The fGarch Package

October 8, 2007

Version 260.72

Date 1997 - 2007

Title Rmetrics - Autoregressive Conditional Heteroskedastic Modelling

Author Diethelm Wuertz and many others, see the SOURCE file

Depends R (>= 2.4.0), fBasics, fArma

Maintainer Diethelm Wuertz and Rmetrics Core Team <Rmetrics-core@r-project.org>

Description Environment for teaching "Financial Engineering and Computational Finance"

NOTE SEVERAL PARTS ARE STILL PRELIMINARY AND MAY BE CHANGED IN THE FUTURE. THIS TYPICALLY INCLUDES FUNCTION AND ARGUMENT NAMES, AS WELL AS DEFAULTS FOR ARGUMENTS AND RETURN VALUES.

LazyLoad yes

LazyData yes

License GPL Version 2 or later

URL http://www.rmetrics.org

R topics documented:

GarchDistributions 2
garchExtractors 5
GarchFitting 6
GarchOxInterface 9
garchPredictor 12
garchReports 13
garchSim 14
garchSpec 15

Index **18**

GarchDistributions *GARCH Distributions and Parameter Estimation*

Description

A collection and description of functions to compute density, distribution function, quantile function and to generate random variates for the skew normal, the skew Student-t, and skew generalized error distribution. In addition maximum likelihood estimators are available to fit the parameters of a distribution and to compute basic statistical properties.

The functions are:

`[dpqr]norm`	Normal distribution from R's base package,
`[dpqr]snorm`	Skew Normal distribution,
`[dpqr]std`	Symmetric Student-t Distribution,
`[dpqr]sstd`	Skew Student-t Distribution,
`[dpqr]ged`	Symmetric GED distribution,
`[dpqr]sged`	Skew GED distribution.

The estimatore are:

`normFit`	MLE parameter fit for a Normal distribution,
`snormFit`	MLE parameter fit for a skew Normal distribution,
`stdFit`	MLE parameter fit for a Sudent-t distribution,
`stdFit`	MLE parameter fit for a skew Sudent-t distribution,
`gedFit`	MLE parameter fit for a generalized error distribution,
`nigFit`	MLE parameter fit for a skew generalized error distribution.

Utility Function:

`absMoments`	Computes absolute moments of a symmetric density.

Usage

```
dsnorm(x, mean = 0, sd = 1, xi = 1.5)
psnorm(q, mean = 0, sd = 1, xi = 1.5)
qsnorm(p, mean = 0, sd = 1, xi = 1.5)
rsnorm(n, mean = 0, sd = 1, xi = 1.5)

dstd(x, mean = 0, sd = 1, nu = 5)
pstd(q, mean = 0, sd = 1, nu = 5)
qstd(p, mean = 0, sd = 1, nu = 5)
rstd(n, mean = 0, sd = 1, nu = 5)

dsstd(x, mean = 0, sd = 1, nu = 5, xi = 1.5)
psstd(q, mean = 0, sd = 1, nu = 5, xi = 1.5)
```

```
qsstd(p, mean = 0, sd = 1, nu = 5, xi = 1.5)
rsstd(n, mean = 0, sd = 1, nu = 5, xi = 1.5)

dged(x, mean = 0, sd = 1, nu = 2)
pged(q, mean = 0, sd = 1, nu = 2)
qged(p, mean = 0, sd = 1, nu = 2)
rged(n, mean = 0, sd = 1, nu = 2)

dsged(x, mean = 0, sd = 1, nu = 2, xi = 1.5)
psged(q, mean = 0, sd = 1, nu = 2, xi = 1.5)
qsged(p, mean = 0, sd = 1, nu = 2, xi = 1.5)
rsged(n, mean = 0, sd = 1, nu = 2, xi = 1.5)

normFit(x, ...)
snormFit(x, ...)
stdFit(x, ...)
sstdFit(x, ...)
gedFit(x, ...)
sgedFit(x, ...)

absMoments(n, density = c("dnorm", "dged", "dstd"), ...)
```

Arguments

`density`	[absMoments] - a character string naming the symmetric density function.
`mean, sd, nu, xi`	location parameter `mean`, scale parameter `sd`, shape parameter `nu`, skewness parameter `xi`.
`n`	[rnorm][r*ged][r*std] - the number of observations. [absMoments] - the number of absolute Moments.
`p`	a numeric vector of probabilities.
`x, q`	a numeric vector of quantiles.
`...`	[*Fit] - parameters parsed to the optimization function `nlm`. [absMoments] - parameters passed to the density function.

