Package 'robnptests'

May 9, 2026

Version 1.1.0

Type Package

Title Robust Nonparametric Two-Sample Tests for Location/Scale

Author Sermad Abbas [aut, cre] (ORCID:
<https://orcid.org/0000-0001-9162-9792>),
Barbara Brune [aut] (ORCID: <https://orcid.org/0000-0002-3154-8445>),
Roland Fried [aut] (ORCID: <https://orcid.org/0000-0002-9830-9713>)

Maintainer Sermad Abbas <abbas@statistik.tu-dortmund.de>

BugReports https://github.com/s-abbas/robnptests/issues

Description Implementations of several robust nonparametric two-sample tests
for location or scale differences. The test statistics are based on robust
location and scale estimators, e.g. the sample median or the Hodges-Lehmann estimators
as described in Fried & Dehling (2011) <doi:10.1007/s10260-011-0164-1>.
The p-
values can be computed via the permutation principle, the randomization principle, or by using
the asymptotic distributions of the test statistics under the null hypothesis, which ensures
(approximate) distribution independence of the test decision. To test for a difference in
scale, we apply the tests for location difference to transformed observa-
tions; see Fried (2012) <doi:10.1016/j.csda.2011.02.012>.
Random noise on a small range can be added to the original observations in order to
hold the significance level on data from discrete distributions.
The location tests assume homoscedasticity and the scale tests require the
location parameters to be zero.

License GPL (>= 2)

Depends R (>= 4.0.0)

URL https://github.com/s-abbas/robnptests

Encoding UTF-8

RoxygenNote 7.2.1

Imports Rdpack, gtools, robustbase, statmod, stats, utils, checkmate

RdMacros Rdpack

Suggests testthat, knitr, rmarkdown, usethis, covr

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2023-02-13 21:10:02 UTC

Contents

`hl1_test` *Two-sample location tests based on one-sample Hodges-Lehmann estimator*

Description

`hl1_test` performs a two-sample location test based on the difference of the one-sample Hodges-Lehmann estimators of both samples.

Usage

```
hl1_test(
  x,
  y,
  alternative = c("two.sided", "greater", "less"),
  delta = ifelse(scale.test, 1, 0),
  method = c("asymptotic", "permutation", "randomization"),
  scale = c("S1", "S2"),
  n.rep = 10000,
```

```
  na.rm = FALSE,
  scale.test = FALSE,
  wobble = FALSE,
  wobble.seed = NULL
)
```

Arguments

x	a (non-empty) numeric vector of data values.
y	a (non-empty) numeric vector of data values.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater", or "less".
delta	a numeric value indicating the true difference in the location or scale parameter, depending on whether the test should be performed for a difference in location or in scale. The default is `delta = 0` for a location test and `delta = 1` for a scale test. In case of `scale.test = TRUE`, `delta` represents the ratio of the squared scale parameters.
method	a character string specifying how the p-value is computed with possible values `"asymptotic"` for an asymptotic test based on a normal approximation, `"permutation"` for a permutation test, and `"randomization"` for a randomization test. The permutation test uses all splits of the joint sample into two samples of sizes `m` and `n`, while the randomization test draws `n.rep` random splits with replacement. The values `m` and `n` denote the sample sizes. If not specified explicitly, defaults to `"permutation"` if `m < 30`, `n < 30` and `n.rep >= choose(m + n, m)`, `"randomization"` if `m < 30`, `n < 30` and `n.rep < choose(m + n, m)`, and `"asymptotic"` if `m >= 30` and `n >= 30`.
scale	a character string specifying the scale estimator used for standardization of the test statistic; must be one of `"S1"` and `"S2"`. The default is `"S1"`. Ignored if `method = "asymptotic"`; see details for the definition of the scale estimators.
n.rep	an integer value specifying the number of random splits used to calculate the randomization distribution if `method = "randomization"`. This argument is ignored if `method = "permutation"` or `method = "asymptotic"`. The default is `n.rep = 10000`.
na.rm	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.
scale.test	a logical value to specify if the samples should be compared for a difference in scale. The default is `scale.test = FALSE`.
wobble	a logical value indicating whether the sample should be checked for duplicated values that can cause the scale estimate to be zero. If such values are present, uniform noise is added to the sample, see `wobble`. Only necessary for the permutation and randomization version of the test. The default is `wobble = FALSE`.
wobble.seed	an integer value used as a seed for the random number generation in case of `wobble = TRUE` or when `scale.test = TRUE` with one of the vectors `x` and `y` containing zeros. When no seed is specified, it is chosen randomly and printed in a message. The argument is ignored if `scale.test = FALSE` and/or `wobble = FALSE`.

Details

The test statistic for this test is based on the difference of the one-sample Hodges-Lehmann estimators of x and y, see hodges_lehmann. Three versions of the test are implemented: randomization, permutation, and asymptotic.

The test statistic for the permutation and randomization version of the test is standardized using a robust scale estimator, see (Fried and Dehling 2011).

With `scale = "S1"`, the scale is estimated by

$$S = med(|x_i - x_j| : 1 \leq i < j \leq m, |y_i - y_j|, 1 \leq i < j \leq n),$$

whereas `scale = "S2"` uses

$$S = med(|z_i - z_j| : 1 \leq i < j \leq m + n).$$

Here, $z = (z_1, ..., z_{m+n}) = (x_1 - med(x), ..., x_m - med(x), y_1 - med(y), ..., y_n - med(y))$ is the median-corrected sample.

