Package ‘MultNonParam’

May 7, 2026

Type Package

Title Multivariate Nonparametric Methods

Version 1.3.9

Date 2023-08-29

Author John E. Kolassa [cre],
Stephane Jankowski [aut]

Maintainer John E. Kolassa <`kolassa@stat.rutgers.edu`>

Description A collection of multivariate nonparametric methods, selected in part to support an MS level course in nonparametric statistical methods. Methods include adjustments for multiple comparisons, implementation of multivariate Mann-Whitney-Wilcoxon testing, inversion of these tests to produce a confidence region, some permutation tests for linear models, and some algorithms for calculating exact probabilities associated with one- and two- stage testing involving Mann-Whitney-Wilcoxon statistics. Supported by grant NSF DMS 1712839. See Kolassa and Seifu (2013) <doi:10.1016/j.acra.2013.03.006>.

License GPL-2

LazyData true

Encoding UTF-8

NeedsCompilation yes

Depends R (>= 2.10), ICSNP

RoxygenNote 7.2.1

Repository CRAN

Date/Publication 2023-08-30 04:40:02 UTC

Contents

`MultNonParam-package` *MultNonParam*

Description

A collection of nonparametric methods.

Author(s)

Maintainer: John E. Kolassa <`kolassa@stat.rutgers.edu`>

Authors:

- Stephane Jankowski

`aov.P` *One-way ANOVA using permutation tests*

Description

`aov.P` uses permutation tests instead of classic theory tests to run a one-way or two-way ANOVA.

Usage

```
aov.P(dattab, treatment = NULL, be = NULL)
```

Arguments

`dattab`	The table on which the ANOVA has to be done, or a vector of responses.
`treatment`	If dattab is a table, ignored. If dattab is a vector, a vector of treatment labels.
`be`	If dattab is a table, ignored. If dattab is a vector, a vector of end points of blocks. In this case, blocks must form contiguous subvectors of dattab. If null, no blocking.

Details

The function calls a Fortran code to perform the permutation tests and the ANOVA. The function has to be applied directly on a cross-table of two variables.

Value

A list with fields pv, the p-value obtained with the permutation tests, and tot, the total number of permutations.

`betatest` *Permutation test of assication*

Description

Calculate the p-value for the test of association between two variables using the permutation method.

Usage

```
betatest(x, y)
```

Arguments

`x`	First vector to be associated.
`y`	First vector to be associated.

Value

p-value

Examples

```
#Example using data from plant Qn1 from the CO2 data set.^M
betatest(CO2[CO2$Plant=="Qn1",4],CO2[CO2$Plant=="Qn1",5])
```

`dconcordant` *Calculate the probability atom of the count of concordant pairs among indpendent pairs of random variables.*

Description

Calculate the probability atom of the count of concordant pairs among indpendent pairs of random variables.

Usage

```
dconcordant(ss, nn)
```

Arguments

`ss`	Integer number of pairs
`nn`	number of pairs

Value

real probability

`dmannwhitney` *Mann Whitney Probability Mass function*

Description

Calculates the Mann Whitney Probability Mass function recursively.

Usage

```
dmannwhitney(u, m, n)
```

Arguments

`u`	Statistic value
`m`	Group 1 size
`n`	Group 2 size

Value

Probability that the Mann-Whitney statistic takes the value u under H0

`ecdfcis` *Confidence Intervals for Empirical Cumulative Distribution Functions*

Description

Confidence Intervals for Empirical Cumulative Distribution Functions

Usage

```
ecdfcis(data, alpha = 0.05, dataname = NA, exact = TRUE, newplot = TRUE)
```

Arguments

`data`	vector of observations
`alpha`	1-confidence level.
`dataname`	Name of variable for use in axis labeling
`exact`	logical value controlling whether confidence intervals are exact or asymlptotic.
`newplot`	logical value controlling whether the estimate is added to an existing plot, or whether a new plot should be constructed.

`exactquantileci` *Exact Quantile Confidence Interval*

Description

Calculates exact quanitle confidence intervals by inverting the generalization of the sign test.

Usage

```
exactquantileci(xvec, tau = 0.5, alpha = 0.05, md = 0)
```

Arguments

`xvec`	vector of observations
`tau`	quantile to be estimated. If this is a vector, separate intervals and tests for each value will be calculated.
`alpha`	1-confidence level.
`md`	null value of quantile

Value

A list with components cis, an array with two columns, representing lower and upper bounds, and a vector pvals, of p-values.

