Package ‘nortsTest’

July 27, 2020

Title Assessing Normality of Stationary Process

Version 1.0.0

Description Despite that several tests for normality in stationary processes have been proposed in the literature, consistent implementations of these tests in programming languages are limited. Four normality test are implemented. The Lobato and Velasco's, Epps, Psaradakis and Vavra, and the random projections tests for stationary process. Some other diagnostics such as, unit root test for stationarity, seasonal tests for seasonality, and arch effect test for volatility; are also performed. The package also offers residual diagnostic for linear time series models developed in several packages.

License LGPL

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.1.0

Depends R (>= 3.5.0), methods

Imports forecast, nortest, ggplot2, gridExtra, tseries, uroot, MASS, zoo

URL https://github.com/asael697/nortsTest

BugReports https://github.com/asael697/nortsTest/issues

Author Asael Alonzo Matamoros [aut, cre],
   Alicia Nieto-Reyes [aut],
   Rob Hyndman [ctb],
   Mitchell O'Hara-Wild [ctb],
   Trapletti A. [ctb]

Maintainer Asael Alonzo Matamoros <asael.alonzo@gmail.com>

Repository CRAN

Date/Publication 2020-07-27 11:50:03 UTC
nortsTest-package

R topics documented:

nortsTest-package .................................................. 2
arch.test ............................................................. 3
autoplot.ts .......................................................... 4
check_plot.ts ......................................................... 6
check_residuals.ts .................................................. 7
epps.statistic ....................................................... 8
epps.test ............................................................ 9
ggacf ................................................................. 11
gghist ............................................................... 11
ggnorm ............................................................... 12
ggpacf ............................................................... 13
Lm.test .............................................................. 14
lobato.statistic ..................................................... 15
lobato.test .......................................................... 16
normal.test ........................................................ 17
random.projection .................................................. 19
rp.sample ........................................................... 20
rp.test .............................................................. 21
seasonal.test ....................................................... 23
sieve.bootstrap ..................................................... 24
uroot.test ........................................................... 25
vavra.sample ......................................................... 26
vavra.test ........................................................... 28

Index 30

nortsTest-package 'Assessing Normality of Stationary Process.'

Description

Despite that several tests for normality in stationary processes have been proposed in the literature, consistent implementations of these tests in programming languages are limited. Four normality test are implemented. The Lobato and Velasco’s, Epps, Psaradakis and Vavra, and the random projections tests for stationary process. Some other diagnostics such as, unit root test for stationarity, seasonal tests for seasonality, and arch effect test for volatility; are also performed. The package also offers residual diagnostic for linear time series models developed in several packages.

Details

We present four main functions, for testing the hypothesis of normality in stationary process, the epps.test, lobato.test, rp.test, and varvra.test. Additionally, we provide functions for unit root, seasonality and ARCH effects tests for stationary, and other additional methods for visual checks using the ggplot2 and forecast packages.
References


---

**arch.test**

*The ARCH effect test function.*

**Description**

Performs the Pormanteau Q and Lagrange Multipliers test for homoscedasticity in a univariate stationary process. The null hypothesis (H0), is that the process is homoscedastic.

**Usage**

```
arch.test(y, arch="box", alpha=0.05, lag.max = 2)
```

**Arguments**

- `y` a numeric vector or an object of the `ts` class containing a stationary time series.
- `arch` A character string naming the desired test for checking stationarity. Valid values are "box" for the Ljung-Box, and "Lm" for the Lagrange Multiplier test. The default value is "box" for the Augmented Ljung-Box test.
- `alpha` Level of the test, possible values range from 0.01 to 0.1. By default `alpha = 0.05` is used
- `lag.max` an integer with the number of used lags.

**Details**

Several different tests are available: Performs Portmanteau Q and Lagrange Multiplier tests for the null hypothesis that the residuals of an ARIMA model are homoscedastic. The ARCH test is based on the fact that if the residuals (defined as $e(t)$) are heteroscedastic, the squared residuals ($e^2(t)$) are autocorrelated. The first type of test is to examine whether the squares of residuals are a sequence of white noise, which is called the Portmanteau Q test, and similar to the Ljung-Box test on the squared residuals. By default, `alpha = 0.05` is used to select the more likely hypothesis.
Value

a h.test class with the main results of unit root hypothesis test.

