Package ‘exuber’

March 1, 2019

Type Package

Title Econometric Analysis of Explosive Time Series

Version 0.2.1

Description Testing for and dating periods of explosive dynamics (exuberance) in time series using the univariate and panel recursive unit root tests proposed by Phillips et al. (2015) <doi:10.1111/iere.12132> and Pavlidis et al. (2016) <doi:10.1007/s11146-015-9531-2>. The recursive least-squares algorithm utilizes the matrix inversion lemma to avoid matrix inversion which results in significant speed improvements. Simulation of a variety of periodically-collapsing bubble processes.

License GPL-3

URL https://github.com/kvasilopoulos/exuber

BugReports https://github.com/kvasilopoulos/exuber/issues

Depends R (>= 2.10)

Imports doSNOW, parallel, foreach, Rcpp (>= 0.12.17), dplyr, glue, ggplot2, purrr, magrittr, lubridate, tibble, zoo, rlang, grid, gridExtra

Suggests knitr, rmarkdown, covr, testthat, withr

LinkingTo Rcpp, RcppArmadillo

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation yes

Author Kostas Vasilopoulos [cre, aut],
Efthymios Pavlidis [aut],
Simon Spavound [aut],
Enríque Martínez-García [aut]

Maintainer Kostas Vasilopoulos <k.vasilopoulo@gmail.com>

Repository CRAN

Date/Publication 2019-03-01 21:40:08 UTC

R topics documented:

`autoplot.datestamp` *Plotting with ggplot2 and tidying with tibble datestamp objects*

Description

Plotting datestamp with geom_segment()

Usage

```
## S3 method for class 'datestamp'
autoplot(object, ...)

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

Arguments

`object`	An object of class `datestamp()`
`...`	further arguments passed to method, ignored.
`model`	datestamp object
`data`	original dataset, not used (required by generic `fortify()` method).

Examples

```
dta <- cbind(sim_dgp1(n = 100), sim_dgp2(n = 100))

dta %>%
  radf() %>%
  datestamp() %>%
  autoplot()

# Change the colour manually
dta %>%
  radf() %>%
  datestamp() %>%
  autoplot() +
  ggplot2::scale_colour_manual(values=rep("black", 4 ))
```

`autoplot.radf` *Plotting with ggplot2 and tidying with tibble radf objects*

Description

`autoplot.radf` takes an `radf` object and returns a (list of) ggplot2 objects. `fortify.radf` takes an `radf` object and converts it into a data.frame. `ggarrange` is a wrapper of `arrangeGrob()`, which can be used directly after autoplot to place grobs on a page.

Usage

```
## S3 method for class 'radf'
autoplot(object, cv, include = FALSE, select = NULL,
  option = c("gsadf", "sadf"), min_duration = 0, ...)

## S3 method for class 'radf'
fortify(model, data, cv, include = FALSE, select = NULL,
  option = c("gsadf", "sadf"), ...)

ggarrange(...)
```

Arguments

`object`	An object of class `radf()`.
`cv`	An object of class "cv". The output of `mc_cv()`, `wb_cv()` or `sb_cv()`
`include`	If not FALSE, plot all variables regardless of rejecting the NULL at the 5% significance level.

select	If not NULL, only plot with names or column number matching this regular expression will be executed.
option	Whether to apply the "gsadf" or "sadf" methodology. Default is "gsadf".
min_duration	The minimum duration of an explosive period for it to be reported. Default is 0.
...	further arguments passed to method, ignored.
model	An object of class radf().
data	original dataset, not used (required by generic fortify() method).

Examples

```
dta <- cbind(sim_dgp1(n = 100), sim_dgp2(n = 100))

dta %>%
  radf() %>%
  autoplot() %>%
  ggarrange(ncol = 2)