The randomization distribution is based on randomly drawn splits with replacement. The function permp (Phipson and Smyth 2010) is used to calculate the p-value. For the asymptotic test, a transformed version of the difference of the HL1-estimators, which asymptotically follows a normal distribution, is used. For more details on the asymptotic test, see Fried and Dehling (2011).

For `scale.test = TRUE`, the test compares the two samples for a difference in scale. This is achieved by log-transforming the original squared observations, i.e. `x` is replaced by `log(x^2)` and `y` by `log(y^2)`. A potential scale difference then appears as a location difference between the transformed samples, see Fried (2012). Note that the samples need to have equal locations. The sample should not contain zeros to prevent problems with the necessary log-transformation. If it contains zeros, uniform noise is added to all variables in order to remove zeros and a message is printed.

If the sample has been modified (either because of zeros if `scale.test = TRUE` or `wobble = TRUE`), the modified samples can be retrieved using

`set.seed(wobble.seed); wobble(x, y)`.

Both samples need to contain at least 5 non-missing values.

Value

A named list with class "`htest`" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value for the test.
`estimate`	the one-sample Hodges-Lehmann estimates of `x` and `y` (if `scale.test = FALSE`) or of `log(x^2)` and `log(y^2)` (if `scale.test = TRUE`).
`null.value`	the specified hypothesized value of the mean difference/squared scale ratio.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating how the p-value was computed.
`data.name`	a character string giving the names of the data.

References

Phipson B, Smyth GK (2010). "Permutation p-values should never be zero: Calculating exact p-values when permutations are randomly drawn." *Statistical Applications in Genetics and Molecular Biology*, **9**(1), Article 39. doi:10.2202/15446115.1585.

Fried R, Dehling H (2011). "Robust nonparametric tests for the two-sample location problem." *Statistical Methods & Applications*, **20**(4), 409–422. doi:10.1007/s1026001101641.

Fried R (2012). "On the online estimation of piecewise constant volatilities." *Computational Statistics & Data Analysis*, **56**(11), 3080–3090. doi:10.1016/j.csda.2011.02.012.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Asymptotic HL1 test
hl1_test(x, y, method = "asymptotic", scale = "S1")

## Not run:
# HL12 test using randomization principle by drawing 1000 random permutations
# with replacement

hl1_test(x, y, method = "randomization", n.rep = 1000, scale = "S2")

## End(Not run)
```

`hl2_test` *Two-sample location tests based on two-sample Hodges-Lehmann estimator.*

Description

`hl2_test` performs a two-sample location test based on the two-sample Hodges-Lehmann estimator for shift.

Usage

```
hl2_test(
  x,
  y,
  alternative = c("two.sided", "greater", "less"),
  delta = ifelse(scale.test, 1, 0),
  method = c("asymptotic", "permutation", "randomization"),
  scale = c("S1", "S2"),
  n.rep = 10000,
  na.rm = FALSE,
```

