Package 'funtimes'

August 8, 2015

Type Package

Title Functions for Time Series Analysis

Version 2.0

Date 2015-08-08

Author Vyacheslav Lyubchich <lyubchic@cbl.umces.edu>;
Yulia R. Gel <ygl@utdallas.edu>;
Xingyu Wang <x263wang@uwaterloo.ca>

Maintainer Vyacheslav Lyubchich <lyubchic@cbl.umces.edu>

Depends Jmisc, R (>= 3.0.0)

License GPL (>= 2)

Description Includes non-parametric estimators and tests for time series analysis. The functions allow to test for presence of possibly non-monotonic trends and for synchronism of trends in multiple time series, using modern bootstrap techniques and robust non-parametric difference-based estimators.

NeedsCompilation no

Repository CRAN

Date/Publication 2015-08-08 17:39:20

R topics documented:

`CExpandSlideCluster` *Slide-level time series cluster expansion*

Description

This is an auxillary function to expand a slide-level time series cluster, based on Ciampi et al. (2010).

Usage

```
CExpandSlideCluster(u, Xuncl, Alpha, Beta, Delta, Theta)
```

Arguments

`u`	a time series vector — a seed to expand the cluster.
`Xuncl`	a time series vector (of the same length as `u`) or a matrix (time series in columns) containing unclustered time series.
`Alpha`	lower limit of the time series domain.
`Beta`	upper limit of the time series domain.
`Delta`	closeness parameter, a real value in [0,1].
`Theta`	connectivity parameter, a real value in [0,1].

Value

A vector of logical values indicating which time series in `Xuncl` should be included in the slide-level cluster with `u`.

Author(s)

Vyacheslav Lyubchich

References

Ciampi, A., Appice, A. and Malerba, D. (2010) Discovering trend-based clusters in spatially distributed data streams. In *International Workshop of Mining Ubiquitous and Social Environments*, pages 107–122.

See Also

`CNeighbor`, `CHomogeneity`, `CSlideCluster`, `CExpandWindowCluster`, `CWindowCluster`.

Examples

```
set.seed(123)
u <- rnorm(10)
Xuncl <- matrix(rt(50, 5), 10, 5)
Alpha <- min(cbind(u,Xuncl))
Beta <- max(cbind(u,Xuncl))
CExpandSlideCluster(u, Xuncl, Alpha, Beta, Delta=0.15, Theta=0.8)
```

CExpandWindowCluster *Window-level time series cluster expansion*

Description

This is an auxillary function to expand a window-level time series cluster, based on Ciampi et al. (2010).

Usage

```
CExpandWindowCluster(e, Euncl)
```

Arguments

e	a vector of logical values identifying which time series among Euncl were clustered together with e over at least w*Epsilon slides within a window (see Definition 7 by Ciampi et al., 2010). This is a seed for window-level clustering.
Euncl	a square matrix identifying the binary window cluster relation for yet unclustered time series.

Value

A vector of logical values indicating which time series in Euncl should be included in the window-level cluster with e.

Author(s)

Vyacheslav Lyubchich

References

Ciampi, A., Appice, A. and Malerba, D. (2010) Discovering trend-based clusters in spatially distributed data streams. In *International Workshop of Mining Ubiquitous and Social Environments*, pages 107–122.

See Also

CNeighbor, CHomogeneity, CSlideCluster, CExpandSlideCluster, CWindowCluster.

Examples

```
set.seed(123)
e <- sample(c(TRUE, FALSE), 5, replace=TRUE)
Euncl <- matrix(sample(c(TRUE, FALSE), 5, replace=TRUE), 5, 5)
CExpandWindowCluster(e, Euncl)
```

CHomogeneity *Time series cluster homogeneity*

Description

This is an auxillary function to check homogeneity of time series cluster, based on Definition 4 by Ciampi et al. (2010).

