G@RCH 1.1 : AN OX PACKAGE FOR
ESTIMATING VARIOUS ARCH MODELS

Sébastien LAURENT & Jean-Philippe PETERS

Octobre 2000

Working Paper
G@RCH 1.1 : An Ox Package for Estimating Various ARCH Models

BY SEBASTIEN LAURENT 1 AND JEAN-PHILIPPE PETERS 2

Faculty of Economy, Business and Social Science, University of Liège, Belgium

October 2000

I. INTRODUCTION .................................................................................................... 3

1. THE G@RCH PACKAGE ......................................................................................... 3
   1.1. Definition ......................................................................................................... 3
   1.2. Program Versions ............................................................................................ 3
2. DISCLAIMER ............................................................................................................ 4
3. AVAILABILITY AND CITATION ................................................................................. 4
4. STRUCTURE OF THE PROGRAM .............................................................................. 4
   4.1. Classes and Functions ..................................................................................... 4
   4.2. GARCH Member Functions List...................................................................... 5

II. G@RCH MEMBERS FUNCTIONS .................................................................... 8

III. FEATURES OF THE PACKAGE .................................................................... 17

1. INTRODUCTION...................................................................................................... 17
2. MODELS OF THE PROGRAM ................................................................................. 17
   2.1 Mean equation ................................................................................................ 17
   2.2 Variance equation ........................................................................................... 18
   2.3 ARCH Model................................................................................................... 18
   2.4 GARCH Model ................................................................................................ 19
   2.5 EGARCH Model.............................................................................................. 20
   2.6 GJR Model ...................................................................................................... 20
   2.7 APARCH Model .............................................................................................. 21
   2.8 Integrated Models ........................................................................................... 21

1 http://www.egss.ulg.ac.be/econometrie/SLaurent.htm
2 http://www.eaa.egss.ulg.ac.be/rogp/peters/
2.9 Fractionally Integrated Models ................................................................. 22

3. Estimation Methods ....................................................................................... 24
   3.1 The Distributions ....................................................................................... 24
   3.2. The Standard Deviation Estimation Methods ........................................ 26
   3.3 Tests ......................................................................................................... 26

IV. USING THE PROGRAM ................................................................................. 28

   1. Installing the Files ...................................................................................... 28
   2. Running the "Full Version" ........................................................................ 28
   3. Running the "Light Version" ..................................................................... 39

V. VERSIONS AND FUTURE IMPROVEMENTS .............................................. 42

   1. Releases History ......................................................................................... 42
   2. Future Improvements ............................................................................... 42

VI. REFERENCES ................................................................................................. 42
I. Introduction

1. The G@RCH Package

1.1. Definition

G@RCH 1.1 is an Ox package dedicated to the estimation of GARCH models and many of its extensions. It can be used via OxPack (with a dialog-oriented interface) or via the traditional (programming) way.

The available models are ARCH (Engle, 1982), GARCH (Bollerslev, 1986), EGARCH (Nelson, 1991), GJR (Glosten et al., 1993), APARCH (Ding et al., 1993), IGARCH (Engle and Bollerslev, 1986), FIGARCH (Baillie et al., 1996 and Chung, 1999), FIEGARCH (Bollerslev and Mikkelsen, 1996) and FIAPARCH (Tse, 1998). These models can be estimated by Approximate (Quasi-) Maximum Likelihood under three assumptions: Normal, Student-t or GED errors. Finally, explanatory variables can enter both the mean and the variance equations.

1.2. Program Versions

Two versions of our program are available called the "Light Version" and the "Full Version".

The "Light Version" is launched from the Ox file GarchEstim.ox. This version requires some experience with the program and its structure. The "Light Version" is therefore dedicated to advanced users. This is also dedicated to users who do not possess the GiveWin software. Indeed, since the OxPack module is only available for registered Ox users, users who have a free Ox version cannot use our dialogs-oriented (or "Full") version. Hence, the "Light Version" is a solution, since its use just requires an Ox executable and a text editor.

The "Full Version" provides the same features as the other version, but also a friendly interface and some graphical features. This package needs to be launched from OxPack.
This tutorial is structured as follows: in Section II, we introduce the G@RCH class members (that is, the procedures composing the class). In Section III, we propose an overview of the package's features, with the presentation of the models and the estimation methods (distributions, standard deviation estimation and testing procedures). Then, a user guide is provided for both versions of our package in Section IV. Finally, we underline the future improvements of the G@RCH package in Section V.

2. Disclaimer
This package is functional but no warranty is given whatsoever. The most appropriate way to discuss about problems and issues of the G@RCH package is the Ox-users forum (see http://www.mailbase.ac.uk/lists/ox-users for registration and archives). Suggestions can be reported to the authors via e-mail: S.Laurent@ulg.ac.be for Sébastien and jp.peters@ulg.ac.be for Jean-Philippe.

3. Availability and Citation
The G@RCH package is available for downloading at the following address:

http://www.egss.ulg.ac.be/garch

For easier validation and replication of empirical findings, please cite this documentation in all reports and publications involving the use of this G@RCH 1.1 package.

4. Structure of the Program

4.1. Classes and Functions
Ox provides support for object-oriented programming. An interesting concept is therefore the "Classes". Indeed, "complex data structures can be encapsulated within instances of a class that only can be accessed by class member functions".3 Furthermore, derived classes such as our GARCH class can be constructed.

In other words, one can create new classes based on other existing parent-classes and use the functions of these parents, avoiding to be obliged of rewriting procedures for derived classes. In our case, the GARCH class is defined as a Modelbase type of class. This Modelbase class derives itself from the Database class (Figure 3.1).

The **Database** class is dedicated to the handling of the database, the sample, the names of the variables, the selection of variables… The **Modelbase** implement model estimation features. It is not intended to be used directly but as a base for a more specialized class, such as our **GARCH** class or already available classes such as ARFIMA, DPD (Panel Data estimation), SVPack (Stochastic Volatility models) or SsfPack (State space forms).

![Figure 3.1. Classes Structure](image)

### 4.2. GARCH Member Functions List

Here is the list of the Garch member functions and a brief description for each of them. Our program also uses functions from other classes (Modelbase and Database).

**Constructor**

Garch Constructor

**Model Formulation (only "Light Version")**

- **APARCH**: Specifies if APARCH is wanted in the variance
- **ARFIMA**: Specifies if ARFIMA is wanted in the mean
- **ARMA**: Specifies the AR and MA orders in the mean
- **CSTS**: Specifies if constants are wanted in the mean and in the variance
- **DISTRI**: Specifies the desired distribution.
- **EGARCH**: Specifies if EGARCH is wanted in the variance
- **FIGARCH**: Specifies if FIGARCH is wanted and selects the estimation method (BBM; i.e. Baillie et al., 1996, or Chung, 1999)
- **GARCH**: Specifies the p and q orders of the GARCH(p,q)
- **GJR**: Specifies if GJR is wanted in the variance
- **IGARCH**: Specifies if IGARCH model is wanted in the variance
- **LAGS**: Specifies the desired lags for the Box-Pierce tests
- **MLE**: Specifies the standard errors estimation method
- **STORE**: Allows storing estimated \( \varepsilon_t \), \( \varepsilon_t^2 \) and \( \sigma_t^2 \) series
- **TESTSONLY**: Allows running only tests, prior to any estimation.
Input
InitData Initializes the characteristics of the model (sample, regressors…)
InitStartValues Initializes the starting values of the parameters to estimate