```
  scale.test = FALSE,
  wobble = FALSE,
  wobble.seed = NULL
)
```

Arguments

x	a (non-empty) numeric vector of data values.
y	a (non-empty) numeric vector of data values.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater", or "less".
delta	a numeric value indicating the true difference in the location or scale parameter, depending on whether the test should be performed for a difference in location or in scale. The default is `delta = 0` for a location test and `delta = 1` for a scale test. In case of `scale.test = TRUE`, `delta` represents the ratio of the squared scale parameters.
method	a character string specifying how the p-value is computed with possible values `"asymptotic"` for an asymptotic test based on a normal approximation, `"permutation"` for a permutation test, and `"randomization"` for a randomization test. The permutation test uses all splits of the joint sample into two samples of sizes `m` and `n`, while the randomization test draws `n.rep` random splits with replacement. The values `m` and `n` denote the sample sizes. If not specified explicitly, defaults to `"permutation"` if `m < 30`, `n < 30` and `n.rep >= choose(m + n, m)`, `"randomization"` if `m < 30`, `n < 30` and `n.rep < choose(m + n, m)`, and `"asymptotic"` if `m >= 30` and `n >= 30`.
scale	a character string specifying the scale estimator used for standardization of the test statistic; must be one of `"S1"` and `"S2"`. The default is `"S1"`. Ignored if `method = "asymptotic"`; see details for the definition of the scale estimators.
n.rep	an integer value specifying the number of random splits used to calculate the randomization distribution if `method = "randomization"`. This argument is ignored if `method = "permutation"` or `method = "asymptotic"`. The default is `n.rep = 10000`.
na.rm	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.
scale.test	a logical value to specify if the samples should be compared for a difference in scale. The default is `scale.test = FALSE`.
wobble	a logical value indicating whether the sample should be checked for duplicated values that can cause the scale estimate to be zero. If such values are present, uniform noise is added to the sample, see `wobble`. Only necessary for the permutation and randomization version of the test. The default is `wobble = FALSE`.
wobble.seed	an integer value used as a seed for the random number generation in case of `wobble = TRUE` or when `scale.test = TRUE` with one of the vectors `x` and `y` containing zeros. When no seed is specified, it is chosen randomly and printed in a message. The argument is ignored if `scale.test = FALSE` and/or `wobble = FALSE`.

Details

The test statistic for this test is based on the two-sample Hodges-Lehmann estimator of `x` and `y`, see `hodges_lehmann_2sample`. Three versions of the test are implemented: randomization, permutation, and asymptotic.

The test statistic for the permutation and randomization version of the test is standardized using a robust scale estimator, see (Fried and Dehling 2011).

With `scale = "S1"`, the scale is estimated by

$$S = med(|x_i - x_j| : 1 \leq i < j \leq m, |y_i - y_j|, 1 \leq i < j \leq n),$$

whereas `scale = "S2"` uses

$$S = med(|z_i - z_j| : 1 \leq i < j \leq m + n).$$

Here, $z = (z_1, ..., z_{m+n}) = (x_1 - med(x), ..., x_m - med(x), y_1 - med(y), ..., y_n - med(y))$ is the median-corrected sample.

The randomization distribution is based on randomly drawn splits with replacement. The function `permp` (Phipson and Smyth 2010) is used to calculate the p-value. For the asymptotic test, a transformed version of the HL2-estimator, which asymptotically follows a normal distribution, is used. For more details on the asymptotic test, see Fried and Dehling (2011).

For `scale.test = TRUE`, the test compares the two samples for a difference in scale. This is achieved by log-transforming the original squared observations, i.e. `x` is replaced by `log(x^2)` and `y` by `log(y^2)`. A potential scale difference then appears as a location difference between the transformed samples, see Fried (2012). Note that the samples need to have equal locations. The sample should not contain zeros to prevent problems with the necessary log-transformation. If it contains zeros, uniform noise is added to all variables in order to remove zeros and a message is printed.

If the sample has been modified (either because of zeros if `scale.test = TRUE` or `wobble = TRUE`), the modified samples can be retrieved using

`set.seed(wobble.seed); wobble(x, y)`.

Both samples need to contain at least 5 non-missing values.

Value

A named list with class "`htest`" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value for the test.
`estimate`	the estimated location difference between `x` and `y` (if `scale.test = FALSE`) or of `log(x^2)` and `log(y^2)` (if `scale.test = TRUE`) based on the two-sample Hodges-Lehmann estimator.
`null.value`	the specified hypothesized value of the mean difference/squared scale ratio.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating how the p-value was computed.
`data.name`	a character string giving the names of the data.

References

Phipson B, Smyth GK (2010). "Permutation p-values should never be zero: Calculating exact p-values when permutations are randomly drawn." *Statistical Applications in Genetics and Molecular Biology*, **9**(1), Article 39. doi:10.2202/15446115.1585.

Fried R, Dehling H (2011). "Robust nonparametric tests for the two-sample location problem." *Statistical Methods & Applications*, **20**(4), 409–422. doi:10.1007/s1026001101641.

Fried R (2012). "On the online estimation of piecewise constant volatilities." *Computational Statistics & Data Analysis*, **56**(11), 3080–3090. doi:10.1016/j.csda.2011.02.012.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Asymptotic HL2 test
hl2_test(x, y, method = "asymptotic", scale = "S1")

## Not run:
# HL22 test using randomization principle by drawing 1000 random permutations
# with replacement

hl2_test(x, y, method = "randomization", n.rep = 1000, scale = "S2")

## End(Not run)
```

`hodges_lehmann` *One-sample Hodges-Lehmann estimator*

Description

`hodges_lehmann` calculates the one-sample Hodges-Lehmann estimator of a sample.

Usage

```
hodges_lehmann(x, na.rm = FALSE)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

Details

The one-sample Hodges-Lehmann estimator for a sample of size n is defined as

$$med(\frac{X_i + X_j}{2}, 1 \leq i < j \leq m).$$

Value

The one-sample Hodges-Lehmann estimator.

References

Hodges JL, Lehmann EL (1963). "Estimates of location based on rank tests." *The Annals of Mathematical Statistics*, **34**(2), 598–611. doi:10.1214/aoms/1177704172.

Examples

```
# Generate random sample
set.seed(108)
x <- rnorm(10)

# Compute one-sample Hodges-Lehmann estimator
hodges_lehmann(x)
```

`hodges_lehmann_2sample`

Two-sample Hodges-Lehmann estimator

Description

`hodges_lehmann_2sample` calculates the two-sample Hodges-Lehmann estimator for the location difference of two samples x and y.

Usage

```
hodges_lehmann_2sample(x, y, na.rm = FALSE)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`y`	a (non-empty) numeric vector of data values.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

Details

The two-sample Hodges-Lehmann estimator for two samples `x` and `y` of sizes `m` and `n` is defined as

$$med(|x_i - y_j|, 1 \leq i \leq m, 1 \leq j \leq n).$$

Value

The two-sample Hodges-Lehmann estimator.

