Package ‘npExact’

July 4, 2015

Type Package

Title Nonparametric (or distribution-free) hypothesis tests

Version 0.2

Date 2015-07-03

Description This package contains several new hypothesis tests, which do not make assumptions on the underlying distributions.

Depends R (>= 2.10.0), stats

License GPL-2

R topics documented:

`npExact-package` *Nonparametric hypothesis tests*

Description

`npExact` provides distribution-free hypothesis tests.

Details

Package:	npExact
Type:	Package
Version:	0.2
Date:	2013-09-26
License:	GPL-2

This package contains several new hypothesis tests, which do not require that the user makes assumptions on the underlying distributions.

However, all tests except `npStochin` can only be applied if there are exogenously given bounds known to the user before gathering the data such that it is known by definition of the underlying process that all observations lie within these bounds.

So for instance, if the data involves percentages then the lower bound is 0 and the upper bound is 100, by definition of the data and not something (like normality) that cannot be deduced from the properties of the data.

Author(s)

Karl Schlag, Oliver Reiter, Peter Saffert, Christian Pechhacker, Simona Jokubauskaite, Tautvilas Janusauskas

References

Karl Schlag, A New Method for Constructing Exact Tests without Making any Assumptions (August, 2008) Department of Economics and Business Working Paper 1109, Universitat Pompeu Fabra

See Also

http://homepage.univie.ac.at/karl.schlag/research/statistics/exacthypothesistesting8.pdf

http://homepage.univie.ac.at/karl.schlag/research/statistics.html

Examples

```
## npMeanPaired
## test whether pain after the surgery is less than before the surgery
data(pain)
npMeanPaired(pain$before, pain$after, lower = 0, upper = 100)

## npMeanSingle
## test whether Americans gave more than 5 dollars in a round of
## the Ultimatum game
data(bargaining)
us_offers <- bargaining$US
npMeanSingle(us_offers, mu = 5, lower = 0, upper = 10, alternative =
"greater", ignoreNA = TRUE) ## no rejection

## npMeanUnpaired
## test whether countries with french origin score lower than
## countries with no french origin
```

```
data(french)
npMeanUnpaired(french[,1], french[,2], alternative = "less", ignoreNA =
TRUE)

## npStochin
data(french)
x <- french[, 1]
y <- french[, 2]
npStochin(x, y, ignoreNA = TRUE)

## npVarianceSingle
## see if the minority share holder shores have a variance greater
## than 0.05
data(mshscores)
scores <- as.vector(as.matrix(mshscores))
npVarianceSingle(scores, lower = 0, upper = 1, v = 0.05, ignoreNA = TRUE)
```

`bargaining` *Amount sent in the Ultimatum Game*

Description

The Ultimatum game was played separately in four different countries. This data contains the offers of 30 students in Israel and 27 in the United States on a scale from 0 to 10. This dataset is taken from Roth et al. (1991).

Usage

```
bargaining
```

Format

A data frame containing 30 observations for Israel and 27 for the US.

References

Roth, A. E., Prasnikar, V., Okuno-Fujiwara, M., & Zamir, S. (1991). Bargaining and market behavior in Jerusalem, Ljubljana, Pittsburgh, and Tokyo: An experimental study. The American Economic Review, 1068-1095.

`french` *Indices of minority shareholder protection of countries with civil law with and without french origin.*

Description

This data contains the indices of minority shareholder protection on a scale from 0 to 1 in 51 countries with civil law, differentiating between those with (32 observations) and those without (19 observations) french origin. A higher value of the index means that country is more protected. The data set is taken from Djankov et al. (2008).

Usage

```
french
```

Format

A dataframe containing 32 observations of countries with french origin and 19 countries without french origin.

References

Djankov, S., La Porta, R., Lopez-de-Silanes, F., & Shleifer, A. (2008). The law and economics of self-dealing. Journal of financial economics, 88(3), 430-465.

