Package 'pairwiseCI'

May 7, 2018

Type Package

Title Confidence Intervals for Two Sample Comparisons

Version 0.1-26

Date 2018-04-30

Author Frank Schaarschmidt [aut, cre],
Daniel Gerhard [aut]

Maintainer Frank Schaarschmidt <schaarschmidt@biostat.uni-hannover.de>

Depends MCPAN

Imports graphics, stats, coin, boot, MASS, mcprofile

Description Calculation of the parametric, nonparametric confidence intervals
for the difference or ratio of location parameters, nonparametric confidence interval
for the Behrens-Fisher problem and for the difference, ratio and odds-ratio of binomial
proportions for comparison of independent samples. Common wrapper functions to split
data sets and apply confidence intervals or tests to these subsets.
A by-statement allows calculation of CI separately for the levels of further factors.
CI are not adjusted for multiplicity.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2018-05-07 08:42:05 UTC

R topics documented:

`pairwiseCI-package` *Wrapper functions for two-sample confidence intervals and tests.*

Description

A collection of wrapper functions for simple evaluation of factorial trials. The function pairwiseCI allows to calculate 2-sample confidence intervals for all-pairs and many-to-one comparisons between levels of a factor. Intervals are NOT adjusted for multiple hypothesis testing per default. The function pairwiseTest allows to calculate p-values of two-sample tests for all-pairs and many-to-one comparisons between levels of a factor. P-values are NOT adjusted for multiple hypothesis testing per default. Both function allow splitting of the data according to additional factors. Intervals can be plotted, summary.pairwiseTest allows to use the p-value adjustments as implemented in p.adjust(stats). Different test and interval methods (parametric, nonparametric, bootstrap for robust estimators of location, binomial proportions) are implemented in a unified user level function.

Author(s)

Frank Schaarschmidt and Daniel Gerhard, for the Institute of Biostatistics, Leibniz Universitaet Hannover Maintainer: Frank Schaarschmidt <schaarschmidt@biostat.uni-hannover.de>

See Also

Multiple comparisons for the differences of means:**multcomp**

`pairwise.t.test(stats)` `pairwise.prop.test(stats)` `p.adjust(stats)`

Examples

```
# some examples:
# In many cases the shown examples might not make sense,
# but display how the functions can be used.

data(Oats)
Oats

# # all pairwise comparisons,
# separately for each level of nitro:

apc <- pairwiseCI(yield ~ Variety, data=Oats,
 by="nitro", method="Param.diff")

apc

# Intervals can be plotted:

plot(apc)

# See ?pairwiseCI or ?pairwiseCImethodsCont
# for further options for intervals of 2 samples
# of continuous data.


# Or a test

apcTest <- pairwiseTest(yield ~ Variety, data=Oats,
 by="nitro", method="t.test")

# with holm-adjusted p-values:
summary(apcTest, p.adjust.method="holm")



# # If only comparisons to one control would be of interest:
# many to one comparisons, with variety Marvellous as control,
# for each level of nitro separately:

m21 <- pairwiseCI(yield ~ Variety, data=Oats,
 by="nitro", method="Param.diff", control="Marvellous")


##############################################
# # Proportions: another structure of the data is needed.
# Currently the data.frame data must contain two columns,
```

```
# specifying the number of successes and failures on each
# unit.

# The rooting example:
# Calculate confidence intervals for the
# difference of proportions between the 3 doses of IBA,
# separately for 4 combinations of "Age" and "Position".
# Note: we pool over Rep in that way. Whether this makes
# sense or not, is decision of the user.

data(rooting)
rooting

# Confidence intervals for the risk difference

aprootsRD<-pairwiseCI(cbind(root, noroot) ~ IBA,
 data=rooting, by=c("Age", "Position"), method="Prop.diff")

# See ?pairwiseCIProp for further options to compare proportions

# Or a test:

aprootsTest<-pairwiseTest(cbind(root, noroot) ~ IBA,
 data=rooting, by=c("Age", "Position"), method="Prop.test")
aprootsTest

summary(aprootsTest, p.adjust.method="holm")
```

`as.data.frame.pairwiseCI` *Coercing pairwiseCI objects to data.frames*

Description

Creates a data.frame from the output of pairwiseCI.

Usage

```
## S3 method for class 'pairwiseCI'
as.data.frame(x, ...)
```

Arguments

`x`	an object of class "pairwiseCI"
`...`	currently not used

Value

A data.frame with the columns

`estimate`	containing the estimates
`lower`	containing the lower bounds
`upper`	containing the upper bounds
`comparison`	containing character strings, specifying which levels have been compared

and if the argument by has been specified,

`by`	containing the levels by which the original data set has been split

and the conf.level and a character string naming the used method.

See Also

`pairwiseCI`, `summary.pairwiseTest`

Examples

```
data(repellent)

out2<-pairwiseCI(decrease~treatment, data=repellent, control="H",
 alternative="two.sided", method="Param.diff")

out2

as.data.frame(out2)
```

---

as.data.frame.pairwiseMEP

Coerce pairwiseMEP objects to data.frames

Description

Coerces the output of the function pairwiseMEP to a data.frame.

Usage

```
## S3 method for class 'pairwiseMEP'
as.data.frame(x, row.names = NULL,
 optional = FALSE, whichep = NULL, ...)
```

Arguments

x	an object of class "pairwiseMEP" as can be obtained by calling pairwiseMEP
row.names	as in as.data.frame
optional	as in as.data.frame
whichep	a vector of integers or character strings, indexing which endpoints (which response variables) from object x shall be coerced to a data.frame; if omitted (default), all endpoints are coerced to a data.frame
...	Further arguments to be passed to as.data.frame

Value

A data.frame with columns

estimate	numeric, the point estimates
lower	numeric, the lower confidence limits
upper	numeric, the upper confidence limits
comparison	character, the name of the groupwise comparison
by	optional, character, the name of subset of the original data.frame
response	character, the name of the response variable
method	character, the name of the method used for calculation of the lower and upper limits, see pairwiseMEP

---

behenic *Measurements of behenic acid in two samples*

Description

Observations below a detection limit: in field trials with transgenic and isogenic corn varieties the behenic acid content was measured. Objective is to prove equivalence.

Usage

```
data(behenic)
```

Format

A data frame with 12 observations on the following 2 variables.

Treatment a factor with 2 levels: transgenic xisogenic

Behenic a numeric vector giving concentration of behenic acid, where 0.002 is the detection limit

Source

Oberdoerfer, R.B. Example dataset from composition analyses of genetically modified oilseed rape seeds. 2003; BCS GmbH.

Examples

```
data(behenic)
boxplot(Behenic~Treatment, data= behenic)
```

---

cabbage *Yield of cabbage in one-factorial field trial*

Description

Cabbage yield in dependence of four doses of fertilizer.

Usage

```
data(cabbage)
```

Format

A data frame with 20 observations on the following 2 variables.

`Fert` a factor with levels `D1` `D2` `D3` `D4`, the different doses of fertilizer

`yield` a numeric vector, cabbage yield

Examples

```
data(cabbage)
boxplot(yield~Fert, data=cabbage)
```

MOVERR *MOVER-R method by Donner and Zhou (2012)*

Description

Compute confidence intervals for the ratio (theta1/theta0) of two parameters based on point estimates and confidence intervals for the two parameters, theta1 and theta0.

Usage

```
MOVERR(theta0, ci0, theta1, ci1, alternative = "two.sided")
```

Arguments

theta0	a single numeric vale, the point estimate for the parameter to appear in the denominator
ci0	vector with two numeric values, the lower and upper confidence limits for the parameter to appear in the denominator
theta1	a single numeric vale, the point estimate for the parameter to appear in the numerator
ci1	vector with two numeric values, the lower and upper confidence limits for the parameter to appear in the numerator
alternative	a character string, "two.sided" for two-sided intervals, "less" for upper limits, "greater" for lower limits only

Details

This function implements Eq. (9) in Donner and Zou (2012), for computing confidence intervals for a ratio of two parameters, when their estimators are uncorrelated.

Value

a list with elements

conf.int	numeric, the lower and upper confidence limits for the ratio
estimate	the point estimate for the ratio

and the input estimates and confidence limits

Author(s)

Frank Schaarschmidt

References

Donner and Zou (2012): Closed-form confidence intervals for functions of the normal mean and standard deviation. Statistical Methods in Medical Research 21(4):347-359.

---

np.re *Nonparametric test and confidence interval based on relative effects*

Description

Nonparametric test and confidence interval for comparing two samples in presence of heterogeneous variances (Behrens Fisher Problem). The confidence intervals estimate the relative effect p(x,y), i.e. the probability that a randomly chosen observation from population x is larger than a randomly chosen observation from population y, $P(x > y) + P(x = y)/2$. Different approximations are available, the method is designed for continuous or ordered categorical data.

Usage

```
np.re(x, y, conf.level = 0.95,
alternative = c("two.sided", "less", "greater"),
 method = c("logit", "probit", "normal", "t.app"))
```

Arguments

x	a numeric vector, containing the observations of the first sample
y	a numeric vector, containing the observations of the second sample
conf.level	a single numeric value, the desired confidence level of the interval to be constructed.
alternative	a single character string, one of "two.sided" (for two-sided testing and confidence intervals), "less" (for testing that x < y in tendency, or upper confidence limits for p(x,y)), or "greater" (for testing x > y in tendency, or lower confidence limits for p(x,y))
method	a single character string, one of "logit","probit","normal","t.app", specifying which of the different available approximative methods should be used. See details below.