Parameters related functions
Dialogs Parameter starting values (only "Full Version")
Filter Allocates the filter number depending on the model specification
GetPara Constructs the parameters vector
GetRes Gets residuals from the mean equation
SplitPara Allocates the value of each element of the parameters vector to the correct variable

Filters
AParch APARCH filter
EGarch EGARCH filter
Garch GARCH Filter
Garch_Leverage GJR filter
GaussLik Computes the log-likelihood for the Gaussian distribution
GEDLik Computes the log-likelihood for the GED
FIAParch FIAPARCH filter
FIEGarch FIEGARCH filter (the Fractional Integration process uses Chung's method)
Figarch_BBM FIGARCH filter with the Baillie et al. (1996) method (BBM)
Figarch_Chung FIGARCH filter with the Chung (1999) method
FigLL Function to be maximized by BFGS
FiltARMA ARMA filter
StudentLik Computes the log-likelihood for the Student distribution

General
GetParName Gets parameters names
GetXNames Gets the names of the regressors in the mean equation
GetZNames Gets the names of the regressors in the variance equation

Model Estimation
DoEstimation Estimates the model ("Light version")
Estimate Estimates the model ("Full version")
ScoreContributions Computes the numerical derivatives.

Post Estimation
ArchTest Computes the Engle's LM ARCH test
BoxPQ Computes the modified Box-Pierce Q-statistics and the associated p-values
ICriterion Computes the four Information Criteria (Akaike, Hannan-Quinn, Schwarz and Shibata)
Normality Computes the skewness, kurtosis and Jarque-Bera test, with associated t-test and p-values
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLEMeth</td>
<td>Prints the estimated parameters, their standard deviations, t-tests and p-values</td>
</tr>
<tr>
<td>Output</td>
<td>Prints the model specification and launches other post-estimation procedures</td>
</tr>
<tr>
<td>SBT</td>
<td>Computes the sign bias test, the negative size bias test, the positive size bias test and a joint test of the three</td>
</tr>
<tr>
<td>Tests</td>
<td>Computes and prints the tests</td>
</tr>
<tr>
<td>TestGraphicAnalysis</td>
<td>Prints the graphics</td>
</tr>
</tbody>
</table>
II. G@RCH Members Functions

Garch::APARCH, Garch::ARFIMA, Garch::CSTS, Garch::EGARCH, Garch::GJR, Garch::INTEGRATE

APARCH(const cAPARCH) ;
ARFIMA(const cARFI) ;
CSTS(const cstM, const cstV) ;
EGARCH(const cEGARCH) ;
IGARCH(const integ) ;
GJR(const cLeverage) ;
cAparch in: 1 or 0, APARCH selection
cArfi in: 1 or 0, ARFIMA selection
cstM in: 1 or 0, constant in the mean equation selection
cstV in: 1 or 0, constant in the variance equation selection
cEgarch in: 1 or 0, EGARCH selection
Integ in: 1 or 0, IGARCH selection
cLeverage in: 1 or 0, GJR selection

No return value

Description
When the argument is 1, the concerned specification is selected. GJR, IGARCH, APARCH, EGARCH and GJR are exclusive selection.

Garch::ARMA, Garch::GARCH,

ARMA(const cAR, const cMA) ;
GARCH(const cP, const cQ) ;
cAR in: integer, AR order, p
cMA in: integer, MA order, q
cP in: integer, GARCH order, p
cQ in: integer, ARCH order, q

No return value

Description
Fixes the ARMA and GARCH orders.

Garch::AParch, Garch::EGarch, Garch::Garch, Garch::GJR_Filt, Garch::Figarch_BBM, Garch::Figarch_Chung, Garch::FIEGarch, Garch::FIAParch,

AParch (const e, const level, const p, const q) ;
EGarch (const e, const level, const p, const q) ;
Garch (const e, const level, const p, const q) ;
GJR_Filt (const e, const level, const p, const q) ;
Figarch_BBMM (const e, const level, const p, const q, const laglamb) ;
Figarch_Chung (const e, const level, const p, const q);
FIEGarch (const e, const level, const p, const q) ;
FIAParch (const e, const level, const p, const q);

\begin{itemize}
\item e \text{ in: (m_cT x 1) matrix, residuals series.}
\item level \text{ in: (m_cT x 1) matrix, independent variables.}
\item p \text{ in: integer, GARCH order, p}
\item q \text{ in: integer, ARCH order, q}
\item laglamb \text{ in: integer, truncation order (BBM method)}
\end{itemize}

\textbf{Return value}

A \((m_cT \times 1)\) matrix with the estimated conditional variance \((\sigma^2)\).

\textbf{Description}

These are the filters of the different models. Recall that two methods are available for the FIGARCH models: the Baillie \textit{et al.} (1996) method (BBM) that includes a truncation order or the Chung (1999) method that do not. See FIGARCH(const d, const method, const trunc) for further information. Note that "level" is a \((m_cT \times 1)\) ones vector if there is no independent variable and a \((m_cT \times m_cX)\) matrix equal to \(b'X\) if there are explanatory variables.

\textbf{Garch::Dialogs}

\textbf{Dialogs () ;}

\textbf{No return value}

\textbf{Description}

Only used with OxPack. Launches dialogs related to the starting values. There are two possibilities: the "Parameter-by-parameter" dialog or the "Vector" dialog. The former is launched if the user has previously selected the "Manually (Individual Form)" option in OxPack while the latter is available when having chosen "Manually (Vector Form)".

\textbf{Garch::DISTRI}

\textbf{DISTRI(const dist) ;}

\begin{itemize}
\item dist \text{ in: integer, distribution selection}
\end{itemize}

\textbf{No return value}
Description
Selection of the distribution. If \( dist \) is 0, this is the Gaussian distribution (Normal), if it is 1, it is the Student-\( t \) distribution and if it is 2, it is the Generalized Error distribution (GED).

**Garch::DoEstimation, Garch::Estimate**

DoEstimation(const vStart) ;

Estimate () :

\[ vStart \text{ in: } (m_cParam x 1) \text{ matrix, starting values of the parameters to be estimated.} \]

**Return value**

1 if model successfully estimated, 0 if it failed.

Description
DoEstimation is launched from the “Light version” of the program while Estimate is launched from the “Full version”. These procedures are the core procedures of the program. They successively launch others procedures to initialize parameters, to estimate the formulated model, to print the results, to run the tests or to print the graphics.

**Garch::FIGARCH**

FIGARCH(const d, const method, const trunc) ;

\[ d \text{ in: 1 or 0, FIGARCH selection} \]

\[ \text{method in: 1 or 0, estimation method selection} \]

\[ \text{trunc in: integer, truncation order (only if method = 1)} \]

**No return value**

Description
It is related to the fractionally integrated (FI) model selection. If \( d = 1 \), a F.I. model will be estimated. If \( method = 0 \), the estimation method will follow Chung’s (1999) specification. If \( method = 1 \), the estimation method will follow BBM’s (1996) specification. One notable difference (among others) between these two method is that BBM uses a fix number of lags equal to \( trunc \) in order to compute the binomial expansion (Taylor’s theorem) while Chung proposes an increasing number of lags (in fact, it includes all previous observations). To estimate a FIEGARCH model for instance, you have to select both EGARCH and FIGARCH ⇒ EGARCH(1), and FIGARCH(1, method, trunc).