`genscorestat` *Normal-theory two sample scorestatistic.*

Description

Calculates the p-value from the normal approximation to the permutation distribution of a two-sample score statistic.

Usage

```
genscorestat(scores, group, correct = 0)
```

Arguments

`scores`	scores of the data.
`group`	numeric or character vector of group identities.
`correct`	half the minimal distance between two potential values of the score statistic.

Value

Object of class htest containing the p-value.

`higgins.fisher.kruskal.test` *Fisher's LSD method applied to the Kruskal-Wallis test*

Description

This function applies a rank-based method for controlling experiment-wise error. Two hypothesis have to be respected: normality of the distribution and no ties in the data. The aim is to be able to detect, among k treatments, those who lead to significant differencies in the values for a variable of interest.

Usage

```
higgins.fisher.kruskal.test(resp, grp, alpha = 0.05)
```

Arguments

`resp`	vector containing the values for the variable of interest.
`grp`	vector specifying in which group is each observation.
`alpha`	level of the test.

Details

First, the Kruskal-Wallis test is used to test the equality of the distributions of each treatment. If the test is significant at the level `alpha`, the method can be applied.

Value

A matrix with two columns. Each row indicates a combinaison of two groups that have significant different distributions.

References

J.J. Higgins, (2004), *Introduction to Modern Nonparametric Statistics*, Brooks/Cole, Cengage Learning.

`kweffectsize` *Sample Size for the Kruskal-Wallis test.*

Description

`kweffectsize` approximates effect size for the Kruskal-Wallis test, using a chi-square approximation under the null, and a non-central chi-square approximation under the alternative. The noncentrality parameter is calculated using alternative means and the null variance structure.

Usage

```
kweffectsize(
  totsamp,
  shifts,
  distname = c("normal", "logistic", "cauchy"),
  targetpower = 0.8,
  proportions = rep(1, length(shifts))/length(shifts),
  level = 0.05
)
```

Arguments

`totsamp`	sample size
`shifts`	The offsets for the various populations, under the alternative hypothesis. This is used for direction on input.
`distname`	The distribution of the underlying observations; normal and logistic are currently supported.
`targetpower`	The distribution of the underlying observations; normal and logistic are currently supported.
`proportions`	The proportions in each group.
`level`	The test level.

Details

The standard noncentral chi-square power formula, or Monte Carlo, is used.

Value

A list with components power, giving the power approximation, ncp, giving the noncentrality parameter, cv, giving the critical value, probs, giving the intermediate output from pairwiseprobability, and expect, the quantities summed before squaring in the noncentrality parameter.

Examples

```
#Calculate the effecct size necessary to have the desired power .8 for a test
#with the level .5 with sample size 60, group centers 0, 1, and 2,
#normally distributed observations, evenly split among the three groups.
kweffectsize(60,c(0,1,2),"normal")
```

`kwpower` *Power for the Kruskal-Wallis test.*

Description

`kwpower` approximates power for the Kruskal-Wallis test, using a chi-square approximation under the null, and a non-central chi-square approximation under the alternative. The noncentrality parameter is calculated using alternative means and the null variance structure.

Usage

```
kwpower(
  nreps,
  shifts,
  distname = c("normal", "cauchy", "logistic"),
  level = 0.05,
  mc = 0,
  taylor = FALSE
)
```

Arguments

`nreps`	The numbers in each group.
`shifts`	The offsets for the various populations, under the alternative hypothesis.
`distname`	The distribution of the underlying observations; normal, cauchy, and logistic are currently supported.
`level`	The test level.
`mc`	0 for asymptotic calculation, or positive for mc approximation.
`taylor`	logical determining whether Taylor series approximation is used for probabilities.

Details

The standard noncentral chi-square power formula, or Monte Carlo, is used.

Value

A list with components power, giving the power approximation, ncp, giving the noncentrality parameter, cv, giving the critical value, probs, giving the intermediate output from pairwiseprobability, and expect, the quantities summed before squaring in the noncentrality parameter.