Details

Symmetric Normal Distibution:

The functions for the normal distribution are part of R's base package. The functions for the symmetric Student-t distribution are rescaled in such a way that they have unit variance in contrast to the Student-t family `dt`, `pt`, `qt` and `rt` which are part of R's base package. The generalized error

distribution functions are defined as described by Nelson (1991).

Skew Normal Distribution:

The skew normal distribution functions are defined as described by Fernandez and Steel (2000). cr

Parameter Estimation:

The function `nlm` is used to minimize the "negative" maximum log-likelihood function. `nlm` carries out a minimization using a Newton-type algorithm.

Value

`d*` returns the density, `p*` returns the distribution function, `q*` returns the quantile function, and `r*` generates random deviates,
all values are numeric vectors.

`*Fit` return a list with the following components:

`estimate`	the point at which the maximum value of the log liklihood function is obtained.
`objective`	the value of the estimated maximum, i.e. the value of the log liklihood function.
`message`	an integer indicating why the optimization process terminated.
`code`	an integer indicating why the optimization process terminated. 1: relative gradient is close to zero, current iterate is probably solution; 2: successive iterates within tolerance, current iterate is probably solution; 3: last global step failed to locate a point lower than `estimate`. Either `estimate` is an approximate local minimum of the function or `steptol` is too small; 4: iteration limit exceeded; 5: maximum step size `stepmax` exceeded five consecutive times. Either the function is unbounded below, becomes asymptotic to a finite value from above in some direction or `stepmax` is too small.
`gradient`	the gradient at the estimated maximum.
`steps`	number of function calls.

`absMoments` returns a numeric vector of length `n` with the values of the absolute moments of the density function.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

References

Nelson D.B. (1991); *Conditional Heteroscedasticity in Asset Returns: A New Approach*, Econometrica, 59, 347–370.

Fernandez C., Steel M.F.J. (2000); *On Bayesian Modelling of Fat Tails and Skewness*, Preprint, 31 pages.

Examples

```
## sged -
   par(mfrow = c(2, 2), cex = 0.75)
   set.seed(1953)
   r = rsged(n = 1000, mean = 1, sd = 0.5, xi = 1.5)
   plot(r, type = "l", main = "sged: xi = 1.5")
   # Plot empirical density and compare with true density:
   hist(r, n = 25, probability = TRUE, border = "white", col = "steelblue")
   x = seq(-1, 5, 0.1)
   lines(x, dsged(x = x, mean = 1, sd = 0.5, xi = 1.5))
   # Plot df and compare with true df:
   plot(sort(r), (1:1000/1000), main = "Probability", col = "steelblue")
   lines(x, psged(x, mean = 1, sd = 0.5, xi = 1.5))
   # Compute quantiles:
   qsged(psged(q = -1:5, mean = 1, sd = 0.5, xi = 1.5),
     mean = 1, sd = 0.5, xi = 1.5)

## sgedFit -
   sgedFit(rsged(1000, mean = -1, sd = 0.5, nu = 3, xi = 3/2),
     print.level = 2)
```

---

garchExtractors *GARCH Extractor Functions*

Description

Extracts residuals and fitted values from a fitted GARCH object.

Usage

```
## S3 method for class 'fGARCH':
residuals(object, ...)
## S3 method for class 'fGARCH':
fitted(object, ...)
```

Arguments

`object`	an object of class `fGARCH` as returned from the function `garchFit()`.
`...`	additional arguments to be passed.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```
## garchSpec -
   spec = garchSpec()
   spec

## garchSim -
   x = garchSim(model = spec@model, n = 500)
   head(x)

## garchFit -
   # fit = garchFit(~garch(1, 1), data = x)
   # print(fit)
   ## Interactive Plot:
   ## plot(fit)
   ## Batch Plot:
   # plot(fit, which = 3)
   # summary(fit)
```

GarchFitting *Univariate GARCH Time Series Fitting*

Description

Estimates the parameters of an univariate GARCH process.

Usage