Usage

```
CHomogeneity(Bu, Bv, Alpha, Beta, Delta)
```

Arguments

Bu	bucket of time series already included in the cluster.
Bv	bucket of time series (neighbors) for potential inclusion in the cluster.
Alpha	lower limit of the time series domain.
Beta	upper limit of the time series domain.
Delta	closeness parameter, a real value in [0,1].

Value

A logical value indicating whether time series in Bu and Bv form a homogeneous cluster.

Author(s)

Vyacheslav Lyubchich

References

Ciampi, A., Appice, A. and Malerba, D. (2010) Discovering trend-based clusters in spatially distributed data streams. In *International Workshop of Mining Ubiquitous and Social Environments*, pages 107–122.

See Also

CNeighbor, CExpandSlideCluster, CSlideCluster, CExpandWindowCluster, CWindowCluster.

Examples

```
Bu <- rnorm(10)
Bv <- rnorm(10)
Alpha <- min(c(Bu,Bv))
Beta <- max(c(Bu,Bv))
CHomogeneity(Bu, Bv, Alpha, Beta, Delta=0.5)
```

CNeighbor *Neighborhood of time series*

Description

This is an auxillary function to identify which time series in Bv are E_δ^θ-neighbors of Bu, based on Definition 2 by Ciampi et al. (2010).

Usage

```
CNeighbor(Bu, Bv, Alpha, Beta, Delta, Theta)
```

Arguments

Bu	a time series vector for which the neighborhood is investigated.
Bv	a time series vector (of the same length as Bu) or a matrix (time series in columns) containing potential neighbors.
Alpha	lower limit of the time series domain.
Beta	upper limit of the time series domain.
Delta	closeness parameter, a real value in [0,1].
Theta	connectivity parameter, a real value in [0,1].

Value

A vector of logical values indicating which time series in Bv are E_δ^θ-neighbors of Bu.

Author(s)

Vyacheslav Lyubchich

References

Ciampi, A., Appice, A. and Malerba, D. (2010) Discovering trend-based clusters in spatially distributed data streams. In *International Workshop of Mining Ubiquitous and Social Environments*, pages 107–122.

See Also

CHomogeneity, CExpandSlideCluster, CSlideCluster, CExpandWindowCluster, CWindowCluster.

Examples

```
Bu <- rnorm(10)
Bv <- rnorm(10)
Alpha <- min(c(Bu,Bv))
Beta <- max(c(Bu,Bv))
CNeighbor(Bu, Bv, Alpha, Beta, Delta=0.5, Theta=0.8)
```

`CSlideCluster` *Slide-level time series clustering*

Description

This function clusters time series at a slide level, based on the Algorithm 1 of Ciampi et al. (2010).

Usage

```
CSlideCluster(X, Alpha = NULL, Beta = NULL, Delta = NULL, Theta = 0.8)
```

Arguments

`X`	a matrix of time series observed within a slide (time series in columns).
`Alpha`	lower limit of the time series domain. Default is `quantile(X)[2]-1.5*(quantile(X)[4]-quantile(X)[2])`.
`Beta`	upper limit of the time series domain. Default is `quantile(X)[2]+1.5*(quantile(X)[4]-quantile(X)[2])`.
`Delta`	closeness parameter, a real value in [0,1]. Default is `0.1*(Beta-Alpha)`.
`Theta`	connectivity parameter, a real value in [0,1]. Default is 0.8.

Value

A vector of length `dim(X)[2]` with cluster labels.

Author(s)

Vyacheslav Lyubchich

References

Ciampi, A., Appice, A. and Malerba, D. (2010) Discovering trend-based clusters in spatially distributed data streams. In *International Workshop of Mining Ubiquitous and Social Environments*, pages 107–122.

See Also

`CNeighbor`, `CHomogeneity`, `CExpandSlideCluster`, `CExpandWindowCluster`, `CWindowCluster`.