References

Hodges JL, Lehmann EL (1963). "Estimates of location based on rank tests." *The Annals of Mathematical Statistics*, **34**(2), 598–611. doi:10.1214/aoms/1177704172.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(10); y <- rnorm(10)

# Compute two-sample Hodges-Lehmann estimator
hodges_lehmann_2sample(x, y)
```

`med_test` *Two-sample location tests based on the sample median*

Description

`med_test` performs a two-sample location test based on the difference of the sample medians for both samples.

Usage

```
med_test(
  x,
  y,
  alternative = c("two.sided", "greater", "less"),
  delta = ifelse(scale.test, 1, 0),
  method = c("asymptotic", "permutation", "randomization"),
  scale = c("S3", "S4"),
  n.rep = 10000,
  na.rm = FALSE,
  scale.test = FALSE,
  wobble = FALSE,
  wobble.seed = NULL
)
```

Arguments

x	a (non-empty) numeric vector of data values.
y	a (non-empty) numeric vector of data values.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater", or "less".
delta	a numeric value indicating the true difference in the location or scale parameter, depending on whether the test should be performed for a difference in location or in scale. The default is `delta = 0` for a location test and `delta = 1` for a scale test. In case of `scale.test = TRUE`, `delta` represents the ratio of the squared scale parameters.
method	a character string specifying how the p-value is computed with possible values `"asymptotic"` for an asymptotic test based on a normal approximation, `"permutation"` for a permutation test, and `"randomization"` for a randomization test. The permutation test uses all splits of the joint sample into two samples of sizes `m` and `n`, while the randomization test draws `n.rep` random splits with replacement. The values `m` and `n` denote the sample sizes. If not specified explicitly, defaults to `"permutation"` if `m < 30`, `n < 30` and `n.rep >= choose(m + n, m)`, `"randomization"` if `m < 30`, `n < 30` and `n.rep < choose(m + n, m)`, and `"asymptotic"` if `m >= 30` and `n >= 30`.
scale	a character string specifying the scale estimator used for standardization of the test statistic, must be one of `"S3"` and `"S4"`. The default is `"S3"`. Ignored if `method = "asymptotic"`; see details for the definition of the scale estimators.
n.rep	an integer value specifying the number of random splits used to calculate the randomization distribution if `method = "randomization"`. This argument is ignored if `method = "permutation"` or `method = "asymptotic"`. The default is `n.rep = 10000`.
na.rm	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.
scale.test	a logical value to specify if the samples should be compared for a difference in scale. The default is `scale.test = FALSE`.
wobble	a logical value indicating whether the sample should be checked for duplicated values that can cause the scale estimate to be zero. If such values are present, uniform noise is added to the sample, see `wobble`. Only necessary for the permutation and randomization version of the test. The default is `wobble = FALSE`.
wobble.seed	an integer value used as a seed for the random number generation in case of `wobble = TRUE` or when `scale.test = TRUE` with one of the vectors `x` and `y` containing zeros. When no seed is specified, it is chosen randomly and printed in a message. The argument is ignored if `scale.test = FALSE` and/or `wobble = FALSE`.

Details

The test statistic for this test is based on the difference of the sample medians of `x` and `y`. Three versions of the test are implemented: randomization, permutation, and asymptotic.

The test statistic for the permutation and randomization version of the test is standardized using a robust scale estimator, see (Fried and Dehling 2011).

With `scale = "S3"`, the scale is estimated by

$$S = 2 * (|x_1 - med(x)|, ..., |x_m - med(x)|, |y_1 - med(y)|, ..., |y_n - med(y)|),$$

whereas `scale = "S4"` uses

$$S = (med(|x_1 - med(x)|, ..., |x_m - med(x)|) + med(|y_1 - med(y)|, ..., |y_n - med(y)|).$$

When computing the randomization distribution based on randomly drawn splits with replacement, the function `permp` (Phipson and Smyth 2010) is used to calculate the p-value. For the asymptotic test, a transformed version of the difference of the sample medians, which asymptotically follows a normal distribution, is used. For more details on the asymptotic test, see Fried and Dehling (2011).

For `scale.test = TRUE`, the test compares the two samples for a difference in scale. This is achieved by log-transforming the original squared observations, i.e. `x` is replaced by `log(x^2)` and `y` by `log(y^2)`. A potential scale difference then appears as a location difference between the transformed samples, see Fried (2012). Note that the samples need to have equal locations. The sample should not contain zeros to prevent problems with the necessary log-transformation. If it contains zeros, uniform noise is added to all variables in order to remove zeros and a message is printed.

If the sample has been modified (either because of zeros for `scale.test = TRUE`, or `wobble = TRUE`), the modified samples can be retrieved using

`set.seed(wobble.seed); wobble(x, y)`

Both samples need to contain at least 5 non-missing values.

Value

A named list with class "`htest`" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value for the test.
`estimate`	the sample medians of `x` and `y` (if `scale.test = FALSE`) or of `log(x^2)` and `log(y^2)` (if `scale.test = TRUE`).
`null.value`	the specified hypothesized value of the mean difference/squared scale ratio.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating how the p-value was computed.
`data.name`	a character string giving the names of the data.

References

Phipson B, Smyth GK (2010). "Permutation p-values should never be zero: Calculating exact p-values when permutations are randomly drawn." *Statistical Applications in Genetics and Molecular Biology*, **9**(1), Article 39. doi:10.2202/15446115.1585.

Fried R, Dehling H (2011). "Robust nonparametric tests for the two-sample location problem." *Statistical Methods & Applications*, **20**(4), 409–422. doi:10.1007/s1026001101641.

Fried R (2012). "On the online estimation of piecewise constant volatilities." *Computational Statistics & Data Analysis*, **56**(11), 3080–3090. doi:10.1016/j.csda.2011.02.012.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Asymptotic MED test
med_test(x, y, method = "asymptotic", scale = "S3")

## Not run:
# MED2 test using randomization principle by drawing 1000 random permutations
# with replacement

med_test(x, y, method = "randomization", n.rep = 1000, scale = "S4")

## End(Not run)
```

---

m_est *M-estimator of location*

Description

`m_est` calculates an M-estimate of location and its variance for different psi functions.