# For custom plotting with ggplot2
dta %>%
  radf() %>%
  fortify()
```

---

col_names *Retrieve/Set column names*

Description

Retrieve or set the column names of a class radf() object. Similar to colnames, with the only difference that col_names is for radf() objects.

Usage

```
col_names(x, ...)

col_names(x) <- value
```

Arguments

x	An object of class radf()
...	Further arguments passed to methods.
value	An ordered vector of the same length as the 'index' attribute of x.

Examples

```
# Simulate bubble processes
dta <- cbind(sim_dgp1(n = 100), sim_dgp2(n = 100))

rfd <- radf(dta)
col_names(rfd) <- c("OneBubble", "TwoBubbles")
```

---

crit *Simulated Monte Carlo critical values*

Description

A dataset containing simulated critical values for up to 700 observations based on default minimum window. The critical values have been simulated and stored as data to save computation time for the user. Critical values can be also obtained with the `mc_cv()` function.

Usage

```
crit
```

Format

A list with lower level lists that contain

adf_cv: Augmented Dickey-Fuller

badf_cv: Backward Augmented Dickey-Fuller

sadf_cv: Supremum Augmented Dickey-Fuller

bsadf_cv: Backward Supremum Augmented Dickey-Fuller

gsadf_cv: Generalized Supremum Augmented Dickey Fuller

Source

simulated from exuber package function `mc_cv()`

`datestamp` *Date stamping periods of mildly explosive behaviour*

Description

Computes the origination, termination and duration of episodes during which the time series display explosive dynamics.

Usage

```
datestamp(object, cv, option = c("gsadf", "sadf"), min_duration = 0)
```

Arguments

`object`	An object of class `radf()`.
`cv`	An object of class "cv". The output of `mc_cv()`, `wb_cv()` or `sb_cv()`
`option`	Whether to apply the "gsadf" or "sadf" methodology. Default is "gsadf".
`min_duration`	The minimum duration of an explosive period for it to be reported. Default is 0.

Details

Datestamp also stores a vector in 0,1 that corresponds to reject, accept respectively, for all series in the time period. This output can be used as a dummy that indicates the occurrence of a bubble.

Setting `min_duration` removes very short episode of exuberance. Phillips et al. (2015) propose two simple rules of thumb to remove short periods of explosive dynamics, "log(T)/T", where T is the number of observations.

Value

Returns a list of values for each explosive sub-period, giving the origin and termination dates as well as the number of periods explosive behavior lasts.

References

Phillips, P. C. B., Shi, S., & Yu, J. (2015). Testing for Multiple Bubbles: Historical Episodes of Exuberance and Collapse in the S&P 500. International Economic Review, 56(4), 1043-1078.

`diagnostics` *Diagnostics*

Description

Finds the series that reject the null for at the 5% significance level.

Usage

```
diagnostics(object, cv, option = c("gsadf", "sadf"))
```

Arguments

`object`	An object of class `radf()`.
`cv`	An object of class "cv". The output of `mc_cv()`, `wb_cv()` or `sb_cv()`
`option`	Whether to apply the "gsadf" or "sadf" methodology. Default is "gsadf".

Details

Diagnostics also stores a vector in 0,1 that corresponds to reject, accept respectively.

Value

Returns a list with the series that reject and the series that do not reject the Null Hypothesis

`index.radf` *Retrieve/Replace the Index*

Description

Retrieve or replace the index of an `radf` object.

Usage

```
## S3 method for class 'radf'
index(x, trunc = FALSE, ...)

