Package Development in R: Implementing GO-GARCH models

Dr. Bernhard Pfaff
bernhard_pfaff@fra.invesco.com

Invesco Asset Management Deutschland GmbH, Frankfurt am Main

Rmetrics: Meielisalp Workshop 2009
06/28/ – 07/02/2009
Meielisalp, Lake Thune, Switzerland
Contents

1 Introduction
2 GO-GARCH
3 Design
4 Implementation
5 Literature
Task: From academic paper to implementation.

- Putting econometric methods/models into action.
- Design of a program structure that meets the needs of the user: form follows function.
- Helpful tools to fulfill this task.
- Example: Multivariate GARCH model class.
Generally, a $m$-dimensional GARCH for a second-order stationary process $\{x_t\}$ is given as:

$$x_t | \mathcal{I}_{t-1} \sim \mathcal{D}(\phi, V_t), \quad (1)$$

whereby the information set up to time $t-1$ is denoted by $\mathcal{I}_{t-1}$ and the variance-covariance matrix $V_t$ is time dependent. The information set contains lagged values of the squares and cross-products of $x_t$ and elements of the conditional covariance matrices. The vector $\phi$ contains distribution dependent parameters. Ordinarily for $\mathcal{D}$ a Normal distribution is assumed.
According to the survey article of Bauwens et al (2006) multivariate GARCH models can be classified into three areas:

- Direct generalizations of the univariate GARCH model, e.g. VEC, BEKK and factor models.
- Linear combinations of univariate GARCH models, e.g. (generalized) orthogonal and latent factor GARCH models.
- Nonlinear combinations of univariate GARCH models, e.g. dynamic conditional correlation and general dynamic covariance models.
Introduction

Resources in R

- **Univariate:**
  1. Package bayesGARCH (CRAN).
  2. Rmetrics package fGarch (CRAN & R-Forge).
  3. Package rgarch (R-Forge).
  4. Function `garch()` in package tseries (CRAN).

- **Multivariate:**
  1. Package ccgarch (CRAN).
  2. Package gogarch (CRAN & R-Forge).
  3. Package rgarch (R-Forge).

---

For the sake of completeness a project rregrach has been registered on R-Forge, but only an empty project skeleton exists as of the time this presentation is drafted.
The observed $m$-dimensional economic process $\{x_t\}$ is governed by a linear combination of uncorrelated economic components $\{y_t\}$:

$$x_t = Z y_t$$  \hspace{1cm} (2)

The linear map $Z$ that links the unobserved components with the observed variables is assumed to be constant over time, and invertible. The unobserved components are normalized to have unit variance, such that:

$$V = \mathbb{E}[x_t x_t'] = ZZ'$$  \hspace{1cm} (3)
An explicit example, whereby the Normal distribution is assumed would then be:

\[ x_t = Z y_t \text{ with } y_t \sim \mathcal{N}(0, H_t) \] (4)

and each component is described by a GARCH(1, 1) process:

\[ H_t = \text{diag}(h_{1,t}, \ldots, h_{m,t}) \] (5)
\[ h_{i,t} = \omega_i + \alpha_i y_{i,t-1}^2 + \beta_i h_{i,t-1}, \text{ for } i = 1, \ldots, m \] (6)

The unconditional covariance matrix of the components is \( H_0 = I \) and the conditional covariances of \( \{x_t\} \) are given by \( V_t = Z H_t Z' \).
Let $Z$ be the map that links the uncorrelated components $\{y_t\}$ with the observed process $\{x_t\}$. Then there exists an orthogonal matrix $U_0$ such that:

$$P \Delta^{\frac{1}{2}} U_0 = Z$$

(7)

The matrices $P$ and $\Delta^{\frac{1}{2}}$ can be retrieved by SVD from sample information, i.e, the unconditional variance/covariance matrix of $\{x_t\}$. And what about $U_0$?
One parametrization of $U$ is given as follows: Every $m$-dimensional orthogonal matrix $U$ with $\det(U) = 1$ can be represented as a product of $\binom{m}{2} = [m(m-1)]/2$ rotation matrices:

$$U = \prod_{i<j} R_{ij}(\theta_{ij}) \text{ with } -\pi \leq \theta_{ij} \leq \pi ,$$

(8)

where $R_{ij}(\theta_{ij})$ performs a rotation in the plane spanned by $e_i$ and $e_j$ over an angle $\theta_{ij}$; the so-called Euler angles.
GO-GARCH

Estimation by . . .