---

mshscores *Indices of minority shareholder protection of countries with common and with civil law.*

Description

This data contains the indices of minority shareholder protection on a scale from 0 to 1 in 51 countries with civil law and 21 countries with common loaw. A higher value of the index means that country is more protected. The data set is taken from Djankov et al. (2008).

Usage

```
mshscores
```

Format

A dataframe containing 51 observations for civil law and 21 for common law.

References

Djankov, S., La Porta, R., Lopez-de-Silanes, F., & Shleifer, A. (2008). The law and economics of self-dealing. Journal of financial economics, 88(3), 430-465.

---

npMeanPaired *A test for the mean difference between two bounded random variables given matched pairs.*

Description

This test requires that the user knows bounds before gathering the data such that the properties of the data generating process imply that all observations will be within these bounds. The data input consists of pairs of observations, each pair consisting of an observation of each random variable, different pairs being independently generated. No further distributional assumptions are made.

Under alternative = "greater", it is a test of the null hypothesis $H_0 : E(x_1) \leq E(x_2)$ against the alternative hypothesis $H_1 : E(x_1) > E(x_2)$.

Usage

```
npMeanPaired(x1, x2, lower = 0, upper = 1, alpha = 0.05,
             alternative = "greater", epsilon = 1 * 10^(-6),
             iterations = 5000)
```

Arguments

`x1, x2`	the (non-empty) numerical data vectors which contain the variables to be tested. The first values of the vectors are assumed to be the first matched pair of observations, the second values the second matched pair and so on.
`lower, upper`	the theoretical lower and upper bounds on the data outcomes known ex-ante before gathering the data.
`alpha`	the type I error.
`alternative`	a character string describing the alternative hypothesis, can take values "greater", "less" or "two.sided".
`iterations`	the number of iterations used, should not be changed if the exact solution should be derived
`epsilon`	the tolerance in terms of probability of the Monte Carlo simulations.

Details

This test uses the known bounds of the variables to transform the data into [0, 1]. Then a random transformation is used to turn the data into binary-valued variables. On this variables the exact McNemar Test with level `pseudoalpha` is performed and the result recorded. The random transformation and the test are then repeated `iterations` times. If the average rejection probability `probrej` of the iterations is at least `theta`, then the null hypothesis is rejected. If however `probrej` is too close to the threshold `theta` then the number of iterations is increased. The algorithm keeps increasing the number of iterations until the bound on the mistake involved by running these iterations is below `epsilon`. This error epsilon is incorporated into the overall level `alpha` in order to maintain that the test is exact.

`theta` (and a value `mu` of the difference between the two means in the set of the alternative hypothesis) is found in an optimization procedure. `theta` and `mu` are chosen as to maximize the set of data generating processes belonging to the alternative hypothesis that yield type II error probability below 0.5. Please see the cited paper below for further information.

Value

A list with class "nphtest" containing the following components:

`method`	a character string indicating the name and type of the test that was performed.
`data.name`	a character string giving the name(s) of the data.
`alternative`	a character string describing the alternative hypothesis.
`estimate`	the sample means of the given data.
`probrej`	numerical estimate of the rejection probability of the randomized test, derived by taking an average of `iterations` realizations of the rejection probability.
`bounds`	the lower and upper bounds of the variables.
`null.value`	the specified hypothesized value of the difference of the variable means.
`alpha`	the type I error.

`theta`	the parameter that minimizes the type II error.
`pseudoalpha`	`theta*alpha`, this is the level used when calculating the average rejection probability during the iterations.
`rejection`	logical indicator for whether or not the null hypothesis can be rejected.
`iterations`	the number of iterations that were performed.

Author(s)

Karl Schlag, Christian Pechhacker and Oliver Reiter

References

Schlag, Karl H. 2008, A New Method for Constructing Exact Tests without Making any Assumptions, Department of Economics and Business Working Paper 1109, Universitat Pompeu Fabra. Available at http://www.econ.upf.edu/en/research/onepaper.php?id=1109.

