Package 'EMT'

January 20, 2025

Type Package

Title Exact Multinomial Test: Goodness-of-Fit Test for Discrete
Multivariate Data

Version 1.3.1

Date 2024-03-26

Author Uwe Menzel

Maintainer Uwe Menzel <uwemenzel@gmail.com>

Description Goodness-of-fit tests for discrete multivariate data. It is
tested if a given observation is likely to have occurred under
the assumption of an ab-initio model. Monte Carlo methods are provided to
make the package capable of solving high-dimensional problems.

License GPL

LazyLoad yes

Repository CRAN

Date/Publication 2024-03-26 16:10:02 UTC

NeedsCompilation no

Contents

EMT–package   *Exact Multinomial Test: Goodness-of-Fit Test for Discrete Multivariate Data*

Description

The package provides functions to carry out a Goodness-of-fit test for discrete multivariate data. It is tested if a given observation is likely to have occurred under the assumption of an ab-initio model. A p-value can be calculated using different distance measures between observed and expected frequencies. A Monte Carlo method is provided to make the package capable of solving high-dimensional problems. The main user functions are `multinomial.test` and `plotMultinom`.

Details

Package:	CCP
Type:	Package
Version:	1.3
Date:	2013-02-06
License:	GPL

Author(s)

Uwe Menzel

Maintainer: Uwe Menzel <uwemenzel@gmail.com>

EMT-internal   *Internal functions for the EMT package*

Description

Internal functions for the EMT package

Usage

```
ExactMultinomialTest(observed, prob, size, groups, numEvents)
ExactMultinomialTestChisquare(observed, prob, size, groups, numEvents)
MonteCarloMultinomialTest(observed, prob, size, groups,
                          numEvents, ntrial, atOnce)
MonteCarloMultinomialTestChisquare(observed, prob, size, groups,
                          numEvents, ntrial, atOnce)
chisqStat(observed,expected)
findVectors(groups,size)
```

Arguments

`observed`	vector describing the observation: contains the *observed numbers* of items in each category.
`prob`	vector describing the model: contains the *hypothetical probabilities* corresponding to each category.
`expected`	vector containing the expected numbers of items in each category under the assumption that the model is valid.
`size`	sample size, sum of the components of the vector `observed`.
`groups`	number of categories in the experiment.
`numEvents`	number of possible outcomes of the experiment.
`ntrial`	number of simulated samples in the Monte Carlo approach.
`atOnce`	a parameter of more technical nature. Determines how much memory is used for big arrays.

Details

These functions are not intended to be called by the user.

---

multinomial.test	*Exact Multinomial Test: Goodness-of-Fit Test for Discrete Multivariate Data*

Description

Goodness-of-fit tests for discrete multivariate data. It is tested if a given observation is likely to have occurred under the assumption of an ab-initio model. Monte Carlo methods are provided to make the function capable of solving high-dimensional problems.

Usage

```
multinomial.test(observed, prob, useChisq = FALSE,
                 MonteCarlo = FALSE, ntrial = 1e6, atOnce = 1e6)
```

Arguments

`observed`	vector describing the observation: contains the *observed numbers* of items in each category.
`prob`	vector describing the model: contains the *hypothetical probabilities* corresponding to each category.
`useChisq`	if TRUE, Pearson's chisquare is used as a distance measure between observed and expected frequencies.
`MonteCarlo`	if TRUE, a Monte Carlo approach is used.
`ntrial`	number of simulated samples in the Monte Carlo approach.
`atOnce`	a parameter of more technical nature. Determines how much memory is used for big arrays.

