Package ‘lgarch’

October 13, 2022

Type Package

Title Simulation and Estimation of Log-GARCH Models

Version 0.6-2

Depends R (>= 2.15.0), zoo

Date 2015-09-14

Author Genaro Sucarrat

Maintainer Genaro Sucarrat <genaro.sucarrat@bi.no>

URL http://www.sucarrat.net/

Description Simulation and estimation of univariate and multivariate log-GARCH models. The main functions of the package are: lgarchSim(), mlgarchSim(), lgarch() and mlgarch(). The first two functions simulate from a univariate and a multivariate log-GARCH model, respectively, whereas the latter two estimate a univariate and multivariate log-GARCH model, respectively.

License GPL-2

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-09-15 19:46:40

R topics documented:

`lgarch-package` *Simulation and estimation of log-GARCH models*

Description

This package provides facilities for the simulation and estimation of univariate log-GARCH models, and for the multivariate CCC-log-GARCH(1,1) model, see Sucarrat, Gronneberg and Escribano (2013), Sucarrat and Escribano (2013), and Francq and Sucarrat (2013).

Let y[t] denote a financial return or the error of a regression at time t such that

y[t] = sigma[t]*z[t],

where sigma[t] > 0 is the conditional standard deviation or volatility at t, and where z[t] is an IID innovation with mean zero and unit variance. The log-volatility specifiction of the log-GARCH-X model is given by

ln sigma[t]^2 = intercept + Sum_i alpha_i * ln y[t-i]^2 + Sum_j beta_j *ln sigma[t-1]^2 + Sum_k lambda_k * x[t]_k,

where the conditioning x-variables can be contemporaneous and/or lagged. The lgarch package estimates this model via its ARMA-X representation, see Sucarrat, Gronneberg and Escribano (2013), and treats zeros on y as missing values, see Sucarrat and Escribano (2013).

Details

Package:	lgarch
Type:	Package
Version:	0.6-2
Date:	2015-09-14
License:	GPL-2
LazyLoad:	yes

The main functions of the package are: `lgarchSim`, `mlgarchSim`, `lgarch` and `mlgarch`. The first two functions simulate from a univariate and a multivariate log-GARCH model, respectively, whereas the latter two estimate a univariate and a multivariate log-GARCH model, respectively.

The lgarch and mlgarch functions return an object (a list) of class 'lgarch' and 'mlgarch', respectively. In both cases a collection of methods can be applied to each of them: coef, fitted, logLik, print, residuals, summary and vcov. In addition, the function `rss` can be used to extract the Residual Sum of Squares of the estimated ARMA representation from an lgarch object.

The output produced by the `lgarchSim` and `mlgarchSim` functions, and by the fitted and residuals methods, are of the Z's ordered observations (`zoo`) class, see Zeileis and Grothendieck (2005), and Zeileis, Grothendieck and Ryan (2014). This means a range of time-series and plotting methods are available for these objects.

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

References

Francq, C. and G. Sucarrat (2013), 'An Exponential Chi-Squared QMLE for Log-GARCH Models via the ARMA Representation', MPRA Paper 51783: http://mpra.ub.uni-muenchen.de/51783/

Sucarrat, G. and A. Escribano (2013), 'Unbiased QML Estimation of Log-GARCH Models in the Presence of Zero Returns', MPRA Paper 50699: http://mpra.ub.uni-muenchen.de/50699/

Sucarrat, G., S. Gronneberg and A. Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: http://mpra.ub.uni-muenchen.de/49344/

Zeileis, A. and G. Grothendieck (2005), 'zoo: S3 Infrastructure for Regular and Irregular Time Series', Journal of Statistical Software 14, pp. 1-27

Zeileis, A., G. Grothendieck, J.A. Ryan and F. Andrews(2014), 'zoo: S3 Infrastructure for Regular and Irregular Time Series (Z's ordered observations)', R package version 1.7-11, http://CRAN.R-project.org/package=zoo/

See Also

lgarchSim, mlgarchSim, lgarch, mlgarch, coef.lgarch, coef.mlgarch, fitted.lgarch, fitted.mlgarch, logLik.lgarch, logLik.mlgarch, print.lgarch, print.mlgarch, residuals.lgarch, residuals.mlgarch, rss, summary.mlgarch, summary.mlgarch, vcov.lgarch, vcov.mlgarch and zoo