Details

The function performs two sample tests for the nonparametric Behrens-Fisher problem. The hypothesis tested is

$$H_0 : p(x, y) = 1/2$$

, where p(x,y) denotes the relative effect of 2 independent samples x and y. Further, confidence intervals for the relative effect p(x,y) are computed. For the computation of p-values as well as confidence limits, a standard normal or Satterthwaite t-approximation can be used directly on the scale of the relative effects (method "normal","t.app"). Based on these methods, the intervals may have bounds outside [0,1] for extreme results. Alternatively variance stabilising transformations (Probit and Logit) may be used in combination with normal approximation (methods "logit", and "probit").

If the samples are completely separated, the variance estimator is zero by construction. In this case, estimated relative effects 0 or 1 are replaced with 0.001, 0.999 respectively. The variance estimator is replaced as described in Neubert and Brunner (2006).

Value

An object of class "htest"

Author(s)

Frank Konietschke, **nparcomp**, with adaptations by Frank Schaarschmidt

References

Brunner, E., Munzel, U. (2000). The Nonparametric Behrens-Fisher Problem: Asymptotic Theory and a Small Sample Approximation. Biometrical Journal 42, 17 -25. *Neubert, K., Brunner, E., (2006).* A Studentized Permutation Test for the Nonparametric Behrens-Fisher Problem. Computational Statistics and Data Analysis.

Examples

```
data(sodium)

iso<-subset(sodium, Treatment=="xisogenic")$Sodiumcontent
tra<-subset(sodium, Treatment=="transgenic")$Sodiumcontent

np.re(x=iso, y=tra, conf.level = 0.95)
np.re(x=iso, y=tra, conf.level = 0.95, alternative="less")
np.re(x=iso, y=tra, conf.level = 0.95, alternative="greater")
```

Oats *The Oats data set*

Description

The yield of three varieties of Oat was recorded in a field trial with 6 Blocks on 4 levels of nitrogen fertilization. Originally a split plot design, here simply for demonstration of pairwiseCI.

Usage

```
data(Oats)
```

Format

A data frame with 72 observations on the following 4 variables.

`Block` an ordered factor with levels `VI` < `V` < `III` < `IV` < `II` < `I`

`Variety` a factor with levels `Golden Rain` `Marvellous` `Victory`

`nitro` a numeric vector

`yield` a numeric vector

Source

See Oats(nlme).

Examples

```
data(Oats)
boxplot(yield ~ nitro*Variety, data=Oats)
```

`overdispersed.binomial.ratio` *Confidence intervals for risk ratios of overdispersed binomial data*

Description

Calculate approximate confidence intervals for ratios of proportions (risk ratios) of two samples. Three functions are available for intervals assuming the beta binomial distribution, based on a generalized assumption of overdispersion estimated from the residuals, and based on the quasibinomial assumption using a generalized linear model.

Usage

```
Betabin.ratio(x, y, conf.level=0.95, alternative="two.sided",
 CImethod=c("FBB", "LBB"), iccpool=FALSE, resbin=FALSE)

ODbin.ratio(x, y, conf.level=0.95, alternative="two.sided",
 CImethod=c("FOD", "LOD"), varmethod=c("res", "san"), resbin=FALSE)

Quasibin.ratio(x, y, conf.level = 0.95, alternative = "two.sided", grid = NULL)
```

Arguments

`x`	a matrix or data.frame of the first sample, with two columns giving the counts of successes and failures in the two columns; each row should correspond to one experimental or observational unit; first column will be treated as 'success'
`y`	a matrix or data.frame of the second sample, with two columns giving the counts of successes and failures in the two columns; each row should correspond to one experimental or observational unit; first column will be treated as 'success'
`conf.level`	a single numeric value between 0 and 1, the confidence level
`alternative`	a character string, `"two.sided"` for two-sided intervals, `"less"` for upper limits, `"greater"` for lower limits only
`CImethod`	a character string, chossing between available methods for interval computation: in `betabin.ratio`: assuming the beta binomial distribution `"FBB"` invokes the Fieller-Bailey interval and `"LBB"` invokes the delta-method on the log scale (Lui et al. 2000); in `ODbin.ratio`: without particular assumptions w.r.t to the distibution, `"FOD"` invokes the Fieller-Bailey interval and `"LOD"` invokes the delta-method on the log scale as described by Zaihra and Paul (2010)
`iccpool`	logical, if `FALSE`, a separate intra-class-correlation coefficient is estimated for each sample (assuming different levels of overdispersion in the two samples); if `TRUE`, a joint intra-class-correlation coefficient (assuming equal ICCs in the two samples)
`resbin`	logical: if `FALSE`, underdispersion is allowed in estimation; if `TRUE`, underdispersion not allowed: when sample estimates suggest underdispersion, the variance is fixed at the binommial variance

varmethod	a character string specifying te type of variance estimation in ODbin.ratio: "res" is the residual variance, "san" corresponds to a sandwich estimator, details see (Zaihra and Paul, 2010)
grid	optional, a numeric vector to be supplied to the profiling used internally in quasibin.ratio to obtain profile deviance intervals for each samples proportion on the logit-scale.

Details

The methods in betabin.ratio are described by Lui et al., (2000), where different estimates for ICC (and thus overdispersion) are computed for each sample. For small sample size or extreme proportions, one may restrict the variance to that of the binomial distribution and/or use a pooled estimator of ICC for a joint model of overdispersion in the two samples.

The methods in ODbin.ratio are described by Zaihra and Paul (2010), where different estimates for overdispersion are computed for each sample. Zaihra and Paul refer to two different methods of estimating the variance (MR, MS), here referred to as "res", "san", respectively. As above one may restrict the variance to that of the binomial distribution.

The method to compute intervals under the quasibinomial assumptions in quasibin.ratio uses a quasibinomial generalized linear model to obtain profile deviance intervals (e.g. Bates and Watts, 1988; relying on package mcprofile), and then applies the MOVERR method by Donner and Zhou(2012); experimental!

Value

a list with elements

conf.int	confidence limits for the ratio of proprotions in x over that in y
estimate	the point estimate for the ratio of proprotions in x over that in y

Note

The method in quasibin.ratio is experimental.

Author(s)

Frank Schaarschmidt

References

Lui K-L, Mayer JA, Eckhardt L (2000): Confidence intervals for the risk ratio under cluster sampling based on the beta-binomial model. Statistics in Medicine 19, 2933-2942.

Zaihra, T and Paul, S (2010): Interval Estimation of Some Epidemiological Measures of Association. The International Journal of Biostatistics. 6 (1), Article 35.

Bates and Watts(1988): Nonlinear Regression Analysis and Its Applications, Wiley, Ch.6

Donner and Zou (2012): Closed-form confidence intervals for functions of the normal mean and standard deviation. Statistical Methods in Medical Research 21(4):347-359.

Examples

```
# Toxicologoical data: Number of Pups alive four days after birth (16 litters of rats)
# Original source: Weil, 1970: Selection of valid number[...].
#  Food and Cosmetics, Toxicology, 8, 177-182.
# Cited from Zaihra and Paul(2010): Interval Estimation  of Some Epidemiological
# Measures of Association. Int. J Biostatistics, 6(1), Article 35.


mchem = c(12, 11, 10, 9, 11, 10, 10, 9, 9, 5, 9, 7, 10, 6, 10, 7)
xchem = c(12, 11, 10, 9, 10,  9,  9, 8, 8, 4, 7, 4,  5, 3,  3, 0)
dchem <- cbind("alive"=xchem, "dead"=mchem-xchem)

mcon = c(13, 12, 9, 9, 8, 8, 13, 12, 10, 10, 9, 13, 5, 7, 10, 10)
xcon = c(13, 12, 9, 9, 8, 8, 12, 11,  9,  9, 8, 11, 4, 5,  7,  7)
dcon <- cbind("alive"=xcon, "dead"=mcon-xcon)


# Zaihra and Paul report: MR2: [0.714; 1.034]
ODbin.ratio(x=dchem, y=dcon, CImethod="LOD", resbin=FALSE)

# Zaihra and Paul report: MR4: [0.710; 1.029]
ODbin.ratio(x=dchem, y=dcon, CImethod="FOD", resbin=FALSE)



Betabin.ratio(x=dchem, y=dcon, CImethod="FBB", iccpool=TRUE, resbin=TRUE)

Quasibin.ratio(x=dchem, y=dcon)


# Solar protection data: intervention and control group (Mayer, 1997:)

# Number of children with adequate level of solar protection
# Source: Mayer, Slymen, Eckhardt et al.(1997): Reducing ultraviolat
# radiation exposure in children. Preventive Medicine 26, 845-861.
# Cited from: Lui et al. (2000)

mint=c(3,2,2,5,4,3,1,2,2,2,1,3,1,3,2,2,6,2,4,2,2,2,2,1,1,1,1,1,1)
xint=c(1,1,1,0,1,2,1,2,2,1,1,2,1,2,2,0,0,0,0,1,2,1,1,1,1,0,0,0,0)
dint <- cbind("adequate"=xint, "non-adequate"=mint-xint)

mcont=c(2,4,3,2,3,4,4,2,2,3,2,2,4,3,2,3,1,1,2,2,2,3,3,4,1,1,1,1,1)
xcont=c(0,0,2,2,0,4,2,1,1,3,2,1,1,3,2,3,1,0,1,2,1,1,2,4,1,1,1,0,0)
dcont <- cbind("adequate"=xcont, "non-adequate"=mcont-xcont)


# Lui et al.(2000) report for the Fieller-Bailey method
# with pooled ICC: 905% CI = [0.964; 2.281]
Betabin.ratio(x=dcont, y=dint, conf.level=0.95, alternative="two.sided",
 CImethod="FBB", iccpool=TRUE, resbin=FALSE)
```

```
# and for the Log-scale delta method with pooled ICC:
# 95% CI = [0.954; 2.248]
Betabin.ratio(x=dcont, y=dint, conf.level=0.95, alternative="two.sided",
 CImethod="LBB", iccpool=TRUE, resbin=FALSE)

ODbin.ratio(x=dcont, y=dint, conf.level=0.95, alternative="two.sided",
 CImethod="FOD", resbin=TRUE)

Quasibin.ratio(x=dcont, y=dint, conf.level = 0.95, alternative = "two.sided")
```

`pairwiseCI` *Wrapper function for two-sample confidence intervals*

Description

Confidence intervals (CI) for difference or ratio of location parameters of two independent samples. The CI are NOT adjusted for multiplicity by default. A by statement allows for separate calculation of pairwise comparisons according to further factors in the given dataframe. The function applies the intervals available from t.test(stats) for difference of means with and without assumptions of homogeneous variances; large sample approximations for the difference and ratio of means of lognormal data; the exact CI for difference (wilcox.exact(exactRankTests)) and ratio of location based on the Hodges-Lehmann estimator; bootstrap intervalls for ratio and difference of Medians and Harrell-Davis estimators a more robust alternatives to the Hodges-Lehmann estimator (boot, Hmisc); the Score test derived CI for the difference (prop.test(stats)), Woolf-interval for the odds-ratio and a crude asymptotic as well as an iterative interval for the ratio of proportions.