**Garch::FigLL**

FigLL(const vP, const adFunc, const avScore, const amHessian);

\[ \text{vP in: } (m_cP x 1) \text{ matrix, parameters to be estimated} \]

\[ \text{out: estimated parameters} \]
adFunc in: address
out: double, log-likelihood function value at vP
avScore in: 0, or an address
out: if not 0 on input, (m_cP x 1) matrix with first derivatives at vP
amHessian in: 0, as MaxBFGS does not require the Hessian.

Return value
1 if successful, 0 if evaluation failed.

Description
This is the procedure optimized by Ox with the MaxBFGS function. This function uses the Broyden, Fletcher, Goldfarb and Shanno (BFGS) quasi-Newton method (see Doornik, 1999, p.240, for further details).

Garch::Filter

Filter() ;

Return value
1 if successful, 0 if failure.

Description
Allocates the correct filter depending on the specification made to the selection procedures (GARCH, FIGARCH, EGARCH, GJR, APARCH, INTEGRATE).

Garch::Garch

Garch() ;

No return value

Description
Constructor.

Garch::GaussLik, Garch:: GEDLik, Garch::StudentLik

GaussLik(const vE, const vSigma2) ;
GEDLik(const vE, const vSigma2, const a) ;
StudentLik(const vE, const vSigma2, const v) ;
vE in: (m_cT x 1) matrix, residuals.
vSigma2 in: (m_cT x 1) matrix, conditional variance.
a in: double, asymmetric coefficient.
v in: double, degree of freedom.

Return value
The log-likelihood function value associated with vE and vSigma2.
**Description**
Computes the log-likelihood functions of the three available distributions.

**Garch:: GetPara;**

GetPara();

*Return value*
1 if successful, 0 if failure.

**Description**
It constructs the parameters vector and allocates it to `m_vPar`. The order of the parameters is the following: constant in mean, regressors in mean coefficients, \( d \) coefficient, AR coefficients, MA coefficients, constant in var., regressors in var. coefficients, F.I.(\( d \)) coefficient, GARCH coefficients, ARCH coefficients, GJR coefficients, EGARCH coefficients, APARCH coefficients, GED degrees of freedom and Student degrees of freedom.

**Garch::GetParNames , Garch::GetXNames, Garch::GetZNames;**

GetParNames();
GetXNames();
GetZNames();

*Return value*
Array of strings with the names of the variables or parameters.

**Description**
These procedures collect the names of the regressors in the mean equation (\( X \)) or in the variance equation (\( Z \)) or the names of the estimated parameters.

**Garch::GetRes**

GetRes(const y, const x);

\[ y \] in: \((m_cT \times 1)\) matrix, dependent variable
\[ x \] in: \((m_cT \times m_cX)\) matrix, regressors

*Return value*
\((m_cT \times 1)\) matrix containing the residuals.

**Description**
Computes the residuals of the mean equation without taking account of AR(FI)MA processes.

**Garch::InitData**
InitData() ;

Return value
1 if successful, 0 if failure.

Description
Allocates the Y series and the regressors to class members, computes the no. of observations of the sample, the no. of parameters to be estimated.

Garch::InitStartValues

InitStartValues() ;

No return value

Description
Initializes the starting values when the user do not enter any specific starting values.
These values are:
- Constant in the mean: 0.05
- Regressors in the mean: 0.01
- ARFIMA(p,d,q): $p_1 = 0.2$, $p_{s1} = 0.05$, $d = 0.1$, $q_1 = 0.15$, $q_{s1} = 0.02$
- Constant term in the variance equation: 0.01
- FIGARCH(p,d,q): $\beta_1 = 0.7$ (if GARCH) or 0.45 (if FIGARCH), $\beta_{s1} = 0.1$, $d = 0.5$
  and $\alpha = 0.1$ (for all q)
- GJR: leverage coefficients = 0.01
- EGARCH: $\phi_1 = -0.1$ and $\phi_2 = 0.2$
- APARCH: $\delta = 1.2$, $\gamma_1 = 0.15$, $\gamma_{s1} = 0.05$
- Student distribution: $\nu = 6$ (degrees of freedom).
- GED distribution: $\nu = 2$.

Garch::Integrate

Integrate(const par) ;
  par in: (m_cPar x 1) matrix, parameters.
  out: modified parameters ("sum equals to one")

Return value
1 if successful, 0 if failure.

Description
Integrated GARCH.

Garch::LAGS

LAGS(const lags) ;
No return value

Description
Fix the lags wanted when computing Box-Pierce statistics. By default, lags is 5;10;20. This means that BP(5), BP(10) and BP(20) are computed for the standardized residuals and squared standardized residuals.

Garch::MLE

MLE(const method);
method in: integer, method selection.

No return value

Description
Selection of the estimation method. If method = 0, both (Approximate) Maximum Likelihood and Quasi-Maximum Likelihood Estimates will be computed. If method = 1, only MLE is selected and if it is equal to 2, only QMLE are computed.

Garch::MLEMeth

MLEMeth(const par, const parnames, const title);
par in: (m_cPar x 1) matrix, estimated parameters
parnames in: array of m_cPar strings, estimated parameters names
title in: string, selected method name

No return value

Description
Prints the estimated parameters, their standard deviations, t-tests and p-values with their names. Depending on the user's choice, ML estimates, QML estimates or both will be printed.

Garch::Output

Output();

No return value

Description
Prints the specification of the formulated model and launches the standard errors computations.

Garch::SplitPara
SplitPara(vP) ;
    vP  in: \((m_{cP} \times 1)\) matrix, parameters vector.

Return value
1 if successful, 0 if failure.

Description
Splits the parameters vector and allocates each one to the correct variable (or class member).

Garch::STORE

STORE(const res, const res2, const condv, const name, const file) ;
    res      in: 1 or 0, to store the residuals or not.
    res2     in: 1 or 0, to store the squared residuals or not.
    condv    in: 1 or 0, to store the cond. Variance or not.
    name     in: string, suffix added to "Res_", "RsqRes_" or "RcondV_" to name the saved series.
    file     in: 1 or 0; to save in a new file or in the used database.

No return value

Description
Stores the residuals, their squared and the conditional variance of the estimated models. Argument 4 provides a default suffix ("01") that can be modified. If argument 5 equals 1, the series will be stored in a new .in7 file. If it is equal to 0, the saved series will be appended in the used database.

Garch::Tests

Tests() ;

No return value

Description
Runs the selected tests.

Garch::TestGraphicAnalysis

TestGraphicAnalysis(const ser, const res, const sqres, const h) ;
    ser     in: 1 or 0; 1 if raw series plot wanted.
    res     in: 1 or 0; 1 if residuals plot wanted.
    sqres   in: 1 or 0; 1 if squared residuals plot wanted.
    h       in: 1 or 0; 1 if cond. variance plot wanted.

No return value
Description
Displays graphics of the series and/or the residuals and/or the squared residuals and/or the conditional variance in the GiveWin front-end.

Garch::TESTONLY

TESTONLY(const t);
  t in: 0 or 1.

No return value

Description
Allows to run the test for the raw series, prior to any estimation.

OxPack Functions

OxPack related functions are described in Doornik (1999).
III. Features of the package

1. Introduction

There are nine models provided with the program:
- the ARCH and GARCH models, certainly the most common ones.
- the IGARCH and FIGARCH models, which allow the integrated and the fractional integrated extensions of the GARCH model.
- three asymmetric models: GJR, EGARCH and APARCH.
- and the fractionally integrated extensions of the last two models (FIEGARCH and FIAPARCH).

