Package ‘rcompanion’

May 27, 2024

Version 2.4.36

Date 2024-05-27

Title Functions to Support Extension Education Program Evaluation

Description Functions and datasets to support Summary and Analysis of Extension Program Evaluation in R, and An R Companion for the Handbook of Biological Statistics. Vignettes are available at <`https://rcompanion.org`>.

URL `https://CRAN.R-project.org/package=rcompanion`

Depends R (>= 4.1.0)

Imports stats (>= 4.1.0), graphics (>= 4.1.0), boot (>= 1.3-28), DescTools (>= 0.99.43), multcompView (>= 0.1-8), plyr (>= 1.8.6), coin (>= 1.4-2), lmtest (>= 0.9-38), nortest (>= 1.0-4)

Maintainer Salvatore Mangiafico <`mangiafico@njaes.rutgers.edu`>

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

NeedsCompilation no

Author Salvatore Mangiafico [aut, cre] (<`https://orcid.org/0000-0003-4668-6821`>)

Repository CRAN

Date/Publication 2024-05-27 16:20:02 UTC

R topics documented:

`rcompanion-package` *Functions to Support Extension Education Program Evaluation*

Description

Functions and datasets to support Summary and Analysis of Extension Program Evaluation in R and An R Companion for the Handbook of Biological Statistics.

Useful functions

There are several functions that provide summary statistics for grouped data. These function titles tend to start with `"groupwise"`. They provide means, medians, geometric means, and Huber M-estimators for groups, along with confidence intervals by traditional methods and bootstrap.

Functions to produce effect size statistics, some with bootstrapped confidence intervals, include those for Cramer's V, Cohen's g and odds ratio for paired tables, Cohen's h, Cohen's w, Vargha and Delaney's A, Cliff's delta, r for one-sample, two-sample, and paired Wilcoxon and Mann-Whitney tests, epsilon-squared, and Freeman's theta.

The `accuracy` function reports statistics for models including minimum maximum accuracy, MAPE, RMSE, Efron's pseudo r-squared, and coefficient of variation.

The functions `nagelkerke` and `efronRSquared` provide pseudo R-squared values for a variety of model types, as well as a likelihood ratio test for the model as a whole.

There are also functions that are useful for comparing models. `compareLM`, `compareGLM`, and `pairwiseModelAnova`. These use goodness-of-fit measures like AIC, BIC, and BICc, or likelihood ratio tests.

Functions for nominal data include post-hoc tests for Cochran-Mantel-Haenszel test (`groupwiseCMH`), for McNemar-Bowker test (`pairwiseMcnemar`), and for tests of association like Chi-square, Fisher exact, and G-test (`pairwiseNominalIndependence`).

There are a few useful plotting functions, including `plotNormalHistogram` that plots a histogram of values and overlays a normal curve, and `plotPredy` which plots of line for predicted values for a bivariate model. Other plotting functions include producing density plots.

A function close to my heart is `cateNelson`, which performs Cate-Nelson analysis for bivariate data.

Vignettes and examples

The functions in this package are used in "Extension Education Program Evaluation in R" which is available at `https://rcompanion.org/handbook/` and "An R Companion for the Handbook of Biological Statistics" which is available at `https://rcompanion.org/rcompanion/`.

The documentation for each function includes an example as well.

Version notes

Version 2.0 is not entirely back-compatible as several functions have been removed. These include some of the pairwise methods that can be replaced with better methods. Also, some functions have been removed or modified in order to import fewer packages.

Removed packages are indicated with 'Defunct' in their titles.

`accuracy` *Minimum maximum accuracy, mean absolute percent error, median absolute error, root mean square error, coefficient of variation, and Efron's pseudo r-squared*

Description

Produces a table of fit statistics for multiple models.

Usage

```
accuracy(fits, plotit = FALSE, digits = 3, ...)
```

Arguments

`fits`	A series of model object names. Must be a list of model objects or a single model object.
`plotit`	If `TRUE`, produces plots of the predicted values vs. the actual values for each model.
`digits`	The number of significant digits in the output.
`...`	Other arguments passed to `plot`.

Details

Produces a table of fit statistics for multiple models: minimum maximum accuracy, mean absolute percentage error, median absolute error, root mean square error, normalized root mean square error, Efron's pseudo r-squared, and coefficient of variation.

For minimum maximum accuracy, larger indicates a better fit, and a perfect fit is equal to 1.

For mean absolute error (MAE), smaller indicates a better fit, and a perfect fit is equal to 0. It has the same units as the dependent variable. Note that here, MAE is simply the mean of the absolute values of the differences of predicted values and the observed values (`MAE = mean(abs(predy - actual))`). There are other definitions of MAE and similar-sounding terms.

Median absolute error (MedAE) is similar, except employing the median rather than the mean.

For mean absolute percent error (MAPE), smaller indicates a better fit, and a perfect fit is equal to 0. The result is reported as a fraction. That is, a result of 0.1 is equal to 10 percent.

Root mean square error (RMSE) has the same units as the predicted values.

Normalized root mean square error (NRMSE) is RMSE divided by the mean or the median of the values of the dependent variable.

Efron's pseudo r-squared is calculated as 1 minus the residual sum of squares divided by the total sum of squares. For linear models (`lm` model objects), Efron's pseudo r-squared will be equal to r-squared. For other models, it should not be interpreted as r-squared, but can still be useful as a relative measure.

`CV.prcnt` is the coefficient of variation for the model. Here it is expressed as a percent. That is, a result of 10 = 10 percent.

Model objects currently supported: lm, glm, nls, betareg, gls, lme, lmer, lmerTest, glmmTMB, rq, loess, gam, glm.nb, glmRob, mblm, and rlm.

Value

A list of two objects: The series of model calls, and a data frame of statistics for each model.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/G_14.html`

See Also

`compareLM`, `compareGLM`, `nagelkerke`

Examples

```
data(BrendonSmall)
BrendonSmall$Calories = as.numeric(BrendonSmall$Calories)
BrendonSmall$Calories2 = BrendonSmall$Calories ^ 2
model.1 = lm(Sodium ~ Calories, data = BrendonSmall)

accuracy(model.1, plotit=FALSE)

model.2 = lm(Sodium ~ Calories + Calories2, data = BrendonSmall)
model.3 = glm(Sodium ~ Calories, data = BrendonSmall, family="Gamma")
quadplat = function(x, a, b, clx) {
          ifelse(x  < clx, a + b * x   + (-0.5*b/clx) * x   * x,
                           a + b * clx + (-0.5*b/clx) * clx * clx)}
model.4 = nls(Sodium ~ quadplat(Calories, a, b, clx),
              data = BrendonSmall,
              start = list(a=519, b=0.359, clx = 2300))

accuracy(list(model.1, model.2, model.3, model.4), plotit=FALSE)

### Perfect and poor model fits
X = c(1, 2,  3,  4,  5,  6, 7, 8, 9, 10, 11, 12)
Y = c(1, 2,  3,  4,  5,  6, 7, 8, 9, 10, 11, 12)
Z = c(1, 12, 13, 6, 10, 13, 4, 3, 5,  6, 10, 14)
perfect = lm(Y ~ X)
poor    = lm(Z ~ X)
accuracy(list(perfect, poor), plotit=FALSE)
```

`Anderson` *Hypothetical data for Alexander Anderson*

Description

A matrix of counts for students passing or failing a pesticide training course across four counties. Hypothetical data.

Usage

```
Anderson
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 2 columns.

Source

https://rcompanion.org/handbook/H_04.html

`AndersonBias` *Hypothetical data for Alexander Anderson with gender bias*

Description

A data frame of counts for students passing or failing a pesticicde training course across four counties, with gender of students. Hypothetical data.

Usage

```
AndersonBias
```

Format

An object of class `data.frame` with 16 rows and 4 columns.

Source

https://rcompanion.org/handbook/H_06.html

`AndersonRainBarrel` *Hypothetical data for Alexander Anderson on rain barrel installation*

Description

A matrix of paired counts for students planning to install rain barrels before and after a class. Hypothetical data.

Usage

```
AndersonRainBarrel
```

Format

An object of class `matrix` (inherits from `array`) with 2 rows and 2 columns.

Source

https://rcompanion.org/handbook/H_05.html

`AndersonRainGarden` *Hypothetical data for Alexander Anderson on rain garden installation*

Description

A matrix of paired counts for students planning to install rain gardens before and after a class. Hypothetical data.

Usage

```
AndersonRainGarden
```

Format

An object of class `matrix` (inherits from `array`) with 3 rows and 3 columns.

Source

https://rcompanion.org/handbook/H_05.html `blom` *Normal scores transformation*

Description

Normal scores transformation (Inverse normal transformation) by Elfving, Blom, van der Waerden, Tukey, and rankit methods, as well as z score transformation (standardization) and scaling to a range (normalization).

Usage

```
blom(
  x,
  method = "general",
  alpha = pi/8,
  complete = FALSE,
  na.last = "keep",
  na.rm = TRUE,
  adjustN = TRUE,
  min = 1,
  max = 10,
  ...
)
```

Arguments

`x`	A vector of numeric values.
`method`	Any one `"general"` (the default), `"blom"`, `vdw`, `"tukey"`, `"elfving"`, `"rankit"`, `zscore`, or `scale`.
`alpha`	A value used in the `"general"` method. If alpha=pi/8 (the default), the `"general"` method reduces to the `"elfving"` method. If alpha=3/8, the `"general"` method reduces to the `"blom"` method. If alpha=1/2, the `"general"` method reduces to the `"rankit"` method. If alpha=1/3, the `"general"` method reduces to the `"tukey"` method. If alpha=0, the `"general"` method reduces to the `"vdw"` method.
`complete`	If `TRUE`, `NA` values are removed before transformation. The default is `FALSE`.
`na.last`	Passed to `rank` in the normal scores methods. See the documentation for the `rank` function. The default is `"keep"`.
`na.rm`	Used in the `"zscore"` and `"scale"` methods. Passed to `mean`, `min`, and `max` functions in those methods. The default is `TRUE`.
`adjustN`	If `TRUE`, the default, the normal scores methods use only non-`NA` values to determine the sample size, `N`. This seems to work well under default conditions where `NA` values are retained, even if there are a high percentage of `NA` values.
`min`	For the `"scale"` method, the minimum value of the transformed values.
`max`	For the `"scale"` method, the maximum value of the transformed values.
`...`	additional arguments passed to `rank`.

Details

By default, `NA` values are retained in the output. This behavior can be changed with the `na.rm` argument for `"zscore"` and `"scale"` methods, or with `na.last` for the normal scores methods. Or `NA` values can be removed from the input with `complete=TRUE`.

For normal scores methods, if there are `NA` values or tied values, it is helpful to look up the documentation for `rank`.

In general, for normal scores methods, either of the arguments `method` or `alpha` can be used. With the current algorithms, there is no need to use both.

Normal scores transformation will return a normal distribution with a mean of 0 and a standard deviation of 1.

The `"scale"` method coverts values to the range specified in `max` and `min` without transforming the distribution of values. By default, the `"scale"` method converts values to a 1 to 10 range. Using the `"scale"` method with `min = 0` and `max = 1` is sometimes called "normalization".

The `"zscore"` method converts values by the usual method for z scores: `(x - mean(x)) / sd(x)`. The transformed values with have a mean of 0 and a standard deviation of 1 but won't be coerced into a normal distribution. Sometimes this method is called "standardization".

Value

A vector of numeric values.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

Conover, 1995, Practical Nonparametric Statistics, 3rd.

Solomon & Sawilowsky, 2009, Impact of rank-based normalizing transformations on the accuracy of test scores.

Beasley and Erickson, 2009, Rank-based inverse normal transformations are increasingly used, but are they merited?

Examples