Usage

```
pairwiseCI(formula, data, by = NULL,
 alternative = "two.sided", conf.level = 0.95,
 method = "Param.diff", control = NULL, ...)
```

Arguments

`formula`	A formula of the structure `response ~ treatment` for numerical variables, and of structure `cbind(success, failure) ~ treatment` for binomial variables
`data`	A data.frame containing the numerical response variable and the treatment and by variable as factors. Note, that for binomial data, two columns containing the number of successes and failures must be present in the data.
`by`	A character string or character vector giving the name(s) of additional factors by which the data set is to be split.
`alternative`	Character string, either "two.sided", "less" or "greater"
`conf.level`	The comparisonwise confidence level of the intervals, where 0.95 is default

`method` A character string specifying the confidence interval method, with the following options `"Param.diff"`: Difference of two means, with additional argument `var.equal=FALSE`(default) as in `t.test(stats)`, `"Param.ratio"`: Ratio of two means, with additional argument `var.equal=FALSE`(default) as in `ttestratio(mratios)`, `"Lognorm.diff"`: Difference of two means, assuming a lognormal distribution, based on generalized pivotal quantities, `"Lognorm.ratio"`: Ratio of two means, assuming a lognormal distribution, based on generalized pivotal quantities, `"np.re"`: nonparametric confidence interval for relative effects suitable for the Behrens Fisher problem, `"HL.diff"`: Exact conditional nonparametric CI for difference of locations based on the Hodges-Lehmann estimator, `"HL.ratio"`: Exact conditional nonparametric CI for ratio of locations, based on the Hodges-Lehmann estimator, `"Median.diff"`: Nonparametric CI for difference of locations, based on the medians (percentile bootstrap, stratified by the grouping variable), `"Median.ratio"`: Nonparametric CI for ratio of locations, based on the medians (percentile bootstrap, stratified by the grouping variable), `"Negbin.ratio"`: Profile-likelihood CI for ratio of expected values assuming the negative binomial distribution, adapting code from the profile function of MASS (Venables and Ripley,2002), `"Quasipoisson.ratio"`: Profile-deviance CI for the ratio of expected values with a quasipoisson assumption, using the adapted code of the profile function in MASS `"Poisson.ratio"`: Profile-likelihood CI for the ratio of expected values with a Poisson assumption, again using slightly changed code from MASS, `"Prop.diff"`: Asymptotic CI for the difference of proportions with the following options: `CImethod="NHS"` Newcombes Hybrid Score interval (Newcombe, 1998), `CImethod="CC"` continuity corrected interval (Newcombe, 1998) as implemented in `prop.test(stats)`, `CImethod="AC"` Agresti-Caffo interval (Agresti and Caffo, 2000) `"Prop.ratio"`: Asymptotic CI (normal approximation on the log: `CImethod="GNC"`) or iterative CI (`CImethod="Score"`) for ratio of proportions, `"Prop.or"`: Asymptotic CI for the odds ratio (normal approximation on the log: `CImethod="Woolf"`), or the exact CI corresponding to Fishers exact test (`CImethod="Exact"`, taken from `fisher.test`, package `stats`) See `?pairwiseCImethods` for details.

`control` Character string, specifying one of the levels of the treatment variable as control group in the comparisons; default is NULL, then CI for all pairwise comparisons are calculated.

`...` further arguments to be passed to the functions specified in `pairwiseCImethodsCont`, `pairwiseCImethodsCount` and `pairwiseCImethodsProp`. In particular, different methods to calculate confidence intervals can be specified via the argument `CImethod` for the comparison of binomial proportions, see details in `pairwiseCImethodsProp`.

Details

Note that all the computed intervals are without adjustment for multiplicity by default. When based on crude normal approximations or crude non-parametric bootstrap methods, the derived confidence intervals can be unreliable for small sample sizes. The method implemented here split the data set into small subsets (according to the grouping variable in `formula` and the variable specified in `by`) and compute confidence intervals only based each particular subset. Please note that, when a (generalized) linear model can be reasonable assumed to describe the complete data set, it is

smarter to compute confidence intervals based on the model estimators. Methods to do this are, e.g., implemented in the packages **stats**, **multcomp**, **mratios**.

Value

an object of class "pairwiseCI" structured as:

a list containing:

`byout`	A named list, ordered by the levels of the by-factor, where each element is a list containing the numeric vectors `estimate`, `lower`, `upper` and the character vector `compnames`
`bynames`	the level names of the by-factor

and further elements as input, try str() for details.

the object is printed by `print.pairwiseCI`.

References

- `Param.diff` simply uses: `t.test` in **stats**, note that the default setting assums possibly heterogeneous variances (`var.equal=FALSE`).
- `Param.ratio` for homogeneous variances (`var.equal=TRUE`): **Fieller EC (1954):** Some problems in interval estimation. Journal of the Royal Statistical Society, Series B, 16, 175-185.
- `Param.ratio` for heterogenous variances (`var.equal=FALSE`): : the test proposed in: **Tamhane, AC, Logan, BR (2004):** Finding the maximum safe dose level for heteroscedastic data. Journal of Biopharmaceutical Statistics 14, 843-856. is inverted by solving a quadratic equation according to Fieller, where the estimated ratio is simply plugged in order to get Satterthwaite approximated degrees of freedom. See also: **Hasler, M, Vonk, R, Hothorn, LA:** Assessing non-inferiority of a new treatment in a three arm trial in the presence of heteroscedasticity. Statistics in Medicine 2007 (in press).
- `Lognorm.ratio` and `Lognorm.ratio`: **Chen, Y-H, Zhou, X-H (2006)**: Interval estimates for the ratio and the difference of two lognormal means. Statistics in Medicine 25, 4099-4113.
- `Np.re`: **Brunner, E., Munzel, U. (2000).** The Nonparametric Behrens-Fisher Problem: Asymptotic Theory and a Small Sample Approximation. Biometrical Journal 42, 17 -25. **Neubert, K., Brunner, E., (2006).** A Studentized Permutation Test for the Nonparametric Behrens-Fisher Problem. Computational Statistics and Data Analysis.
- `HL.diff`: `wilcox.exact` in **exactRankTests**
- `HL.ratio`: **Hothorn, T, Munzel, U:** Non-parametric confidence interval for the ratio. Report University of Erlangen, Department Medical Statistics 2002; available via: http://www.imbe.med.uni-erlangen.de/~hothorn/ For the Hodges-Lehmann estimator see: **Hodges, J.L., Lehmann E.L.(1963):** Estimates of location based on rank tests. Ann. Math. Statist. 34, 598-611.
- `Median.diff`: The interval is calculated from a bootstrap sample (stratified according to the group variable) of the difference of sample Medians, using package boot.
- `Median.ratio`: The interval is calculated from a bootstrap sample (stratified according to the group variable) of the ratio of sample Medians, using package boot.

  Note, that the 4 bootstrap versions will hardly make sense for small samples, because of the discreteness of the resulting bootstrap sample.

- `Poisson.ratio`: Profile-likelihood CI, based on the assumption of a Poisson distribution, basic method described in **Venables, W.N., Ripley, B.D. (2002):** Modern Applied Statistics with S. Springer Verlag, New York.
- `Quasipoisson.ratio`: Profile-deviance CI, based on the quasipoisson assumption, basic method described in Venables, **W.N., Ripley, B.D. (2002):** Modern Applied Statistics with S. Springer Verlag, New York.
- `Negbin.ratio`: Profile-likelihood CI, based on the assumption of the negative binomial distribution, basic method described in **Venables, W.N., Ripley, B.D. (2002):** Modern Applied Statistics with S. Springer Verlag, New York.
- `Prop.diff`: provides three approximate methods to calculate confidence intervals for the difference of proportions: Default is `CImethod="NHS"`: Newcombes Hybrid Score interval (Newcombe, 1998), other options are `CImethod="CC"` continuity corrected interval (Newcombe, 1998) as implemented in `prop.test(stats)` and `CImethod="AC"` Agresti-Caffo interval (Agresti and Caffo, 2000) **Newcombe R.G. (1998):** Interval Estimation for the Difference Between Independent Proportions: Comparison of Eleven Methods. Statistics in Medicine 17, 873-890.
- `Prop.ratio` calculates the crude asymptotic interval (normal approximation on the log-scale, `CImethod="GNC"`) or the Score interval (`CImethod="Score"`), both described in: **Gart, JJ and Nam, J (1988)**: Approximate interval estimation of the ratio of binomial parameters: A review and corrections for skewness. Biometrics 44, 323-338.
- `Prop.or` Adjusted Woolf interval (`CImethod="Woolf"`), e.g. in: **Lawson, R (2005):** Small sample confidence intervals for the odds ratio. Communication in Statistics Simulation and Computation, 33, 1095-1113. or the exact interval corresponding to Fishers exact test (`CImethod="Exact"`, taken from `fisher.test(stats)`, see references there)

See Also

`as.data.frame.pairwiseCI` to create a data.frame from the output, `summary.pairwiseCI` to create a more easily accessable list from the output, `plot.pairwiseCI` to plot the intervals. Further, see **multcomp** for simultaneous intervals for difference for various contrasts, **mratios** for simultaneous intervals for the ratio of means, **stats** `p.adjust`, `pairwise.t.test`, `pairwise.prop.test`, `pairwise.wilcox.test`, `TukeyHSD` for methods of multiple comparisons.