Examples

```
#Calculate the power for the Kruskal Wallis test for normal observations,
#10 observations in each of three groups, with groups centered at 0, 1, 2.
#Level is 0.05 by default.
kwpower(rep(10,3),c(0,1,2),"normal")
```

`kwsamplesize` *Sample Size for the Kruskal-Wallis test.*

Description

`kwsamplesize` approximates sample size for the Kruskal-Wallis test, using a chi-square approximation under the null, and a non-central chi-square approximation under the alternative. The noncentrality parameter is calculated using alternative means and the null variance structure.

Usage

```
kwsamplesize(
  shifts,
  distname = c("normal", "logistic", "cauchy"),
  targetpower = 0.8,
  proportions = rep(1, length(shifts))/length(shifts),
  level = 0.05,
  taylor = FALSE
)
```

Arguments

`shifts`	The offsets for the various populations, under the alternative hypothesis.
`distname`	The distribution of the underlying observations; normal and logistic are currently supported.
`targetpower`	The distribution of the underlying observations; normal and logistic are currently supported.
`proportions`	The proportions in each group.
`level`	The test level.
`taylor`	Logical flag forcing the approximation of exceedence probabilities using the first derivative at zero.

Details

The standard noncentral chi-square power formula, is used.

Value

A list with the total number of observations needed to obtain approximate power, as long as this number is split amomg groups according to argument proportion.

Examples

```
#Calculate the sample size necessary to detect differences among three
#groups with centers at 0,1,2, from normal observations, using a test of
#level 0.05 and power 0.80.
kwsamplesize(c(0,1,2),"normal")
```

`mannwhitney.test` *Perform the Mann Whitney two-sample test*

Description

Perform the Mann Whitney two-sample test

Usage

```
mannwhitney.test(x, y, alternative = c("two.sided", "less", "greater"))
```

Arguments

`x`	A vector of values from the first sample.
`y`	A vector of values from the first sample.
`alternative`	Specification of alternative hypothesis.

Value

Test results of class htest

Examples

```
mannwhitney.test(rnorm(10),rnorm(10)+.5)
```

`mood.median.test` *Mood's Median test, extended to odd sample sizes.*

Description

Test whether two samples come from the same distribution. This version of Mood's median test is presented for pedagogical purposes only. Many authors successfully argue that it is not very powerful. The name "median test" is a misnomer, in that the null hypothesis is equality of distributions, and not just equality of median. Exact calculations are not optimal for the odd sample size case.

Usage

```
mood.median.test(x, y, exact = FALSE)
```

Arguments

`x`	First data set.
`y`	Second data set.
`exact`	Indicator for whether the test should be done exactly or approximately.

Details

The exact case reduces to Fisher's exact test.

Value

The two-sided p-value.

`nextp` *Next permutation*

Description

Cycles through permutations of first argument

Usage

```
nextp(perm, b = 1)
```

Arguments

`perm`	indices to be permutedj
`b`	number to begin at. Set equal to 1.

Value

The next permutation

`page.test.unbalanced` *Perform Page test for unbalanced two-way design*

Description

Perform Page test for unbalanced two-way design

Usage

```
page.test.unbalanced(x, trt, blk, sides = 2)
```

Arguments

`x`	A vector of responses
`trt`	A vector of consecutive integers starting at 1 indicating treatment
`blk`	A vector of consecutive integers starting at 1 indicating block
`sides`	A single integer indicating sides. Defaults to 2.

Value

P-value for Page test.

Examples

```
page.test.unbalanced(rnorm(15),rep(1:3,5),rep(1:5,rep(3,5)))
```

`pairwiseprobabilities` *Pairwise probabilities of Exceedence*

Description

`pairwiseprobabilities` calculates probabilities of one variable exceeding another, where the variables are independent, and with identical distributions except for a location shift. This calculation is useful for power of Mann-Whitney-Wilcoxon, Jonckheere-Terpstra, and Kruskal-Wallis testing.

Usage

```
pairwiseprobabilities(
  shifts,
  distname = c("normal", "cauchy", "logistic"),
  taylor = FALSE
)
```

Arguments

`shifts`	The offsets for the various populations, under the alternative hypothesis.
`distname`	The distribution of the underlying observations; normal, cauchy, and logistic are currently supported.
`taylor`	Logical flag forcing the approximation of exeedence probabilities using a Taylor series.

Details

Probabilities of particular families must be calculated analytically.