Usage

```
m_est(
  x,
  psi,
  k = robustbase::.Mpsi.tuning.default(psi),
  tol = 1e-06,
  max.it = 15,
  na.rm = FALSE
)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`psi`	kernel used for optimization. Must be one of `"bisquare"`, `"hampel"` and `"huber"`. The default is `"huber"`.
`k`	tuning parameter(s) for the respective kernel function, defaults to parameters implemented in `.Mpsi.tuning.default(psi)` in the package robustbase.

`tol`	tolerance for convergence. The default is 1e-06.
`max.it`	the maximum number of iterations. The default is 15.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

Details

To compute the M-estimate, the iterative algorithm described in Maronna et al. (2019) is used. The variance is estimated as in Huber (1981).

If `max.it` contains decimal places, it is truncated to an integer value.

Value

A named list containing the components:

`est`	estimated mean.
`var`	estimated variance.

References

Maronna RA, Martin DR, Yohai VJ, Salibián-Barrera M (2019). *Robust Statistics: Theory and Methods (with R)*, Wiley Series in Probability and Statistics, Second edition edition. Wiley. doi:10.1002/9781119214656.

Huber PJ (1981). *Robust Statistics*. Wiley, New York. doi:10.1002/0471725250.

Examples

```
# Generate random sample
set.seed(108)
x <- rnorm(10)

# Computer Huber's M-estimate
m_est(x, psi = "huber")
```

`m_test` *Two sample location test based on M-estimators*

Description

`m_test` performs a two-sample location test based on an M-estimator.

Usage

```
m_test(
  x,
  y,
  alternative = c("two.sided", "greater", "less"),
  delta = ifelse(scale.test, 1, 0),
  method = c("asymptotic", "permutation", "randomization"),
  psi = c("huber", "hampel", "bisquare"),
  k = robustbase::.Mpsi.tuning.default(psi),
  n.rep = 10000,
  na.rm = FALSE,
  scale.test = FALSE,
  wobble.seed = NULL,
  ...
)
```

Arguments

x	a (non-empty) numeric vector of data values.
y	a (non-empty) numeric vector of data values.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater", or "less".
delta	a numeric value indicating the true difference in the location or scale parameter, depending on whether the test should be performed for a difference in location or in scale. The default is `delta = 0` for a location test and `delta = 1` for a scale test. In case of `scale.test = TRUE`, `delta` represents the ratio of the squared scale parameters.
method	a character string specifying how the p-value is computed with possible values `"asymptotic"` for an asymptotic test based on a normal approximation, `"permutation"` for a permutation test, and `"randomization"` for a randomization test. The permutation test uses all splits of the joint sample into two samples of sizes `m` and `n`, while the randomization test draws `n.rep` random splits with replacement. The values `m` and `n` denote the sample sizes. If not specified explicitly, defaults to `"permutation"` if `m < 30`, `n < 30` and `n.rep >= choose(m + n, m)`, `"randomization"` if `m < 30`, `n < 30` and `n.rep < choose(m + n, m)`, and `"asymptotic"` if `m >= 30` and `n >= 30`.
psi	kernel used for optimization. Must be one of `"bisquare"`, `"hampel"` and `"huber"`. The default is `"huber"`.
k	tuning parameter(s) for the respective kernel function, defaults to parameters implemented in `.Mpsi.tuning.default(psi)` in the package robustbase.
n.rep	an integer value specifying the number of random splits used to calculate the randomization distribution if `method = "randomization"`. This argument is ignored if `method = "permutation"` or `method = "asymptotic"`. The default is `n.rep = 10000`.
na.rm	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

`scale.test`	a logical value to specify if the samples should be compared for a difference in scale. The default is `scale.test = FALSE`.
`wobble.seed`	an integer value used as a seed for the random number generation in case that `scale.test = TRUE` and one of the vectors `x` and `y` contains zeros. When no seed is specified, it is chosen randomly and printed in a message. The argument is ignored if `scale.test = FALSE`.
`...`	additional arguments `c1` and `c2` that can be passed to the function `scaleTau2()`, which is used internally for estimating the within-sample dispersion, in order to account for non-normal distributions; see Maronna and Zamar (2002).

Details

The test statistic for this test is based on the difference of the M-estimates of location of `x` and `y`, see `m_est`.

Three different psi-functions can be used: `huber`, `hampel`, and `bisquare`. The corresponding tuning parameter(s) can be set by the argument `k` of the function.

The estimate for the location difference is scaled by a pooled estimate for the standard deviation. This estimate is based on the tau-estimate of scale and is computed with the default parameter settings of the function `scaleTau2`. These can be changed if by setting `c1` and `c2`.

More details on the construction of the test statistic are given in the vignettes `vignette("robnptests")` and `vignette("m_tests")`.

Three versions of the test are implemented: randomization, permutation, and asymptotic.

The randomization distribution is based on randomly drawn splits with replacement. The function `permp` (Phipson and Smyth 2010) is used to calculate the p-value. The psi-function for the the M-estimate is computed with the implementations in the package robustbase.

For the asymptotic test, the distribution of the test statistic is approximated by a standard normal distribution. However, this is only justified under the normality assumption. When the observations do not come from a normal distribution, the tests might not keep the desired significance level. Simulations indicate that the level is kept under symmetric distributions if the variance exists. Under skewed distributions, it tends to be anti-conservative, see the vignette `vignette("m_tests")`. The test statistic can be corrected by a factor which has to be determined individually for a specific distribution in such cases.

For `scale.test = TRUE`, the test compares the two samples for a difference in scale. This is achieved by log-transforming the original squared observations, i.e. `x` is replaced by `log(x^2)` and `y` by `log(y^2)`. A potential scale difference then appears as a location difference between the transformed samples, see Fried (2012). Note that the samples need to have equal locations. The sample should not contain zeros to prevent problems with the necessary log-transformation. If it contains zeros, uniform noise is added to all variables in order to remove zeros and a message is printed.

If the sample has been modified because of zeros when `scale.test = TRUE`, the modified samples can be retrieved using