Author(s)

Asael Alonzo Matamoros

References


See Also

normal.test, seasonal.test, uroot.test

Examples

```r
# stationary ar process
y = arima.sim(100, model = list(ar = 0.3))
arch.test(y)
```

Description

`autoplot` takes an object of type `ts` or `mts` and creates a `ggplot` object suitable for usage with `stat_forecast`.

Usage

```r
## S3 method for class 'ts'
autoplot(
  object,
  series = NULL,
  xlab = "Time",
  ylab = deparse(substitute(object)),
  main = NULL,
  facets = FALSE,
  colour = FALSE,
  ...)
```

`autoplot.ts` Automatically create a `ggplot` for time series objects
# S3 method for class 'numeric'
autoplot(
  object, 
  series = NULL, 
  xlab = "Time", 
  ylab = deparse(substitute(object)), 
  main = NULL, 
  ...
)

# S3 method for class 'ts'
fortify(model, data, ...)

## Arguments

- **object**: Object of class “ts” or “mts”.
- **series**: Identifies the time series with a colour, which integrates well with the functionality of geom_forecast.
- **xlab**: A string with the plot’s x axis label. By default a NULL value.
- **ylab**: A string with the plot’s y axis label. By default a counts” value.
- **main**: A string with the plot’s title.
- **facets**: If TRUE, multiple time series will be faceted (and unless specified, colour is set to FALSE). If FALSE, each series will be assigned a colour.
- **colour**: If TRUE, the time series will be assigned a colour aesthetic
- **...**: Other plotting parameters to affect the plot.
- **model**: Object of class “ts” to be converted to “data.frame”.
- **data**: Not used (required for fortify method)

## Details

`fortify.ts` takes a ts object and converts it into a data frame (for usage with ggplot2).

## Value

None. Function produces a ggplot2 graph.

## Author(s)

Mitchell O’Hara-Wild

## See Also

`plot.ts`, `fortify`
Examples

```r
library(ggplot2)
autoplot(USAccDeaths)

lungDeaths <- cbind(mdeaths, fdeaths)
autoplot(lungDeaths)
autoplot(lungDeaths, facets=TRUE)
```

---

**check_plot.ts**  
*Generic function for a visual check of residuals in time series models*

**Description**

Generic function for a visual check of residuals in time series models, these methods are inspired in the `check.residuals` function provided by the `forecast` package.

**Usage**

```r
## S3 method for class 'ts'
check_plot(y, model = "", ...) 
```

**Arguments**

- `y`: a numeric vector or an object of the `ts` class containing a stationary time series.
- `model`: A string with the model name.
- `...`: Other plotting parameters to affect the plot.

**Value**

A graph object from `ggplot2`

**Author(s)**

Asael Alonzo Matamoros

**See Also**

`check_residuals`

**Examples**

```r
y = arima.sim(n=100, model = list(ar = 0.3))
check_plot(y)
```
Generic functions for checking residuals in time series models

Description

Generic function for residuals check analysis, these methods are inspired in the `check.residuals` function provided by the `forecast` package.

Usage

```r
## S3 method for class 'ts'
check_residuals(
  y,
  normality = "epps",
  unit_root = NULL,
  seasonal = NULL,
  arch = NULL,
  alpha = 0.05,
  plot = FALSE,
  ...
)
```

Arguments

- `y` Either a time series model, the supported classes are `arima0`, `Arima`, `sarima`, `fGarch`, or a time series (assumed to be residuals).
- `normality` A character string naming the desired test for checking Gaussian distribution. Valid values are "epps" for the Epps, "lobato" for Lobato and Velasco's, "vavras" for the Psaradakis and Vavra, "rp" for the random projections, "jb" for the Jarque and Beras, "ad" for Anderson Darling test, and "shapiro" for the Shapiro-Wilk's test. The default value is "epps" test.
- `unit_root` A character string naming the desired unit root test for checking stationarity. Valid values are "adf" for the Augmented Dickey-Fuller, "pp" for the Phillips-Perron, and "kpss" for Kwiatkowski, Phillips, Schmidt, and Shin. The default value is "adf" for the Augmented Dickey-Fuller test.
- `seasonal` A character string naming the desired unit root test for checking seasonality. Valid values are "ocsb" for the Osborn, Chui, Smith, and Birchenhall, "ch" for the Canova and Hansen, and "hegy" for Hylleberg, Engle, Granger, and Yoo. The default value is "ocsb" for the Osborn, Chui, Smith, and Birchenhall test.
- `arch` A character string naming the desired test for checking stationarity. Valid values are "box" for the Ljung-Box, and "Lm" for the Lagrange Multiplier test. The default value is "box" for the Augmented Ljung-Box test.
- `alpha` Level of the test, possible values range from 0.01 to 0.1. By default `alpha = 0.05` is used.
- `plot` A boolean value. If `TRUE`, will produce a time plot of the residuals, the corresponding ACF, and a histogram.
Other testing parameters

