

# Package ‘monotonicity’

February 14, 2019

**Type** Package

**Title** Test for Monotonicity in Expected Asset Returns, Sorted by Portfolios

**Version** 1.2

**Date** 2019-02-13

**Maintainer** Siegfried Köstlmeier <siegfried.koestlmeier@gmail.com>

**Description** Test for monotonicity in financial variables sorted by portfolios. It is conventional practice in empirical research to form portfolios of assets ranked by a certain sort variable. A t-test is then used to consider the mean return spread between the portfolios with the highest and lowest values of the sort variable. Yet comparing only the average returns on the top and bottom portfolios does not provide a sufficient way to test for a monotonic relation between expected returns and the sort variable. This package provides nonparametric tests for the full set of monotonic patterns by Patton, A. and Timmermann, A. (2010) <doi:10.1016/j.jfineco.2010.06.006> and compares the proposed results with extant alternatives such as t-tests, Bonferroni bounds, and multivariate inequality tests through empirical applications and simulations.

**License** BSD\_3\_clause + file LICENSE

**URL** <https://github.com/skoestlmeier/monotonicity>

**Imports** lmtest, MASS, sandwich, stats, methods, utils

**Suggests** testthat

**Depends** R (>= 3.3)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Siegfried Köstlmeier [aut, cre, trl]  
(<<https://orcid.org/0000-0002-7221-6981>>)

**Repository** CRAN

**Date/Publication** 2019-02-14 11:50:04 UTC

## R topics documented:

demo_returns . . . . .	2
monoBonferroni . . . . .	3
monoRelation . . . . .	4
monoSummary . . . . .	5
monoUpDown . . . . .	7
statBootstrap . . . . .	8
wolak . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

demo_returns	<i>Asset returns used in Ang, Chen and Xing (RFS, 2006), sorted into ten portfolios.</i>
--------------	--

---

### Description

demo\_returns is a sample of asset returns from July 1963 to December 2001 of all stocks listed on the NYSE and is computed as follows: at the beginning of each month, stocks are sorted into deciles using estimates of beta based on the past year of daily returns, and value-weighted portfolios are formed. Two tests from Wolak (1989, JoE) of inequality constraints in linear econometric models.

### Usage

```
data(demo_returns)
```

### References

Patton, A. and Timmermann, A. (2010): Monotonicity in asset returns: New tests with applications to the term structure, the CAPM, and portfolio sorts. *Journal of Financial Economics*, **98**, No. 3, p. 605–625. doi: [10.1016/j.jfineco.2010.06.006](https://doi.org/10.1016/j.jfineco.2010.06.006).

Ang, A., Chen, J. and Xing, Y. (2006): Downside Risk. *Review of Financial Studies*, **19**, No. 4, p. 1191–1239. doi: [10.1093/rfs/hhj035](https://doi.org/10.1093/rfs/hhj035).

### Examples

```
## load demo data
data(demo_returns)

## calculate the mean difference return between the top and bottom portfolio
mean(demo_returns[, ncol(demo_returns)] - demo_returns[, 1])
```

---

monoBonferroni	Test of weak monotonicity using Bonferroni bounds
----------------	---

---

### Description

monoBonferroni implements the test of weak monotonicity using Bonferroni bounds described in Patton & Timmermann (2010, JFE): **Test 1:**  $H0^* : d1 \geq 0, d2 \geq 0, \dots, dK \geq 0$  vs.  $H1^* : dj < 0 \text{ for some } j = 1, 2, \dots, K$

**Test 2:**  $H0^{**} : d1 \leq 0, d2 \leq 0, \dots, dK \leq 0$  vs.  $H1^{**} : dj > 0 \text{ for some } j = 1, 2, \dots, K$ .

### Usage

```
monoBonferroni(data, difference = FALSE)
```

### Arguments

data	an object of class "matrix" (or one that can be coerced to that class): asset returns or differences in asset returns for the sorting application.
difference	An object of class "logical": If data is already differences in asset returns, use TRUE. Otherwise data will be transformed to difference returns $r_p(n+1) - r_p(n)$ between portfolio $n+1$ and portfolio $n$

### Value

monoBonferroni returns an object of class "list"

The returning list contains p-values (see Note) using Bonferroni-bounds for the two statistical tests described above:

TestOnePvalBonferroni:  
p-value for  $H0^*$  of Test 1.

TestTwoPvalBonferroni:  
p-value for  $H0^{**}$  of Test 2.

### Note

The "Bonferroni p-values" are in the sense that we reject the null hypothesis if they are less than the size of the test. NOTE of course that unlike usual p-values these won't be uniformly distributed between 0 and 1 under the null hypothesis. In fact, they do not even have to lie in  $[0,1]$  - they could be lesser than 0 or greater than 1. In such a case, monoBonferroni returns  $\min(\text{pvalBonferroni}, 1)$  if  $\text{pvalBonferroni} > 1$  and  $\max(\text{pvalBonferroni}, 0)$  if  $\text{pvalBonferroni} < 0$ .