- Maximum-Likelihood, see van der Weide (2002).
- Non-linear Least-Squares, see Boswijk and van der Weide (2006).
- Methods-of-Moments, see Boswijk and van der Weide (2009).
- Fast ICA, see Broda and Paolella (2008).
Think of a model class in terms of an object, i.e., GO-GARCH.

Write methods for estimating your model, retrieving and/or displaying items from that object.

Provide the user with as many options for a tailor-made usage.

Adhere to a naming convention for classes and functions.

Error checking and handling as early as possible.

Use coercing where appropriate and possible.

Employ checks, tests and/or validation.
Class definition for orthogonal matrices: *Orthom*.

Class for initial model object: *Goinit*.

Class for GO-GARCH models: *GoGARCH* (inherits from *Goinit*)

Dependent upon the chosen estimation method (inherits from *GoGARCH*):

1. Fast ICA: *Goestica*
2. Methods of Moments: *Goestmm*
3. Maximum-Likelihood: *Goestml*
4. Non-linear Least-Squares: *Goestnls*

Class for summary objects: *Gosum*

Class for predict objects: *Gopredict*
Think of methods as the answers to the question: “What does the user wants or is interested in?”.

- Estimation (goest) and displaying the result, i.e., show, print, summary.
- Displaying the result graphically, i.e., plot.
- Update the model’s structure and/or estimation method, i.e., update.
- Retrieval the conditional variances, i.e., cvar, ccor, ccov.
- Calculate forecasts, i.e., predict, and the predicted conditional variances should be made available (cvar, ccor, ccov).
Main functions for creating objects of a certain formal class: gogarch and goinit.

Auxiliary functions for clearer code appearance, *i.e.* objective function to optimize (gollh, gonls), manipulation of objects (unvech, Umatch, UprodR, Rd2, gotheta).

Validation: validGoinitObject for objects of class *Goinit* and validOrthomObject for objects of class *Orthom*.

Hint: Prefix function’s name with a dot if it should not be documented in a package.
Implementation
Key Points

- Package gogarch purely written in R.
- Current version 0.6-9 as of 2009-04-28.
- S4 classes/methods and NAMESPACE employed.
- Dependencies: R (>= 2.7.0), methods, stats, graphics, fGarch, fastICA
- License: GPL (>= 2)
- Available at:
  1. http://cran.r-project.org
Release your code early and often.

Package your code as early as possible.

Employ a source control software, e.g. Subversion.

Document your code (within the functions and in form of a manual).

Provide examples and/or a “How to” and/or a vignette.

Publicize your package, e.g. JSS [http://www.jstatsoft.org/](http://www.jstatsoft.org/).

If possible, replicate trustworthy results from published articles.
### Implementation

#### Package’s Structure

<table>
<thead>
<tr>
<th>Methods</th>
<th>Goinit</th>
<th>GoGARCH</th>
<th>Goestica</th>
<th>Goestmm</th>
<th>Classes</th>
<th>Goestml</th>
<th>Goestnls</th>
<th>Gosum</th>
<th>Gopredict</th>
<th>Orthom</th>
</tr>
</thead>
<tbody>
<tr>
<td>angles</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ccor</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>ccov</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>converged</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>coef</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cvar</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>formula</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>logLik</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>plot</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>predict</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>print</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>resid</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>residuals</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>show</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>summary</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>update</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>
Implementation
Economic coding, 1a

> getMethod("coef", "GoGARCH")