Examples

```
# some examples:
# In many cases the shown examples might not make sense,
# but display how the functions can be used.

data(Oats)

# # all pairwise comparisons,
# separately for each level of nitro:

apc <- pairwiseCI(yield ~ Variety, data=Oats,
 by="nitro", method="Param.diff")

apc
```

```
plot(apc)

# # many to one comparisons, with variety Marvellous as control,
# for each level of nitro separately:

m21 <- pairwiseCI(yield ~ Variety, data=Oats,
 by="nitro", method="Param.diff", control="Marvellous")

 plot(m21)

# # the same using confidence intervals for the ratio of means:

m21 <- pairwiseCI(yield ~ Variety, data=Oats,
 by="nitro", method="Param.diff", control="Marvellous")

 plot(m21, CIvert=TRUE, H0line=0.9)

################################################

# The repellent data set (a trial on repellent
# effect of sulphur on honey bees): Measured was
# the decrease of sugar solutions (the higher the decrease,
# the higher the feeding, and the less the repellent effect).
# Homogeneity of variances is questionable. Which of the doses
# leads to decrease of the variable decrease compared to the
# control group "H"?

data(repellent)
boxplot(decrease ~ treatment, data=repellent)

# as difference to control (corresponding to Welch tests)
beeCId<-pairwiseCI(decrease ~ treatment, data=repellent,
 method="Param.diff", control="H", alternative="less",
 var.equal=FALSE)
beeCId
plot(beeCId)

# as ratio to control:

## Not run:
beeCIr<-pairwiseCI(decrease ~ treatment, data=repellent,
 method="Param.ratio", control="H", alternative="less",
 var.equal=FALSE)
beeCIr
plot(beeCIr)

# Bonferroni-adjustment can be applied:

beeCIrBonf<-pairwiseCI(decrease ~ treatment, data=repellent,
 method="Param.ratio", control="H", alternative="less",
 var.equal=FALSE, conf.level=1-0.05/7)
beeCIrBonf
```

```
plot(beeCIrBonf)

## End(Not run)

##############################################

# Proportions:

# The rooting example:
# Calculate confidence intervals for the
# difference of proportions between the 3 doses of IBA,
# separately for 4 combinations of "Age" and "Position".
# Note: we pool over Rep in that way. Whether this makes
# sense or not, is decision of the user.

data(rooting)

# Risk difference

aprootsRD<-pairwiseCI(cbind(root, noroot) ~ IBA,
 data=rooting, by=c("Age", "Position"), method="Prop.diff")

aprootsRD

# Odds ratio


aprootsOR<-pairwiseCI(cbind(root, noroot) ~ IBA,
 data=rooting, by=c("Age", "Position"), method="Prop.or")

aprootsOR

# Risk ratio

aprootsRR<-pairwiseCI(cbind(root, noroot) ~ IBA,
 data=rooting, by=c("Age", "Position"), method="Prop.ratio")

aprootsRR

# CI can be plotted:

plot(aprootsRR)

##############################################

# CIs assuming lognormal distribution of the response:

resp<-rlnorm(n=20, meanlog = 0, sdlog = 1)
treat<-as.factor(rep(c("A","B")))
datln<-data.frame(resp=resp, treat=treat)

pairwiseCI(resp~treat, data=datln, method="Lognorm.diff")
pairwiseCI(resp~treat, data=datln,  method="Lognorm.ratio")
```

pairwiseCIInt *Internal functions for pairwiseCI*

Description

For internal use. Two different methods for data representable as a two numeric vectors (pairwiseCICont) and data representable as matrix with two columns like `cbind(successes, failures)`. Functions that split up a data.frame according to one factor, and perform all pairwise comparisons and comparisons to control among the levels of the factor by calling methods documented in `pairwiseCImethodsCont` and `pairwiseCImethodsProp`.

Usage

```
pairwiseCICont(formula, data, alternative="two.sided",
 conf.level=0.95, method, control=NULL, ...)

pairwiseCIProp(formula, data, alternative="two.sided",
 conf.level=0.95, control=NULL, method, ...)
```

Arguments

`formula`	A formula of the structure `response ~ treatment` for numerical variables, and of structure `cbind(success, failure) ~ treatment` for binomial variables
`data`	A data.frame containing the numerical response variable and the treatment and by variable as factors. Note, that for binomial data, two columns containing the number of successes and failures must be present in the data.
`alternative`	Character string, either `"two.sided"`, `"less"` or `"greater"`
`conf.level`	The *comparisonwise confidence level* of the intervals, where 0.95 is default
`method`	A character string specifying the confidence interval method, one of the following options `"Param.diff"`: Difference of two means, with additional argument `var.equal=FALSE`(default) as in `t.test(stats)` `"Param.ratio"`: Ratio of two means, with additional argument `var.equal=FALSE`(default) as in `ttestratio(mratios)` `"Lognorm.diff"`: Difference of two means, assuming a lognormal distribution, `"Lognorm.ratio"`: Ratio of two means, assuming a lognormal distribution, `"HL.diff"`: Exact nonparametric CI for difference of locations based on the Hodges-Lehmann estimator, `"HL.ratio"`: Exact nonparametric CI for ratio of locations, based on the Hodges-Lehmann estimator, `"Median.diff"`: Nonparametric CI for difference of locations, based on the medians (percentile bootstrap CI), `"Median.ratio"`: Nonparametric CI for ratio of locations, based on the medians (percentile bootstrap CI), `"Prop.diff"`: Asymptotic CI for difference of proportions `prop.test(stats)` `"Prop.ratio"`: Asymptotic CI for ratio of proportions `"Prop.or"`: Asymptotic CI for the odds ratio See `?pairwiseCImethods` for details.

control	Character string, specifying one of the levels of the treatment variable as control group in the comparisons; default is NULL, then CI for all pairwise comparisons are calculated.
...	further arguments to be passed to the functions specified in methods

Details

These functions are for internal use in pairwiseCI.

Value

a list containing:

estimate	numeric vector: the point estimates
lower	numeric vector: lower confidence bounds
upper	numeric vector: upper confidence bounds
compnames	character vector with the names of comparisons

See Also

pairwiseCI for the user level function; pairwiseCImethodsCont, and pairwiseCImethodsProp for a more detailed documentation of the implemented methods; summary.pairwiseCI for a summary function.

t.test(stats), wilcox.exact(exactRankTests), prop.test(stats) for the sources of some of the CI methods, **multcomp** for simultaneous intervals for difference for various contrasts, **mratios** for simultaneous intervals for the ratio in many-to-one comparisons

---

pairwiseCImethodsCont *Confidence intervals for two sample comparisons of continuous data*

Description

Confidence interval methods available for pairwiseCI for comparison of two independent samples. Methods for continuous variables.

Usage

```
Param.diff(x, y, conf.level=0.95, alternative="two.sided", ...)
Param.ratio(x, y, conf.level=0.95, alternative="two.sided", ...)

Lognorm.diff(x, y, conf.level=0.95, alternative="two.sided", sim=10000, ...)
Lognorm.ratio(x, y, conf.level=0.95, alternative="two.sided", sim=10000, ...)

HL.diff(x, y, conf.level=0.95, alternative="two.sided", ...)
HL.ratio(x, y, conf.level=0.95, alternative="two.sided", ...)
```

```
Median.diff(x, y, conf.level=0.95, alternative="two.sided", ...)
Median.ratio(x, y, conf.level=0.95, alternative="two.sided", ...)
```

Arguments

x	vector of observations in the first sample
y	vector of observations in the second sample
alternative	character string, either "two.sided", "less" or "greater"
conf.level	the comparisonwise confidence level of the intervals, where 0.95 is default
sim	a single integer value, specifying the number of samples to be drawn for calculation of the empirical distribution of the generalized pivotal quantities
...	further arguments to be passed to the individual methods, see details