Details

The Exact Multinomial Test is a Goodness-of-fit test for discrete multivariate data. It is tested if a given observation is likely to have occurred under the assumption of an ab-initio model. In the experimental setup belonging to the test, *n* items fall into *k* categories with certain probabilities (sample size *n* with *k* categories). The **observation**, described by the vector observed, indicates how many items have been observed in each category. The **model**, determined by the vector prob, assigns to each category the hypothetical probability that an item falls into it. Now, if the observation is unlikely to have occurred under the assumption of the model, it is advisable to regard the model as *not* valid. The p-value estimates how likely the observation is, given the model. In particular, low p-values suggest that the model is *not* valid. The **default approach** used by multinomial.test obtains the p-values by calculating the exact probabilities of *all* possible outcomes given n and k, using the multinomial probability distribution function [dmultinom](#) provided by R. Then, by default, the p-value is obtained by summing the probabilities of all outcomes which are less likely than the observed outcome (or equally likely as the observed outcome), i.e. by summing all $p(i) <= p(observed)$ (distance measure based on probabilities). Alternatively, the p-value can be obtained by summing the probabilities of all outcomes connected with a chisquare no smaller than the chisquare connected with the actual observation (distance measure based on chisquare). The latter is triggered by setting useChisq = TRUE. Having a sample of size *n* in an experiment with *k* categories, the number of distinct possible outcomes is the binomial coefficient choose(n+k-1,k-1). This number grows rapidly with increasing parameters *n* and *k*. If the parameters grow too big, numerical calculation might fail because of time or memory limitations. In this case, usage of a **Monte Carlo approach** provided by multinomial.test is suggested. A Monte Carlo approach, activated by setting MonteCarlo = TRUE, simulates withdrawal of *ntrial* samples of size *n* from the hypothetical distribution specified by the vector prob. The default value for *ntrial* is 100000 but might be incremented for big *n* and *k*. The advantage of the Monte Carlo approach is that memory requirements and running time are essentially determined by *ntrial* but not by *n* or *k*. By default, the p-value is then obtained by summing the relative frequencies of occurrence of unusual outcomes, i.e. of outcomes occurring less frequently than the observed one (or equally frequent as the observed one). Alternatively, as above, Pearson's chisquare can be used as a distance measure by setting useChisq = TRUE. The parameter *atOnce* is of more technical nature, with a default value of 1000000. This value should be decremented for computers with low memory to avoid overflow, and can be incremented for large-CPU computers to speed up calculations. The parameter is only effective for Monte Carlo calculations.

Value

id	textual description of the method used.
size	sample size *n*, equals the sum of the components of the vector observed.
groups	number of categories *k* in the experiment, equals the number of components of the vector observed.
numEvents	number of different events for the model considered.
stat	textual description of the distance measure used.
allProb	vector containing the probabilities (rel. frequencies for the Monte Carlo approach) of all possible outcomes (might be huge for big *n* and *k*).
criticalValue	the critical value of the hypothesis test.
ntrial	number of trials if the Monte Carlo approach was used, NULL otherwise.
p.value	the calculated p-value rounded to four significant digits.

Note

For two categories ($k = 2$), the test is called Exact Binomial Test.

Author(s)

Uwe Menzel <uwemenzel@gmail.com>

References

H. Bayo Lawal (2003) *Categorical data analysis with SAS and SPSS applications*, Volume 1, Chapter 3 ISBN: 978-0-8058-4605-8

Read, T. R. C. and Cressie, N. A. C. (1988). *Goodness-of-fit statistics for discrete multivariate data.* Springer, New York.

See Also

The Multinomial Distribution: dmultinom

Examples

```
## Load the EMT package:
library(EMT)

## Input data for a three-dimensional case:
observed <- c(5,2,1)
prob <- c(0.25, 0.5, 0.25)

## Calculate p-value using default options:
out <- multinomial.test(observed, prob)
# p.value = 0.0767

## Plot the probabilities for each event:
plotMultinom(out)

## Calculate p-value for the same input using Pearson's chisquare:
out <- multinomial.test(observed, prob, useChisq = TRUE)
# p.value = 0.0596 ; not the same!

## Test the hypothesis that all sides of a dice have the same probabilities:
prob <- rep(1/6, 6)
observed <- c(4, 5, 2, 7, 0, 1)
out <- multinomial.test(observed, prob)
# p.value = 0.0357 -> better get another dice!

# the same problem using a Monte Carlo approach:
## Not run:
  out <- multinomial.test(observed, prob, MonteCarlo = TRUE, ntrial = 5e+6)

## End(Not run)
```

---

plotMultinom   *Plot the Probability distribution fot the Exact Multinomial Test*

Description

This function takes the results of `multinomial.test` as input and plots the calculated probability distribution.

Usage

```
plotMultinom(listMultinom)
```

Arguments

listMultinom   a list created by running the function `multinomial.test`.

Details

The function `plotMultinom` displays a barplot of the probabilities for the individual events. The probabilities are shown in descending order from the left to the right. Events contributing to the p-value are marked red. Plots are only made if the number of different events is lower than or equal to 100 and for low number of trials in Monte Carlo simulations.

Value

The first argument (*listMultinom*) is returned without modification.

Author(s)

Uwe Menzel <uwemenzel@gmail.com>

See Also

The Multinomial Distribution: multinomial.test

Examples

```
## Load the EMT package:
library(EMT)

## input and calculation of p-values:
observed <- c(5,2,1)
prob <- c(0.25, 0.5, 0.25)
out <- multinomial.test(observed, prob)      # p.value = 0.0767

## Plot the probability distribution:
plotMultinom(out)
```

Index