## S3 method for class 'data.frame'
index(x, ...)

## S3 method for class 'datestamp'
index(x, ...)

## S3 replacement method for class 'radf'
index(x) <- value
```

Arguments

x	An object of class radf()
trunc	default FALSE. If TRUE the index formed by truncating the value in the minimum window.
...	Further arguments passed to methods.
value	An ordered vector of the same length as the 'index' attribute of x.

Details

If the user does not specify an index for the estimation a pseudo-index is generated which is a sequential numeric series. After the estimation, the user can use `index` to retrieve or `` `index<-` `` to replace the index. The index can be either numeric or Date.

---

mc_cv *Monte Carlo Critical Values*

Description

`mc_cv` computes Monte Carlo critical values for the recursive unit root tests.

Usage

```
mc_cv(n, nrep = 2000, minw, opt_badf = c("fixed", "asymptotic",
  "simulated"), opt_bsadf = c("conventional", "conservative"))
```

Arguments

n	A positive integer. The sample size.
nrep	A positive integer. The number of Monte Carlo simulations.
minw	A positive integer. The minimum window size, which defaults to $(0.01 + 1.8/\sqrt(T)) * T$.
opt_badf	Options for badf critical value calculation. "fixed" corresponds to log(log(n*s))/100 rule, "asymptotic" to asymptotic critical values and simulated to the monte carlo simulations.
opt_bsadf	Options for bsadf critical value calculation. "conventional" corresponds to the max of the quantile of the simulated distribution, while "conservative" corresponds to the quantile of the max which is more conservative in nature, thus the name.

Value

A list that contains the critical values for ADF, BADF, BSADF and GSADF t-statistics.

See Also

`wb_cv` for Wild Bootstrapped critical values and `sb_cv` for Sieve Bootstrapped critical values

Examples

```
# Default minimum window
mc <- mc_cv(n = 100)

# Change the minimum window and the number of simulations
mc <- mc_cv(n = 100, nrep = 2500,  minw = 20)
```

---

radf *Recursive Augmented Dickey-Fuller Test*

Description

`radf` returns the t-statistics from a recursive Augmented Dickey-Fuller test.

Usage

```
radf(data, minw, lag = 0)
```

Arguments

data	A univariate or multivariate numeric ts object, data.frame or matrix. The estimation process cannot handle NA values.
minw	A positive integer. The minimum window size, which defaults to $(0.01 + 1.8/\sqrt(T)) * T$.
lag	A non-negative integer. The lag of the Augmented Dickey-Fuller regression.

Value

A list that contains the t-statistic (sequence) for:

adf	Augmented Dickey-Fuller
badf	Backward Augmented Dickey-Fuller
sadf	Supremum Augmented Dickey-Fuller
bsadf	Backward Supremum Augmented Dickey-Fuller
gsadf	Generalized Supremum Augmented Dickey-Fuller

References

Phillips, P. C. B., Wu, Y., & Yu, J. (2011). Explosive Behavior in The 1990s Nasdaq: When Did Exuberance Escalate Asset Values? International Economic Review, 52(1), 201-226.

Phillips, P. C. B., Shi, S., & Yu, J. (2015). Testing for Multiple Bubbles: Historical Episodes of Exuberance and Collapse in the S&P 500. International Economic Review, 56(4), 1043-1078.

Examples

```
# Simulate bubble processes
dta <- cbind(sim_dgp1(n = 100), sim_dgp2(n = 100))

rfd <- radf(dta)

# For lag = 1 and minimum window = 20
rfd <- radf(dta, minw = 20, lag = 1)
```

---

sb_cv *Panel Sieve Bootstrap Critical Values*

Description

sb_cv computes p-values for the panel recursive unit root test using the sieve bootstrap procedure outlined in Pavlidis et al. (2016)

Usage

```
sb_cv(data, minw, lag = 0, nboot = 1000)
```

Arguments

data	A univariate or multivariate numeric ts object, data.frame or matrix. The estimation process cannot handle NA values.
minw	A positive integer. The minimum window size, which defaults to $(0.01 + 1.8/\sqrt(T)) * T$.
lag	A non-negative integer. The lag of the Augmented Dickey-Fuller regression.
nboot	A positive integer indicating the number of bootstraps. Default is 1000 repetitions.

Value

A list that contains the panel critical values for BSADF and GSADF t-statistics.

References

Pavlidis, E., Yusupova, A., Paya, I., Peel, D., Martínez-García, E., Mack, A., & Grossman, V. (2016). Episodes of exuberance in housing markets: in search of the smoking gun. The Journal of Real Estate Finance and Economics, 53(4), 419-449.

See Also

mc_cv for Monte Carlo critical values and wb_cv for Wild Bootstrapped critical values

Examples

