Package 'ExactMultinom'

September 4, 2020

Type Package

Title Multinomial Goodness-of-Fit Tests

Version 0.1.2

Author Johannes Resin

Maintainer Johannes Resin <johannes.resin@h-its.org>

Description

Computes exact p-values for multinomial goodness-of-fit tests based on multiple test statistics, namely, Pearson's chi-square, the log-likelihood ratio and the probability mass statistic. Implements the algorithm detailed in Resin (2020) <arXiv:2008.12682>. Estimates based on the classical asymptotic chi-square approximation or Monte-Carlo simulation can also be computed.

License GPL (>= 2)

Imports Rcpp (>= 1.0.4), stats

LinkingTo Rcpp

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-09-04 21:42:10 UTC

R topics documented:

multinom.test 2
multinom_test_cpp 4

Index **5**

`multinom.test` *Multinomial Goodness-of-Fit Tests*

Description

Computes exact p-values for multinomial goodness-of-fit tests based on multiple test statistics, namely, Pearson's chi-square, the log-likelihood ratio and the probability mass statistic. Implements the algorithm detailed in Resin (2020). Estimates based on the classical asymptotic chi-square approximation or Monte-Carlo simulation can also be computed.

Usage

```
multinom.test(
  x,
  p,
  stat = "Prob",
  method = "exact",
  theta = 1e-04,
  timelimit = 10,
  N = 10000
)
```

Arguments

`x`	Vector of non-negative integers - number of times each outcome was observed.
`p`	A vector of positive numbers - the hypothesized probabilities for each outcome. Need not sum to 1, but only encode hypothesized proportions correctly.
`stat`	Test statistic to be used. Can be "Prob" for the probability mass, "Chisq" for Pearson's chi-square and "LLR" for the log-likelihood ratio.
`method`	Method used to compute the p-values. Can be "exact", "asymptotic" and "Monte-Carlo".
`theta`	Parameter used with the exact method. p-values less than theta will not be determined precisely. Values >= 10^-8 are recommended. Very small p-values (< 10^-8) may be falsified by rounding errors.
`timelimit`	Time limit in seconds, after which the exact method is interrupted to avoid long runtime.
`N`	Number of samples generated by the Monte-Carlo approach.

Details

The "exact" method implements the algorithm detailed in Resin (2020). The method improves on the full enumeration implemented in some other R packages. It should work well if the number of categories is small. To avoid long runtimes the exact computation is interrupted after `timelimit` seconds. However, it may take longer than the specified time limit for the actual interrupt to occur. Only p-values greater `theta` are determined precisely. For p-values less than `theta`, the algorithm will only determine that the p-value is smaller than `theta`.

The "asymptotic" method returns classical chi-square approximations. The asymptotic approximation to the probability mass statistic is detailed in Section 2 of Resin (2020).

The "Monte-Carlo" method returns estimates based on `N` random draws from the hypothesized distribution.

Value

A list with class "mgof" containing

`x`	As input by user.
`p`	As input by user.
`stat`	As input by user or default.
`method`	As input by user or default.
`theta`	As input by user or default.
`pvals_ex`	Exact p-values or NA. WARNING: Values less than theta are NOT exact p-values, but only imply that the actual p-value is less than that value.
`pvals_as`	Asymptotic approximation to p-values or NA.
`pvals_mc`	Monte-Carlo estimated p-values or NA.

The first p-value (e.g., `pvals_ex[1]`) is obtained from the probability mass, the second from Pearson's chi-square and the third from the log-likelihood ratio.

Note

Each method computes p-values for all three test statistics simultaneously.

References

Resin, J. (2020), A Simple Algorithm for Exact Multinomial Tests, *Preprint* https://arxiv.org/abs/2008.12682

Examples

```
# Test fairness of a die (that is, whether each side has the same probability)
p_fair = rep(1/6,6) # Hypothesized probabilities for each side
x = c(16,17,12,15,15,25) # Observed number of times each side appeared on 100 throws
# Exact multinomial test (using probability ordering by default):
multinom.test(x,p_fair)
# Exact multinomial test using log-likelihood ratio:
multinom.test(x,p_fair,stat = "LLR")
# Classical chi-square test (using asymptotics to estimate p-value and Pearson's chi-square):
multinom.test(x,p_fair,stat = "Chisq",method = "asymptotic")
# Using Monte-Carlo approach and probability ordering
multinom.test(x,p_fair,stat = "Prob",method = "Monte-Carlo")
```

`multinom_test_cpp` *C++ Function for Exact Multinomial Test*

Description

C++ function computing exact multinomial p-values. Does not perform any safety checks. Incorrect input may result in unwanted behavior. Use only through `multinom.test` with method = "exact" is recommended.

Usage

```
multinom_test_cpp(x, p, theta = 1e-04)
```

Arguments

`x`	Vector of non-negative integers - number of times each outcome was observed.
`p`	A vector of positive numbers - the hypothesized probabilities for each outcome. Need to sum to 1!
`theta`	Parameter - p-values less than theta will not be determined precisely.

Details

The outcomes should be ordered by the hypothesized probabilities from largest to smallest for optimal performance.

Value

Returns a vector containing three values which are the p-values computed from the probability mass, Pearson's chi-square and the log-likelihood ratio statistic. Values below the threshold theta are upper bounds only and not exact p-values!

See Also

`multinom.test`

Index

ExactMultinom *(*multinom.test*)*, 2

multinom.test, 2, *4*
multinom_test_cpp, 4