Examples

```
##simulate 500 observations w/default parameter values from
##a univariate log-garch(1,1):
set.seed(123)
y <- lgarchSim(500)

##estimate a log-garch(1,1):
mymod <- lgarch(y)

##print results:
print(mymod)

##extract coefficients:
coef(mymod)

##extract Gaussian log-likelihood (zeros excluded, if any) of the log-garch model:
logLik(mymod)
```

```
##extract Gaussian log-likelihood (zeros excluded, if any) of the arma representation:
logLik(mymod, arma=TRUE)

##extract variance-covariance matrix:
vcov(mymod)

##extract and plot the fitted conditional standard deviation:
sdhat <- fitted(mymod)
plot(sdhat)

##extract and plot standardised residuals:
zhat <- residuals(mymod)
plot(zhat)

##extract and plot all the fitted series:
myhat <- fitted(mymod, verbose=TRUE)
plot(myhat)

##simulate 1000 observations from a two-dimensional
##ccc-log-garch(1,1) w/default parameter values:
set.seed(123)
yy <- mlgarchSim(1000)

##estimate a 2-dimensional ccc-log-garch(1,1):
myymod <- mlgarch(yy)

##print results:
print(myymod)
```

coef.lgarch *Extraction methods for 'lgarch' objects*

Description

Extraction methods for objects of class 'lgarch' (i.e. the result of estimating a log-GARCH model)

Usage

```
## S3 method for class 'lgarch'
coef(object, arma = FALSE, ...)
## S3 method for class 'lgarch'
fitted(object, verbose = FALSE, ...)
## S3 method for class 'lgarch'
logLik(object, arma = FALSE, ...)
## S3 method for class 'lgarch'
print(x, arma = FALSE, ...)
## informal method for class 'lgarch'
rss(object, ...)
```

```
## S3 method for class 'lgarch'
residuals(object, arma = FALSE, ...)
## S3 method for class 'lgarch'
summary(object, ...)
## S3 method for class 'lgarch'
vcov(object, arma = FALSE, ...)
```

Arguments

`object`	an object of class 'lgarch'
`x`	an object of class 'lgarch'
`verbose`	logical. If FALSE (default), then only basic information is returned
`arma`	logical. If FALSE (default), then information relating to the log-garch model is returned. If TRUE, then information relating to the ARMA representation is returned
`...`	additional arguments

Details

Note: The rss function is not a formal S3 method.

Value

`coef:`	A numeric vector containing the parameter estimates
`fitted:`	A `zoo` object. If verbose = FALSE (default), then the zoo object is a vector containing the fitted conditional standard deviations. If verbose = TRUE, then the zoo object is a matrix containing the conditional standard deviations and additional information
`logLik:`	The value of the log-likelihood (contributions at zeros excluded) at the maximum
`print:`	Prints the most important parts of the estimation results
`residuals:`	A `zoo` object with the residuals. If arma = FALSE (default), then the standardised residuals are returned. If arma = TRUE, then the residuals of the ARMA representation is returned
`rss:`	A numeric; the Residual Sum of Squares of the ARMA representation
`summary:`	A print of the items in the `lgarch` object
`vcov:`	The variance-covariance matrix

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

See Also

`lgarch`

Examples

```
##simulate 500 observations w/default parameter values:
set.seed(123)
y <- lgarchSim(500)

##estimate a log-garch(1,1):
mymod <- lgarch(y)

##print results:
print(mymod)

##extract coefficients:
coef(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the log-garch model:
logLik(mymod)

##extract the Residual Sum of Squares of the ARMA representation:
rss(mymod)

##extract log-likelihood (zeros excluded) of the ARMA representation:
logLik(mymod, arma=TRUE)

##extract variance-covariance matrix:
vcov(mymod)

##extract and plot the fitted conditional standard deviation:
sdhat <- fitted(mymod)
plot(sdhat)

##extract and plot standardised residuals:
zhat <- residuals(mymod)
plot(zhat)

##extract and plot all the fitted series:
myhat <- fitted(mymod, verbose=TRUE)
plot(myhat)
```

