ is defined as the minimizing value of (2.3.1).

In EViews, the Nonlinear Least Squares method has the same implementation as the OLS. The only difference os that the model in the **Equation specification** box should be entered as a mathematical expression instead of a list of variables. for example,

```
y=@exp(c(1)*x)+(c(2)*z+4)^2
```

Interpretation of the estimation output, residual diagnostic and inference can be performed in the same way as for the OLS regression.

Chapter 3

Univariate Time Series: Linear Models

3.1 Introduction

Time series is a sequence of numerical data in which observations are measured at a particular instant of time. The frequency of observation can, for example, be annual, quarterly, monthly, daily, etc. The main goal of time series analysis is to study the dynamics of the data.

In this chapter we introduce basic time series models for estimation and forecasting of financial data. Further details about theory of time series analysis cab be found in Hamilton (1994), Greene (2000), Enders (2004), Tsay (2002) and others.

3.2 Stationarity and Autocorrelations

3.2.1 Stationarity

A time series $\{Y_t\}$ is said to be *strictly stationary*if for all integers i, j and all possible integers k the multivariate distribution function of $(Y_i, Y_{i+1}, ..., Y_{i+k-1})$ is identical to $(Y_j, Y_{j+1}, ..., Y_{j+k-1})$. In practice we are very often interested in consequences of this assumption regarding moments of the distribution. If Y_i and Y_j have identical distribution this implies that their means are identical, thus $E[Y_t]$ does not depend on time and equal to some constant μ. Also, because the pairs (Y_i, Y_{i+s}) and (Y_j, Y_{j+s}) have identical bivariate distributions it follows that the autocovariances

$$cov(Y_t, Y_{t+s}) = E\left[(Y_t - \mu)(Y_{t+s} - \mu)\right] = \lambda_s$$

depend only on the time lag s. This implies also that Y_t have constant variance $\lambda_0 = \sigma^2$.

A stochastic process whose first and second order moments (means, variances, and covariances) do not change with time is said to be second order stationary. More precisely, a time series Y_t is called stationary if the following conditions are satisfied:

$$E[Y_t] = \mu, \quad E[(Y_t - \mu)^2] = \gamma_0, \quad E[(Y_t - \mu)(Y_{t-s} - \mu)] = \gamma_s \quad \text{for all } t$$

Here μ, γ_0, and γ_k are finite-valued numbers that do not depend on time t.

3.2.2 Autocorrelation

The autocorrelations of a stationary process are defined by $\rho_s = \frac{\gamma_s}{\gamma_0}$. These correlations describe the short-run dynamic relations within the time series, in contrast with the trend, which corresponds to the long-run behaviour of the time series.

The simplest possible autocorrelations occur when a stationary process consists of uncorrelated random variables. In this case $\rho_0 = 1$, $\rho_s = 0$ for all $s > 0$. Such time series is called *white noise*.

It is important when modeling financial returns to appreciate that if $\{Y_t\}$ is white noise then Y_t and Y_{t+s} are not necessarily independent for $s > 0$.

The *partial autocorrelation* ϕ_s at lag s measures the correlation of Y_t values that are s periods apart after removing the correlation from the intervening lags. It equals the regression coefficient on Y_{t-s} when Y_t is regressed on a constant, Y_{t-1},..., Y_{t-s}.

Time series prediction To describe the correlations, we imagine that our observed time series comes from a stationary process that existed before we started observing it. We denote the past of the stationary process Y_t by $\mathcal{Y}_{t-1} = \{Y_{t-1}, Y_{t-1}, ...\}$, where the "dots" mean that there is no clear-cut beginning of this past. We call it also the information set available at time point $t-1$. The least squares predictor of Y_t based on the past $\mathcal{Y}_{t-1}$ is the function $f(\mathcal{Y}_{t-1})$ that minimizes $E\left[(Y_t - f(\mathcal{Y}_{t-1}))^2\right]$. This predictor is given by the conditional mean $f(\mathcal{Y}_{t-1}) = E[Y_t|\mathcal{Y}_{t-1}]$ with corresponding (one-step-ahead) prediction errors $e_t = Y_t - f(\mathcal{Y}_{t-1}) = Y_t - E[Y_t|\mathcal{Y}_{t-1}]$.

The process e_t is also called the innovation process, as it corresponds to the unpredictable movements in Y_t. If the observations are jointly normally distributed, then the conditional mean is a linear function of the past observations

$$E[Y_t|\mathcal{Y}_{t-1}] = a + p_1 Y_{t-1} + p_2 Y_{t-2} + ...$$

Here a models the mean $E[Y_t] = \mu$ of the series. From the above equation we get $\mu = a + \sum p_k \mu$, so that $\mu = (1 - \sum p_k)^{-1}$. As the process is assumed to be stationary, the coefficients p_k do not depend on time and the innovation process e_t is also stationary. It has the following properties:

- $E[e_t] = 0$ for all t
- $E\left[e_t^2\right] = \sigma^2$ for all t;
- $E[e_s e_t] = 0$ for all $s \neq t$.

Here the variance σ^2 is constant over time.

3.2.3 Example: Variance Ratio Test

Very often a predictability of stock returns is linked to the presence of autocorrelation in the returns series. If stock returns form an iid process, then variances of holding period returns should increase in proportion to the length of the holding period. If the log return is constant, then under the rational expectation hypothesis stock prices follows a random walk

$$p_{t+h} = \mu + p_{t+h-1} + u_{t+h} = \mu + \mu + p_{t+h-2} + u_{t+h} + u_{t+h-1} = p_t + \mu h + \sum_{i=0}^{h} u_{t+i}.$$

Variance of the returns forecasts

$$var\left[p_{t+h}-p_t\right]=\sum_{i=0}^{h}E\left[u_{t+i}^2\right]=h\sigma^2$$

due to the independence. Alternatively, if log returns are iid, then

$$var\left[r_{t,t+h}\right]=var\left[r_{t,t+1}+r_{t+1,t+2}+...+r_{t+h-1,t+h}\right]=hvar\left[r_{t+1}\right]$$

The variance-ratio statistic is defined as

$$VR_h=\frac{1}{h}\frac{var\left[r_{t,t+k}\right]}{var\left[r_{t+1}\right]}=1+\frac{2}{h}\sum_{j=1}^{h-1}(h-j)\rho_j,$$

which should be unity if returns are iid and less than unity under mean reversion.

The variance ratio test is set up as $H_0\colon VR_h=1$ and under the null

$$Z_h=\frac{VR_h-1}{\sqrt{2(2h-1)(h-1)/3hT}}\sim N(0,1).$$

See Cuthbertson and Nitzsche (2004) for more details about the test. Let us consider as an example how to program the variance ratio test in EViews.

In this test uses overlapping h-period returns. As an input to the program, the workfile should contain a series of log prices p used to test for predictability. We start the program in a usual way.

```
smpl @all
!h=2
```

The variable !h denotes the horizon of the returns forecast. The next we create one period and h period returns.

```
smpl @first+1 @last
series r=p-p(-1)
```

In order to build the variance ratio statistics we need to have the actual number of observations (returns), mean and variance of returns series.

```
scalar T=@obs(p)
scalar mu=@mean(r)
scalar var1=@sumsq(r-mu)/(T-1)
smpl @first+!h @last
series rh=p-p(-!h)
```

```
scalar varh=@sumsq(rq-!h*mu)/(T-!h+1)
```

We can now compute the variance ratio statistic

```
scalar VRh=varh/(!h*var1)

scalar Zh=(VRh-1)/@sqrt((2*(2*!q-1)*(!q-1))/(3*!q*T))
```

We need a p-value in order to test the hypothesis. Two-sided significance level (p-value) can be calculated as a follows

```
scalar Zh_level=2*(1-@cnorm(@abs(Zh)))
```

Finally, we create a table to report the results. We declare a new table VRTEST object with 2 rows and 5 columns, set the width of each column and write the context of each sell down.

```
table(2,5) VRTEST
Setcolwidth(VRTEST,1,15)
Setcolwidth(VRTEST,2,15)
Setcolwidth(VRTEST,3,10)
Setcolwidth(VRTEST,4,10)
Setcolwidth(VRTEST,5,13)
Setcell(VRTEST,1,1,"Nr of obs")
Setcell(VRTEST,1,2,"Horizon h")
Setcell(VRTEST,1,3,"VRh")
Setcell(VRTEST,1,4,"test stat Zh")
Setcell(VRTEST,1,5,"p-value")
Setcell(VRTEST,2,1,T,0)
Setcell(VRTEST,2,2,!h,0)
Setcell(VRTEST,2,3,VRh,4)
Setcell(VRTEST,2,4,Zh,4)
Setcell(VRTEST,2,5,Zh_level,5)
delete r mu rh T var1 varh Zh Zh_level
next
```

3.3 ARMA processes

A zero mean white noise process $\{\varepsilon_t\}$ can be used to construct new processes. We describe two commonly used examples first and afterwards their generalization – autoregressive-moving average (ARMA) model.

3.3.1 Autoregressive process

A simple way to model dependence between consecutive observations is

$$Y_t = \alpha_0 + \alpha_1 Y_{t-1} + \varepsilon_t,$$

where ε_t is white noise.Such process is called a *first-order autoregressive process* or $AR(1)$ process. It is stationary if the coefficient $|\alpha_1| < 1$.

Since $E[\varepsilon_t] = 0$ it follows that under the stationarity condition the mean of the process $E[Y_t] = \frac{\alpha_0}{1-\alpha_1}$ and variance $var[Y_y] = \frac{\sigma_\varepsilon^2}{1-\alpha_1^2}$ where $\sigma_\varepsilon^2 = var[\varepsilon_t]$. An $AR(1)$ process has autocorrelations $\rho_s = \alpha_1^s$ for $s > 1$.

A more general representation of the autoregressive process is

$$Y_t = \alpha_0 + \alpha_1 Y_{t-1} + ... + \alpha_p Y_{t-p} + \varepsilon_t$$

and called an autoregressive process of order p, or in short, $AR(p)$.

3.3.2 Moving average process

Consider the process $\{Y_t\}$ defined by

$$Y_t = \alpha_0 + \varepsilon_t + \beta_1 \varepsilon_{t-1}$$

so Y_t is a linear function of the present and immediately preceding innovations. This process is called a moving average process of order 1 and denoted by $MA(1)$.

A $MA(1)$ process is always stationary with mean α_0 and variance $(1+\beta_1^2)\,\sigma_\varepsilon^2$. Its autocorrelations are $\rho_1 = \frac{\beta_1}{1+\beta_1^2}$ and $\rho_s = 0$ for $s > 1$.

Comparing two time series we see that a shock ε_t in $MA(1)$ process affects Y_t in two periods (only two positive autocorrelation coefficients), while a shock in the $AR(1)$ process affects all future observations with a decreasing effect.

The $MA(1)$ process may be inverted to give ε_t as an infinite series in Y_t, Y_{t-1},..., namely

$$\varepsilon_t = Y_t + \beta_1 Y_{t-1} + \beta_1^2 Y_{t-2} + ...$$

that is

$$Y_t = -\beta_1 Y_{t-1} - \beta_1^2 Y_{t-2} - ... + \varepsilon_t.$$

Thus, $MA(1)$ time series can be represented as $AR(\infty)$ process. It is possible to invert $MA(1)$ process into a stationary AR process only if $|\beta_1| < 1$. This condition is known as *invertibility* condition.

A more general representation of a moving average process is

$$Y_t = \alpha_0 + \varepsilon_t + \beta_1 \varepsilon_{t-1} + ... + \beta_q \varepsilon_{t-q}$$

and called a moving average process of order q, or in short, $MA(q)$.

3.3.3 ARMA process

It is possible to combine the autoregressive and moving average specification into $ARMA(p, q)$ model

$$Y_t = \alpha_1 Y_{t-1} + ... + \alpha_p Y_{t-p} + \varepsilon_t + \beta_1 \varepsilon_{t-1} + ... + \beta_q \varepsilon_{t-q}. \tag{3.3.1}$$

An $ARMA(p, q)$ time series can be represented in a shorter form using the notion of lag operator.

The lag operator L, is defined as $LY_t = Y_{t-1}$, the operator which gives the previous value of the series. This operator can also be used to represent the lags of the second or higher orders in the following way:

$$L^2(Y_t) = L(L(Y_t)) = L(Y_{t-1}) = Y_{t-2}.$$

In general $ARMA(p,q)$ process is

$$A(L)Y_t = B(L)\varepsilon_t,$$

where

$$A(L) = 1 - \alpha_1 L - \alpha_2 L^2 - ... - \alpha_p L^p$$

$$B(L) = 1 - \beta_1 L - \beta_2 L^2 - ... - \beta_q L^q$$

Stationarity requires the roots of $A(L)$ to lie outside the unit circle, and invertibility places the same condition on the roots of $B(L)$.

Table 3.1: Correlation patterns

Time series	**acf**	**pacf**
$AR(p)$	Infinite: decays towards zero	Finite: disappears after lag p
$MA(q)$	Finite: disappears after lag q	Infinite: decays towards zero
$ARMA(p,q)$	Infinite: damps out	Infinite: decays towards zero

3.3.4 Estimation of ARMA processes

$ARMA(p,q)$ models are generally estimated using the technique of maximum likelihood.

An often ignored aspect of the maximum likelihood estimation of $ARMA(p,q)$ models is the treatment of initial values. These initial values are the first p values of Y_t and q values of ε_t in (3.3.1). The exact likelihood utilizes the stationary distribution of the initial values in the construction of the likelihood. The conditional likelihood treats the p initial values of Y_t as fixed and often sets the q initial values of ε_t to zero. The exact maximum likelihood estimates (MLE) maximize the exact log-likelihood, and the conditional MLE maximize the conditional log-likelihood. The exact and conditional MLEs are asymptotically equivalent but can differ substantially in small samples, especially for models that are close to being non-stationary or non-invertible.

For pure AR models, the conditional MLEs are equivalent to the least squares estimates

Model Selection Criteria Before an $ARMA(p,q)$ may be estimated for a time series Y_t, the AR and MA orders p and q must be determined by visually inspecting the autocorrelation and partial autocorrelation functions for Y_t. If the autocorrelation function decays smoothly and the partial autocorrelations are 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 decay slowly towards zero, a first-order moving average process would seem appropriate.

Alternatively, statistical model selection criteria may be used. The idea is to fit all $ARMA(p,q)$ models with orders p and q and choose the values of p and q which minimizes model selection criteria:

$$AIC(p,q) = \ln\left(\tilde{\sigma}^2(p,q)\right) + \frac{2}{T}(p+q)$$

$$BIC(p,q) = \ln\left(\tilde{\sigma}^2(p,q)\right) + \frac{\ln(T)}{T}(p+q)$$

where $\tilde{\sigma}^2(p,q)$ is the MLE of $var[\varepsilon_t] = \sigma^2$ without a degrees of freedom correction from the $ARMA(p,q)$ model.

3.3.5 Example: ARMA in EViews

We start our example from the simulation of ARMA process and then we take a look at its estimation. In order to illustrate the statements in Table 3.1, let us simulate $AR(3)$, $MA(2)$ and $ARMA(3,2)$ processes and compute their autocorrelation and partial autocorrelation functions.

In particular, we simulate

$$
\begin{aligned}
Y_t &= 0.8Y_{t-1} + 0.15Y_{t-2} - 0.1Y_{t-3} + u_t \\
Y_t &= u_t - 0.95u_{t-1} + 0.3u_{t-2} \\
Y_t &= 0.8Y_{t-1} + 0.15Y_{t-2} - 0.1Y_{t-3} + u_t - 0.95u_{t-1} + 0.3u_{t-2}
\end{aligned}
\tag{3.3.2}
$$

To start with, we generate a series of uncorrelated normally distributed residuals (remember, command nrnd generates standard normally distributed random number)

```
series u=0.5*nrnd
```

Also, we have to generate initial values for the series. Since the highest order of the series is 3, let us generate first three values. This can be done by setting sample to only fist three observations and assign zero values to all of three series.