All these models can be estimated under three different distributions: the Gaussian, the Student-\(t\) and the Generalized Error (GED) distributions. Moreover, explanatory variables can be added both in the mean and in the variance equations.

2. Models of the Program

2.1 Mean equation

Let us consider a discrete time series \(\{y_t\}\). If \(\psi_{t-1}\) is the information set (i.e. all the information) available at time \(t-1\), we can define its functional form as

\[
y_t = g(\psi_{t-1}, b) + \varepsilon_t
\]

(2.1)

where \(g(\cdot)\) is a function, \(b\) is a vector of the parameters to be estimated and \(\varepsilon_t\) is the disturbance term.

This equation is the mean equation and it has been studied and modeled in hundreds of different ways. Two of the most famous specifications are the Auto Regressive (AR) and Moving Average (MA) models. Mixing these two processes and introducing \(k\) explanatory variables in the equation, we obtain this ARMA\((m,l)\) process,

\[
\Phi(L)(y_t - \mu_t) = \Theta(L)\varepsilon_t
\]

\[
\mu_t = \mu + \sum_{i=1}^{k} \delta_i x_{ik}
\]

(2.2)

where \(L\) is the lag operator \((L^r y_t = y_{t-r})\), \(\Phi(L) = 1 - \sum_{i=1}^{m} \phi_i L^i\) and \(\Theta(L) = 1 - \sum_{i=1}^{l} \theta_i L^i\).
Several studies have shown that the dependent variable (interest rate returns, exchange rate returns, etc.) exhibit significant autocorrelation between observations widely separated in time. In such a case, we can say that \( y_t \) displays long memory, or long-term dependence and is best modelled by a fractionally integrated ARMA process (so called ARFIMA process) initially developed by Granger (1980), Granger and Joyeux (1980) among others. Such a model is described as follows:

\[
\Phi(L)(1-L)^d(y_t - \mu_t) = \Theta(L)\epsilon_t
\]  

(2.3)

### 2.2 Variance equation

The \( \epsilon_t \) term in equations (2.1)- (2.3) is the innovation of the process. The conditional expectation is the expectation conditional to all past information available at time \( t-1 \). About twenty years ago, Engle (1982) defined as an Autoregressive Conditional Heteroskedastic (ARCH) process, all \( \{\epsilon_t\} \) of the form

\[
\epsilon_t = z_t \sigma_t
\]  

(2.4)

where \( z_t \) is an independently and identically distributed (i.i.d.) process, \( \mathbb{E}(z_t) = 0 \), \( \mathbb{V}ar(z_t) = 1 \) and where \( \sigma_t \) is a time-varying, positive and measurable function of the information set at time \( t-1 \). By definition, \( \epsilon_t \) is serially uncorrelated with mean zero, but its conditional variance equals to \( \sigma_t^2 \) and, therefore, may change over time, contrary to what is assumed in OLS estimations.

The models provided by our program are all ARCH-type. They differ on the functional form of \( \sigma_t^2 \) but the basic ideas are the same. Besides the traditional ARCH and GARCH models, we focus mainly on two kinds of models: the asymmetric models and the fractionally integrated models. The former are defined to take into account the so-called "leverage effect" observed in many stock returns, while the latter allows for long-memory in the variance.

### 2.3 ARCH Model

The Autoregressive Conditional Heteroskedasticity (ARCH) model is introduced in Engle (1982). Let \( \epsilon_t \) be the residuals of the mean equation and let \( \psi_t \) be the information set a time \( t \). The ARCH \((q)\) model can be expressed as:
\[
\varepsilon_i = z_i \sigma_i \quad \text{with } E(z_i) = 0 \text{ and } \text{Var}(z_i) = 0,
\]
\[
\varepsilon_i \mid \psi_{i-1} \sim N(0, \sigma_i^2),
\]
\[
\sigma_i^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \varepsilon_{i-1}^2 \quad (2.5)
\]

In the rest of this tutorial, \( \alpha_0 \) is assumed fixed. If explanatory variables are introduced in the models, \( \alpha_{0i} = \alpha_0 + \sum_{j=1}^{k} \kappa_j X_{it} \) with an exception for the exponential models (EGARCH and FIEGARCH) where \( \alpha_{0i} = \alpha_0 + \ln \left(1 + \sum_{j=1}^{k} \kappa_j X_{it}\right) \).

### 2.4 GARCH Model

Tim Bollerslev proposes the Generalized ARCH (GARCH) model (Bollerslev, 1986) (based on an infinite ARCH specification). The GARCH \((p,q)\) model can be expressed as:

\[
\sigma_i^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \varepsilon_{i-1}^2 + \sum_{j=1}^{p} \beta_j \sigma_{i-j}^2 \quad (2.6)
\]

where \( p \geq 0, q > 0, \alpha_0 > 0, \alpha_i \geq 0 (i = 1, \ldots, q) \) and \( \beta_j \geq 0 (p = 1, \ldots, p) \).

Using the lag or backshift operator \( L \), the GARCH \((p,q)\) model is

\[
\sigma_i^2 = \alpha_0 + \alpha(L) \varepsilon_i^2 + \beta(L) \sigma_i^2
\]

with \( \alpha(L) = \alpha_0 L + \alpha_1 L^2 + \ldots + \alpha_q L^q \) and \( \beta(L) = \beta_0 + \beta_1 L + \ldots + \beta_p L^p \).

If all the roots of the polynomial \( 1 - \beta(L) = 0 \) lie outside the unit circle, we have

\[
\sigma_i^2 = \alpha_0 (1 - \beta(L))^{-1} + \alpha(L) (1 - \beta(L))^{-1} \varepsilon_i^2
\]

or

\[
\sigma_i^2 = \frac{\alpha_0}{1 - \beta_1 - \ldots - \beta_p} + \sum_{i=1}^{\infty} \lambda_i \varepsilon_{i-1}^2,
\]

which may be seen as an ARCH\((\infty)\) process since the conditional variance linearly depends on all previous squared residuals.
2.5 EGARCH Model

The Exponential GARCH (EGARCH) model is proposed in Nelson (1991) and can be presented as:

\[
\ln \sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i g(z_{t-i}) + \sum_{j=1}^{p} \beta_j \ln(\sigma_{t-j}^2) 
\]

(2.7)

where \( z_i = \frac{\varepsilon_i}{\sigma_i} \) is the normalized residuals series.

The value of \( g(z_t) \) depends on several elements. Nelson (1991) notes that, "to accommodate the asymmetric relation between stock returns and volatility changes (...) the value of \( g(z_t) \) must be a function of both the magnitude and the sign of \( z_t \)."\(^4\)

That is why he suggests to express the function \( g(.) \) as

\[
g(z_t) \equiv \theta \left( \frac{z_t}{\sigma_t} \right) + \theta \left( \left| z_t - E[z_t] \right| \right)
\]

(2.8)

2.6 GJR Model

This popular model is proposed by in Glosten, Jagannathan and Runkle (1993). Its generalized version is given by:

\[
\sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} (\alpha_i \varepsilon_{t-i}^2 + \omega_i S_{t-i} \varepsilon_{t-i}^2) + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2
\]

(2.9)

where \( S_t \) is a dummy variable.

In this model, it is assumed that the impact of \( \varepsilon_t^2 \) on the conditional variance \( \sigma_t^2 \) is different when \( \varepsilon_t \) is positive or negative. That is why the dummy variable \( S_t \) takes the value "0" (respectively "1") when \( \varepsilon_t \) is positive (negative). Note that the TGARCH model of Zakoian (1992) is very similar to the GJR but models the conditional standard deviation instead of the conditional variance.

