Package ‘betategarch’

February 19, 2015

Type Package

Title Simulation, estimation and forecasting of Beta-Skew-t-EGARCH models

Version 3.2

Depends R (>= 2.15.0), zoo

Date 2014-09-18

Author Genaro Sucarrat

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

URL http://www.sucarrat.net/

Description Simulation, estimation and forecasting of first-order Beta-Skew-t-EGARCH models with leverage (one-component, two-component, skewed versions).

License GPL-2

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-09-18 17:28:18

R topics documented:

betategarch-package .............................................. 2
coeff.tegarch ..................................................... 3
dST ................................................................. 6
nasdaq ............................................................ 7
predict.tegarch .................................................. 8
tegarch ............................................................ 10
tegarchLogl ...................................................... 12
tegarchSim ....................................................... 14

Index 17
Description

This package provides facilities for the simulation, estimation and forecasting of first order Beta-Skew-t-EGARCH models with leverage (one-component and two-component versions), see Harvey and Sucarrat (2013).

Let y[t] denote a financial return at time t equal to

\[ y[t] = \sigma[t] \epsilon[t] \]

where \( \sigma[t] > 0 \) is the scale or volatility (generally not equal to the conditional standard deviation), and where \( \epsilon[t] \) is IID and t-distributed (possibly skewed) with df degrees of freedom. Then the first order log-volatility specification of the one-component Beta-Skew-t-EGARCH model can be parametrised as

\[
\begin{align*}
\sigma[t] &= \exp(\lambda[t]), \\
\lambda[t] &= \omega + \lambda\dagger, \\
\lambda\dagger[t] &= \phi_1 \lambda\dagger[t-1] + \kappa_1 u[t-1] + \kappa_{\ast} \text{sign}[-y](u[t-1]+1).
\end{align*}
\]

So the scale or volatility is given by \( \sigma[t] = \exp(\lambda[t]) \). The omega is the unconditional or long-term log-volatility, \( \phi_1 \) is the GARCH parameter (|\( \phi_1 | < 1 \) implies stability), \( \kappa_1 \) is the ARCH parameter, \( \kappa_{\ast} \) is the leverage or volatility-asymmetry parameter and \( u[t] \) is the conditional score or first derivative of the log-likelihood with respect to \( \lambda \). The score \( u[t] \) is zero-mean and IID, and \( (u[t]+1)/(df+1) \) is Beta distributed when there is no skew in the conditional density of \( \epsilon[t] \). The two-component specification is given by

\[
\begin{align*}
\sigma[t] &= \exp(\lambda[t]), \\
\lambda[t] &= \omega + \lambda_1\dagger + \lambda_2\dagger, \\
\lambda_1\dagger[t] &= \phi_1 \lambda\dagger[t-1] + \kappa_1 u[t-1], \\
\lambda_2\dagger[t] &= \phi_2 \lambda\dagger[t-1] + \kappa_2 u[t-1] + \kappa_{\ast} \text{sign}[-y](u[t-1]+1).
\end{align*}
\]

The first component, \( \lambda_1\dagger \), is interpreted as the long-term component, whereas the second component, \( \lambda_2\dagger \), is interpreted as the short-term component.

Details

| Package: | betategarch |
| Type: | Package |
| Version: | 3.2 |
| Date: | 2014-09-18 |
| License: | GPL-2 |
| LazyLoad: | yes |

The two main functions of the package are tegarchSim and tegarch. The first simulates a Beta-Skew-t-EGARCH models whereas the second estimates one. The second object returns an object (a
list) of class ‘tegarch’, and a collection of methods can be applied to this class: \texttt{coef.tegarch}, \texttt{fitted.tegarch}, \texttt{logLik.tegarch}, \texttt{predict.tegarch}, \texttt{print.tegarch}, \texttt{residuals.tegarch}, \texttt{summary.tegarch} and \texttt{vcov.tegarch}. In addition, the output produced by the \texttt{tegarchSim} function and the \texttt{fitted.tegarch} and \texttt{residuals.tegarch} methods are of the \texttt{Zoo} class, which means a range of time-series methods are available for these objects.

