Package 'R2sample'

February 1, 2025

Title Various Methods for the Two Sample Problem

Version 3.0.0

Description The routine twosample_test() in this package runs the two sample test using various test statistic. The p values are found via permutation or large sample theory. The routine twosample_power() allows the calculation of the power in various cases, and plot_power() draws the corresponding power graphs. The routine run.studies allows a user to quickly study the power of a new method and how it compares to some of the standard ones.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp

Imports Rcpp, parallel, shiny, ggplot2, stats, graphics

Suggests rmarkdown, knitr

VignetteBuilder knitr

Depends R (>= 3.5)

LazyData true

NeedsCompilation yes

Author Wolfgang Rolke [aut, cre] (<https://orcid.org/0000-0002-3514-726X>)

Maintainer Wolfgang Rolke <wolfgang.rolke@upr.edu>

Repository CRAN

Date/Publication 2025-02-01 20:20:05 UTC

Contents

asymptotic_pvalues 2
case.studies 2
chi_power 3
myTS2 4
plot_power 4

power_newtest 5
power_studies_results 5
pvaluecdf 6
run.studies 6
run_shiny 7
signif.digits 8
twosample_power 8
twosample_test 9
twosample_test_adjusted_pvalue 11

Index **13**

`asymptotic_pvalues` *This function finds the p values of several tests based on large sample theory*

Description

This function finds the p values of several tests based on large sample theory

Usage

```
asymptotic_pvalues(x, n, m)
```

Arguments

`x`	a vector of test statistics
`n`	size of sample 1
`m`	size of sample 2

Value

A vector of p values.

`case.studies` *This function creates the functions needed to run the various case studies.*

Description

This function creates the functions needed to run the various case studies.

Usage

```
case.studies(which, nsample = 500)
```

Arguments

which	name of the case study.
nsample	=500, sample size.

Value

a list of functions

---

chi_power *This function runs the chi-square test for continuous or discrete data*

Description

This function runs the chi-square test for continuous or discrete data

Usage

```
chi_power(
  rxy,
  alpha = 0.05,
  B = 1000,
  xparam,
  yparam,
  nbins = c(50, 10),
  minexpcount = 5,
  typeTS
)
```

Arguments

rxy	a function to generate data
alpha	=0.05 type I error probability of test
B	=1000 number of simulation runs
xparam	vector of parameter values
yparam	vector of parameter values
nbins	=c(50, 10) number of desired bins
minexpcount	=5 smallest number of counts required in each bin
typeTS	type of problem, continuous/discrete, with/without weights

Value

A matrix of power values

myTS2 *a local function needed for the vignette*

Description

a local function needed for the vignette

Usage

```
myTS2(x, y, vals)
```

Arguments

x	An integer vector.
y	An integer vector.
vals	A numeric vector with the values of the discrete rv.

Value

A vector with test statistics

plot_power *This function draws the power graph, with curves sorted by the mean power and smoothed for easier reading.*

Description

This function draws the power graph, with curves sorted by the mean power and smoothed for easier reading.

Usage

```
plot_power(pwr, xname = " ", title = " ", Smooth = TRUE, span = 0.25)
```

Arguments

pwr	a matrix of power values, usually from the twosample_power command
xname	Name of variable on x axis
title	(Optional) title of graph
Smooth	=TRUE lines are smoothed for easier reading
span	=0.25bandwidth of smoothing method

Value

plt, an object of class ggplot.

power_newtest *This function estimates the power of test routines that calculate p value(s)*

Description

This function estimates the power of test routines that calculate p value(s)

Usage

```
power_newtest(TS, f, param_alt, TSextra, alpha = 0.05, B = 1000)
```

Arguments

TS	routine to calculate test statistics.
f	routine that generates data.
param_alt	values of parameter under the alternative hypothesis.
TSextra	list passed to TS.
alpha	=0.05 type I error.
B	= 1000 number of simulation runs to estimate the power.