Examples

```
set.seed(123)
X <- matrix(rnorm(50), 10, 5)
CSlideCluster(X)
```

`CWindowCluster` *Window-level time series clustering*

Description

This function clusters time series at a window level, based on the Algorithm 2 of Ciampi et al. (2010).

Usage

```
CWindowCluster(X, Alpha = NULL, Beta = NULL, Delta = NULL, Theta = 0.8,
  p, w, s, Epsilon = 1)
```

Arguments

`X`	a matrix of time series to be clustered (time series in columns).
`Alpha`	lower limit of the time series domain, passed to `CSlideCluster`.
`Beta`	upper limit of the time series domain, passed to `CSlideCluster`.
`Delta`	closeness parameter, passed to `CSlideCluster`.
`Theta`	connectivity parameter, passed to `CSlideCluster`.
`p`	number of layers (time series observations) in each slide.
`w`	number of slides in each window.
`s`	step to shift a window. Possible values include `p` (overlapping windows) or `w` (non-overlapping windows).
`Epsilon`	a real value in [0,1] used to identify each pair of time series that are clustered together over at least `w*Epsilon` slides within a window (see Definition 7 by Ciampi et al., 2010). Default is 1.

Details

This is the upper-level function for time series clustering. It exploits the function `CSlideCluster` to cluster time series within each slide based on closeness and homogeneity measures. Then, it uses slide-level cluster assignments to cluster time series within each window.

The total length of time series (number of levels, i.e., `nrow(X)`) should be divisible by `p`.

Value

A vector (if `X` contains only one window) or matrix with cluster labels for each time series (columns) and window (rows).

Author(s)

Vyacheslav Lyubchich

References

Ciampi, A., Appice, A. and Malerba, D. (2010) Discovering trend-based clusters in spatially distributed data streams. In *International Workshop of Mining Ubiquitous and Social Environments*, pages 107–122.

See Also

CNeighbor, CHomogeneity, CExpandSlideCluster, CExpandWindowCluster, CSlideCluster.

Examples

```
#For example, weekly data come in slides of 4 weeks (1 month)
p <- 4 #number of layers in each slide (data come in a slide)

#We want to analyze the trend clusters within a window of 1 year
w <- 13 #number of slides in each window
s <- w  #step to shift a window

#Simulate 26 autoregressive time series with two years of weekly data (52*2 weeks)
N <- 26
T <- 2*p*w

set.seed(123)
phi <- c(0.5) #parameter of autoregression
X <- sapply(1:N, function(x) arima.sim(n=T+100,
  list(order=c(length(phi),0,0),ar=phi)))[101:(T+100),]
colnames(X) <- paste("TS", c(1:dim(X)[2]), sep="")

tmp <- CWindowCluster(X, Delta=NULL, Theta=0.8, p=p, w=w, s=s, Epsilon=1)

#Time series were simulated with the same parameters, but based on the clustering parameters,
#not all time series join the same cluster. We can plot the main cluster for each window, and
#time series out of the cluster:
par(mfrow=c(2,2))
ts.plot(X[1:(p*w),tmp[1,]==1], ylim=c(-4,4),
  main="Main time series cluster in window 1")
ts.plot(X[1:(p*w),tmp[1,]!=1], ylim=c(-4,4),
  main="Time series out of the main cluster in window 1")
ts.plot(X[(p*w+1):(2*p*w),tmp[2,]==1], ylim=c(-4,4),
  main="Main time series cluster in window 2")
ts.plot(X[(p*w+1):(2*p*w),tmp[2,]!=1], ylim=c(-4,4),
  main="Time series out of the main cluster in window 2")
```

HVK *HVK estimator*

Description

Estimates coefficients in non-parametric autoregression using the difference-based approach by Hall and Van Keilegom (2003).