```
# Simulate bubble processes
set.seed(124)
pdta <- cbind(sim_dgp1(100), sim_dgp1(100), sim_div(100), sim_div(100), sim_div(100))

# Panel critical vales should have the same lag length with the estimation
sb <- sb_cv(pdta, lag = 1)

pdta %>%
  radf(lag = 1) %>%
  summary(cv = sb)

pdta %>%
  radf(lag = 1) %>%
  autoplot(cv = sb)
```

---

sim_blan *Simulation of a Blanchard (1979) bubble process*

Description

Simulation of a Blanchard (1979) rational bubble process.

Usage

```
sim_blan(n, pi = 0.7, sigma = 0.03, r = 0.05)
```

Arguments

n	A strictly positive integer specifying the length of the simulated output series.
pi	A positive value in (0, 1) which governs the probability of the bubble continuing to grow.
sigma	A positive scalar indicating the standard deviation of the innovations.
r	A positive scalar that determines the growth rate of the bubble process.

Details

Blanchard's bubble process has two regimes, which occur with probability π and $1 - \pi$. In the first regime, the bubble grows exponentially, whereas in the second regime, the bubble collapses to a white noise.

With probability π:

$$B_{t+1} = \frac{1+r}{\pi} B_t + \epsilon_{t+1}$$

With probability $1 - \pi$:

$$B_{t+1} = \epsilon_{t+1}$$

where r is a positive constant and $\epsilon \sim iid(0, \sigma^2)$.

Value

A numeric vector of length n.

References

Blanchard, O. J. (1979). Speculative bubbles, crashes and rational expectations. Economics letters, 3(4), 387-389.

See Also

sim_dgp1, sim_dgp2, sim_evans

Examples

```
sim_blan(n = 100)
```

---

sim_dgp1 *Simulation of a single-bubble process*

Description

The following function generates a time series which switches from a martingale to a mildly explosive process and then back to a martingale.

Usage

```
sim_dgp1(n, te = 0.4 * n, tf = 0.15 * n + te, c = 1, alpha = 0.6,
  sigma = 6.79)
```

Arguments

n	A strictly positive integer specifying the length of the simulated output series.
te	A scalar in (0, tf) specifying the observation in which the bubble originates.
tf	A scalar in (te, n) specifying the observation in which the bubble collapses.
c	A positive scalar determining the autoregressive coefficient in the explosive regime.
alpha	A positive scalar in (0, 1) determining the value of the expansion rate in the autoregressive coefficient.
sigma	A positive scalar indicating the standard deviation of the innovations.

Details

The data generating process is described by the following equation:

$$X_t = X_{t-1}1\{t < \tau_e\} + \delta_T X_{t-1}1\{\tau_e \leq t \leq \tau_f\} + \left(\sum_{k=\tau_f+1}^{t} \epsilon_k + X^*_{\tau_f}\right)1\{t > \tau_f\} + \epsilon_t 1\{t \leq \tau_f\}$$

where the autoregressive coefficient δ_T is given by:

$$\delta_T = 1 + cT^{-a}$$

with $c > 0$, $\alpha \in (0, 1)$, $\epsilon \sim iid(0, \sigma^2)$ and $X_{\tau_f} = X_{\tau_e} + X^*$. During the pre- and post- bubble periods, $N_0 = [1, \tau_e)$, X is a pure random walk process. During the bubble expansion period $B = [\tau_e, \tau_f]$ is a mildly explosive process with expansion rate given by the autoregressive coefficient δ_T, and continues its martingale path for the subsequent period $N_1 = (\tau_f, \tau]$.

For further details the user can refer to Phillips et al. (2015) p. 1054.

Value

A numeric vector of length n.

References

Phillips, P. C. B., Shi, S., & Yu, J. (2015). Testing for Multiple Bubbles: Historical Episodes of Exuberance and Collapse in the S&P 500. International Economic Review, 5 6(4), 1043-1078.

See Also

sim_dgp2, sim_blan, sim_evans

Examples