```
set.seed(12345)
A = rlnorm(100)
## Not run: hist(A)
### Convert data to normal scores by Elfving method
B = blom(A)
## Not run: hist(B)
### Convert data to z scores
C = blom(A, method="zscore")
## Not run: hist(C)
### Convert data to a scale of 1 to 10
D = blom(A, method="scale")
## Not run: hist(D)
```

```
### Data from Sokal and Rohlf, 1995,
### Biometry: The Principles and Practice of Statistics
### in Biological Research
Value = c(709,679,699,657,594,677,592,538,476,508,505,539)
Sex   = c(rep("Male",3), rep("Female",3), rep("Male",3), rep("Female",3))
Fat   = c(rep("Fresh", 6), rep("Rancid", 6))
ValueBlom = blom(Value)
Sokal = data.frame(ValueBlom, Sex, Fat)
model = lm(ValueBlom ~ Sex * Fat, data=Sokal)
anova(model)
## Not run:
hist(residuals(model))
plot(predict(model), residuals(model))

## End(Not run)
```

`BobBelcher` *Hypothetical data for ratings of instructors in unreplicated CBD*

Description

A data frame of Likert responses for five instructors for each of 8 respondents. Arranged in unreplicated complete block design. Hypothetical data.

Usage

```
BobBelcher
```

Format

An object of class `data.frame` with 40 rows and 3 columns.

Source

https://rcompanion.org/handbook/F_10.html

`Breakfast` *Hypothetical data for students' breakfast habits and travel to school*

Description

A two-dimensional contingency table, in which Breakfast is an ordered nominal variable, and Travel is a non-ordered nominal variable. Hypothetical data.

Usage

```
Breakfast
```

Format

An object of class `table` with 3 rows and 5 columns.

Source

https://rcompanion.org/handbook/H_09.html

`BrendonSmall` *Hypothetical data for Brendon Small and company*

Description

A data frame of the intake of calories and sodium for students in five classes. Hypothetical data.

Usage

```
BrendonSmall
```

Format

An object of class `data.frame` with 45 rows and 6 columns.

Source

https://rcompanion.org/handbook/I_10.html

`BullyHill` *Hypothetical data for proportion of students passing a certification*

Description

A data frame of counts of students passing and failing. Hypothetical data.

Usage

```
BullyHill
```

Format

An object of class `data.frame` with 12 rows and 5 columns.

Source

https://rcompanion.org/handbook/J_02.html

`Catbus` *Hypothetical data for Catbus and company*

Description

A data frame of the number of steps taken by students in three classes. Hypothetical data.

Usage

```
Catbus
```

Format

An object of class `data.frame` with 26 rows and 5 columns.

Source

https://rcompanion.org/handbook/C_03.html

`cateNelson` *Cate-Nelson models for bivariate data*

Description

Produces critical-x and critical-y values for bivariate data according to a Cate-Nelson analysis.

Usage

```
cateNelson(
  x,
  y,
  plotit = TRUE,
  hollow = TRUE,
  xlab = "X",
  ylab = "Y",
  trend = "positive",
  clx = 1,
  cly = 1,
  xthreshold = 0.1,
  ythreshold = 0.1,
  progress = TRUE,
  verbose = TRUE,
  listout = FALSE
)
```

Arguments

`x`	A vector of values for the x variable.
`y`	A vector of values for the y variable.
`plotit`	If `TRUE`, produces plots of the output.
`hollow`	If `TRUE`, uses hollow circles on the plot to indicate data not fitting the model.
`xlab`	The label for the x-axis.
`ylab`	The label for the y-axis.
`trend`	`"postive"` if the trend of y vs. `x` is generally positive. `"negative"` if negative.
`clx`	Indicates which of the listed critical x values should be chosen as the critical x value for the final model.
`cly`	Indicates which of the listed critical y values should be chosen as the critical y value for the final model.
`xthreshold`	Indicates the proportion of potential critical x values to display in the output. A value of `1` would display all of them.
`ythreshold`	Indicates the proportion of potential critical y values to display in the output. A value of `1` would display all of them.
`progress`	If `TRUE`, prints an indicator of progress as for loops progress.
`verbose`	If `FALSE`, suppresses printed output of tables.
`listout`	If `TRUE`, outputs a list of data frames instead of a a single data frame. This allows a data frame of critical values and associated statistics to be extracted, for example if one would want to sort by Cramer's V.

Details

Cate-Nelson analysis divides bivariate data into two groups. For data with a positive trend, one group has a large `x` value associated with a large `y` value, and the other group has a small `x` value associated with a small `y` value. For a negative trend, a small `x` is associated with a large `y`, and so on.

The analysis is useful for bivariate data which don't conform well to linear, curvilinear, or plateau models.

This function will fail if either of the largest two or smallest two x values are identical.

Value

A data frame of statistics from the analysis: number of observations, critical level for x, sum of squares, critical value for y, the number of observations in each of the quadrants (I, II, III, IV), the number of observations that conform with the model, the proportion of observations that conform with the model, the number of observations that do not conform to the model, the proportion of observations that do not conform to the model, a p-value for the Fisher exact test for the data divided into the groups indicated by the model, and Cramer's V for the data divided into the groups indicated by the model.

Output also includes printed lists of critical values, explanation of the values in the data frame, and plots: y vs. `x`; sum of squares vs. critical x value; the number of observations that do not conform to the model vs. critical y value; and y vs. `x` with the critical values shown as lines on the plot, and the quadrants labeled.

Note

The method in this function follows *Cate, R. B., & Nelson, L.A. (1971). A simple statistical procedure for partitioning soil test correlation data into two classes. Soil Science Society of America Proceedings 35, 658-660.*

An earlier version of this function was published in *Mangiafico, S.S. 2013. Cate-Nelson Analysis for Bivariate Data Using R-project. J.of Extension 51:5, 5TOT1.*

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/rcompanion/h_02.html`

Cate, R. B., & Nelson, L.A. (1971). A simple statistical procedure for partitioning soil test correlation data into two classes. Soil Science Society of America Proceedings 35, 658–660.

See Also

`cateNelsonFixedY`

Examples

```
data(Nurseries)
cateNelson(x          = Nurseries$Size,
           y          = Nurseries$Proportion,
           plotit     = TRUE,
           hollow     = TRUE,
           xlab       = "Nursery size in hectares",
           ylab       = "Proportion of good practices adopted",
           trend      = "positive",
           clx        = 1,
           xthreshold = 0.10,
           ythreshold = 0.15)
```

`cateNelsonFixedY` *Cate-Nelson models for bivariate data with a fixed critical Y value*

Description

Produces critical-x values for bivariate data according to a Cate-Nelson analysis for a given critical Y value.

Usage

```
cateNelsonFixedY(
  x,
  y,
  cly = 0.95,
  plotit = TRUE,
  hollow = TRUE,
  xlab = "X",
  ylab = "Y",
  trend = "positive",
  clx = 1,
  outlength = 20,
  sortstat = "error"
)
```

Arguments

`x`	A vector of values for the x variable.
`y`	A vector of values for the y variable.
`cly`	= Critical Y value.
`plotit`	If `TRUE`, produces plots of the output.
`hollow`	If `TRUE`, uses hollow circles on the plot to indicate data not fitting the model.
`xlab`	The label for the x-axis.
`ylab`	The label for the y-axis.
`trend`	`"postive"` if the trend of y vs. x is generally positive. `"negative"` if negative.
`clx`	Indicates which of the listed critical x values should be chosen as the critical x value for the plot.
`outlength`	Indicates the number of potential critical x values to display in the output.
`sortstat`	The statistic to sort by. Any of `"error"` (the default), `"phi"`, `"fisher"`, or `"pearson"`.

Details

Cate-Nelson analysis divides bivariate data into two groups. For data with a positive trend, one group has a large `x` value associated with a large `y` value, and the other group has a small `x` value associated with a small `y` value. For a negative trend, a small `x` is associated with a large `y`, and so on.

The analysis is useful for bivariate data which don't conform well to linear, curvilinear, or plateau models.

Value

A data frame of statistics from the analysis: critical level for x, critical value for y, the number of observations in each of the quadrants (I, II, III, IV), the number of observations that conform with the model, the number of observations that do not conform to the model, the proportion of observations that conform with the model, the proportion of observations that do not conform to the model, a p-value for the Fisher exact test for the data divided into the groups indicated by the model, phi for the data divided into the groups indicated by the model, and Pearson's chi-square for the data divided into the groups indicated by the model.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/rcompanion/h_02.html`

See Also

`cateNelson`

Examples

```
data(Nurseries)
cateNelsonFixedY(x          = Nurseries$Size,
                 y          = Nurseries$Proportion,
                 cly        = 0.70,
                 plotit     = TRUE,
                 hollow     = TRUE,
                 xlab       = "Nursery size in hectares",
                 ylab       = "Proportion of good practices adopted",
                 trend      = "positive",
                 clx        = 1,
                 outlength  = 15)
```

---

cldList *Compact letter display for lists of comparisons*

Description

Produces a compact letter display (cld) from pairwise comparisons that were summarized in a table of comparisons

Usage

```
cldList(
  formula = NULL,
  data = NULL,
  comparison = NULL,
  p.value = NULL,
  threshold = 0.05,
  print.comp = FALSE,
```

```
  remove.space = TRUE,
  remove.equal = TRUE,
  remove.zero = TRUE,
  swap.colon = TRUE,
  swap.vs = FALSE,
  ...
)
```

Arguments

`formula`	A formula indicating the variable holding p-values and the variable holding the comparisons. e.g. P.adj ~ Comparison.
`data`	The data frame to use.
`comparison`	A vector of text describing comparisons, with each element in a form similar to "Treat.A - Treat.B = 0". Spaces and "=" and "0" are removed by default
`p.value`	A vector of p-values corresponding to the comparisons in the `comparison` argument
`threshold`	The alpha value. That is, the p-value below which the comparison will be considered significant
`print.comp`	If `TRUE`, prints out a data frame of the modified text of the comparisons. Useful for debugging
`remove.space`	If `TRUE`, removes spaces from the text of the comparisons
`remove.equal`	If `TRUE`, removes "=" from the text of the comparisons
`remove.zero`	If `TRUE`, removes "0" from the text of the comparisons
`swap.colon`	If `TRUE`, swaps ":" with "-" in the text of the comparisons
`swap.vs`	If `TRUE`, swaps "vs" with "-" in the text of the comparisons
`...`	Additional arguments passed to `multcompLetters`

Details

The input should include either `formula` and `data`; or `comparison` and `p.value`.

This function relies upon the `multcompLetters` function in the `multcompView` package. The text for the comparisons passed to `multcompLetters` should be in the form "Treat.A-Treat.B". Currently by default `cldList` removes spaces, equal signs, and zeros, by default, and so can use text in the form e.g. "Treat.A - Treat.B = 0". It also changes ":" to "-", and so can use text in the form e.g. "Treat.A : Treat.B".

Value

A data frame of group names, group separation letters, and monospaced separtions letters

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

It is often helpful to reorder the factor levels in the data set so that the group with the largest e.g. mean or median is first, and so on.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/G_06.html

See Also

`fullPTable`

Examples

```
data(BrendonSmall)

model = aov(Calories ~ Instructor, data=BrendonSmall)

TUK = TukeyHSD(model, "Instructor", ordered = TRUE)

### Convert the TukeyHSD output to a standard data frame

TUK = as.data.frame(TUK$Instructor)
names(TUK) = gsub(" ", ".", names(TUK))

HSD = data.frame(Comparison=row.names(TUK),
                 diff=TUK$diff, lwr=TUK$lwr, lwr=TUK$lwr, p.adj=TUK$p.adj)

HSD

cldList(p.adj ~ Comparison, data = HSD,
        threshold = 0.05,
        remove.space=FALSE)
```

`cliffDelta` *Cliff's delta*

Description

Calculates Cliff's delta with confidence intervals by bootstrap

Usage

```
cliffDelta(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  ci = FALSE,
```

```
    conf = 0.95,
    type = "perc",
    R = 1000,
    histogram = FALSE,
    reportIncomplete = FALSE,
    brute = FALSE,
    verbose = FALSE,
    digits = 3,
    ...
  )
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`brute`	If `FALSE`, the default, the statistic is based on the U statistic from the `wilcox.test` function. If `TRUE`, the function will compare values in the two samples directly.
`verbose`	If `TRUE`, reports the proportion of ties and the proportions of (Ya > Yb) and (Ya < Yb).
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `wilcox.test` function.