Usage

```
HVK(X, m1=NULL, m2=NULL, ar.order=1)
```

Arguments

`X`	univariate time series. Missing values are not allowed.
`m1, m2`	subsidiary smoothing parameters. Default `m1 = round(length(X)^(0.1))`, `m2 = round(length(X)^(0.5))`.
`ar.order`	order of the non-parametric autoregression (specified by user).

Details

First, autocovariances are estimated (formula (2.6) by Hall and Van Keilegom, 2003):

$$\hat{\gamma}(0) = \frac{1}{m_2 - m_1 + 1} \sum_{m=m_1}^{m_2} \frac{1}{2(n-m)} \sum_{i=m+1}^{n} \{(D_m X)_i\}^2,$$

$$\hat{\gamma}(j) = \hat{\gamma}(0) - \frac{1}{2(n-j)} \sum_{i=j+1}^{n} \{(D_j X)_i\}^2,$$

where n=`length(X)` is sample size, D_j is a difference operator such that $(D_j X)_i = X_i - X_{i-j}$. Then, Yule-Walker method is used to derive autoregression coefficients.

Value

Vector of length `ar.order` with estimated autoregression coefficients.

Author(s)

Yulia R. Gel, Vyacheslav Lyubchich, Xingyu Wang

References

Hall, P. and Van Keilegom, I. (2003) Using difference-based methods for inference in nonparametric regression with time series errors. *J. R. Statist. Soc. B* 65, Part 2, 443–456.

Examples

```
X <- arima.sim(n=300, list(order=c(1,0,0), ar=c(0.6)))
HVK(as.vector(X), ar.order=1)
```

`i.tails` *Interval-based tails comparison*

Description

This function compares tails of two sample distributions using an interval-based approach.

Usage

```
i.tails(x, y, d=NULL)
```

Arguments

`x,y`	vectors of the same length (preferably). Tail in y is compared against the tail in x. $x \geq d$ is used to obtain interval width.
`d`	a threshold defining the tail. The threshold is the same for both x and y. Default is `quantile(x, probs=0.99)`.

Details

Sturges' formula is used to calculate number of intervals and interval width for $x \geq d$. The same interval width is applied to $y \geq d$. The tails, $x \geq d$ and $y \geq d$, are divided into the intervals, and number of y-values is compared with the number of x-values within each interval.

Value

A list with two elements:

`Nk`	vector that tells how many more y-values compared with x-values there are within each interval.
`Ck`	vector of intervals' centers.

Author(s)

Calvin Chu, Yulia R. Gel, Vyacheslav Lyubchich

See Also

`q.tails`

Examples

```
x <- rnorm(1000); y <- rt(1000, 5)
i.tails(x, y)
```

`q.tails` *Quantile-based tails comparison*

Description

This function compares right tails of two sample distributions using a quantile-based approach.

Usage

```
q.tails(x, y, q=0.99)
```

Arguments

`x,y`	vectors of the same length (preferably). Tail in y is compared against the tail in x.
`q`	a threshold defining the tail for both x and y, set as a quantile. Default is 99th percentile.

Details

Sturges' formula is used to calculate number of intervals (k) to split the upper $100(1-q)\%$ portion of x and y (the right tail). Then, the tail is divided into equally-filled intervals with a quantile step $d = (1-q)/k$. The difference between intervals' centers obtained from x and y is reported as Pk.

Value

A list with two elements:

`d`	the quantile step.
`Pk`	vector of differences of intervals' centers.

Author(s)

Vyacheslav Lyubchich, Yulia R. Gel

References

Soliman, M., Lyubchich, V., Gel, Y. R., Naser, D. and Esterby, S. (2015) Evaluating the impact of climate change on dynamics of house insurance claims. In V. Lakshmanan et al. (eds.) *Machine Learning and Data Mining Approaches to Climate Science*, pages 175–183. Springer, Cham.

Soliman, M., Naser, D., Lyubchich, V., Gel, Y. R. and Esterby, S. (2014) Evaluating the impact of climate change on dynamics of house insurance claims. In *Proceedings of the 4th International Workshop on Climate Informatics*.

