Package ‘GEVStableGarch’

July 21, 2015

Type Package

Title ARMA-GARCH/APARCH models with GEV and stable distributions

Version 1.1

Date 2015-07-19

Author Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes

Maintainer Thiago do Rego Sousa <thiagoestatistico@gmail.com>

Description Package for ARMA-GARCH or ARMA-
APARCH modelling with GEV and stable conditional distributions.

License GPL (>=2)

Depends R(>= 2.15.0),fGarch,fExtremes,stabledist,skewt,Rsolnp

Additional_repositories http://www.robustanalysis.com

R topics documented:

`GEVStableGarch-package`   *ARMA-GARCH/APARCH modelling with GEV and stable distributions*

Description

This package is design to perform maximum likelihood estimation of ARMA-GARCH/APARCH models with Generalized Extreme Distribution (GEV) or stable conditional distributions.

Details

Package:	GEVStableGarch
Type:	Package
Version:	1.1
Date:	2015-07-19
License:	GPL(>=2)
Depends:	R(>= 2.15.0), fGarch, fExtremes, stabledist, skewt, Rsolnp

This package contains functions for estimating and simulating combined ARMA-GARCH or ARMA-APARCH models with error distributions following GEV or stable densities. The current version has a new algorithm that allows the user to enforce stationarity during estimation. The package also allows the user to estimate several models and choose the best one according to a goodness-of-fit Criteria.

Author(s)

Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes

Maintainer: Thiago do Rego Sousa <thiagoestatistico@gmail.com>

References

Brockwell, P. J, e Davis, R. A. Introduction to Time Series and Forecasting. Springer, New York, 1996.

Mittnik S. , Paolella M.S. , Rachev, S.T. Stationarity of stable power- GARCH processes. Journal of Econometrics 106, 97-107. 2002.

Nolan, J. P. Numerical calculations of stable densities and distribution functions. Communications in Statistics - Stochastic Models 13: 759- 774, 1997.

Nolan, J. P. Maximum likelihood estimation and diagnostics for stable distribution. In O. E.Barndorff-Nielsen, T. Mikosch, e S. I. Resnick (Eds.), Levy Processes: Theory and Applications: 379-400. Boston, Birkhauser, 2000.

Wurtz, D., Y. Chalabi, e L. Luksan. Parameter estimation of ARMA models with GARCH/APARCH errors: An R and SPlus software implementation. Journal of Statistical Software, 2006.

Zhao X. ,Scarrott C. J. , Oxley L. , Reale M. GARCH dependence in extreme value models with Bayesian inference. Mathematics an Computers in Simulation, Vol. 81, Issue 7, 1430-1440. 2011.

fGEVSTABLEGARCH-class *Class* "fGEVSTABLEGARCH"

Objects from the Class

Objects can be created by calls of the form `new("fGEVSTABLEGARCH", ...)`.

Slots

`call`: Object of class "call" ~~

`formula`: Object of class "formula" ~~

`method`: Object of class "character" ~~

`convergence`: Object of class "numeric" ~~

`messages`: Object of class "list" ~~

`data`: Object of class "numeric" ~~

`fit`: Object of class "list" ~~

`residuals`: Object of class "numeric" ~~

`h.t`: Object of class "numeric" ~~

`sigma.t`: Object of class "numeric" ~~

`title`: Object of class "character" ~~

`description`: Object of class "character" ~~

Methods

show `signature(object = "fGEVSTABLEGARCH")`: ...

Examples

```
showClass("fGEVSTABLEGARCH")
```

fGEVSTABLEGARCHSPEC-class *Class* "fGEVSTABLEGARCHSPEC"

Objects from the Class

Objects can be created by calls of the form `new("fGEVSTABLEGARCHSPEC", ...)`.

Slots

`call`: Object of class "call" ~~

`formula`: Object of class "formula" ~~

`model`: Object of class "list" ~~

`presample`: Object of class "matrix" ~~

`distribution`: Object of class "character" ~~

`rseed`: Object of class "numeric" ~~

Methods

show `signature(object = "fGEVSTABLEGARCHSPEC")`: ...

Examples

```
showClass("fGEVSTABLEGARCHSPEC")
```

`GAt` *Generalized Asymmetric t Distribution*

Description

Functions to compute density, distribution function, quantile function and to generate random variates for the generalized asymmetric t distribution.