Details

Cliff's delta is an effect size statistic appropriate in cases where a Wilcoxon-Mann-Whitney test might be used. It ranges from -1 to 1, with 0 indicating stochastic equality, and 1 indicating that the first group dominates the second. It is linearly related to Vargha and Delaney's A.

By default, the function calculates Cliff's delta from the "W" U statistic from the `wilcox.test` function. Specifically, `VDA = U/(n1*n2); CD = (VDA-0.5)*2`.

The input should include either `formula` and `data`; or `x`, and `y`. If there are more than two groups, only the first two groups are used.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, Cliff's delta is positive. When the data in the second group are greater than in the first group, Cliff's delta is negative.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

When Cliff's delta is close to 1 or close to -1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, Cliff's delta. Or a small data frame consisting of Cliff's delta, and the lower and upper confidence limits.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_04.html`

See Also

`vda`, `multiVDA`

Examples

```
data(Catbus)
cliffDelta(Steps ~ Gender, data=Catbus)
```

`cohenG` *Cohen's g and odds ratio for paired contingency tables*

Description

Calculates Cohen's g and odds ratio for paired contingency tables, such as those that might be analyzed with McNemar or McNemar-Bowker tests.

Usage

```
cohenG(
  x,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  ...
)
```

Arguments

`x`	A two-way contingency table. It must be square. It can have two or more levels for each dimension.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`...`	Additional arguments (ignored).

Details

For a 2 x 2 table, where a and d are the concordant cells and b and c are discordant cells: Odds ratio is b/c; P is b/(b+c); and Cohen's g is P - 0.5.

In the 2 x 2 case, the statistics are directional. That is, when cell [1, 2] in the table is greater than cell [2, 1], OR is greater than 1, P is greater than 0.5, and g is positive.

In the opposite case, OR is less than 1, P is less than 0.5, and g is negative.

In the 2 x 2 case, when the effect is small, the confidence interval for OR can pass through 1, for g can pass through 0, and for P can pass through 0.5.

For tables larger than 2 x 2, the statistics are not directional. That is, OR is always >= 1, P is always >= 0.5, and g is always positive. Because of this, if `type="perc"`, the confidence interval will never cross the values for no effect (OR = 1, P = 0.5, or g = 0). Because of this, the confidence interval range in this case should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross the values for no effect.

When the reported statistics are close to their extremes, or with small counts, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A list containing: a data frame of results of the global statistics; and a data frame of results of the pairwise statistics.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/H_05.html`

See Also

`nominalSymmetryTest`, `cohenH`

Examples

```
### 2 x 2 repeated matrix example
data(AndersonRainBarrel)
cohenG(AndersonRainBarrel)

### 3 x 3 repeated matrix
data(AndersonRainGarden)
cohenG(AndersonRainGarden)
```

`cohenH` *Cohen's h to compare proportions for 2 x 2 contingency tables*

Description

Calculates Cohen's h for 2 x 2 contingency tables, such as those that might be analyzed with a chi-square test of association.

Usage

```
cohenH(x, observation = "row", verbose = TRUE, digits = 3)
```

Arguments

`x`	A 2 x 2 contingency table.
`observation`	If `"row"`, the row constitutes an observation. That is, the sum of each row is 100 percent. If `"column"`, the column constitutes an observation. That is, the sum of each column is 100 percent.
`verbose`	If `TRUE`, prints the proportions for each observation.
`digits`	The number of significant digits in the output.

Details

Cohen's h is an effect size to compare two proportions. For a 2 x 2 table: Cohen's h equals Phi2 - Phi1, where, If observations are in rows, P1 = a/(a+b) and P2 = c/(c+d). If observations are in columns, P1 = a/(a+c) and P2 = b/(b+d). Phi = 2 * asin(sqrt(P))

Value

A single statistic.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/H_10.html`

See Also

`cohenG`

Examples

```
data(Pennsylvania18)
Pennsylvania18
cohenH(Pennsylvania18, observation="row")
```

---

cohenW *Cohen's w (omega)*

Description

Calculates Cohen's w for a table of nominal variables.

Usage

```
cohenW(
  x,
  y = NULL,
  p = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 4,
  reportIncomplete = FALSE,
```

```
  ...
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations for one dimension of a two-way table.
`y`	If `x` is a vector, `y` is the vector of observations for the second dimension of a two-way table.
`p`	If `x` is a vector of observed counts, `p` can be given as a vector of theoretical probabilties, as in a chi-square goodness of fit test.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure. In the case of the goodness-of-fit scenario, setting this to `TRUE` will have no effect.
`...`	Additional arguments passed to `chisq.test`.

Details

Cohen's w is used as a measure of association between two nominal variables, or as an effect size for a chi-square test of association. For a 2 x 2 table, the absolute value of the phi statistic is the same as Cohen's w. The value of Cohen's w is not bound by 1 on the upper end.

Cohen's w is "naturally nondirectional". That is, the value will always be zero or positive. Because of this, if `type="perc"`, the confidence interval will never cross zero. The confidence interval range should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When w is close to 0 or very large, or with small counts, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, Cohen's w. Or a small data frame consisting of Cohen's w, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/H_10.html

Cohen J. 1992. "A Power Primer". Psychological Bulletin 12(1): 155-159.

Cohen, J. 1988. Statistical Power Analysis for the Behavioral Sciences, 2nd Ed. Routledge.

See Also

`cramerV`

Examples

```
### Example with table
data(Anderson)
fisher.test(Anderson)
cohenW(Anderson)

### Example for goodness-of-fit
### Bird foraging example, Handbook of Biological Statistics
observed = c(70,   79,   3,    4)
expected = c(0.54, 0.40, 0.05, 0.01)
chisq.test(observed, p = expected)
cohenW(observed, p = expected)

### Example with two vectors
Species = c(rep("Species1", 16), rep("Species2", 16))
Color   = c(rep(c("blue", "blue", "blue", "green"),4),
            rep(c("green", "green", "green", "blue"),4))
fisher.test(Species, Color)
cohenW(Species, Color)
```

`compareGLM` *Compare fit statistics for glm models*

Description

Produces a table of fit statistics for multiple glm models.

Usage

```
compareGLM(fits, ...)
```

Arguments

`fits`	A series of model object names, separated by commas.
`...`	Other arguments passed to `list`.

Details

Produces a table of fit statistics for multiple glm models: AIC, AICc, BIC, p-value, pseudo R-squared (McFadden, Cox and Snell, Nagelkerke).

Smaller values for AIC, AICc, and BIC indicate a better balance of goodness-of-fit of the model and the complexity of the model. The goal is to find a model that adequately explains the data without having too many terms.

BIC tends to choose models with fewer parameters relative to AIC.

For comparisons with AIC, etc., to be valid, both models must have the same data, without transformations, use the same dependent variable, and be fit with the same method. They do not need to be nested.

The function will fail if a model formula is longer than 500 characters.

Value

A list of two objects: The series of model calls, and a data frame of statistics for each model.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/rcompanion/e_07.html`

See Also

`compareLM`, `pairwiseModelAnova`, `accuracy`

Examples

```
### Compare among logistic regresion models
data(AndersonBias)
model.0 = glm(Result ~ 1, weight = Count, data = AndersonBias,
              family = binomial(link="logit"))
model.1 = glm(Result ~ County, weight = Count, data = AndersonBias,
              family = binomial(link="logit"))
model.2 = glm(Result ~ County + Gender, weight = Count, data = AndersonBias,
              family = binomial(link="logit"))
model.3 = glm(Result ~ County + Gender + County:Gender, weight = Count,
              data = AndersonBias, family = binomial(link="logit"))
compareGLM(model.0, model.1, model.2, model.3)
```

`compareLM` *Compare fit statistics for lm models*

Description

Produces a table of fit statistics for multiple lm models.

Usage

```
compareLM(fits, ...)
```

Arguments

`fits`	A series of model object names, separated by commas.
`...`	Other arguments passed to `list`.

Details

Produces a table of fit statistics for multiple lm models: AIC, AICc, BIC, p-value, R-squared, and adjusted R-squared.

Smaller values for AIC, AICc, and BIC indicate a better balance of goodness-of-fit of the model and the complexity of the model. The goal is to find a model that adequately explains the data without having too many terms.

BIC tends to choose models with fewer parameters relative to AIC.

In the table, `Shapiro.W` and `Shapiro.p` are the W statistic and p-value for the Shapiro-Wilks test on the residuals of the model.

For comparisons with AIC, etc., to be valid, both models must have the same data, without transformations, use the same dependent variable, and be fit with the same method. They do not need to be nested.

The function will fail if a model formula is longer than 500 characters.

Value

A list of two objects: The series of model calls, and a data frame of statistics for each model.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/I_10.html`, `https://rcompanion.org/rcompanion/e_05.html`

See Also

`compareGLM`, `pairwiseModelAnova`, `accuracy`

Examples

```
### Compare among polynomial models
data(BrendonSmall)
BrendonSmall$Calories = as.numeric(BrendonSmall$Calories)

BrendonSmall$Calories2 = BrendonSmall$Calories * BrendonSmall$Calories
BrendonSmall$Calories3 = BrendonSmall$Calories * BrendonSmall$Calories *
                         BrendonSmall$Calories
BrendonSmall$Calories4 = BrendonSmall$Calories * BrendonSmall$Calories *
                         BrendonSmall$Calories * BrendonSmall$Calories
model.1 = lm(Sodium ~ Calories, data = BrendonSmall)
model.2 = lm(Sodium ~ Calories + Calories2, data = BrendonSmall)
model.3 = lm(Sodium ~ Calories + Calories2 + Calories3, data = BrendonSmall)
model.4 = lm(Sodium ~ Calories + Calories2 + Calories3 + Calories4,
             data = BrendonSmall)
compareLM(model.1, model.2, model.3, model.4)
```

`countRSquare` *Count pseudo r-squared for logistic and other binary outcome models*

Description

Produces the count pseudo r-squared measure for models with a binary outcome.

Usage

```
countRSquare(
  fit,
  digits = 3,
  suppressWarnings = TRUE,
  plotit = FALSE,
  jitter = FALSE,
  pch = 1,
  ...
)
```

Arguments

`fit`	The fitted model object for which to determine pseudo r-squared. `glm` and `glmmTMB` are supported. Others may work as well.
`digits`	The number of digits in the outputted values.
`suppressWarnings`	If `TRUE`, suppresses warning messages.
`plotit`	If `TRUE`, produces a simple plot of actual vs. predicted values.
`jitter`	If `TRUE`, jitters the "actual" values in the plot.
`pch`	Passed to `plot`.
`...`	Additional arguments.

Details

The count pseudo r-squared is simply the number of correctly predicted observations divided the total number of observations.

This version is appropriate for models with a binary outcome.

The adjusted value deducts the count of the most frequent outcome from both the numerator and the denominator.

It is recommended that the model is fit on data in long format. That is, that the `weight` option not be used in the model.

The function makes no provisions for `NA` values. It is recommended that `NA` values be removed before the determination of the model.

Value

A list including a description of the submitted model, a data frame with the pseudo r-squared results, and a confusion matrix of the results.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

https://stats.oarc.ucla.edu/other/mult-pkg/faq/general/faq-what-are-pseudo-r-squareds/, https://rcompanion.org/handbook/H_08.html, https://rcompanion.org/rcompanion/e_06.html

See Also

`nagelkerke`, `efronRSquared`, `accuracy`

Examples

```
data(AndersonBias)

### Covert data to long format

Long = AndersonBias[rep(row.names(AndersonBias), AndersonBias$Count),
                    c("Result", "County", "Gender")]
rownames(Long) = seq(1:nrow(Long))
str(Long)

### Fit model and determine count r-square

model = glm(Result ~ County + Gender + County:Gender,
            data = Long,
            family = binomial())

countRSquare(model)
```

`cramerV` *Cramer's V (phi)*

Description

Calculates Cramer's V for a table of nominal variables; confidence intervals by bootstrap.

Usage

```
cramerV(
  x,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 4,
  bias.correct = FALSE,
  reportIncomplete = FALSE,
  verbose = FALSE,
  tolerance = 1e-16,
  ...
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations for one dimension of a two-way table.
`y`	If `x` is a vector, `y` is the vector of observations for the second dimension of a two-way table.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`bias.correct`	If `TRUE`, a bias correction is applied.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`verbose`	If `TRUE`, prints additional statistics.
`tolerance`	If the variance of the bootstrapped values are less than `tolerance`, NA is returned for the confidence interval values.
`...`	Additional arguments passed to `chisq.test`.