See Also

`i.tails`

Examples

```
x <- rnorm(1000); y <- rt(1000, 5)
q.tails(x, y)
```

`sync.test` *Time series trend synchronism test*

Description

Non-parametric test for synchronism of parametric trends in multiple time series. The method tests whether N observed time series exhibit the same trend of some pre-specified smooth parametric form. Current version of the code assumes a linear trend $f(\theta, t) = \theta_0 + \theta_1 t$, where θ's are trend parameters, t is a regular sequence on the interval (0,1]. Note that an assumption of 'no trend' among all N observed time series is a subcase of a linear trend.

Usage

```
sync.test(X, B=1000, Window=NULL, q=NULL, j=NULL, ar_order=NULL, BIC=TRUE, robust = TRUE)
```

Arguments

`X`	$T \times N$ matrix with multiple time series in columns, where T is length of the time series, N is number of the time series. Missing values are not allowed.
`B`	number of bootstrap resamples.
`Window`	scalar or N-vector with lengths of the local windows (factors). If only one value is set, the same `Window` is applied to each time series. N-vector specifies a particular window for each time series. If no `Window` is specified, the automatic algorithm for optimal window selection is performed as a default option (see details).
`q`	scalar from 0 to 1 to define the set of possible windows `T*q^j` and to automatically select an optimal window for each time series. Default is $3/4$. This argument is ignored if `Window` is set by user.
`j`	numeric vector to define the set of possible windows `T*q^j` and to automatically select an optimal window for each time series. Default is `c(8:11)`. This argument is ignored if `Window` is set by user.
`ar_order`	order of autoregressive filter when `BIC = FALSE`, or the maximal order for BIC-based filtering. Default is `floor(10*log10(T))`. `ar_order` can be a scalar or N-vector. If scalar, the same `ar_order` is applied to each time series. N-vector specifies a separate `ar_order` for each time series.
`BIC`	logical value indicates whether the order of autoregressive filter should be selected by Bayesian information criterion (BIC). If `TRUE` (default), models of orders 1,...,`ar_order` or 1,...,`floor(10*log10(T))` are considered, depending on whether `ar_order` is defined or not.
`robust`	logical value indicates whether to use robust estimates of autoregression coefficients using `HVK` function (default), or to use Yule-Walker estimates delivered by `ar` function.

Details

Currently this function allows to test only for a common parametric linear trend, including the case of non-significant or zero-slope trend (check the function output for significance of the coefficient for t).

Arguments `Window`, `j` and `q` are used to set windows for the local regression. Current version of the function assumes two options: (1) user specifies one fixed window for each time series using the argument `Window` (if `Window` is set, `j` and `q` are ignored), and (2) user specifies a set of windows by `j` and `q` to apply this set to each time series and to select an optimal window using a heuristic m-out-of-n subsampling algorithm (Bickel and Sakov, 2008). The option of selecting windows automatically for some of the time series, while for other time series the window is fixed, is not available yet. If none of these three arguments is set, default `j` and `q` are used. Values `T*q^j` are mapped to the largest previous integer, then only those greater than 2 are used.

Value

A list of class htest containing the following components:

`method`	name of the method.
`data.name`	name of the data.
`statistic`	value of the test statistic.
`p.value`	p-value of the test.
`alternative`	alternative hypothesis.
`estimate`	list with elements `common_trend_estimates`, `ar_order_used`, `Window_used` and `all_considered_windows`. The latter is a table with bootstrap and asymptotic test results for all considered windows, i.e., without adaptive selection of the local window.

Author(s)

Yulia R. Gel, Vyacheslav Lyubchich, Xingyu Wang

References

Bickel, P. J. and Sakov, A. (2008) On the choice of m in the m out of n bootstrap and confidence bounds for extrema. *Statistica Sinica* 18, 967–985.