Usage

```
dGAt(x, mean = 0, sd = 1, d = 2, xi = 1, log = FALSE)
pGAt(q, mean = 0, sd = 1, d = 2, xi = 1)
qGAt(p, mean = 0, sd = 1, d = 2, xi = 1)
rGAt(n, mean = 0, sd = 1, d = 2, xi = 1)
```

Arguments

`mean, sd, d, xi`	location parameter `mean`, scale parameter `sd`, shape 1 parameter `d`. shape 2 parameter `xi`.
`n`	the number of observations.
`p`	a numeric vector of probabilities.
`x, q`	a numeric vector of quantiles.
`log`	a logical; if TRUE, densities are given as log densities.

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.

Author(s)

Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes.

gsFit *Estimation of ARMA-GARCH/APARCH models*

Description

This function uses Maximum Likelihood technique to estimate the parameters of ARMA-GARCH or ARMA-APARCH model with several conditional distributions. The user can also enforce stationarity during estimation

Usage

```
gsFit(formula = ~garch(1, 1), data, cond.dist = c("stableS0", "stableS1", "stableS2", "gev", "GAt"
```

Arguments

`formula`

`data`

`cond.dist` The conditional distribution of the model to be estimated. This parameter should be one of the following strings: “norm” (Normal Distribution), “t-student” (t-Student distribution), “sstd” (skew t-Student Distribution), “GEV” (Generalized Extreme Value Distribution) or “stable” (Stable Distribution)

`include.mean` This is a boolean variable. It intercept is TRUE than we estimate the model with intercept, otherwise we will not use the intercept variable.

`algorithm` The algorithm to be used to optimize the likelihood of the process. This parameter should be one of the following strings: “nlminb” (nlminb R internal Routine for optimization) or “SQP” (Sequential quadratic programming routine implemented in package RSolnp).

`control` Control parameters used either in “SQP” routine (see the RSolnp documentation) or in “nlminb” (see the nlminb R documentation).

`tolerance` Tolerance for parameter estimation.

`title`

`description`

Details

There is also the possibility to fit models with conditional “Normal”, “t-student” and “skew t-student” distributions but the main purpose of this routine is to deal with “GEV” and “stable”. The parameters will be interpreted according to the following equations (see Wurtz et al. ,2006)

$$X_t = \mu + \sum_{i=1}^{m} a_i X_{t-i} + \sum_{j=1}^{n} b_j \varepsilon_{t-j} + \varepsilon_t$$

$$\varepsilon_t = \sigma_t z_t, \quad z_t \overset{iid}{\sim} D(0,1)\ ,$$

$$\sigma_t^\delta = \omega + \sum_{i=1}^{p} \alpha_i (\varepsilon_{t-i} - \gamma_i |\varepsilon_{t-i}|)^\delta + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^\delta$$

where

$$X_t$$

is the ARMA process,

$$\sigma_t$$

is the APARCH process and

$$D(0,1)$$

is the standard density distribution to be used in the model.

Value

The function returns an object containing the following items:

`order`	The estimated parameter set for the chosen model.
`hessian`	The estimated Hessian matrix.
`model`	A string describing the estimated model.
`cond.dist`	The conditional distribution used to fit the model.
`data`	The time series data used to fit the model.
`llh`	The negative log likelihood of the estimated model
`par`	a vector containing the estimated parameters
`hessian`	The estimated Hessian matrix.
`ARMA.res`	ARMA residuals
`GARCH.sig`	GARCH or APARCH residuals.
`aic`	The reported AIC (Akaike Information Criterion) for the current model.
`aicc`	The reported corrected AIC (AICc) for the current model.
`bic`	The reported BIC (Bayesian Information Criterion.
`se.coef`	Standard error for the estimated parameters.
`tval`	Calculated t-value for the estimated parameters.
`matcoef`	An organized matrix to present the objects.