**Author(s)**

Genaro Sucarrat, \url{http://www.sucarrat.net/}

**References**


**Examples**

```r
# simulate 500 observations from model with default parameter values:
set.seed(123)
y <- tegarchSim(500)

# estimate and store as 'mymod':
mymod <- tegarch(y)

# print estimates and standard errors:
print(mymod)

# graph of fitted volatility (conditional standard deviation):
plot(fitted(mymod))

# plot forecasts of volatility 1-step ahead up to 10-steps ahead:
plot(predict(mymod, n.ahead=10))
```

**Description**

Extraction methods for objects of class ‘tegarch’ (i.e. the result of estimating a Beta-Skew-t-EGARCH model)
Usage

## S3 method for class 'tegarch'
coef(object, ...)
## S3 method for class 'tegarch'
fitted(object, verbose = FALSE, ...)
## S3 method for class 'tegarch'
logLik(object, ...)
## S3 method for class 'tegarch'
print(x, ...)
## S3 method for class 'tegarch'
residuals(object, standardised = TRUE, ...)
## S3 method for class 'tegarch'
summary(object, verbose = FALSE, ...)
## S3 method for class 'tegarch'
vcov(object, ...)

Arguments

object an object of class 'tegarch'
x an object of class 'tegarch'
verbose logical. If FALSE (default) then only basic information is returned
standardised logical. If TRUE (default) then the standardised residuals are returned. If FALSE then the scaled (by sigma) residuals are returned
... additional arguments

Details

Empty

Value

calc: A numeric vector containing the parameter estimates
fit: 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 return series y, fitted scale (sigma), fitted conditional standard deviation (stdev), fitted log-scale (lambda), dynamic component(s) (lambdadagger in the 1-component specification, lambda1dagger and lambda2dagger in the 2-component specification), the score (u), scaled residuals (epsilon) and standardised residuals (residstd)
logLik: The value of the log-likelihood at the maximum
print: Prints the most important parts of the estimation results
residuals: A zoo object. If standardised = TRUE (default), then the zoo object is a vector with the standardised residuals. If standardised = FALSE, then the zoo vector contains the scaled residuals
summary: A list. If verbose = FALSE, then only the most important entries are returned. If verbose = TRUE, then all entries apart from the 1st. (the y series) is returned
**coef.tegarch**

The variance-covariance matrix of the estimated coefficients. The matrix is obtained by inverting the numerically estimated Hessian.

**Author(s)**

Genaro Sucarrat, [http://www.sucarrat.net/](http://www.sucarrat.net/)

**References**


**See Also**

tegarch, coef, fitted, logLik, predict, predict.tegarch, print, summary, vcov

**Examples**

```r
# simulate 500 observations from model with default parameter values:
set.seed(123)
y <- tegarchSim(500)

# estimate and store as 'mymodel':
mymod <- tegarch(y)

# print estimation result:
print(mymod)

# extract coefficients:
coef(mymod)

# extract log-likelihood:
logLik(mymod)

# plot fitted conditional standard deviations:
plot(fitted(mymod))

# plot all the fitted series:
plot(fitted(mymod, verbose=TRUE))

# histogram of standardised residuals:
hist(residuals(mymod))
```
The skewed t distribution

Description
Density, random number generation, mean, variance, skewness and kurtosis functions for the uncentred skewed t distribution. The skewing method is that of Fernandez and Steel (1998).

Usage
```
dST(y, df = 10, sd = 1, skew = 1, log = FALSE)
rST(n, df = 10, skew = 1)
stmean(df, skew = 1)
stvar(df, skew = 1)
stskewness(df, skew = 1)
stkurtosis(df, skew = 1)
```

Arguments
- `y`: numeric vector of quantiles
- `n`: integer, the number of observations
- `df`: degrees of freedom, greater than 0 and less than Inf
- `sd`: scale, greater than 0 and less than Inf
- `skew`: skewness, greater than 0 and less than Inf. Symmetry obtains when skew = 1 (default).
- `log`: logical. TRUE returns the natural log of the density value, FALSE (default) returns the density value.