```
# 100 periods with bubble origination date 40 and termination date 55
sim_dgp1(n = 100)

# 200 periods with bubble origination date 80 and termination date 110
sim_dgp1(n = 200)

# 200 periods with bubble origination date 100 and termination date 150
sim_dgp1(n = 200, te = 100, tf = 150)
```

`sim_dgp2` *Simulation of a two-bubble process*

Description

The following data generating process is similar to `sim_dgp1`, with the difference that there are two episodes of mildly explosive dynamics.

Usage

```
sim_dgp2(n, te1 = 0.2 * n, tf1 = 0.2 * n + te1, te2 = 0.6 * n,
  tf2 = 0.1 * n + te2, c = 1, alpha = 0.6, sigma = 6.79)
```

Arguments

`n`	A strictly positive integer specifying the length of the simulated output series.
`te1`	A scalar in (0, n) specifying the observation in which the first bubble originates.
`tf1`	A scalar in (te1, n) specifying the observation in which the first bubble collapses.
`te2`	A scalar in (tf1, n) specifying the observation in which the second bubble originates.
`tf2`	A scalar in (te2, n) specifying the observation in which the second bubble collapses.
`c`	A positive scalar determining the autoregressive coefficient in the explosive regime.
`alpha`	A positive scalar in (0, 1) determining the value of the expansion rate in the autoregressive coefficient.
`sigma`	A positive scalar indicating the standard deviation of the innovations.

Details

The data generating process is described by:

$$X_t = X_{t-1}1\{t \in N_0\} + \delta_T X_{t-1}1\{t \in B_1 \cup B_2\} + \left(\sum_{k=\tau_{1f}+1}^{t} \epsilon_k + X^*_{\tau_{1f}}\right) 1\{t \in N_1\}$$

$$+ \left(\sum_{l=\tau_{2f}+1}^{t} \epsilon_l + X^*_{\tau_{2f}}\right) 1\{t \in N_2\} + \epsilon_t 1\{t \in N_0 \cup B_1 \cup B_2\}$$

where the autoregressive coefficient δ_T is given:

$$\delta_T = 1 + cT^{-a}$$

with $c > 0$, $\alpha \in (0, 1)$, $\epsilon \sim iid(0, \sigma^2)$, $X_{\tau_{1f}} = X_{\tau_{1e}} + X^*$ and $X_{\tau_{2f}} = X_{\tau_{2e}} + X^*$. We use the notation $N_0 = [1, \tau_{1e})$, $B_1 = [\tau_{1e}, \tau_{1f}]$, $N_1 = (\tau_{1f}, \tau_{2e})$, $B_2 = [\tau_{2e}, \tau_{2f}]$, $N_2 = (\tau_{2f}, \tau]$, where τ is the last observation of the sample. After the collapse of the first bubble, X_t resumes a martingale path until time $\tau_{2e} - 1$, and a second episode of exuberance begins at τ_{2e}. The expansion process lasts until τ_{2f} and collapses to a value of $X^*_{\tau_{2f}}$. The process then continues on a martingale path until the end of the sample period τ. The expansion duration of the first bubble is assumed to be longer than that of the second bubble, i.e. $\tau_{1f} - \tau_{1e} > \tau_{2f} - \tau_{2e}$.

For further details the user can refer to Phillips et al., (2015) p. 1055.

Value

A numeric vector of length `n`.

References

Phillips, P. C. B., Shi, S., & Yu, J. (2015). Testing for Multiple Bubbles: Historical Episodes of Exuberance and Collapse in the S&P 500. International Economic Review, 5 6(4), 1043-1078.

See Also

`sim_dgp1`, `sim_blan`, `sim_evans`

Examples