Details

The function performs a residuals analysis, it prints a unit root and seasonal test to check stationarity, and a normality test for checking Gaussian distribution. In addition, if the plot option is TRUE a time plot, ACF, and histogram of the series are presented.

Value

The function does not return any value

Author(s)

Asael Alonzo Matamoros

References


Examples

```
# Generating an stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
check_residuals(y, unit_root = "adf")
```

---

epps.statistic Estimates the Epps statistic

Description

Estimates the Epps statistic minimizing the quadratic loss of the process’ characteristic function in terms of the first two moments.

Usage

epps.statistic(y)

Arguments

`y` a numeric vector or an object of the `ts` class containing a stationary time series.
Details


Value

a real value with the Epps test's statistic.

Author(s)

Alicia Nieto-Reyes and Asael Alonzo Matamoros.

References


See Also

lobato.statistic

Examples

# Generating an stationary arma process
y = arima.sim(100,model = list(ar = 0.3))
epps.statistic(y)
Arguments

y a numeric vector or an object of the ts class containing a stationary time series.

Details


Value

a h.test class with the main results of the Epps hypothesis test. The h.test class have the following values:

- "epps" the Epps statistic
- "df" the test degrees freedoms
- "p.value" the p value
- "alternative" the alternative hypothesis
- "method" the used method
- "data.name" the data name.

Author(s)

Asael Alonzo Matamoros and Alicia Nieto-Reyes.

References


See Also

lobato.test

Examples

# Generating an stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
epps.test(y)
The `ggacf` function is used to plot the auto-correlation function for a univariate time series.

**Usage**

```r
ggacf(y, title = NULL)
```

**Arguments**

- `y`: a numeric vector or an object of the `ts` class containing a stationary time series.
- `title`: a string with the plot’s title.

**Value**

None.

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```r
x = rnorm(100)
ggacf(x)
```

---

The `gghist` function plots a histogram and density estimates using `ggplot`.

**Usage**

```r
gghist(y, title = NULL, xlab = NULL, ylab = "counts", bins, add.normal = TRUE)
```
Arguments

\begin{itemize}
\item \textbf{y} a numeric vector or an object of the \texttt{ts} class containing a stationary time series.
\item \textbf{title} a string with the plot's title.
\item \textbf{xlab} a string with the plot's x axis label. By default a NULL value
\item \textbf{ylab} a string with the plot's y axis label. By default a "counts" value
\item \textbf{bins} The number of bins to use for the histogram. Selected by default using the Friedman-Diaconis rule.
\item \textbf{add.normal} A boolean value. Add a normal density function for comparison, by default \texttt{add.normal = TRUE}.
\end{itemize}

Value

None.

Author(s)

Rob J Hyndman

Examples

\begin{verbatim}
x = rnorm(100)
gghist(x, add.normal = TRUE)
\end{verbatim}

Description

Plot the quantile-quantile plot and quantile-quantile line using ggplot.

Usage

\begin{verbatim}
ggnorm(y, title = NULL, add.normal = TRUE)
\end{verbatim}

Arguments

\begin{itemize}
\item \textbf{y} a numeric vector or an object of the \texttt{ts} class containing a stationary time series.
\item \textbf{title} a string with the plot's title.
\item \textbf{add.normal} Add a normal density function for comparison.
\end{itemize}

Value

None.
Authors
Asael Alonzo Matamoros

Examples
\[ \text{x = rnorm}(100) \]
\[ \text{ggnorm(x)} \]

Description
Plot of the partial autocorrelation function for a univariate time series.

Usage
\[ \text{ggpacf}(y, \text{title} = \text{NULL}) \]

Arguments
- \text{y}: a numeric vector or an object of the \text{ts} class containing a stationary time series.
- \text{title}: a string with the plot’s title.

Value
None.

Author(s)
Mitchell O’Hara-Wild and Asael Alonzo Matamoros

Examples
\[ \text{x = rnorm}(100) \]
\[ \text{ggpacf(x)} \]
**Lm.test**

*The Lagrange Multiplier test for arch effect.*

**Description**

Performs the Lagrange Multipliers test for homoscedasticity in a stationary process. The null hypothesis (H0), is that the process is homoscedastic.

**Usage**