Method Definition:

function (object, ...) {
  .local <- function (object)
  {
    garchc <- matrix(unlist(lapply(object@models, coef)),
                     nrow = ncol(object@X), byrow = TRUE)
    colnames(garchc) <- names(object@models[[1]]@fit$par)
    rownames(garchc) <- paste("y", 1:nrow(garchc), sep = "")
    return(garchc)
  }
  .local(object, ...)
}
<environment: namespace:gogarch>

Signatures:
  object
  target "GoGARCH"
  defined "GoGARCH"
Implementation

Economic coding, IIb

Due to inheritance one only needs:

```r
> getMethod("coef", "Goestmm")
```

Method Definition:

```r
function (object, ...)
{
  .local <- function (object)
  {
    callNextMethod()
  }
  .local(object, ...)
}
<environment: namespace:gogarch>
```

Signatures:

- object
- target "Goestmm"
- defined "Goestmm"

which results from the sources:

```r
setMethod(f = "coef", signature(object = "Goestmm"), definition = function(object){
  callNextMethod()
})
```
Implementation
Economic coding, IIa

> getMethod("cvar", "GoGARCH")

Method Definition:

function (object, ...)
{
  .local <- function (object)
  {
    m <- ncol(object@X)
    n <- nrow(object@X)
    cvar <- matrix(c(unlist(lapply(object@H, function(x) diag(x)))),
      ncol = m, nrow = n, byrow = TRUE)
    colnames(cvar) <- paste("V.", colnames(object@X), sep = "")
    rownames(cvar) <- rownames(object@X)
    cvar <- as.ts(cvar)
    return(cvar)
  }
  .local(object, ...)
}
<environment: namespace:gogarch>

Signatures:
  object
  target "GoGARCH"
  defined "GoGARCH"
Implementation
Economic coding, IIb

Due to inheritance one only needs:

```r
> getMethod("cvar", "Goestmm")
```

Method Definition:

```r
function (object, ...)
{
  .local <- function (object)
  {
    cvar(as(object, "GoGARCH"))
  }
  .local(object, ...)
}
<environment: namespace:gogarch>
```

Signatures:

```r
object
target "Goestmm"
defined "Goestmm"
```

which results from the sources:

```r
setMethod(f = "cvar", signature(object = "Goestmm"), definition = function(object){
  cvar(as(object, "GoGARCH"))
})
```
Implementation
Example Ia, van der Weide (2002)

```r
> library(gogarch)
> library(vars)
> data(VDW)
> var1 <- VAR(scale(VDW), p = 1,
+     type = "const")
> resid <- residuals(var1)
> gogml <- gogarch(resid, ~garch(1, 1),
+     scale = TRUE, estby = "ml",
+     control = list(iter.max = 1000))
> gogml
> solve(gogml@Z)
> plot(gogml)
```

****************
*** GO-GARCH ***
****************

Components estimated by: maximum likelihood
Dimension of data matrix: (3081 x 2).
Formula for component GARCH models: ~ garch(1, 1)

Orthogonal Matrix U:
```
[,1] [,2]
[1,] 0.6554845 -0.7552086
[2,] 0.7552086 0.6554845
```

Linear Map Z:
```
[,1] [,2]
[1,] -0.2766742 -0.9607949
[2,] -0.9123173 -0.4090874
```

Estimated GARCH coefficients:
```
omega alpha1 beta1
y1 0.003037073 0.09098498 0.9071785
y2 0.010014348 0.05231209 0.9401891
```