Details

Cramer's V is used as a measure of association between two nominal variables, or as an effect size for a chi-square test of association. For a 2 x 2 table, the absolute value of the phi statistic is the same as Cramer's V.

Because V is always positive, if `type="perc"`, the confidence interval will never cross zero. In this case, the confidence interval range should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When V is close to 0 or very large, or with small counts, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, Cramer's V. Or a small data frame consisting of Cramer's V, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/H_10.html

See Also

`phi`, `cohenW`, `cramerVFit`

Examples

```
### Example with table
data(Anderson)
fisher.test(Anderson)
cramerV(Anderson)

### Example with two vectors
Species = c(rep("Species1", 16), rep("Species2", 16))
Color   = c(rep(c("blue", "blue", "blue", "green"),4),
            rep(c("green", "green", "green", "blue"),4))
fisher.test(Species, Color)
cramerV(Species, Color)
```

---

cramerVFit *Cramer's V for chi-square goodness-of-fit tests*

Description

Calculates Cramer's V for a vector of counts and expected counts; confidence intervals by bootstrap.

Usage

```
cramerVFit(
  x,
  p = rep(1/length(x), length(x)),
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 4,
  reportIncomplete = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`x`	A vector of observed counts.
`p`	A vector of expected or default probabilities.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`verbose`	If `TRUE`, prints additional statistics.
`...`	Additional arguments passed to `chisq.test`.

Details

This modification of Cramer's V could be used to indicate an effect size in cases where a chi-square goodness-of-fit test might be used. It indicates the degree of deviation of observed counts from the expected probabilities.

In the case of equally-distributed expected frequencies, Cramer's V will be equal to 1 when all counts are in one category, and it will be equal to 0 when the counts are equally distributed across categories. This does not hold if the expected frequencies are not equally-distributed.

Because V is always positive, if `type="perc"`, the confidence interval will never cross zero, and should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When V is close to 0 or 1, or with small counts, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

In addition, the function will not return a confidence interval if there are zeros in any cell.

Value

A single statistic, Cramer's V. Or a small data frame consisting of Cramer's V, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/H_03.html

See Also

`cramerV`

Examples

```
### Equal probabilities example
### From https://rcompanion.org/handbook/H_03.html
nail.color = c("Red", "None", "White", "Green", "Purple", "Blue")
observed   = c( 19,    3,      1,       1,       2,        2    )
expected   = c( 1/6,   1/6,    1/6,     1/6,     1/6,      1/6  )
chisq.test(x = observed, p = expected)
cramerVFit(x = observed, p = expected)

### Unequal probabilities example
### From https://rcompanion.org/handbook/H_03.html
race = c("White", "Black", "American Indian", "Asian", "Pacific Islander",
          "Two or more races")
observed = c(20, 9, 9, 1, 1, 1)
expected = c(0.775, 0.132, 0.012, 0.054, 0.002, 0.025)
chisq.test(x = observed, p = expected)
cramerVFit(x = observed, p = expected)
```

```
### Examples of perfect and zero fits
cramerVFit(c(100, 0, 0, 0, 0))
cramerVFit(c(10, 10, 10, 10, 10))
```

`efronRSquared` *Efron's pseudo r-squared*

Description

Produces Efron's pseudo r-squared from certain models, or vectors of residuals, predicted values, and actual values. Alternately produces minimum maximum accuracy, mean absolute percent error, root mean square error, or coefficient of variation.

Usage

```
efronRSquared(
  model = NULL,
  actual = NULL,
  predicted = NULL,
  residual = NULL,
  statistic = "EfronRSquared",
  plotit = FALSE,
  digits = 3,
  ...
)
```

Arguments

`model`	A model of the class lm, glm, nls, betareg, gls, lme, lmerMod, lmerModLmerTest, glmmTMB, rq, loess, gam, negbin, glmRob, rlm, or mblm.
`actual`	A vector of actual y values
`predicted`	A vector of predicted values
`residual`	A vector of residuals
`statistic`	The statistic to produce. One of `"EfronRSquared"`, `"MinMaxAccuracy"`, `"MAE"`, `"MAPE"`, `"MSE"`, `"RMSE"`, `"NRMSE.Mean"`, `"CV"`.
`plotit`	If `TRUE`, produces plots of the predicted values vs. the actual values.
`digits`	The number of significant digits in the output.
`...`	Other arguments passed to `plot`.

Details

Efron's pseudo r-squared is calculated as 1 minus the residual sum of squares divided by the total sum of squares. For linear models (`lm` model objects), Efron's pseudo r-squared will be equal to r-squared.

This function produces the same statistics as does the `accuracy` function. While the `accuracy` function extracts values from a model object, this function allows for the manual entry of residual, predicted, or actual values.

It is recommended that the user consults the `accuracy` function for further details on these statistics, such as if the reported value is presented as a percentage or fraction.

If `model`is not supplied, two of the following need to passed to the function: `actual`, `predicted`, `residual`.

Note that, for some model objects, to extract residuals and predicted values on the original scale, a `type="response"` option needs to be added to the call, e.g. `residuals(model.object, type="response")`.

Value

A single statistic

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_16.html`

See Also

`accuracy`, `nagelkerke`

Examples

```
data(BrendonSmall)
BrendonSmall$Calories = as.numeric(BrendonSmall$Calories)
BrendonSmall$Calories2 = BrendonSmall$Calories ^ 2
model.1 = lm(Sodium ~ Calories + Calories2, data = BrendonSmall)

efronRSquared(model.1)

efronRSquared(model.1, statistic="MAPE")

efronRSquared(actual=BrendonSmall$Sodium, residual=model.1$residuals)
efronRSquared(residual=model.1$residuals, predicted=model.1$fitted.values)
efronRSquared(actual=BrendonSmall$Sodium, predicted=model.1$fitted.values)

summary(model.1)$r.squared
```

`epsilonSquared` *Epsilon-squared*

Description

Calculates epsilon-squared as an effect size statistic, following a Kruskal-Wallis test, or for a table with one ordinal variable and one nominal variable; confidence intervals by bootstrap

Usage

```
epsilonSquared(
  x,
  g = NULL,
  group = "row",
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  ...
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations of an ordinal variable.
`g`	If `x` is a vector, `g` is the vector of observations for the grouping, nominal variable.
`group`	If `x` is a table or matrix, `group` indicates whether the `"row"` or the `"column"` variable is the nominal, grouping variable.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`...`	Additional arguments passed to the `kruskal.test` function.

Details

Epsilon-squared is used as a measure of association for the Kruskal-Wallis test or for a two-way table with one ordinal and one nominal variable.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

Because epsilon-squared is always positive, if `type="perc"`, the confidence interval will never cross zero, and should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When epsilon-squared is close to 0 or very large, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, epsilon-squared. Or a small data frame consisting of epsilon-squared, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

King, B.M., P.J. Rosopa, and E.W. Minium. 2018. Statistical Reasoning in the Behavioral Sciences, 7th ed. Wiley.

https://rcompanion.org/handbook/F_08.html

See Also

`multiVDA`, `ordinalEtaSquared`

Examples

```
data(Breakfast)
library(coin)
chisq_test(Breakfast, scores = list("Breakfast" = c(-2, -1, 0, 1, 2)))
epsilonSquared(Breakfast)

data(PoohPiglet)
kruskal.test(Likert ~ Speaker, data = PoohPiglet)
epsilonSquared(x = PoohPiglet$Likert, g = PoohPiglet$Speaker)

### Same data, as matrix of counts
data(PoohPiglet)
XT = xtabs( ~ Speaker + Likert , data = PoohPiglet)
epsilonSquared(XT)
```

`freemanTheta` *Freeman's theta*

Description

Calculates Freeman's theta for a table with one ordinal variable and one nominal variable; confidence intervals by bootstrap.

Usage

```
freemanTheta(
  x,
  g = NULL,
  group = "row",
  verbose = FALSE,
  progress = FALSE,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations of an ordinal variable.
`g`	If `x` is a vector, `g` is the vector of observations for the grouping, nominal variable.
`group`	If `x` is a table or matrix, `group` indicates whether the `"row"` or the `"column"` variable is the nominal, grouping variable.
`verbose`	If `TRUE`, prints statistics for each comparison.
`progress`	If `TRUE`, prints a message as each comparison is conducted.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.

Details

Freeman's coefficent of differentiation (theta) is used as a measure of association for a two-way table with one ordinal and one nominal variable. See Freeman (1965).

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

Because theta is always positive, if `type="perc"`, the confidence interval will never cross zero, and should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When theta is close to 0 or very large, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, Freeman's theta. Or a small data frame consisting of Freeman's theta, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

Freeman, L.C. 1965. Elementary Applied Statistics for Students in Behavioral Science. Wiley.

`https://rcompanion.org/handbook/H_11.html`

See Also

`epsilonSquared`

Examples

```
data(Breakfast)
library(coin)
chisq_test(Breakfast, scores = list("Breakfast" = c(-2, -1, 0, 1, 2)))
freemanTheta(Breakfast)

### Example from Freeman (1965), Table 10.6
Counts = c(1, 2, 5, 2, 0, 10, 5, 5, 0, 0, 0, 0, 2, 2, 1, 0, 0, 0, 2, 3)
Matrix = matrix(Counts, byrow=TRUE, ncol=5,
                dimnames = list(Marital.status = c("Single", "Married",
                                                   "Widowed", "Divorced"),
                                Social.adjustment = c("5","4","3","2","1")))
Matrix
freemanTheta(Matrix)

### Example after Kruskal Wallis test
data(PoohPiglet)
kruskal.test(Likert ~ Speaker, data = PoohPiglet)
freemanTheta(x = PoohPiglet$Likert, g = PoohPiglet$Speaker)
```

```
### Same data, as table of counts
data(PoohPiglet)
XT = xtabs( ~ Speaker + Likert , data = PoohPiglet)
freemanTheta(XT)

### Example from Freeman (1965), Table 10.7
Counts = c(52, 28, 40, 34, 7, 9, 16, 10, 8, 4, 10, 9, 12,6, 7, 5)
Matrix = matrix(Counts, byrow=TRUE, ncol=4,
                dimnames = list(Preferred.trait = c("Companionability",
                                                    "PhysicalAppearance",
                                                    "SocialGrace",
                                                    "Intelligence"),
                                Family.income = c("4", "3", "2", "1")))
Matrix
freemanTheta(Matrix, verbose=TRUE)
```

`fullPTable` *Convert a lower triangle matrix to a full matrix*

Description

Converts a lower triangle matrix to a full matrix.

Usage