```
garchFit(formula, data, init.rec = c("mci", "uev"), delta = 2, skew = 1,
    shape = 4, cond.dist = c("dnorm", "dsnorm", "dged", "dsged", "dstd", "dsstd"),
    include.mean = TRUE, include.delta = NULL, include.skew = NULL,
    include.shape = NULL, leverage = NULL, trace = TRUE,
    algorithm = c("nlminb", "sqp", "lbfgsb", "nlminb+nm", "lbfgsb+nm"),
    control = list(), title = NULL, description = NULL, ...)

garchKappa(cond.dist = c("dnorm", "dged", "dstd", "dsnorm", "dsged", "dsstd"),
    gamma = 0, delta = 2, skew = NA, shape = NA)
```

Arguments

`algorithm`	a string parameter that determines the algorithm used for maximum likelihood estimation. Allowed values are `"sqp"`, `"nlminb"`, and `"bfgs"` where the first is the default setting.
`cond.dist`	a character string naming the desired conditional distribution. Valid values are `"dnorm"`, `"dged"`, `"dstd"`, `"dsnorm"`, `"dsged"`, `"dsstd"`. The default value is the normal distribution.
`control`	control parameters, the same as used for the functions from `nlminb`, and 'bfgs' and 'Nelder-Mead' from `optim`.

data — an optional timeSeries or data frame object containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which armaFit is called. If data is an univariate series, then the series is converted into a numeric vector and the name of the response in the formula will be neglected.

delta, include.delta — the exponent delta of the variance recursion. By default, this value will be fixed, otherwise the exponent will be estimated together with the other model parameters if include.delta=FALSE.

description — a character string which allows for a brief description.

formula — formula object describing the mean and variance equation of the ARMA-GARCH/APARCH model. A pure GARCH(1,1) model is selected when e.g. formula=~garch(1,1). To specify for example an ARMA(2,1)-APARCH(1,1) use formula = ~arma(2,1)+apaarch(1,

gamma — APARCH leverage parameter entering into the formula for calculating the expectation value.

include.mean — this flag determines if the parameter for the mean will be estimated or not. If include.mean=TRUE this will be the case, otherwise the parameter will be kept fixed durcing the process of parameter optimization.

include.skew, include.shape — this flag determines if the parameters for the skew and shape of the conditional distribution will be estimated or not. If include.skew=TRUE and/or include.shape=TRUE this will be the case, otherwise the parameters will be kept fixed durcing the process of parameter optimization.

init.rec — a character string indicating the method how to initialize the mean and varaince recursion relation.

leverage — a logical flag for APARCH models. Should the model be leveraged? By default leverage=TRUE.

skew, shape — skewness and shape parameter of the conditional distribution.

title — a character string which allows for a project title.

trace — a logical flag. Should the optimization process of fitting the model parameters be printed? By default trace=TRUE.

... — additional arguments to be passed.

Value

garchFit

returns a S4 object of class fGARCH with the following slots:

@call — the call of the garch function.

@formula — a list with two formula entries, one for the mean and the other one for the variance equation.

@method — a string denoting the optimization method, by default the returneds string is "Max Log-Likelihood Estimation".

`@data`	a list with one entry named `x`, containing the data of the time series to be estimated, the same as given by the input argument `series`.
`@fit`	a list with the results from the parameter estimation. The entries of the list depend on the selected algorithm, see below.
`@residuals`	a numeric vector with the residual values.
`@fitted`	a numeric vector with the fitted values.
`@h.t`	a numeric vector with the conditional variances.
`@sigma.t`	a numeric vector with the conditional variances.
`@title`	a title string.
`@description`	a string with a brief description.

The entries of the @fit slot show the results from the optimization.

Author(s)

Diethelm Wuertz for the Rmetrics R-port,
R Core Team for the 'optim' R-port,
Douglas Bates and Deepayan Sarkar for the 'nlminb' R-port,
Bell-Labs for the underlying PORT Library,
Ladislav Luksan for the underlying Fortran SQP Routine,
Zhu, Byrd, Lu-Chen and Nocedal for the underlying L-BFGS-B Routine.

References

ATT (1984); *PORT Library Documentation*, http://netlib.bell-labs.com/netlib/port/.

Bera A.K., Higgins M.L. (1993); *ARCH Models: Properties, Estimation and Testing*, J. Economic Surveys 7, 305–362.

Bollerslev T. (1986); *Generalized Autoregressive Conditional Heteroscedasticity*, Journal of Econometrics 31, 307–327.

Byrd R.H., Lu P., Nocedal J., Zhu C. (1995); *A Limited Memory Algorithm for Bound Constrained Optimization*, SIAM Journal of Scientific Computing 16, 1190–1208.

Engle R.F. (1982); *Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation*, Econometrica 50, 987–1008.

Nash J.C. (1990); *Compact Numerical Methods for Computers*, Linear Algebra and Function Minimisation, Adam Hilger.

Nelder J.A., Mead R. (1965); *A Simplex Algorithm for Function Minimization*, Computer Journal 7, 308–313.

Nocedal J., Wright S.J. (1999); *Numerical Optimization*, Springer, New York.

Examples