Details

- Param.diff calculates the confidence interval for the difference in means of two Gaussian samples by calling t.test in package **stats**, assuming homogeneous variances if var.equal=TRUE, and heterogeneous variances if var.equal=FALSE (default);
- Param.ratio calculates the Fiellers (1954) confidence interval for the ratio of two Gaussian samples by calling ttestratio in package **mratios**, assuming homogeneous variances if var.equal=TRUE. If heterogeneous variances are assumed (setting var.equal=FALSE, the default), the test by Tamhane and Logan (2004) is inverted by solving a quadratic equation according to Fieller, where the estimated ratio is simply plugged in order to get Satterthwaite approximated degrees of freedom. See Hasler and Vonk (2006) for some simulation results.
- Lognorm.diff calculates the confidence interval for the difference in means of two Lognormal samples, based on general pivotal quantities (Chen and Zhou, 2006); currently, further arguments (...) are not used;
- Lognorm.ratio calculates the confidence interval for the ratio in means of two Lognormal samples, based on general pivotal quantities (Chen and Zhou, 2006); currently, further arguments (...) are not used;
- HL.diff calculates the Hodges-Lehmann confidence interval for the difference of locations by calling wilcox_test in package **coin**, further arguments ... are passed to wilcox_test and corresponding methods for Independence problems, for example distribution may be used to switch from exact (default), to approximate or asymptotic versions;
- HL.ratio calculates a Hodges-Lehmann-like confidence interval for the ratio of locations for positive data by calling wilcox_test in package **coin** on the logarithms of observations and backtransforming (Hothorn and Munzel, 2002), further arguments ... are passed to wilcox_test and corresponding methods for Independence problems, for example distribution may be used to switch from exact (default), to approximate or asymptotic versions;
- Median.diff calculates a percentile bootstrap confidence interval for the difference of Medians using boot.ci in package **boot**, the number of bootstrap replications can be set via R=999 (default);
- Median.ratio calculates a percentile bootstrap confidence interval for the ratio of Medians using boot.ci in package **boot**, the number of bootstrap replications can be set via R=999 (default);

Value

A list containing:

`conf.int`	a vector containing the lower and upper confidence limit
`estimate`	a single named value

References

- `Param.diff` uses `t.test` in **stats**.
- **Fieller EC (1954)**: Some problems in interval estimation. Journal of the Royal Statistical Society, Series B, 16, 175-185.
- **Tamhane, AC, Logan, BR (2004)**: Finding the maximum safe dose level for heteroscedastic data. Journal of Biopharmaceutical Statistics 14, 843-856.
- **Hasler, M, Vonk, R, Hothorn, LA**: Assessing non-inferiority of a new treatment in a three arm trial in the presence of heteroscedasticity (submitted).
- **Chen, Y-H, Zhou, X-H (2006)**: Interval estimates for the ratio and the difference of two lognormal means. Statistics in Medicine 25, 4099-4113.
- **Hothorn, T, Munzel, U**: Exact Nonparametric Confidence Interval for the Ratio of Medians. Technical Report, Universitaet Erlangen-Nuernberg, Institut fuer Medizininformatik, Biometrie und Epidemiologie, 2002; available via: `http://www.statistik.uni-muenchen.de/~hothorn/bib/TH_TR_bib.`

Examples

```
data(sodium)

iso<-subset(sodium, Treatment=="xisogenic")$Sodiumcontent
trans<-subset(sodium, Treatment=="transgenic")$Sodiumcontent

iso
trans

## CI for the difference of means,
# assuming normal errors and homogeneous variances :

thomo<-Param.diff(x=iso, y=trans, var.equal=TRUE)

# allowing heterogeneous variances
thetero<-Param.diff(x=iso, y=trans, var.equal=FALSE)


## Fieller CIs for the ratio of means,
# also assuming normal errors:

Fielhomo<-Param.ratio(x=iso, y=trans, var.equal=TRUE)

# allowing heterogeneous variances
```

```
Fielhetero<-Param.ratio(x=iso, y=trans, var.equal=FALSE)


HLD<-HL.diff(x=iso, y=trans)

thomo
thetero

Fielhomo
Fielhetero

HLD


# # #

# Lognormal CIs:

x<-rlnorm(n=10, meanlog=0, sdlog=1)
y<-rlnorm(n=10, meanlog=0, sdlog=1)

Lognorm.diff(x=x, y=y)
Lognorm.ratio(x=x, y=y)
```

`pairwiseCImethodsCount` *Confidence intervals for two sample comparisons of count data*

Description

Confidence interval methods available for pairwiseCI for comparison of two independent samples. Methods for count data.

Usage

```
Poisson.ratio(x, y, conf.level=0.95, alternative="two.sided")
Quasipoisson.ratio(x, y, conf.level=0.95, alternative="two.sided")
Negbin.ratio(x, y, conf.level=0.95, alternative="two.sided")
```

Arguments

`x`	vector of observations in the first sample
`y`	vector of observations in the second sample
`alternative`	character string, either "two.sided", "less" or "greater"
`conf.level`	the comparisonwise confidence level of the intervals, where 0.95 is default

Details

- `Poisson.ratio` calculates a confidence interval for the ratio of means assuming the Poisson distribution of the response by fitting a generalized linear model with log-link using `glm` in package **stats**, constructing a likelihood profile and deriving a equal-tailed confidence interval from this profile. Please not that confidence intervals from this method produce severely misleading results, when there is extra-Poisson variation in the data.
- `Quasipoisson.ratio` calculates a confidence interval for the ratio of means of the response by fitting a generalized linear model with family `quasipoisson` and log-link using `glm` in package **stats**, constructing a deviance profile and deriving a equal-tailed confidence interval from this profile.
- `Negbin.ratio` calculates a confidence interval for the ratio of means assuming the negative binomial distribution of the response by fitting a generalized linear model with log-link using `glm.nb` in package **MASS**, constructing a likelihood profile and deriving a equal-tailed confidence interval from this profile.

Note, that for all the methods, a separate glm is fitted for each two-sample comparison! When a common model can be reasonbly assumed for all the data, there are smarter methods of constructing confidence intervals for groupwise comparisons, based on a common model, see e.g. the function `confint` in package **stats**, the function `confint.glm` in package **MASS** and the function `confint.glht` in package **multcomp**.

Note, that the code used here is slightly changed from the original code by Venables and Ripley, or Bates and Watts. An limit is imposed on the parameter space in which the profile is constructed. By that limitation, intervals can also be constructed for extreme cases with all observations in one group being zero.

Note, that the `Poisson.ratio` can be used when only one count is present in each group. For `Quasipoisson.ratio`, `Negbin.ratio`, repeated observations are necessary in each group.

Value

A list containing:

`conf.int`	a vector containing the lower and upper confidence limit
`estimate`	a single named value

Author(s)

Daniel Gerhard, Frank Schaarschmidt

References

Venables WN and Ripley BD (2002). Modern Applied Statistics using S, Fourth Edition. Springer New York. **Bates, D.M. and Watts, D.G.(1988).** Nonlinear Regression Analysis and Its Applications. John Wiley and Sons, New York.

Examples

```
df <- data.frame(count = rpois(n=20, lambda=5), treat=rep(LETTERS[1:4], each=5))

QPCI<-pairwiseCI(count ~ treat, data=df,
 alternative="two.sided", control="A", method="Quasipoisson.ratio")

QPCI
```

`pairwiseCImethodsProp` *Confidence intervals for two sample comparisons of binomial proportions*

Description

For the comparison of two independent samples of binomial observations, confidence intervals for the difference (RD), ratio (RR) and odds ratio (OR) of proportions are implemented.

Usage

```
Prop.diff(x, y, conf.level=0.95, alternative="two.sided",
 CImethod=c("NHS", "CC", "AC"), ...)

Prop.ratio(x, y, conf.level=0.95, alternative="two.sided",
 CImethod=c("Score", "MNScore", "MOVER", "GNC"))

Prop.or(x, y, conf.level=0.95, alternative="two.sided",
 CImethod=c("Exact", "Woolf"), ...)
```

Arguments

`x`	observations of the first sample: either a vector with number of successes and failures, or a data.frame with two columns (the successes and failures))
`y`	observations of the second sample: either a vector with number of successes and failures, or a data.frame with two columns (the successes and failures))
`alternative`	character string, either "two.sided", "less" or "greater"
`conf.level`	the comparisonwise confidence level of the intervals, where 0.95 is the default
`CImethod`	a single character string, see below for details
`...`	further arguments to be passed to the individual methods, see details

Details

Generally, the input are two vectors `x` and `y` giving the number of successes and failures in the two samples, or, alternatively, two `data.frames` `x` and `y` each containing one column for the successes and one column for the failures, and the rows containing repeated observations from the same treatment.

Please note, that the confidence intervals available in this function assume counts of successes and failures from a binomial distribution and thus do NOT APPROPRIATELY account for extra-binomial variability between repeated observations for the same treatment! When there are repeated observations (input as a data.frame with several rows), intervals are calculated based on the sums over the rows of success and failure!

The following methods are available for the risk difference:

- `Prop.diff`: asymptotic continuity corrected confidence interval for the difference of proportions by calling `prop.test` in package **stats**, see `?prop.test` for details,
- `Prop.diff` with `CImethod="AC"`: asymptotic Wald-type interval after adding one pseudo-observation to each cell (Agresti and Caffo, 2000).
- `Prop.diff` with `CImethod="NHS"`: asymptotic Newcombes Hybrid Score Interval (Newcombe, 1998).

For the risk ratio:

- `Prop.ratio` with `CImethod="Score"`: asymptotic Score interval for the ratio of proportions (by iteratively inverting a Chi-Squared test) according to Koopman (1984), following the description by Gart and Nam (1988). This method does NOT involve the skewness correction or extensions to stratification described in Gart and Nam (1988).
- `Prop.ratio` with `CImethod="MNScore"`: asymptotic Score interval for the ratio of proportions (by iteratively inverting a Chi-Squared test) according to Miettinen and Nurminen (1985), following the description by Gart and Nam (1988).
- `Prop.ratio` with `CImethod="MOVER"`: asymptotic method of variance estimate recovery for ratios (Donner and Zou, 2012; Fagerland and Newcombe, 2013), relying on the Wilson Score interval to obtain the confidence limits for the single proportions.
- `Prop.ratio` with `CImethod="GNC"`: crude normal approximation on the log-scale after adding 0.5 to the observed number of successes and the samples sizes.