```
fullPTable(PT)
```

Arguments

`PT`	A lower triangle matrix.

Details

This function is useful to convert a lower triangle matrix of p-values from a pairwise test to a full matrix. A full matrix can be passed to `multcompLetters` in the `multcompView` package to produce a compact letter display.

Value

A full matrix.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_08.html`

See Also

`cldList`

Examples

```
### Example with pairwise.wilcox.test
data(BrendonSmall)
BrendonSmall$Instructor = factor(BrendonSmall$Instructor,
                          levels = c('Brendon Small', 'Jason Penopolis',
                                     'Paula Small', 'Melissa Robbins',
                                     'Coach McGuirk'))
P   = pairwise.wilcox.test(x = BrendonSmall$Score, g = BrendonSmall$Instructor)
PT  = P$p.value
PT
PT1 = fullPTable(PT)
PT1
library(multcompView)
multcompLetters(PT1)
```

`groupwiseCMH` *Post-hoc tests for Cochran-Mantel-Haenszel test*

Description

Conducts groupwise tests of association on a three-way contingency table.

Usage

```
groupwiseCMH(
  x,
  group = 3,
  fisher = TRUE,
  gtest = FALSE,
  chisq = FALSE,
  method = "fdr",
  correct = "none",
  digits = 3,
  ...
)
```

Arguments

`x`	A three-way contingency table.
`group`	The dimension of the table to use as the grouping variable. Will be `1`, `2`, or `3`.
`fisher`	If `TRUE`, conducts Fisher exact test.
`gtest`	If `TRUE`, conducts G test of association.

`chisq`	If `TRUE`, conducts Chi-square test of association.
`method`	The method to use to adjust p-values. See `?p.adjust`.
`correct`	The correction to apply to the G test. See `GTest`.
`digits`	The number of digits for numbers in the output.
`...`	Other arguments passed to `chisq.test` or `GTest`.

Details

If more than one of `fisher`, `gtest`, or `chisq` is set to `TRUE`, only one type of test of association will be conducted.

Value

A data frame of groups, test used, p-values, and adjusted p-values.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/H_06.html`

See Also

`nominalSymmetryTest`, `pairwiseMcnemar`, `pairwiseNominalIndependence`, `pairwiseNominalMatrix`

Examples

```
### Post-hoc for Cochran-Mantel-Haenszel test
data(AndersonBias)
Table = xtabs(Count ~ Gender + Result + County,
              data=AndersonBias)
ftable(Table)
mantelhaen.test(Table)
groupwiseCMH(Table,
             group   = 3,
             fisher  = TRUE,
             gtest   = FALSE,
             chisq   = FALSE,
             method  = "fdr",
             correct = "none",
             digits  = 3)
```

`groupwiseGeometric` *Groupwise geometric means and confidence intervals*

Description

Calculates geometric means and confidence intervals for groups.

Usage

```
groupwiseGeometric(
  formula = NULL,
  data = NULL,
  var = NULL,
  group = NULL,
  conf = 0.95,
  na.rm = TRUE,
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`var`	The measurement variable to use. The name is in double quotes.
`group`	The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)
`conf`	The confidence interval to use.
`na.rm`	If `TRUE`, removes NA values in the measurement variable.
`digits`	The number of significant figures to use in output.
`...`	Other arguments. Not currently used.

Details

The input should include either `formula` and `data`; or `data`, `var`, and `group`. (See examples).

The function computes means, standard deviations, standard errors, and confidence intervals on log-transformed values. Confidence intervals are calculated in the traditional manner with the t-distribution on the transformed values, and then back-transforms the confidence interval limits. These statistics assume that the data are log-normally distributed. For data not meeting this assumption, medians and confidence intervals by bootstrap may be more appropriate.

Value

A data frame of geometric means, standard deviations, standard errors, and confidence intervals.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The variables on the right side are used for the grouping variables.

Results for ungrouped (one-sample) data can be obtained by either setting the right side of the formula to 1, e.g. y ~ 1, or by setting `group=NULL`.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/C_03.html`

See Also

`groupwiseMean`, `groupwiseMedian`, `groupwiseHuber`

Examples

```
### Example with formula notation
data(Catbus)
groupwiseGeometric(Steps ~ Gender + Teacher,
                   data   = Catbus)

### Example with variable notation
data(Catbus)
groupwiseGeometric(data   = Catbus,
                   var    = "Steps",
                   group  = c("Gender", "Teacher"))
```

`groupwiseHuber` *Groupwise Huber M-estimators and confidence intervals*

Description

Calculates Huber M-estimator and confidence intervals for groups.

Usage

```
groupwiseHuber(
  formula = NULL,
  data = NULL,
  var = NULL,
  group = NULL,
  conf.level = 0.95,
  ci.type = "wald",
```

```
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`var`	The measurement variable to use. The name is in double quotes.
`group`	The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)
`conf.level`	The confidence interval to use.
`ci.type`	The type of confidence interval to use. Can be `"wald"` or `"boot"`. See `HuberM` for details.
`digits`	The number of significant figures to use in output.
`...`	Other arguments passed to the `HuberM` function.

Details

A wrapper for the `DescTools::HuberM` function to allow easy output for multiple groups.

The input should include either `formula` and `data`; or `data`, `var`, and `group`. (See examples).

Results for ungrouped (one-sample) data can be obtained by either setting the right side of the formula to 1, e.g. y ~ 1, or by setting `group=NULL`.

Value

A data frame of requested statistics by group.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The variables on the right side are used for the grouping variables.

It is recommended to remove `NA` values before using this function. At the time of writing, `NA` values will cause the function to fail if confidence intervals are requested.

At the time of writing, the `ci.type="boot"` option produces `NA` results. This is a result from the `DescTools::HuberM` function.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/rcompanion/d_08a.html`

See Also

`groupwiseMean`, `groupwiseMedian`, `groupwiseGeometric`

Examples

```
### Example with formula notation
data(Catbus)
groupwiseHuber(Steps ~ Teacher + Gender,
               data      = Catbus,
               ci.type   = "wald")

### Example with variable notation
data(Catbus)
groupwiseHuber(data      = Catbus,
               var       = "Steps",
               group     = c("Teacher", "Gender"),
               ci.type   = "wald")

### Example with NA value and without confidence intervals
data(Catbus)
Catbus1 = Catbus
Catbus1[1, 'Steps'] = NA
groupwiseHuber(Steps ~ Teacher + Gender,
               data      = Catbus1,
               conf.level   = NA)
```

`groupwiseMean` *Groupwise means and confidence intervals*

Description

Calculates means and confidence intervals for groups.

Usage

```
groupwiseMean(
  formula = NULL,
  data = NULL,
  var = NULL,
  group = NULL,
  trim = 0,
  na.rm = FALSE,
  conf = 0.95,
  R = 5000,
  boot = FALSE,
  traditional = TRUE,
  normal = FALSE,
```

```
    basic = FALSE,
    percentile = FALSE,
    bca = FALSE,
    digits = 3,
    ...
  )
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`var`	The measurement variable to use. The name is in double quotes.
`group`	The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)
`trim`	The proportion of observations trimmed from each end of the values before the mean is calculated. (As in `mean()`)
`na.rm`	If `TRUE`, `NA` values are removed during calculations. (As in `mean()`)
`conf`	The confidence interval to use.
`R`	The number of bootstrap replicates to use for bootstrapped statistics.
`boot`	If `TRUE`, includes the mean of the bootstrapped means. This can be used as an estimate of the mean for the group.
`traditional`	If `TRUE`, includes the traditional confidence intervals for the group means, using the t-distribution. If `trim` is not 0, the traditional confidence interval will produce `NA`. Likewise, if there are `NA` values that are not removed, the traditional confidence interval will produce `NA`.
`normal`	If `TRUE`, includes the normal confidence intervals for the group means by bootstrap. See `boot.ci`.
`basic`	If `TRUE`, includes the basic confidence intervals for the group means by bootstrap. See `boot.ci`.
`percentile`	If `TRUE`, includes the percentile confidence intervals for the group means by bootstrap. See `boot.ci`.
`bca`	If `TRUE`, includes the BCa confidence intervals for the group means by bootstrap. See `boot.ci`.
`digits`	The number of significant figures to use in output.
`...`	Other arguments passed to the `boot` function.

Details

The input should include either `formula` and `data`; or `data`, `var`, and `group`. (See examples).

Results for ungrouped (one-sample) data can be obtained by either setting the right side of the formula to 1, e.g. y ~ 1, or by setting `group=NULL` when using `var`.

Value

A data frame of requested statistics by group.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The variables on the right side are used for the grouping variables.

In general, it is advisable to handle `NA` values before using this function. With some options, the function may not handle missing values well, or in the manner desired by the user. In particular, if `bca=TRUE` and there are `NA` values, the function may fail.

For a traditional method to calculate confidence intervals on trimmed means, see Rand Wilcox, Introduction to Robust Estimation and Hypothesis Testing.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/C_03.html`

See Also

`groupwiseMedian`, `groupwiseHuber`, `groupwiseGeometric`

Examples

```
### Example with formula notation
data(Catbus)
groupwiseMean(Steps ~ Teacher + Gender,
              data        = Catbus,
              traditional = FALSE,
              percentile  = TRUE)

### Example with variable notation
data(Catbus)
groupwiseMean(data        = Catbus,
              var         = "Steps",
              group       = c("Teacher", "Gender"),
              traditional = FALSE,
              percentile  = TRUE)
```

`groupwiseMedian` *Groupwise medians and confidence intervals*

Description

Calculates medians and confidence intervals for groups.

Usage

```
groupwiseMedian(
  formula = NULL,
  data = NULL,
  var = NULL,
  group = NULL,
  conf = 0.95,
  R = 5000,
  boot = FALSE,
  pseudo = FALSE,
  basic = FALSE,
  normal = FALSE,
  percentile = FALSE,
  bca = TRUE,
  wilcox = FALSE,
  exact = FALSE,
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`var`	The measurement variable to use. The name is in double quotes.
`group`	The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)
`conf`	The confidence interval to use.
`R`	The number of bootstrap replicates to use for bootstrapped statistics.
`boot`	If `TRUE`, includes the mean of the bootstrapped medians. This can be used as an estimate of the median for the group.
`pseudo`	If `TRUE`, includes the pseudo median from `wilcox.test`.
`basic`	If `TRUE`, includes the basic confidence intervals for the group means by bootstrap. See `boot::boot.ci`.
`normal`	If `TRUE`, includes the normal confidence intervals for the group means by bootstrap. See `boot::boot.ci`.

`percentile`	If `TRUE`, includes the percentile confidence intervals for the group means by bootstrap. See `boot::boot.ci`.
`bca`	If `TRUE`, includes the BCa confidence intervals for the group means by bootstrap. See `boot::boot.ci`.
`wilcox`	If `TRUE`, includes the wilcox confidence intervals from `stats::wilcox.test`.
`exact`	If `TRUE`, includes the "exact" confidence intervals from `DescTools::MedianCI`.
`digits`	The number of significant figures to use in output.
`...`	Other arguments passed to the `boot` function.

Details

The input should include either `formula` and `data`; or `data`, `var`, and `group`. (See examples).

With some options, the function may not handle missing values well. This seems to happen particularly with `bca = TRUE`.

Value

A data frame of requested statistics by group.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The variables on the right side are used for the grouping variables.

Results for ungrouped (one-sample) data can be obtained by either setting the right side of the formula to 1, e.g. y ~ 1, or by setting `group=NULL`.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/E_04.html`

See Also

`groupwiseMean`, `groupwiseHuber`, `groupwiseGeometric`

Examples

```
### Example with formula notation
data(Catbus)
groupwiseMedian(Steps ~ Teacher + Gender,
                data        = Catbus,
                bca         = FALSE,
                percentile  = TRUE,
                R           = 1000)

### Example with variable notation
```

```
data(Catbus)
groupwiseMedian(data         = Catbus,
                var         = "Steps",
                group       = c("Teacher", "Gender"),
                bca         = FALSE,
                percentile  = TRUE,
                R           = 1000)
```

`groupwisePercentile` *Groupwise percentiles and confidence intervals*

Description

Calculates percentiles and confidence intervals for groups.

Usage

```
groupwisePercentile(
  formula = NULL,
  data = NULL,
  var = NULL,
  group = NULL,
  conf = 0.95,
  tau = 0.5,
  type = 7,
  R = 5000,
  boot = FALSE,
  basic = FALSE,
  normal = FALSE,
  percentile = FALSE,
  bca = TRUE,
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`var`	If no formula is given, the measurement variable to use. The name is in double quotes.
`group`	The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)
`conf`	The confidence interval to use.