## References

Patton, A. and Timmermann, A. (2010): Monotonicity in asset returns: New testes with applications to the term structure, the CAPM, and portfolio sorts. *Journal of Financial Economics*, **98**, No. 3, p. 605–625. doi: [10.1016/j.jfineco.2010.06.006](https://doi.org/10.1016/j.jfineco.2010.06.006).

Bonferroni, Carlo E. (1936): Teoria statistica delle classi e calcolo delle probabilita. [Statistical Class Theory and Calculation of Probability] *Pubbl. d. R. Ist. Super. di Sci. Econom. e Commerciali di Firenze*, **8**, p. 1–62.

## Examples

```
## load non-difference return data and calculate the p-value for H0* of Test 1.
data(demo_returns)
tmp <- monoBonferroni(demo_returns, difference = FALSE)
tmp$TestOnePvalBonferroni
```

---

monoRelation

*Testing the monotonic relationship in asset returns*

---

## Description

monoRelation implements the 'monotonic relationship' tests from Patton & Timmermann (2010, JFE). We define  $\Delta_i = E[r(i, t)] - E[r(i - 1, t)]$  and test

$H_0 : \Delta \leq 0$

vs.

$H_1 : \min(i = 1..N)\Delta_i > 0$

## Usage

```
monoRelation(data, bootstrapRep = 1000, increasing = TRUE,
  difference = FALSE, block_length)
```

## Arguments

data	an object of class "matrix" (or one that can be coerced to that class): asset returns or differences in asset returns which are sorted in a maximum of 15 portfolios. Each column of the matrix 'data' represents a single portfolio. data is therefore limited to a 15 columns.
bootstrapRep	A numeric scalar: the number of used bootstrap samples.
increasing	An object of class "logical": Assume an increasing or a decreasing pattern in monotonicity for the returns of the sorted portfolios.
difference	An object of class "logical": If data is already differences in asset returns, use TRUE. Otherwise data will be transformed to difference returns $r_p(n + 1) - r_p(n)$ between portfolio $n + 1$ and portfolio $n$
block_length	A numeric scalar: The average length of the block to use for the stationary bootstrap. This parameter is related to how much serial correlation is in your data. Use 10/6/3/2 as the block length if data is measured in daily/monthly/quarterly/annual returns.

**Value**

monoRelation returns an object of class "matrix".

The returning matrix consists of the following components:

matrix	a 4x2 matrix. The values of the first column are non-studentised, the values of the second column are studentised. Row (1): the t-statistic associated with a t-test that $\Delta_i \leq 0$ Row (2): the p-value associated with a t-test that $\Delta_i \leq 0$ Row (3): the MR test p-value from the proposed test, based on adjacent portfolios Row (4): the MR test p-value from the proposed test, on all pair-wise portfolio comparisons
--------	--

**References**

Patton, A. and Timmermann, A. (2010): Monotonicity in asset returns: New testes with applications to the term structure, the CAPM, and portfolio sorts. *Journal of Financial Economics*, **98**, No. 3, p. 605–625. doi: [10.1016/j.jfineco.2010.06.006](https://doi.org/10.1016/j.jfineco.2010.06.006).

**Examples**

```
## load non-difference return data and apply test with daily returns.
data(demo_returns)
monoRelation(demo_returns, block_length = 10)
```

---

monoSummary

*Summary of Patton and Timmermann monotonicity (JoE, 2010) tests*

---

**Description**

monoSummary implements the test for monotonicity in asset returns, based on portfolio sorts in (JoE, 2010)

**Usage**

```
monoSummary(data, bootstrapRep = 1000, wolakRep = 100,
  increasing = TRUE, difference = FALSE, plot = FALSE, block_length)
```

**Arguments**

data	an object of class "matrix" (or one that can be coerced to that class): asset returns or differences in asset returns which are sorted in a maximum of 15 portfolios. Each column of the matrix 'data' represents a single portfolio. data is therefore limited to a 15 columns.
bootstrapRep	A numeric scalar: the number of bootstrap samples.
wolakRep	A numeric scalar, stating the number of simulations to use to estimate the weight function in the weighted-sum of chi-square variables.
increasing	An object of class "logical": Assume an increasing or a decreasing pattern in monotonicity for the sorted portfolios.

difference	An object of class "logical": If data is already differences in asset returns, use TRUE. Otherwise data will be transformed to difference returns $r_p(n + 1) - r_p(n)$ between portfolio $n + 1$ and portfolio $n$
plot	An object of class "logical": If plot is TRUE, a plot is generated of the average returns on sorted portfolios with the p-value of the test on monotonicity from monoRelation.R. Otherwise data will be transformed to difference returns $r_p(n + 1) - r_p(n)$ between portfolio $n + 1$ and portfolio $n$
block_length	A numeric scalar: The average length of the block to use for the stationary bootstrap. This parameter is related to how much serial correlation is in your data. Use 10/6/3/2 as the block length if data is measured in daily/monthly/quarterly/annual returns.