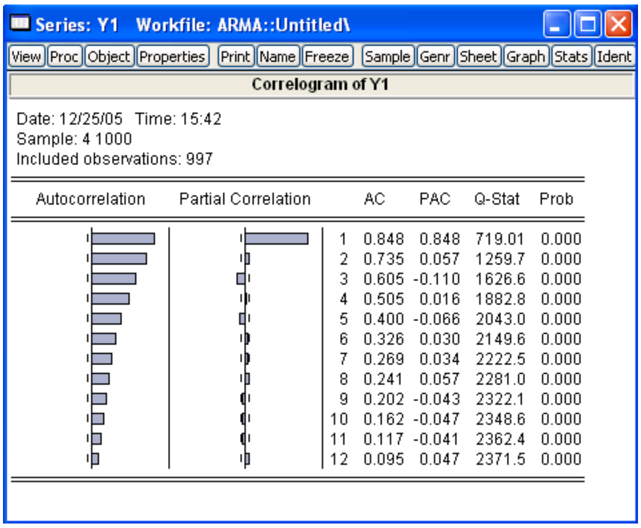
```
smpl @first @first+2
series y1=0
series y2=0
series y3=0
```

Now, we set the sample for the rest of observations and generate series according to formulae (3.3.2)

```
smpl @first+3 @last
y1=0.8*y1(-1)+0.15*y1(-2)-0.1*y1(-3)+u
y2=u-0.95*u(-1)+0.3*u(-2)
y3=0.8*y1(-1)+0.15*y1(-2)-0.1*y1(-3)+u-0.95*u(-1)+0.3*u(-2)
```

Now, we are ready to build and inspect their correlograms. Remind, that in order to build a correlogram, one should click on the icon if the time series being investigated and choose **View/Correlogram...** option. The correlograms of three time series is given on Figures **??**-**??**.

As we have expected, the autocorrelation function for the first series ($AR(3)$) damps out slowly towards zero while its partial autocorrelation function has spikes at first three lags. The autocorrelation function of the second series ($MA(2)$) has spikes at two first lags and disappears afterwards (becomes insignificant) while the partial autocorrelation function decays oscillating towards zero. Both autocorrelation and partial autocorrelation functions of the third series ($ARMA(3, 2)$) decay slowly towards zero without any clear spikes.



Series: Y1 Workfile: ARMA::Untitled\

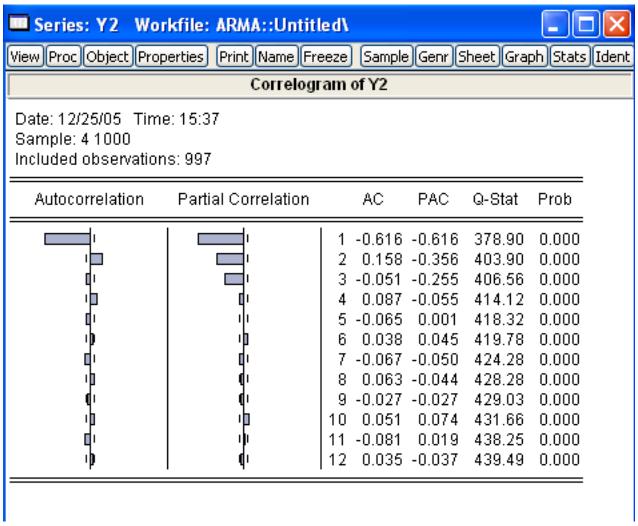
View Proc Object Properties Print Name Freeze Sample Genr Sheet Graph Stats Ident

Correlogram of Y1

Date: 12/25/05 Time: 15:42
Sample: 4 1000
Included observations: 997

Lag	AC	PAC	Q-Stat	Prob
1	0.848	0.848	719.01	0.000
2	0.735	0.057	1259.7	0.000
3	0.605	-0.110	1626.6	0.000
4	0.505	0.016	1882.8	0.000
5	0.400	-0.066	2043.0	0.000
6	0.326	0.030	2149.6	0.000
7	0.269	0.034	2222.5	0.000
8	0.241	0.057	2281.0	0.000
9	0.202	-0.043	2322.1	0.000
10	0.162	-0.047	2348.6	0.000
11	0.117	-0.041	2362.4	0.000
12	0.095	0.047	2371.5	0.000

Figure 3.1: Correlogram of an $AR(3)$ process



Series: Y2 Workfile: ARMA::Untitled\

View Proc Object Properties Print Name Freeze Sample Genr Sheet Graph Stats Ident

Correlogram of Y2

Date: 12/25/05 Time: 15:37
Sample: 4 1000
Included observations: 997

Lag	AC	PAC	Q-Stat	Prob
1	-0.616	-0.616	378.90	0.000
2	0.158	-0.356	403.90	0.000
3	-0.051	-0.255	406.56	0.000
4	0.087	-0.055	414.12	0.000
5	-0.065	0.001	418.32	0.000
6	0.038	0.045	419.78	0.000
7	-0.067	-0.050	424.28	0.000
8	0.063	-0.044	428.28	0.000
9	-0.027	-0.027	429.03	0.000
10	0.051	0.074	431.66	0.000
11	-0.081	0.019	438.25	0.000
12	0.035	-0.037	439.49	0.000

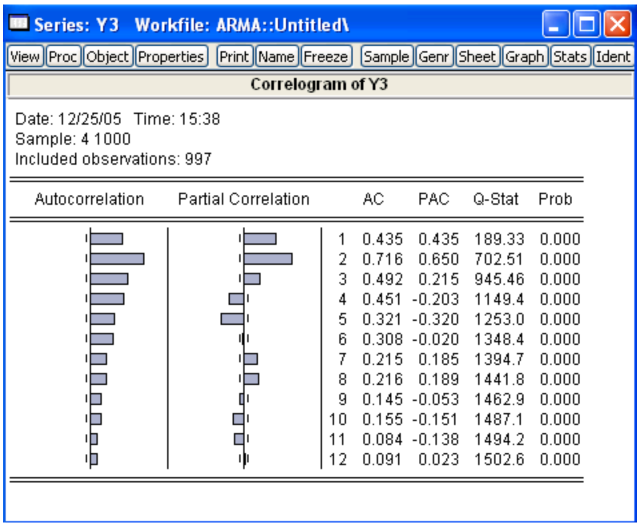
Figure 3.2: Correlogram of a $MA(2)$ process

Estimation An estimation of the ARMA processes is performed in EViews in the same way as OLS estimation of a linear regression. The only difference is in specifying autoregressive and moving average terms in the model. If the series has got autoregressive components, we should include terms ar(1), ar(2), etc, as regressors up to the required order. For example, to estimate the first series, type

y1 c ar(1) ar(2) ar(3)

in the estimation equation box. EViews produces an output given in Figure **??**

All coefficients are significant as expected and are very close to the true values.



Series: Y3 Workfile: ARMA::Untitled\

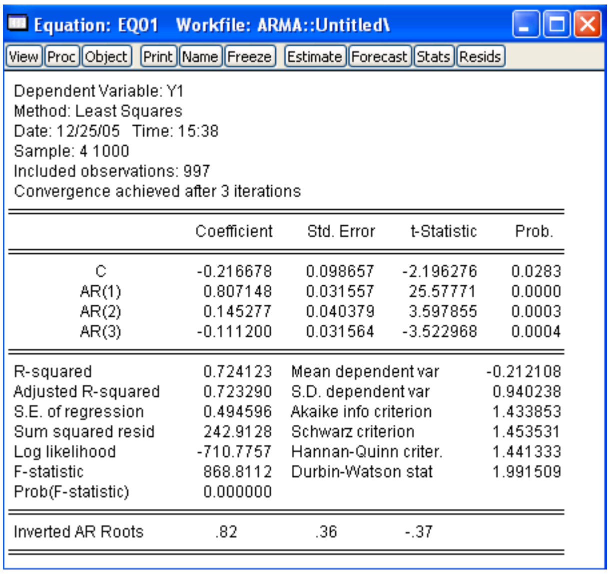
View Proc Object Properties Print Name Freeze Sample Genr Sheet Graph Stats Ident

Correlogram of Y3

Date: 12/25/05 Time: 15:38
Sample: 4 1000
Included observations: 997

	AC	PAC	Q-Stat	Prob
1	0.435	0.435	189.33	0.000
2	0.716	0.650	702.51	0.000
3	0.492	0.215	945.46	0.000
4	0.451	-0.203	1149.4	0.000
5	0.321	-0.320	1253.0	0.000
6	0.308	-0.020	1348.4	0.000
7	0.215	0.185	1394.7	0.000
8	0.216	0.189	1441.8	0.000
9	0.145	-0.053	1462.9	0.000
10	0.155	-0.151	1487.1	0.000
11	0.084	-0.138	1494.2	0.000
12	0.091	0.023	1502.6	0.000

Figure 3.3: Correlogram of an $ARMA(3,2)$ process



Equation: EQ01 Workfile: ARMA::Untitled\

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Dependent Variable: Y1
Method: Least Squares
Date: 12/25/05 Time: 15:38
Sample: 4 1000
Included observations: 997
Convergence achieved after 3 iterations

	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.216678	0.098657	-2.196276	0.0283
AR(1)	0.807148	0.031557	25.57771	0.0000
AR(2)	0.145277	0.040379	3.597855	0.0003
AR(3)	-0.111200	0.031564	-3.522968	0.0004

R-squared	0.724123	Mean dependent var	-0.212108
Adjusted R-squared	0.723290	S.D. dependent var	0.940238
S.E. of regression	0.494596	Akaike info criterion	1.433853
Sum squared resid	242.9128	Schwarz criterion	1.453531
Log likelihood	-710.7757	Hannan-Quinn criter.	1.441333
F-statistic	868.8112	Durbin-Watson stat	1.991509
Prob(F-statistic)	0.000000		

Inverted AR Roots	.82	.36	-.37

Figure 3.4: Estimation output of $ARMA$ process

Inference and tests can be performed in the same way as it was done for the OLS regression.

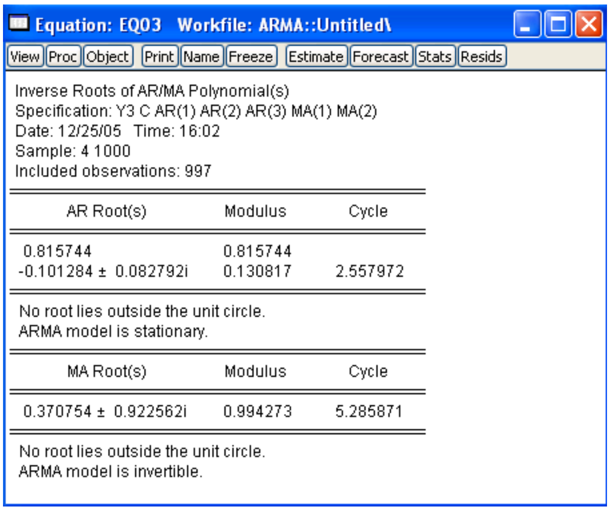
If one needs to estimate the model containing moving average components, ma(1), mar(2), etc terms should be included into the model specification. For example, to estimate the second time series, we write

y2 c ma(1) ma(2)

Autoregressive and moving average terms can be combined to estimate ARMA model. Thus, specification of the third series looks like

```
y3 c ar(1) ar(2) ar(3) ma(1) ma(2)
```

After having estimated an ARMA model, one can check whether the estimated coefficients satisfy the stationarity assumptions. This can be done through **View/ARMA structure** of the **Equation** object. For the third series we obtain



Equation: EQ03 Workfile: ARMA::Untitled\

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Inverse Roots of AR/MA Polynomial(s)
Specification: Y3 C AR(1) AR(2) AR(3) MA(1) MA(2)
Date: 12/25/05 Time: 16:02
Sample: 4 1000
Included observations: 997

AR Root(s)	Modulus	Cycle
0.815744	0.815744	
-0.101284 ± 0.082792i	0.130817	2.557972

No root lies outside the unit circle.
ARMA model is stationary.

MA Root(s)	Modulus	Cycle
0.370754 ± 0.922562i	0.994273	5.285871

No root lies outside the unit circle.
ARMA model is invertible.

Figure 3.5: Table of the roots of the estimated *ARMA* process

It says that our ARMA series is both stationary and invertible.

3.3.6 Programming example

If we had not known the order of the ARMA series, we would need to apply one of the information criteria to select the most appropriate order of the series. The following program illustrates how this can be done using the Akaike criterion.

First we need to define the maximal orders for autoregressive and moving average parts and store them into variables pmax and qmax. Also we need to declare a matrix object aic where the values of the Akaike statistic will be written for each specification of the ARMA process.

```
smpl @all

scalar pmax=3

scalar qmax=3

matrix(pmax+1,qmax+1) aic
```

Next, we define nested loops which will run through all possible ARMA specification with orders within the maximal values.

```
for !p=0 to pmax
for !q=0 to qmax
```

As the number of lags included in the model increases we add a new AR term in the model. For this purpose we create a new string variable textsf%order containing the model specification.

```
if !p=0 then %order=""
else
for !i=1 to !p
%order=%order+" ar("+@str(!i)+")"
next
endif
```

We perform the same procedure with the MA term specification.

```
if !q=0 then %order=%order+""
else
for !i=1 to !q
%order=%order+" ma("+@str(!i)+")"
next
endif
```

Once the model specification is determined and written in the variable %order we can use a substitution to estimate the corresponding model.

```
equation e.ls y3 c %order
%order=""
```

The last command nullify the variable %order for the use in the next step of the loops. Now we can write the value of the Akaike criterion for the current in the table.

```
aic(!p+1,!q+1)=e.@aic
next
next
delete e
```

After the program run, the values of the Akaike criterion are stored in the table aic. Now we can choose that specification of the ARMA model which produces the smallest AIC value.

Chapter 4

Stationarity and Unit Roots Tests

4.1 Introduction

Many financial time series, like exchange rate levels of stock prices appear to be non-stationary. New statistical issues arises when analyzing non-stationary data. Unit root tests are used to detect the presence and form of non-stationarity.

This chapter reviews main concepts of non-stationarity of time series and provides a description of some tests for time series stationarity. More information about such tests can be found in Hamilton (1994), Fuller (1996), Enders (2004), Harris (1995), Verbeek (2008).

There are two principal methods of detecting nonstationarity:

- Visual inspection of the time series graph and its correlogram;
- Formal statistical tests of unit roots.

We will start with formal testing procedures first.

A nonstationaty time series is called *integrated* if it can be transformed by first differencing once or a very few times into a stationary process. The *order of integration* is the minimum number of of times the series needs to be first differenced to yield a stationary series. An integrated of order 1 time series is denoted by $I(1)$. A stationary time series is said to be integrated of order zero, $I(0)$.

4.2 Unit Roots tests

Let us consider a time series Y_t in the form

$$\begin{aligned} Y_t &= \alpha + \beta Y_{t-1} + u_t \\ u_t &= \rho u_{t-1} + \varepsilon_t \end{aligned} \tag{4.2.1}$$

Unit root tests are based on testing the null hypothesis that $H_0\colon \rho = 1$ against the alternative $H_1\colon \rho < 1$. They are called unit root tests because under the null hypothesis the characteristic polynomial has a root equal to unity. On the other hand, stationarity tests take the null hypothesis that Y_t is trend stationary.

4.2.1 Dickey-Fuller test

One commonly used test for unit roots is the Dickey-Fuller test. In its simplest form it considers a $AR(1)$ process

$$Y_t = \rho Y_{t-1} + u_t$$

where u_t is an IID sequence of random variables. We want to test

$$H_0\colon \rho = 1 \text{ vs. } H_1\colon \rho < 1.$$

Under the null hypothesis Y_t is non-stationary (random walk without drift). Under the alternative hypothesis, Y_t is a stationary $AR(1)$ process.

Due to non-stationarity of Y_t under the null, the standard t-statistic does not follow t distribution, not even asymptotically. To test the null hypothesis, it is possible to use

$$DF = \frac{\hat{\rho} - 1}{s.e\,(\hat{\rho})}.$$

Critical values, however, have to be taken from the appropriate distribution, which is under the null hypothesis of non-stationarity is nonstandard. The asymptotic critical values of DF based on computer simulations are given in Fuller (1996).

The above test is based on the assumption that the error terms are iid and there is no drift (intercept term) in the model. The limiting distribution will be wrong if these assumptions are false.

More general form of the Dickey-Fuller test employs other variants of the time series process. Consider the following three models for the data generating process of Y_t:

$$Y_t = \rho Y_{t-1} + u_t \tag{4.2.2}$$

$$Y_t = \rho Y_{t-1} + \alpha + u_t \tag{4.2.3}$$

$$Y_t = \rho Y_{t-1} + \alpha + \beta t + u_t \tag{4.2.4}$$

with u_t being iid process.