```
# 100 periods with bubble origination dates 20/60 and termination dates 40/70 respectively
sim_dgp2(n = 100)

# 200 periods with bubble origination dates 40/120 and termination dates 80/140 respectively
sim_dgp2(n = 200)
```

`sim_div` *Simulation of dividends*

Description

Simulate (log) dividends from a random walk with drift.

Usage

```
sim_div(n, mu, sigma, r = 0.05, log = FALSE, output = c("pf", "d"))
```

Arguments

`n`	A strictly positive integer specifying the length of the simulated output series.
`mu`	A scalar indicating the drift.
`sigma`	A positive scalar indicating the standard deviation of the innovations.
`r`	A positive value indicating the discount factor.

log	A logical. If true dividends follow a lognormal distribution.
output	A character string giving the fundamental price("pf") or dividend series("d"). Default is 'pf'.

Details

If log is set to FALSE (default value) the dividends follow:

$$d_t = \mu + d_{t-1} + \epsilon_t$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The default parameters are $\mu = 0.0373$, $\sigma^2 = 0.1574$ and $d[0] = 1.3$ (the initial value of the dividend sequence). The above equation can be solved to yield the fundamental price:

$$F_t = \mu(1 + r)r^{-2} + r^{-1}d_t$$

If log is set to TRUE then dividends follow a lognormal distribution or log(dividends) follow:

$$\ln(d_t) = \mu + \ln(d_{t-1}) + \epsilon_t$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Default parameters are $\mu = 0.013$, $\sigma^2 = 0.16$. The fundamental price for this case is:

$$F_t = \frac{1 + g}{r - g} d_t$$

where $1 + g = \exp(\mu + \sigma^2/2)$. All default parameter values are those suggested by West (1988).

Value

A numeric vector of length n.

References

West, K. D. (1988). Dividend innovations and stock price volatility. Econometrica: Journal of the Econometric Society, p. 37-61.

Examples

```
# Price is the sum of the bubble and fundamental components
# 20 is the scaling factor
pf <- sim_div(100, r = 0.05, output = "pf")
pb <- sim_evans(100, r = 0.05)
p <- pf + 20*pb
```

sim_evans — *Simulation of an Evans (1991) bubble process*

Description

Simulation of an Evans (1991) rational periodically collapsing bubble process.

Usage

```
sim_evans(n, alpha = 1, delta = 0.5, tau = 0.05, pi = 0.7,
  r = 0.05, b1 = delta)
```

Arguments

`n`	A strictly positive integer specifying the length of the simulated output series.
`alpha`	A positive scalar, with restrictions (see details).
`delta`	A positive scalar, with restrictions (see details).
`tau`	The standard deviation of the innovations.
`pi`	A positive value in (0, 1) which governs the probability of the bubble continuing to grow.
`r`	A positive scalar that determines the growth rate of the bubble process.
`b1`	A positive scalar, the initial value of the series. Defaults to `delta`.

Details

`delta` and `alpha` are positive parameters which satisfy $0 < \delta < (1+r)\alpha$. `delta` represents the size of the bubble after collapse. The default value of `r` is 0.05. The function checks whether `alpha` and `delta` satisfy this condition and will return an error if not.

The Evans bubble has two regimes. If $B_t \leq \alpha$ the bubble grows at an average rate of $1+r$:

$$B_{t+1} = (1+r)B_t u_{t+1},$$

When $B_t > \alpha$ the bubble expands at an increased rate of $(1+r)\pi^{-1}$:

$$B_{t+1} = [\delta + (1+r)\pi^{-1}\theta_{t+1}(B_t - (1+r)^{-1}\delta B_t)]u_{t+1},$$

where θ is an indicator function taking a value of 0 with probability $1-\pi$ and 1 with probability π. In this secondary phase there is a probability $(1-\pi)$ that the bubble collapses to `delta` and the process starts again. By modifying the values of `delta`, `alpha` and `pi` the user can change the frequency at which bubbles appear, the mean duration of a bubble before collapse and the scale of the bubble.

Value

A numeric vector of length `n`.

References

Evans, G. W. (1991). Pitfalls in testing for explosive bubbles in asset prices. The American Economic Review, 81(4), 922-930.

See Also

sim_dgp1, sim_dgp2, sim_blan

---

summary — *Summarizing radf Models*

Description

summary method for class "radf"

Usage

