Package 'DriftBurstHypothesis'

June 6, 2019

Type Package

Title Calculates the Test-Statistic for the Drift Burst Hypothesis

Version 0.1.3

Date 2019-06-06

Author Emil Sjoerup

Maintainer Emil Sjoerup <emilsjoerup@live.dk>

Description Calculates the T-Statistic for the drift burst hypothesis from the working paper Christensen, Oomen and Reno (2018) <DOI:10.2139/ssrn.2842535>. The authors' MATLAB code is available upon request, see: <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2842535>.

License GPL-3

BugReports https://github.com/emilsjoerup/DriftBurstHypothesis/issues

URL https://github.com/emilsjoerup/DriftBurstHypothesis

Imports Rcpp (>= 0.12.18), xts, zoo,

LinkingTo Rcpp, RcppArmadillo

Suggests testthat

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-06-06 19:32:14 UTC

R topics documented:

DriftBurstHypothesis-package 2
drift_bursts 3

Index **8**

`DriftBurstHypothesis-package` *Calculates the Test-Statistic for the Drift Burst Hypothesis*

Description

Calculates the T-Statistic for the drift burst hypothesis from the working paper Christensen, Oomen and Reno (2018) <DOI:10.2139/ssrn.2842535>. The authors' MATLAB code is available upon request, see: <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2842535>.

Details

The DESCRIPTION file:

Package:	DriftBurstHypothesis
Type:	Package
Title:	Calculates the Test-Statistic for the Drift Burst Hypothesis
Version:	0.1.3
Date:	2019-06-06
Author:	Emil Sjoerup
Maintainer:	Emil Sjoerup <emilsjoerup@live.dk>
Description:	Calculates the T-Statistic for the drift burst hypothesis from the working paper Christensen, Oomen and Reno (
License:	GPL-3
BugReports:	https://github.com/emilsjoerup/DriftBurstHypothesis/issues
URL:	https://github.com/emilsjoerup/DriftBurstHypothesis
Imports:	Rcpp (>= 0.12.18), xts, zoo,
LinkingTo:	Rcpp, RcppArmadillo
Suggests:	testthat

Index of help topics:


```
DriftBurstHypothesis-package
                        Calculates the Test-Statistic for the Drift
                        Burst Hypothesis
drift_bursts            Drift Bursts
```


Author(s)

Emil Sjoerup

Maintainer: Emil Sjoerup <emilsjoerup@live.dk>

References

Christensen, Oomen and Reno (2018) <DOI:10.2139/ssrn.2842535>.

drift_bursts *Drift Bursts*

Description

Calculates the Test-Statistic for the Drift Burst Hypothesis.

Usage

```
drift_bursts(time = NULL, logprices, testtimes = seq(34200, 57600, 60),
             PreAverage = 5, AcLag = -1L, Mean_bandwidth = 300L,
             Variance_bandwidth = 900L, bParallelize = FALSE, iCores = NA,
             warnings = TRUE)
```

Arguments

time	Either: A numeric of timestamps for the trades in seconds after midnight. Or: NULL, when the time argument is NULL, the logprices argument must be an xts object.
logprices	A numeric or xts object containing the log-prices of the asset.
testtimes	A numeric containing the times at which to calculate the tests. The standard of seq(34200, 57600, 60) denotes calculating the test-statistic once per minute, i.e. 391 times for a typical 6.5 hour trading day from 9:30:00 to 16:00:00. See details.
PreAverage	An integer denoting the length of pre-averaging estimates of the log-prices.
AcLag	An integer greater than 1 denoting which lag is to be used for the HAC estimator of the variance - the standard of -1 denotes using an automatic lag selection algorithm.
Mean_bandwidth	An integer denoting the bandwidth for the left-sided exponential kernel for the mean.
Variance_bandwidth	An integer denoting the bandwidth for the left-sided exponential kernel for the variance.
bParallelize	A Boolean to determine whether to parallelize the underlying C++ code. (Using OpenMP)
iCores	An integer denoting the number of cores to use for calculating the code when paralellized. If this argument is not provided, sequential evaluation will be used even though bParallelize is TRUE
warnings	A logical denoting whether warnings should be shown.

Details

If the testtimes vector contains instructions to test before the first trade (excluding the very first entry), or more than 15 minutes after the last trade, these entries will be deleted, as not doing so may cause crashes.

The DBH test statistic cannot be calculated before 1 period of testtimes has passed.

The test statistic is unstable before `max(Mean_bandwidth , Variance_bandwidth)` seconds has passed.

If times is provided and logprices is an `xts` object, the indices of logprice will be used regardless.

Note that using an `xts` argument is slower than using a `numeric` and the timestamps due to the creation of the timestamps from the index of the input.

The lags from the Newey-West algorithm is increased by 2 * (PreAveage-1) as due to the Pre-Averaging we know atleast this many lags should be corrected for due to the auto-correlation. The maximum of 20 lags is also increased by this same factor for the same reason.

Value

An object of class `DBH` and `list` containing the series of the drift burst hypothesis test-statistic as well as the estimated spot drift and variance series.