Value

A matrix of power values

power_studies_results *power_studies_results*

Description

the results of the included power studies

Usage

```
power_studies_results
```

Format

'power_studies_results':
A list of matrices with powers

pvaluecdf *pvaluecdf*

Description

data to draw a graph in vignette

Usage

```
pvaluecdf
```

Format

'pvaluecdf':
A matrix

run.studies *This function runs the case studies included in the package*

Description

This function runs the case studies included in the package

Usage

```
run.studies(
  TS,
  study,
  TSextra,
  With.p.value = FALSE,
  BasicComparison = TRUE,
  nsample = 500,
  alpha = 0.05,
  param_alt,
  maxProcessor,
  B = c(1000, 1000)
)
```

Arguments

TS	routine to calculate test statistics.
study	either the name of the study, or its number. If missing all the studies are run.
TSextra	list passed to TS.
With.p.value	=FALSE does user supplied routine return p values?

BasicComparison	=TRUE if true compares tests on one default value of parameter of the alternative distribution.
nsample	= 500, desired sample size.
alpha	=0.05 type I error
param_alt	(list of) values of parameter under the alternative hypothesis. If missing included values are used.
maxProcessor	number of cores to use for parallel programming
B	= c(1000,1000)

Value

A (list of) matrices of p.values

Examples

```
#The new test is a simple chisquare test:
chitest = function(x, y, TSextra) {
   nbins=TSextra$nbins
   nx=length(x);ny=length(y);n=nx+ny
   xy=c(x,y)
   bins=quantile(xy, (0:nbins)/nbins)
   Ox=hist(x, bins, plot=FALSE)$counts
   Oy=hist(y, bins, plot=FALSE)$counts
   tmp=sqrt(sum(Ox)/sum(Oy))
   chi = sum((Ox/tmp-Oy*tmp)^2/(Ox+Oy))
   pval=1-pchisq(chi, nbins-1)
   out=ifelse(TSextra$statistic,chi,pval)
   names(out)="ChiSquare"
   out
}
TSextra=list(nbins=5,statistic=FALSE) # Use 5 bins and calculate p values
run.studies(chitest,TSextra=TSextra, With.p.value=TRUE, B=c(100, 100))
```

---

run_shiny *Runs the shiny app associated with R2sample package*

Description

Runs the shiny app associated with R2sample package

Usage

```
run_shiny()
```

Value

No return value, called for side effect of opening a shiny app

`signif.digits` *This function does some rounding to nice numbers*

Description

This function does some rounding to nice numbers

Usage

```
## S3 method for class 'digits'
signif(x, d = 4)
```

Arguments

`x`	a list of two vectors
`d`	=4 number of digits to round to

Value

A list with rounded vectors

`twosample_power` *Find the power of various two sample tests using Rcpp and parallel computing.*

Description

Find the power of various two sample tests using Rcpp and parallel computing.

Usage

```
twosample_power(
  f,
  ...,
  TS,
  TSextra,
  alpha = 0.05,
  B = c(1000, 1000),
  nbins = c(50, 10),
  minexpcount = 5,
  UseLargeSample,
  samplingmethod = "independence",
  maxProcessor
)
```

Arguments

f	function to generate a list with data sets x, y and (optional) vals, weights
...	additional arguments passed to f, up to 2
TS	routine to calculate test statistics for non-chi-square tests
TSextra	additional info passed to TS, if necessary
alpha	=0.05, the level of the hypothesis test
B	=c(1000, 2000), number of simulation runs for power and permutation test.
nbins	=c(50,10), number of bins for chi large and chi small.
minexpcount	=5 minimum required count for chi square tests
UseLargeSample	should p values be found via large sample theory if n,m>10000?
samplingmethod	=independence or MCMC in discrete data case
maxProcessor	maximum number of cores to use. If maxProcessor=1 no parallel computing is used.

Value

A numeric vector of power values.

Examples

```
f=function(mu) list(x=rnorm(25), y=rnorm(25, mu))
twosample_power(f, mu=c(0,2), B=c(100, 100), maxProcessor = 1)
f=function(n, p) list(x=table(sample(1:5, size=1000, replace=TRUE)),
      y=table(sample(1:5, size=n, replace=TRUE,
      prob=c(1, 1, 1, 1, p))), vals=1:5)
twosample_power(f, n=c(1000, 2000), p=c(1, 1.5), B=c(100, 100), maxProcessor = 1)
```

---

twosample_test *This function runs a number of two sample tests using Rcpp and parallel computing.*

Description

This function runs a number of two sample tests using Rcpp and parallel computing.