`coef.mlgarch` *Extraction methods for 'mlgarch' objects*

Description

Extraction methods for objects of class 'mlgarch' (i.e. the result of estimating a multivariate CCC-log-GARCH model)

Usage

```
## S3 method for class 'mlgarch'
coef(object, varma = FALSE, ...)
## S3 method for class 'mlgarch'
fitted(object, varma = FALSE, verbose = FALSE, ...)
## S3 method for class 'mlgarch'
logLik(object, varma = FALSE, ...)
## S3 method for class 'mlgarch'
print(x, varma = FALSE, ...)
## S3 method for class 'mlgarch'
residuals(object, varma = FALSE, ...)
## S3 method for class 'mlgarch'
summary(object, ...)
## S3 method for class 'mlgarch'
vcov(object, varma = FALSE, ...)
```

Arguments

`object`	an object of class 'mlgarch'
`x`	an object of class 'mlgarch'
`verbose`	logical. If FALSE (default), then only basic information is returned
`varma`	logical. If FALSE (default), then information relating to the multivariate CCC-log-GARCH model is returned. If TRUE, then information relating to the VARMA representation is returned
`...`	additional arguments

Details

Empty

Value

`coef:`	A numeric vector containing the parameter estimates
`fitted:`	A `zoo` object (a matrix). If verbose = FALSE (default), then the zoo object contains the fitted conditional standard deviations of each equation. If verbose = TRUE, then the zoo object also contains additional information
`logLik:`	The value of the log-likelihood (contributions at zeros excluded) at the maximum
`print:`	Prints the most important parts of the estimation results
`residuals:`	A `zoo` object (a matrix) with the residuals. If varma = FALSE (default), then the standardised residuals are returned. If varma = TRUE, then the residuals of the VARMA representation is returned
`summary:`	A print of the items in the `mlgarch` object
`vcov:`	The variance-covariance matrix

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

See Also

mlgarch

Examples

```
##simulate 1000 observations from 2-dimensional model w/default parameter values:
set.seed(123)
y <- mlgarchSim(1000)

##estimate a 2-dimensional ccc-log-garch(1,1):
mymod <- mlgarch(y)

##print results:
print(mymod)

##extract ccc-log-garch coefficients:
coef(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the ccc-log-garch model:
logLik(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the varma representation:
logLik(mymod, varma=TRUE)

##extract variance-covariance matrix:
vcov(mymod)

##extract and plot the fitted conditional standard deviations:
sdhat <- fitted(mymod)
plot(sdhat)

##extract and plot standardised residuals:
zhat <- residuals(mymod)
plot(zhat)
```

---

gdiff *Difference a vector or a matrix, with special treatment of zoo objects*

Description

Similar to the diff function from the base package, but gdiff enables padding (e.g. NAs or 0s) of the lost entries. Contrary to the diff function in the base package, however, the default in gdiff is to pad (with NAs). The gdiff function is particularly suited for zoo objects, since their indexing is retained

Usage

```
gdiff(x, lag = 1, pad = TRUE, pad.value = NA)
```

Arguments

x	a numeric vector or matrix
lag	integer equal to the difference-length (the default is 1)
pad	logical. If TRUE (default), then the lost entries are padded with pad.value. If FALSE, then no padding is undertaken
pad.value	numeric. The pad-value

Value

A vector or matrix with the differenced values

Note

Empty

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

See Also

diff, glag, lag

Examples

```
##1st difference of a series:
x <- rnorm(5)
gdiff(x)

##1st difference with no padding:
gdiff(x, pad=FALSE)

##1st difference retaining the original zoo-index ordering:
gdiff(as.zoo(x))

##1st difference of a matrix:
y <- matrix(rnorm(8),4,2)
gdiff(y)

##2nd difference of the same matrix:
gdiff(y, lag=2)
```

glag *Lag a vector or a matrix, with special treatment of zoo objects*

Description

Similar to the lag function from the stats package, but glag enables padding (e.g. NAs or 0s) of the lost entries. Contrary to the lag function in the stats package, however, the default in glag is to pad (with NAs). The glag is particularly suited for zoo objects, since their indexing is retained

Usage

```
glag(x, k = 1, pad = TRUE, pad.value = NA)
```

Arguments

x	a numeric vector or matrix
k	integer equal to the lag (the default is 1)
pad	logical. If TRUE (default), then the lost entries are padded with pad.value. If FALSE, then no padding is undertaken
pad.value	the pad-value