Dickey and Fuller (1979) derive a limiting distribution for the least squares t-statistic for the null hypothesis that $\rho = 1$ and F-statistic (Wald statistic) for the null hypotheses of validity of combinations of linear restrictions $\rho = 0$, $\alpha = 0$ and $\beta = 0$ where the estimated models are from (4.2.2) to (4.2.4) but in each case that (4.2.2) is the true data generating process.

4.2.2 Augmented Dickey-Fuller test

Dickey and Fuller (1981) show that the limiting distributions and critical values that they obtain under the assumption of iid u_t process are also valid when u_t is autoregressive, when augmented Dickey-Fuller (ADF) regression is run. Assume the data are generated according to (4.2.2) with $\rho = 1$ and that

$$u_t = \theta_1 u_{t-1} + \theta_2 u_{t-2} + ... + \theta_p u_{t-p} + \varepsilon_t \tag{4.2.5}$$

where ε_t are iid. Consider the regression

$$\Delta Y_t = \phi Y_{t-1} + \alpha + \beta t + u_t$$

and test $H_0\colon \phi = 0$ versus $H_1\colon \phi < 0$. Given the equation for u_t in (4.2.5) we can write

$$\Delta Y_t = \phi Y_{t-1} + \alpha + \beta t + \theta_1 u_{t-1} + \theta_2 u_{t-2} + ... + \theta_p u_{t-p} + \varepsilon_t.$$

Since under $\rho = 1$ we have $u_t = Y_t - Y_{t-1}$, this equation can be rewritten as

$$\Delta Y_t = \phi Y_{t-1} + \alpha + \beta t + \theta_1 \Delta Y_{t-1} + \theta_2 \Delta Y_{t-2} + ... + \theta_p \Delta_{t-p} + \varepsilon_t. \tag{4.2.6}$$

Said and Dickey (1984) provide a generalization of this result for $ARMA(p, q)$ error terms.

Procedure Before using the ADF test we have to decide how many lags of ΔY to include in the regression. This can be done by sequentially adding lags and testing for serial correlation using Lagrange multiplier tests to archive a white noise residuals.

Use F-test to test the null $(\beta, \rho) = (0, 1)$ against the alternative $(\beta, \rho) \neq (0, 1)$. If the null is rejected we know that

$$\text{either} \begin{bmatrix} \beta \neq 0 \\ \rho = 1 \end{bmatrix} \text{ or } \begin{bmatrix} \beta = 0 \\ \rho \neq 1 \end{bmatrix} \text{ or } \begin{bmatrix} \beta \neq 0 \\ \rho \neq 1 \end{bmatrix}$$

and the next step is to test $\rho = 1$ using the t-statistic obtained from the estimating the augmented version of (4.2.4), with the critical values taken from the standard normal tables. Critical values from the standard normal are appropriate when β is non-zero, so that if the null hypothesis is not rejected we can rule out the second and third cases (if β is zero the critical values are non-standard, but will be smaller than the standard normal ones). Thus, if $\rho = 1$ is accepted we conclude that $\beta \neq 0$ and $\rho = 1$, so that series has a unit root and a linear trend.

If we reject the null then the first alternative can be dismissed. This leaves the following two alternatives

$$\text{either} \begin{bmatrix} \beta = 0 \\ \rho \neq 1 \end{bmatrix} \text{ or } \begin{bmatrix} \beta \neq 0 \\ \rho \neq 1 \end{bmatrix}$$

In either case ρ is not 1, there is no unit root and conventional test procedures can be used. Thus we may carry out a t test for the null that $\beta = 0$.

If we cannot reject $(\beta, \rho) = (0, 1)$ we know that the series has a unit root with no trend but with possible drift. To support the conclusion that $\rho = 1$ we may test this, given β is assumed to be zero.

If we wish to establish whether the series has non-zero drift, further tests will be required. Note that we know $(\beta, \rho) = (0, 1)$, and so we might carry out the F test. This tests

$$H_0\colon (\alpha, \beta, \rho) = (0, 0, 1) \text{ vs. } (\alpha, \beta, \rho) \neq (0, 0, 1).$$

If we cannot reject the null hypothesis, the series is random walk without drift. If we reject it, the series is a random walk with drift.

We may wish to support these findings on the basis of estimating (4.2.3) by setting β at zero as suggested by the various previous tests. If β is actually zero then tests on α and ρ should have greater power once this this restriction is imposed.

4.2.3 Phillips and Perron tests

The statistics proposed by Phillips and Perron (1988) (Z statistics) arise from their considerations of the limiting distributions of the various Dickey-Fuller statistics when the assumption that u_t is an iid process is relaxed.

The test regression in the Phillips-Perron test is

$$\Delta Y_t = \phi Y_{t-1} + \alpha + \beta t + u_t$$

where u_t is a stationary process (which also may be heteroscedastic). The PP tests correct for any serial correlation and heteroscedasticity in the errors ut of the test regression by directly modifying the test statistics. These modified statistics, denoted Z_t and Z_ϕ, are given by

$$Z_t = \left(\frac{\hat{\sigma}^2}{\hat{\lambda}^2}\right)^{\frac{1}{2}} t_{\phi=0} - \frac{1}{2}\left(\frac{\hat{\lambda}^2 - \hat{\sigma}^2}{\hat{\lambda}^2}\right)\left(\frac{T s.e\left(\hat{\rho}\right)}{\hat{\sigma}^2}\right)$$

$$Z_\phi = T\phi - \frac{1}{2}\left(\frac{T^2 s.e\left(\hat{\rho}\right)}{\hat{\sigma}^2}\right)\left(\hat{\lambda}^2 - \hat{\sigma}^2\right)$$

The terms $\hat{\sigma}^2$ and $\hat{\lambda}^2$ are consistent estimates of the variance parameters

$$\hat{\sigma}^2 = \lim_{T\to\infty} T^{-1} \sum_{t=1}^{T} E\left[u_t^2\right]$$

$$\hat{\lambda}^2 = \lim_{T\to\infty} \sum_{t=1}^{T} E\left[T^{-1} S_T^2\right]$$

where $S_T = \sum\limits_{t=1}^{T} u_t$.

In the Dickey-Fuller specification we can use the critical values given by Dickey and Fuller for the various statistics if u_t is an iid and we should use Phillips-Perron's counterparts if it is not iid.

An indication as to whether the Z statistic should be used in addition to (or instead of) the ADF tests might be obtained in the diagnostic statistics from the DF and ADF regressions. If normality, autocorrelation or heterogeneity statistics are significant, one might adopt the Phillips-Perron approach. Furthermore, power may be adversely affected by misspecifying the lag length in the augmented Dickey-Fuller regression, although it is unclear how far this problem is mitigated by choosing the number of lags using data-based criteria, and the Z-tests have the advantage that this choice does not have to be made. Against this, one should avoid the use of the Z test if the presence of negative moving average components is somehow suspected in the disturbances.

Under the null hypothesis that $\phi = 0$, the PP Z_t and Z_ϕ statistics have the same asymptotic distributions as the ADF t-statistic and normalized bias statistics. One advantage of the PP tests over the ADF tests is that the PP tests are robust to general forms of heteroskedasticity in the error term u_t. Another advantage is that the user does not have to specify a lag length for the test regression.

4.3 Stationarity tests

The ADF and PP unit root tests are for the null hypothesis that a time series Y_t is $I(1)$. Stationarity tests, on the other hand, are for the null that Y_t is $I(0)$. The most commonly used stationarity test, the KPSS test, is due to Kwiatkowski, Phillips, Schmidt and Shin (1992) (KPSS). They derive their test by starting with the model

$$Y_t = \alpha + \beta t + \mu_t + u_t$$

$$\mu_t = \mu_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim WN(0, \sigma_\varepsilon^2)$$

where u_t is $I(0)$ and may be heteroskedastic.

The null hypothesis that Y_t is $I(0)$ is formulated as $H_0\colon \sigma_\varepsilon^2 = 0$, which implies that μ_t is a constant. Although not directly apparent, this null hypothesis also implies a unit moving average root in the ARMA representation of ΔY_t.

The KPSS test statistic is the Lagrange multiplier (LM) or score statistic for testing $\sigma_\varepsilon^2 = 0$ against the alternative that $\sigma_\varepsilon^2 > 0$ and is given by

$$KPSS = \left(\frac{1}{T^2}\sum_{t=1}^{T}\hat{S}_t^2\right)/\hat{\lambda}^2$$

where $\hat{S}_t = \sum_{j=1}^{t}\hat{u}_j$, $\hat{u}_t$ is the residual of a regression Y_t on t and $\hat{\lambda}_t^2$.

Critical values from the asymptotic distributions must be obtained by simulation methods. The stationary test is a one-sided right-tailed test so that one rejects the null of stationarity at the α level if the KPSS test statistic is greater than the $100(1-\alpha)$ quantile from the appropriate asymptotic distribution.

4.4 Example: Purchasing Power Parity

It is very easy to perform unit root and stationarity tests in EViews. As an example, consider a Purchasing Power Parity condition between two countries: USA and UK. In efficient frictionless markets with internationally tradeable goods, the law of one price should hold. That is,

$$s_t = p_t - p_t^*,$$

where s_t is a natural logarithm of the spot exchange rate (price of a foreign currency in units of a domestic one), p_t is a logarithm of the aggregate price index in the domestic country and p_t^* is a log price in the foreign country. This condition is referred to as absolute purchasing power parity condition.

This condition is usually verified by testing for non-stationarity of the real exchange rate $q_t = s_t + p_t^* - p_t$. Before we perform this let us look at properties of the constituent series.

We consider monthly data for USA and UK over the period from January 1989 to November 2008.

Although plots of both consumer price indices and the exchange rate indicate non-stationarity, we perform formal tests for unit root and stationarity.

In the dataset PPP.wf1, there are levels of the exchange rate and consumer price indices are given, so we need to create log series to carry out tests. This is done as usually,

```
series lcpi_uk=log(cpi_uk)
series lcpi_uk=log(cpi_uk)
series lgbp_usd=log(gbp_usd)
```

Let us start with the UK consumer price index. We can find the option **Unit Root Tests...** in the **View** section of the series object menu (double click on lcpi_uk icon). In the **Test Type** box there is a number of tests available in EViews. We start with Augmented-Dickey-Fuller test. As we are interested in testing for unit roots in levels of log consumer price index, we choose **Test for unit root in levels** in the next combo-box, and finally we select testing with both intercept and trend as it is the most general case.

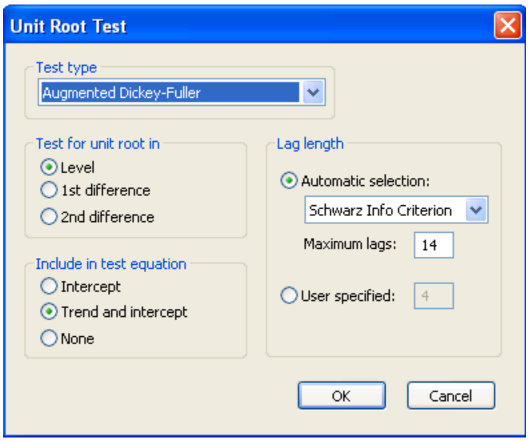
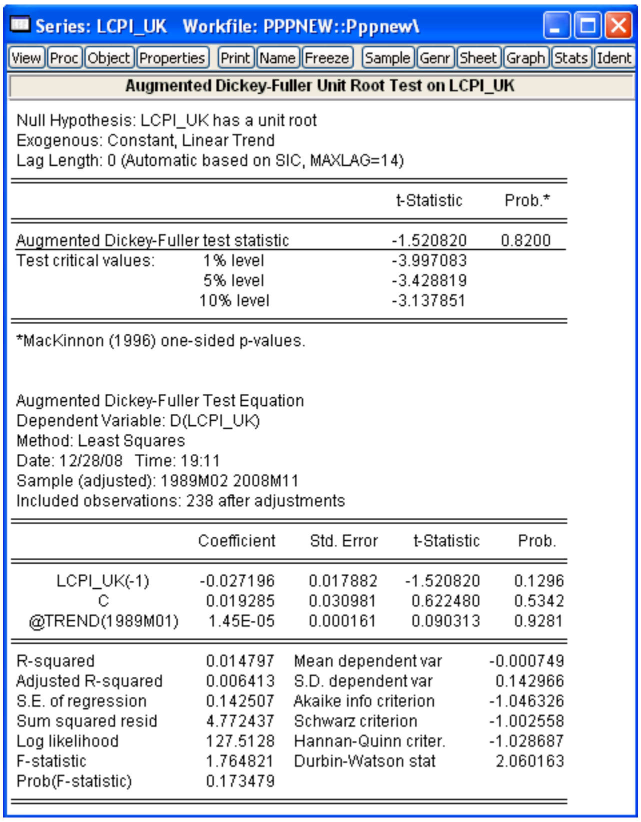


Figure 4.1: Augmented Dickey-Fuller test dialog window

EViews will also select the most appropriate number of lags of the residuals to be included in the regression using the selected criteria (it is possible to specify a number of lags manually is necessary by ticking **User specified** option).

Click **OK** and EViews produces the following output



Series: LCPI_UK Workfile: PPPNEW::Pppnew\

View | Proc | Object | Properties | Print | Name | Freeze | Sample | Genr | Sheet | Graph | Stats | Ident

Augmented Dickey-Fuller Unit Root Test on LCPI_UK

Null Hypothesis: LCPI_UK has a unit root
Exogenous: Constant, Linear Trend
Lag Length: 0 (Automatic based on SIC, MAXLAG=14)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.520820	0.8200
Test critical values:	1% level	-3.997083	
	5% level	-3.428819	
	10% level	-3.137851	

*MacKinnon (1996) one-sided p-values.

Augmented Dickey-Fuller Test Equation
Dependent Variable: D(LCPI_UK)
Method: Least Squares
Date: 12/28/08 Time: 19:11
Sample (adjusted): 1989M02 2008M11
Included observations: 238 after adjustments

	Coefficient	Std. Error	t-Statistic	Prob.
LCPI_UK(-1)	-0.027196	0.017882	-1.520820	0.1296
C	0.019285	0.030981	0.622480	0.5342
@TREND(1989M01)	1.45E-05	0.000161	0.090313	0.9281

R-squared	0.014797	Mean dependent var	-0.000749
Adjusted R-squared	0.006413	S.D. dependent var	0.142966
S.E. of regression	0.142507	Akaike info criterion	-1.046326
Sum squared resid	4.772437	Schwarz criterion	-1.002558
Log likelihood	127.5128	Hannan-Quinn criter.	-1.028687
F-statistic	1.764821	Durbin-Watson stat	2.060163
Prob(F-statistic)	0.173479		

Figure 4.2: Output for the Augmented Dickey-Fuller test

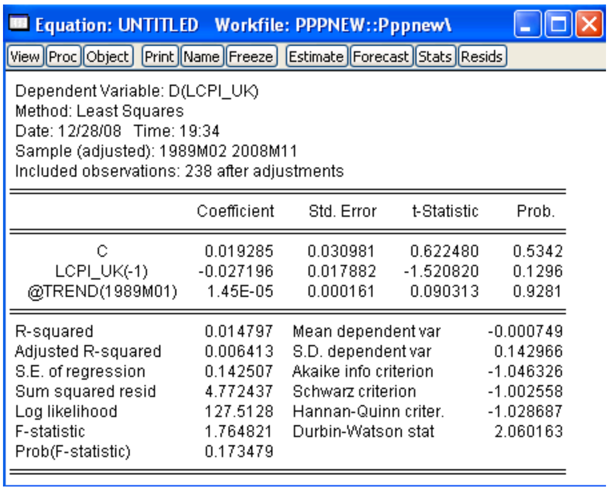
The absolute value of the t-statistic does not exceed any of the critical values given below so we cannot reject the null hypothesis of the presence of unit root in the series.

Unfortunately, EViews provides only the test of the null hypothesis $H_0\colon \phi = 0$. One can perform more general test by estimating Dickey-Fuller regression (4.2.6). In the command line type the following specification

```
ls d(lcpi_uk) c lcpi_uk(-1) @trend(1989M01)
```

to run the ADF regression with intercept and trend component. As you noted, the function `@trend` allows to include the time trend component that increases by one for each date in the workfile. The optional date argument `1989M01` is provided to indicate the starting date for the trend. We did not include any MA components in the regression since based on the previous results (see Figure **??**) zero lag is optimal according to the Schwartz selection criterium.