2.7 APARCH Model

Ding et al. (1993) introduce the Asymmetric Power ARCH (APARCH) model. The APARCH \((p, q)\) model can be expressed as:

\[
\sigma_i^4 = \alpha_0 + \sum_{i=1}^{q} \alpha_i (|\epsilon_{t-i}| - \gamma \epsilon_{t-i})^\delta + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^\delta
\]  

(2.10)

where \(\alpha_0 > 0, \delta \geq 0, \beta_j \geq 0 (j = 1, \ldots, p), \alpha_i \geq 0 \) and \(-1 < \gamma_i < 1 (i = 1, \ldots, q)\).

This model couples the flexibility of a varying exponent with the asymmetry coefficient (to take the "leverage effect" into account). It includes seven other ARCH extensions as special cases:

- ARCH when \(\delta = 2, \gamma = 0 (i = 1, \ldots, p)\) and \(\beta_j = 0 (j = 1, \ldots, p)\).
- GARCH when \(\delta = 2\) and \(\gamma_i = 0 (i = 1, \ldots, p)\).
- Taylor/Schwert's GARCH when \(\delta = 1, \gamma = 0 (i = 1, \ldots, p)\).
- GJR when \(\delta = 2\).
- TARCH when \(\delta = 1\).
- NARCH when \(\gamma = 0 (i = 1, \ldots, p)\) and \(\beta_j = 0 (j = 1, \ldots, p)\).
- The Log-ARCH of Geweke (1986) and Pantula (1986), when \(\delta \to 0.5\).

2.8 Integrated Models

In many high-frequency time-series applications, the conditional variance estimated using a GARCH\((p,q)\) process exhibits persistence, that is

\[
\sum_{j=1}^{p} \beta_j + \sum_{i=1}^{q} \alpha_i = 1.
\]

When this sum is equal to one, we are in presence of an Integrated GARCH (IGARCH) model, meaning that current information remains of importance when forecasting the volatility for all horizons. A similar concept is present in the mean equation: when the sum of all AR coefficients and MA coefficients is equal to one, we face an ARIMA process.

Recall that the GARCH\((p,q)\) model can be expressed as an ARMA\((m,l)\) process. With the lag operator \(L\), we can rearrange Equation (2.6) as

\[
[1 - \alpha(L) - \beta(L)]\epsilon_i^2 = \alpha_0 + [1 - \beta(L)](\epsilon_i^2 - \sigma_i^2)
\]
When the \( \left[ 1 - \alpha(L) - \beta(L) \right] \) polynomial contains a unit root, i.e. the sum of all the \( \alpha_i \) and the \( \beta_j \) is one, we have the IGARCH\((p,q)\) model of Engle and Bollerslev (1986). It can then be written as

\[
\phi(L)(1-L)e_i^2 = \alpha_0 + \left[ 1 - \beta(L) \right] \left( e_i^2 - \sigma_i^2 \right),
\]

where \( \phi(L) = [1 - \alpha(L) - \beta(L)](1-L)^{-1} \) is of order \([\max\{p,q\}-1]\).

We can rearrange Equation (2.11) to express the conditional variance as a function of the squared residuals. After some manipulations, we have

\[
\sigma_i^2 = \alpha_0 + \left[ 1 - \phi(L)(1-L) \left[ 1 - \beta(L) \right]^{-1} \right] e_i^2.
\]

2.9 Fractionally Integrated Models

Volatility tends to change quite slowly, and, as shown in Ding et al. (1993) among others, the effects of shocks take a considerable time to decay.\(^6\) Therefore, the distinction between I(0) and I(1) processes is far too restrictive. Indeed, the propagation of shocks in I(0) processes occurs at an exponential rate of decay (so that it only captures the short-memory), while, in I(1) processes, the persistence of shocks is infinite. Introducing the fractional degree of integration implies a slow hyperbolic rate of decay, thus allowing a shock to be transitory. In the conditional mean, the ARFIMA specification has been proposed so that the short-run behavior of the time-series is captured by the ARMA parameters, while the fractional differencing parameter allows for modeling the long-run dependence.\(^7\)

Similar issues existing in financial time series volatility, Baillie et al. (1996) (hereafter denoted BBM) introduced the Fractionally Integrated GARCH (FIGARCH) model by replacing the first difference operator of Equation (2.12) by \((1-L)^d\).

\(^5\) Complete developments leading to these conclusions are available in Ding et al. (1993).

\(^6\) In their study of the daily S&P500 index, they find that the squared returns series has positive autocorrelations over more than 2,500 lags (or more than 10 years !!).

\(^7\) See Bollerslev and Mikkelsen (1996, p.158) for a discussion on the importance of non-integer values of integration when modeling long-run dependencies in the conditional mean of economic time series.
The conditional variance is therefore given by:

\[
\sigma_t^2 = \alpha_0 \left[ 1 - \beta(L) \right]^{-1} + \left[ 1 - \left[ 1 - \beta(L) \right]^{-1} \phi(L)(1-L)^d \right] \epsilon_t^2
\]  

(2.13)

or \( \sigma_t^2 = \omega + \lambda(L)\epsilon_t^2 \), with \( 0 < d < 1 \).\(^8\)

Chung (1999) underscores some little drawbacks in the BBM model: there is a structural problem in the BBM specification since the parallel with the ARFIMA framework of the mean equation is not perfect, leading to difficult interpretations of the estimated parameters. Indeed the fractional differencing operator of the ARFIMA process applies to the constant term of the mean equation (ARFIMA) while it does not in the variance equation (FIGARCH). Chung proposes a slightly different process:

\[
\phi(L)(1-L)^d \left( \epsilon_t^2 - \sigma^2 \right) = \left[ 1 - \beta(L) \right] \left( \epsilon_t^2 - \sigma_t^2 \right)
\]

(2.14)

where \( \sigma^2 \) is the unconditional variance of \( \epsilon_t \).

If we keep the same definition of \( \lambda(L) \) as in Equation (2.13), we can formulate the conditional variance as

\[
\sigma_t^2 = \sigma^2 + \left[ 1 - \left[ 1 - \beta(L) \right]^{-1} \phi(L)(1-L)^d \right] \left( \epsilon_t^2 - \sigma^2 \right)
\]

(2.15)

or \( \sigma_t^2 = \sigma^2 + \lambda(L) \left( \epsilon_t^2 - \sigma^2 \right) \).

Notice that \( \lambda(L) \) is an infinite summation which, in practice, has to be truncated. BBM propose to truncate \( \lambda(L) \) at 1000 lags (this truncation order may be changed by the user) and initialize the unobserved \( \epsilon_t^2 \) at their unconditional moment. Contrary to BBM, Chung proposes to truncate \( \lambda(L) \) at the size of the information set \((t-1)\) and to initialize the unobserved \( \left( \epsilon_t^2 - \sigma^2 \right) \) at 0 (this quantity is small in absolute values and has a zero average).

The idea of fractional integration has then been applied to other GARCH types of models, including the Fractionally Integrated EGARCH (FIEGARCH) of Bollerslev and Mikkelsen (1996) and the Fractionally Integrated APARCH (FIAPARCH) of Tse (1998).