```
set.seed(wobble.seed); wobble(x, y)
```

Both samples need to contain at least 5 non-missing values.

Value

A named list with class "htest" containing the following components:

statistic	the value of the test statistic.
parameter	the degrees of freedom for the test statistic.
p.value	the p-value for the test.
estimate	the M-estimates of x and y (if scale.test = FALSE) or of log(x^2) and log(y^2) (if scale.test = TRUE).
null.value	the specified hypothesized value of the mean difference/squared scale ratio.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating how the p-value was computed.
data.name	a character string giving the names of the data.

References

Fried R (2012). "On the online estimation of piecewise constant volatilities." *Computational Statistics & Data Analysis*, **56**(11), 3080–3090. doi:10.1016/j.csda.2011.02.012.

Maronna RA, Zamar RH (2002). "Robust estimates of location and dispersion of high-dimensional datasets." *Technometrics*, **44**(4), 307–317. doi:10.1198/004017002188618509.

Phipson B, Smyth GK (2010). "Permutation p-values should never be zero: Calculating exact p-values when permutations are randomly drawn." *Statistical Applications in Genetics and Molecular Biology*, **9**(1), Article 39. doi:10.2202/15446115.1585.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Asymptotic test based on Huber M-estimator
m_test(x, y, method = "asymptotic", psi = "huber")

## Not run:
# Randomization test based on Hampel M-estimator with 1000 random permutations
# drawn with replacement

m_test(x, y, method = "randomization", n.rep = 1000, psi = "hampel")

## End(Not run)
```

`m_test_statistic` *Test statistics for the M-tests*

Description

`m_test_statistic` calculates the test statistics for tests based on M-estimators.

Usage

```
m_test_statistic(x, y, psi, k = robustbase::.Mpsi.tuning.default(psi), ...)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`y`	a (non-empty) numeric vector of data values.
`psi`	kernel used for optimization. Must be one of `"bisquare"`, `"hampel"` and `"huber"`. The default is `"huber"`.
`k`	tuning parameter(s) for the respective kernel function, defaults to parameters implemented in `.Mpsi.tuning.default(psi)` in the package robustbase.
`...`	additional arguments `c1` and `c2` that can be passed to the function `scaleTau2()`, which is used internally for estimating the within-sample dispersion, in order to account for non-normal distributions; see Maronna and Zamar (2002).

Details

For details on how the test statistic is constructed, we refer to the vignette `vignette("m_tests")`

Value

A named list containing the following components:

`statistic`	standardized test statistic.
`estimates`	M-estimates of location for both `x` and `y`.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Compute Huber-M-statistic
m_test_statistic(x, y, psi = "huber")
```

`rob_perm_statistic` *Robust test statistics based on robust location estimators*

Description

`rob_perm_statistic` calculates test statistics for robust permutation/randomization tests based on the sample median, the one-sample Hodges-Lehmann estimator, or the two-sample Hodges-Lehmann estimator.

Usage

```
rob_perm_statistic(
  x,
  y,
  type = c("HL11", "HL12", "HL21", "HL22", "MED1", "MED2"),
  na.rm = FALSE
)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`y`	a (non-empty) numeric vector of data values.
`type`	a character string specifying the desired test statistic. It must be one of `"HL11"` (default), `"HL12"`, `"HL21"`, `"HL22"`, `"MED1"`, and `"MED2"`, where `"HL1"`, `"HL2"` and `"MED"` specify the location estimator and the numbers `1` and `2` the scale estimator, see the vignette `vignette("robnptests")` for more information.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

Details

The test statistics returned by `rob_perm_statistic` are of the form

$$D_i/S_j$$

where the D_i, i = 1,...,3, are different estimators of location and the S_j, j = 1,...,4, are estimates for the mutual sample scale. See Fried and Dehling (2011) or the vignette `vignette("robnptests")` for details.

Value

A named list containing the following components:

`statistic`	the selected test statistic.
`estimates`	estimate of location for each sample if available.

References

Fried R, Dehling H (2011). "Robust nonparametric tests for the two-sample location problem." *Statistical Methods & Applications*, **20**(4), 409–422. doi:10.1007/s1026001101641.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Compute HL21-statistic
rob_perm_statistic(x, y, type = "HL21")
```

`rob_scale` *Robust scale estimators based on median absolute deviation*

Description

`rob_scale` calculates an estimator for the within-sample dispersion based on two samples.