Value

A vector or matrix with the lagged values

Note

Empty

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

See Also

lag, gdiff, diff

Examples

```
##lag series with NA for the missing entries:
x <- rnorm(5)
glag(x)

##lag series with no padding:
x <- rnorm(5)
glag(x, pad=FALSE)

##lag series and retain the original zoo-index ordering:
```

```
x <- as.zoo(rnorm(5))
glag(x)

##lag two periods:
glag(x, k=2)
```

---

lgarch *Estimate a log-GARCH model*

Description

Fit a log-GARCH model by either (nonlinear) Least Squares (LS) or Quasi Maximum Likelihood (QML) via the ARMA representation. For QML either the Gaussian or centred exponential chi-squared distribution can be used as instrumental density, see Sucarrat, Gronneberg and Escribano (2013), and Francq and Sucarrat (2013). Zero-values on the dependent variable y are treated as missing values, as suggested in Sucarrat and Escribano (2013). Estimation is via the `nlminb` function, whereas a numerical estimate of the Hessian is obtained with `optimHess` for the computation of the variance-covariance matrix

Usage

```
lgarch(y, arch = 1, garch = 1, xreg = NULL, initial.values = NULL,
  lower = NULL, upper = NULL, nlminb.control = list(), vcov = TRUE,
  method=c("ls","ml","cex2"), mean.correction=FALSE,
  objective.penalty = NULL, solve.tol = .Machine$double.eps,
  c.code = TRUE)
```

Arguments

`y`	numeric vector, typically a financial return series or the error of a regression
`arch`	the arch order (i.e. an integer equal to or greater than 0). The default is 1. NOTE: in the current version the order canno be greater than 1
`garch`	the garch order (i.e. an integer equal to or greater than 0). The default is 1. NOTE: in the current version the order canno be greater than 1
`xreg`	vector or matrix with conditioning variables
`initial.values`	NULL (default) or a vector with the initial values of the ARMA-representation
`lower`	NULL (default) or a vector with the lower bounds of the parameter space (of the ARMA-representation). If NULL, then the values are automatically chosen
`upper`	NULL (default) or a vector with the upper bounds of the parameter space (of the ARMA-representation). If NULL, then the values are automatically chosen
`nlminb.control`	list of control options passed on to the `nlminb` optimiser
`vcov`	logical. If TRUE (default), then the variance-covariance matrix is computed. The FALSE options makes estimation faster, but the variance-covariance matrix cannot be extracted subsequently

`method` Estimation method to use. Either "ls", i.e. Nonlinear Least Squares (default), "ml", i.e. Gaussian QML or "cex2", i.e. Centred exponential Chi-squared QML, see Francq and Sucarrat (2013). Note: For the cex2 method mean-correction = FALSE is not available

`mean.correction` Whether to mean-correct the ARMA representation. Mean-correction is usually faster, but not always recommended if covariates are added (i.e. if xreg is not NULL)

`objective.penalty` NULL (default) or a numeric value. If NULL, then the log-likelihood value associated with the initial values is used. Sometimes estimation can result in NA and/or +/-Inf values (this can be fatal for simulations). The value objective.penalty is the value returned by the objective function `lgarchObjective` in the presence of NA or +/-Inf values

`solve.tol` The function `solve` is used for the inversion of the negative of the Hessian in computing the variance-covariance matrix. The value solve.tol is passed on to `solve`, and is the tolerance for detecting linear dependencies in the columns

`c.code` logical. TRUE (default) is (much) faster, since it makes use of compiled C-code in the recursions

Value

A list of class 'lgarch'

Note

Empty

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

References

Francq, C. and G. Sucarrat (2013), 'An Exponential Chi-Squared QMLE for Log-GARCH Models via the ARMA Representation', MPRA Paper 51783: http://mpra.ub.uni-muenchen.de/51783/

Sucarrat and Escribano (2013), 'Unbiased QML Estimation of Log-GARCH Models in the Presence of Zero Returns', MPRA Paper 50699: http://mpra.ub.uni-muenchen.de/50699/

Sucarrat, Gronneberg and Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: http://mpra.ub.uni-muenchen.de/49344/