`tau`	The percentile to use, expressed as a quantile, e.g. 0.5 for median, 0.25 for 25th percentile.
`type`	The `type` value passed to the `quantile` function.
`R`	The number of bootstrap replicates to use for bootstrapped statistics.
`boot`	If `TRUE`, includes the mean of the bootstrapped percentile. This can be used as an estimate of the percentile for the group.
`basic`	If `TRUE`, includes the basic confidence intervals for the group means by bootstrap. See `boot.ci`.
`normal`	If `TRUE`, includes the normal confidence intervals for the group means by bootstrap. See `boot.ci`.
`percentile`	If `TRUE`, includes the percentile confidence intervals for the group means by bootstrap. See `boot.ci`.
`bca`	If `TRUE`, includes the BCa confidence intervals for the group means by bootstrap. See `boot.ci`.
`digits`	The number of significant figures to use in output.
`...`	Other arguments passed to the `boot` function.

Details

The input should include either `formula` and `data`; or `data`, `var`, and `group`. (See examples).

With some options, the function may not handle missing values well. This seems to happen particularly with `bca = TRUE`.

Value

A data frame of requested statistics by group

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The variables on the right side are used for the grouping variables.

Results for ungrouped (one-sample) data can be obtained by either setting the right side of the formula to 1, e.g. y ~ 1, or by setting `group=NULL`.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_15.html`

See Also

`groupwiseMean`, `groupwiseHuber`, `groupwiseGeometric`, `groupwiseMedian`

Examples

```
### Example with formula notation
data(Catbus)
groupwisePercentile(Steps ~ Teacher + Gender,
                    data        = Catbus,
                    tau         = 0.25,
                    bca         = FALSE,
                    percentile  = TRUE,
                    R           = 1000)

### Example with variable notation
data(Catbus)
groupwisePercentile(data         = Catbus,
                    var         = "Steps",
                    group       = c("Teacher", "Gender"),
                    tau         = 0.25,
                    bca         = FALSE,
                    percentile  = TRUE,
                    R           = 1000)
```

`groupwiseSum` *Groupwise sums*

Description

Calculates sums for groups.

Usage

```
groupwiseSum(
  formula = NULL,
  data = NULL,
  var = NULL,
  group = NULL,
  digits = NULL,
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`var`	The measurement variable to use. The name is in double quotes.
`group`	The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)

`digits`	The number of significant figures to use in output. The default is `NULL`, which results in no rounding of values.
`...`	Other arguments passed to the `sum` function

Details

The input should include either `formula` and `data`; or `data`, `var`, and `group`. (See examples).

Value

A data frame of statistics by group.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The variables on the right side are used for the grouping variables.

Beginning in version 2.0, there is no rounding of results by default. Rounding results can cause confusion if the user is expecting exact sums.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

See Also

`groupwiseMean`, `groupwiseMedian`, `groupwiseHuber`, `groupwiseGeometric`

Examples

```
### Example with formula notation
data(AndersonBias)
groupwiseSum(Count ~ Result + Gender,
             data        = AndersonBias)

### Example with variable notation
data(AndersonBias)
groupwiseSum(data        = AndersonBias,
             var         = "Count",
             group       = c("Result", "Gender"))
```

`HayleySmith` *Hypothetical data for responses about adopting lawn care practices*

Description

A data frame in long form with yes/no responses for four lawn care practices for each of 14 respondents. Hypothetical data.

Usage

```
HayleySmith
```

Format

An object of class `data.frame` with 56 rows and 3 columns.

Source

https://rcompanion.org/handbook/H_05.html

`kendallW` *Kendall's W with bootstrapped confidence interval*

Description

Calculates Kendall's W coefficient of concordance, which can be used as an effect size statistic for unreplicated complete block design such as where Friedman's test might be used. This function is a wrapper for the `KendallW` function in the `DescTools` package, with the addition of bootstrapped confidence intervals.

Usage

```
kendallW(
  x,
  correct = TRUE,
  na.rm = FALSE,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  ...
)
```

Arguments

`x`	A k x m matrix or table, with k treatments in rows and m raters or blocks in columns.
`correct`	Passed to `KendallW`.
`na.rm`	Passed to `KendallW`.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `KendallW` function.

Details

See the `KendallW` function in the `DescTools` package for details.

When W is close to 0 or very large, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Because W is always positive, if `type="perc"`, the confidence interval will never cross zero, and should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When producing confidence intervals by bootstrap, this function treats each rater or block as an observation. It is not clear to the author if this approach produces accurate confidence intervals, but it appears to be reasonable.

Value

A single statistic, W. Or a small data frame consisting of W, and the lower and upper confidence limits.

Acknowledgments

My thanks to Indrajeet Patil, author of `ggstatsplot`, and `groupedstats` for help in the inspiring and coding of this function.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_10.html`

Examples

```
data(BobBelcher)
Table = xtabs(Likert ~ Instructor + Rater, data = BobBelcher)
kendallW(Table)
```

`mangiaficoD` *Mangiafico's d*

Description

Calculates Mangiafico's d, which is the difference in medians divided by the pooled median absolute deviation, with confidence intervals by bootstrap

Usage

```
mangiaficoD(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  reportIncomplete = FALSE,
  verbose = FALSE,
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.

`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`verbose`	If `TRUE`, reports the median difference and MAD.
`digits`	The number of significant digits in the output.
`...`	Other arguments passed to `mad()`.

Details

Mangiafico's d is an appropriate effect size statistic where Mood's median test, or another test comparing two medians, might be used. Note that the response variable is treated as at least interval.

For normal samples, the result will be somewhat similar to Cohen's d.

The input should include either `formula` and `data`; or `x`, and `y`. If there are more than two groups, only the first two groups are used.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, d is positive. When the data in the second group are greater than in the first group, d is negative.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

When d is close to 0 or close to 1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, d. Or a small data frame consisting of d, and the lower and upper confidence limits.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_05.html`

See Also

`multiMangiaficoD`

Examples

```
data(Catbus)
mangiaficoD(Steps ~ Gender, data=Catbus, verbose=TRUE)

Nadja = c(5,5,6,6,6,7,7,11,11,11)
Nandor = c(0,1,2,3,4,5,6,7,8,9,10,11)
mangiaficoD(x = Nadja, y = Nandor, verbose=TRUE)
```

`Monarchs` *Hypothetical data for monarch butterflies in gardens*

Description

A data frame of the number of monarch butterflies in three gardens. Hypothetical data.

Usage

```
Monarchs
```

Format

An object of class `data.frame` with 24 rows and 2 columns.

Source

https://rcompanion.org/handbook/J_01.html

`multiMangiaficoD` *Mangiafico's d*

Description

Calculates Mangiafico's d, which is the difference in medians divided by the pooled median absolute deviation, for several groups in a pairwise manner.

Usage

```
multiMangiaficoD(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable.
`g`	If no formula is given, the grouping variable.
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `mad()` function.

Details

Mangiafico's d is an appropriate effect size statistic where Mood's median test, or another test comparing two medians, might be used. Note that the response variable is treated as at least interval.

When the data in the first group are greater than in the second group, d is positive. When the data in the second group are greater than in the first group, d is negative.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

Value

A list containing a data frame of pairwise statistics, and the comparison with the most extreme value of the statistic.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_09.html`

See Also

`mangiaficoD`

Examples

```
data(Catbus)
multiMangiaficoD(Steps ~ Teacher, data=Catbus)
```

`multiVDA` *Pairwise Vargha and Delaney's A and Cliff's delta*

Description

Calculates Vargha and Delaney's A (VDA), Cliff's delta (CD), and the Glass rank biserial coefficient, rg, for several groups in a pairwise manner.

Usage

```
multiVDA(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  statistic = "VDA",
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable.
`g`	If no formula is given, the grouping variable.
`statistic`	One of `"VDA"`, `"CD"`, or `"rg"`. This determines which statistic will be evaluated to determine the comparison with the most divergent groups.
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `wilcox.test` function.

Details

VDA and CD are effect size statistic appropriate in cases where a Wilcoxon-Mann-Whitney test might be used. Here, the pairwise approach would be used in cases where a Kruskal-Wallis test might be used. VDA ranges from 0 to 1, with 0.5 indicating stochastic equality, and 1 indicating that the first group dominates the second. CD ranges from -1 to 1, with 0 indicating stochastic equality, and 1 indicating that the first group dominates the second. rg ranges from -1 to 1, depending on sample size, with 0 indicating no effect, and a positive result indicating that values in the first group are greater than in the second.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

In the function output, `VDA.m` is the greater of VDA or 1-VDA. `CD.m` is the absolute value of CD. `rg.m` is the absolute value of rg.

The function calculates VDA and Cliff's delta from the "W" U statistic from the `wilcox.test` function. Specifically, `VDA = U/(n1*n2); CD = (VDA-0.5)*2`.

rg is calculated as 2 times the difference of mean of ranks for each group divided by the total sample size. It appears that rg is equivalent to Cliff's delta.

The input should include either `formula` and `data`; or `var`, and `group`.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

Value

A list containing a data frame of pairwise statistics, and the comparison with the most extreme value of the chosen statistic.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_08.html`

See Also

`vda`, `cliffDelta`

Examples

```
data(PoohPiglet)
multiVDA(Likert ~ Speaker, data=PoohPiglet)
```

`nagelkerke` *Pseudo r-squared measures for various models*

Description

Produces McFadden, Cox and Snell, and Nagelkerke pseudo r-squared measures, along with p-values, for models.

Usage

```
nagelkerke(fit, null = NULL, restrictNobs = FALSE)
```

Arguments

`fit`	The fitted model object for which to determine pseudo r-squared.
`null`	The null model object against which to compare the fitted model object. The null model must be nested in the fitted model to be valid. Specifying the null is optional for some model object types and is required for others.
`restrictNobs`	If `TRUE`, limits the observations for the null model to those used in the fitted model. Works with only some model object types.

Details

Pseudo R-squared values are not directly comparable to the R-squared for OLS models. Nor can they be interpreted as the proportion of the variability in the dependent variable that is explained by model. Instead pseudo R-squared measures are relative measures among similar models indicating how well the model explains the data.

Cox and Snell is also referred to as ML. Nagelkerke is also referred to as Cragg and Uhler.

Model objects accepted are lm, glm, gls, lme, lmer, lmerTest, nls, clm, clmm, vglm, glmer, glmmTMB, negbin, zeroinfl, betareg, and rq.

Model objects that require the null model to be defined are nls, lmer, glmer, and clmm. Other objects use the `update` function to define the null model.

Likelihoods are found using ML (`REML = FALSE`).