```
## garchSpec -
   spec = garchSpec()
   spec
```

```
## garchSim -
   x = garchSim(model = spec@model, n = 500)
   head(x)

## garchFit -
   # fit = garchFit(~garch(1, 1), data = x)
   # print(fit)
   ## Interactive Plot:
   ## plot(fit)
   ## Batch Plot:
   # plot(fit, which = 3)
   # summary(fit)
```

`GarchOxInterface` *R Interface for Garch Ox*

Description

A collection and description of functions to fit the parameters of an univariate time series to GARCH models interfacing the G@RCH Ox Package.

The family of GARCH time series models includes the following processes:

1	garch	generalized AR conditional heteroskedastic models,
2	egarch	exponential GARCH models,
3	aparch	asymmetretic power ARCH models.

Usage

```
garchOxFit(formula, data, cond.dist = c("gaussian", "t", "ged", "skewed-t"),
    include.mean = TRUE, trace = TRUE, control = list(), title = NULL,
    description = NULL)

## S3 method for class 'garchOx':
print(x, digits, ...)
## S3 method for class 'garchOx':
summary(object, ...)
## S3 method for class 'garchOx':
plot(x, ...)
```

Arguments

`cond.dist`	a character string describing the distribution of innovations. By default the optimization is based on gaussian log likelihood parameter optimization denoted by "gaussian". Alternatively, a Student-t "t", a generalized error "sged", or a skewed Student-t "skewed-t" can be chosen.
`control`	a list of additional control parameters:

	truncation - the number of truncation points,by default 100, xscale - should the time series be scaled by the standard deviation ?
data	an optional timeSeries or data frame object containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which armaFit is called. If data is an univariate series, then the series is converted into a numeric vector and the name of the response in the formula will be neglected.
description	a character string which allows for a brief description.
digits	the number of digits to be printed.
formula	[garchFit] - formula object describing the mean and variance equation of the ARMA-GARCH/APARCH model. A pure GARCH(1,1) model is selected when e.g. formula=~garch(1,1). To specify for example an ARMA(2,1)-APARCH(1,1) use formula = ~arma(2,1)+apaarch(1,
include.mean	should the mean be included? By default TRUE.
object	an object of class garchOx as returned from the function codegarchOxFit.
title	a character string which allows for a project title.
trace	a logical flag. Should the estimation process be ttraced? By default TRUE.
x	an object of class garchOx as returned from the function garchOxFit.
...	additional arguments to be passed to the print, summary, and plot methods.

Details

Ox Interface:

The function garchOxFit interfaces a subset of the functionality of the G@ARCH 4.0 Package written in Ox. G@RCH 4.0 is one of the most sophisticated packages for modelling univariate GARCH processes including GARCH, EGARCH, GJR, APARCH, IGARCH, FIGARCH, FIEGARCH, FIAPARCH and HYGARCH models. Parameters can be estimated by approximate (Quasi-) maximum likelihood methods under four assumptions: normal, Student-t, GED or skewed Student-t errors.

About Ox:

Ox (tm) is an object-oriented matrix language with a comprehensive mathematical and statistical function library. Many packages were written for Ox including software mainly for econometric modelling. The Ox packages for time series analysis and forecasting, Arfima, Doornik and Ooms [2003], Garch, Laurent and Peters [2005], and State Space Modelling, Koopman, Shepard and Doornik [1998], are especially worth to note. Since most of the R-users wan't to change to another Statistical Computing environment, we made selected parts of the G@RCH Ox software available for them through an R-Interface. What you have to do, is to read carefully the "Ox citation and copyright" rules and if you agree and fullfill the conditions, then download the Ox-Console Software together with the "OxGarch" Package, currently G@RCH 4.0. If you are not qualified for a free license, order your copy from Timberlake Consultants. We recommend to install the "Setup.exe" under the path "C:\Ox\Ox3" and to unzip the OxGarch Package in the directory "C:\Ox\Ox3\Packages". An Update to Ox4 has not yet be done.