```r
Lm.test(y, lag.max = 2, alpha = 0.05)
```

**Arguments**

- `y`: a numeric vector or an object of the `ts` class containing a stationary time series.
- `lag.max`: an integer with the number of used lags.
- `alpha`: Level of the test, possible values range from 0.01 to 0.1. By default alpha = 0.05 is used.

**Details**

The Lagrange Multiplier test proposed by Engle (1982) fits a linear regression model for the squared residuals and examines whether the fitted model is significant. So the null hypothesis is that the squared residuals are a sequence of white noise, namely, the residuals are homoscedastic.

**Value**

A h.test class with the main results of the Lagrange multiplier hypothesis test. The h.test class have the following values:

- "Lm" The lagrange multiplier statistic
- "df" The test degrees freedoms
- "p.value" The p value
- "alternative" The alternative hypothesis
- "method" The used method
- "data.name" The data name.

**Author(s)**

A. Trapletti and Asael Alonzo Matamoros

**References**


See Also

arch.test

Examples

# generating an stationary arma process
y = arima.sim(100,model = list(ar = 0.3))
Lm.test(y)

lobato.statistic(y,c = 1)

Arguments

y  
a numeric vector or an object of the ts class containing a stationary time series.
c  
a positive real value that identifies the total amount of values used in the cumulative sum.

Details

This function is the equivalent of GestadisticoVn of Nieto-Reyes, A., Cuesta-Albertos, J. & Gamboa, F. (2014).

Value

A real value with the Lobato and Velasco test's statistic.

Author(s)

Alicia Nieto-Reyes and Asael Alonzo Matamoros.

References


lobato.test

The Lobato and Velasco’s Test for normality

Description

Performs the Lobato and Velasco’s test for normality. The null hypothesis (H0), is that the given data follows a Gaussian process.

Usage

lobato.test(y, c = 1)

Arguments

- `y`: a numeric vector or an object of the ts class containing a stationary time series.
- `c`: a positive real value that identifies the total amount of values used in the cumulative sum.

Details

This test proves a normality assumption in correlated data employing the skewness-kurtosis test statistic, but studentized by standard error estimates that are consistent under serial dependence of the observations. The test was proposed by Lobato, I., & Velasco, C. (2004) and implemented by Nieto-Reyes, A., Cuesta-Albertos, J. & Gamboa, F. (2014).

Value

A h.test class with the main results of the Lobato and Velasco’s hypothesis test. The h.test class have the following values:

- "lobato" The Lobato and Velasco’s statistic
- "df" The test degrees freedoms
- "p.value" The p value
- "alternative" The alternative hypothesis
- "method" The used method
- "data.name" The data name.

See Also

epps.statistic

Examples

# Generating an stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
lobato.statistic(y, 3)
normal.test

Author(s)

Asael Alonzo Matamoros and Alicia Nieto-Reyes.

References


See Also

lobato.statistic.epps.test

Examples

# Generating an stationary arma process
y = arima.sim(100,model = list(ar = 0.3))
lobato.test(y)

normal.test The normality test for stationary process

Description

Perform a normality test. The null hypothesis (H0) is that the given data follows a stationary Gaussian process.

Usage

normal.test(y,normality="epps",alpha=0.05)

Arguments

y a numeric vector or an object of the ts class containing a stationary time series.

normality A character string naming the desired test for checking normality. Valid values are "epps" for the Epps, "lobato" for Lobato and Velasco’s, "vavra" for the Psaradakis and Vavra, "rp" for the random projections, "jb" for the Jarque and Beras, "ad" for Anderson Darling test, and "shapiro" for the Shapiro-Wilk’s test. The default value is "epps" test.

alpha Level of the test, possible values range from 0.01 to 0.1. By default alpha = 0.05 is used.
Details

Several different tests are available: "lobato", "epps", "vavras" and "rp" test are for testing normality in stationary process. "jb", "ad", and "shapiro" tests are for numeric data. In all cases, the alternative hypothesis is that y follows a Gaussian process. By default, alpha = 0.05 is used to select the more likely hypothesis.

Value

An h.test class with the main results of normal hypothesis test.

Author(s)

Asael Alonzo Matamoros

References


See Also

uroot.test,seasonal.test

Examples

# stationary ar process
y = arima.sim(100,model = list(ar = 0.3))
normal.test(y) # epps test

# normal random sample
y = rnorm(100)
normal.test(y,normality = "shapiro")

# exponential random sample
y = rexp(100)
normal.test(y,normality = "ad")
random.projection

Generate a random projection

Description

Generates a random projection of a univariate stationary stochastic process. Using a beta(shape1,shape2) distribution.

Usage

random.projection(y,shape1,shape2,seed = NULL)

Arguments

y a numeric vector or an object of the ts class containing a stationary time series.
shape1 an optional real value with the first shape parameters of the beta distribution.
shape2 an optional real value with the second shape parameters of the beta distribution.
seed An optional seed to use.

Details


Value

a real vector with the projected stochastic process.

Author(s)

Alicia Nieto-Reyes and Asael Alonzo Matamoros

References


See Also

lobato.test epps.test
Examples