Enhances

The estimation of ARMA-GARCH/APARCH models with conditional stable distribution is mainly dependent on the time taken during the calculation of density points. Our routine uses the standard R stable density implemented in package **stabledist**. There is also another numerical routine to calculate stable densities which is part of an R package called **stable**. This package implements a faster computation of stable densities that are accurately enough to perform numerical optimization. When **GEVStableGarch** loads it verifies if package **stable** is available. If it is found the function GSGarch.Fit will perform optimization using **stable**'s routine. Otherwise it will use stable density from package **stabledist**. Package **stable** is available at '`http://www.robustanalysis.com`'.

Author(s)

Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes

References

Mittnik S. , Paolella M.S. , Rachev, S.T. Stationarity of stable power- GARCH processes. Journal of Econometrics 106, 97-107. 2002.

Wurtz, D., Y. Chalabi, e L. Luksan. Parameter estimation of ARMA models with GARCH/APARCH errors: An R and SPlus software implementation. Journal of Statistical Software, 2006.

Zhao X. ,Scarrott C. J. , Oxley L. , Reale M. GARCH dependence in extreme value models with Bayesian inference. Mathematics an Computers in Simulation, Vol. 81, Issue 7, 1430-1440. 2011.

Yinyu Ye. Interior Algorithms for Linear, Quadratic, and Linearly Constrained Non-Linear Programming. Ph.D. Thesis, Department of EES, Stanford University, 1987.

Examples

```
# This examples uses the dataset of the package fGarch to estimate
# an ARMA(1,1)-GARCH(1,1) with GEV conditional distribution.
library(fGarch)
data(dem2gbp)
x = dem2gbp[, 1]
gev.model = gsFit(data = x , formula = ~garch(1,1), cond.dist = "norm")
```

`gsMomentAparch` *Evaluate the moments expression E(|Z| - gamma * Z) ^ delta*

Usage

```
gsMomentAparch(cond.dist = c("stableS1", "gev", "GAt", "norm", "std", "sstd", "skstd", "ged"), sha
```

Arguments

`cond.dist`

`shape`

`skew`

`delta`

`gm`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (cond.dist = c("stableS1", "gev", "GAt", "norm", "std",
    "sstd", "skstd", "ged"), shape = 1.5, skew = 0, delta = 1,
    gm = 0)
{
    cond.dist = match.arg(cond.dist)
    if (cond.dist == "stableS1")
        kappa = .stableS1MomentAparch(shape = shape, skew = skew,
            delta = delta, gm = gm)
    if (cond.dist == "gev")
```

```
        kappa = .gevMomentAparch(shape = shape, delta = delta,
            gm = gm)
    if (cond.dist == "GAt")
        kappa = .GAtMomentAparch(shape = shape, delta = delta,
            skew = skew, gm = gm)
    if (cond.dist == "norm")
        kappa = .normMomentAparch(delta = delta, gm = gm)
    if (cond.dist == "std")
        kappa = .stdMomentAparch(shape = shape, delta = delta,
            gm = gm)
    if (cond.dist == "sstd")
        kappa = .sstdMomentAparch(shape = shape, skew = skew,
            delta = delta, gm = gm)
    if (cond.dist == "skstd")
        kappa = .skstdMomentAparch(shape = shape, skew = skew,
            delta = delta, gm = gm)
    if (cond.dist == "ged")
        kappa = .gedMomentAparch(shape = shape, delta = delta,
            gm = gm)
    kappa
  }
```

`gsSelect` *Selects the best model according to goodness-of-fit*

Description

This function estimates ARMA-GARCH/APARCH models with varying order and return the one with minimum AIC, BIC or AICc

Usage

```
gsSelect(data, order.max = c(1, 1, 1, 1), selection.criteria = c("AIC", "AICc", "BIC"), is.aparch =
```

Arguments

`data`	Data with the time series to be estimated. This object can not contain NULL elements.
`order.max`	Maximum order of models to search
`selection.criteria`	The goodness-of-fit measure to use be minimized.
`is.aparch`	Boolean variable indicating whether to search for ARMA-GARCH or ARMA-APARCH models
`cond.dist`	The conditional distribution of the model to be estimated. This parameter should be one of the following strings: "norm" (Normal Distribution), "t-student" (t-Student distribution), "sstd" (skew t-Student Distribution), "GEV" (Generalized Extreme Value Distribution) or "stable" (Stable Distribution).
`include.mean`	This is a boolean variable. It intercept is TRUE than we estimate the model with intercept, otherwise we will not use the intercept variable.

`algorithm`	The algorithm to be used to optimize the likelihood of the process. This parameter should be one of the following strings: "nlminb" (nlminb R internal Routine for optimization) or "SQP" (Sequential quadratic programming routine implemented in package RSolnp).
`...`	Additional parameters to be passed to function GSfit during estimation.

Value

The model with minimum AIC found within the specified range in parameters mMAX, nMAX, pMAX, qMAX.

Author(s)

Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes

References

Brockwell, P. J, e Davis, R. A. Introduction to Time Series and Forecasting. Springer, New York, 1996.

Examples