See Also

http://homepage.univie.ac.at/karl.schlag/research/statistics.html

Examples

```
## test whether pain after the surgery is less than before the surgery
data(pain)
npMeanPaired(pain$before, pain$after, lower = 0, upper = 100)

## when the computer was used in the surgery
before_pc <- pain[pain$pc == 1, "before"]
after_pc <- pain[pain$pc == 1, "after"]
npMeanPaired(before_pc, after_pc, lower = 0, upper = 100)

## test whether uncertainty decreased from the first to the second round
data(uncertainty)
npMeanPaired(uncertainty$w1, uncertainty$w2, upper = 60) ## or
with(uncertainty, npMeanPaired(w1, w2, upper = 60))
```

---

npMeanSingle — *A test for the mean of a bounded random variable based on a single sample of iid observations.*

Description

This test requires that the user knows upper and lower bounds before gathering the data such that the properties of the data generating process imply that all observations will be within these bounds. The data input consists of a sequence of observations, each being an independent realization of the random variable. No further distributional assumptions are made.

For any μ that lies between the two bounds, under alternative = "greater", it is a test of the null hypothesis $H_0 : E(X) \leq \mu$ against the alternative hypothesis $H_1 : E(X) > \mu$.

Usage

```
npMeanSingle(x, mu, lower = 0, upper = 1, iterations = 5000,
             alpha = 0.05, alternative = "greater",
             epsilon = 1 * 10^(-6), ignoreNA = FALSE)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`mu`	threshold value for the null hypothesis.
`lower, upper`	the theoretical lower and upper bounds on the data outcomes known ex-ante before gathering the data.
`iterations`	the number of iterations used, should not be changed if the exact solution should be derived
`alpha`	the type I error.
`alternative`	a character string describing the alternative hypothesis, can take values "greater", "less" or "two.sided".
`epsilon`	the tolerance in terms of probability of the Monte Carlo simulations.
`ignoreNA`	if `TRUE`, NA values will be omitted. Default: `FALSE`

Details

Using the known bounds, the data is transformed to lie in [0, 1] using an affine transformation. Then the data is randomly transformed into a new data set that has values 0, `mu` and 1 using a mean preserving transformation. The exact randomized binomial test is then used to calculate the rejection probability of this under new data when level is `theta*alpha`. This random transformation is repeated `iterations` times. If the average rejection probability is greater than theta, one can reject the null hypothesis. If however the average rejection probability is too close to theta then the iterations are continued. The values of `theta` and a value of `mu` in the alternative hypothesis is found in an optimization procedure to maximize the set of parameters in the alternative hypothesis under which the type II error probability is below 0.5. Please see the cited paper below for further information.

Value

A list with class "nphtest" containing the following components:

`method`	a character string indicating the name and type of the test that was performed.
`data.name`	a character string giving the name(s) of the data.
`alternative`	a character string describing the alternative hypothesis.
`estimate`	the estimated mean or difference in means depending on whether it was a one-sample test or a two-sample test.
`probrej`	numerical estimate of the rejection probability of the randomized test, derived by taking an average of `iterations` realizations of the rejection probability.
`bounds`	the lower and upper bounds of the variables.
`null.value`	the specified hypothesized value of the correlation between the variables.
`alpha`	the type I error
`theta`	the parameter that minimizes the type II error.

pseudoalpha	theta*alpha, this is the level used when calculating the average rejection probability during the iterations.
rejection	logical indicator for whether or not the null hypothesis can be rejected.
iterations	the number of iterations that were performed.

Author(s)

Karl Schlag, Peter Saffert and Oliver Reiter

References

Schlag, Karl H. 2008, A New Method for Constructing Exact Tests without Making any Assumptions, Department of Economics and Business Working Paper 1109, Universitat Pompeu Fabra. Available at http://www.econ.upf.edu/en/research/onepaper.php?id=1109.

See Also

http://homepage.univie.ac.at/karl.schlag/research/statistics.html

Examples