```r
# Generating an stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
rp.test(y)
```

---

**rp.sample**  
*Generates a test statistics sample of random projections*

**Description**

Generates a sample of test statistics using $k$ independent random projections of a stationary process. The first half values of the sample, are estimated using a Lobato and Velasco’s statistic test. The last half values with an Epps statistic test.

**Usage**

```r
rp.sample(y, k = 2, pars1 = c(100, 1), pars2 = c(2, 7), seed = NULL)
```

**Arguments**

- **y**: a numeric vector or an object of the `ts` class containing a stationary time series.
- **k**: an integer with the number of random projections to be used, by default $k = 2$.
- **pars1**: an optional real vector with the shape parameters of the beta distribution used for the odd number random projection. By default, `pars1 = c(100, 1)` where, `shape1 = 100` and `shape2 = 1`.
- **pars2**: an optional real vector with the shape parameters of the beta distribution used for the even number random projection. By default, `pars2 = c(2, 7)` where, `shape1 = 2` and `shape2 = 7`.
- **seed**: An optional seed to use.

**Details**

The `rp.sample` function generates $k$ independent random projections of the process. A Lobatos and Velasco’s test is applied to the first half of the projections. And an Epps test for the other half.

For generating the $k$ random projections a beta distribution is used. By default a beta$(shape1 = 100, shape = 1)$ and a beta$(shape1 = 2, shape = 7)$ are used to generate the odd and even projections respectively. For using a different parameter set, change `pars1` or `pars2` values.

The test was proposed by Nieto-Reyes, A., Cuesta-Albertos, J. & Gamboa, F. (2014).

**Value**

A list with 2 real value vectors:

- "lobato"A vector with the Lobato and Velasco’s statistics sample
- "epps"A vector with the Epps statistics sample.
**rp.test**

**Author(s)**
Alicia Nieto-Reyes and Asael Alonzo Matamoros

**References**


**See Also**
lobato.test epps.test

**Examples**

```r
# Generating a stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
rp.test(y)
```

---

**Description**
Performs the random projection test for normality. The null hypothesis (H0) is that the given data follows a stationary Gaussian process, and k is the number of used random projections.

**Usage**