Usage

```
rob_scale(
  x,
  y,
  type = c("S1", "S2", "S3", "S4"),
  na.rm = FALSE,
  check.for.zero = FALSE
)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`y`	a (non-empty) numeric vector of data values.
`type`	character that specifies the estimator for the variance, can be `"S1"`, `"S2"`, `"S3"` and `"S4"`; see details for description of the scale estimators.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.
`check.for.zero`	logical value indicating a warning should be triggered if the scale estimate is zero. The default is `FALSE`.

Details

For definitions of the scale estimators, see Fried and Dehling (2011).

If `check.for.zero = TRUE`, an error is thrown when the scale estimate is zero. This argument is only included because the function is used in `rob_perm_statistic` to compute values of robust test statistics where the scale estimate is used for standardization. A scale estimate of zero leads to a non-existing test statistic, so that the corresponding test cannot be performed.

Value

An estimate of the pooled variance of the two samples.

References

Fried R, Dehling H (2011). "Robust nonparametric tests for the two-sample location problem." *Statistical Methods & Applications*, **20**(4), 409–422. doi:10.1007/s1026001101641.

`trimmed_t` *Test statistic for the two-sample trimmed t-test (Yuen's t-test)*

Description

`trimmed_t` calculates the test statistic for the two-sample trimmed t-test.

Usage

```
trimmed_t(x, y, gamma = 0.2, na.rm = FALSE)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`y`	a (non-empty) numeric vector of data values.
`gamma`	a numeric value in [0, 0.5] specifying the fraction of observations to be trimmed from each end of the sample before calculating the mean. The default value is 0.2.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

Value

A named list containing the following components:

`statistic`	the value of the test statistic.
`estimates`	the trimmed means for both samples.
`df`	the degrees of freedom for the test statistic.

References

Yuen KK, Dixon WT (1973). "The approximate behaviour and performance of the two-sample trimmed t." *Biometrika*, **60**(2), 369–374. doi:10.2307/2334550.

Yuen KK (1974). "The two-sample trimmed t for unequal population variances." *Biometrika*, **61**(1), 165–170. doi:10.2307/2334299.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Compute trimmed t-statistic
trimmed_t(x, y, gamma = 0.2)
```

`trimmed_test` *Two-sample trimmed t-test (Yuen's t-Test)*

Description

`trimmed_test` performs the two-sample trimmed t-test.

Usage

```
trimmed_test(
  x,
  y,
  gamma = 0.2,
  alternative = c("two.sided", "less", "greater"),
  method = c("asymptotic", "permutation", "randomization"),
  delta = ifelse(scale.test, 1, 0),
  n.rep = 1000,
  na.rm = FALSE,
  scale.test = FALSE,
  wobble.seed = NULL
)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`y`	a (non-empty) numeric vector of data values.
`gamma`	a numeric value in [0, 0.5] specifying the fraction of observations to be trimmed from each end of the sample before calculating the mean. The default value is 0.2.

alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater", or "less".
method	a character string specifying how the p-value is computed with possible values "asymptotic" for an asymptotic test based on a normal approximation, "permutation" for a permutation test, and "randomization" for a randomization test. The permutation test uses all splits of the joint sample into two samples of sizes m and n, while the randomization test draws n.rep random splits with replacement. The values m and n denote the sample sizes. If not specified explicitly, defaults to "permutation" if m < 30, n < 30 and n.rep >= choose(m + n, m), "randomization" if m < 30, n < 30 and n.rep < choose(m + n, m), and "asymptotic" if m >= 30 and n >= 30.
delta	a numeric value indicating the true difference in the location or scale parameter, depending on whether the test should be performed for a difference in location or in scale. The default is delta = 0 for a location test and delta = 1 for a scale test. In case of scale.test = TRUE, delta represents the ratio of the squared scale parameters.
n.rep	an integer value specifying the number of random splits used to calculate the randomization distribution if method = "randomization". This argument is ignored if method = "permutation" or method = "asymptotic". The default is n.rep = 10000.
na.rm	a logical value indicating whether NA values in x and y should be stripped before the computation proceeds. The default is na.rm = FALSE.
scale.test	a logical value to specify if the samples should be compared for a difference in scale. The default is scale.test = FALSE.
wobble.seed	an integer value used as a seed for the random number generation in case of wobble = TRUE or when scale.test = TRUE with one of the vectors x and y containing zeros. When no seed is specified, it is chosen randomly and printed in a message. The argument is ignored if scale.test = FALSE and/or wobble = FALSE.

Details

The function performs Yuen's t-test based on the trimmed mean and winsorized variance (Yuen and Dixon 1973). The amount of trimming/winsorization is set in gamma and defaults to 0.2, i.e. 20% of the values are removed/replaced. In addition to the asymptotic distribution a permutation and a randomization version of the test are implemented.

When computing a randomization distribution based on randomly drawn splits with replacement, the function permp (Phipson and Smyth 2010) is used to calculate the p-value.