Distribution:
Ox and G@RCH are distributed by Timberlake Consultants Ltd. Timberlake Consultants can be contacted through the following web site: *www.timberlake.co.uk*.

Installation of the Interface:
In addition you have to copy the file "GarchOxModelling.ox" (which is the interface written especially for Rmetrics) from the "fSeries/ox/" directory to the Ox library directory "C:\Ox\lib".

Ox Citation and Copyright Rules:
Ox and all its components are copyright of Jurgen A. Doornik. The Console (command line) versions may be used freely for academic research and teaching purposes only. Commercial users and others who do not qualify for the free version must purchase the Windows version of Ox and GiveWin with documentation, regardless of which version they use (so even when only using Ox on Linux or Unix). Ox should be cited whenever it is used. Refer to the two references given below. Note, failure to cite the use of Ox in published work may result in loss of the right to use the free version, and an invoice at the full commercial price. Ox is available from Timberlake Consultants. The Ox syntax is public, and you may do with your own Ox code whatever you wish, including the file "GarchOxModelling.ox".

Work to do:
Note, only a small part of the functionalities are interfaced until now to R. But, principally it would be possible to interface also other functionalities offered by the Ox Garch Package. This work is left to the Ox/Rmetrics user.

Author(s)

Jurgen A. Doormik for the Ox Environment, *www.doornik.com*,
Sebastian Laurent for the Ox Garch package, *www.garch.org*,
Diethelm Wuertz for R's Ox Garch interface.

References

Doornik J.A. (2002), Object-Oriented Matrix Programming Using Ox, London, 3rd ed.: Timberlake Consultants Press and Oxford: *www.doornik.com*.

Doornik J.A., Ooms M. (2003), Computational Aspects of Maximum Likelihood Estimation of Autoregressive Fractionally Integrated Moving Average Models, Computational Statistics and Data Analysis 42, 333–348.

Koopman J.S., Shepard N., Doornik J.A. (1999), Statistical Algorithms for Models in State Space using SsfPack 2.2, Econometrics Journal 2, 113–166.

Laurent S., Peters J.P. (2002); G@RCH 2.2: An Ox Package for Estimating and Forecasting Various ARCH Models, Journal of Economic Surveys 16, 447–485.

Laurent S., Peters J.P., [2005], G@RCH 4.0, Estimating and Forecasting ARCH Models, Timberlake Consultants, www.timberlake.co.uk

Examples

```
## Not run:
```

```
## Load Benchmark Data Set:
   data(dem2gbp)
   x = dem2gbp[, 1]

## garchOxFit -
   # Fit GARCH(1,1):
   garchOxFit(formula = ~arma(0,0) + ~garch(1,1))
## End(Not run)
```

`garchPredictor` *GARCH Prediction Function*

Description

Predicts a time series from a fitted GARCH object.

Usage

```
## S3 method for class 'fGARCH':
predict(object, n.ahead = 10, trace = FALSE, ...)
```

Arguments

`n.ahead`	number of steps to be forecasted, an integer value, by default 10.
`object`	an object of class `fGARCH` as returned from the function `garchFit()`.
`trace`	a logical flag. Should the prediction process be printed? By default `trace=FALSE`.
`...`	additional arguments to be passed.

Value

returns ...

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```
## garchSpec -
   spec = garchSpec()
   spec

## garchSim -
   x = garchSim(model = spec@model, n = 500)
   head(x)

## garchFit -
   # fit = garchFit(~garch(1, 1), data = x)
```

```
   # print(fit)
   ## Interactive Plot:
   ## plot(fit)
   ## Batch Plot:
   # plot(fit, which = 3)
   # summary(fit)

## predict -
   # predict(object, n.ahead = 10)
```

`garchReports` *GARCH Reports and Graphs*

Description

Creates reports and graphs of a GARCH modelling process.