Value

A matrix with as many rows and colums as there are shift parameters. Row i and column j give the probability of an observation from group j exceeding one from group i.

Examples

```
pairwiseprobabilities(c(0,1,2),"normal")
```

`pconcordant` *Calculate the cumulative distribution of the count of concordant pairs among indpendent pairs of random variables.*

Description

Calculate the cumulative distribution of the count of concordant pairs among indpendent pairs of random variables.

Usage

```
pconcordant(ss, nn)
```

Arguments

`ss`	Integer number of pairs
`nn`	number of pairs

Value

real probability

`powerplot` *Power Plot*

Description

Plots powers for the Kruskall-Wallis test, via Monte Carlo and two approximations.

Usage

```
powerplot(
  numgrps = 3,
  thetadagger = NULL,
  nnvec = 5:30,
  nmc = 50000,
  targetpower = 0.8,
  level = 0.05
)
```

Arguments

`numgrps`	Number of groups to compare
`thetadagger`	Direction of effect
`nnvec`	vector of numbers per group.
`nmc`	Number of Monte Carlo trials
`targetpower`	Target power for test
`level`	level for test.

`probabilityderiv` *Derivative of pairwise probabilities of Exceedence*

Description

`probabilityderiv` calculates derivatives probabilities of one variable exceeding another, where the variables are independent, and with identical distributions except for a location shift, at the null hypothesis. This calculation is useful for power of Mann-Whitney-Wilcoxon, Jonckheere-Terpstra, and Kruskal-Wallis testing.

Usage

```
probabilityderiv(distname = c("normal", "cauchy", "logistic"))
```

Arguments

`distname`	The distribution of the underlying observations; normal and logistic are currently supported.

Details

Probabilities of particular families must be calculated analytically, and then differentiated.

Value

The scalar derivative.

`probest` *Stratified Multivariate Kawaguchi Koch Wang Estimators*

Description

Function that return the estimators and their variance-covariance matrix calculated with the Kawaguchi - Koch - Wang method.

Usage

```
probest(ds, resp, grp, str = NULL, covs = NULL, delta = NA, correct = FALSE)
```

Arguments

`ds`	The data frame to be used.
`resp`	The vector of the response manifest variable. There can be more than one variable. It has to be the name of the variable as a character string.
`grp`	The vector of the variable that divides the population into groups. It has to be the name of the variable as a character string.
`str`	The vector of the variable used for the strata. It has to be the name of the variable as a character string.
`covs`	The covariates to be used in the model. It has to be the name of the variable as a character string.
`delta`	Offeset for covariates.
`correct`	Should the variance estimator be corrected as in Chen and Kolassa?

Details

The function calls a Fortran code to calculate the estimators `b` and their variance-covariance matrix `Vb`

Value

A list with components b, the vector of adjusted estimates from the method, and Vb, the corresponding estimated covariance matrix.

References

A. Kawaguchi, G. G. Koch and X. Wang (2012), "Stratified Multivariate Mann-Whitney Estimators for the Comparison of Two Treatments with Randomization Based Covariance Adjustment", *Statistics in Biopharmaceutical Research* 3 (2) 217-231.

J. E. Kolassa and Y. Seifu (2013), Nonparametric Multivariate Inference on Shift Parameters, *Academic Radiology* 20 (7), 883-888.

Examples

```
# Breast cancer data from the MultNonParam package.
data(sotiriou)
attach(sotiriou)
#First simple plot of the data
plot(AGE,TUMOR_SIZE,pch=(recur+1),main="Age and Tumor Size",
  sub="Breast Cancer Recurrence Data",xlab="Age (years)",
  ylab="Tumor Size",col=c("blue","darkolivegreen"))
legend(31,8,legend=c("Not Recurrent","Recurrent"),
  pch=1:2,col=c("blue","darkolivegreen"))
#AGE and TUMOR_SIZE are the response variables, recur is used for the groups,
#TAMOXIFEN_TREATMENT for the stratum and ELSTON.ELLIS_GRADE is a covariate.
po<-probest(sotiriou,c("AGE","TUMOR_SIZE"),"recur",
  "TAMOXIFEN_TREATMENT","ELSTON.ELLIS_GRADE")
```

`prostate` *prostate*

Description

221 prostate cancer patients are collected in this data set.

