Package 'fasano.franceschini.test'

May 29, 2022

Type Package

Title Fasano-Franceschini Test: A Multidimensional Kolmogorov-Smirnov
Two-Sample Test

Version 2.0.0

Description An implementation of the two-sample multidimensional
Kolmogorov-Smirnov test described by Fasano and Franceschini (1987)
<doi:10.1093/mnras/225.1.155>. This test evaluates the null hypothesis
that two i.i.d. random samples were drawn from the same underlying
probability distribution. The data can be of any dimension, and can be
of any type (continuous, discrete, or mixed).

License MIT + file LICENSE

URL https://github.com/nesscoder/fasano.franceschini.test

BugReports https://github.com/nesscoder/fasano.franceschini.test/issues

Depends R (>= 3.0.2)

Imports Rcpp (>= 1.0.0), RcppParallel (>= 5.0.1), stats

Suggests testthat (>= 3.0.0)

LinkingTo Rcpp (>= 1.0.0), RcppParallel (>= 5.0.1)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.0

SystemRequirements GNU make

NeedsCompilation yes

Author Connor Puritz [aut, cre] (<https://orcid.org/0000-0001-7602-0444>),
Elan Ness-Cohn [aut] (<https://orcid.org/0000-0002-3935-6667>),
Rosemary Braun [ctb, ths] (<https://orcid.org/0000-0001-9668-9866>),
Luca Weihs [cph] (Copyright holder and author of 'RangeTree' class.)

Maintainer Connor Puritz <connorpuritz2025@u.northwestern.edu>

Repository CRAN

Date/Publication 2022-05-29 05:30:09 UTC

R **topics documented:**

---

fasano.franceschini.test

Fasano-Franceschini Test

Description

Performs a two-sample multidimensional Kolmogorov-Smirnov test as described by Fasano and Franceschini (1987). This test evaluates the null hypothesis that two i.i.d. random samples were drawn from the same underlying probability distribution. The data can be of any dimension, and can be of any type (continuous, discrete, or mixed).

Usage

```
fasano.franceschini.test(
  S1,
  S2,
  nPermute = 100,
  threads = 1,
  cores,
  seed = NULL,
  p.conf.level = 0.95,
  verbose = TRUE,
  method = c("r", "b")
)
```

Arguments

S1	matrix or data.frame.
S2	matrix or data.frame.
nPermute	a nonnegative integer setting the number of permuted samples to generate when estimating the permutation test p-value. Default is 100. If set to 0, no p-value is estimated.
threads	a positive integer or "auto" setting the number of threads used for performing the permutation test. If set to "auto", the number of threads is determined by RcppParallel::defaultNumThreads(). Default is 1.
cores	allowed for backwards compatibility. threads is now the preferred argument name.
seed	optional integer to seed the PRNG used for the permutation test. Default is NULL. Only available for serial version (threads = 1).
p.conf.level	confidence level for the confidence interval of the permutation test p-value.

verbose	a `boolean` indicating whether to display a progress bar. Default is `TRUE`. Only available for serial version (`threads = 1`).
method	a `character` indicating which method to use to compute the test statistic. Must be either `'r'` for the range-tree method (default), or `'b'` for the brute force method. Both return the same results, but may vary in computation speed. See the Details section for more information.

Details

The test statistic can be computed using two different methods. Both methods return identical results, but vary in computation time.

- Range tree method (`method = 'r'`): This method has a time complexity of $O(n*log(n)^{(d-1)})$, where n is the size of the larger sample and d is the dimension of the data.

- Brute force method (`method = 'b'`): This method has a time complexity of $O(n^2)$.

When $d = 2$ (regardless of n), or $d > 2$ and n is large, the range tree method tends to outperform the brute force method. When $d > 2$ and n is small, the brute force method tends to outperform the range tree method.

The p-value for the test is computed empirically using a permutation test. As it is almost always infeasible to compute the exact permutation test p-value, a Monte Carlo approximation is made instead. This estimate is a binomially distributed random variable, and thus a confidence interval can be computed. The confidence interval is obtained using the procedure given in Clopper and Pearson (1934).

Value

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

statistic	the value of the test statistic Z.
estimate	the value of the difference statistics D1 and D2.
p.value	the permutation test p-value.
conf.int	a binomial confidence interval for the p-value.
method	a character string indicating what type of test was performed.
data.name	a character string giving the names of the data.

References

- Fasano, G. & Franceschini, A. (1987). A multidimensional version of the Kolmogorov-Smirnov test. *Monthly Notices of the Royal Astronomical Society*, 225:155-170. doi:10.1093/mnras/225.1.155.

- Clopper, C. J. & Pearson, E. S. (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26, 404–413. doi:10.2307/2331986.

Examples

```
set.seed(0)

# create 2-D samples using data frames
S1 <- data.frame(x = rnorm(n = 50, mean = 1, sd = 2),
                 y = rnorm(n = 50, mean = 3, sd = 1))
S2 <- data.frame(x = rnorm(n = 150, mean = 1, sd = 2),
                 y = rnorm(n = 150, mean = 3, sd = 1))

# perform test (serial version)
fasano.franceschini.test(S1, S2)

# perform test with more permutations
fasano.franceschini.test(S1, S2, nPermute = 200)

# set seed for reproducible p-value
fasano.franceschini.test(S1, S2, seed = 0)

# change confidence level for p-value confidence interval
fasano.franceschini.test(S1, S2, p.conf.level = 0.99)

# perform test (parallel version, 2 threads)
## Not run:
fasano.franceschini.test(S1, S2, threads = 2)

## End(Not run)


# create 3-D mixed samples using matrices
S1 <- cbind(rgamma(n = 43, shape = 2),
            rpois(n = 43, lambda = 5),
            rpois(n = 43, lambda = 3.5))
S2 <- cbind(rgamma(n = 72, shape = 2),
            rpois(n = 72, lambda = 5),
            rpois(n = 72, lambda = 5))

# perform test using range tree method
fasano.franceschini.test(S1, S2, seed = 0, method = 'r')

# perform test using brute force method
fasano.franceschini.test(S1, S2, seed = 0, method = 'b')
```

Index