Lyubchich, V., Gel, Y. R. and El-Shaarawi, A. (2013) On detecting non-monotonic trends in environmental time series: a fusion of local regression and bootstrap. *Environmetrics* 24, 209–226.

Wang, L., Akritas, M. G. and Van Keilegom, I. (2008) An ANOVA-type nonparametric diagnostic test for heteroscedastic regression models. *Journal of Nonparametric Statistics* 20(5), 365–382.

Wang, L. and Van Keilegom, I. (2007) Nonparametric test for the form of parametric regression with time series errors. *Statistica Sinica* 17, 369–386.

See Also

`HVK`, `WAVK`, `wavk.test`.

Examples

```
# Fix seed for reproduceable simulations.
set.seed(123)

# Simulate two autoregressive time series of length n without trend (i.e., with zero trend)
# and apply the synchronism test.
n <- 200
y1 <- arima.sim(n=n, list(order=c(1,0,0), ar=c(0.6)))
y2 <- arima.sim(n=n, list(order=c(1,0,0), ar=c(-0.2)))
X1 <- cbind(y1, y2)

## Not run:
sync.test(X1, B=1000)

## End(Not run)
# Sample output:
##
##  Non-parametric test for synchronism of parametric linear trends
##
##data:  X1
##Test statistic = -0.0712, p-value = 0.452
##alternative hypothesis: trends are not synchronized.
##sample estimates:
##$common_trend_estimates
##               Estimate Std. Error    t value  Pr(>|t|)
##(Intercept)  0.02944134 0.09871156  0.2982563 0.7658203
##t           -0.05858974 0.17033482 -0.3439681 0.7312353
##
##$ar_order_used
##         y1 y2
##ar_order  1  1
##
##$Window_used
##       y1 y2
##Window  8 15
##
##$all_considered_windows
## Window   Statistic p-value Asympt. p-value
##      8 -0.09419625   0.295       0.3423957
##     11 -0.08168139   0.363       0.4103374
##     15 -0.08831680   0.459       0.3733687
##     20 -0.09337623   0.451       0.3466142


# Add a time series y3 with a different linear trend and apply the synchronism test.
t <- c(1:n)/n
y3 <- 1 + 2*t + arima.sim(n=n, list(order=c(1,0,0), ar=c(-0.2)))
X2 <- cbind(y1, y3)

## Not run:
sync.test(X2, B=1000)
```

```
## End(Not run)
# Sample output:
##
##  Non-parametric test for synchronism of parametric linear trends
##
##data:  X2
##Test statistic = 0.3027, p-value < 2.2e-16
##alternative hypothesis: trends are not synchronized.
##sample estimates:
##$common_trend_estimates
##              Estimate Std. Error   t value     Pr(>|t|)
##(Intercept) -0.4047268 0.09862909 -4.103523 5.943524e-05
##t            0.8054264 0.17019251  4.732443 4.215118e-06
##
##$ar_order_used
##         y1 y3
##ar_order  1  1
##
##$Window_used
##       y1 y3
##Window  8  8
##
##$all_considered_windows
## Window Statistic p-value Asympt. p-value
##      8 0.3027026       0    3.464035e-04
##     11 0.3527386       0    3.055570e-05
##     15 0.3608431       0    1.998331e-05
##     20 0.3655885       0    1.552063e-05
```

WAVK *WAVK statistic*

Description

Computes statistic for testing the parametric form of a regression function, suggested by Wang, Akritas and Van Keilegom (2008).