For scale.test = TRUE, the test compares the two samples for a difference in scale. This is achieved by log-transforming the original squared observations, i.e. x is replaced by log(x^2) and y by log(y^2). A potential scale difference then appears as a location difference between the transformed samples, see Fried (2012). Note that the samples need to have equal locations. The sample should not contain zeros to prevent problems with the necessary log-transformation. If it contains zeros, uniform noise is added to all variables in order to remove zeros and a message is printed.

If the sample has been modified because of zeros when scale.test = TRUE, the modified samples can be retrieved using

`set.seed(wobble.seed); wobble(x, y)`

Both samples need to contain at least 5 non-missing values.

Value

A named list with class "`htest`" containing the following components:

`statistic`	the value of the test statistic.
`parameter`	the degrees of freedom for the test statistic.
`p.value`	the p-value for the test.
`estimate`	the trimmed means of `x` and `y` (if `scale.test = FALSE`) or of `log(x^2)` and `log(y^2)` (if `scale.test = TRUE`).
`null.value`	the specified hypothesized value of the mean difference/squared scale ratio.
`alternative`	a character string describing the alternative hypothesis.
`method`	a character string indicating how the p-value was computed.
`data.name`	a character string giving the names of the data.

References

Yuen KK, Dixon WT (1973). "The approximate behaviour and performance of the two-sample trimmed t." *Biometrika*, **60**(2), 369–374. doi:10.2307/2334550.

Yuen KK (1974). "The two-sample trimmed t for unequal population variances." *Biometrika*, **61**(1), 165–170. doi:10.2307/2334299.

Fried R (2012). "On the online estimation of piecewise constant volatilities." *Computational Statistics & Data Analysis*, **56**(11), 3080–3090. doi:10.1016/j.csda.2011.02.012.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(20); y <- rnorm(20)

# Trimmed t-test
trimmed_test(x, y, gamma = 0.1)
```

`trim_mean` *Trimmed mean*

Description

`trim_mean` calculates a trimmed mean of a sample.

Usage

```
trim_mean(x, gamma = 0.2, na.rm = FALSE)
```

Arguments

x	a (non-empty) numeric vector of data values.
gamma	a numeric value in [0, 0.5] specifying the fraction of observations to be trimmed from each end of the sample before calculating the mean. The default value is 0.2.
na.rm	a logical value indicating whether NA values in x and y should be stripped before the computation proceeds. The default is na.rm = FALSE.

Details

This is a wrapper function for the function mean.

Value

The trimmed mean.

Examples

```
# Generate random sample
set.seed(108)
x <- rnorm(10)

# Compute 20% trimmed mean
trim_mean(x, gamma = 0.2)
```

---

win_mean *Winsorized mean*

Description

win_mean calculates the winsorized mean of a sample.

Usage

```
win_mean(x, gamma = 0.2, na.rm = FALSE)
```

Arguments

x	a (non-empty) numeric vector of data values.
gamma	a numeric value in [0, 0.5] specifying the fraction of observations to be replaced at each end of the sample before calculating the mean. The default value is 0.2.
na.rm	a logical value indicating whether NA values in x and y should be stripped before the computation proceeds. The default is na.rm = FALSE.

Value

The winsorized mean.

Examples

```
# Generate random samples
set.seed(108)
x <- rnorm(10)

# Compute 20% winsorized mean
win_mean(x, gamma = 0.2)
```

`win_var` *Winsorized variance*

Description

`win_var` calculates the winsorized variance of a sample.

Usage

```
win_var(x, gamma = 0, na.rm = FALSE)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`gamma`	a numeric value in [0, 0.5] specifying the fraction of observations to be replaced at each end of the sample before calculating the mean. The default value is 0.2.
`na.rm`	a logical value indicating whether NA values in `x` and `y` should be stripped before the computation proceeds. The default is `na.rm = FALSE`.

Value

A named list containing the following items:

`var`	winsorized variance.
`h`	degrees of freedom used for tests based on trimmed means and the winsorized variance.

Examples

```
# Generate random sample
set.seed(108)
x <- rnorm(10)

# Compute 20% winsorized variance
win_var(x, gamma = 0.2)
```

wobble *Add random noise to remove ties*

Description

wobble adds noise from a continuous uniform distribution to the observations to remove ties.

Usage

```
wobble(x, y, check = TRUE)
```

Arguments

x	a (non-empty) numeric vector of data values.
y	a (non-empty) numeric vector of data values.
check	a logical value indicating whether the samples should be checked for bindings prior to adding uniform noise or not, defaults to TRUE.

Details

If check = TRUE the function checks whether all values in the two numeric input vectors are distinct. If so, it returns the original values, otherwise the ties are removed by adding noise from a continuous uniform distribution to all observations. If check = FALSE, it simply determines the number of digits and adds uniform noise.

More precisely, we determine the minimum number of digits d_min in the sample and then add random numbers from the U[-0.5 10^(-d_min), 0.5 10^(-d_min)] distribution to each of the observations.

Value

A named list of length two containing the modified input samples x and y.

References

Fried R, Gather U (2007). "On rank tests for shift detection in time series." *Computational Statistics & Data Analysis*, **52**(1), 221–233. doi:10.1016/j.csda.2006.12.017.

Examples

```
x <- rnorm(20); y <- rnorm(20); x <- round(x)
wobble(x, y)
```

Index