Package ‘symmetry’

August 23, 2019

Title A Package for Testing the Symmetry of Data and Model Residuals

Version 0.1.0

Description Implementations a large number of tests for symmetry (and their bootstrap variants), which can be used to test the symmetry of IID samples or of model residuals. Currently, the supported models are linear models and GARCH models (fitted with the fGarch package). The tests are implemented using Rcpp which ensures great performance.

Depends R (>= 3.1.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports Rcpp, Rdpack

RdMacros Rdpack

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.1.1

SystemRequirements C++11

Suggests knitr, rmarkdown, sn, fGarch, testthat

R topics documented:

rmixnorm 2
rsl 2
symmetry 3
symmetry_test 3
TestStatistics 5

Index **10**

`rmixnorm` *Mixture of 2 normal distributions*

Description

Generates random numbers from a mixture of 2 normal distributions

Usage

```
rmixnorm(n, mean1 = 0, sd1 = 1, mean2 = 0, sd2 = 1, p = 0.5)
```

Arguments

`n`	number of observations
`mean1`	mean of the first normal
`sd1`	standard deviation of the first normal
`mean2`	mean of the second normal
`sd2`	standard deviation of the second normal
`p`	probability of the first normal

Value

Vector of random numbers from the specified mixture of normals.

`rsl` *Azzalini skew logistic distribution*

Description

Generates random numbers from the skew logistic distribution

Usage

```
rsl(n = 1, xi = 0, omega = 1, alpha = 0, dp = NULL)
```

Arguments

`n`	sample size.
`xi`	vector of location parameters.
`omega`	vector of (positive) scale parameters.
`alpha`	vector of slant parameters.
`dp`	a vector of length 3 whose elements represent the parameters described above. If `dp` is specified, the individual parameters cannot be set.

Value

Vector of random numbers from Azzalini skew logistic distribution.

`symmetry` *symmetry: A package which implements tests for symmetry of IID data, linear models and GARCH models*

Description

The package caontains a large number of tests for symmetry (and their bootstrap variants), which can be used to test the symmetry of IID samples or of model residuals. Currently, the supported models are linear models and GARCH models (fitted with the fGarch package). The tests are implemented using Rcpp which ensures great performance.

Details

To see the available tests, see TestStatistics

For documentation on how to perform the tests, see symmetry_test

---

`symmetry_test` *Perform symmetry tests*

Description

This is a generic function used to perform symmetry tests on numeric vectors or objects of class lm (linear models) and objects of class fGARCH (GARCH mdels fitted with the fGarch package).

Usage

```
symmetry_test(x, ...)

## Default S3 method:
symmetry_test(x, stat, mu = NULL, bootstrap = FALSE,
  B = 1000, boot_method = c("sign", "reflect"), trim = 0, k = 0,
  ...)

## S3 method for class 'lm'
symmetry_test(x, stat, B = 1000, boot_method = c("sign",
  "reflect"), k = 0, ...)

## S3 method for class 'fGARCH'
symmetry_test(x, stat, B = 1000, burn = 0,
  boot_method = c("sign", "reflect"), k = 0, approximate = FALSE,
  ...)
```

Arguments

`x`	an object of class numeric, lm or fGARCH
`...`	not used
`stat`	a character vector indicating the test statistic to be used (see Available Test Statistics)
`mu`	the location parameter around which to
`bootstrap`	a logical indicationg whether to use bootstrap
`B`	the number of bootstrap replications
`boot_method`	the method of bootstrap sample generation (see Details)
`trim`	the trim value used for estimating the location parameter (as used in "mean")
`k`	the k parameter of the statistic, ignored if the test statistic doesn't depend on a parameter (see Test Statistics)
`burn`	the number of elements to remove from the beggining of the time series for testing
`approximate`	a logical indicating whether to use the faster approximate bootstrap method (see Details)

Details

The tests are performed using bootstrap procedures or using asymptotic results, where applicable. Currently, two methods of generating a bootstrap sample from the null distribution are available. The "sign" method generates the bootstrap sample by multiplying the existing sample by -1 or 1 at random (with equal probabilities), essentially randomizing the sign of the data, giving a symmetric distribution. The "reflect" method reflects the sample around zero and samples length(x) elements with replacement. In practive, it has been shown that the "sign" method is almost always better, thus is the default.