```
## test whether Americans gave more than 5 dollars in a round of
## the Ultimatum game
data(bargaining)
us_offers <- bargaining$US
npMeanSingle(us_offers, mu = 5, lower = 0, upper = 10, alternative =
"greater", ignoreNA = TRUE) ## no rejection

## test if the decrease in pain before and after the surgery is smaller
## than 50
data(pain)
pain$decrease <- with(pain, before - after)
without_pc <- pain[pain$pc == 0, "decrease"]
npMeanSingle(without_pc, mu = 50, lower = 0, upper = 100,
alternative = "less")
```

---

npMeanUnpaired — *A test for comparing the means of two bounded random variables given two independent samples*

Description

This test requires that the user knows upper and lower bounds before gathering the data such that the properties of the data generating process imply that all observations will be within these bounds. The data input consists of a sequence of independent observations for each random variable, the two sequences being generated independently. No further distributional assumptions are made.

This is a test of the null hypothesis: $H_0 : E(X_1) \leq E(X_2)$ against $H_1 : E(X_1) > E(X_2)$.

Usage

```
npMeanUnpaired(x1, x2, lower = 0, upper = 1, iterations = 5000, alpha = 0.05,
                alternative = "greater", epsilon = 1 * 10^(-6), ignoreNA = FALSE)
```

Arguments

`x1, x2`	the (non-empty) numerical data vectors which contain the variables to be tested.
`lower, upper`	the theoretical lower and upper bounds on the data outcomes known ex-ante before gathering the data.
`iterations`	the number of iterations used, should not be changed if the exact solution should be derived.
`alpha`	the type I error.
`alternative`	a character string describing the alternative hypothesis, can take values "greater", "less" or "two.sided".
`epsilon`	the tolerance in terms of probability of the Monte Carlo simulations.
`ignoreNA`	if `TRUE`, NA values will be omitted. Default: `FALSE`

Details

This test uses the known bounds of the variables to transform the data into [0, 1]. Then a random transformation is used to turn the data into binary-valued variables. On this variables the exact Fischer-Tocher Test with level `pseudoalpha` is performed and the result recorded. The random transformation and the test are then repeated `iterations` times. If the average rejection probability `probrej` of the iterations is at least `theta`, then the null hypothesis is rejected. If however `probrej` is too close to the threshold `theta` then the number of iterations is increased. The algorithm keeps increasing the number of iterations until the bound on the mistake involved by running these iterations is below `epsilon`. This error epsilon is incorporated into the overall level `alpha` in order to maintain that the test is exact.

`theta` is found in an optimization procedure. `theta` is chosen as to bring the type II error to 0.5. Please see the cited paper below for further information.

Value

A list with class "nphtest" containing the following components:

`method`	a character string indicating the name and type of the test that was performed.
`data.name`	a character string giving the name(s) of the data.
`alternative`	a character string describing the alternative hypothesis.
`estimate`	the sample means of the two variables.
`probrej`	numerical estimate of the rejection probability of the randomized test, derived by taking an average of `iterations` realizations of the rejection probability.
`bounds`	the lower and upper bounds of the variables.
`null.value`	the specified hypothesized value of the correlation between the variables.
`alpha`	the type I error.
`theta`	the parameter that minimizes the type II error.
`pseudoalpha`	`theta*alpha`, this is the level used when calculating the average rejection probability during the iterations
`rejection`	logical indicator for whether or not the null hypothesis can be rejected
`iterations`	the number of iterations that were performed.

Author(s)

Karl Schlag, Christian Pechhacker, Peter Saffert and Oliver Reiter

References

Karl Schlag (2008), A New Method for Constructing Exact Tests without Making any Assumptions. Available at http://www.econ.upf.edu/en/research/onepaper.php?id=1109.

See Also

http://homepage.univie.ac.at/karl.schlag/research/statistics.html

Examples

```
## test whether countries with french origin score lower than
## countries with no french origin
data(french)
npMeanUnpaired(french[,1], french[,2], alternative = "less", ignoreNA =
TRUE)