Usage

```
show.fGARCH(object)

## S3 method for class 'fGARCH':
plot(x, which = "ask", ...)
## S3 method for class 'fGARCH':
summary(object, ...)
```

Arguments

`object, x`	an object of class `fGARCH` as returned from the function `garchFit()`.
`which`	if `which` is set to `"ask"` the function will interactively ask which plot should be displayed. This is the default value and then those plots will be displayed for which the elements in the logical vector `which` ar set to `TRUE`; by default all four elements are set to `"all"`.
`...`	additional arguments to be passed.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```
## garchSpec -
   spec = garchSpec()
   spec

## garchSim -
   x = garchSim(model = spec@model, n = 500)
   head(x)
```

```
## garchFit -
   # fit = garchFit(~garch(1, 1), data = x)
   # print(fit)
   ## Interactive Plot:
   ## plot(fit)
   ## Batch Plot:
   # plot(fit, which = 3)
   # summary(fit)
```

`garchSim` *Univariate GARCH Time Series Simulation*

Description

Simulates an univariate GARCH time series model.

Usage

```
garchSim(model = list(omega = 1.0e-6, alpha = 0.1, beta = 0.8), n = 100,
    n.start = 100, presample = NULL, cond.dist = c("rnorm", "rged", "rstd",
    "rsnorm", "rsged", "rsstd"), rseed = NULL)
```

Arguments

`cond.dist`	a character string naming the desired conditional distribution. Valid values are `"dnorm"`, `"dged"`, `"dstd"`, `"dsnorm"`, `"dsged"`, `"dsstd"`. The default value is the normal distribution.
`model`	a list of GARCH model parameters: `omega` - the constant coefficient of the variance equation, by default 1e-6; `alpha` - the value or vector of autoregressive coefficients, by default 0.1, specifying a model of order 1; `beta` - the value or vector of variance coefficients, by default 0.8, specifying a model of order 1; The optional values for the linear part are: `mu` - the mean value, by default 0; `ar` - the autoregressive ARMA coefficients, by default 0; `ma` - the moving average ARMA coefficients, by default 0. The optional parameters for the conditional distributions are: `skew` - the skewness parameter (also named xi), by default 0.9, effective only for the `"dsnorm"`, the `"dsged"`, and the `"dsstd"` skewed conditional distributions; `shape` = the shape parameter (also named nu), by default 2 for the `"dged"` and `"dsged"`, and by default 4 for the `"dstd"` and `"dsstd"` conditional distributions. Note, the default model specifies Bollerslev's GARCH(1,1) model with normal distributed innovations.

n	length of output series, an integer value. An integer value, by default n=100.
n.start	length of "burn-in" period, by default 100.
presample	a numeric three column matrix with start values for the series, for the innovations, and for the conditional variances. For an ARMA(m,n)-GARCH(p,q) process the number of rows must be at least max(m,n,p,q), longer presamples are cutted.
rseed	single integer argument, the seed for the intitialization of the random number generator for the innovations.

Value

returns an objects of class ts atrributed with an appropriate specification structure as returned by the function garchSpec.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```
## garchSpec -
   spec = garchSpec()
   spec

## garchSim -
   x = garchSim(model = spec@model, n = 500)
   head(x)
```

---

garchSpec *Univariate GARCH Time Series Specification*

Description

Specifies an univariate GARCH time series model.

Usage

```
garchSpec(model = list(omega = 1.0e-6, alpha = 0.1, beta = 0.8),
    presample = NULL, cond.dist = c("rnorm", "rged", "rstd", "rsnorm",
    "rsged", "rsstd"), rseed = NULL)

show.garchSpec(object)
```