\(^8\) Note that, in the ARFIMA specification, we have \( 0 < d < 0.5 \)
Similarly to the GARCH\((p,q)\) process, the EGARCH\((p,q)\) of Equation (2.7) can be rearranged as an ARMA process:

\[
\ln(\sigma_t^2) = \omega + \frac{1 + \psi_1 L + \ldots + \psi_q L^q}{1 - \Delta L - \ldots - \Delta_q L^q} g\left(z_{t-1}\right) = \omega + \frac{1 + \psi(L)}{1 - \Delta(L)} g\left(z_{t-1}\right).
\]

Factorizing the autoregressive polynomial \(1 - \Delta(L)\) = \(\phi(L)(1 - L)^d\) where all the roots of \(\phi(z) = 0\) lie outside the unit circle, we can define the FIEGARCH \((p,d,q)\) as

\[
\ln(\sigma_t^2) = \omega + \phi(L)^{-1} (1 - L)^{-d} \left[1 + \psi(L)\right] g\left(z_{t-1}\right). \tag{2.16}
\]

Finally, the FIAPARCH \((p,d,q)\) model can be written as

\[
\sigma_t^d = \omega + \left[1 - (1 - \beta(L))^{-1} \phi(L)(1 - L)^d\right] (|\varepsilon_t| - \gamma) \delta.
\]

3. Estimation Methods

3.1 The Distributions

Three distributions are available in our program: the usual Gaussian (Normal) distribution, the Student-t distribution and the Generalized Error Distribution (GED).

The GARCH models are estimated using a maximum likelihood (ML) approach. The logic of ML is to interpret the density as a function of the parameters set, conditional on a set of sample outcomes. This function is called the likelihood function. It is quite evident from Equation (2.5) (and all the following equations of Section 2) that the recursive evaluation of this function is conditional on unobserved values. The ML estimation is therefore not perfectly exact. To solve the problem of unobserved values, we have set this quantities to their unconditional expected values.

---

9 Using the option BBM, \((1 - L)^d\) is truncated at some predefined value (see above). It is also possible to truncate this polynomial at the information size at time \(t\), i.e. \(t-1\) (using option Chung in the Light version).

10 Notice that the FIAPARCH is also available in the ARFIMA-type specification, i.e. \(\sigma_t^d = \sigma^2 + \left[1 - (1 - \beta(L))^{-1} \phi(L)(1 - L)^d\right] (|\varepsilon_t| - \gamma) \delta - \sigma^2\) by using the option "Chung".
If we express the mean equation as in Equation (2.1) and $\epsilon_t = z_t \sigma_t$, the density function of the standard normal distribution is written as

$$f_u(\epsilon_t | \psi_{t-1}) = \frac{\exp(-0.5 z_t^2)}{\sigma_t \sqrt{2\pi}}.$$ 

The log-likelihood function to maximize is given by

$$L_T = -\frac{1}{2} \sum_{t=1}^{T} \left[ \ln(2\pi) + \ln(\sigma_t^2) + z_t^2 \right],$$

where $T$ is the number of observations.

For a Student-$t$ distribution, the log-likelihood is

$$L_T = \ln \left[ \Gamma \left( \frac{v+1}{2} \right) \right] - \ln \left[ \Gamma \left( \frac{v}{2} \right) \right] - 0.5 \ln(v-2) - 0.5 \sum_{t=1}^{T} \ln(\sigma_t^2 + (1+v)\ln(1 + \frac{z_t^2}{v-2})),$$

(2.19)

where $v$ is the degrees of freedom, $0 < v \leq \infty$ and $\Gamma(\cdot)$ is the gamma function.

The GED log-likelihood function of a normalized random variable is given by

$$L_T = \sum_{t=1}^{T} \left[ \ln(v/\lambda) - 0.5 \frac{z_t^2}{\lambda} - (1+v^{-1})\ln(2) - \ln(\Gamma(1/v)) - 0.5 \ln(\sigma_t^2) \right],$$

(2.20)

where $-\infty < z_t < \infty$, $0 < v \leq \infty$, $\Gamma(\cdot)$ is the gamma function and

$$\lambda = \sqrt{\frac{2^{1-v}(v)}{\Gamma(1/v)}}.$$

Of course, the choice of distribution has an particular impact on some models, such as the EGARCH.11

---

11 For the Normal (Gaussian) distribution, the $g(z_t)$ function of equation 2.4 is

$$g(z_t) = \theta_1 z_t + \theta_2 \left[ |z_t| - \sqrt{\frac{2}{\pi}} \right].$$

For the Student-$t$ distribution,

$$g(z_t) = \theta_1 z_t + \theta_2 \left[ |z_t| - \frac{2\sqrt{(v-2)}}{(v-1)\Gamma\left(\frac{v+1}{2}\right) - \Gamma\left(\frac{v}{2}\right) - \Gamma(0.5)} \right].$$
3.2. The Standard Deviation Estimation Methods

Approximate Maximum Likelihood Estimates (MLE) and Quasi-Maximum Likelihood Estimates (QMLE) of the standard deviations of the estimated parameters are available. With the MLE method, when the assumption of conditional normality does not hold, the ARCH parameters estimates will still be consistent (provided the mean and variance functions are correctly specified) but the estimates of the covariance matrix will not be consistent. Thus QMLE provide a solution to this problem. The method used for the QMLE estimation is the one proposed by Bollerslev and Wooldridge (1992).

3.3 Tests

In addition to the possibilities offered by GiveWin (ACF, PACF, QQ-plots…), several tests are provided in our package:

1) Four Information Criteria that can be expressed as: \(^{12}\)

- Akaike = \(-2 \frac{\text{LogL}}{n} + 2 \frac{k}{n}\)

- Hannan-Quinn = \(-2 \frac{\text{LogL}}{n} + 2 \frac{k \log [\log (n)]}{n}\)

- Schwarz = \(-2 \frac{\text{LogL}}{n} + 2 \frac{\log (k)}{n}\)

- Shibata = \(-2 \frac{\text{LogL}}{n} + \log \left( \frac{n + 2k}{n} \right)\)

2) The value of the skewness and the kurtosis of the standardized residuals of the estimated model, their \(t\)-tests and \(p\)-values. Moreover, the Jarque-Bera normality test (Jarque and Bera, 1987) is also reported.

\[ g(z_i) = |z_i| - \lambda 2^{\left(\frac{2}{\upsilon}\right)} \frac{\Gamma\left(\frac{2}{\upsilon}\right)}{\Gamma\left(\frac{1}{\upsilon}\right)} - I \# z_i \text{ where } \lambda = \sqrt{\frac{2^{\left(-\frac{2}{\upsilon}\right)} \Gamma\left(\frac{1}{\upsilon}\right)}{\Gamma\left(\frac{3}{\upsilon}\right)}} \]

with \(\upsilon\) being the degree of freedom of the distribution and \(\Gamma(.)\) denoting the gamma function.

Using the same notation for the GED, we have

\[ g(z_i) = |z_i| - \lambda 2^{\left(\frac{2}{\upsilon}\right)} \frac{\Gamma\left(\frac{2}{\upsilon}\right)}{\Gamma\left(\frac{1}{\upsilon}\right)} - I \# z_i \text{ where } \lambda = \sqrt{\frac{2^{\left(-\frac{2}{\upsilon}\right)} \Gamma\left(\frac{1}{\upsilon}\right)}{\Gamma\left(\frac{3}{\upsilon}\right)}} \]

\(^{12}\) LogL = log likelihood value, n = #observations and k = # parameters

- 26 -
3) The modified Box-Pierce statistics at lag \( l^* \) for both standardized, i.e. \( BP(l^*) \), and squared standardized residuals, i.e. \( BP^2(l^*) \). Under the null hypothesis of no autocorrelation, the statistics \( BP(l^*) \) and \( BP^2(l^*) \) should be evaluated against the \( \chi^2(l^* - m - l) \) and \( \chi^2(l^* - p - q) \), respectively (see McLeod and Li, 1983).