## test whether American tend to be more generous than Isrealis
## in a round of the Ultimatum game
data(bargaining)
npMeanUnpaired(bargaining$US, bargaining$IS, lower = 0, upper = 10, ignoreNA = TRUE)
```

---

npStochin — *A test of a stochastic inequality given two independent samples*

Description

The data input consists of a sequence of independent realizations observations of each random variable, observations of the different sequences also being independent.

Given $-1 < d < 1$ it is a test of the null hypothesis $H_0 : P(X_2 > X_1) \le P(X_2 < X_1) + d$ against the alternative hypothesis $H_1 : P(X_2 > X_1) > P(X_2 < X_1) + d$.

Usage

```
npStochin(x1, x2, d = 0, alternative = "greater", iterations = 5000,
          alpha = 0.05, epsilon = 1 * 10^(-6), ignoreNA = FALSE)
```

Arguments

`x1, x2`	the (non-empty) numerical data vectors which contain the variables to be tested.
`d`	the maximal difference in probabilities assumed $H_0 : P(X_2 > X_1) - P(X_2 < X_1) <= d$. Default is 0.
`alternative`	a character string describing the alternative hypothesis. Default is "greater". If "less" is given, `x1` and `x2` are switched for each other.
`iterations`	the number of iterations used, should not be changed if the exact solution should be derived.
`alpha`	the type I error.
`epsilon`	the tolerance in terms of probability of the Monte Carlo simulations.
`ignoreNA`	if `TRUE`, NA values will be omitted. Default: `FALSE`

Details

The data is randomly matched into pairs and then treats them as matched pairs. The number of pairs is equal to the number of observations in the smaller sequence. The exact randomized test is then used to determine if sufficiently many occurrences of $x_2 > x_1$ occur when compared to how often $x_2 < x_1$ occurs, using level `theta*alpha`. The matching into pairs is repeated `iterations` times. The test gives a rejection of the average rejection probability in these iterations lies above `theta`. If the average rejection probability lies too close to theta then the number of iterations is increased.

`theta` is determined to maximize the set of differences $P(X_2 > X_1) - P(X_2 < X_1)$ belonging to the alternative hypothesis in which the type II error probability lies below 0.5. For more details see the paper.

Value

A list with class "nphtest" containing the following components:

`method`	a character string indicating the name and type of the test that was performed.
`data.name`	a character string giving the name(s) of the data.
`alternative`	a character string describing the alternative hypothesis.
`estimate`	an estimate of $P(x_2 > x_1) - P(x_2 < x_1)$.
`probrej`	numerical estimate of the rejection probability of the randomized test, derived by taking an average of `iterations` realizations of the rejection probability.
`bounds`	the lower and upper bounds of the variables.
`null.value`	the specified hypothesized value of the correlation between the variables.
`alpha`	the type I error.
`theta`	the parameter that minimizes the type II error.
`pseudoalpha`	`theta*alpha`, this is the level used when calculating the average rejection probability during the iterations.
`rejection`	logical indicator for whether or not the null hypothesis can be rejected.
`iterations`	the number of iterations that were performed.

Author(s)

Karl Schlag, Peter Saffert and Oliver Reiter

References

Schlag, Karl H. 2008, A New Method for Constructing Exact Tests without Making any Assumptions, Department of Economics and Business Working Paper 1109, Universitat Pompeu Fabra. Available at http://www.econ.upf.edu/en/research/onepaper.php?id=1109.

See Also

http://homepage.univie.ac.at/karl.schlag/research/statistics.html

Examples

```
data(french)
x <- french[, 1]
y <- french[, 2]
npStochin(x, y, ignoreNA = TRUE)
```

npVarianceSingle — *A test for the variance of a bounded random variable based on a single sample of iid observations.*

Description

This test requires that the user knows upper and lower bounds before gathering the data such that the properties of the data generating process imply that all observations will be within these bounds. The data input consists of a sequence of observations, each being an independent realization of the random variable. No further distributional assumptions are made.

This is a test of the null hypothesis $H_0 : Var(X) \leq v$ against $H_1 : Var(X) > v$.

Usage