See Also

lgarchSim, coef.lgarch, fitted.lgarch, logLik.lgarch, print.lgarch, residuals.lgarch and vcov.lgarch

Examples

```
##simulate 500 observations w/default parameter values:
set.seed(123)
y <- lgarchSim(500)

##estimate a log-garch(1,1) w/least squares:
mymod <- lgarch(y)

##estimate the same model, but w/cex2 method:
mymod2 <- lgarch(y, method="cex2")

##print results:
print(mymod); print(mymod2)

##extract coefficients:
coef(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the log-garch model:
logLik(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the arma representation:
logLik(mymod, arma=TRUE)

##extract variance-covariance matrix:
vcov(mymod)

##extract and plot the fitted conditional standard deviation:
sdhat <- fitted(mymod)
plot(sdhat)

##extract and plot standardised residuals:
zhat <- residuals(mymod)
plot(zhat)

##extract and plot all the fitted series:
myhat <- fitted(mymod, verbose=TRUE)
plot(myhat)
```

---

lgarchObjective *Auxiliary functions*

Description

lgarchObjective and lgarchRecursion1 are auxiliary functions called by lgarch. The functions are not intended for the average user.

Usage

```
lgarchObjective(pars, aux)
lgarchRecursion1(pars, aux)
```

Arguments

pars	numeric vector with the parameters of the ARMA representation
aux	auxiliary list

Details

To understand the structure and content of pars and aux, see the source code of the lgarch function

Value

lgarchObjective returns the value of the objective function (either the log-likelihood or the residual sum of squares) used in estimating the ARMA representation. lgarchRecursion1 returns the residuals of the ARMA representation associated with the ARMA parameters pars

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

References

Francq, C. and G. Sucarrat (2013), 'An Exponential Chi-Squared QMLE for Log-GARCH Models via the ARMA Representation', MPRA Paper 51783: http://mpra.ub.uni-muenchen.de/51783/

Sucarrat, Gronneberg and Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: http://mpra.ub.uni-muenchen.de/49344/

See Also

lgarch

---

lgarchSim *Simulate from a univariate log-GARCH model*

Description

Simulate the y series (typically a financial return or the error in a regression) from a log-GARCH model. Optionally, the conditional standard deviation, the standardised error (z) and their logarithmic transformations are also returned.

Usage

```
lgarchSim(n, constant = 0, arch = 0.05, garch = 0.9, xreg = NULL,
  backcast.values = list(lnsigma2 = NULL, lnz2 = NULL, xreg = NULL),
  check.stability = TRUE, innovations = NULL, verbose = FALSE,
  c.code=TRUE)
```

Arguments

`n`	integer, length of y (i.e. number of observations)
`constant`	the value of the intercept in the log-volatility specification
`arch`	numeric vector with the arch coefficients
`garch`	numeric vector with the garch coefficients
`xreg`	numeric vector (of length n) with the conditioning values
`backcast.values`	backcast values for the recursion (chosen automatically if NULL)
`check.stability`	logical. If TRUE (default), then the roots of arch+garch are checked for stability
`innovations`	Etiher NULL (default) or a vector of length n with the standardised errors (i.e. z). If NULL, then the innovations are normal with mean zero and unit variance
`verbose`	logical. If FALSE (default), then only the vector y is returned. If TRUE, then a matrix with all the output is returned
`c.code`	logical. If TRUE (default), then compiled C-code is used for the recursion (faster)

Details

Empty

Value

A `zoo` vector of length n if verbose = FALSE (default), or a `zoo` matrix with n rows if verbose = TRUE.

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

References

Sucarrat, Gronneberg and Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: http://mpra.ub.uni-muenchen.de/49344/

See Also

`mlgarchSim`, `lgarch`, `mlgarch` and `zoo`

Examples

```
##simulate 500 observations w/default parameter values:
set.seed(123)
y <- lgarchSim(500)

##simulate the same series, but with more output:
set.seed(123)
y <- lgarchSim(500, verbose=TRUE)
head(y)

##plot the simulated values:
plot(y)

##simulate w/conditioning variable:
x <- rnorm(500)
y <- lgarchSim(500, xreg=0.05*x)

##simulate from a log-GARCH with a simple form of leverage:
z <- rnorm(500)
zneg <- as.numeric(z < 0)
zneglagged <- glag(zneg, pad=TRUE, pad.value=0)
y <- lgarchSim(500, xreg=0.05*zneglagged, innovations=z)

##simulate from a log-GARCH w/standardised t-innovations:
set.seed(123)
n <- 500
df <- 5
z <- rt(n, df=df)/sqrt(df/(df-2))
y <- lgarchSim(n, innovations=z)
```