The regression output is identical with that produced by the Augmented Dickey-Fuller test


Equation: UNTITLED Workfile: PPPNEW::Ppppnew\
View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Dependent Variable: D(LCPI_UK)
Method: Least Squares
Date: 12/28/08 Time: 19:34
Sample (adjusted): 1989M02 2008M11
Included observations: 238 after adjustments

	Coefficient	Std. Error	t-Statistic	Prob.
C	0.019285	0.030981	0.622480	0.5342
LCPI_UK(-1)	-0.027196	0.017882	-1.520820	0.1296
@TREND(1989M01)	1.45E-05	0.000161	0.090313	0.9281

R-squared	0.014797	Mean dependent var	-0.000749
Adjusted R-squared	0.006413	S.D. dependent var	0.142966
S.E. of regression	0.142507	Akaike info criterion	-1.046326
Sum squared resid	4.772437	Schwarz criterion	-1.002558
Log likelihood	127.5128	Hannan-Quinn criter.	-1.028687
F-statistic	1.764821	Durbin-Watson stat	2.060163
Prob(F-statistic)	0.173479		

Figure 4.3: Output of the regression-based procedure of the Augmented Dickey-Fuller test

However, the approach enables us to perform the Wald test of linear restrictions and specify the null hypothesis $H_0\colon (\beta, \phi) = (0, 0)$ (or more general, $H_0\colon (\alpha, \beta, \phi) = (0, 0, 0)$).

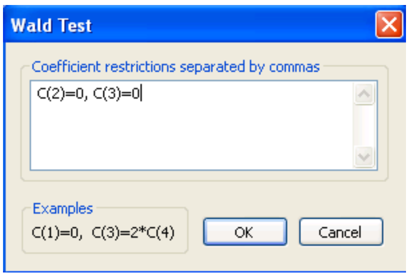


Figure 4.4: Wald test results for the Augmented Dickey-Fuller test specifications

The value of Wald test statistic in the case of the null $H_0\colon (\beta, \phi) = (0, 0)$ is 1.7648; this has to be compared with the critical values tabulated in MacKinnon (1996).

As the test statistic is smaller than all of the critical values, we cannot reject the null hypothesis, which confirms non-stationarity of the log consumer price index series.

This conclusion is also confirmed by other stationarity tests. For example, in Kwiatkowski-Phillips-Schmidt-Shin test (specification with the intercept and trend), the test statistic is 0.4257 which is higher than the critical value at 1% significance level (which is 0.216). Thus, we reject the null hypothesis of stationarity of the series.

If the Purchasing Power Parity condition holds one would expect the real exchange rate $q_t = s_t - p_t + p_t^*$ to be stationary and mean reverting. The presence of unit root in the deviations series would indicate the existence of permanent shocks which do not disappear in a long run.

We create a series of deviations

```
d=lgbp_usd-lcpi_uk-lcpi_us
```

Augmented Dickey-Fuller does not reject the null hypothesis of the presence of unit root in the deviations series. Also, the value of Wald test statistic is 1.8844 indicates which confirms nonstationarity of the deviations from the Purchasing Power Parity condition.

Chapter 5

Univariate Time Series: Volatility Models

5.1 Introduction

In Chapter 3 we have considered approaches to modelling conditional mean of a univariate time series. However, many areas of financial theory are concerned with the second moment of time series – conditional volatility as a proxy for risk.

In this chapter we introduce time series models that represent the dynamics of conditional variances. In particular we consider ARCH, GARCH model as well as their extensions.

The reader is also referred to Engle (1982), Bollerslev (1986), Nelson (1991), Hamilton (1994), Enders (2004), Zivot and Wang (2006).

5.2 The ARCH Model

Besides a time varying conditional mean of financial time series, most of them also exhibit changes in volatility regimes. This is especially applicable to many high frequency macroeconomic and financial time series.

While modelling such time series, we cannot use homoscedastic models. The simplest way to allow volatility to vary is to model conditional variance using a simple autoregressive (AR) process.

Let Y_t denote a stationary time series, then Y_t can be expressed as its mean plus a white noise:

$$Y_t = c + u_t \tag{5.2.1}$$

where c is the mean of Y_t, and u_t is i.i.d. with mean zero. To allow for conditional heteroscedasticity, assume that

$$var_{t-1}[u_t] = \sigma_t^2.$$

Here var_{t-1} denotes the variance conditional on information at time $t-1$, and is modelled in the following way:

$$\sigma_t^2 = \alpha_0 + \alpha_1 u_{t-1}^2 + ... + \alpha_p u_{t-p}^2. \tag{5.2.2}$$

In order to show that this specification is equivalent to AR representation of squared residuals, note that $var_{t-1}[u_t] = E\left[u_{t-1}^2\right] = \sigma_t^2$ since $E[u_t] = 0$. Thus, equation (5.2.2) can be rewritten as:

$$u_t^2 = \alpha_0 + \alpha_1 u_{t-1}^2 + ... + \alpha_p u_{t-p}^2 + \varepsilon_t, \tag{5.2.3}$$

where $\varepsilon_t = u_t^2 - E_{t-1}\left[u_t^2\right]$ is a zero mean white noise process. The model in (5.2.1) and (5.2.3) is known as the autoregressive conditional heteroscedasticity (ARCH) model of Engle (1982), which is usually referred to as the $ARCH(p)$ model. More generally, ARCH model can be rewritten as

$$\begin{aligned} Y_t &= c + u_t \\ u_t &= \sigma_t \eta_t \\ \sigma_t^2 &= \alpha_0 + \alpha_1 u_{t-1}^2 + ... + \alpha_p u_{t-p}^2, \end{aligned}$$

where η_t is an iid normal random variable.

5.2.1 Example: Simulating an $ARCH(p)$ model in EViews

It is relatively easy to simulate ARCH process in EViews. Let us consider as example the following $ARCH(2)$ model

$$\begin{aligned} Y_t &= \sigma_t \eta_t \\ \sigma_t^2 &= 3.5 + 0.5 Y_{t-1}^2 + 0.48 Y_{t-2}^2 \end{aligned} \tag{5.2.4}$$

with η_t being independent random variables following $N(0,1)$distribution. Similarly to ARMA process we need to generate error term process η_t and first two initial values of Y_t after which the whole process can be simulated. Create a new workfile and in the command line enter

```
smpl @all

series eta=nrnd

smpl @first @first+1
```

```
series y=@sqrt(3.5/(0.5+0.48))*eta
smpl @first+2 @last
y=@sqrt(3.5+0.5*y(-1)^2+0.48*y(-2)^2)*eta
smpl @all
```

The last statement is included to ensure that we come back the whole data range. The plot of the simulated series is given in the following figure.

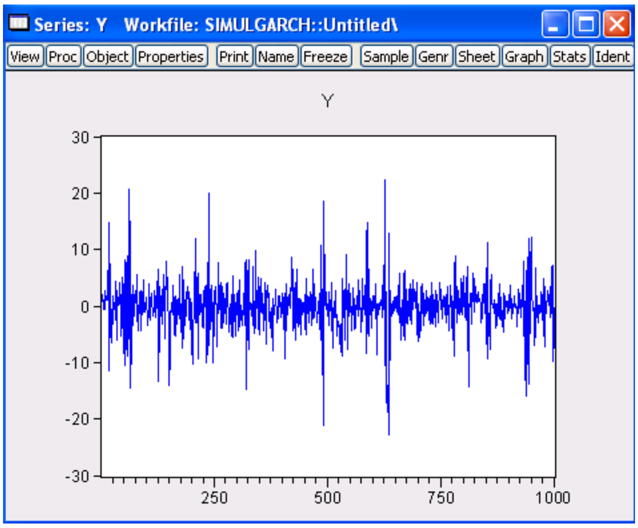


Figure 5.1: Plot of simulated ARCH process

Visually, the process looks stationary, mean reverting and with zero mean as expected from the equation (5.2.4).

Testing for ARCH Effects In order to test for the presence of ARCH effects in the residuals, we can use AR representation of squared residuals in the following way. Based on equation (5.2.2), construct an auxiliary regression

$$\hat{u}_t^2 = \alpha_0 + \alpha_1 \hat{u}_{t-1}^2 + ... + \alpha_p \hat{u}_{t-p}^2 + \varepsilon_t, \tag{5.2.5}$$

. The significance of parameters α_i would indicate the presence of conditional volatility. Under the null hypothesis that there are no ARCH effects:

$$\alpha_1 = \alpha_2 = ... = \alpha_p = 0,$$

the test statistic $LM = TR^2 \overset{a}{\sim} \chi_p^2$ where T is the sample size and R^2 is computed from the regression (5.2.5).

5.3 The GARCH Model

More general form of conditional volatility is based on ARMA specification as an extension of AR process of squared residuals. Bollerslev (1986) introduces GARCH model (which stands for generalized ARCH) where he replaces the AR model in (5.2.2) by:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i u_{t-i}^2 + \sum_{j=1}^{q} \beta_i \sigma_{t-j}^2, \tag{5.3.1}$$

where the coefficients α_i and β_j are positive to ensure that the conditional variance σ_t^2 is always positive. In order to emphasize the number of lags used in (5.3.1) we denote the model by $GARCH(p, q)$.

When $q = 0$, the GARCH model reduces to the ARCH model. Under the $GARCH(p, q)$ model, the conditional variance of u_t, σ_t^2, depends on the squared residuals in the previous p periods, and the conditional variance in the previous q periods. The most commonly used model is a $GARCH(1, 1)$ model with only three parameters in the conditional variance equation.

A GARCH model can be expressed as an ARMA model of squared residuals. For example, for a $GARCH(1, 1)$ model:

$$\sigma_t^2 = \alpha_0 + \alpha_1 u_{t-1}^2 + \beta_1 \sigma_{t-1}^2.$$

Since $E_{t-1}\left[u_t^2\right] = \sigma_t^2$, the above equation can be rewritten as:

$$u_t^2 = \alpha_0 + (\alpha_1 + \beta_1) u_{t-1}^2 + \varepsilon_t - \beta_1 \varepsilon_{t-1}, \tag{5.3.2}$$

which is an $ARMA(1,1)$ model. Here $\varepsilon_t = u_t^2 - E_{t-1}\left[u_t^2\right]$ is the white noise error term.

Given the ARMA representation of the GARCH model, we conclude that stationarity of the $GARCH(1,1)$ model requires $\alpha_1 + \beta_1 < 1$. The unconditional variance of u_t is given by

$$var[u_t] = E\left[u_t^2\right] = \alpha_0/(1 - \alpha_1 - \beta_1),$$

Indeed, from (5.3.2)

$$E\left[u_t^2\right] = \alpha_0 + (\alpha_1 + \beta_1)E\left[u_{t-1}^2\right]$$

and thus $E\left[u_t^2\right] = \alpha_0 + (\alpha_1 + \beta_1)E[u_t^2]$ since u_t^2 is stationary. For the general $GARCH(p,q)$ model (5.3.2), the squared residuals u_t^2 behave like an $ARMA(\max(p,q),q)$ process.

One can identify the orders of the GARCH model using the correlogram of the squared residuals. They will coincide with ARMA orders of the squared residuals of the time series.

GARCH Model and Stylized Facts In practice, researchers have uncovered many so-called stylized facts about the volatility of financial time series; Bollerslev, Engle and Nelson (1994) give a complete account of these facts. Using the ARMA representation of GARCH models shows that the GARCH model is capable of explaining many of those stylized facts. This section will focus on three important ones: volatility clustering, fat tails, and volatility mean reversion. Other stylized facts are illustrated and explained in later sections.

Volatility Clustering Usually the GARCH coefficient β_1 is found to be around 0.9 for many weekly or daily financial time series. Given this value of β_1, it is obvious that large values of σ_{t-1}^2 will be followed by large values of σ_t^2, and small values of σ_{t-1}^2 will be followed by small values of σ_t^2. The same reasoning can be obtained from the ARMA representation in (5.3.2), where large/small changes in u_{t-1}^2 will be followed by large/small changes in σ_t^2.

Fat Tails It is well known that the distribution of many high frequency financial time series usually have fatter tails than a normal distribution. This means that large changes are more often to occur than under a normal distribution. Thus a GARCH model can replicate the fat tails usually observed in financial time series.

Volatility Mean Reversion Although financial markets may experience excessive volatility from time to time, it appears that volatility will eventually settle down to a long run level. The previous subsection showed that the long run variance of u_t for the stationary $GARCH(1,1)$ model is $\alpha_0/(1 - \alpha_1 - \beta_1)$. In this case, the volatility is always pulled toward this long run level.

5.3.1 Example: Simulating an $GARCH(p, q)$ model in EViews

It is slightly trickier to simulate GARCH process than the ARCH one in EViews. Since it is necessary simultaneously to generate Y_t and σ_t processes, we will need to use loop to accomplish it. Therefore, it is more convenient to use program object rather than doing it in the command line. Consider as an example $GARCH(2, 1)$ series

$$\begin{aligned} Y_t &= \sigma_t \eta_t \\ \sigma_t^2 &= 3.5 + 0.5Y_{t-1}^2 + 0.28Y_{t-2}^2 + 0.2\sigma_{t-1}^2 \end{aligned} \tag{5.3.3}$$

We start the program with the same commands as in the ARCH case; the only difference is that we generate a conditional variance process s.

```
smpl @all
series eta=nrnd
scalar n=@obs(eta)
smpl @first @first+1
series s=3.5/(0.5+0.28+0.2)
series y=@sqrt(s)*eta
```

The next part of the program creates the loop where both series Y_t and σ_t^2 are generated observation after observation.

```
for !i=2 to n-2
smpl @first+!i @first+!i
s=3.5+0.5*y(-1)^2+0.28*y(-2)^2+0.2*s(-1)
y=@sqrt(s)*eta
next
smpl @all
```

The graph of the simulated GARCH process is given on Figure ??.

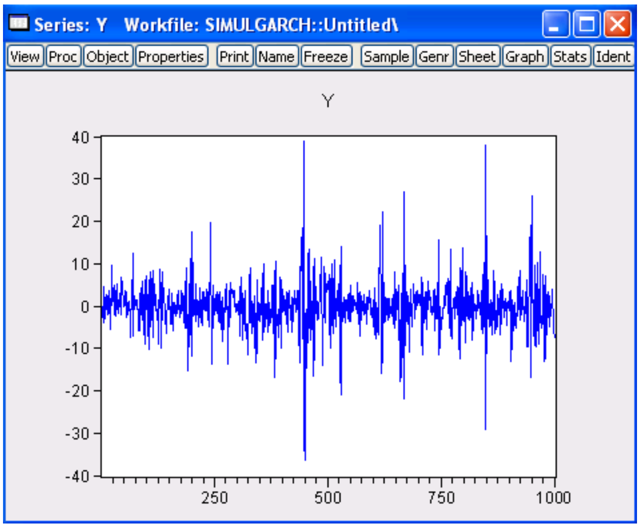


Figure 5.2: Plot of the simulated GARCH process

We can see on the graph a clear effect of volatility clustering. In most cases volatility stays low but there are several spikes with high volatility which persist for a number of periods. Another stylized fact can be seen from the histogram of the simulated observations (click on **View/Descriptive Statistic and Tests/Histogram and Stats**). Jarque-Bera test strongly rejects the null hypothesis of normality and the kurtosis is extremely high indicating fat tails of the generated distribution.

5.4 GARCH model estimation

This section illustrates how to estimate a GARCH model. Assuming that u_t follows normal or Gaussian distribution conditional on past history, the prediction error

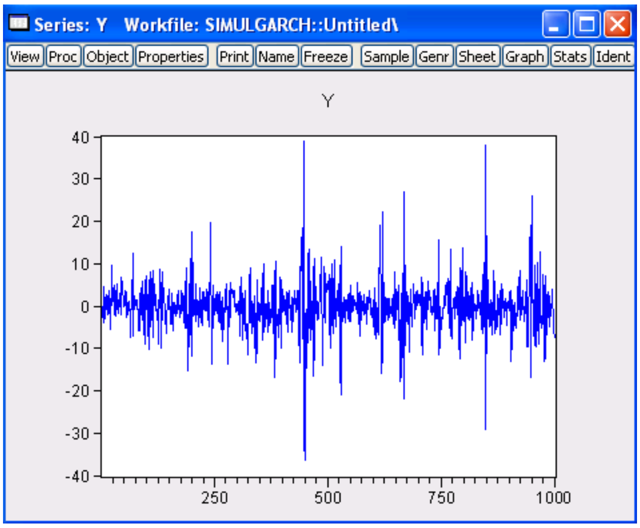


Figure 5.3: Histogram of the simulated GARCH process

decomposition of the log-likelihood function of the GARCH model conditional on initial values is:

$$\log L = -\frac{T}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{T}\log\left(\sigma_t^2\right) - \frac{1}{2}\sum_{i=1}^{T}\frac{u_t^2}{\sigma_t^2}.$$

The unknown model parameters c, α_i $(i = 0, ..., p)$ and β_j, $(j = 1, ..., q)$ can be estimated using conditional maximum likelihood estimation (MLE). Details of the maximization are given in Hamilton (1994). Once the MLE estimates of the parameters are found, estimates of the time varying volatility σ_2 $(t = 1, ..., T)$ are also obtained as a side product.