For numeric data, the tests can be performed around a known (parameter "mu") or unknown location parameter. For unknown location (when mu = NULL), bootstrap must be used and the estimate of the location parameter used is the trimmed mean, with trim parameter "trim". By default, the mean is taken (trim = 0).

For linear models, the tests are based on a bootstrap procedure as in (Allison and Pretorius 2016) and are used to test the symmetry of the residuals around zero.

For GARCH models (must be fitted with the fGarch package), the tests are also based on bootstrap and test for symmetry of the residuals around zero. An approximation of the bootstrap procedure is available where the residuals are treated as iid data, which is much faster and has been shown to give similar results to the default bootstrap procedure (described in (Klar et al. 2012)).

Value

An object of class "htest" containing the results of the testing.

References

Allison JS, Pretorius C (2016). "A Monte Carlo evaluation of the performance of two new tests for symmetry." *Computational Statistics*, **32**(4), 1323–1338. doi: 10.1007/s0018001606804, https://doi.org/10.1007/s00180-016-0680-4.

Klar B, Lindner F, Meintanis S (2012). "Specification tests for the error distribution in GARCH models." *Computational Statistics \& Data Analysis*, **56**(11), 3587–3598. doi: 10.1016/j.csda.2010.05.029, https://doi.org/10.1016/j.csda.2010.05.029.

Examples

```
set.seed(1)

# IID samples
x <- rnorm(50)
symmetry_test(x, "MOI", bootstrap = TRUE, k = 3)
x <- rsl(50, alpha = 1.5)
symmetry_test(x, "MOI", bootstrap = TRUE, k = 3)

# Linear models
lin_model <- lm(dist ~ speed, cars)
symmetry_test(lin_model, "B1")

# Garch models
library(fGarch)
specskew19 = fGarch::garchSpec(model = list(omega = 0.1,
                                    alpha = 0.3,
                                    beta = 0.3,
                                    skew = 1.9),
                                    cond.dist = "snorm")

x <- fGarch::garchSim(specskew19, n = 500)
g <- fGarch::garchFit(~garch(1,1), x, cond.dist = "QMLE",
              include.mean = FALSE, trace = FALSE)
symmetry_test(g, "CH", B=200, burn = 100)
symmetry_test(g, "CH", B=200, burn = 100, approximate = TRUE)
```

`TestStatistics` *Available test statistics for symmetry tests*

Description

The list of implemented test statistics and their functions

Usage

```
B1(X)
```

```
BHI(X)

BHK(X)

CH(X)

CM(X)

K2U(X)

K2(X)

KS(X)

SGN(X)

WCX(X)

L2(X, k)

MGG(X)

MI(X, k)

MK(X, k)

MOI(X, k)

MOK(X, k)

M(X)

NAI(X, k)

NAK(X, k)

S1(X, k)

S2(X, k)

T1(X, k)

T2(X, k)
```

Arguments

X	the numeric vector for which to calculate the test statistic
k	the 'k' parameter in the formula (if applicable)

Details

Below is a list of the implemented test statistics in the package. Each statistic is listed by it's name, a code string (e.g. 'B1', CM','MOI') and the formula of the statistic which is evaluated. The code string is used as an argument to the symmetry_test function. Some statistics depend on a parameter 'k' which can be seen from the formulas and is also passed as an argument.

Each statistic is implemented as a function with the same name as the code string, so the name of the function is passed as the argument "stat" to the symmetry_test function

Value

The value of the test statistic.