---

mlgarch *Estimate a multivariate CCC-log-GARCH(1,1) model*

Description

Fit a multivariate Constant Conditional Correlation (CCC) log-GARCH(1,1) model with multivariate Gaussian Quasi Maximum Likelihood (QML) via the VARMA representation, see Sucarrat, Gronneberg and Escribano (2013). Zero-values on y are treated as missing values, as suggested in Sucarrat and Escribano (2013). Estimation is via the `nlminb` function, whereas a numerical estimate of the Hessian is obtained with `optimHess` for the computation of the variance-covariance matrix

Usage

```
mlgarch(y, arch = 1, garch = 1, xreg = NULL, initial.values = NULL,
  lower = NULL, upper = NULL, nlminb.control = list(), vcov = TRUE,
  objective.penalty = NULL, solve.tol = .Machine$double.eps, c.code = TRUE)
```

Arguments

y	a numeric matrix, typically financial returns or regression errors
arch	the arch order (i.e. an integer equal to or greater than 0). The default is 1. NOTE: in the current version the order cannot be greater than 1
garch	the garch order (i.e. an integer equal to or greater than 0). The default is 1. NOTE: in the current version the order cannot be greater than 1
xreg	a vector or a matrix with the conditioning variables. The x-variables enter in each of the equations
initial.values	NULL (default) or a vector with the initial values of the VARMA representation
lower	NULL (default) or a vector with the lower bounds of the parameter space (of the VARMA representation). If NULL, then the values are automatically chosen
upper	NULL (default) or a vector with the upper bounds of the parameter space (of the VARMA representation). If NULL, then the values are automatically chosen
nlminb.control	list of control options passed on to the `nlminb` optimiser
vcov	logical. If TRUE (default), then the variance-covariance matrix is computed. The FALSE options makes estimation faster, but the variance-covariance matrix cannot be extracted subsequently
objective.penalty	NULL (default) or a numeric value. If NULL, then the log-likelihood value associated with the initial values is used. Sometimes estimation can result in NA and/or +/-Inf values (this can be fatal for simulations). The value objective.penalty is the value returned by the log-likelihood function `lgarchObjective` in the presence of NA or +/-Inf values
solve.tol	The function `solve` is used for the inversion of the Hessian in computing the variance-covariance matrix. The value solve.tol is passed on to `solve`, and is the tolerance for detecting linear dependencies in the columns
c.code	logical. TRUE (default) is (much) faster, since it makes use of compiled C-code

Value

A list of class 'mlgarch'

Note

Empty

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

References

Sucarrat and Escribano (2013), 'Unbiased QML Estimation of Log-GARCH Models in the Presence of Zero Returns', MPRA Paper 50699: http://mpra.ub.uni-muenchen.de/50699/

Sucarrat, Gronneberg and Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: http://mpra.ub.uni-muenchen.de/49344/

See Also

lgarchSim, coef.lgarch, fitted.lgarch, logLik.lgarch, print.lgarch, residuals.lgarch and vcov.lgarch

Examples

```
##simulate 1000 observations from a 2-dimensional
##ccc-log-garch(1,1) w/default parameter values:
set.seed(123)
y <- mlgarchSim(1000)

##estimate a 2-dimensional ccc-log-garch(1,1):
mymod <- mlgarch(y)

##print results:
print(mymod)

##extract ccc-log-garch coefficients:
coef(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the ccc-log-garch model:
logLik(mymod)

##extract Gaussian log-likelihood (zeros excluded) of the varma representation:
logLik(mymod, varma=TRUE)

##extract variance-covariance matrix:
vcov(mymod)

##extract and plot the fitted conditional standard deviations:
sdhat <- fitted(mymod)
plot(sdhat)

##extract and plot standardised residuals:
zhat <- residuals(mymod)
plot(zhat)
```

---

mlgarchObjective *Auxiliary functions*

Description

mlgarchObjective and mlgarchRecursion1 are auxiliary functions called by mlgarch. The functions are not intended for the average user.