The fitted model and the null model should be properly nested. That is, the terms of one need to be a subset of the the other, and they should have the same set of observations. One issue arises when there are `NA` values in one variable but not another, and observations with `NA` are removed in the model fitting. The result may be fitted and null models with different sets of observations. Setting `restrictNobs` to `TRUE` ensures that only observations in the fit model are used in the null model. This appears to work for `lm` and some `glm` models, but causes the function to fail for other model object types.

Some pseudo R-squared measures may not be appropriate or useful for some model types.

Calculations are based on log likelihood values for models. Results may be different than those based on deviance.

Value

A list of six objects describing the models used, the pseudo r-squared values, the likelihood ratio test for the model, the number of observations for the models, messages, and any warnings.

Acknowledgments

My thanks to Jan-Herman Kuiper of Keele University for suggesting the `restrictNobs` fix.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/G_10.html`

See Also

`efronRSquared`

Examples

```
### Logistic regression example
data(AndersonBias)
model = glm(Result ~ County + Gender + County:Gender,
           weight = Count,
           data = AndersonBias,
           family = binomial(link="logit"))
nagelkerke(model)

### Quadratic plateau example
### With nls, the  null needs to be defined
data(BrendonSmall)
quadplat = function(x, a, b, clx) {
          ifelse(x  < clx, a + b * x   + (-0.5*b/clx) * x   * x,
                           a + b * clx + (-0.5*b/clx) * clx * clx)}
model = nls(Sodium ~ quadplat(Calories, a, b, clx),
            data = BrendonSmall,
            start = list(a   = 519,
                         b   = 0.359,
                         clx = 2304))
nullfunct = function(x, m){m}
null.model = nls(Sodium ~ nullfunct(Calories, m),
             data = BrendonSmall,
             start = list(m   = 1346))
nagelkerke(model, null=null.model)
```

`nagelkerkeHermite` *[Defunct!] Pseudo r-squared measures for hermite models*

Description

Defunct. Produces McFadden, Cox and Snell, and Nagelkerke pseudo R-squared measures, along with p-value for the model, for hermite regression objects.

Usage

```
nagelkerkeHermite(...)
```

Arguments

`...`	Anything.

`nominalSymmetryTest` *Exact and McNemar symmetry tests for paired contingency tables*

Description

Conducts an omnibus symmetry test for a paired contingency table and then post-hoc pairwise tests. This is similar to McNemar and McNemar-Bowker tests in use.

Usage

```
nominalSymmetryTest(x, method = "fdr", digits = 3, exact = FALSE, ...)
```

Arguments

`x`	A two-way contingency table. It must be square. It can have two or more levels for each dimension.
`method`	The method to adjust multiple p-values. See `stats::p.adjust`.
`digits`	The number of significant digits in the output.
`exact`	If `TRUE`, uses the `binom.test` function. If `FALSE`, uses the `mcnemar.test` function.
`...`	Additional arguments

Details

The omnibus McNemar test may fail when there are zeros in critical cells.

Currently, the `exact=TRUE` with a table greater than 2 x 2 will not produce an omnibus test result.

Value

A list containing: a data frame of results of the global test; a data frame of results of the pairwise results; and a data frame mentioning the p-value adjustment method.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/H_05.html`

See Also

`pairwiseMcnemar`, `groupwiseCMH`, `pairwiseNominalIndependence`, `pairwiseNominalMatrix`

Examples

```
### 2 x 2 repeated matrix example
data(AndersonRainBarrel)
nominalSymmetryTest(AndersonRainBarrel)

### 3 x 3 repeated matrix example
data(AndersonRainGarden)
nominalSymmetryTest(AndersonRainGarden,
                    exact = FALSE)
```

`Nurseries` *Data for proportion of good practices followed by plant nuseries*

Description

A data frame with two variables: size of plant nursery in hectares, and proportion of good practices followed by the nursery

Usage

```
Nurseries
```

Format

An object of class `data.frame` with 38 rows and 2 columns.

Source

Mangiafico, S.S., Newman, J.P., Mochizuki, M.J., and Zurawski, D. (2008). Adoption of sustainable practices to protect and conserve water resources in container nurseries with greenhouse facilities. Acta horticulturae 797, 367-372.

`oneSampleDominance` *Dominance statistic for one-sample data*

Description

Calculates a dominance effect size statistic compared with a theoretical median for one-sample data with confidence intervals by bootstrap

Usage

```
oneSampleDominance(
  x,
  mu = 0,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  na.rm = TRUE,
  ...
)
```

Arguments

`x`	A vector of numeric values.
`mu`	The median against which to compare the values.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`na.rm`	If `TRUE`, removes `NA` values from the input vector `x`.
`...`	Additional arguments.

Details

The calculated `Dominance` statistic is simply the proportion of observations greater than `mu` minus the the proportion of observations less than `mu`.

It will range from -1 to 1, with 0 indicating that the median is equal to `mu`, and 1 indicating that the observations are all greater in value than `mu`, and -1 indicating that the observations are all less in value than `mu`.

This statistic is appropriate for truly ordinal data, and could be considered an effect size statistic for a one-sample sign test.

Ordered category data need to re-coded as numeric, e.g. as with `as.numeric(Ordinal.variable)`.

When the statistic is close to 1 or close to -1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

VDA is the analogous statistic, converted to a probability, ranging from 0 to 1, specifically, `VDA = Dominance / 2 + 0.5`.

Value

A small data frame consisting of descriptive statistics, the dominance statistic, and potentially the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_03.html`

See Also

`pairedSampleDominance`, `cliffDelta`, `vda`

Examples

```
data(Catbus)
library(DescTools)
SignTest(Catbus$Rating, mu=5.5)
oneSampleDominance(Catbus$Rating, mu=5.5)
```

`ordinalEtaSquared` *Eta-squared for ordinal variables*

Description

Calculates eta-squared as an effect size statistic, following a Kruskal-Wallis test, or for a table with one ordinal variable and one nominal variable; confidence intervals by bootstrap.

Usage

```
ordinalEtaSquared(
  x,
  g = NULL,
  group = "row",
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  ...
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations of an ordinal variable.
`g`	If `x` is a vector, `g` is the vector of observations for the grouping, nominal variable.
`group`	If `x` is a table or matrix, `group` indicates whether the `"row"` or the `"column"` variable is the nominal, grouping variable.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`...`	Additional arguments passed to the `kruskal.test` function.

Details

Eta-squared is used as a measure of association for the Kruskal-Wallis test or for a two-way table with one ordinal and one nominal variable.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

Because eta-squared is always positive, if `type="perc"`, the confidence interval will never cross zero, and should not be used for statistical inference. However, if `type="norm"`, the confidence interval may cross zero.

When eta-squared is close to 0 or very large, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, eta-squared. Or a small data frame consisting of eta-squared, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

Cohen, B.H. 2013. Explaining Psychological Statistics, 4th ed. Wiley.

`https://rcompanion.org/handbook/F_08.html`

See Also

`freemanTheta`, `epsilonSquared`

Examples

```
data(Breakfast)
library(coin)
chisq_test(Breakfast, scores = list("Breakfast" = c(-2, -1, 0, 1, 2)))
ordinalEtaSquared(Breakfast)

data(PoohPiglet)
kruskal.test(Likert ~ Speaker, data = PoohPiglet)
ordinalEtaSquared(x = PoohPiglet$Likert, g = PoohPiglet$Speaker)

### Same data, as matrix of counts
data(PoohPiglet)
XT = xtabs( ~ Speaker + Likert , data = PoohPiglet)
ordinalEtaSquared(XT)
```

`pairedSampleDominance` *Dominance statistic for two-sample paired data*

Description

Calculates a dominance effect size statistic for two-sample paired data with confidence intervals by bootstrap

Usage

```
pairedSampleDominance(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  na.rm = TRUE,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`na.rm`	If `TRUE`, removes `NA` values from the input vectors or data frame.
`...`	Additional arguments.

Details

The calculated `Dominance` statistic is simply the proportion of observations in `x` greater the paired observations in `y`, minus the proportion of observations in `x` less than the paired observations in `y`

It will range from -1 to 1, with and 1 indicating that the all the observations in `x` are greater than the paired observations in `y`, and -1 indicating that the all the observations in `y` are greater than the paired observations in `x`.

The input should include either `formula` and `data`; or `x`, and `y`. If there are more than two groups, only the first two groups are used.

This statistic is appropriate for truly ordinal data, and could be considered an effect size statistic for a two-sample paired sign test.

Ordered category data need to re-coded as numeric, e.g. as with `as.numeric(Ordinal.variable)`.

When the statistic is close to 1 or close to -1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

VDA is the analogous statistic, converted to a probability, ranging from 0 to 1, specifically, `VDA = Dominance / 2 + 0.5`

Value

A small data frame consisting of descriptive statistics, the dominance statistic, and potentially the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_07.html`

See Also

`oneSampleDominance`, `vda`, `cliffDelta`

Examples

```
data(Pooh)
Time.1 = Pooh$Likert[Pooh$Time == 1]
Time.2 = Pooh$Likert[Pooh$Time == 2]
library(DescTools)
SignTest(x = Time.1, y = Time.2)
pairedSampleDominance(x = Time.1, y = Time.2)
pairedSampleDominance(Likert ~ Time, data=Pooh)
```

`pairwiseDifferences` *[Defunct!] Pairwise differences for unreplicated CBD*

Description

Defunct. Calculates the differences in the response variable for each pair of levels of a grouping variable in an unreplicated complete block design.

Usage

```
pairwiseDifferences(...)
```

Arguments

`...`	Anything.

`pairwiseMcnemar` *Pairwise McNemar and related tests for Cochran Q test post-hoc*

Description

Conducts pairwise McNemar, exact, and permutation tests as a post-hoc to Cochran Q test.

Usage

```
pairwiseMcnemar(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  block = NULL,
  test = "exact",
  method = "fdr",
  digits = 3,
  correct = FALSE
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group | block.
`data`	The data frame to use.
`x`	The response variable.
`g`	The grouping variable.
`block`	The blocking variable.
`test`	If `"exact"`, conducts an exact test of symmetry analogous to a McNemar test. If `"mcnemar"`, conducts a McNemar test of symmetry. If `"permutation"`, conducts a permutation test analogous to a McNemar test.
`method`	The method for adjusting multiple p-values. See `p.adjust`.
`digits`	The number of significant digits in the output.
`correct`	If `TRUE`, applies a continuity correction for the McNemar test.

Details

The component tables for the pairwise tests must be of size 2 x 2.

The input should include either `formula` and `data`; or `x`, `g`, and `block`.

Value

A list containing: a data frame of results of the global test; a data frame of results of the pairwise results; and a data frame mentioning the p-value adjustment method.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable. The second variable on the right side is used for the blocking variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/H_07.html`

See Also

`nominalSymmetryTest`, `groupwiseCMH`, `pairwiseNominalIndependence`, `pairwiseNominalMatrix`

Examples

```
### Cochran Q post-hoc example
data(HayleySmith)
library(DescTools)
CochranQTest(Response ~ Practice | Student,
             data = HayleySmith)
HayleySmith$Practice = factor(HayleySmith$Practice,
                          levels = c("MowHeight", "SoilTest",
                                     "Clippings", "Irrigation"))
PT = pairwiseMcnemar(Response ~ Practice | Student,
                     data    = HayleySmith,
                     test    = "exact",
                     method  = "fdr",
                     digits  = 3)
PT
PT = PT$Pairwise
cldList(comparison = PT$Comparison,
        p.value    = PT$p.adjust,
        threshold  = 0.05)
```

`pairwiseMedianMatrix` *Pairwise Mood's median tests with matrix output*

Description

Conducts pairwise Mood's median tests across groups.

Usage

```
pairwiseMedianMatrix(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  digits = 4,
  method = "fdr",
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	The response variable as a vector.
`g`	The grouping variable as a vector.
`digits`	The number of significant digits to round output.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.
`...`	Additional arguments passed to `coin::median_test`.