Format

- hosp : Hospital in which the patient is hospitalized.
- stage : stage of the cancer.
- gleason score : used to help evaluate the prognosis of the cancer.
- psa : prostate-specific antigen.
- age : age of the patient.
- advanced : boolean. `TRUE` if the cancer is advanced.

References

A. V. D'Amico, R. Whittington, S. B. Malkowicz, D. Schultz, K. Blank, G. A. Broderick, J. E. Tomaszewski, A. A. Renshaw, I. Kaplan, C. J. Beard, A. Wein (1998) , *Biochemical outcome after radical prostatectomy, external beam radiation therapy, or interstitial radiation therapy for clinically localized prostate cancer*, JAMA : the journal of the American Medical Association 280 969-74.

Examples

```
data(prostate)
attach(prostate)
plot(age,psa,main="Age and PSA",sub="Prostate Cancer Data",
   xlab="Age (years)",ylab="PSA")
```

`qconcordant` *Calculate the quantiles of the count of concordant pairs among indpendent pairs of random variables.*

Description

Calculate the quantiles of the count of concordant pairs among indpendent pairs of random variables.

Usage

```
qconcordant(qq, nn, exact = TRUE)
```

Arguments

`qq`	Desired quantile
`nn`	number of pairs
`exact`	flag to trigger exact calculation when possible.

Value

Integer quantile

`sensitivity.plot` *Compare the sensitivity of different statistics.*

Description

Compare the sensitivity of different statistics.

Usage

```
sensitivity.plot(y, sub, stats)
```

Arguments

`y`	vector of the data.
`sub`	subtitle for the plot.
`stats`	vector of functions to be plotted.

Details

To compare the sensitivity, outliers are added to the original data. The shift of each statistics due to the new value is measured and plotted.

`shiftcr` *Nonparametric Confidence Region for a Vector Shift Parameter*

Description

Inversion of a one-sample bivariate rank test is used to produce a confidence region. The region is constructed by building a grid of potential parameter values, evaluating the test statistic on each grid point, collecting the p-values, and then drawing the appropriate countour of the p-values. The grid is centered at the bivariate median of the data set.

Usage

```
shiftcr(xm,  hpts = 50)
```

Arguments

`xm`	A two-column matrix of bivariate data whose two location parameters are to be estimated.
`hpts`	Controls the number of grid points, by constructing a grid of 2*hpts+1 on each side.

Value

nothing

`solvencp` *Noncentrality Parameter for a Given Level and Power*

Description

This function calculates the noncentrality parameter required to give a test whose null distribution is central chi-square and whose alternative distribution is noncentral chi-square the required level and power.

Usage

```
solvencp(df, level = 0.05, targetpower = 0.8)
```

Arguments

`df`	Common degrees of freedom for null and alternative distributions.
`level`	Level (that is, type I error rate) for the test.
`targetpower`	Desired power

Value

required noncentrality parameter.

Examples

```
solvencp(4)
```

`sotiriou` *Breast cancer data set*

Description

187 breast cancer patients are collected in this data set.

Usage

```
data(sotiriou)
```

Format

A data set with the following variables

- AGE : Age of the patient
- TUMOR_SIZE : The size of the tumor, numeric variable
- recur : 1 if the patient has a recurent breast cancer, 0 if it is not reccurent.
- ELSTON.ELLIS_GRADE : Elston Ellis grading system in order toclassify the breast cancers. It can be a low, intermediate or high grade (high being the worst prognosis)
- TAMOXIFEN_TREATMENT : boolean. `TRUE` if the patient is treated with the Tamoxifen treatment.

Source

https://gdoc.georgetown.edu/gdoc/

References

S. Madhavan, Y. Gusev, M. Harris, D. Tanenbaum, R. Gauba, K. Bhuvaneshwar, A. Shinohara, K. Rosso, L. Carabet, L. Song, R. Riggins, S. Dakshanamurthy, Y. Wang, S. Byers, R. Clarke, L. Weiner (2011), *A systems medicine platform for personalized oncology*, Neoplasia 13.

C. Sotiriou, P. Wirapati, S. Loi, A. Harris, S. Fox, J. Smeds, H. Nordgren, P. Farmer, V. Praz, B. Haibe-Kains, C. Desmedt, D. Larsimont, F. Cardoso, H. Peterse, D. Nuyten, M. Buyse, M. Van de Vijver, J. Bergh, M. Piccart, M. Delorenzi (2006), *Gene expression profiling in breast cancer: understanding the molecular basis of histologic grade to improve prognosis*, Journal of the National Cancer Institute 98 262-72.