For the odds ratio:

- `Prop.or` with `CImethod="Woolf"` asymptotic adjusted Woolf confidence interval for the odds ratio of proportions (normal approximation after adding 0.5 to each cell count), as, e.g., described in Lawson (2005).
- `Prop.or` with `CImethod="Exact"` calculates the exact confidence interval for the odds ratio of proportions corresponding to Fishers exact test, by calling to `fisher.test` in **stats**. For details, see `?fisher.test`.

Value

A list containing:

conf.int	a vector containing the lower and upper confidence limit, and the methods name as an attribute
estimate	a single named value
quantile	the quantile used for constructing the interval
conf.level	the confidence level

References

- **Agresti A and Caffo B (2000):** Simple and effective confidence intervals for proportions and differences of proportions result from adding two successes and two failures. American Statistician 54 (4), 280-288.
- **Donner A and Zou GY (2012):** Closed-form confidence intervals for functions of the normal mean and standard deviation.Statistical Methods in Medical Research 21(4):347-359.
- **Fagerland MW, Newcombe RG (2013):** Confidence intervals for odds ratio and relative risk based on the inverse hyperbolic sine transformation. Statistics in Medicine, DOI: 10.1002/sim.5714
- **Gart JJ and Nam J (1988)**: Approximate interval estimation of the ratio of binomial parameters: A review and corrections for skewness. Biometrics 44, 323-338.
- **Koopman, PAR (1984):** Confidence Intervals for the Ratio of Two Binomial Proportions. Biometrics 40(2), 513-517.
- **Lawson R (2005)**: Small sample confidence intervals for the odds ratio. Communication in Statistics Simulation and Computation, 33, 1095-1113.
- **Miettinen O and Nurminen M (1985):** Comparative Analysis of Two Rates. Statistics in Medicine 4, 213-226.
- **Newcombe RG (1998):** Interval Estimation for the Difference Between Independent Proportions: Comparison of Eleven Methods. Statistics in Medicine 17, 873-890.
- **Venables WN and Ripley BD (2002). Modern Applied Statistics with S. Fourth Edition. Springer New York.**

See Also

An alternative implementation of the Score interval for the risk ratio in package `propCIs`, function `riskscoreci`.

Examples

```
# The rooting data.

data(rooting)

# the first comparison should be the same as:

Age5_PosB_IBA0 <- subset(rooting,
 Age=="5" & Position=="B" & IBA=="0")[,c("root", "noroot")]
Age5_PosB_IBA0.5 <- subset(rooting,
 Age=="5" & Position=="B" & IBA=="0.5")[,c("root", "noroot")]
```

```
Age5_PosB_IBA0
Age5_PosB_IBA0.5

Prop.diff(x=Age5_PosB_IBA0, y=Age5_PosB_IBA0.5)

Prop.ratio(x=Age5_PosB_IBA0, y=Age5_PosB_IBA0.5)

Prop.or(x=Age5_PosB_IBA0, y=Age5_PosB_IBA0.5)

# is the same as input two vectors x,y each containing
# the count of successes and the count of failures

 colSums(Age5_PosB_IBA0)
 colSums(Age5_PosB_IBA0.5)

Prop.diff(x=c(16,32),y=c(29,19))

Prop.ratio(x=c(16,32),y=c(29,19))

Prop.or(x=c(16,32),y=c(29,19))

# # #

# Comparison with original publications:

# I. Risk difference:

# Continuity corrected interval:

# 1.Comparison with results presented in Newcombe (1998),
# Table II, page 877, 10. Score, CC
# column 1 (a): 56/70-48/80: [0.0441; 0.3559]

Prop.diff(x=c(56,70-56),y=c(48,80-48), alternative="two.sided",
 conf.level=0.95, CImethod="CC")


# Risk difference, NHS
# Newcombes Hybrid Score interval:

# 1.Comparison with results presented in Newcombe (1998),
# Table II, page 877, 10. Score, noCC
# column 1 (a): 56/70-48/80: [0.0524; 0.3339]


Prop.diff(x=c(56,70-56),y=c(48,80-48), alternative="two.sided",
 conf.level=0.95, CImethod="NHS")

Prop.diff(x=c(56,70-56),y=c(48,80-48), alternative="greater",
 conf.level=0.975, CImethod="NHS")

Prop.diff(x=c(56,70-56),y=c(48,80-48), alternative="less",
 conf.level=0.975, CImethod="NHS")
```

```
# 2.Comparison with results presented in Newcombe (1998),
# Table II, page 877, 10. Score, noCC
# column 2 (b): 9/10-3/10: [0.1705; 0.8090]

Prop.diff(x=c(9,1),y=c(3,7), alternative="two.sided",
 conf.level=0.95, CImethod="NHS")


# 3.Comparison with results presented in Newcombe (1998),
# Table II, page 877, 10. Score, noCC
# column 2 (h): 10/10-0/10: [0.6075; 1.000]

Prop.diff(x=c(10,0),y=c(0,10), alternative="two.sided",
 conf.level=0.95, CImethod="NHS")


# II. Risk ratio,
# Score interval according to Koopman(1984), Gart and Nam (1988)

# 1.Comparison with results presented in Gart and Nam (1998),
# Section 5 (page 327), Example 1
# x1/n1=8/15 x0/n0=4/15:
# Log: [0.768, 4.65]
# Score: [0.815; 5.34]

# Log (GNC)
Prop.ratio(x=c(8,7),y=c(4,11), alternative="two.sided",
 conf.level=0.95, CImethod="GNC")

# Score (Score)
Prop.ratio(x=c(8,7),y=c(4,11), alternative="two.sided",
 conf.level=0.95, CImethod="Score")

Prop.ratio(x=c(8,7),y=c(4,11), alternative="less",
 conf.level=0.975, CImethod="Score")

Prop.ratio(x=c(8,7),y=c(4,11), alternative="greater",
 conf.level=0.975, CImethod="Score")



# 2.Comparison with results presented in Gart and Nam (1998),
# Section 5 (page 328), Example 2
# x1/n1=6/10 x0/n0=6/20:
# Crude Log: [0.883, 4.32]
# Score: [0.844; 4.59]

Prop.ratio(x=c(6,4),y=c(6,14), alternative="two.sided",
 conf.level=0.95, CImethod="GNC")

Prop.ratio(x=c(6,4),y=c(6,14), alternative="two.sided",
```

```
 conf.level=0.95, CImethod="Score")


# Koopman (1984), page 517
# x1=36, n1=40, x0=16, n0=80: [2.94, 7.15]

Prop.ratio(x=c(36, 4), y=c(16, 64), CImethod="Score")$conf.int


# Miettinen, Nurminen (1985) p. 217 (Example 6):
# x1=10, n1=10, x0=20, n0=20: [0.72, 1.20]

Prop.ratio(x=c(10, 0), y=c(20, 0), CImethod="MNScore")$conf.int


# MOVER-R Wilson in Newcombe and Fagerland, 2013, Table VIII:
#  x1=24, n1=73,x0=53, n0=65: [0.282, 0.563]
Prop.ratio(x=c(24, 49), y=c(53, 12), CImethod="MOVER")$conf.int

#  x1=29, n1=55, x0=11, n0=11: [0.398, 0.761]
Prop.ratio(x=c(29, 26), y=c(11,0), CImethod="MOVER")$conf.int

#  x1=7, n1=18, x0=1, n0=18: [(1.27, 40.8)]
Prop.ratio(x=c(7, 11), y=c(1, 17), CImethod="MOVER")$conf.int
```

`pairwiseMEP` *Wrapper to compute confidence intervals for multiple endpoints*

Description

This is a test version! Computes confidence intervals for pair wise comparisons of groups (assuming independent observations) for multiple endpoints. The methods available in pairwiseCI for continuous and count data can be called. Methods for binary data are currently not available. NOTE: Although multiple endpoints and multiple group wise comparisons are considered, there is no adjustment for multiplicity implemented in this function!

Usage

```
pairwiseMEP(x, ...)

## S3 method for class 'data.frame'
pairwiseMEP(x, ep, f,
 control = NULL, conf.level = 0.95,
 alternative = c("two.sided", "less", "greater"),
 method = "Param.diff", ...)
```

Arguments

x	a data.frame
ep	a vector of character strings, naming the variables in x which are the response variables (endpoints) of interest
f	a single character string, naming a factor variable in data which splits the dataset into treatment groups
control	optionally, a single character string, naming a factor level in variable f, which shall be considered as control group; if omitted (default) all pairwise comparisons are computed
conf.level	a single numeric between 0.5 and 1, specifying the local confidence level of the single confidence intervals
alternative	a single character string, one of 'two.sided', 'less', 'greater'
method	a vector of character strings, specifying the method for computation of the confidence intervals, see pairwiseCImethodsCont and pairwiseCImethodsCount for possible options; must have length 1 or the same length as ep!
...	further arguments to be passed to pairwiseCI, options are listed in pairwiseCImethodsCont and pairwiseCImethodsCount

Details

Calls pairwiseCI.