The S3 class DBH has the following methods so far:

- `getDB`: Extracts the test-statistic.
- `getSigma`: Extracts the estimated local variance.
- `getMu`: Extracts the estimated local drift.
- `plot`: Plotting routine. The plotting routine accepts 10 arguments for customization in the ... argument, which must be passed as a named list:
  - `which`:Used for choosing which series to plot, valid choices are: `"DriftBursts"`, `"Sigma"`, `"Mu"`, and `c("Sigma","Mu")`, the order of the latter is irrelevant. Default = `"DriftBursts"`
  - `price`:The price series which, if provided, will be overlayed in a red dotted line and the level will be shown at the right y-axis. (Only used if which is `"DriftBursts"`). Default = `NULL`
  - `time`:Timestamps for the trades in seconds after midnight, which will be used for the x-axis of the plot if the price is overlayed. Default = `NULL`
  - `startTime`:Start of the trading day in seconds after midnight. Default = `34200`
  - `endTime`:End of the trading day in seconds after midnight. Default = `57600`
  - `xlab`:The label of the x-axis. Default = `"time"`
  - `ylab`:The label of the y-axis. Default = `"value"`
  - `leg.x`:X-position of the legend in the case `which` is `"DriftBursts"` AND the price is overlayed. Default = `"topleft"`. Usage is as in the base `R` engine.
  - `leg.y`:Y-position of the legend in the case `which` is `"DriftBursts"` AND the price is overlayed. Default = `"NULL"`. Usage is as in the base `R` engine.

Author(s)

Emil Sjoerup

References

Christensen, Oomen and Reno (2018) <DOI:10.2139/ssrn.2842535>.

Examples

```
#Simulate from a stochastic volatility model.
#Both a flash crash and flash rally are coded into the function.
StochasticVolatilitySim = function(iT, dSigma, dPhi, dMu){
  vSim = numeric(iT)
  vEps = rnorm(iT , sd =dSigma)
  vEpsy = rnorm(iT)
  vEps[30001:32000] = rnorm(2000 ,sd =seq(from = dSigma ,
                                          to = 2*dSigma , length.out = 2000))
  vEps[32001:34000] = rnorm(2000 ,sd =seq(from = 2*dSigma ,
                                          to = dSigma , length.out = 2000))
  vEpsy[30001:32000] = -rnorm(2000 ,mean =seq(from = 0,
                                              to = 0.3 , length.out = 2000))
  vEpsy[32001:34000] = -rnorm(2000 ,mean =seq(from = 0.3,
                                              to = 0 , length.out = 2000))


  vEps[60001:63000] = rnorm(3000,sd = seq(from = dSigma ,
                                          to = 2*dSigma , length.out = 3000))
  vEps[63001:66000] = rnorm(3000,  sd = seq(from = 2*dSigma ,
                                            to =  dSigma, length.out = 3000))

  vEpsy[60001:63000] = rnorm(3000 ,mean =seq(from = 0,
                                             to = 0.2 , length.out = 3000))
  vEpsy[63001:66000] = rnorm(3000 ,mean =seq(from = 0.2,
                                             to = 0 , length.out = 3000))
  vSim[1] = dMu + dPhi *rnorm(1 , mean = dMu , sd = dSigma /sqrt(1-dPhi^2))
  for (i in 2:iT) {
    vSim[i] = dMu + dPhi * (vSim[(i-1)] - dMu) + vEps[i]
  }
  vY = exp(vSim/2) * vEpsy
  return(vY)
}
#Set parameter values of the simulation
iT = 66500; dSigma = 0.3; dPhi = 0.98; dMu = -10;
#set seed for reproducibility
set.seed(123)
#Simulate the series
vY = 500+cumsum(StochasticVolatilitySim(iT, dSigma, dPhi, dMu))

#insert an outlier to illustrate robustness.
vY[50000] = 500

#Here, the observations are equidistant, but the code can handle unevenly spaced observations.
timestamps = seq(34200 , 57600 , length.out = iT)
testtimes = seq(34200, 57600, 60)
logprices = log(vY)
```

```
#calculating drift burst hypothesis

DBHtStat = drift_bursts(timestamps,  logprices,
                        testtimes, PreAverage = 5, AcLag = -1L,
                        Mean_bandwidth = 300L, Variance_bandwidth = 900L,
                        bParallelize = FALSE)


#plot test statistic
plot(DBHtStat)
#plot both test statistic and price
plot(DBHtStat, price = vY, time = timestamps)
#Plot the mu series
plot(DBHtStat, which = "Mu")
#plot the sigma series
plot(DBHtStat, which = "Sigma")
#plot both
plot(DBHtStat, which = c("Mu", "Sigma"))


#Means of the tstat, sigma, and mu series.
mean(getDB(DBHtStat))
mean(getSigma(DBHtStat))
mean(getMu(DBHtStat))



## Not run:
################## same example with xts object:
#Set parameter values of the simulation
iT = 66500; dSigma = 0.3; dPhi = 0.98; dMu = -10;
#set seed for reproducibility
set.seed(123)
#Simulate the series
vY = 500+cumsum(StochasticVolatilitySim(iT, dSigma, dPhi, dMu))

#insert an outlier to illustrate robustness.
vY[50000] = 500

#Here, the observations are equidistant, but the code can handle unevenly spaced observations.
timestamps = seq(34200 , 57600 , length.out = iT)
StartTime = strptime("1970-01-01 00:00:00.0000", "%Y-%m-%d %H:%M:%OS", tz = "GMT")
Tradetime = StartTime + timestamps
testtimes = seq(34200, 57600, 60)


price = xts(vY, Tradetime)


DBH = drift_bursts(time = NULL,  log(price),
                   testtimes, PreAverage = 5, AcLag = -1L,
                   Mean_bandwidth = 300L, Variance_bandwidth = 900L,
                   bParallelize = FALSE)
```

```
#check for equality
all.equal(as.numeric(DBHtStat$DriftBursts), as.numeric(DBH$DriftBursts))

## End(Not run)
```

Index

∗Topic **Drift burst hypothesis**
DriftBurstHypothesis-package, 2

drift_bursts, 3
DriftBurstHypothesis
(DriftBurstHypothesis-package), 2
DriftBurstHypothesis-package, 2

getDB (drift_bursts), 3
getMu (drift_bursts), 3
getSigma (drift_bursts), 3