Examples

```
data(sotiriou)
plot(sotiriou$AGE,sotiriou$TUMOR_SIZE,pch=(sotiriou$recur+1),
   main="Age and Tumor Size",
   sub="Breast Cancer Recurrence Data",
   xlab="Age (years)",ylab="Tumor Size",
   col=c("blue","darkolivegreen"))
legend(31,8,legend=c("Not Recurrent","Recurrent"),pch=1:2,
   col=c("blue","darkolivegreen"))
```

`symscorestat` *Generalization of Wilcoxon signed rank test*

Description

This function returns either exact or asymptotic p-values for score tests of the null hypothesis of univariate symmetry about 0.

Usage

```
symscorestat(y, scores = NULL, exact = F, sides = 1)
```

Arguments

`y`	Vector of data on which test will be run.
`scores`	Scores to be used for the test. Defaults to integers 1:length(y).
`exact`	Logical variable indicating whether the exact p-value should be calculate. Default is false.
`sides`	Integer; 1 for one sided test rejecting for large values of the statistic, and 2 for the two-sided test. Defaults to 1.

Details

The statistic considered here is the sum of scores corresponding to those entries in y that are positive. If exact=T, the function calls a Fortran code to cycle through all permutations. If exact=F, the expectation of the statistic is calculated as half the sum of the scores, the variance is calculated as one quarter the sum of squares of scores about their mean, and the statistic is compared to its approximating normal distribution.

Value

A list with components pv, the p-value obtained with the permutation tests, and tot, the total number of rearrangements of the data considred in calculating the p-value.

References

J.J. Higgins, (2004), *Introduction to Modern Nonparametric Statistics*, Brooks/Cole, Cengage Learning.

Examples

```
symscorestat(y=c(1,-2,3,-4,5),exact=TRUE)
```

`terpstra.test` *Perform the Terpstra version of the multi-ordered-sample test*

Description

Perform the Terpstra version of the multi-ordered-sample test

Usage

```
terpstra.test(x, g, alternative = c("two.sided", "less", "greater"))
```

Arguments

`x`	A vector of values from all samples.
`g`	A vector of group labels.
`alternative`	Specification of alternative hypothesis.

Value

Test results of class htest

Examples

```
terpstra.test(rnorm(15),rep(1:3,5))
```

`terpstrapower` *Power for the nonparametric Terpstra test for an ordered effect.*

Description

`terpstrapower` approximates power for the one-sided Terpstra test, using a normal approximation with expectations under the null and alternative, and using the null standard deviation.

Usage

```
terpstrapower(
  nreps,
  shifts,
  distname = c("normal", "logistic"),
  level = 0.025,
  mc = 0
)
```

Arguments

`nreps`	The numbers in each group.
`shifts`	The offsets for the various populations, under the alternative hypothesis.
`distname`	The distribution of the underlying observations; normal and logistic are currently supported.
`level`	The test level.
`mc`	Zero indicates asymptotic calculation. Positive for MC calculation.

Details

The standard normal-theory power formula is used.

Value

A list with components power, giving the power approximation, expect, giving null and alternative expectations, var, giving the null variance, probs, giving the intermediate output from pairwiseprobability, and level.

Examples

```
terpstrapower(rep(10,3),c(0,1,2),"normal")
terpstrapower(c(10,10,10),0:2,"normal",mc=1000)
```

`testve` *Diagnosis for multivariate stratified Kawaguchi - Koch - Wang method*

Description

Diagnostic tool that verifies the normality of the estimates of the probabilities b with the Kawaguchi - Koch - Wang method. The diagnostic method is based on a Monte Carlo method.

Usage

```
testve(n, m, k, nsamp = 100, delta = 0, beta = 0, disc = 0)
```

Arguments

`n`	number of observations in the first group.
`m`	number of observations in the second group.
`k`	number of strata.
`nsamp`	The number of estimates that will be calculated. Must be enough to be sure that the results are interpretable.
`delta`	Offset that depends on group.
`beta`	Correlation between x and y.
`disc`	The Mann Whitney test is designed to handle continuous data, but this method applies to discretized data; `disc` adjusts the discreteness.