```r
rp.test(y, k = 64, FDR = FALSE, pars1 = c(100, 1), pars2 = c(2, 7), seed = NULL)
```

**Arguments**

- `y` a numeric vector or an object of the ts class containing a stationary time series.
- `k` an integer with the number of random projections to be used, by default `k = 2`.
- `FDR` a logical value for mixing the p-values using a dependent False discovery rate method. By default `FDR = FALSE`.
- `pars1` an optional real vector with the shape parameters of the beta distribution used for the odd number random projection. By default, `pars1 = c(100, 1)` where, `shape1 = 100` and `shape2 = 1`. 
pars2 an optional real vector with the shape parameters of the beta distribution used for the even number random projection. By default, pars2 = c(2, 7) where, shape1 = 2 and shape2 = 7.

seed An optional seed to use.

Details

The random projection test generates k independent random projections of the process. A Lobato and Velasco’s test are applied to the first half of the projections, and an Epps test for the other half. By default, a Monte Carlo p-value estimate is used for mixing the tests. A False discovery rate can be used for mixing by setting FDR = TRUE.

For generating the k random projections a beta distribution is used. By default a beta(shape1 = 100, shape = 1) and a beta(shape1 = 2, shape = 7) are used to generate the odd and even projections respectively. For using a different parameter set, change pars1 or pars2.

The test was proposed by Nieto-Reyes, A., Cuesta-Albertos, J. & Gamboa, F. (2014).

Value

a h.test class with the main results of the Epps hypothesis test. The h.test class have the following values:

- "k" The number of used projections
- "lobato" The average Lobato and Velasco’s test statistics of the k projected samples
- "epps" The average Epps test statistics of the k projected samples
- "p.value" The mixed p value
- "alternative" The alternative hypothesis
- "method" The used method: rp.test
- "data.name" The data name.

Author(s)

Asael Alonzo Matamoros and Alicia Nieto-Reyes.

References


See Also

lobato.test epps.test
seasonal.test

Examples

```r
# Generating an stationary arma process
y <- arima.sim(100,model = list(ar = 0.3))
rp.test(y)
```

### seasonal.test

**The Seasonal unit root tests function**

#### Description

Perform a seasonal unit root test to check seasonality in a linear stochastic process

#### Usage

```r
seasonal.test(y,seasonal="ocsb",alpha=0.05)
```

#### Arguments

- `y`: a numeric vector or an object of the `ts` class containing a stationary time series.
- `seasonal`: A character string naming the desired seasonal unit root test for checking seasonality. Valid values are "ocsb" for the Osborn, Chui, Smith, and Birchenhall, "ch" for the Canova and Hansen, and "hegy" for Hylleberg, Engle, Granger, and Yoo. The default value is "ocsb" for the Osborn, Chui, Smith, and Birchenhall test.
- `alpha`: Level of the test, possible values range from 0.01 to 0.1. By default `alpha = 0.05` is used

#### Details

Several different tests are available: In the kpss test, the null hypothesis that `y` has a stationary root against a unit-root alternative. In the two remaining tests, the null hypothesis is that `y` has a unit root against a stationary root alternative. By default, `alpha = 0.05` is used to select the more likely hypothesis.

#### Value

A `h.test` class with the main results of unit root hypothesis test.

#### Author(s)

Asael Alonzo Matamoros
References


See Also

*normal.test*, *uroot.test*

Examples

```r
# stationary ar process
y = ts(rnorm(100), frequency = 6)
seasonal.test(y)
```

---

**sieve.bootstrap**  
*Generates a sieve bootstrap sample*

**Description**

The function generates a sieve bootstrap sample for a univariate stochastic process.

**Usage**

```r
sieve.bootstrap(y, reps = 1000, pmax = NULL, h = 100, seed = NULL)
```

**Arguments**

- `y`  
a numeric vector or an object of the `ts` class containing a stationary time series.
- `reps`  
an integer with the total bootstrap repetitions.
- `pmax`  
an integer with the max considered lags for the generated AR(p) process. By default, `pmax = NULL`.
- `h`  
an integer with the first burn-in sieve bootstrap replicates.
- `seed`  
An optional `seed` to use.

**Details**

Simulates bootstrap samples for the stochastic process y, using a stationary auto-regressive model of order “pmax”, AR(pmax). If pmax = NULL (default), the function estimates the process maximum lags using an AIC as a model selection criteria.
uroot.test

Value
A matrix or reps row and n columns, with the sieve bootstrap sample and n the time series length.

Author(s)
Asael Alonzo Matamoros

References

See Also
lobato.test,epps.test

Examples

# Generating an stationary arma process
y = arima.sim(100,model = list(ar = 0.3))
M = sieve.bootstrap(y)

uroot.test # The Unit root tests function

Description
Perform a unit root test to check stationarity in a linear stochastic process.

Usage

uroot.test(y,unit_root="adf",alpha=0.05)

Arguments

y
a numeric vector or an object of the ts class containing a stationary time series.

unit_root
A character string naming the desired unit root test for checking stationarity. Valid values are "adf" for the Augmented Dickey-Fuller, "pp" for the Phillips-Perron, "kpss" for Kwiatkowski, Phillips, Schmidt, and Shin, and "box" for the Ljung-Box. The default value is "adf" for the Augmented Dickey-Fuller test.

alpha
Level of the test, possible values range from 0.01 to 0.1. By default alpha = 0.05 is used.

Details
Several different tests are available: In the kpss test, the null hypothesis that y has a stationary root against a unit-root alternative. In the two remaining tests, the null hypothesis is that y has a unit root against a stationary root alternative. By default, alpha = 0.05 is used to select the more likely hypothesis.
Value

A h.test class with the main results of unit root hypothesis test.

Author(s)

Asael Alonzo Matamoros and A. Trapletti

References


See Also

`normal.test`, `seasonal.test`

Examples