5.5 GARCH Model Extensions

In many cases, the basic GARCH model (5.3.2) provides a reasonably good model for analyzing financial time series and estimating conditional volatility. However, there are some aspects of the model which can be improved so that it can better capture the characteristics and dynamics of a particular time series.

In the basic GARCH model, since only squared residuals u_{t-i}^2 enter the equation, the signs of the residuals or shocks have no effects on conditional volatility. However, a stylized fact of financial volatility is that bad news (negative shocks) tends to have a larger impact on volatility than good news (positive shocks).

5.5.1 EGARCH Model

Nelson (1991) proposed the following exponential GARCH (EGARCH) model to allow for leverage effects:

$$h_t = \alpha_0 + \sum_{i=1}^{p} \alpha_i \frac{|u_{t-i}| + \gamma_i u_{t-i}}{\sigma_{t-i}} + \sum_{i=1}^{q} h_{t-j},$$

where $h_t = \log \sigma_t^2$. Note that when u_{t-i} is positive, the total effect of u_{t-i} is $(1 + \gamma_i)|u_{t-i}|$; in contrast, when u_{t-i} is negative, the total effect of u_{t-i} is $(1 - \gamma_i)|u_{t-i}|$. Bad news can have a larger impact on volatility, and the value of γ_i would be expected to be negative.

5.5.2 TGARCH Model

Another GARCH variant that is capable of modeling leverage effects is the threshold GARCH (TGARCH) model, which has the following form:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i u_{t-i}^2 + \sum_{i=1}^{p} \alpha_i S_{t-i} u_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2,$$

where

$$S_{t-i} = \begin{cases} 1 & u_{t-i} < 0 \\ 0 & u_{t-i} \geq 0 \end{cases}.$$

That is, depending on whether u_{t-i} is above or below the threshold value of zero, u_{t-i}^2 has different effects on the conditional variance σ_t^2: when u_{t-i} is positive, the total effects are given by $\alpha_i u_{t-i}^2$; when u_{t-i} is negative, the total effects are given by $(\alpha_i + \gamma_i)u_{t-i}^2$. So one would expect γ_i to be positive for bad news to have larger impacts.

5.5.3 PGARCH Model

The basic GARCH model can be also extended to allow for leverage effects. This is made possible by treating the basic GARCH model as a special case of the power GARCH (PGARCH) model proposed by Ding and (1993) (1993):

$$\sigma_t^d = \alpha_0 + \sum_{i=1}^{p} \alpha_i \left(|u_{t-i}| + \gamma_i u_{t-i}\right)^d + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^d, \tag{5.5.1}$$

where d is a positive exponent, and γ_i denotes the coefficient of leverage effects. Note that when $d = 2$, (5.5.1) reduces to the basic GARCH model with leverage effects.

Two Components Model The GARCH model can be used to model mean reversion in conditional volatility. Recall the mean reverting form of the basic $GARCH(1,1)$ model:

$$(u_t^2 - \bar{\sigma}^2) = (\alpha_1 + \beta_1)(u_{t-1}^2 - \bar{\sigma}^2) + \varepsilon_t - \beta_1 \varepsilon_{t-1},$$

where $\bar{\sigma}^2 = \alpha_0/(1 - \alpha_1 - \beta_1)$ is the unconditional long run level of volatility which is constant over time. Engle and Lee (1999) propose a model with time varying long run volatility level. The general form of the two components model is:

$$\sigma_t^2 = q_t + s_t \tag{5.5.2}$$

$$q_t^d = \alpha_1 |u_{t-1}|^d + \beta_1 q_{t-1}^d \tag{5.5.3}$$

$$s_t^d = \alpha_0 + \alpha_2 |u_{t-1}|^d + \beta_2 s_{t-1}^d. \tag{5.5.4}$$

The long run component q_t follows a highly persistent $PGARCH(1,1)$ model, and the transitory component s_t follows another $PGARCH(1,1)$ model.

GARCH-in-the-Mean Model In financial investment, high risk is often expected to lead to high returns. Although modern capital asset pricing theory does not imply such a simple relationship, it does suggest there are some interactions between expected returns and risk as measured by volatility. Engle, Lilien and Robins (1987) propose to extend the basic GARCH model so that the conditional volatility can generate a risk premium which is part of the expected returns. This extended GARCH model is often referred to as GARCH-in-the-mean (GARCH-M) model.

The GARCH-M model extends the conditional mean equation (5.2.1) as follows:

$$Y_t = c + \alpha g(\sigma_t) + u_t,$$

where $g(\cdot)$ can be an arbitrary function of volatility σ_t.

Exogenous Variables in Conditional Mean So far the conditional mean equation has been restricted to a constant when considering GARCH models, except for the GARCH-M model where volatility was allowed to enter the mean equation as an explanatory variable. It is possible to add ARMA terms as well as exogenous explanatory variables in the conditional mean equation. A more general form for the conditional mean equation is

$$Y_t = c + \delta' X_t + u_t$$

where X_t is a $k \times 1$ vector of regressors and δ is a vector of coefficients.

Also, one can add explanatory variables into the conditional variance formula which may have impacts on conditional volatility.

Error Distributions In all the examples illustrated so far, a normal error distribution has been exclusively used. However, given the well known fat tails in financial time series, it may be more desirable to use a distribution which has fatter tails than the normal distribution.

EViews allows two fat-tailed error distributions for fitting GARCH models: the Student t distribution and the generalized error distribution.

5.5.4 Prediction

GARCH models are frequently used to forecast volatility of return. It is straightforward to forecast the conditional variance from an ARCH model. Assuming that the model parameters are known, the one-period ahead forecast is

$$\sigma^2_{t+1|t} = \alpha_0 + \alpha_1 u_t^2 + ... + \alpha_p u^2_{t-p+1}$$

Forecasting the conditional volatility for h periods ahead can be done by a recursion

$$\sigma^2_{t+h|t} = \alpha_0 + \alpha_1 \sigma^2_{t+h-1|t} + ... + \alpha_p \sigma^2_{t+h-p},$$

where $\sigma^2_{t+j} = u^2_{t+j}$ for $j \leq 0$.

The h-period ahead variance forecast for a $GARCH(1,1)$ model is

$$\sigma^2_{t+1|t} = \alpha_0 \left[\sum_{i=0}^{h-1} (\alpha_1 + \beta_1)^i \right] + (\alpha_1 + \beta_1)^h \sigma_t^2.$$

5.5.5 Example: GARCH Estimation

As an example of GARCH model estimation in EViews we consider a series of 2 minutes exchange rates between the Euro and the British Pound for 21 August 2007 between 7:00 and 16:00 GMT. The data is contained in the file EURGBP.wf1. Plot of the EUR/GBP returns is given in Figure ??.

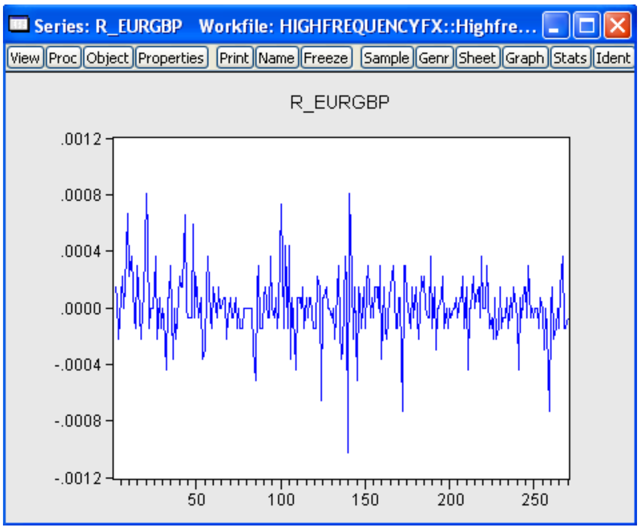


Figure 5.4: 2 minutes EUR/GBP returns on 21 August 2007

We can clearly see periods of high and low volatility of the returns, thus an ARCH type model can be appropriate to model volatility.

Let us first estimate an OLS regression of returns on a constant term. This will give us an opportunity to test for the presence of ARCH effect more formally. Having typed

ls r_eurgbp c

in the command line and pressed **Enter**, go to **View/Residual Tests/ Heteroscedasticity Tests...** in the equation object window and choose **ARCH** option there. The test result is given in Figure **??**

Equation: UNTITLED Workfile: HIGHFREQUENCYFX::Hi...

View | Proc | Object | Print | Name | Freeze | Estimate | Forecast | Stats | Resids

Heteroskedasticity Test: ARCH

F-statistic	7.169664	Prob. F(1,266)	0.0079
Obs*R-squared	7.033980	Prob. Chi-Square(1)	0.0080

Test Equation:
Dependent Variable: RESID^2
Method: Least Squares
Date: 01/09/09 Time: 12:36
Sample (adjusted): 3 270
Included observations: 268 after adjustments

	Coefficient	Std. Error	t-Statistic	Prob.
C	4.39E-08	7.80E-09	5.634401	0.0000
RESID^2(-1)	0.162027	0.060511	2.677623	0.0079

R-squared	0.026246	Mean dependent var	5.24E-08
Adjusted R-squared	0.022585	S.D. dependent var	1.18E-07
S.E. of regression	1.17E-07	Akaike info criterion	-29.08398
Sum squared resid	3.62E-12	Schwarz criterion	-29.05718
Log likelihood	3899.254	Hannan-Quinn criter.	-29.07322
F-statistic	7.169664	Durbin-Watson stat	1.986218
Prob(F-statistic)	0.007876		

Figure 5.5: ARCH test results

The p-value of the test is very small which rejects the null hypothesis of homoscedasticity of residuals in favor of ARCH alternative. Thus, based on this result we decide to estimate regression with ARCH specification. Go to **Quick/Estimate Equation** option of the main workfile menu and specify the model as you were specifying it for the OLS regression. That is, type r_eurusd c in the **Equation Specification** box. Now, in the **Method** field, choose **ARCH – Autoregressive Conditional Heteroscedasticity**. This will open you more option for ARCH model specification. In the **ARCH-M** we can indicate whether we want to include $ARCH$-in-mean term in the equation and, if yes, whether variance or standard deviation should enter it. In the **Variance and distribution specification** part we can select between simple ARCH/GARCH/TARCH model, EGARCH, PGARCH or two component GARCH model. We will stay with the simplest first option. In the **Order** field we should write orders of ARCH and GARCH terms in the variance equation. Let us specify GARCH(3,3) model, so enter 3 and 3 respectively in each of the box. If we do not want to include GARCH terms, simply put 0 in front of the **GARCH** field. **Variance regressors** box will be useful if we want to include some exogenous variables in the variance equation. **Errors distribution** box allows us to choose between Gaussian and Student - distributions of the error term. The model estimation output is given in Figure **??**.

Equation: UNTITLED Workfile: HIGHFREQUENCYFX::Hi...

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Dependent Variable: R_EURGBP
Method: ML - ARCH (Marquardt) - Normal distribution
Date: 01/09/09 Time: 12:51
Sample (adjusted): 2 270
Included observations: 269 after adjustments
Convergence achieved after 198 iterations
Presample variance: backcast (parameter = 0.7)
GARCH = C(2) + C(3)*RESID(-1)^2 + C(4)*RESID(-2)^2 + C(5)*RESID(-3)^2 + C(6)*GARCH(-1) + C(7)*GARCH(-2) + C(8)*GARCH(-3)

	Coefficient	Std. Error	z-Statistic	Prob.
C	1.99E-05	1.43E-05	1.390795	0.1643
	Variance Equation			
C	2.94E-08	7.73E-09	3.809684	0.0001
RESID(-1)^2	0.049098	0.046037	1.066483	0.2862
RESID(-2)^2	-0.056776	0.051782	-1.096441	0.2729
RESID(-3)^2	0.177889	0.056450	3.151275	0.0016
GARCH(-1)	0.520865	0.274545	1.897193	0.0578
GARCH(-2)	0.113259	0.317913	0.356259	0.7216
GARCH(-3)	-0.349023	0.159471	-2.188629	0.0286

R-squared	-0.001633	Mean dependent var	1.07E-05
Adjusted R-squared	-0.028496	S.D. dependent var	0.000229
S.E. of regression	0.000232	Akaike info criterion	-13.91588
Sum squared resid	1.41E-05	Schwarz criterion	-13.80897
Log likelihood	1879.686	Hannan-Quinn criter.	-13.87295
Durbin-Watson stat	2.149537		

Figure 5.6: GARCH estimation output

The resid terms of the output correspond to α_i coefficients (ARCH terms) and GARCH terms correspond to β_i coefficients in (5.3.2).

We can see that α_3 and β_3 coefficients are statistically significant and α_2 is on a border line of significance.

In **View/Representation** section one can find the variance specification. Also EViews allow to plot both standardized and on-standardized residual plots (in **Actual, Fitted, Residuals**), test for parameter constancy, linear restriction, build correlogram of residuals and squared residuals in the same way as it is done for the OLS regression.

In order to estimate the above model using the command line one should type

```
equation e.arch(3,3) r_eurgbp c
```

where the term arch indicates that an ARCH estimation method should be used, order of the ARCH and GARCH terms follow in parentheses. Then you should specify the conditional mean equation as it is done on the least squares model case (the dependent variable should be in the first place).

Chapter 6

Multivariate Time Series Analysis

6.0.1 Introduction

Multivariate analysis investigates dependence and interactions among a set of variables in multi-values processes. One of the most powerful method of analyzing multivariate time series is the vector autoregression model. It is a natural extension of the univariate autoregressive model to the multivariate case.

In this chapter we cover concepts of VAR modelling, non-stationary multivariate time series and cointegration.

More detailed discussion can be found in Hamilton (1994), Harris (1995), Enders (2004), Tsay (2002), Zivot and Wang (2006).

6.1 Vector Autoregression Model

Let $\mathbf{Y}_t = (Y_{1,t}, Y_{2,t}, ..., Y_{n,t})'$ denote an $k \times 1$ vector of time series variables. The basic vector autoregressive model of order p, $VAR(p)$, is

$$\mathbf{Y}_t = \mathbf{c} + \Pi_1 \mathbf{Y}_{t-1} + \Pi_2 \mathbf{Y}_{t-2} + ... + \Pi_p \mathbf{Y}_{t-p} + \mathbf{u}_t, \quad t = 1, ..., T, \tag{6.1.1}$$

where Π_i are $k \times k$ matrices of coefficients, $\mathbf{c}$ is a $k \times 1$ vector of constants and $\mathbf{u}_t$ is an $k \times 1$ unobservable zero mean white noise vector process with covariance matrix Σ.

If we consider a special case of two dimensional vector $\mathbf{Y}$, the VAR consists of two equation (also called a bivariate VAR)

$$\begin{aligned} \mathbf{Y}_t &= \begin{pmatrix} Y_{1,t} \\ Y_{2,t} \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} \pi_{11}^1 & \pi_{12}^1 \\ \pi_{21}^1 & \pi_{22}^1 \end{pmatrix} \begin{pmatrix} Y_{1,t-1} \\ Y_{2,t-1} \end{pmatrix} \\ &+ \begin{pmatrix} \pi_{11}^2 & \pi_{12}^2 \\ \pi_{21}^2 & \pi_{22}^2 \end{pmatrix} \begin{pmatrix} Y_{1,t-2} \\ Y_{2,t-2} \end{pmatrix} + \begin{pmatrix} u_{1,t} \\ u_{2,t} \end{pmatrix} \end{aligned} \tag{6.1.2}$$

with $cov\,(u_{1,t}, u_{2,s}) = \sigma_{12}$ for $t \neq s$.