Value

conf.int	a list with one element for each element in ep, containing the estimates, lower and upper limits and the comparison names and by levels in the format of a data.frame
data	as input x
ep	as input
f	as input
control	as input
conf.level	as input
alternative	as input
method	as input

See Also

The result can be plotted: plotCI.pairwiseMEP, and coerced to a data.frame: as.data.frame.pairwiseMEP

Examples

```
x1<-rnorm(80,100,8)
x2<-rnbinom(80,mu=10, size=10)
A<-rep(c("a1","a2"), c(40,40))
B<-rep(rep(c("b1","b2"), c(20,20)), times=2)
dat<-data.frame(x1=x1,x2=x2,A=A, B=B)

test<-pairwiseMEP(x=dat, ep=c("x1","x2"), control="a1",
 f="A", by="B", method=c("Param.ratio","Negbin.ratio"))
test


plotCI(test, whichep=c("x1","x2"))

as.data.frame(test, whichep=c(1,2))

as.data.frame(test, whichep=c("x1","x2"))
```

`pairwiseTest` *Wrapper to calculate unadjusted p-values for pairwise comparisons*

Description

Calculation of raw p-values for pairwise comparisons of several groups. The data can be split by additional factors. Any test function can be used, that takes two samples x,y as input and returns a list containing the p.value in an element named `p.value`. The output of this function might be further processed using `p.adjust` in order to adjust for multiple comparisons.

Usage

```
pairwiseTest(formula, data, by = NULL,
 method = "t.test", control = NULL, ...)
```

Arguments

`formula`	a formula specifiying the response and the factor variable: `response ~ factor`
`data`	a data frame, containing the variables specified in `formula`
`by`	optional vector of character strings, defining factors by which to split the data set. Then, pairwise comparisons are performed separately for each level of the specified factors.
`method`	character string, giving the name of the function, which shall be used to calculate local p-values. Any function, taking two vectors x, and y as first arguments and returning a list with the p.value in a list element named `p.value` can be specified.

control	optional character string, defining the name of a control group. Must be one of the levels of the factor variable defined in formula. By default control=NULL, then all pairwise comparisons between the levels of the factor variable are computed.
...	Arguments to be passed the function defined in method

Details

This function splits the response variable according to the factor(s) specified in by, and within each subset according to the grouping variable specified in formula. The function specified in method is called to calculate a p.value for all pairwise comparisons of between the subsets, within each level of by. The p-values are NOT adjusted for multiple hypothesis testing.

For binomial proportions, only "Prop.test" can be specified in the argument method; For continous variables, any function can be specified, which takes x and y as first arguments, and returns a list containing a list containing the appropriate p-value in the element named p.value (as do the functions of class "htest"). See the examples for details.

Value

A named list with elements

byout	a list, containing the output of pairwiseTestint for each level of by, i.e. a data.frame containing with columns p.value,compnames groupx, groupy
bynames	a character vector containing the names of the levels of the factors specified in by
method	a character string, name of the function used
control	a character string
by	vector of character strings, same as argument by
...	further arguments that were passed to FUN

Author(s)

Frank Schaarschmidt

See Also

You can use summary.pairwiseTest to calculate multiplicity adjusted p-values from the output of pairwiseTest.

The following methods provide multiplicity adjusted p-values for various situations: pairwise.t.test, pairwise.prop.test, \link{p.adjust}, summary.glht(multcomp), simtest.ratio(mratios)

Examples

```
######################################################
# The rooting example:
# Calculate confidence intervals for the
# difference of proportions between the 3 doses of IBA,
```

```
# separately for 4 combinations of "Age" and "Position".
# Note: we pool over Rep in that way. Whether this makes
# sense or not, is decision of the user.

data(rooting)

# Pairwise Chi-square tests:

aproots<-pairwiseTest(cbind(root, noroot) ~ IBA,
 data=rooting, by=c("Age", "Position"), method="Prop.test")

aproots

# With Holm adjustment for multiple hypotheses testing:

summary(aproots, p.adjust.method="holm")


############################################################

data(Oats)


apc <- pairwiseTest(yield ~ nitro, data=Oats,
 by="Variety", method="wilcox.test")

apc

summary(apc)
summary(apc, p.adjust.method="holm")
```

`pairwiseTestInt` *Internal functions for pairwiseTest*

Description

Only for internal use by pairwiseTest. Two different functions for data representable as a two numeric vectors (pairwiseTestCont) and data representable as matrix with two columns (pairwiseTestProp) as created can be done with a formula like `cbind(successes, failures) ~ group`. Functions that split up a data.frame according to one factor, and perform all pairwise comparisons and comparisons to control among the levels of the factor by calling functions that can deal with two vectors x and y or the one documented in ?Prop.test.

Usage

```
pairwiseTestCont(formula, data, control=NULL, method, ...)
pairwiseTestProp(formula, data, control=NULL, method, ...)
```

Arguments

formula	a formula specifiying the response and the factor variable: response ~ factor
data	a data frame, containing the variables specified in formula
method	character string, giving the name of the function, which shall be used to calculate local p-values. Any function, taking two vectors x, and y as first arguments and returning a list with the p.value in a list element named p.value can be specified.
control	optional character string, defining the name of a control group. Must be one of the levels of the factor variable defined in formula. By default control=NULL, then all pairwise comparisons between the levels of the factor variable are computed.
...	Arguments to be passed the function defined in method

Details

For internal use in pairwiseTest.

Value

a data.frame containing the columns

p.value	numeric vector: the p.values
compnames	character vector: the names of the comparisons performed
groupx	character vector: the names of the first group
groupy	character vector: the names of the second group

See Also

pairwiseTest for a user level function, and pairwise.t.test, pairwise.prop.test, p.adjust for further functions to calculate multiplicity adjusted p-values.

---

plot.pairwiseCI *Plotting the output of pairwiseCI*

Description

Easy method for plotting estimates and confidence bounds calculated using pairwiseCI.

Usage

```
## S3 method for class 'pairwiseCI'
plot(x,
 CIvert=NULL, CIlty = 1, CIlwd=1, CIcex=1,
 H0line=NULL, H0lty=1, H0lwd=1,
 main=NULL, ylab="", xlab="",
 ... )
```

Arguments

x	an object of class "pairwiseCI", the output of function \link{pairwiseCI}
CIvert	logical, whether confidence intervals shall be plotted vertical if CIvert=TRUE and horizontal if CIvert=FALSE
CIlty	integer, giving the line type of the CI, as documented for cex in ?par
CIlwd	integer, giving the line width of the CI, as documented for lwd ?par
CIcex	numerical value giving the size of CIsymbols relative to the default value, see cex in ?par
H0line	Value to be plotted as vertical or horizontal line, depending on the value of CIvert
H0lty	integer, giving the line type of the CI, as documented for lty in ?par
H0lwd	integer, giving the line width of the CI, as documented for lwd in ?par
main	as main in plot
ylab	label of y-axis as ylab in plot, default is no label
xlab	label of x-axis as ylab in plot, default is no label
...	Further arguments to be passed to axis. Note, that arguments las, at, labels are defined internally and can not be set via ...

Author(s)

Frank Schaarschmidt

Examples

```
data(Oats)

output <- pairwiseCI(yield ~ Block, data=Oats,
 by="nitro",method="Param.diff", control="I")

# default plot for difference methods:
plot(output)

# some small changes:
plot(output, CIvert=TRUE, H0line=c(-2,0,2), H0lty=c(2,1,2))

output <- pairwiseCI(yield ~ Block, data=Oats,
 by="nitro", method="Param.ratio", control="I")

# default plot for ratio methods:
plot(output)

# some small changes:
plot(output, CIvert=FALSE, H0line=c(0.7, 1, 1/0.7),
 H0lty=c(3,2,3))
```

`plotCI.methods` *Plot confidence intervals*

Description

Creates plot of confidence intervals calculated by calling "pairwiseMEP".

Usage

```
## S3 method for class 'pairwiseCI'
plotCI(x, ...)
## S3 method for class 'pairwiseMEP'
plotCI(x, whichep = NULL, ...)
```

Arguments

`x`	an object of class `'pairwiseMEP'` or `'pairwiseCI'`
`whichep`	an optional vector of character strings (or integers); specifying the names (or indices in the element `conf.int` of the list returned by `pairwiseMEP`) of those response variables for which the confidence intervals shall be plotted
`...`	further arguments to be passed to `plotCI` in package **MCPAN**, see `?plotCI` for details

Value

A plot.

Examples

```
x1<-rnorm(120,20,2)
x2<-rnorm(120,100,8)
x3<-rpois(120,10)
A<-rep(c("a1","a2","a3"), c(40,40,40))
B<-rep(rep(c("b1","b2"), c(20,20)), times=3)
dat<-data.frame(x1=x1,x2=x2,x3=x3,A=A, B=B)


test<-pairwiseMEP(x=dat, ep=c("x1","x2","x3"),
 f="A", by="B", conf.level=0.9, control="a1",
 method=c("Param.ratio","Param.ratio","Poisson.ratio"))

plotCI(test, whichep=c("x1","x2"), lines=c(0.8,1.25))

plotCI(test, whichep=c(1,2,3))
```

print.pairwiseCI *Print function for "pairwiseCI"*

Description

Print out confidence intervals calculated using pairwiseCI.

Usage

```
## S3 method for class 'pairwiseCI'
print(x , digits=4, ... )
```

Arguments

x	an object of class "pairwiseCI", the output of function pairwiseCI()
digits	integer, to which decimal output shall be rounded
...	further arguments to be passed to print

print.pairwiseTest *Print function for "pairwiseTest"*

Description

Print function for pairwiseTest objects

Usage

```
## S3 method for class 'pairwiseTest'
print(x, digits = 4, ...)
```

Arguments

x	an object of class "pairwiseTest"
digits	digits for rounding
...	further arguments to be passed to print

`print.summary.pairwiseCI`
Print function for "summary.pairwiseCI"

Description

Print function for summary.pairwiseCI

Usage

```
## S3 method for class 'summary.pairwiseCI'
print(x, ...)
```

Arguments

`x`	an object of class "summary.pairwiseCI", created by the function `summary.pairwiseCI`
`...`	further arguments to be passed to `print`

`print.summary.pairwiseTest`
Print function for "summary.pairwiseTest"

Description

Print function for summary.pairwiseCI

Usage

```
## S3 method for class 'summary.pairwiseTest'
print(x, ...)
```

Arguments

`x`	an object of class "summary.pairwiseTest", created by the function `summary.pairwiseTest`
`...`	further arguments to be passed to `print`

profileDG *Construct a (quasi-) likelihood-profile*

Description

Construct a (quasi-) likelihood-profile based on a glm-fit. For internal use for functions constructing profile-likelihood confidence intervals.