## Value

monoSummary returns an object of class "data.frame".

The returning value of "monoSummary" is a "data.frame" containing the following components:

TopMinusBottom	Mean difference return between top and bottom portfolio.
t_stat	the residuals, that is response minus fitted values.
t_pval	the fitted mean values.
MR_pval	the numeric rank of the fitted linear model.
MRall_pval	the numeric rank of the fitted linear model.
UP_pval	studentized p-value from Patton and Timmermanns (JoE, 2010) "Up and Down" test for assumed increasing monotonicity pattern and using absolute difference returns.
DOWN_pval	studentized p-value from Patton and Timmermanns (JoE, 2010) "Up and Down" test for assumed decreasing monotonicity pattern and using absolute difference returns.
Wolak_pval	p-value "TestOnePvalueWolak" for $H_0^*$ of Test 1 in wolak.R
Bonferroni_pval	p-value for $H_0^*$ of Test 1 from monoBonferroni.R.

## References

Patton, A. and Timmermann, A. (2010): Monotonicity in asset returns: New testes with applications to the term structure, the CAPM, and portfolio sorts. *Journal of Financial Economics*, **98**, No. 3, p. 605–625. doi: [10.1016/j.jfineco.2010.06.006](https://doi.org/10.1016/j.jfineco.2010.06.006).

## See Also

[monoRelation](#), [monoUpDown](#), [wolak](#).

**Examples**

```
## load daily non-difference return data.
## test an increasing pattern of monotonicity

data(demo_returns)
monoSummary(demo_returns, increasing = TRUE, block_length = 10)
```

---

monoUpDown	<i>Up and Down test</i>
------------	-------------------------

---

**Description**

monoUpDown implements the 'Up and Down' tests from Patton & Timmermann (2010, JFE) based on:

(1) sum of squared differences for positive diffs and negative diffs, (2) sum of absolute differences for positive diffs and negative diffs,

and uses the stationary bootstrap method from Politis & Romano (1994, JASA).

**Usage**

```
monoUpDown(data, difference = FALSE, bootstrapRep = 1000, block_length)
```

**Arguments**

data	an object of class " <b>matrix</b> " (or one that can be coerced to that class): asset returns or differences in asset returns which are sorted in a maximum of 15 portfolios. Each column of the matrix 'data' represents a single portfolio. data is therefore limited to a 15 columns.
difference	An object of class " <b>logical</b> ": If data is already differences in asset returns, use TRUE. Otherwise data will be transformed to difference returns $r_p(n+1) - r_p(n)$ between portfolio $n+1$ and portfolio $n$
bootstrapRep	A numeric scalar: the number of bootstrap samples.
block_length	A numeric scalar: The average length of the block to use for the stationary bootstrap. This parameter is related to how much serial correlation is in your data. Use 10/6/3/2 as the block length if data is measured in daily/monthly/quarterly/annual returns.

**Value**

monoUpDown returns an object of class "**matrix**":

"matrix": A named 4x2 matrix with the bootstrap p-values from a test for a monotonic relationship. The first row contains p-values for squared diffs in an assumed increasing monotonic pattern, the second row respectively for a decreasing pattern.

The third row contains p-values for absolute differences in an assumed increasing monotonic pattern, the fourth row respectively for a decreasing pattern. The first column gives p-values which are not studentised, the second column the equivalent studentised p-values.

## References

Patton, A. and Timmermann, A. (2010): Monotonicity in asset returns: New testes with applications to the term structure, the CAPM, and portfolio sorts. *Journal of Financial Economics*, **98**, No. 3, p. 605–625. doi: [10.1016/j.jfineco.2010.06.006](https://doi.org/10.1016/j.jfineco.2010.06.006).

Wolak, Frank A. (1989): Testing Inequality Constraints in Linear Econometric Models. *Journal of Econometrics*, **41**, p. 205–235. doi: [10.1016/03044076\(89\)900948](https://doi.org/10.1016/03044076(89)900948).

## Examples

```
## load demo data and apply monoUpDown with daily data, which are not yet in differences
data(demo_returns)
test <- monoUpDown(demo_returns,block_length = 10)
```

---

statBootstrap	<i>Stationary bootstrap method</i>
---------------	------------------------------------

---

## Description

statBootstrap implements the stationary bootstrap method from Politis & Romano (1994, JASA). This function generates bootstrap samples of the matrix data and returns the time indices for each sample.