```
npVarianceSingle(x, lower = 0, upper = 1, v,
      alternative = "greater", alpha = 0.05, iterations = 5000,
      epsilon = 1 * 10^(-6), ignoreNA = FALSE)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`lower, upper`	the theoretical lower and upper bounds on the data outcomes known ex-ante before gathering the data.
`v`	the value of the variance to be tested as $H_0 : Var(x) \leq v$.
`alternative`	a character string describing the alternative hypothesis, can take values "greater", "less" or "two.sided"
`alpha`	the type I error.
`iterations`	the number of iterations used, should not be changed if the exact solution should be derived.
`epsilon`	the tolerance in terms of probability of the Monte Carlo simulations.
`ignoreNA`	if `TRUE`, NA values will be omitted. Default: `FALSE`

Details

This test randomly matches the data into pairs, then computes for each pair the square of the difference and continues with the resulting sequence with half as many observations as npMeanSingle. See the cited paper for more information.

Value

A list with class "nphtest" containing the following components:

`method`	a character string indicating the name and type of the test that was performed.
`data.name`	a character string giving the name(s) of the data.
`alternative`	a character string describing the alternative hypothesis.
`estimate`	the estimated mean or difference in means depending on whether it was a one-sample test or a two-sample test.

`probrej`	numerical estimate of the rejection probability of the randomized test, derived by taking an average of `iterations` realizations of the rejection probability.
`bounds`	the lower and upper bounds of the variables.
`null.value`	the specified hypothesized value of the correlation between the variables.
`alpha`	the type I error.
`theta`	the parameter that minimizes the type II error.
`pseudoalpha`	`theta*alpha`, this is the level used when calculating the average rejection probability during the iterations.
`rejection`	logical indicator for whether or not the null hypothesis can be rejected.
`iterations`	the number of iterations that were performed.

Author(s)

Karl Schlag and Oliver Reiter

References

Karl Schlag (2008). Exact tests for correlation and for the slope in simple linear regressions without making assumptions. Available at http://www.econ.upf.edu/en/research/onepaper.php?id=1097.

See Also

http://homepage.univie.ac.at/karl.schlag/research/statistics.html

Examples

```
## see if the minority share holder shores have a variance greater
## than 0.05
data(mshscores)

scores <- as.vector(as.matrix(mshscores))
npVarianceSingle(scores, lower = 0, upper = 1, v = 0.05, ignoreNA = TRUE)
```

`pain` *Pain experienced before and after a knie operation*

Description

There are two ways to determine where to start an operation on a knee, either with a computer or manually. The data describes the pain experienced by the patients before and after the surgery.

Usage

```
pain
```

Format

A dataframe containing 50 observations. Column "pc" indicates if a computer was used (coded with "1") or not (coded with "0")

References

Sabeti-Aschraf, M., Dorotka, R., Goll, A., & Trieb, K. (2005). Extracorporeal shock wave therapy in the treatment of calcific tendinitis of the rotator cuff. The American journal of sports medicine, 33(9), 1365-1368.

`uncertainty` *Uncertainty in a game theoretical experiment.*

Description

In an experiment, subjects played a similar game twice. Choices could be between 110 and 170. Each time, before they made their own choice, they had to indicate an interval [L, U] that they believed would contain the choice of their opponent. They paid some additional money if the choice of their opponent was in the interval they specified, and were paid more the smaller this interval was. So the width W_i of this interval in round i gives an indication of how uncertain they are in round i. The data contains the interval width in round 1 and 2 which makes this a sample of matched pairs.

Usage

```
uncertainty
```

Format

A dataframe containing the 25 intervals in each round of the game.

References

Galbiati, R., Schlag, K., & van der Weele, J. Sanctions that Signal: an Experiment. Journal of Economic Behavior and Organization, Forthcoming

Index