Package ‘RPDTest’

July 21, 2025

Title A New Type of Test Statistic and Method for Multinomial Goodness-of-Fit Test

Version 0.0.2

Description Performs multinomial goodness-of-fit test on multinomially distributed data using the Randomized phi-divergence test statistics. Details of this kind of statistics can be found at Nikita Puchkin, Vladimir Ulyanov (2023) <doi:10.1214/22-AIHP1299>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Imports doParallel, foreach, parallel, stats

NeedsCompilation no

Author Renkang Liu [aut, cre]

Maintainer Renkang Liu <eG0im@outlook.com>

Repository CRAN

Date/Publication 2024-05-20 12:50:02 UTC

Contents

`pVals` *Randomized phi-divergence test: simulated p-value part*

Description

This is one of the auxiliary functions used to execute the rpdTest function. This function can be used to calculate p-values based on Monte Carlo simulation. Users generally do not need to call this function except for testing purposes. For more detailed description one can find inrpdTest.

Usage

```
pVals(x, p, lambda = 1, B = 200, z = 40, rs = 1250, n.cores, nDim, r)
```

Arguments

`x` the obtained multinomial distribution data.Same data structure as the data parameter in rpdTest.

`p` the probability vector in the null hypothesis. It is necessary to ensure beforehand that the vectors are valid.

`lambda` a control parameter of the statistic calculation, adjusting it will significantly change the final obtained statistic.

`B` an integer specifying the number of simulation data on the expected null distribution (p) of the Monte Carlo simulation.

`z` an integer specifying the number by which to divide the observation data group in a Monte Carlo simulation.

`rs` an integer that adjusts the number of statistics calculated in simulation.

`n.cores` an integer used to specify the number of cores used to perform parallel operations. The default is to use the maximum number of cores available to the computer minus one.

`nDim` an integer indicating the dimension of the uniformly distributed vectors generated during the computation of the statistic. It is equal to the number of experiments for the multinomial distribution.

`r` an integer indicating the dimension of the data parameter. It is equal to the number of possible outcomes of the multinomial distribution.

Value

an numeric value indicating simulated p-value.

Examples

```
d <- c(20,40)
#The next line is equivalent to rpdTest(d,sim.pValue = TRUE,n.cores = 2)$p.value
#It usually takes 1-2 minutes to perform this calculation process

pVals(d, c(1/2,1/2), B = 200, z = 40, rs = 1250, n.cores = 2, nDim = sum(d), r = length(d))
```

`rpdStat` *Randomized phi-divergence test: statistic part*

Description

This is one of the auxiliary functions used to execute the rpdTest function. This function calculates the statistic for a single Randomized phi-divergence test. Users generally do not need to call this function except for testing purposes.

Usage

```
rpdStat(data, probability, lambda = 1, nDim, r, random.state = NULL)
```

Arguments

`data`	the same data structure that provided in rpdTest.
`probability`	the same numeric vector that provided in rpdTest.
`lambda`	the same parameter that provided in rpdTest.
`nDim`	an integer indicating the dimension of the uniformly distributed vectors generated during the computation of the statistic. It is equal to the number of experiments for the multinomial distribution.
`r`	an integer indicating the dimension of the data parameter. It is equal to the number of possible outcomes of the multinomial distribution.
`random.state`	a numeric that controls the randomness of the samples used when generating uniformly distributed random vector on the n-sphere.

Value

a numeric value that reflects the statistic obtained after an execution of rpdTest at that time.

Examples

```
d <- c(20,40)
#The next line is equivalent to rpdTest(d)$statistic

rpdStat(d, c(1/2,1/2), nDim = sum(d), r = length(d))
```

`rpdTest` *Randomized phi-divergence test*

Description

The most important part of the package: a function for performing hypothesis testing —- An analogue of Chi-square Goodness-of-Fit Test. Accept a vector, matrix or a data.frame as observed data. Then obtain a specific Randomized phi-divergence statistic, which is computed based on a uniformly distributed random vector on the n-sphere. This random vector is uniquely generated at runtime. However, a p-values in Monte Carlo simulation is available as an option. It executes in parallel way, comparing the empirical distribution function. In specific, it simulates data under the null hypothesis and compares it to the observed data. It generates B datasets based on the expected null distribution (p) and the observed control data (v0). For each simulated dataset and the observed data and v0, rs statistics are computed using different random seeds. The Kolmogorov-Smirnov statistic is used to compare the distributions of the simulated and observed data and the simulated and control data. We get B K-S statistics in both observed data group and control data group. The function then calculates a p-value based on how often the within-group mean of the Kolmogorov-Smirnov statistic after dividing the observed data group into z groups is more extreme than the mean of the statistic observed for the control vector group. In the current version (0.0.2), this feature is still being debugged and improved, so this option is not enabled by default.

Usage

```
rpdTest(
  data,
  p = rep(1/length(data), length(data)),
  lambda = 1,
  sim.pValue = FALSE,
  B = 200,
  z = 40,
  rs = 1250,
  n.cores = NULL,
  random.state = NULL
)
```

Arguments

`data` a one-dimensional vector or matrix of this shape (data.frame) in which observation data for some multinomial distribution are stored.

`p` the probability vector in the null hypothesis. Will check the validity of this vector.

`lambda` a control parameter of the statistic calculation, adjusting it will significantly change the final obtained statistic.

`sim.pValue` a logical variable. It decides whether to compute p-values in Monte Carlo simulation.

`B` an integer specifying the number of simulation data on the expected null distribution (p) of the Monte Carlo simulation.

`z` an integer specifying the number by which to divide the observation data group in a Monte Carlo simulation.

`rs` an integer that adjusts the number of statistics calculated in simulation.

`n.cores` an integer used to specify the number of cores used to perform parallel operations. The default is to use the maximum number of cores available to the computer minus one.

`random.state` a numeric that controls the randomness of the samples used when generating uniformly distributed random vector on the n-sphere.

Value

standard list object with class "htest".

Examples

```
d <- rmultinom(1, 120, c(1/4,3/4))
#following will only obtain statistic
rpdTest(d)
#following will obtain sim.p.value either. You can also specify the number of
#cores to use. For example, two:
#It usually takes 1-2 minutes to perform this calculation process
```

```
rpdTest(d,sim.pValue = TRUE,n.cores = 2)
```

Index