Usage

```
WAVK(z, kn = NULL)
```

Arguments

z	pre-filtered univariate time series (see formula (2.1) by Wang and Van Keilegom, 2007):

$$Z_i = \left(Y_{i+p} - \sum_{j=1}^{p} \hat{\phi}_{j,n} Y_{i+p-j} \right) - \left(f(\hat{\theta}, t_{i+p}) - \sum_{j=1}^{p} \widehat{\phi}_{j,n} f(\widehat{\theta}, t_{i+p-j}) \right),$$

where Y_i is observed time series of length n, $\widehat{\theta}$ is an estimator of hypothesized parametric trend $f(\theta, t)$, and $\hat{\phi}_p = (\hat{\phi}_{1,n}, \ldots, \hat{\phi}_{p,n})'$ are estimated coefficients of an autoregressive filter of order p. Missing values are not allowed.

`kn` length of the local window.

Value

A list with following components:

`Tn` test statistic based on artificial ANOVA and defined by Wang and Van Keilegom (2007) as a difference of mean square for treatments (MST) and mean square for errors (MSE):

$$T_n = MST - MSE = \frac{k_n}{n-1}\sum_{t=1}^{T}\left(\overline{V}_{t.} - \overline{V}_{..}\right)^2 - \frac{1}{n(k_n-1)}\sum_{t=1}^{n}\sum_{j=1}^{k_n}\left(V_{tj} - \overline{V}_{t.}\right)^2,$$

where $\{V_{t1}, \ldots, V_{tk_n}\} = \{Z_j : j \in W_t\}$, W_t is a local window, $\overline{V}_{t.}$ and $\overline{V}_{..}$ are the mean of the tth group and the grand mean, respectively.

`Tns` standardized version of `Tn` according to Theorem 3.1 by Wang and Van Keilegom (2007):

$$Tns = \left(\frac{n}{kn}\right)^{\frac{1}{2}} Tn \Big/ \left(\frac{4}{3}\right)^{\frac{1}{2}} \sigma^2,$$

where n is length and σ^2 is variance of the time series. Robust difference-based Rice's estimator (Rice, 1984) is used to estimate σ^2.

`p.value` p-value for `Tns` based on its asymptotic $N(0, 1)$ distribution.

Author(s)

Yulia R. Gel, Vyacheslav Lyubchich

References

Rice, J. (1984) Bandwidth choice for nonparametric regression. *The Annals of Statistics* 12, 1215–1230.

Wang, L., Akritas, M. G. and Van Keilegom, I. (2008) An ANOVA-type nonparametric diagnostic test for heteroscedastic regression models. *Journal of Nonparametric Statistics* 20(5), 365–382.

Wang, L. and Van Keilegom, I. (2007) Nonparametric test for the form of parametric regression with time series errors. *Statistica Sinica* 17, 369–386.

See Also

`wavk.test`.

Examples

```
z <- rnorm(300)
WAVK(z, kn=7)
```

wavk.test *WAVK trend test*

Description

Non-parametric test to detect possibly non-monotonic parametric trend in a time series.

Usage

```
wavk.test(x, factor.length=c("user.defined", "adaptive.selection"),
    Window=round(0.1*length(x)), q=3/4, j=c(8:11), B=1000,
    H0=c("no trend","linear"), method=c("boot", "asympt"),
    ar.order=NULL, BIC=TRUE, robust=TRUE, out=FALSE)
```

Arguments

x	univariate time series. Missing values are not allowed.
factor.length	method to define the length of local windows (factors). Default option "user.defined" allows to set only one value of the argument Window. The option "adaptive.selection" sets method = "boot" and employs heuristic m-out-of-n subsampling algorithm (Bickel and Sakov, 2008) to select an optimal window from the set of possible windows length(x)*q^j whose values are mapped to the largest previous integer and greater than 2.
Window	length of the local window (factor), default is round(0.1*length(x)). This argument is ignored if factor.length = "adaptive.selection".
q	scalar from 0 to 1 to define the set of possible windows when factor.length = "adaptive.selection". Default is $3/4$. This argument is ignored if factor.length = "user.defined".
j	numeric vector to define the set of possible windows when factor.length = "adaptive.selection". Default is c(8:11). This argument is ignored if factor.length = "user.defined".
B	number of bootstrap simulations to obtain empirical critical values. Default is 1000.
H0	null hypothesis: "no trend" (default) for testing the absence of trend (in other words, constant trend) vs. any, possibly non-monotonic, trend; "linear" for testing the presence of parametric linear trend vs. alternative nonlinear trend.
method	method of obtaining critical values: from asymptotical ("asympt") or bootstrap ("boot") distribution. If factor.length = "adaptive.selection" the option "boot" is used.
ar.order	order of autoregressive filter when BIC = FALSE, or the maximal order for BIC-based filtering. Default is floor(10*log10(length(x))).
BIC	logical value indicates whether the order of autoregressive filter should be selected by Bayesian information criterion (BIC). If TRUE (default), models of orders 1,...,ar.order or 1,...,floor(10*log10(length(x))) are be considered, depending on whether ar.order is defined or not.