As in the univariate case with AR processes, we can use the lag operator to represent $VAR(p)$

$$\Pi(L)\mathbf{Y}_t = \mathbf{c} + \mathbf{u}_t,$$

where $\Pi(L) = I_n - \Pi_1 L - ... - \Pi_p L^p$.

If we impose stationarity on $\mathbf{Y}_t$ in (6.1.2), the unconditional expected value is given by

$$\mu = (I_n - \Pi_1 - ... - \Pi_p)^{-1}\mathbf{c}.$$

Very often other deterministic terms or stochastic exogenous variables may be included into the VAR specification to represent. More general form of the $VAR(p)$ model is

$$\mathbf{Y}_t = \Pi_1\mathbf{Y}_{t-1} + \Pi_2\mathbf{Y}_{t-2} + ... + \Pi_p\mathbf{Y}_{t-p} + \Gamma\mathbf{X}_t + \mathbf{u}_t,$$

where $\mathbf{X}_t$ represents an $m \times 1$ matrix of exogenous or deterministic variables, and Γ is a matrix of parameters.

6.1.1 Estimation of VARs and Inference on coefficients

Since the $VAR(p)$ may be written as a system of equations with the same sets of explanatory variables, its coefficients can be efficiently and consistently estimated by estimating each of the components using the OLS method (see Hamilton (1994)). Under standard assumptions regarding the behavior of stationary and ergodic VAR models (see Hamilton (1994) the estimators of the coefficients are asymptotically normally distributed.

An element of $\hat{\Pi}_i$ is asymptotically normally distributed, so asymptotically valid t-tests on individual coefficients may be constructed in the usual way (see Chapter 2). More general linear hypotheses can also be tested using the Wald statistic.

Lag Length Selection A reasonable strategy how to determine the lag length of the VAR model is to fit $VAR(p)$ models with different orders $p = 0, ..., p_{max}$ and choose the value of p which minimizes some model selection criteria. Model selection criteria for $VAR(p)$ could be based on Akaike (AIC), Schwarz-Bayesian (BIC) and Hannan-Quinn (HQ) information criteria:

$$\begin{aligned} AIC(p) &= \ln|\bar{\Sigma}(p)| + \frac{2}{T}pn^2 \\ BIC(p) &= \ln|\bar{\Sigma}(p)| + \frac{\ln T}{T}pn^2 \\ HQ(p) &= \ln|\bar{\Sigma}(p)| + \frac{2\ln\ln T}{T}pn^2 \end{aligned}$$

Forecasting We can use VAR model to forecast times series in a similar way to forecasting from a univariate AR model.

The one-period-ahead forecast based on information available at time T is

$$\mathbf{Y}_{T+1|T} = \mathbf{c} + \Pi_1\mathbf{Y}_T + ... + \Pi_p\mathbf{Y}_{T-p+1}$$

while h-step forecast is

$$\mathbf{Y}_{T+h|T} = \mathbf{c} + \Pi_1\mathbf{Y}_T + ... + \Pi_p\mathbf{Y}_{T-p+1},$$

where $\mathbf{Y}_{T+j|T} = \mathbf{Y}_{T+j}$ for $j < 0$. The h-step forecast errors may be expressed as

$$\mathbf{Y}_{T+h} - \mathbf{Y}_{T+h|T} = \sum_{s=0}^{h-1}\Psi_s\varepsilon_{T+h-s},$$

where the matrices Ψ_s are determined by recursive substitution

$$\Psi_s = \sum_{j=1}^{p-1}\Psi_{s-j}\Pi_j \tag{6.1.3}$$

with $\Psi_0 = I_n$ and $\Pi_j = 0$ for $j > p$. The forecasts are unbiased since all of the forecast errors have expectation zero and the MSE matrix for $\mathbf{Y}_{t+h|T}$ is

$$\Sigma(h) = MSE\left(\mathbf{Y}_{T+h} - \mathbf{Y}_{T+h|T}\right) = \sum_{s=0}^{h-1} \Psi_s \Sigma \Psi_s'.$$

The h-step forecast in the case of estimated parameters is

$$\hat{\mathbf{Y}}_{T+h|T} = \hat{\Pi}_1 \hat{\mathbf{Y}}_{T+h-1|T} + ... + \hat{\Pi}_p \hat{\mathbf{Y}}_{T+h-p|T},$$

where $\hat{\Pi}_j$ are the estimated matrices of parameters. The h-step forecast error is now

$$\mathbf{Y}_{T+h} - \hat{\mathbf{Y}}_{T+h|T} = \sum_{s=0}^{h-1} \Psi_s \varepsilon_{T+h-s} + \left(\mathbf{Y}_{t+h} - \hat{\mathbf{Y}}_{T+h|T}\right)$$

The estimate of the MSE matrix of the h-step forecast is then

$$\hat{\Sigma}(h) = \sum_{s=0}^{h-1} \hat{\Psi}_s \hat{\Sigma} \hat{\Psi}_s'$$

with $\Psi_s = \sum_{j=1}^{s} \hat{\Psi}_{s-j} \hat{\Pi}_j$.

6.1.2 Granger Causality

One of the main uses of VAR models is forecasting. The structure of the VAR model provides information about a variable's or a group of variables' forecasting ability for other variables. The following intuitive notion of a variable's forecasting ability is due to Granger (1969). If a variable, or group of variables, Y_1 is found to be helpful for predicting another variable, or group of variables, Y_2 then Y_1 is said to Granger-cause Y_2; otherwise it is said to fail to Granger-cause Y_2. Formally, Y_1 fails to Granger-cause Y_2 if for all $s > 0$ the MSE of a forecast of $Y_{2,t+s}$ based on $(Y_{2,t}, Y_{2,t-1}, ...)$ is the same as the MSE of a forecast of $Y_{2,t+s}$ based on $(Y_{2,t}, Y_{2,t-1}, ...)$ and $(Y_{1,t}, Y_{1,t-1}, ...)$. Note that the notion of Granger causality only implies forecasting ability.

In a bivariate $VAR(p)$ model for $\mathbf{Y}_t = (Y_{1t}, Y_{2t})'$, Y_2 fails to Granger-cause Y_1 if all of the p VAR coefficient matrices $\Pi_1, ..., \Pi_p$ are lower triangular. That is, all of the coefficients on lagged values of Y_2 are zero in the equation for Y_1. The p linear coefficient restrictions implied by Granger non-causality may be tested using the Wald statistic. Notice that if Y_2 fails to Granger-cause Y_1 and Y_1 fails to Granger-cause Y_2, then the VAR coefficient matrices $\Pi_1, ..., \Pi_p$ are diagonal.

6.1.3 Impulse Response and Variance Decompositions

As in the univariate case, a $VAR(p)$ process can be represented in the form of a vector moving average (VMA) process.

$$\mathbf{Y}_t = \mu + \mathbf{u}_t + \Psi_1 \mathbf{u}_{t-1} + \Psi_2 \mathbf{u}_{t-2} + ...,$$

where the $k \times k$ moving average matrices Ψ_s are determined recursively using (6.1.3).

The elements of coefficient matrices Ψ_s mean effects of $\mathbf{u}_{t-s}$ shocks on $\mathbf{Y}_t$. That is, the (i, j)-th element, ψ_{ij}^s, of the matrix Ψ_s is interpreted as the impulse response

$$\frac{\partial Y_{i,t+s}}{\partial u_{j,t}} = \frac{\partial Y_{i,t}}{\partial u_{j,t-s}} = \psi_{ij}^s, \qquad i, j = 1, ..., T.$$

Sets of coefficients $\psi_{ij}(s) = \psi_{ij}^s$, $i, j = 1, ..., T$ are called the **impulse response functions**.

It is possible to decompose the h-step-ahead forecast error variance into the proportions due to each shock u_{jt}.

The forecast variance decomposition determines the proportion of the variation Y_{jt} due to the shock u_{jt} versus shocks of other variables u_{it} for $i \neq j$.

6.1.4 VAR in EViews

As an example of VAR estimation in EViews, consider two time series of returns of monthly IBM stocks and the market portfolio returns from Fama-French database (data is contained in IBM1.wf1).

There are several ways to estimate VAR model in EViews. The first one is through the main menu. Clicking on **View/Estimate VAR...** will open a dialog window for VAR model estimation.

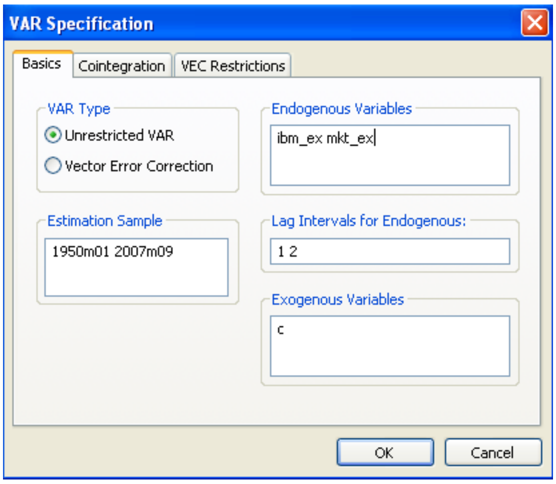


Figure 6.1: VAR model estimation dialog window

We choose **Unrestricted VAR** and in the **Endogenous Variables** box we have to specify the list of endogenous time series variables to be included in the VAR model. We consider two excess return series of the IBM stock IBM_ex and the market portfolio Mkt_ex.

In the **Lag Intervals for Endogenous** we have to specify the order of the model, that is interval of lags to be included in the model. If we want to build a model with only two lags, we write 1 2. This means, we include all lags beginning from the first one and ending with the lag of order 2. We do not specify any exogenous variables apart from the intercept term c.

Another way of calling the VAR estimation dialog window is to select both endogenous variables in the workfile and in the context menu (right button click) choose **Open/as VAR...**. The **Endogenous Variables** box will be filled in automatically.

Finally, we can estimate VAR model from the command line. There is a separate object, called **var**, to declare the VAR model. The estimation of the above mentioned example will look like var ibm2.ls 1 2 ibm_ex mkt_ex

Here ibm2 is a name of the var-object which will be saved in the workfile, ls indicates the estimation method; in this case it is OLS estimation method of the unrestricted VAR model. Then, specifications of the lags pairs and the list of endogenous variables follow. If one wishes to include exogenous variables besides the intercept, it can be done by typing a symbol @ followed by a list of exogenous variables. For example,

var ibm2.ls 1 2 ibm_ex mkt_ex @ exvar1 exvar2

Click **OK** and EViews produces an estimation output for the specified VAR model.

Var: IBM2 Workfile: IBM2::Data_ibm_ff\

View Proc Object Print Name Freeze Estimate Stats Impulse Resids

Vector Autoregression Estimates

Vector Autoregression Estimates
Date: 01/03/09 Time: 11:54
Sample (adjusted): 1950M03 2007M09
Included observations: 691 after adjustments
Standard errors in ( ) & t-statistics in [ ]

	IBM_EX	MKT_EX
IBM_EX(-1)	0.009969	0.004618
	(0.04688)	(0.02849)
	[0.21267]	[0.16208]
IBM_EX(-2)	0.061395	0.011865
	(0.04689)	(0.02850)
	[1.30944]	[0.41630]
MKT_EX(-1)	0.069736	0.074885
	(0.07747)	(0.04709)
	[0.90017]	[1.59023]
MKT_EX(-2)	-0.198868	-0.044936
	(0.07745)	(0.04708)
	[-2.56754]	[-0.95443]
C	0.008388	0.001999
	(0.00268)	(0.00163)
	[3.12617]	[1.22556]
R-squared	0.011416	0.007309
Adj. R-squared	0.005651	0.001520
Sum sq. resids	3.306425	1.221712
S.E. equation	0.069425	0.042201
F-statistic	1.980399	1.262645
Log likelihood	865.2686	1209.253
Akaike AIC	-2.489923	-3.485538
Schwarz SC	-2.457086	-3.452700
Mean dependent	0.008725	0.002207
S.D. dependent	0.069622	0.042233
Determinant resid covariance (dof adj.)		5.64E-06
Determinant resid covariance		5.56E-06
Log likelihood		2219.343
Akaike information criterion		-6.394625
Schwarz criterion		-6.328950

Figure 6.2: Output for the VAR model estimation

Two columns correspond to two equation in the VAR model. The only significant coefficient besides the intercept one is at the second lag of the market portfolio returns in the IBM equation. As expected, there is a unidirectional dynamic relationship from the market portfolio returns to the IBM returns, Thus, the IBM return is affected by the past movements of the market while past movements of IBM stock returns do not affect the market portfolio returns. The second equation (for market portfolio) is not significant as suggested by the F-statistics. This means that the the estimated model cannot explain variation in the market portfolio returns. This can happen because we possibly omitted some important exogenous variables or the order of the model is inappropriately selected. EViews provides a tool to choose the most suitable lag order. In the workfile menu choose **View/Lag Structure/Lag Length Criteria...** to determine the optimal model structure. In the appeared **Lag Specification** window we choose $p_{max} = 8$ (maximal lag order).

All criteria indicate that the optimal lag order of the model is 0. This means that the VAR model is inappropriate model to explain IBM and market portfolio returns. Indeed, we know from the CAPM that market portfolio returns affect the stock returns contemporaneously and are not in lag relationship. Thus, either additional exogenous factors should be found to include in the model or another structure of the model should be employed in this case.

Var: IBM2 Workfile: IBM2::Data_ibm_ff\

View Proc Object Print Name Freeze Estimate Stats Impulse Resids

VAR Lag Order Selection Criteria
Endogenous variables: IBM_EX MKT_EX
Exogenous variables: C
Date: 01/03/09 Time: 12:27
Sample: 1950M01 2007M09
Included observations: 685

Lag	LogL	LR	FPE	AIC	SC	HQ
0	2194.622	NA*	5.69e-06*	-6.401817*	-6.388593*	-6.396700*
1	2196.775	4.287075	5.72e-06	-6.396424	-6.356751	-6.381073
2	2199.864	6.131986	5.73e-06	-6.393763	-6.327640	-6.368177
3	2201.623	3.481326	5.77e-06	-6.387219	-6.294647	-6.351399
4	2203.219	3.150236	5.81e-06	-6.380200	-6.261179	-6.334146
5	2205.939	5.354029	5.83e-06	-6.376465	-6.230995	-6.320176
6	2206.141	0.395622	5.90e-06	-6.365375	-6.193455	-6.298852
7	2206.432	0.568871	5.96e-06	-6.354545	-6.156177	-6.277787
8	2207.274	1.642798	6.02e-06	-6.345326	-6.120508	-6.258334

* indicates lag order selected by the criterion
LR: sequential modified LR test statistic (each test at 5% level)
FPE: Final prediction error
AIC: Akaike information criterion
SC: Schwarz information criterion
HQ: Hannan-Quinn information criterion

Figure 6.3: Output for the lag length selection procedure

Lag selection can be programmed manually in the same way as it is done for ARMA model (see Chapter 3). There are some command references given below which can be used to assess various statistic values in the VAR analysis in EViews.

6.2 Cointegration

The assumption of stationary of regressors and regressands is crucial for the properties and the OLS estimators discussed in Chapter 2. In this case, the usual statistical results for the linear regression model and consistency of estimators hold. However, when variables are non-stationary then the usual statistical results may not hold.

6.2.1 Spurious Regression

If there are trends in the data (deterministic or stochastic) this can lead to a spurious results when running OLS regression. This is because time trend will dominate other stationary variables and the OLS estimators will pick up covariances generated by time trends only. While the effects of deterministic trends can be removed from the regression by either including time trend regressor or simply de-trending variables, non-stationary variables with stochastic trends may lead to invalid inferences.