Usage

```
mlgarchObjective(pars, aux)
mlgarchRecursion1(pars, aux)
```

Arguments

pars	numeric vector of VARMA parameters
aux	auxiliary list

Details

To understand the structure and content of pars and aux, see the source code of the `mlgarch` function

Value

mlgarchObjective returns the log-likelihood of the VARMA representation, whereas mlgarchRecursion1 returns the residuals of the VARMA representation associated with the VARMA parameters pars

Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

References

Sucarrat, Gronneberg and Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: http://mpra.ub.uni-muenchen.de/49344/

See Also

`mlgarch` and `lgarchObjective`

---

mlgarchSim *Simulate from a multivariate log-GARCH(1,1) model*

Description

Simulate the y series (typically a collection of financial returns or regression errors) from a log-GARCH model. Optionally, the conditional standard deviation and the standardised error, together with their logarithmic transformations, are also returned.

Usage

```
mlgarchSim(n, constant = c(0,0), arch = diag(c(0.1, 0.05)),
  garch = diag(c(0.7, 0.8)), xreg = NULL,
  backcast.values = list(lnsigma2 = NULL, lnz2 = NULL, xreg = NULL),
  innovations = NULL, innovations.vcov = diag(rep(1,
  length(constant))), check.stability = TRUE, verbose = FALSE)
```

Arguments

`n`	integer, i.e. number of observations
`constant`	vector with the values of the intercepts in the log-volatility specification
`arch`	matrix with the arch coefficients
`garch`	matrix with the garch coefficients
`xreg`	a vector (of length n) or matrix (with rows n) with the values of the conditioning variables. The first column enters the first equation, the second enters the second equation, and so on
`backcast.values`	backcast values for the recursion (chosen automatically if NULL)
`check.stability`	logical. If TRUE (default), then the system is checked for stability
`innovations`	Either NULL (default) or a vector or matrix of length n with the standardised errors. If NULL, then the innovations are multivariate N(0,1) with correlations equal to zero
`innovations.vcov`	numeric matrix, the variance-covariance matrix of the standardised multivariate normal innovations. Only applicable if innovations = NULL
`verbose`	logical. If FALSE (default), then only the matrix with the y series is returned. If TRUE, then also additional information is returned

Details

Empty

Value

A `zoo` matrix with n rows.

Author(s)

Genaro Sucarrat, `http://www.sucarrat.net/`

References

Sucarrat, Gronneberg and Escribano (2013), 'Estimation and Inference in Univariate and Multivariate Log-GARCH-X Models When the Conditional Density is Unknown', MPRA Paper 49344: `http://mpra.ub.uni-muenchen.de/49344/`

See Also

`lgarchSim`, `mlgarch` and `zoo`

Examples

```
##simulate 1000 observations from a multivariate
##ccc-log-garch(1,1) w/default parameter values:
set.seed(123)
y <- mlgarchSim(1000)

##simulate the same series, but with more output:
set.seed(123)
y <- mlgarchSim(1000, verbose=TRUE)
head(y)

##plot the simulated values:
plot(y)
```

`rmnorm` *Random number generation from the multivariate normal distribution*

Description

This function is a speed-optimised version of the `rmnorm` function from the mnormt package of Adelchi Azzalini (2013).

Usage

```
rmnorm(n, mean = NULL, vcov = 1)
```

Arguments

`n`	integer, the number of observations to generate
`mean`	numeric vector, i.e. the mean values
`vcov`	numeric matrix, i.e. the variance-covariance matrix

Details

Empty

Value

A matrix of n rows

Author(s)

Genaro Sucarrat, `http://www.sucarrat.net/`

References

Adelchi Azzalini (2013): 'mnormt: The multivariate normal and t distributions', R package version 1.4-7, `http://CRAN.R-project.org/package=mnormt`

Examples

```
##generate from univariate standardised normal:
z1 <- rmnorm(100)

##generate from bivariate, independent standardised normal:
z2 <- rmnorm(100, vcov=diag(c(1,1)))

##generate from bivariate, dependent standardised normal:
z3 <- rmnorm(100, vcov=cbind(c(1,0.3),c(0.3,1)))
```

Index