Usage

```
twosample_test(
  x,
  y,
  vals = NA,
  TS,
  TSextra,
  wx = rep(1, length(x)),
```

```
  wy = rep(1, length(y)),
  B = 5000,
  nbins = c(50, 10),
  maxProcessor,
  UseLargeSample,
  samplingmethod = "independence",
  doMethods = "all"
)
```

Arguments

x	a vector of numbers if data is continuous or of counts if data is discrete.
y	a vector of numbers if data is continuous or of counts if data is discrete.
vals	=NA, a vector of numbers, the values of a discrete random variable. NA if data is continuous data.
TS	routine to calculate test statistics for non-chi-square tests
TSextra	additional info passed to TS, if necessary
wx	A numeric vector of weights of x.
wy	A numeric vector of weights of y.
B	=5000, number of simulation runs for permutation test
nbins	=c(50,10), number of bins for chi square tests.
maxProcessor	maximum number of cores to use. If missing (the default) no parallel processing is used.
UseLargeSample	should p values be found via large sample theory if n,m>10000?
samplingmethod	="independence" or "MCMC" for discrete data
doMethods	="all" Which methods should be included? If missing all methods are used.

Value

A list of two numeric vectors, the test statistics and the p values.

Examples

```
R2sample::twosample_test(rnorm(1000), rt(1000, 4), B=1000)
myTS=function(x,y) {z=c(mean(x)-mean(y),sd(x)-sd(y));names(z)=c("M","S");z}
R2sample::twosample_test(rnorm(1000), rt(1000, 4), TS=myTS, B=1000)
vals=1:5
x=table(sample(vals, size=100, replace=TRUE))
y=table(sample(vals, size=100, replace=TRUE, prob=c(1,1,3,1,1)))
R2sample::twosample_test(x, y, vals)
```

`twosample_test_adjusted_pvalue` *This function runs a number of two sample tests using Rcpp and parallel computing and then finds the correct p value for the combined tests.*

Description

This function runs a number of two sample tests using Rcpp and parallel computing and then finds the correct p value for the combined tests.

Usage

```
twosample_test_adjusted_pvalue(
  x,
  y,
  vals = NA,
  TS,
  TSextra,
  wx = rep(1, length(x)),
  wy = rep(1, length(y)),
  B = c(5000, 1000),
  nbins = c(50, 10),
  samplingmethod = "independence",
  doMethods
)
```

Arguments

x	a vector of numbers if data is continuous or of counts if data is discrete.
y	a vector of numbers if data is continuous or of counts if data is discrete.
vals	=NA, a vector of numbers, the values of a discrete random variable. NA if data is continuous data.
TS	routine to calculate test statistics for non-chi-square tests
TSextra	additional info passed to TS, if necessary
wx	A numeric vector of weights of x.
wy	A numeric vector of weights of y.
B	=c(5000, 1000), number of simulation runs for permutation test
nbins	=c(50,10), number of bins for chi square tests.
samplingmethod	="independence" or "MCMC" for discrete data
doMethods	Which methods should be included?

Value

A list of two numeric vectors, the test statistics and the p values.

Examples

```
x=rnorm(100)
y=rt(200, 4)
R2sample::twosample_test_adjusted_pvalue(x, y, B=c(500, 500))
vals=1:5
x=table(c(1:5, sample(1:5, size=100, replace=TRUE)))-1
y=table(c(1:5, sample(1:5, size=100, replace=TRUE, prob=c(1,1,3,1,1))))-1
R2sample::twosample_test_adjusted_pvalue(x, y, vals, B=c(500, 500))
```

Index

∗ **datasets**
power_studies_results, 5
pvaluecdf, 6

asymptotic_pvalues, 2

case.studies, 2
chi_power, 3

myTS2, 4

plot_power, 4
power_newtest, 5
power_studies_results, 5
pvaluecdf, 6

run.studies, 6
run_shiny, 7

signif.digits, 8

twosample_power, 8
twosample_test, 9
twosample_test_adjusted_pvalue, 11