4) The diagnostic test of Engle and Ng (1993) that investigate possible misspecification of the conditional variance equation. The Sign Bias Test examines the impact of positive and negative return shocks on volatility not predicted by the model under construction. The negative Size Bias Test (resp. positive Size Bias Test) focuses on the different effects that large and small negative (resp. positive) return shocks have on volatility, which is not predicted by the volatility model. Finally, a joint test for these three tests is also provided.

5) The Engle LM ARCH test (Engle, 1982) to test the presence of ARCH effects in a series.
IV. Using the Program

1. Installing the Files
To run the G@RCH 1.1 package, unzip the garch11.zip file in the ox/packages directory (using folders names). A new Garch folder should be automatically created. The list of the files contained in this garch11.zip file is available in readme.txt.

To run the "Full Version" of the program, you need to be sure that garch.oxo and garch.h are present. Moreover, you have to be sure that you have a registered version of Ox Professional 2.20, and that GiveWin and OxPack are correctly installed on your computer. If you want to use the "Light Version", you need GarchEstim.ox, garch.oxo and garch.h.

2. Running the “Full Version”
Open the database you want to use in GiveWin, and then select the OxPack module by clicking on Modules\Start OxPack (see Figure 4.1).

Figure 4.1. Launching OxPack
Once you are in the OxPack environment, select the Add/Remove Package... option in the Package menu. Click on the Browse button, then find and select \texttt{garch.oxo}. Click on the Add button (see Figure 4.2).

![Figure 4.2. Adding the Garch package](image)

Then select Package/Garch and launch Model/Estimate. We are now in the windows environment of our program. The list of all the variables of the database appears in the Database section. Select those you want to use in your estimation and click on the Add button. There are three possible statuses for each variable: dependent variable (Y variable), regressor in the Mean (Mean) or regressor in the variance (Variance).

Our program provides estimation for univariate models, so only one Y variable per model is accepted. However you can include several regressors in the mean and variance equations and the same variable can be a regressor in both equations as in our example (Figure 4.3). Once you click on OK, the Model Settings box automatically appears. This box allows you to select the specification of the model: AR(FI)MA orders for the mean equation, GARCH orders, type of GARCH model for the variance equation and the distribution (Figure 4.4).
Figure 4.3. Selecting the variables

Figure 4.4. Model Settings
When you click OK, you are asked to make a choice regarding the starting values (Figure 4.5): you may (1) let the program choose the starting values, (2) enter them manually, element by element, or (3) enter the starting values in a vector form (the required form is "value1;value2;value3").

Figure 4.5. Selecting the Starting Values Method

The first method is obviously the easiest, and may be indicated for beginning users, since it prevents from entering aberrant values. If a user wants particular starting values for the estimation and if he does not know the sequence of the parameters in the parameter vector used in our program, the second method should be a solution. An advanced user knowing the program quite well may use the third option as it is faster to do than the previous one. Note that, in the output, the estimated parameters are notably given in a vector form, so that a user can just copy the vector and paste it in this box for a subsequent estimation.

Then, the estimation method for standard deviations is asked: Maximum Likelihood (ML) or Quasi-Maximum Likelihood (QML) or both. On this box (see Figure 4.6), one may also select the sample and some maximization options when clicking on the Options button (such has the number of iterations between printing intermediary results...).
When the user clicks on OK, the Estimation procedure is launched and he comes back to GiveWin. A new dialog box is launched if the starting values are entered manually.

Let us assume that the element-by-element method has been selected. A new window appears with all the possible parameters to be estimated. Depending on the specification, some parameters have a value, other have not. The user should replace only the former, since they are the parameters to be estimated for the specified model.

An example is proposed on Figure 4.7. If there are more than one value for the same parameter (for instance an AR(2) process requires 2 autoregressive coefficients), they have to be separated by semicolons and entered as follows: value1;value2;…;valuek.
When the user clicks on the OK button, the program starts iterating and estimating. Depending on the options selected earlier, it prints intermediary iteration results such as this one:

Position after 10 BFGS iterations
parameters
// [0][0]...[0][9]
-0.053958  0.072994  -0.065922  0.0025936   0.059744
  0.64508  0.23925   -0.043541    1.4105    1.0566
gradients
// [0][0]...[0][9]
  2383.8   -4.8684    572.86    17393.    1854.0    2104.8
  1629.5   -28.949    423.38   -375.30
function value =
  -774.187995086

Obviously, the first part corresponds to the current estimated values of the parameters, the second to their gradients and the third to the current Log Likelihood value of the function. The final output is divided by default in two main parts: the model specification reminder and the estimated values and other useful statistics of the parameters.
Here is an example:

SPECIFICATIONS
-------------
Mean Equation : ARMA (0, 0) model.
No regressor in the mean
Variance Equation : EGARCH (1, 1) model.
No regressor in the variance
The distribution is a Student distribution, with 4.99811
degrees of freedom.

Strong convergence using numerical derivatives
Log-likelihood = -3080.02

Quasi Maximum Likelihood Estimation

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Std.Error</th>
<th>t-value</th>
<th>t-prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cst(M)</td>
<td>0.050169</td>
<td>0.01778</td>
<td>2.822</td>
</tr>
<tr>
<td>Cst(V)</td>
<td>1.585800</td>
<td>1.02031</td>
<td>1.554</td>
</tr>
<tr>
<td>GARCH(Beta1)</td>
<td>0.982485</td>
<td>0.00598</td>
<td>164.2</td>
</tr>
<tr>
<td>ARCH(Alpha1)</td>
<td>0.296244</td>
<td>0.77182</td>
<td>0.3838</td>
</tr>
<tr>
<td>EGARCH(Theta1)</td>
<td>-0.019184</td>
<td>0.01667</td>
<td>-1.151</td>
</tr>
<tr>
<td>EGARCH(Theta2)</td>
<td>0.077116</td>
<td>0.04642</td>
<td>1.661</td>
</tr>
<tr>
<td>Student(DF)</td>
<td>4.998116</td>
<td>0.63231</td>
<td>7.905</td>
</tr>
</tbody>
</table>

Estimated Parameters Vector :

0.050169; 1.585800; 0.982485; 0.296244; -0.019184; 0.077116; 4.998116

Note that the estimations are based on the numerical gradients (and not the analytical gradients).

Once the model is estimated and once the results are printed, the program returns to OxPack. New options are available in the Test menu: Graphic Analysis, Tests and Store. The Graphic Analysis option allows to plot four different graphics: the raw series, the residuals, the squared residuals and the conditional variance. These graphics are drawn in GiveWin. Just as any other graph in the GiveWin environment, they can be easily edited (colour, size...) and exported in many formats (.eps, .ps, .wmf, .emf and .gwg).
Figure 4.8 provides an example:

![Figure 4.8. Graphic Analysis](image)

The *Tests* option launches a new dialog box (see Figure 4.9) where different tests are available:

- the Information Criteria (Akaike, Schwarz, Hannan-Quinn and Shibata),
- the skewness, kurtosis and Jarque-Bera normality test,
- the Box-Pierce statistics for both residuals and squared residuals,
- the Sign Bias test of Engle & Ng (1993) and
- the Engle' LM ARCH test.
The results of these tests are printed in GiveWin.