Var Data Members

Data Member	Description
@eqlogl(k)	log likelihood for equation k
@eqncoef(k)	number of estimated coefficients in equation k
@eqregobs(k)	number of observations in equation k
@meandep(k)	mean of the dependent variable in equation k
@r2(k)	R-squared statistic for equation k
@rbar2(k)	adjusted R-squared statistic for equation k
@sddep(k)	standard deviation of dependent variable in equation k
@se(k)	standard error of the regression in equation k
@ssr(k)	sum of squared residuals in equation k
@aic	Akaike information criterion for the system
@detresid	determinant of the residual covariance matrix
@hq	Hannan-Quinn information criterion for the system
@logl	log likelihood for system
@ncoefs	total number of estimated coefficients in the var
@neqn	number of equations
@regobs	number of observations in the var
@sc	Schwarz information criterion for the system
@svarcvgtype	Returns an integer indicating the convergence type of the structural decomposition estimation: 0 (convergence achieved), 2 (failure to improve), 3 (maximum iterations reached), 4 (no convergence-structural decomposition not estimated)
@svaroverid	over-identification LR statistic from structural factorization
@totalobs	sum of "@eqregobs" from each equation ("@regobs*@neqn")
@coefmat	coefficient matrix (as displayed in output table)
@coefse	matrix of coefficient standard errors (corresponding to the output table)
@impfact	factorization matrix used in last impulse response view
@lrrsp	accumulated long-run responses from last impulse response view
@lrrspse	standard errors of accumulated long-run responses
@residcov	covariance matrix of the residuals
@svaramat	estimated A matrix for structural factorization
@svarbmat	estimated B matrix for structural factorization
@svarcovab	covariance matrix of stacked A and B matrix for structural factorization
@svarrcov	restricted residual covariance matrix from structural factorization

Consider, for example,

$$\begin{aligned} Y_{1,t} &= Y_{1,t-1} + u_{1,t}, \qquad && u_{1,t} \sim IN(0,1) \\ Y_{2,t} &= Y_{2,t-1} + u_{2,t}, \qquad && u_{2,t} \sim IN(0,1) \end{aligned}$$

Both of the variables are non-stationary and independent from each other. In the regression $Y_{1,t} = \beta_0 + \beta_1 Y_{2,t} + \varepsilon_t$, the value of true slope parameter $\beta_1 = 0$. Thus, the value of the OLS estimate $\hat{\beta}_1$ should be insignificant. The actual estimations produce high R^2 coefficients and highly significant β_1.

The problem with the spurious regression is that t- and F-statistics do not follow standard distributions. As shown in Phillips (1986), $\hat{\beta}_1$ does not converge in probability to zero, R^2 converges to unity as $T \to \infty$ so that the model will appear to fit well even though it is misspecified.

Regression with $I(1)$ data only makes sense when the data are *cointegrated*.

6.2.2 Cointegration

Let $\mathbf{Y}_t = (Y_{1t}, ..., Y_{kt})'$ denote an $k \times 1$ vector of $I(1)$ time series. $\mathbf{Y}_t$ is cointegrated if there exists an $k \times 1$ vector $\beta = (\beta_1, ..., \beta_k)'$ such that

$$\mathbf{Z}_t = \beta' \mathbf{Y}_t = \beta_1 Y_{1t} + ... + \beta_k Y_{kt} \sim I(0). \tag{6.2.1}$$

The non-stationary time series in $\mathbf{Y}_t$ are cointegrated if there is a linear combination of them that is stationary. If some elements of β are equal to zero then only the subset of the time series in $\mathbf{Y}_t$ with non-zero coefficients is cointegrated.

There may be different vectors β such that $\mathbf{Z}_t = \beta' \mathbf{Y}_t$ is stationary. In general, there can be $0 < r < k$ linearly independent cointegrating vectors. All cointegrating vectors form a *cointegrating matrix* $\mathbf{B}$. This matrix is again not unique. Some normalization assumption is required to eliminate ambiguity from the definition.

A typical normalization is

$$\beta = (1, -\beta_2, ..., -\beta_k)'$$

so that the cointegration relationship may be expressed as

$$\mathbf{Z}_t = \beta'\mathbf{Y}_t = Y_{1t} - \beta_2 Y_{2t} - ... - \beta_k Y_{kt} \sim I(0).$$

6.2.3 Error Correction Models

Engle and Granger (1987) state that if a bivariate $I(1)$ vector $\mathbf{Y}_t = (Y_{1t}, Y_{2t})'$ is cointegrated with cointegrating vector $\beta = (1, -\beta_2)'$ then there exists an error correction model (ECM) of the form

$$\Delta Y_{1t} = \delta_1 + \phi_1(Y_{1,t-1} - \beta_1 Y_{2,t-1} + \sum_{j=1} \alpha_{11}^j \Delta Y_{1,t-j} + \sum_{s=1} \alpha_{12}^j \Delta Y_{2,t-j} + \varepsilon_{1t} \quad (6.2.2)$$

$$\Delta Y_{2t} = \delta_2 + \phi_2(Y_{1,t-1} - \beta_2 Y_{2,t-1} + \sum_{j=1} \alpha_{21}^j \Delta Y_{1,t-j} + \sum_{s=1} \alpha_{22}^j \Delta Y_{2,t-j} + \varepsilon_{2t} \quad (6.2.3)$$

that describes the long-term relations of Y_{1t} and Y_{2t}. If both time series are $I(1)$ but are cointegrated (have a long-term stationary relationship), there is a force that brings the error term back towards zero. If the cointegrating parameter β_1 or β_2 is known, the model can be estimated by the OLS method.

6.2.4 Tests for Cointegration: The Engle-Granger Approach

Engle and Granger (1987) show that if there is a cointegrating vector, a simple two-step residual-based testing procedure can be employed to test for cointegration. In this case, a long-run equilibrium relationship between components of $\mathbf{Y}_t$ can be estimated by running

$$Y_{1,t} = \beta\mathbf{Y}_{2,t} + u_t, \quad (6.2.4)$$

where $\mathbf{Y}_{2,t} = (Y_{2,t}, ..., Y_{k,t})'$ is an $(k-1) \times 1$ vector. To test the null hypothesis that $\mathbf{Y}_t$ is not cointegrated, we should test whether the residuals $\hat{u}_t \sim I(1)$ against the alternative $\hat{u}_t \sim I(0)$. This can be done by any of the tests for unit roots. The most commonly used is the augmented Dickey-Fuller test with the constant term and without the trend term. Critical values for this test is tabulated in Phillips and Ouliaris (1990) or MacKinnon (1996).

Potential problems with Engle-Granger approach is that the cointegrating vector will not involve $Y_{1,t}$ component. In this case the cointegrating vector will not be consistently estimated from the OLS regression leading to spurious results. Also, if there are more than one cointegrating relation, the Engle-Granger approach cannot detect all of them.

Estimation of the static model (6.2.4) is equivalent to omitting the short-term components from the error-correction model (6.2.3). If this results for autocorrelation in residuals, although the results will still hold asymptotically, it might create a severe bias in finite samples. Because of this, it makes sense to estimate the full dynamic model. Since all variables in the ECM are $I(0)$, the model can be consistently estimated using the OLS method. This approach leads to a better performance as it does not push the short-term dynamics into residuals.

6.2.5 Example in EViews: Engle-Granger Approach

Consider as an example the Forward Premium Puzzle. Due to rational expectation hypothesis, forward rate should be unbiased predictor of future spot exchange rate. This means that in the regression of levels of spot S_{t+1} on forward rate F_t the intercept coefficient should be equal to zero and the slope coefficient should be equal to unity.

Consider monthly data of the USG/GBP spot and forward exchange rate for the period from January 1986 to November 2008 (the data is in FPP.wf1 file).

Unit roots are often found in in levels of spot and forward exchange rates. Augmented Dickey-Fuller test statistic values are -2.567 and -2.688 which are high enough to fail rejecting the null hypothesis at 5% significance level. Phillips-Perron test produces test statistic which value os on the border of the rejection region. Thus, if two series are not cointegrated, there is a danger to obtain spurious results from the OLS regression. However, if we look at plots of the two series we can see that they co-move together very closely, so we can expect existence of cointegrating relation between them.

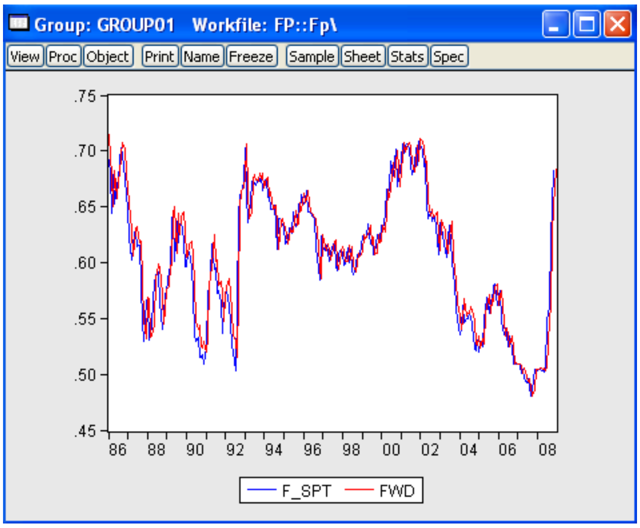


Figure 6.4: Plots of forward and future spot USD/GBP exchange rates

To perform Engle-Granger test for cointegration let us run OLS regression $S_{t+1} = \beta F_t + u_t$ in EViews and generate residuals from the model.

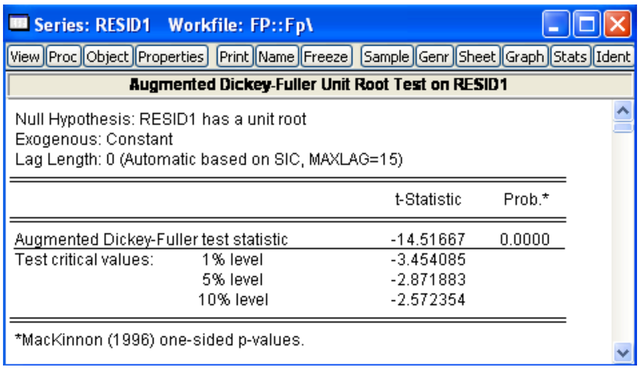
ls f_spt fwd

series resid1=resid

The second step is to test the residuals for stationarity. Augmented Dickey-Fuller test strongly rejects the presence of a unit root in the residual series in the favour of stationarity hypothesis.

Similar results are generated by other testing procedures. Thus, we conclude that future spot and forward exchange rates are cointegrated. Hence, the OLS results are valid for the regression in levels as well. In this case the slope coefficient is equal to 0.957 which is positive and close to unity. However, we reject the null hypothesis $H_0\colon \beta_1 = 1$ with the Wald test.

Thus, the forward premium puzzle also exists even for the model in levels for the exchange rates.


Series: RESID1 Workfile: FP::Fp\

View Proc Object Properties Print Name Freeze Sample Genr Sheet Graph Stats Ident

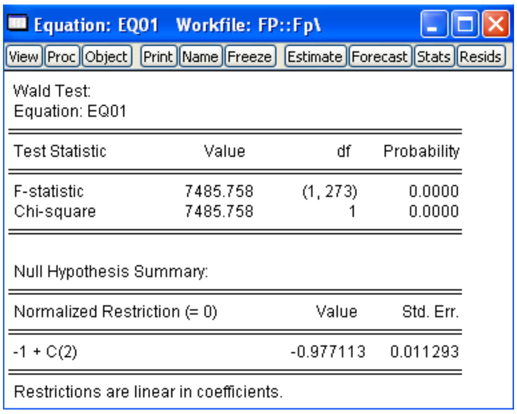
Augmented Dickey-Fuller Unit Root Test on RESID1

Null Hypothesis: RESID1 has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic based on SIC, MAXLAG=15)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-14.51667	0.0000
Test critical values:	1% level	-3.454085	
	5% level	-2.871883	
	10% level	-2.572354	

*MacKinnon (1996) one-sided p-values.

Figure 6.5: Results of Augmented Dickey-Fuller test for residuals from the long-run equilibrium relationship


Equation: EQ01 Workfile: FP::Fp\

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Wald Test:
Equation: EQ01

Test Statistic	Value	df	Probability
F-statistic	7485.758	(1, 273)	0.0000
Chi-square	7485.758	1	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-1 + C(2)	-0.977113	0.011293

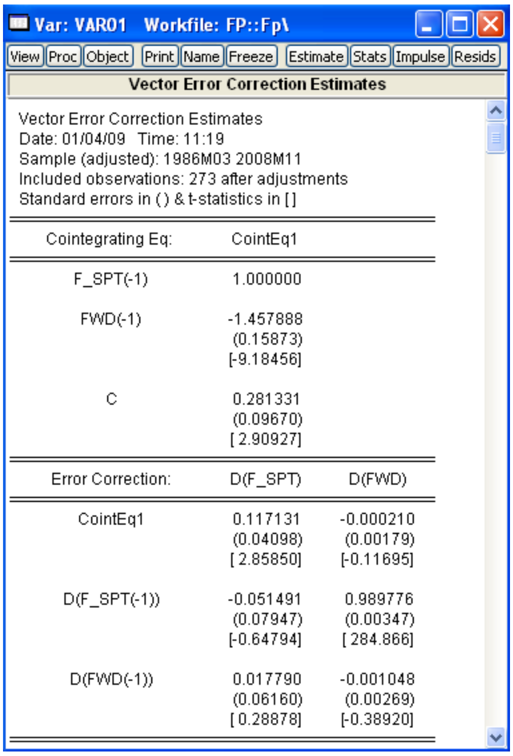
Restrictions are linear in coefficients.

Figure 6.6: Wald test results for testing $H_0\colon \beta_1 = 1$

Another way of estimating cointegrating equation is to estimate a vector error correction model. To do this, open both forward and spot series as VAR system (select both series and in the context menu choose **Open/as VAR...**). In the VAR type box select **Vector Error Correction** and in the **Cointegration** tab click on **Intercept (no trend) in CE - no intercept in VAR**. EViews' output is given in Figure **??**.

As expected, the output shows that the stationary series is approximately $S_{t+1} - F_t$ with the mean around zero. Deviations from the long-run equilibrium equation have significant effect on changes of the spot exchange rate. Another highly significant coefficient α_{22}^1 indicates a significant impact of ΔS_t on ΔF_t which is not surprising. This underlies the relationships between the spot and forward rate through the Covered Interest rate Parity condition (CIP).

The following subsection introduces an approach of testing for cointegration



Var: VAR01 Workfile: FP::Fp\

View Proc Object Print Name Freeze Estimate Stats Impulse Resids

Vector Error Correction Estimates

Vector Error Correction Estimates
Date: 01/04/09 Time: 11:19
Sample (adjusted): 1986M03 2008M11
Included observations: 273 after adjustments
Standard errors in ( ) & t-statistics in [ ]

Cointegrating Eq:	CointEq1
F_SPT(-1)	1.000000
FWD(-1)	-1.457888
	(0.15873)
	[-9.18456]
C	0.281331
	(0.09670)
	[2.90927]

Error Correction:	D(F_SPT)	D(FWD)
CointEq1	0.117131	-0.000210
	(0.04098)	(0.00179)
	[2.85850]	[-0.11695]
D(F_SPT(-1))	-0.051491	0.989776
	(0.07947)	(0.00347)
	[-0.64794]	[284.866]
D(FWD(-1))	0.017790	-0.001048
	(0.06160)	(0.00269)
	[0.28878]	[-0.38920]

Figure 6.7: Output of the vector error correction model

when there exists more than one cointegrating relationship.

6.2.6 Tests for Cointegration: The Johansen's Approach

An alternative approach to test for cointegration was introduced by Johansen (1988). His approach allows to avoid some drawbacks existing in the Engle-Granger's approach and test the number of cointegrating relations directly. The method is based on the VAR model estimation.

Consider the $VAR(p)$ model for the $k \times 1$ vector $\mathbf{Y}_t$

$$\mathbf{Y}_t = \Pi_1 \mathbf{Y}_{t-1} + ... + \Pi_p \mathbf{Y}_{t-p} + \mathbf{u}_t, \qquad t = 1, ..., T, \tag{6.2.5}$$

where $\mathbf{u}_t \sim IN(0, \Sigma)$.

Since levels of time series $\mathbf{Y}_t$ might be non-stationary, it is better to transform Equation (6.2.5) into a dynamic form, calling vector error correction model (VECM)

$$\Delta \mathbf{Y}_t = \Pi \mathbf{Y}_{t-1} + \Gamma_1 \Delta \mathbf{Y}_{t-1} + ... + \Gamma_{p-1} \Delta \mathbf{Y}_{t-p+1} + \mathbf{u}_t,$$

where $\Pi = \Pi_1 + ... + \Pi_p - I_n$ and $\Gamma_k = -\sum_{j=k+1}^{p} \Pi_j$, $k = 1, ..., p-1$.