Details
Empty

Value
- `dST`: a numeric value, either the density value or the natural log of the density value
- `rST`: a numeric vector with `n` random numbers
- `stmean`: The mean of an uncentred skewed t variable
- `stvar`: The variance of an uncentred skewed t variable
- `stskewness`: 3rd. moment of a standardised skewed t variable
- `stkurtosis`: 4th. moment of a standardised skewed t variable

Note
Empty
Author(s)
Genaro Sucarrat, http://www.sucarrat.net/

References

See Also
tegarchSim

Examples
```r
# generate 1000 random numbers from the skewed t:
set.seed(123)
eps <- rST(500, df=5) # symmetric t
eps <- rST(500, df=5, skew=0.8) # skewed to the left
eps <- rST(500, df=5, skew=2) # skewed to the right

# compare empirical mean with analytical:
mean(eps)
STmean(5, skew=2)

# compare empirical variance with analytical:
var(eps)
STvar(5, skew=2)
```

Description
The dataset contains two variables, day and nasdaqret. Day is the date of the return and nasdaqret is the daily (closing value) log-return in percent of the Apple stock over the period 10 September 1985 - 10 May 2011 (a total of 6835 observations).

Usage
data(nasdaq)

Format
A data frame with 3215 observations:

- day a factor
- nasdaqret a numeric vector
predict.tegarch

Details
The data is studied in more detail in Harvey and Sucarrat (2013).

Source
The source of the original raw data is http://yahoo.finance.com/.

References

Examples
data(nasdaq) #load data into workspace
mymod <- tegarch(nasdaq[,"nasdaqret"]) #estimate volatility model of Apple returns
print(mymod)

predict.tegarch
Generate volatility forecasts n-steps ahead

Description
Generates volatility forecasts from a model fitted by tegarch (i.e. a Beta-Skew-t-EGARCH model)

Usage
## S3 method for class 'tegarch'
predict(object, n.ahead = 1, initial.values = NULL, n.sim = 10000,
verbose = FALSE, ...)

Arguments
object an object of class 'tegarch'.
n.ahead the number of steps ahead for which prediction is required.
initial.values a vector containing the initial values of lambda and lambdadagger (lambda1dagger
and lambda2dagger for 2-component models). If NULL (default) then the fitted
values associated with the last return-observation are used
n.sim number of simulated skew t variates.
verbose logical. If FALSE (default) then only the conditional standard deviations are
returned. If TRUE then also the scale is returned.
... additional arguments
The forecast formulas of exponential ARCH models are much more complicated than those of ordinary or non-exponential ARCH models. This is particularly the case when the conditional density is skewed. The forecast formula of the conditional scale of the Beta-Skew-t-EGARCH model is not available in closed form. Accordingly, some terms (expectations involving the skewed t) are estimated numerically by means of simulation.

A zoo object. If verbose = FALSE, then the zoo object is a vector with the forecasted conditional standard deviations. If verbose = TRUE, then the zoo object is a matrix with forecasts of both the conditional scale and the conditional standard deviation.

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


See Also
tegarch, predict

Examples
```r
set.seed(123)
y <- tegarchSim(500, omega=0.01, phi1=0.9, kappa1=0.1, kappastar=0.05, df=10, skew=0.8)
mymod <- tegarch(y)
plot(predict(mymod, n.ahead=10))
```
Estimate first order Beta-Skew-t-EGARCH models

Description

Fits a first order Beta-Skew-t-EGARCH model to a univariate time-series by exact Maximum Likelihood (ML) estimation. Estimation is via the \texttt{nlminb} function

Usage

\texttt{tegarch(y, asym = TRUE, skew = TRUE, components = 1, initial.values = NULL, lower = NULL, upper = NULL, hessian = TRUE, lambda.initial = NULL, c.code = TRUE, logl.penalty = NULL, aux = NULL, \ldots)}