**TESTS :**

-------

**Information Criterium (minimize)**
Akaike 2.716250  Shibata 2.716231
Schwarz 2.733893  Hannan-Quinn 2.722686

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

**Statistic**  **t-Test**  **P-Value**
Skewness -1.6246 31.642 9.7533e-220
Excess Kurtosis 22.347 217.72 0.00000
Jarque-Bera 48296. 48296. 0.00000

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

**BOX-PIERCE :**

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean of standardized residuals</td>
</tr>
<tr>
<td>Mean of squared standardized residuals</td>
</tr>
</tbody>
</table>

Value

-0.00029

1.07289
H0 : No serial correlation ==> Accept H0 when prob. is High [Q < Chisq(lag)]

Box-Pierce Q-statistics on residuals
Q(5) = 8.27589  [0.141672 ]
Q(10) = 9.44816  [0.490164 ]
Q(50) = 46.8869  [0.599069 ]
---------------
Box-Pierce Q-statistics on squared residuals
--> P-values adjusted by 2 degree(s) of freedom
Q(5) = 7.05218  [0.070253 ]
Q(10) = 10.1216  [0.256595 ]
Q(50) = 15.9991  [0.999996 ]
---------------
Diagnostic test based on the news impact curve (EGARCH vs. GARCH)

<table>
<thead>
<tr>
<th>Test</th>
<th>Test</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sign Bias t-Test</td>
<td>0.97629</td>
<td>0.32892</td>
</tr>
<tr>
<td>Negative Size Bias t-Test</td>
<td>4.66895</td>
<td>0.00000</td>
</tr>
<tr>
<td>Positive Size Bias t-Test</td>
<td>0.17200</td>
<td>0.86344</td>
</tr>
<tr>
<td>Joint Test for the Three Effects</td>
<td>23.30871</td>
<td>0.00003</td>
</tr>
</tbody>
</table>
---------------

ARCH 1-2 test: F(2,2266)=2.39961 [0.0910]
---------------

Finally, the residuals, the squared residuals and the conditional variance series can be stored in the database as a new variable. When selecting this option, a first window appears and the user selects the series to be stored (Figure 4.10a). A default name is then proposed for this series (Figure 4.10b).
Figure 4.10a. Storing in the Database

Figure 4.10b. Storing in the Database
3. Running the "Light Version"

First, to specify the model you want to estimate, you have to edit *GarchEstim.ox* with any text editor. Yet we recommend OxEdit. It is a shareware that highlights Ox syntax in color and its 1.50 version is free (see http://www.oxedit.com for more details). From now on, we will assume that OxEdit is used.

The file is displayed like this:

![Figure 4.11. GarchEstim.ox (part I)](image)

The #include and #import statements show the files and the libraries that are linked with *GarchEstim*. First, a new Garch object is created and a database is loaded. The user has to enter the correct path of the database, but also has to pay attention to the kind of database he wants to use. Indeed, the LoadIn7 function is dedicated to In7 files (GiveWin databases). For other extensions, other functions need to be called. For instance, to use a Microsoft Excel file, one has to pay attention to two elements: the use of the correct function (*LoadXls*) and the format of the spreadsheet. The following convention has to be adopted when loading an Excel spreadsheet: variables are in columns, columns with variables are labeled, there is an unlabeled column containing the dates (with the form Year-Period) and the data form a contiguous sample.
Here is a small example:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>RET</td>
<td>MON</td>
<td>HOL</td>
</tr>
<tr>
<td>2</td>
<td>1990-1</td>
<td>0.0439</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1990-2</td>
<td>-0.0302</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1990-3</td>
<td>0.0845</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Coming back to Figure 4.11, we note then that the dependent variable (Y), the regressor(s) in the mean equation (X) and the regressor(s) in the variance equation (Z) are selected with the Select function. Ox being case-sensitive, the exact name of the variable has to be entered. The second and third arguments denote the starting and ending observations to be considered. By default, "0" and "0" mean that all the observations are selected. From this selection, a sample can be extracted with the SetSelSample function. The arguments are ordered as (StartYear, StartPeriod, EndYear, EndPeriod2) and the default (-1, 1, -1, 1) means all the selected observations. Next, the specification of the model for both the conditional mean and the conditional variance equations can be fixed with the functions displayed on Figure 4.12.

---

13 See Doornik (1996) for the supported formats, the Load functions and other related information. You can also take a look at the DJIA.xls file included in the package for an example of Excel file ready to be loaded by Ox.
Quite explicit explanations are given within the *GarchEstim* file, so we will not explain every line. We will nevertheless mention several elements. The specification consists in four parts:

- the GARCH related part (from *GARCH* to *APARCH*) where orders, specific models or distribution are specified.
- the Tests and Estimation Procedures part. *MLE* is related to the computation method of the standard deviations of the estimated parameters, *LAGS* requires a vector of integers corresponding to the lags of the Box-Pierce statistics and *TESTONLY* is useful when you want to run some tests on the raw series, prior to any estimation.
- the Parameters related part consists in the *DoEstimation* function that launches the estimation of the model. The argument is a vector containing starting values of the parameters in a specified order. If "<>" is entered as argument, the program will use predefined values as the starting values for the parameters. The order of the parameters is the following (see the equation in Models section of this tutorial):

  <Cst in Mean; Regressors in mean coefficients; d in ARFIMA ; AR coefficients ; MA coefficients ; Cst in Variance ; Regressors in variance coefficients ; FI coefficient ; GARCH β coefficients ; ARCH α coefficients ; GJR ω coefficient ; EGARCH θ₁ ; EGARCH θ₂ ; APARCH γ coefficient ; APARCH δ coefficient ; GED degree of freedom ; Student degree of freedom>

- the Results part include the *GRAPHS* function to print or not the graphics after the estimation and the *STORE* function, which allows storing the estimated residuals, squared residuals or conditional variances in the opened database or as a new file.

When the parameterization is done, run the file. If you use *oxl*, select *Modules/Run Default Module* and the results will be printed in OxEdit. If you use *OxRun*, select *Modules/OxRun* and the results will be printed in GiveWin. Graphics will only be displayed if you are using GiveWin.
V. Versions and Future Improvements

1. Releases History
v.1.1 : October, 30th 2000.
v.1.0 : September, 4th, 2000.

2. Future Improvements
- Analytical gradients
- Mean / Variance / Density forecast
- Inclusion of other densities (mixtures of Normal or Student-t)
- Specification tests (stability, goodness-of-fit, …)
- GARCH-In-Mean and extensions
- Faster procedures for Fractionally Integrated models
- …

We wish you a productive use of GARCH 1.1!

VI. References


GRANGER, C.W.J., & JOYEUX, R., (1980), "An Introduction to Long-Memory Time
Series Models and Fractional Differencing", Journal of Time-Series Analysis,
Vol.1, 15-29.

JARQUE, C.M., & BERA, A.K., (1987), "A Test for Normality of Observations and

Using Squared Residuals Autocorrelations", Journal of Time Series Analysis, 4,
269-273.


PENTULA, S.G., (1986), "Modeling the Persistence of Conditional Variances: A

SCHWERT, W., (1990), "Stock Volatility and the Crash of '87", Review of Financial

Sons.


Dynamics and Control, Vol.18, 931-955.