Usage

```
profileDG(fit, steps = 100, wh = 1:p)
```

Arguments

fit	an object of class "glm" or "negbin"
steps	a single integer value, the number of steps for the profile
wh	wh

Details

An adaptation of the code in profile.glm. Will work also for the case that only 0-values occur in one group. For internal use

Value

An object of class "profile.glm", "profile".

Author(s)

Daniel Gerhard

Prop.test *Wrapper to prop.test(stats)*

Description

Only for internal use in pairwiseTest.

Usage

```
Prop.test(x, y, alternative = "two.sided", test=c("prop.test", "fisher.test"), ...)
```

Arguments

x	a vector of success and failure in sample x, or a data.frame with a column of successes and a column of failures, then colSums are used.
y	a vector of success and failure in sample y, or a data.frame with a column of successes and a column of failures, then colSums are used.
alternative	character string defining the alternative hypothesis
test	a single character string: which function to be called, choices are "prop.test" for the chi-square test, and "fisher.test" for Fishers exact test, as they are defined in package stats
...	arguments to be passed to prop.test(stats) or fisher.test(stats)

Details

Just a wrapper function to call prop.test(stats). If x, y are data.frames containing two columns taken to be counts of successes and counts of failures, columnwise sums of x,y are calculated. The total number of successes and the total number of trials is then passed to prop.test.

Value

An object of class "htest", as defined by prop.test(stats)

See Also

prop.test, and pairwise.prop.test in **stats**

Examples

```
# If input is a data.frame:

set.seed(1234)

trials=rep(20,8)
success <- rbinom(n=8, size=trials,
 prob=c(0.2,0.2,0.2,0.2, 0.3,0.3,0.3,0.3))
failure <- trials-success

f<-as.factor(rep(c("group1", "group2"), each=4))

data<-data.frame(success=success, failure=failure, f=f)

g1<-subset(data, f=="group1")[,c("success","failure")]
g2<-subset(data, f=="group2")[,c("success","failure")]

g1
g2

# Prop.test calculates the columnwise sums and calls prop.test stats:
```

```
Prop.test(x=g1, y=g2)

# should be the same as:

CS1<-colSums(g1)
CS2<-colSums(g2)

CS1
CS2

prop.test(x=c(CS1[1], CS2[1]), n=c(sum(CS1), sum(CS2)))
```

QBmover *Confidence intervals for ratios of proportions based on the quasibinomial assumption*

Description

Confidence intervals for ratios of proportions with overdispersed binomial data in a one-factor quasibinomial generalized linear model. Intervals are computed using the MOVER-R method on profile deviance intervals (as implemented in mcprofile) for the single proportions.

Usage

```
QBmover(succ, fail, trt, conf.level = 0.95,
 alternative = "two.sided", grid = NULL)
```

Arguments

succ	vector of counts of successes
fail	vector of counts of failures
trt	factor variable distinguishing the treatment groups
conf.level	a single numeric value, the confidence level
alternative	a character string, "two.sided" for two-sided intervals, "less" for upper limits, "greater" for lower limits only
grid	optional, a numeric vector to be supplied to the profiling used internally in quasibin.ratio to obtain profile deviance intervals for each samples proportion on the logit-scale.

Value

A data.frame with three columns

est	estimated ratios
lower	lower confidence limits
upper	upper confidence limits

Note

Experimental

Author(s)

Frank Schaarschmidt

References

Donner and Zou (2012): Closed-form confidence intervals for functions of the normal mean and standard deviation. Statistical Methods in Medical Research 21(4):347-359. *Gerhard (2014):* Simultaneous Small Sample Inference For Linear Combinations Of Generalized Linear Model Parameters. Communications in Statistics - Simulation and Computation. DOI:10.1080/03610918.2014.895836

Examples

```
QBmover(succ=c(0,0,1,  0,6,8), fail=c(20,20,18, 20,14,12),
 trt=factor(rep(c("A", "B"), c(3,3))), conf.level = 0.95,
 alternative = "two.sided", grid = NULL)
```

---

repellent *Repellent effect of sulphur in eight concentrations*

Description

Sugar solutions were presented to certain number of bees. To assess the repellent effect of the fungicide sulphur of bees, increasing concentration of sulphur was added to the sugar solutions. The decrease of sugar solutions (i.e. uptake by bees) was measured. Low values of decrease therefore can be interpreted as high repellent effect of the sulphur concentration. Assumed to be a completely randomized design.

Usage

```
data(repellent)
```

Format

A data frame with 64 observations on the following 2 variables.

`decrease` a numeric vector, the absolut decrease of sugar solutions from begin to end of the experiment

`treatment` a factor with levels `A` `B` `C` `D` `E` `F` `G` `H`, the concentration of sulphur

Examples

```
data(repellent)
boxplot(decrease ~ treatment, data=repellent)
```

rooting *Rooting (success/failure) of plants in a 3-factorial field trial*

Description

Part of an experiment on propagation of plant genera Acer and Pyrus. Cuttings were taken from motherplants of age 5 and age 20, from top or base, and were treated with 0, 0.5 and 2 percent IBA to induce rooting. Treatments were arranged in a completely randomized design, among other variables, the number of cuttings with and without roots was recorded.

Usage

```
data(rooting)
```

Format

A data frame with 48 observations on the following 6 variables.

`Age` a numeric vector: age of mother plants

`Position` a factor with levels `B` and `T`, for "base" and "top" cuttings

`IBA` a numeric vector, specifying the concentration of IBA

`Rep` a numeric vector, number of replication

`root` a numeric vector, number of cuttings with successfull rooting, out of 12 trials

`noroot` a numeric vector, number of cuttings showing no rooting, out of 12 trials

Source

Data taken from Msc thesis by Dawit Mamushet Yifru, Institute of Floriculture, Tree Nursery Science and Plant Breeding, University of Hannover, 2005.

Examples

```
data(rooting)

rooting$IBAf<-as.factor(rooting$IBA)
rooting$Rep<-as.factor(rooting$Rep)

fitB<-glm(cbind(root,noroot)~Rep+(Age + Position + IBA)^2,
 data=rooting, family=binomial)

fitQB<-glm(cbind(root,noroot)~Rep+(Age + Position + IBA)^2,
 data=rooting, family=quasibinomial)

summary(fitB)
summary(fitQB)

anova(fitB, test="Chisq")
anova(fitQB, test="F")
```

sodium *Sodium contents in transgenic and isogenic corn*

Description

Sodium was measured in transgenic corn and the original isogenic corn variety.

Usage

```
data(sodium)
```

Format

A data frame with 12 observations on the following 2 variables.

`Treatment` a factor with levels `transgenic` `xisogenic`

`Sodiumcontent` a numeric vector

Source

Oberdoerfer, R.B. Example dataset from composition analyses of genetically modified oilseed rape seeds. 2003; BCS GmbH.

Examples

```
data(sodium)
boxplot(Sodiumcontent ~Treatment, data=sodium)
```

summary.pairwiseCI *Summary function for pairwiseCI*

Description

Creates a list of data.frames from the output of pairwiseCI

Usage

```
## S3 method for class 'pairwiseCI'
summary(object, digits = 4, ...)
```

Arguments

`object`	An object of class `"pairwiseCI"`, created using the function `pairwiseCI`
`digits`	number of digits for rounding of results
`...`	Currently not used.

Value

A list.

See Also

link{as.data.frame.pairwiseCI}

Examples

```
data(rooting)

rootRR<-pairwiseCI(cbind(root,noroot) ~ IBA,
 data=rooting, by="Age", method="Prop.ratio")

# after calling summary,
# extracting parts of the output is easier:

srootRR<-summary(rootRR)

srootRR$'20'$conf.int$upper
```

---

summary.pairwiseTest *Summary function for "pairwiseTest"*

Description

Creates a data.frame from the output of pairwiseTest, allows to adjust raw p-values by methods implemented in p.adjust.

Usage

```
## S3 method for class 'pairwiseTest'
summary(object, digits = 4,
 p.adjust.method = "none", ...)
```

Arguments

object	An object of class "pairwiseTest", created using the function pairwiseTest
digits	number of digits for rounding of results
p.adjust.method	Method to adjust p-values for multiple hypothesis testing, see options in p.adjust.method in **stats**. The default in this function in "none", resulting in unadjusted p-values
...	Currently not used.

Details

Coerces the raw p-values and the corresponding group levels to a data.frame and applies p.adjust to it.

Value

A dataframe, with columns

`p.val.raw`	raw p-values
`p.val.adj`	adjusted p-values, according to the method specified in `p.adjust.method`
`comparison`	the calculated differences or ratios of parameters
`groupx`	levels of group x
`groupy`	levels of group y

and possibly further columns containing levels of by.

Examples

```
data(Oats)
apOats<-pairwiseTest(yield~nitro, data=Oats,
 by="Variety", method="t.test", var.equal=FALSE)
apOats

# summary just creates a data.frame from the output
summary(apOats)

# an allows application of p.adjust
# on the p.values:

summary(apOats, p.adjust.method="holm")

# See ?p.adjust.methods for the methods available.
```

Index