Arguments

- \texttt{y}: numeric vector, typically a financial return series.
- \texttt{asym}: logical. TRUE (default) includes leverage or volatility asymmetry in the log-scale specification.
- \texttt{skew}: logical. TRUE (default) enables and estimates the skewness in conditional density (epsilon). The skewness method is that of Fernandez and Steel (1998).
- \texttt{components}: Numeric value, either 1 (default) or 2. The former estimates a 1-component model, the latter a 2-component model.
- \texttt{initial.values}: NULL (default) or a vector with the initial values. If NULL, then the values are automatically chosen according to model (with or without skewness, 1 or 2 components, etc.)
- \texttt{lower}: NULL (default) or a vector with the lower bounds of the parameter space. If NULL, then the values are automatically chosen.
- \texttt{upper}: NULL (default) or a vector with the upper bounds of the parameter space. If NULL, then the values are automatically chosen.
- \texttt{hessian}: logical. If TRUE (default) then the Hessian is computed numerically via the \texttt{optimHess} function. Setting hessian=FALSE speeds up estimation, which might be particularly useful in simulation. However, it also slows down the extraction of the variance-covariance matrix by means of the \texttt{vcov} method.
- \texttt{lambda.initial}: NULL (default) or a vector with the initial value(s) of the recursion for lambda and lambdadagger. If NULL then the values are chosen automatically.
- \texttt{c.code}: logical. TRUE (default) is faster since it makes use of compiled C-code.
- \texttt{logl.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, which are fatal for simulations. The value logl.penalty is the value returned by the log-likelihood function in the presence of NA or +/-Inf values.
- \texttt{aux}: NULL (default) or a list, se code. Useful for simulations (speeds them up).
- \texttt{\ldots}: further arguments passed to the \texttt{nlminb} function.
Value

Returns a list of class 'tegarch' with the following elements:

- **y**: the series used for estimation.
- **date**: date and time of estimation.
- **initial.values**: initial values used in estimation.
- **lower**: lower bounds used in estimation.
- **upper**: upper bounds used in estimation.
- **lambda.initial**: initial values of lambda provided by the user, if any.
- **model**: type of model estimated.
- **hessian**: the numerically estimated Hessian.
- **sic**: the value of the Schwarz (1978) information criterion.
- **par**: parameter estimates.
- **objective**: value of the log-likelihood at the maximum.
- **convergence**: an integer code. 0 indicates successful convergence, see the documentation of nlminb.
- **iterations**: number of iterations, see the documentation of nlminb.
- **evaluations**: number of evaluations of the objective and gradient functions, see the documentation of nlminb.
- **message**: a character string giving any additional information returned by the optimizer, or NULL. For details, see PORT documentation and the nlminb documentation.
- **NOTE**: an additional message returned if one tries to estimate a 2-component model without leverage.

Note

Empty

Author(s)

Genaro Sucarrat, [http://www.sucarrat.net/](http://www.sucarrat.net/)

References


See Also
tegarchSim, coef.tegarch, fitted.tegarch, logLik.tegarch, predict.tegarch, print.tegarch,
residuals.tegarch, summary.tegarch, vcov.tegarch

Examples

```r
##simulate series with 500 observations:
set.seed(123)
y <- tegarchSim(500, omega=0.01, phi1=0.9, kappa1=0.1, kappastar=0.05,
               df=10, skew=0.8)

##estimate a 1st. order Beta-t-EGARCH model and store the output in mymod:
mymod <- tegarch(y)

# print estimates and standard errors:
print(mymod)

# graph of fitted volatility (conditional standard deviation):
plot(fitted(mymod))

# graph of fitted volatility and more:
plot(fitted(mymod, verbose=TRUE))

# plot forecasts of volatility 1-step ahead up to 20-steps ahead:
plot(predict(mymod, n.ahead=20))

# full variance-covariance matrix:
vcov(mymod)
```

tegarchLogl

**Auxiliary functions**

**Description**

tegarchLogl, tegarchLogl2, tegarchRecursion and tegarchRecursion2 are auxiliary functions called by tegarch, and which are not intended to be used for the average user. Henceforth they are thun-
somly scarcely documented, but most should either be self-explanatory (for the non-average user!) or more or less documented in relation with the tegarch and tegarchSim functions.