Convergence codes of component GARCH models:
```
y1  y2
 1   1
```
Implementation
Example Ib, van der Weide (2002)
Implementation
Example II, Boswijk and van der Weide (2006)

```r
> data(BVDW)
> BVDW <- zoo(x = BVDW[, -1],
+     order.by = BVDW[, 1])
> BVDW <- diff(log(BVDW)) * 100
> gognls <- gogarch(BVDW, formula =
+     ~garch(1,1), scale = TRUE,
+     estby = "nls")
> gognls
```

```
****************
*** GO-GARCH ***
****************
Components estimated by: non-linear Least-Squares
Dimension of data matrix: (2609 x 2).
Formula for component GARCH models: ~ garch(1, 1)

Orthogonal Matrix U:

<table>
<thead>
<tr>
<th>[,1]</th>
<th>[,2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>-0.5241201</td>
</tr>
<tr>
<td>[2,]</td>
<td>0.8516443</td>
</tr>
</tbody>
</table>

Linear Map Z:

<table>
<thead>
<tr>
<th>[,1]</th>
<th>[,2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>0.8141711</td>
</tr>
<tr>
<td>[2,]</td>
<td>0.1511830</td>
</tr>
</tbody>
</table>

Estimated GARCH coefficients:

<table>
<thead>
<tr>
<th>omega</th>
<th>alpha1</th>
<th>beta1</th>
</tr>
</thead>
</table>
y1 0.009343703 0.08749236 0.9049185
y2 0.005866510 0.04367927 0.9520289

Convergence codes of component GARCH models:

y1 y2

1 1
```
Implementation
Example III, Boswijk and van der Weide (2009)

```r
> data(BVDWSTOXX)
> BVDWSTOXX <- zoo(x = BVDWSTOXX[, -1],
+ order.by = BVDWSTOXX[, 1])
> BVDWSTOXX <- window(BVDWSTOXX,
+ end = as.POSIXct("2007-12-31"))
> BVDWSTOXX <- diff(log(BVDWSTOXX))
> sectors <- BVDWSTOXX[, c("AutoParts",
+ "Banks", "OilGas")]
> sectors <- apply(sectors, 2, scale,
+ scale = FALSE)
> gogmm <- gogarch(sectors, formula =
+ ~garch(1,1), estby = "mm",
+ lag.max = 100)
> gogmm
```

```
****************
*** GO-GARCH ***
****************

Components estimated by: Methods of Moments
Dimension of data matrix: (5420 x 3).
Formula for component GARCH models: ~ garch(1, 1)

Orthogonal Matrix U:

<table>
<thead>
<tr>
<th></th>
<th>[1]</th>
<th>[2]</th>
<th>[3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>0.97243228</td>
<td>-0.1580023</td>
<td>0.1714956</td>
</tr>
<tr>
<td>[2,]</td>
<td>0.04054402</td>
<td>0.8388089</td>
<td>0.5429142</td>
</tr>
<tr>
<td>[3,]</td>
<td>-0.2296337</td>
<td>-0.5209942</td>
<td>0.8220909</td>
</tr>
</tbody>
</table>

Linear Map Z:

<table>
<thead>
<tr>
<th></th>
<th>[1]</th>
<th>[2]</th>
<th>[3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>0.012150308</td>
<td>0.006416573</td>
<td>-0.003053992</td>
</tr>
<tr>
<td>[2,]</td>
<td>0.003914599</td>
<td>0.010216543</td>
<td>-0.003626263</td>
</tr>
<tr>
<td>[3,]</td>
<td>0.004364691</td>
<td>0.008865709</td>
<td>0.006737270</td>
</tr>
</tbody>
</table>

Estimated GARCH coefficients:

<table>
<thead>
<tr>
<th></th>
<th>omega</th>
<th>alpha1</th>
<th>beta1</th>
</tr>
</thead>
<tbody>
<tr>
<td>y1</td>
<td>0.013982146</td>
<td>0.06081926</td>
<td>0.9256579</td>
</tr>
<tr>
<td>y2</td>
<td>0.003267435</td>
<td>0.04320838</td>
<td>0.9542282</td>
</tr>
<tr>
<td>y3</td>
<td>0.021353037</td>
<td>0.07085826</td>
<td>0.9065798</td>
</tr>
</tbody>
</table>

Convergence codes of component GARCH models:

<table>
<thead>
<tr>
<th></th>
<th>y1</th>
<th>y2</th>
<th>y3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
```