Functions

- `B1`: The $\sqrt{b_1}$ test statistic (Cabilio and Masaro 1996)
  to be added
- `BHI`: The Cabilio–Masaro test statistic (Cabilio and Masaro 1996)

$$\frac{1}{n\binom{n}{2}}\sum_{\mathcal{I}_2}\sum_{i_3=1}^{n}\left(\frac{1}{2}I\{|X_{i_1}|<|X_{i_3}|\}+\frac{1}{2}I\{|X_{i_2}|<|X_{i_3}|\}-I\{|X_{(2),X_{i_1},X_{i_2}}|<|X_{i_3}|\}\right)$$

- `BHK`: The Cabilio–Masaro test statistic (Cabilio and Masaro 1996)

$$\sup_{t>0}\left|\frac{1}{\binom{n}{2}}\sum_{\mathcal{I}_2}\left(\frac{1}{2}I\{|X_{i_1}|<t\}+\frac{1}{2}I\{|X_{i_2}|<t\}-I\{|X_{(2),X_{i_1},X_{i_2}}|<t\}\right)\right|$$

- `CH`: The Cabilio–Masaro test statistic (Cabilio and Masaro 1996)
  to be added
- `CM`: The Cabilio–Masaro test statistic (Cabilio and Masaro 1996)
  to be added
- `K2U`: The Miao, Gel and Gastwirth test statistic (Miao et al. 2006)

$$\sup_{t>0}\frac{1}{\binom{n}{2}}\left|\sum_{1\le i<j\le n}I\{|X_i-X_j|<t\}-I\{|X_i+X_j|<t\}\right|$$

- `K2`: The Miao, Gel and Gastwirth test statistic (Miao et al. 2006)

$$\sup_{t>0}\frac{1}{n^2}\left|\sum_{i,j=1}^{n}I\{|X_i-X_j|<t\}-I\{|X_i+X_j|<t\}\right|$$

- `KS`: Kolmogorov–Smirnov test statistic (Miao et al. 2006)

$$\sup_{t}|F_n(t)-(1-F_n(-t))|$$

- `SGN`: The Sign test statistic (Miao et al. 2006)

$$\frac{1}{n}\sum_{i=1}^{n} I\{X_i > 0\} - \frac{1}{2}$$

- `WCX`: The Wilcoxon test statistic (Miao et al. 2006)

$$\frac{1}{\binom{n}{2}} \sum_{1 \le i < j \le n} I\{X_i + X_j > 0\} - \frac{1}{2}$$

- `L2`: The Miao, Gel and Gastwirth test statistic (Miao et al. 2006)

$$\frac{1}{n^4}\sum_{i,j,m,l=1}^{n} \quad \Big(\frac{1}{k+|X_i-X_j|+|X_m-X_l|} - \frac{1}{k+|X_i-X_j|+|X_m+X_l|} \\ - \frac{1}{k+|X_i+X_j|+|X_m-X_l|} + \frac{1}{k+|X_i+X_j|+|X_m+X_l|}\Big)$$

- `MGG`: The Miao, Gel and Gastwirth test statistic (Miao et al. 2006)
  to be added
- `MI`: The Mira test statistic (Mira 1999)

$$\frac{1}{n\binom{n}{2k+1}} \sum_{\mathcal{I}_{2k}} \sum_{i_{2k+1}=1}^{n} I\{-(X_{(k+1),X_{i_1},\ldots,X_{i_{2k}}}| < X_{i_{2k+1}}\} - I\{X_{(k+1),X_{i_1},\ldots,X_{i_{2k}}} < X_{i_{2k+1}}\}$$

- `MK`: The Mira test statistic (Mira 1999)

$$\sup_{t>0}\left|\frac{1}{\binom{n}{2k}} \sum_{\mathcal{I}_{2k}} I\{-(X_{(k+1),X_{i_1},\ldots,X_{i_{2k}}}) < t\} - I\{X_{(k+1),X_{i_1},\ldots,X_{i_{2k}}} < t\}\right|$$

- `MOI`: The Mira test statistic (Mira 1999)

$$\frac{1}{n\binom{n}{2k}} \sum_{\mathcal{I}_{2k}} \sum_{i_{2k+1}=1}^{n} I\{|X_{(k),X_{i_1},\ldots,X_{i_{2k}}}| < |X_{i_{2k+1}}|\} - I\{|X_{(k+1),X_{i_1},\ldots,X_{i_{2k}}}| < |X_{i_{2k+1}}|\}$$