**Usage**

```r
##the '2' relates to the 2-component specification:
tegarchLogl(y, pars, lower = -Inf, upper = Inf, lambda.initial = NULL,
            logl.penalty = -1e+100, c.code = TRUE, aux = NULL)
tegarchLogl2(y, pars, lower = -Inf, upper = Inf, lambda.initial = NULL,
            logl.penalty = -1e+101, c.code = TRUE, aux = NULL)
tegarchRecursion(y, omega = 0.1, phi1 = 0.4, kappa1 = 0.2, kappastar = 0.1,
                 df = 10, skew = 0.6, lambda.initial = NULL, c.code = TRUE, verbose = FALSE,
                 aux = NULL)
```
tegarchLogl

    aux = NULL
    tegarchRecursion2(y, omega = 0.1, phi1 = 0.4, phi2 = 0.2, kappa1 = 0.05,
        kappa2 = 0.1, kappastar = 0.02, df = 10, skew = 0.6, lambda.initial = NULL,
        c.code = TRUE, verbose = FALSE, aux = NULL)

Arguments

y          numeric vector, typically a financial return series
omega      numeric
phi1       numeric, must be less than 1 in absolute value
phi2       numeric, must be less than 1 in absolute value
kappa1     numeric
kappa2     numeric
kappastar  numeric
df         numeric, the value of df (degrees of freedom)
skew       numeric (positive), the value of skew (skewness parameter)
verbose    logical. If FALSE (default) then only lambda is returned. If TRUE then a matrix
            with y and the fitted values of, amongst other, sigma, the log-scale (lambda), the
            conditional standard deviation (stdev), u, epsilon and the standardised residuals
            (residstd) are returned
pars       numeric vector, the parameter values
lower      numeric vector, the lower bounds used during estimation
upper      numeric vector, the upper bounds used during estimation
lambda.initial NULL (default) or initial value(s) of the recursion for lambda. If NULL, then
                the values are chosen automatically
logl.penalty numeric value
c.code     logical. TRUE (default) is faster since it makes use of compiled C-code
aux        NULL (default) or a list, se tegarch code

Details

tegarchLogl and tegarchLogl2 return the value of the log-likelihood for a 1-component and 2-
component model, respectively.

Value

tegarchLogl:   The log-likelihood value (i.e. a numeric) of a 1-component specification
tegarchLogl2:  The log-likelihood value (i.e. a numeric) of a 2-component specification
tegarchRecursion:
                A numeric vector containing the lambda values if verbose=FALSE (default). If
                verbose=TRUE then a matrix then a matrix with y and the fitted values of sigma,
                the log-scale (lambda), the conditional standard deviation (stdev), u, epsilon and
                the standardised residuals (residstd) are returned
tegarchRecursion2:
A numeric vector containing the lambda values if verbose=FALSE (default). If
verbose=TRUE, then a matrix then a matrix with y and the fitted values of sigma,
the log-scale (lambda), the conditional standard deviation (stdev), u, epsilon and
the standardised residuals (residstd) are returned.

Author(s)
Genaro Sucarrat, http://www.sucarrat.net/

References
Fernandez and Steel (1998), 'On Bayesian Modeling of Fat Tails and Skewness', Journal of the

Harvey and Sucarrat (2013), 'EGARCH models with fat tails, skewness and leverage', forthcoming
in Computational Statistics and Data Analysis. Working paper version: Cambridge Working Papers
in Economics 1236, Faculty of Economics, University of Cambridge.