Arguments

cond.dist	a character string naming the desired conditional distribution. Valid values are "dnorm", "dged", "dstd", "dsnorm", "dsged", "dsstd". The default value is the normal distribution.
model	a list of GARCH model parameters: omega - the constant coefficient of the variance equation, by default 1e-6; alpha - the value or vector of autoregressive coefficients, by default 0.1, specifying a model of order 1; beta - the value or vector of variance coefficients, by default 0.8, specifying a model of order 1; The optional values for the linear part are: mu - the mean value, by default 0; ar - the autoregressive ARMA coefficients, by default 0; ma - the moving average ARMA coefficients, by default 0. The optional parameters for the conditional distributions are: skew - the skewness parameter (also named xi), by default 0.9, effective only for the "dsnorm", the "dsged", and the "dsstd" skewed conditional distributions; shape = the shape parameter (also named nu), by default 2 for the "dged" and "dsged", and by default 4 for the "dstd" and "dsstd" conditional distributions. Note, the default model specifies Bollerslev's GARCH(1,1) model with normal distributed innovations.
object	an object of class garchSpec as returned from the function garchSpec().
presample	a numeric three column matrix with start values for the series, for the innovations, and for the conditional variances. For an ARMA(m,n)-GARCH(p,q) process the number of rows must be at least max(m,n,p,q), longer presamples are cutted.
rseed	single integer argument, the seed for the intitialization of the random number generator for the innovations.

Value

garchSpec

returns a S4 object of class garchSpec with the following slots:

@call	the call of the garch function.
@formula	a list with two formula entries for the mean and variance equation.
@model	a list with the model parameters.
@presample	a numeric matrix with presample values.
@distribution	a character string with the name of the conditional distribution.
@rseed	an integer with the random number generator seed.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```
## garchSpec -

   # Normal Conditional Distribution:
   spec = garchSpec()
   spec

   # Skewed Normal Conditional Distribution:
   spec = garchSpec(model = list(skew = 0.8), cond.dist = "rsnorm")
   spec

   # Skewed GED Conditional Distribution:
   spec = garchSpec(model = list(skew = 0.9, shape = 4.8), cond.dist = "rsged")
   spec
```

Index

∗Topic **distribution**
GarchDistributions, 1
∗Topic **models**
garchExtractors, 5
GarchFitting, 6
GarchOxInterface, 9
garchPredictor, 12
garchReports, 13
garchSim, 14
garchSpec, 15

absMoments *(GarchDistributions)*, 1

dged *(GarchDistributions)*, 1
dsged *(GarchDistributions)*, 1
dsnorm *(GarchDistributions)*, 1
dsstd *(GarchDistributions)*, 1
dstd *(GarchDistributions)*, 1

fGARCH-class *(garchReports)*, 13
fitted.fGARCH *(garchExtractors)*, 5

GarchDistributions, 1
garchExtractors, 5
garchFit *(GarchFitting)*, 6
GarchFitting, 6
garchKappa *(GarchFitting)*, 6
garchOxFit *(GarchOxInterface)*, 9
GarchOxInterface, 9
garchPredictor, 12
garchReports, 13
garchSim, 14
garchSpec, 15
garchSpec-class *(garchSpec)*, 15
gedFit *(GarchDistributions)*, 1

nlm, *3*
normFit *(GarchDistributions)*, 1

pged *(GarchDistributions)*, 1
plot.fGARCH *(garchReports)*, 13
plot.garchOx *(GarchOxInterface)*, 9
predict.fGARCH *(garchPredictor)*, 12
print.garchOx *(GarchOxInterface)*, 9
psged *(GarchDistributions)*, 1
psnorm *(GarchDistributions)*, 1
psstd *(GarchDistributions)*, 1
pstd *(GarchDistributions)*, 1

qged *(GarchDistributions)*, 1
qsged *(GarchDistributions)*, 1
qsnorm *(GarchDistributions)*, 1
qsstd *(GarchDistributions)*, 1
qstd *(GarchDistributions)*, 1

residuals.fGARCH *(garchExtractors)*, 5
rged *(GarchDistributions)*, 1
rsged *(GarchDistributions)*, 1
rsnorm *(GarchDistributions)*, 1
rsstd *(GarchDistributions)*, 1
rstd *(GarchDistributions)*, 1

sgedFit *(GarchDistributions)*, 1
show,fGARCH-method *(garchReports)*, 13
show,garchSpec-method *(garchSpec)*, 15
show.fGARCH *(garchReports)*, 13
show.garchSpec *(garchSpec)*, 15
snormFit *(GarchDistributions)*, 1
sstdFit *(GarchDistributions)*, 1
stdFit *(GarchDistributions)*, 1
summary.fGARCH *(garchReports)*, 13
summary.garchOx *(GarchOxInterface)*, 9