```r
# stationary ar process
y = arima.sim(100, model = list(ar = 0.3))
uroot.test(y)

# a random walk process
y = cumsum(y)
uroot.test(y, unit_root = "pp")
```

Description

Generates a sieve bootstrap sample of the Anderson-Darling statistic test.

Usage

```r
vavra.sample(y, reps = 1000, h = 100, seed = NULL)
```
Arguments

- **y**: a numeric vector or an object of the `ts` class containing a stationary time series.
- **reps**: an integer with the total bootstrap repetitions.
- **h**: an integer with the first burn-in sieve bootstrap replicates.
- **seed**: An optional seed to use.

Details

The Vavra test approximates the empirical distribution function of the Anderson-Darlings statistic, using a sieve bootstrap approximation. The test was proposed by Psaradakis, Z. & Vavra, M (2017). This function is the equivalent of `xarsieve` of Psaradakis, Z. & Vavra, M (2017).

Value

A numeric array with the Anderson Darling sieve bootstrap sample

Author(s)

Asael Alonzo Matamoros.

References


See Also

epps.statistic lobato.statistic

Examples

```r
# Generating a stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
adbs = vavra.sample(y)
mean(adbs)
```
vavra.test

The Psaradakis and Vavra test for normality

Description

Performs the Psaradakis and Vavra distance test for normality. The null hypothesis (H0), is that the given data follows a Gaussian process.

Usage

vavra.test(y, reps = 1000, h = 100, seed = NULL)

Arguments

y a numeric vector or an object of the ts class containing a stationary time series.
reps an integer with the total bootstrap repetitions.
h an integer with the first burn-in sieve bootstrap replicates.
seed An optional seed to use.

Details

The Psaradakis and Vavra test approximates the empirical distribution function of the Anderson Darling’s statistic, using a sieve bootstrap approximation. The test was proposed by Psaradakis, Z. & Vavra, M (2017).

Value

a h.test class with the main results of the Epps hypothesis test. The h.test class have the following values:

- "bootstrap A" The sieve bootstrap A statistic
- "p.value" The p value
- "alternative" The alternative hypothesis
- "method" The used method
- "data.name" The data name.

Author(s)

Asael Alonzo Matamoros.

References


See Also

lobato.test, epps.test

Examples

# Generating an stationary arma process
y = arima.sim(100, model = list(ar = 0.3))
vavra.test(y)
Index

arch.test, 3, 15
autoplot.numeric (autoplot.ts), 4
autoplot.ts, 4

check_plot (check_plot.ts), 6
check_plot.ts, 6
check_residuals (check_residuals.ts), 7
check_residuals.ts, 7
epps.statistic, 8, 16, 27
epps.test, 9, 17, 19, 21, 22, 25, 29

fortify, 5
fortify.ts (autoplot.ts), 4
gem_forecast, 5
ggacg, 11
gghist, 11
ggnorm, 12
ggapacg, 13

Lm.test, 14
lobato.statistic, 9, 15, 17, 27
lobato.test, 10, 16, 19, 21, 22, 25, 29

normal.test, 4, 17, 24, 26
nortsTest (nortsTest-package), 2
nortsTest-package, 2

plot.ts, 5
random.projection, 19
rp.sample, 20
rp.test, 21

seasonal.test, 4, 18, 23, 26
seed, 19, 20, 22, 24, 27, 28
sieve.bootstrap, 24

uroot.test, 4, 18, 24, 25
vavra.sample, 26
vavra.test, 28