```
## S3 method for class 'radf'
summary(object, cv, ...)
```

Arguments

object	An object of class radf().
cv	An object of class "cv". The output of mc_cv(), wb_cv() or sb_cv()
...	further arguments passed to methods, not used.

Value

Returns a list of summary statistics, the t-statistic and the critical values of the ADF, SADF and GSADF.

Examples

```
# Simulate bubble processes, compute the t-stat and critical values
set.seed(4441)
dta <- cbind(sim_dgp1(n = 100), sim_dgp2(n = 100))
rfd <- radf(dta)

# Summary, diagnostics and datestamp (default)
summary(rfd)
diagnostics(rfd)
datestamp(rfd)

#' # Diagnostics for 'sadf'
diagnostics(rfd, option = "sadf")

# Use log(T)/T rule of thumb to omit periods of explosiveness which are short-lived
```

```
rot = round(log(NROW(rfd))/NROW(rfd))
datestamp(rfd, min_duration = rot)


# Summary, diagnostics and datestamp (Wild Bootstrapped critical values)

wb <- wb_cv(dta)

summary(rfd, cv = wb)
diagnostics(rfd, cv = wb)
datestamp(rfd, cv = wb)
```

`wb_cv` *Wild Bootstrap Critical values*

Description

`wb_cv` performs the Harvey et al. (2016) wild bootstrap re-sampling scheme, which is asymptotically robust to non-stationary volatility, to generate critical values for the recursive unit root tests.

Usage

```
wb_cv(data, minw, nboot = 1000, dist_rad = FALSE)
```

Arguments

`data`	A univariate or multivariate numeric ts object, data.frame or matrix. The estimation process cannot handle NA values.
`minw`	A positive integer. The minimum window size, which defaults to $(0.01 + 1.8/\sqrt(T)) * T$.
`nboot`	A positive integer indicating the number of bootstraps. Default is 1000 repetitions.
`dist_rad`	Logical. If `TRUE` then the Rademacher distribution will be used.

Details

This approach involves applying a wild bootstrap re-sampling scheme to construct the bootstrap analogue of the Phillips et al. (2015) test which is asymptotically robust to non-stationary volatility.

Value

A list that contains the critical values for ADF, BADF, BSADF and GSADF t-statistics.

References

Harvey, D. I., Leybourne, S. J., Sollis, R., & Taylor, A. M. R. (2016). Tests for explosive financial bubbles in the presence of non-stationary volatility. Journal of Empirical Finance, 38(Part B), 548-574.

Phillips, P. C. B., Shi, S., & Yu, J. (2015). Testing for Multiple Bubbles: Historical Episodes of Exuberance and Collapse in the S&P 500. International Economic Review, 56(4), 1043-1078.

See Also

mc_cv for Monte Carlo critical values and sb_cv for Sieve Bootstrapped critical values

Examples

```
# Simulate bubble processes
dta <- data.frame("dg1" = sim_dgp1(n = 100), "dgp2" = sim_dgp2(n = 100))

# Default minimum window
wb <- wb_cv(dta)

# Change the minimum window and the number of bootstraps
wb <- wb_cv(dta, nboot = 1500,  minw = 20)
```

Index