See Also
tegarch, tegarchSim, fitted.tegarch

tegearchSim
Simulate from a first order Beta-Skew-t-EGARCH model

Description
Simulate the y series (typically interpreted as a financial return or the error in a regression) from
a first order Beta-Skew-t-EGARCH model. Optionally, the conditional scale (sigma), log-scale
(lambda), conditional standard deviation (stdev), dynamic components (lambdadagger in the 1-
component specification, lambda1dagger and lambda2dagger in the 2-component specification),
score (u) and centred innovations (epsilon) are also returned.

Usage
tegarchSim(n, omega = 0, phi1 = 0.95, phi2 = 0, kappa1 = 0.01, kappa2 = 0,
kappastar = 0, df = 10, skew = 1, lambda.initial = NULL, verbose = FALSE)

Arguments
n integer, length of y (i.e. no of observations)
omega numeric, the value of omega
phi1 numeric, the value of phi1
phi2 numeric, the value of phi2
tegarchSim

kappa1 numeric, the value of kappa1
kappa2 numeric, the value of kappa2
kappastar numeric, the value of kappastar
df numeric, the value of df (degrees of freedom)
skew numeric, the value of skew (skewness parameter)
lambda.initial NULL (default) or initial value(s) of the recursion for lambda or log-volatility. If NULL then the values are chosen automatically
verbose logical, TRUE or FALSE (default). If TRUE then a matrix with n rows containing y, sigma, lambda, lambdadagger, u and epsilon is returned. If FALSE then only y is returned

Details

Empty

Value

A zoo vector of length n or a zoo matrix with n rows, depending on the value of verbose.

Author(s)

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

References


See Also
tegarch, zoo

Examples

#1-component specification: simulate series with 500 observations:
set.seed(123)
y <- tegarchSim(500, omega=0.01, phi1=0.9, kappa1=0.1, kappastar=0.05, df=10, skew=0.8)

#simulate the same series, but with more output (volatility, log-volatility or #1lambda, lambdadagger, u and epsilon)
set.seed(123)
y <- tegarchSim(500, omega=0.01, phi1=0.9, kappa1=0.1, kappastar=0.05, df=10, skew=0.8,
verbose=TRUE)

# plot the simulated values:
plot(y)

# 2-component specification: simulate series with 500 observations:
set.seed(123)
y <- tegarchSim(500, omega=0.01, phi1=0.95, phi2=0.9, kappa1=0.01, kappa2=0.05,
               kappa2=0.03, df=10, skew=0.8)
**Index**

**Topic Statistical Models**
- coef.tegarch, 3
- dst, 6
- predict.tegarch, 8
- tegarch, 10
- tegarchLogl, 12
- tegarchSim, 14
**Topic datasets**
- nasdaq, 7

- betategarch (betategarch-package), 2
- betategarch-package, 2
- coef, 5
- coef.tegarch, 3, 12
- dst, 6
- fitted, 5
- fitted.tegarch, 3, 12, 14
- fitted.tegarch (coef.tegarch), 3
- logLik, 5
- logLik.tegarch, 3, 12
- logLik.tegarch (coef.tegarch), 3
- nasdaq, 7
- nlminb, 10
- predict, 5, 9
- predict.tegarch, 3, 5, 8, 12
- print, 5
- print.tegarch, 3, 12
- print.tegarch (coef.tegarch), 3
- residuals.tegarch, 3, 12
- residuals.tegarch (coef.tegarch), 3
- rST (dst), 6
- STkurtosis (dst), 6
- STmean (dst), 6
- STskewness (dst), 6
- STvar (dst), 6
- summary, 5
- summary.tegarch, 3, 12
- summary.tegarch (coef.tegarch), 3
- tegarch, 2, 5, 8, 9, 10, 12–15
- tegarchLogl, 12
- tegarchLogl2 (tegarchLogl), 12
- tegarchRecursion (tegarchLogl), 12
- tegarchRecursion2 (tegarchLogl), 12
- tegarchSim, 2, 3, 7, 12, 14, 14
- vcov, 5
- vcov.tegarch, 3, 12
- vcov.tegarch (coef.tegarch), 3
- zoo, 3, 15