robust	logical value indicates whether to use robust estimates of autoregression coefficients using HVK function (default), or to use Yule-Walker estimates delivered by ar function.
out	logical value indicates whether full output should be shown. Default is FALSE.

Value

A list with class htest containing the following components:

method	name of the method.
data.name	name of the data.
statistic	value of the test statistic.
p.value	p-value of the test.
alternative	alternative hypothesis.
parameter	window that was used.
estimate	list, containing the estimated coefficients of linear trend (if H0="linear"); estimated AR coefficients; test results for all considered windows (if factor.length = "adaptive.selection").

Author(s)

Yulia R. Gel, Vyacheslav Lyubchich

References

Bickel, P. J. and Sakov, A. (2008) On the choice of m in the m out of n bootstrap and confidence bounds for extrema. *Statistica Sinica* 18, 967–985.

Lyubchich, V., Gel, Y. R. and El-Shaarawi, A. (2013) On detecting non-monotonic trends in environmental time series: a fusion of local regression and bootstrap. *Environmetrics* 24, 209–226.

Wang, L., Akritas, M. G. and Van Keilegom, I. (2008) An ANOVA-type nonparametric diagnostic test for heteroscedastic regression models. *Journal of Nonparametric Statistics* 20(5), 365–382.

Wang, L. and Van Keilegom, I. (2007) Nonparametric test for the form of parametric regression with time series errors. *Statistica Sinica* 17, 369–386.

See Also

HVK, WAVK, sync.test.

Examples

```
# Fix seed for reproduceable simulations.
set.seed(123)

# Simulate autoregressive time series of length n with linear trend 1+2*t,
# where t is a regular sequence on the interval (0,1].
n <- 100
```

```
t <- c(1:n)/n
U <- 1+2*t + arima.sim(n=n, list(order = c(2,0,0), ar = c(-0.7, -0.1)))

# Test for linear trend with output of all results.
## Not run:
wavk.test(U, factor.length = "adaptive.selection", H0="linear", out=TRUE, B=1000)

## End(Not run)
# Sample output:
##
##  Trend test by Wang, Akritas and Van Keilegom
##
##data:  U
##WAVK test statistic = 0.8562, adaptively selected window = 4, p-value = 0.356
##alternative hypothesis: presence of a nonlinear trend
##sample estimates:
##$linear_trend_coefficients
##(Intercept)           t
##  0.9917251   2.0224272
##
##$AR_coefficients
##     phi_1      phi_2
##-0.6814546 -0.2404422
##
##$all_considered_windows
## Window WAVK-statistic p-value
##      4      0.8561654   0.356
##      5      0.8620023   0.320
##      7      0.8691870   0.288
##     10      0.6837790   0.306


# Test H0 of absence of a trend using asymptotic distribution of statistic.
wavk.test(U, method="asympt")

# Sample output:
##
##        Trend test by Wang, Akritas and Van Keilegom
##
##data:  U
##WAVK test statistic = 18.4712, user-defined window = 10, p-value < 2.2e-16
##alternative hypothesis: presence of a trend
```

Index