- `MOK`: The Mira test statistic (Mira 1999)

$$\sup_{t>0}\left|\frac{1}{\binom{n}{2k}} \sum_{\mathcal{I}_{2k}} I\{|X_{(k),X_{i_1},\ldots,X_{i_{2k}}}| < t\} - I\{|X_{(k+1),X_{i_1},\ldots,X_{i_{2k}}}| < t\}\right|$$

- `M`: The Mira test statistic (Mira 1999)
  to be added
- `NAI`: Allison T_2 statistic (Klar et al. 2012)

$$\frac{1}{n\binom{n}{k}} \sum_{\mathcal{I}_{k}} \sum_{i_{k+1}=1}^{n} I\{|X_{(1),X_{i_1},\ldots,X_{i_k}}| < |X_{i_{k+1}}|\} - I\{|X_{(k),X_{i_1},\ldots,X_{i_k}}| < |X_{i_{k+1}}|\}$$

- `NAK`: Allison T_2 statistic (Klar et al. 2012)

$$\sup_{t>0}\left|\frac{1}{\binom{n}{k}}\sum_{\mathcal{I}_k} I\{|X_{(1),X_{i_1},\ldots,X_{i_k}}|<t\} - I\{|X_{(k),X_{i_1},\ldots,X_{i_k}}|<t\}\right|$$

- `S1`: Allison T_2 statistic (Klar et al. 2012)
  to be added
- `S2`: Allison T_2 statistic (Klar et al. 2012)
  to be added
- `T1`: Allison T_2 statistic (Klar et al. 2012)
  to be added
- `T2`: Allison T_2 statistic (Klar et al. 2012)
  to be added

References

Cabilio P, Masaro J (1996). "A simple test of symmetry about an unknown median." *Canadian Journal of Statistics*, **24**(3), 349–361. doi: 10.2307/3315744, `https://doi.org/10.2307/3315744`.

Klar B, Lindner F, Meintanis S (2012). "Specification tests for the error distribution in GARCH models." *Computational Statistics \& Data Analysis*, **56**(11), 3587–3598. doi: 10.1016/j.csda.2010.05.029, `https://doi.org/10.1016/j.csda.2010.05.029`.

Miao W, Gel YR, Gastwirth JL (2006). "A new test of symmetry about an unknown median." In *Random Walk, Sequential Analysis And Related Topics: A Festschrift in Honor of Yuan-Shih Chow*, 199–214. World Scientific.

Mira A (1999). "Distribution-free test for symmetry based on Bonferroni's measure." *Journal of Applied Statistics*, **26**(8), 959–972. doi: 10.1080/02664769921963, `https://doi.org/10.1080/02664769921963`.

Index

Available Test Statistics, *4*

B1 *(*TestStatistics*)*, 5
BHI *(*TestStatistics*)*, 5
BHK *(*TestStatistics*)*, 5

CH *(*TestStatistics*)*, 5
CM *(*TestStatistics*)*, 5

K2 *(*TestStatistics*)*, 5
K2U *(*TestStatistics*)*, 5
KS *(*TestStatistics*)*, 5

L2 *(*TestStatistics*)*, 5

M *(*TestStatistics*)*, 5
MGG *(*TestStatistics*)*, 5
MI *(*TestStatistics*)*, 5
MK *(*TestStatistics*)*, 5
MOI *(*TestStatistics*)*, 5
MOK *(*TestStatistics*)*, 5

NAI *(*TestStatistics*)*, 5
NAK *(*TestStatistics*)*, 5

rmixnorm, 2
rsl, 2

S1 *(*TestStatistics*)*, 5
S2 *(*TestStatistics*)*, 5
SGN *(*TestStatistics*)*, 5
symmetry, 3
symmetry-package *(*symmetry*)*, 3
symmetry_test, *3*, 3, *7*

T1 *(*TestStatistics*)*, 5
T2 *(*TestStatistics*)*, 5
Test Statistics, *4*
TestStatistics, *3*, 5

WCX *(*TestStatistics*)*, 5