Let us assume that $\mathbf{Y}_t$ contains non-stationary $I(1)$ time series components. Then in order to get a stationary error term $\mathbf{u}_t$, $\Pi\mathbf{Y}_{t-1}$ should also be stationary. Therefore, $\Pi\mathbf{Y}_{t-1}$ must contain $r < k$ cointegrating relations. If the $VAR(p)$ process has unit roots then Π has reduced rank $\text{rank}(\Pi) = r < k$. Effectively, testing for cointegration is equivalent to checking out the rank of the matrix Π.

If Π has a full rank then all time series in $\mathbf{Y}$ are stationary, if the rank of Π is zero then there are no cointegrating relationships.

If $0 < \text{rank}\,(\Pi) = r < k$. This implies that $\mathbf{Y}_t$ is $I(1)$ with r linearly independent cointegrating vectors and $k - r$ non-stationary vectors. Since Π has rank r it can be written as the product

$$\underset{(k\times k)}{\Pi} = \underset{(k\times r)}{\alpha}\underset{(r\times k)}{\beta'},$$

where α and β are $k \times r$ matrices with $\text{rank}(\alpha) = \text{rank}(\beta) = r$. The matrix β is a matrix of long-run coefficients and α represents the speed of adjustment to disequilibrium. The VECM model becomes

$$\Delta\mathbf{Y}_t = \alpha\beta'\mathbf{Y}_{t-1} + \Gamma_1\mathbf{Y}_{t-1} + ... + \Gamma_{p-1}\Delta\mathbf{Y}_{t-p+1} + \mathbf{u}_t, \tag{6.2.6}$$

with $\beta'\mathbf{Y}_{t-1} \sim I(0)$.

Johansen's methodology of obtaining estimates of α and β is given below.

Johansen's Methodology

Specify and estimate a $VAR(p)$ model (6.2.5) for $\mathbf{Y}_t$.

Determine the rank of Π; the maximum likelihood estimate for β equals the matrix of eigenvectors corresponding to the r largest eigenvalues of a $k \times k$ residual matrix (see Hamilton (1994), Lutkepohl (1991), Harris (1995) for more detailed description).

Construct likelihood ratio statistics for the number of cointegrating relationships. Let estimated eigenvalues are $\hat{\lambda}_1 > \hat{\lambda}_2 > ... > \hat{\lambda}_k$ of the matrix Π.

Johansen's likelihood ratio statistic tests the nested hypotheses

$$H_0\colon r \leq r_0 \text{ vs. } H_1\colon r > r_0$$

The likelihood ratio statistic, called the *trace statistic*, is given by

$$LR_{trace}(r_0) = -T \sum_{i=r_0+1}^{k} \log\left(1 - \hat{\lambda}_i\right).$$

It checks whether the smallest $k - r_0$ eigenvalues are statistically different from zero. If rank $(\Pi) = r_0$ then $\hat{\lambda}_{r_0+1}, ..., \hat{\lambda}_k$ should all be close to zero and $LR_{trace}(r_0)$ should be small. In contrast, if rank $(\Pi) > r_0$ then some of $\hat{\lambda}_{r_0+1}, ..., \hat{\lambda}_k$ will be nonzero (but less than 1) and $LR_{trace}(r_0)$ should be large.

We can also test $H_0\colon r = r_0$ against $H_1\colon r_0 = r_0 + 1$ using so called the *maximum eigenvalue statistic*

$$LR_{max}(r_0) = -T \log\left(1 - \hat{\lambda}_{r_0+1}\right).$$

Critical values for the asymptotic distribution of $LR_{trace}(r_0)$ and $LR_{max}(r_0)$ statistics are tabulated in Osterwald-Lenum (1992) for $k - r_0 = 1, ..., 10$.

In order to determine the number of cointegrating vectors, first test $H_0\colon r_0 = 0$ against the alternative $H_1\colon r_0 > 0$. If this null is not rejected then it is concluded that there are no cointegrating vectors among the k variables in $\mathbf{Y}_t$. If $H_0\colon r_0 = 0$ is rejected then there is at least one cointegrating vector. In this case we should test $H_0\colon r_0 \leq 1$ against $H_1\colon r_0 > 1$. If this null is not rejected then we say that there is only one cointegrating vector. If the null is rejected then there are at least two cointegrating vectors. We test $H_0\colon r_0 \leq 2$ and so on until the null hypothesis is not rejected.

In a small samples tests are biased if asymptotic critical values are used without a correction. Reinsel and Ahn (1992) and Reimars (1992) suggested small samples bias correction by multiplying the test statistics with $T - kp$ instead of T in the construction of the likelihood ratio tests.

6.2.7 Example in EViews: Johansen's Approach

A very good example of a model with several cointegrating equations has been given by Johansen and Juselius (1990) (1992) (see also Harris (1995)). They considered a single equation approach to combine both Purchasing Power Parity and Uncovered Interest rate Parity condition in one model.

In this model we expect two cointegrating equations between the UK consumer price index P, the US consumer price index P^*, USD/GBP exchange rate S and two interest rates I and I^* in the domestic and foreign countries respectively. If we denote their log counterparts by the corresponding small letter, the theory suggest that the following two relationships should hold in efficient markets with rational investors: $p_t - p_t^* = s_t$ and $\Delta s_{t+1} = i_t - i_t^*$. The data is considered within the range from January 1989 to November 2008 is given in PPPFP1.wf1 file.

We create the log counterparts of the variables in the standard ways, like

```
series lcpi_uk=log(cpi_uk)
```

and so on. In order to check for cointegration we can either estimate VECM (open 5 series as VAR model) or create a Group with the variables. Johansen and Juselius (1990) included into the model seasonal dummy variables as well as crude oil prices. We restrict ourself with only seasonal dummy for simplicity. We can create dummy variables by using a command @expand, which allows to create a group of dummy variables by expanding out one or more series into individual categories. For this purposes we need first to create a variable indicating the quarter of the observation. We do it in the following way

```
series quarter=@quarter(cpi_uk)
```

The command @quarter returns the quarter of the year in which the current observation begins. The second step is to create the dummy variables:

```
group dum=@expand(quarter)
```

EViews will create a new group object dum containing four dummy variables for each of the quarter of the observation.

In both cases, either with VAR or with group objects, one can perform Johansen's test procedure by clicking on **View/Cointegration Test...**.

The dialog window will ask offer to specify the form of the VECM and the cointegrating equation (with or without intercept or trend components). We choose the first option with no trend and intercept to avoid perfect collinearity since we include four dummy variables as exogenous in the model. In the box **Exogenous Variables** enter the name of the dummy variables group dum.

In the box **Lag Intervals for D(Endogenous)** we set 1 4 – we include 4 lags

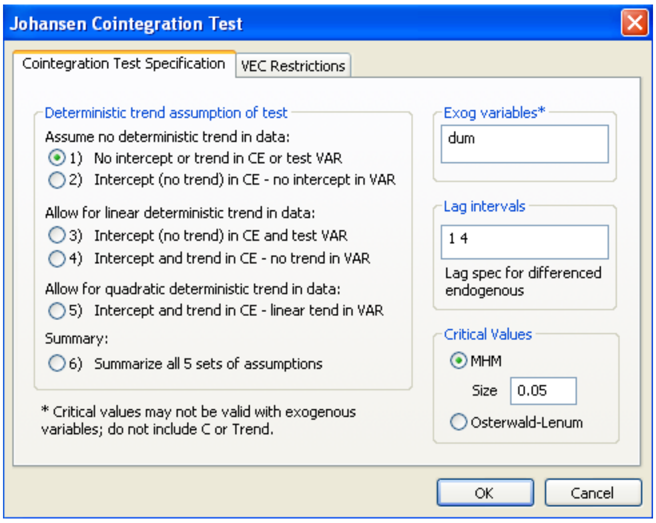


Figure 6.8: Johansen's Cointegration test dialog window

in the model. This is determined by EViews as optimal according to 3 criteria (first estimate VAR with any of the lag specifications, check the optimality of the lag order in **View/Lag Structure/Lag Specification/Lag Length Criteria** and then re-estimate the VECM with the optimal lag order).

Var: UNTITLED Workfile: PPPFP1::Pppfp1\

View Proc Object Print Name Freeze Estimate Stats Impulse Resids

Johansen Cointegration Test

Sample (adjusted): 1989M06 2007M12
Included observations: 223 after adjustments
Trend assumption: No deterministic trend
Series: LIR_UK LIR_US LSPOT LCPI_UK LCPI_US
Exogenous series: DUM
Warning: Critical values assume no exogenous series
Lags interval (in first differences): 1 to 4

Unrestricted Cointegration Rank Test (Trace)

Hypothesized No. of CE(s)	Eigenvalue	Trace Statistic	0.05 Critical Value	Prob.**
None *	0.131576	62.80651	60.06141	0.0288
At most 1	0.075536	31.34664	40.17493	0.2881
At most 2	0.027939	13.83190	24.27596	0.5512
At most 3	0.022044	7.512787	12.32090	0.2770
At most 4	0.011334	2.542019	4.129906	0.1310

Trace test indicates 1 cointegrating eqn(s) at the 0.05 level
* denotes rejection of the hypothesis at the 0.05 level
**MacKinnon-Haug-Michelis (1999) p-values

Unrestricted Cointegration Rank Test (Maximum Eigenvalue)

Hypothesized No. of CE(s)	Eigenvalue	Max-Eigen Statistic	0.05 Critical Value	Prob.**
None *	0.131576	31.45988	30.43961	0.0372
At most 1	0.075536	17.51473	24.15921	0.3056
At most 2	0.027939	6.319117	17.79730	0.8721
At most 3	0.022044	4.970768	11.22480	0.4814
At most 4	0.011334	2.542019	4.129906	0.1310

Max-eigenvalue test indicates 1 cointegrating eqn(s) at the 0.05 level
* denotes rejection of the hypothesis at the 0.05 level
**MacKinnon-Haug-Michelis (1999) p-values

Unrestricted Cointegrating Coefficients (normalized by b'*S11*b=I):

LIR_UK	LIR_US	LSPOT	LCPI_UK	LCPI_US
2.232559	0.525994	6.452675	-2.917368	2.559393
5.211599	-1.166817	6.414865	-1.468834	-2.740902
3.216709	-1.075969	-5.465533	0.241471	0.724072
-1.402045	3.158308	-2.904258	0.444976	-1.929946
2.594429	-1.478534	-8.945306	-1.222705	0.152110

Unrestricted Adjustment Coefficients (alpha):

D(LIR_UK)	-0.001125	-0.004051	-0.002644	0.001682
D(LIR_US)	-0.005073	-0.000671	-0.003628	-0.004836
D(LSPOT)	-0.005088	-0.002292	0.001254	0.000930
D(LCPI_UK)	0.031334	0.001275	-0.005798	-0.004144
D(LCPI_US)	-0.007115	0.024209	-0.006007	0.003939

Figure 6.9: Output for Johansen's Cointegration test

EViews produces results for various hypothesis tested, from no cointegration ($r = 0$) to to increasing number of cointegrating vectors (see Figure **??**). The eigenvalues of matrix $\hat{\Pi}$ is given in the second column. In the third column λ_{trace} statistic is higher than the corresponding critical value at 5% significance for the first hypothesis. This means that we reject the null hypothesis of no cointegration.

However, we cannot reject the hypothesis that there is at most one cointegrating equation. On the basis of λ_{max} statistics (the second panel) it is also possible to accept that there is only one cointegrating relationship. The following two panels provide estimates of matrices β and α respectively.

Note the warning on the top of the output window that saying that critical values assume no exogenous series. This means that we have to take into account that the critical values we are using might not be fully correct as we included exogenous dummy variables in the model. This may give as an explanation why we detected only one cointegrating equation instead of two which were expected. Another reason may be that the second relation based on the UIP condition involves changes of exchange rate rather than levels considered in the VAR model.

Bibliography

Bollerslev, T.: 1986, Generilized autoregressive conditional heterpscedasticity, *Journal of Econometrics* **31**, 307–327.

Bollerslev, T., Engle, R. and Nelson, D.: 1994, *ARCH Models*, Vol. IV, Elsvier Science, chapter Handbook in Econometrics, pp. 2961–3038.

Cuthbertson, K. and Nitzsche, D.: 2004, *Quantitative Financial Economics: Stocks, Bonds and Foreign Exchange*, Jahn Wiley and Sons.

Dickey, D. and Fuller, W.: 1979, Distribution of the estimators for autoregressive time series with a unit root, *Journal of the American Statistical Association* **74**, 427–431.

Dickey, D. and Fuller, W.: 1981, Likelihood ratio statistics for autoregressive time series with a unit root, *Econometrica* **49**, 1057–1072.

Ding, Z., C. W. J. G. and (1993), R. F. E.: 1993, A long memory property of stock market returns and a new model, *Journal of Empirical Finance* **1**, 83–106.

Durbin, J. and Watson, G.: 1950, Testing for serial correlation in least squares regression – I, *Biometrica* **37**, 409–428.

Enders, W.: 2004, *Applied Econometric Time Series*, John Wiley and Sons.

Engle, R.: 1982, Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation, *Econometrica* **50**, 987–1007.

Engle, R. and Granger, C.: 1987, Cointegration and error correction: Representation, estimation and testing, *Econometrics* **55**, 251–276.

Engle, R. and Lee, G.: 1999, *A Long-Run and Short-Run Component Model of Stock Return Volatility*, Cointegration, Causality, and Forecasting, Oxford University Press.

Engle, R., Lilien, D. and Robins, R.: 1987, Estimating time varying premia in the term structure: The ARCH-M model, *Econmetrica* **55**, 591–407.

Fama, E. and French, K.: 1993, Common risk factors in the returns on stocks and bonds, *Journal of Financial Economics* **33**, 3–56.

Fuller, W.: 1996, *Introduction to Statistical Time Series*, John Wiley and Sons, New-York.

Granger, C.: 1969, Investigating causal relations by econometric models and cross spectral methods, *Econometric* **37**, 424–438.

Greene, W.: 2000, *Econometric Analysis*, 5th edition edn, Prentice Hall, New Jersey.

Hamilton, J.: 1994, *Time Series Analysis*, Princeton University Press, New Jersey.

Harris, R.: 1995, *Using Cointegration Analysis in Econometric Modelling*, Prentice Hall, London.

Hayashi, F.: 2000, *Econometrics*, Prinseton University Press.

Johansen, S. and Juselius, K.: 1990, Maximum likelihood estimation and inference on cointegration with application to the demand for money, *Oxford Bulletin of Economics and Statistics* **52**, 169–209.

Kwiatkowski, D., Phillips, P., Schmidt, P. and Shin, Y.: 1992, Testing the null hypothesis of stationarity against the alternative of a unit root, *Journal of Econometrics* **54**, 159–178.

Lutkepohl, H.: 1991, *Introduction to Multiple Time Series Analysis*, Springer, Berlin.

MacKinnon, J.: 1996, Numerical distribution functions for unit root and cointegration tests, *Journal of Applied Econometrics* **11**, 601–618.

Mills, T.: 1999, *The Econometrics Modelling of Financial Time Series*, Cambridge University Press, Cambridge.

Nelson, D.: 1991, Conditional heteroskedasticity in asset returns: a new approach, *Econometrica* **59**(2), 347–370.

Osterwald-Lenum, M.: 1992, A note with quantiles of the asymptotic distribution of the maximum likelihood cointegration rank statistics, *Oxford Bulletin of Economics and Statistics* **54**, 461–472.

Phillips, P. and Ouliaris, S.: 1990, Asymptotic properties of residual based tests for cointegration, *Econometrica* **58**, 73–93.

Phillips, P. and Perron, P.: 1988, Testing for a unit root in time series regression, *Biometrica* **75**, 335–346.

Reimars, H.-E.: 1992, Comparisons of tests formultivariate cointegration, *Statistical Papers* **33**, 335–359.

Reinsel, G. and Ahn, S.: 1992, Vector autoregression models with unit roots and reduced rank structure: Estimation, likelihood ratio test, and forecasting, *Journal of Time Series Analysis* **13**, 353–375.

Said, S. and Dickey, D.: 1984, Testing for unit roots in autoregressive moving average models of unknown orders, *Biometrica* **71**, 599–607.

Savin, N. and White, K.: 1977, The durbin-watson test for serial correlation with extreme sample sizes or many regressors, *Econometrica* **45**, 1989–1996.

Tsay, R.: 2002, *Analysis of Financial Time Series*, John Wiley and Sons.

Verbeek, M.: 2008, *A Guide to Modern Econometrics*, John Wiley and Sons.

Zivot, E. and Wang, J.: 2006, *Modeling Financial Time Series with S-PLUS*, Springer.