## Usage

```
statBootstrap(T, bootstrapRep = 1000, block_length)
```

## Arguments

T	A scalar, the number of time series observations to generate.
bootstrapRep	A numeric scalar: the number of used bootstrap samples.
block_length	A numeric scalar: The average length of the block to use for the stationary bootstrap. This parameter is related to how much serial correlation is in your data. Use 10/6/3/2 as the block length if data is measured in daily/monthly/quarterly/annual returns.

## Value

statBootstrap returns an object of class "matrix":

"matrix": A "T x bootstrapRep" matrix of time indices for each bootstrap sample.



## References

Politis, Dimitris N. & Romano, Joseph P. (1994): The Stationary Bootstrap. *Journal of The American Statistical Association*, **89**, No. 428, p. 1303–1313. doi: [10.1080/01621459.1994.10476870](https://doi.org/10.1080/01621459.1994.10476870).

## Examples

```
## Assuming daily return data for 100 time series observations.
## The returning matrix for default settings contains 1,000 bootstrap samples.
bootstrap_sample <- statBootstrap(T = 100, block_length = 10)

## 200 bootstrap samples using monthly return data with 250 time series observations.
bootstrap_sample <- statBootstrap(T = 250, bootstrapRep = 200, block_length = 6)
```

---

wolak

*Testing inequality constraints in linear econometric models*


---

## Description

wolak implements two tests from Wolak (1989, JoE) of inequality constraints in linear econometric models.

**Test 1:**  $H0^* : d1 \geq 0, d2 \geq 0, \dots, dK \geq 0$  vs.  $H1^* : (d1, d2, \dots, dK) \in R^K$ , (ie : *general alternative*)

**Test 2:**  $H0^{**} : d1 = d2 = \dots = dK = 0$  vs.  $H1^{**} : d1 > 0, d2 > 0, \dots, dK > 0$ .

## Usage

```
wolak(data, increasing = TRUE, difference = FALSE, wolakRep = 100)
```

## Arguments

data	an object of class " <b>matrix</b> " (or one that can be coerced to that class): asset returns or differences in asset returns for the sorting application.
increasing	An object of class " <b>logical</b> ": Assume an increasing or a decreasing pattern in monotonicity for the sorted portfolios. If a decreasing pattern is assumed, then $H0^*$ of Test 1 changes to $H0^* : d1 \leq 0, d2 \leq 0, \dots, dK \leq 0$ and respectively $H1^{**}$ of Test 2 changes to $H1^{**} : d1 < 0, d2 < 0, \dots, dK < 0$ .
difference	An object of class " <b>logical</b> ": If data is already differences in asset returns, use TRUE. Otherwise data will be transformed to difference returns $r_p(n + 1) - r_p(n)$ between portfolio $n + 1$ and portfolio $n$
wolakRep	A numeric scalar, stating the number of simulations to use to estimate the weight function in the weighted-sum of chi-square variables.

## Details

Currently supported as input type of data are classes "`matrix`", "`data.frame`", `ts`, `xts` and `zoo`.

Using demo data shows for `wolakRep`, that results do not change much at all for 100 or 1000 simulations, but the running time dramatically increases with the number of simulations. However, for robust results a minimum of 100 runs is highly recommended.

The HAC estimator of the covariance matrix of follows the adjustment of Newey-West (1987, 1994). The kernel ues is "Bartlett". See [NeweyWest](#) for further information.

## Value

`wolak` returns an object of class "`list`".

The returning list contains p-values for the following components:

TestOnePvalueWolak:

p-value for  $H_0^*$  of Test 1.

TestTwoPvalueWolak:

p-value for  $H_0^{**}$  of Test 2.

## References

Wolak, Frank A. (1989): Testing Inequality Constraints in Linear Econometric Models. *Journal of Econometrics*, **41**, p. 205–235. doi: [10.1016/03044076\(89\)900948](https://doi.org/10.1016/03044076(89)900948).

## Examples

```
## load demo data and apply Wolak tests

data(demo_returns)
tmp <- wolak(demo_returns)
tmp$TestOnePvalueWolak

## transform existing data (asset returns) into difference returns before applying wolak()
## as data is finally in difference returns, apply wolak() with difference = TRUE

data <- demo_returns[, 2:ncol(demo_returns)] - (demo_returns[, 1:(ncol(demo_returns) - 1)])
wolak(data, difference = TRUE)
```

# Index

`class`, [3](#), [5–8](#), [10](#)

`data.frame`, [10](#)

`demo_returns`, [2](#)

logical, [3–7](#), [9](#)

matrix, [3–5](#), [7](#), [9](#), [10](#)

`monoBonferroni`, [3](#)

`monoRelation`, [4](#), [6](#)

`monoSummary`, [5](#)

`monoUpDown`, [6](#), [7](#)

`NeweyWest`, [10](#)

`statBootstrap`, [8](#)

`wolak`, [6](#), [9](#)