Details

This functions serves as a diagnosis to prove that the Kawaguchi - Koch - Wang method gives Gaussian estimates for b. It generates random data sets, to which the Mann Whitney test gets applied. `y` is the generated response variable and `x` the generated covariable related to `y` through a regression model.

Value

Nothing is returned. A QQ plot is drawn.

References

A. Kawaguchi, G. G. Koch and X. Wang (2012), "Stratified Multivariate Mann-Whitney Estimators for the Comparison of Two Treatments with Randomization Based Covariance Adjustment", *Statistics in Biopharmaceutical Research* 3 (2) 217-231.

J. E. Kolassa and Y. Seifu (2013), Nonparametric Multivariate Inference on Shift Parameters, *Academic Radiology* 20 (7), 883-888.

Examples

```
testve(10,15,3,100,0.4)
```

`theil` *Perform the Theil nonparametric estimation and confidence interval for a slope parameter.*

Description

Perform the Theil nonparametric estimation and confidence interval for a slope parameter.

Usage

```
theil(x, y, conf = 0.9)
```

Arguments

`x`	A vector of values of the explanatory variable.
`y`	A vector of values of the response variable.
`conf`	Level of confidence interval.

Value

A list with letters and numbers.

- est - An estimate, the median of pairwise slopes.
- ci - A vector of confidence interval endpoints.

Examples

```
a<-0:19;b<-a^2.5
theil(a,b)
```

`tukey.kruskal.test` *Tukey HSD procedure*

Description

Rank-based method for controlling experiment-wise error. Assume normality of the distribution for the variable of interest.

Usage

```
tukey.kruskal.test(resp, grp, alpha = 0.05)
```

Arguments

`resp`	vector containing the values for the variable of interest.
`grp`	vector specifying in which group is each observation.
`alpha`	level of the test.

Details

The original Tuckey HSD procedure is supposed to be applied for equal sample sizes. However, the `tukey.kruskal.test` function performs the Tukey-Kramer procedure that works for unequal sample sizes.

Value

A logical vector for every combinaison of two groups. `TRUE` if the distribution in one group is significantly different from the distribution in the other group.

References

J.J. Higgins, (2004), *Introduction to Modern Nonparametric Statistics*, Brooks/Cole, Cengage Learning.

`twosamplesurvpvs` *Two Sample Omnibus Tests of Survival Curves*

Description

Returns the Kolmogorov-Smirnov and Anderson-Darling test statistics for two right-censored data sets.

Usage

```
twosamplesurvpvs(times, delta, grp, nmc = 10000, plotme = TRUE, exact = FALSE)
```

Arguments

`times`	Event and censoring times
`delta`	Indicator of event (1) or censoring (0).
`grp`	Variable that divides the population into groups.
`nmc`	Number of Monte Carlo samples for p value calculation
`plotme`	logical; indicates whether to plot or not.
`exact`	logical; indicates whether to use exhaustive enumeration of permutations or not.

Details

The function calls a Fortran code to calculate the estimators `b` and their variance-covariance matrix `Vb`

Value

A vector of length two, with the Kolmogorov-Smirnov and Anderson-Darling statistics.

Examples

```
twosamplesurvpvs(rexp(20),rbinom(20,1,.5),rbinom(20,1,.5))
```

`twosamplesurvtests` *Two Sample Omnibus Tests of Survival Curves*

Description

Returns the Kolmogorov-Smirnov and Anderson-Darling test statistics for two right-censored data sets.

Usage

```
twosamplesurvtests(times, delta, grp)
```

Arguments

`times`	Event and censoring times
`delta`	Indicator of event (1) or censoring (0).
`grp`	Variable that divides the population into groups.

Value

A vector of length two, with the Kolmogorov-Smirnov and Anderson-Darling statistics.

Examples

```
twosamplesurvpvs(rexp(20),rbinom(20,1,.5),rbinom(20,1,.5))
```

`util.jplot` *Plot a curve, skipping bits where there is a large jump.*

Description

Plot a curve, skipping bits where there is a large jump.

Usage

```
util.jplot(x, y, ...)
```

Arguments

`x`	Ordinates to be plotted.
`y`	Abcissas to be plotted.
`...`	Arguents passed directly to plot.

Index