```
# AIC fit using models from ARMA(0,0)-GARCH(1,0) to ARMA(1,1)-GARCH(1,1)
# with GEV conditional distribution
#library(fGarch)
#data(dem2gbp)
#x = dem2gbp[, 1]
# GSgarch.FitAIC(data = x,1,0,1,0,cond.dist = "gev")
```

`gsSim` *Simulation of ARMA-GARCH/APARCH process*

Description

This functions simulate time series following ARMA-GARCH/APARCH models with several conditional distributions, including GEV and stable distributions.

Usage

```
gsSim(spec = garchSpec(), n = 100, n.start = 100)
```

Arguments

`spec`	A model specified with function gsSpec
`n`	The size of simulated time serie
`n.start`	

Details

The initial values of the time series are fixed and the recursion formulas of the model are used to simulate the dynamics of the process. We do not verify the stationarity conditions of the model because the simulation of non-stationary process could also be of interest.

Value

The function returns an object containing the following items:

- `model` A string describing the estimated model.
- `cond.dist` The conditional distribution used to fit the model.
- `series` An array of two columns. The first column is the simulated process

$$X_t$$

and the second one is the

$$\sigma_t$$

simulated process.

Author(s)

Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes.

References

Brockwell, P. J, e Davis, R. A. Introduction to Time Series and Forecasting. Springer, New York, 1996.

Nolan, J. P. Numerical calculations of stable densities and distribution functions. Communications in Statistics - Stochastic Models 13: 759- 774, 1997.

Examples

```
# Simulation of a ARMA-APARCH process with stable conditional distribution
#x <- GSgarch.Sim(N = 2500, mu = 0.1,a = c(0.2,0.3),b = c(0.2,0.5),
#omega = 0.1, alpha = c(0.1,0.2),beta = c(0.1,0.1),gm=c(0.3,-0.3),
#delta = 1,skew = 0.3,shape = 1.9, cond.dis = "stable")
```

`gsSpec` *Specification of ARMA-GARCH/APARCH models*

Usage

```
gsSpec(model = list(), presample = NULL, cond.dist = c("stableS0", "stableS1", "stableS2", "gev",
```

Arguments

- `model`
- `presample`
- `cond.dist`
- `rseed`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

`show-methods` *~~ Methods for Function* `show` *~~*

Description

~~ Methods for function `show` ~~

Methods

`signature(object = "fGEVSTABLEGARCH")`

`signature(object = "fGEVSTABLEGARCHSPEC")`

`skstd` *Skew Student's t Distribtuion from Fernandez and Steel (2002)*

Description

Functions to compute density, distribution function, quantile function and to generate random variates for the Skew Student's t Distribtuion from Fernandez and Steel (2002) withouth reparemeterization.

Usage

```
dskstd(x, mean = 0, sd = 1, nu = 3, xi = 1, log = FALSE)
pskstd(q, mean = 0, sd = 1, nu = 3, xi = 1)
qskstd(p, mean = 0, sd = 1, nu = 3, xi = 1)
rskstd(n, mean = 0, sd = 1, nu = 3, xi = 1)
```

Arguments

`mean, sd, nu, xi`	location parameter `mean`, scale parameter `sd`, shape parameter `nu`. asymmetry parameter `xi`.
`n`	the number of observations.
`p`	a numeric vector of probabilities.
`x, q`	a numeric vector of quantiles.
`log`	a logical; if TRUE, densities are given as log densities.

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.

Author(s)

Thiago do Rego Sousa, Cira Etheowalda Guevara Otiniano and Silvia Regina Costa Lopes.

Index