Details

The input should include either `formula` and `data`; or `x`, and `g`.

Mood's median test compares medians among two or more groups. See `https://rcompanion.org/handbook/F_09.html` for futher discussion of this test.

The `pairwiseMedianMatrix` function can be used as a post-hoc method following an omnibus Mood's median test. It passes the data for pairwise groups to `coin::median_test`.

The matrix output can be converted to a compact letter display, as in the example.

Value

A list consisting of: a matrix of p-values; the p-value adjustment method; a matrix of adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

https://rcompanion.org/handbook/F_09.html

See Also

pairwiseMedianTest

Examples

```
data(PoohPiglet)
PoohPiglet$Speaker = factor(PoohPiglet$Speaker,
                          levels = c("Pooh", "Tigger", "Piglet"))
PT = pairwiseMedianMatrix(Likert ~ Speaker,
                          data   = PoohPiglet,
                          exact  = NULL,
                          method = "fdr")$Adjusted
PT
library(multcompView)
multcompLetters(PT,
                compare="<",
                threshold=0.05,
                Letters=letters)
```

`pairwiseMedianTest` *Pairwise Mood's median tests*

Description

Conducts pairwise Mood's median tests across groups.

Usage

```
pairwiseMedianTest(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  digits = 4,
  method = "fdr",
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group.
`data`	The data frame to use.

`x`	The response variable as a vector.
`g`	The grouping variable as a vector.
`digits`	The number of significant digits to round output.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.
`...`	Additional arguments passed to `coin::median_test`.

Details

The input should include either `formula` and `data`; or `x`, and `g`.

Mood's median test compares medians among two or more groups. See `https://rcompanion.org/handbook/F_09.html` for further discussion of this test.

The `pairwiseMedianTest` function can be used as a post-hoc method following an omnibus Mood's median test. It passes the data for pairwise groups to `coin::median_test`.

The output can be converted to a compact letter display, as in the example.

Value

A dataframe of the groups being compared, the p-values, and the adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_09.html`

See Also

`pairwiseMedianMatrix`

Examples

```
data(PoohPiglet)
PoohPiglet$Speaker = factor(PoohPiglet$Speaker,
                     levels = c("Pooh", "Tigger", "Piglet"))
PT = pairwiseMedianTest(Likert ~ Speaker,
                        data   = PoohPiglet,
                        exact  = NULL,
                        method = "fdr")
PT
cldList(comparison = PT$Comparison,
        p.value    = PT$p.adjust,
        threshold  = 0.05)
```

`pairwiseModelAnova` *Compare model objects with F test and likelihood ratio test*

Description

Compares a series of models with pairwise F tests and likelihood ratio tests.

Usage

```
pairwiseModelAnova(fits, ...)
```

Arguments

`fits`	A series of model object names, separated by commas.
`...`	Other arguments passed to `list`.

Details

For comparisons to be valid, both models must have the same data, without transformations, use the same dependent variable, and be fit with the same method.

To be valid, models need to be nested.

Value

A list of: The calls of the models compared; a data frame of comparisons and F tests; and a data frame of comparisons and likelihood ratio tests.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

See Also

`compareGLM`, `compareLM`

Examples

```
### Compare among polynomial models
data(BrendonSmall)
BrendonSmall$Calories = as.numeric(BrendonSmall$Calories)

BrendonSmall$Calories2 = BrendonSmall$Calories * BrendonSmall$Calories
BrendonSmall$Calories3 = BrendonSmall$Calories * BrendonSmall$Calories *
                         BrendonSmall$Calories
BrendonSmall$Calories4 = BrendonSmall$Calories * BrendonSmall$Calories *
                         BrendonSmall$Calories * BrendonSmall$Calories
model.1 = lm(Sodium ~ Calories, data = BrendonSmall)
model.2 = lm(Sodium ~ Calories + Calories2, data = BrendonSmall)
```

```
model.3 = lm(Sodium ~ Calories + Calories2 + Calories3, data = BrendonSmall)
model.4 = lm(Sodium ~ Calories + Calories2 + Calories3 + Calories4,
             data = BrendonSmall)
pairwiseModelAnova(model.1, model.2, model.3, model.4)
```

`pairwiseNominalIndependence` *Pairwise tests of independence for nominal data*

Description

Conducts pairwise tests for a 2-dimensional matrix, in which at at least one dimension has more than two levels, as a post-hoc test. Conducts Fisher exact, Chi-square, or G-test.

Usage

```
pairwiseNominalIndependence(
  x,
  compare = "row",
  fisher = TRUE,
  gtest = TRUE,
  chisq = TRUE,
  method = "fdr",
  correct = "none",
  yates = FALSE,
  stats = FALSE,
  cramer = FALSE,
  digits = 3,
  ...
)
```

Arguments

`x`	A two-way contingency table. At least one dimension should have more than two levels.
`compare`	If `"row"`, treats the rows as the grouping variable. If `"column"`, treats the columns as the grouping variable.
`fisher`	If `"TRUE"`, conducts fisher exact test.
`gtest`	If `"TRUE"`, conducts G-test.
`chisq`	If `"TRUE"`, conducts Chi-square test of association.
`method`	The method to adjust multiple p-values. See `stats::p.adjust`.
`correct`	The correction method to pass to `DescTools::GTest`.
`yates`	Passed to `correct` in `stats::chisq.test`.

`stats`	If `"TRUE"`, includes the Chi-square value and degrees of freedom for Chi-square tests, and the G value.
`cramer`	If `"TRUE"`, includes an effect size, Cramer's V in the output.
`digits`	The number of significant digits in the output.
`...`	Additional arguments, passed to `stats::fisher.test`, `DescTools::GTest`, or `stats::chisq.test`.

Value

A data frame of comparisons, p-values, and adjusted p-values.

Acknowledgments

My thanks to Carole Elliott of Kings Park & Botanic Gardens for suggesting the inclusion on the chi-square statistic and degrees of freedom in the output.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/H_04.html`

See Also

`pairwiseMcnemar`, `groupwiseCMH`, `nominalSymmetryTest`, `pairwiseNominalMatrix`

Examples

```
### Independence test for a 4 x 2 matrix
data(Anderson)
fisher.test(Anderson)
Anderson = Anderson[(c("Heimlich", "Bloom", "Dougal", "Cobblestone")),]
PT = pairwiseNominalIndependence(Anderson,
                                 fisher = TRUE,
                                 gtest  = FALSE,
                                 chisq  = FALSE,
                                 cramer = TRUE)
PT
cldList(comparison = PT$Comparison,
        p.value    = PT$p.adj.Fisher,
        threshold  = 0.05)
```

`pairwiseNominalMatrix` *Pairwise tests of independence for nominal data with matrix output*

Description

Conducts pairwise tests for a 2-dimensional matrix, in which at at least one dimension has more than two levels, as a post-hoc test. Conducts Fisher exact, Chi-square, or G-test.

Usage

```
pairwiseNominalMatrix(
  x,
  compare = "row",
  fisher = TRUE,
  gtest = FALSE,
  chisq = FALSE,
  method = "fdr",
  correct = "none",
  digits = 3,
  ...
)
```

Arguments

`x`	A two-way contingency table. At least one dimension should have more than two levels.
`compare`	If `"row"`, treats the rows as the grouping variable. If `"column"`, treats the columns as the grouping variable.
`fisher`	If `"TRUE"`, conducts fisher exact test.
`gtest`	If `"TRUE"`, conducts G-test.
`chisq`	If `"TRUE"`, conducts Chi-square test of association.
`method`	The method to adjust multiple p-values. See `p.adjust`.
`correct`	The correction method to pass to `DescTools::GTest`.
`digits`	The number of significant digits in the output.
`...`	Additional arguments, passed to `stats::fisher.test`, `DescTools::GTest`, or `stats::chisq.test`.

Value

A list consisting of: the test used, a matrix of unadjusted p-values, the p-value adjustment method used, and a matrix of adjusted p-values.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

https://rcompanion.org/handbook/H_04.html

See Also

`pairwiseMcnemar`, `groupwiseCMH`, `nominalSymmetryTest`, `pairwiseNominalIndependence`

Examples

```
### Independence test for a 4 x 2 matrix
data(Anderson)
fisher.test(Anderson)
Anderson = Anderson[(c("Heimlich", "Bloom", "Dougal", "Cobblestone")),]
PT = pairwiseNominalMatrix(Anderson,
                           fisher = TRUE,
                           gtest  = FALSE,
                           chisq  = FALSE)$Adjusted
PT
library(multcompView)
multcompLetters(PT)
```

`pairwiseOrdinalIndependence` *Pairwise tests of independence for tables with one ordered nominal variable*

Description

Conducts pairwise tests for a 2-dimensional table, in which one variable is ordered nominal and one variable is non-ordered nominal. The function relies on the `coin` package.

Usage

```
pairwiseOrdinalIndependence(
  x,
  compare = "row",
  scores = NULL,
  method = "fdr",
  digits = 3,
  ...
)
```

Arguments

`x`	A two-way contingency table. One dimension is ordered and one is non-ordered nominal.

`compare`	If `"row"`, treats the rows as the grouping variable. If `"column"`, treats the columns as the grouping variable.
`scores`	Optional vector to specify the spacing of the ordered variable.
`method`	The method to adjust multiple p-values. See `stats::p.adjust`.
`digits`	The number of significant digits in the output.
`...`	Additional arguments, passed to `stats::chisq_test`.

Value

A data frame of comparisons, p-values, and adjusted p-values.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/H_09.html`

See Also

`pairwiseNominalIndependence`

Examples

```
### Independence test for table with one ordered variable
data(Breakfast)
require(coin)
chisq_test(Breakfast,
           scores = list("Breakfast" = c(-2, -1, 0, 1, 2)))
PT = pairwiseOrdinalIndependence(Breakfast, compare = "row")
PT
cldList(comparison = PT$Comparison,
        p.value    = PT$p.value,
        threshold  = 0.05)

### Similar to Kruskal-Wallis test for Likert data
data(PoohPiglet)
XT = xtabs(~ Speaker + Likert, data = PoohPiglet)
XT
require(coin)
chisq_test(XT,
           scores = list("Likert" = c(1, 2, 3, 4, 5)))
PT=pairwiseOrdinalIndependence(XT, compare = "row")
PT
cldList(comparison = PT$Comparison,
        p.value    = PT$p.value,
        threshold  = 0.05)
```

`pairwiseOrdinalMatrix` *[Defunct!] Pairwise two-sample ordinal regression with matrix output*

Description

Defunct. Performs pairwise two-sample ordinal regression across groups.

Usage

```
pairwiseOrdinalMatrix(...)
```

Arguments

`...`	Anything.

`pairwiseOrdinalPairedMatrix` *[Defunct!] Pairwise two-sample ordinal regression for paired data with matrix output*

Description

Defunct. Performs pairwise two-sample ordinal regression across groups for paired data with matrix output.

Usage

```
pairwiseOrdinalPairedMatrix(...)
```

Arguments

`...`	Anything.

`pairwiseOrdinalPairedTest` *[Defunct!] Pairwise two-sample ordinal regression for paired data*

Description

Defunct. Performs pairwise two-sample ordinal regression across groups for paired data.

Usage

```
pairwiseOrdinalPairedTest(...)
```

Arguments

`...`	Anything.

`pairwiseOrdinalTest` *[Defunct!] Pairwise two-sample ordinal regression*

Description

Defunct. Performs pairwise two-sample ordinal regression across groups.

Usage

```
pairwiseOrdinalTest(...)
```

Arguments

`...`	Anything.

`pairwisePercentileTest` *Pairwise permutation tests for percentiles*

Description

Conducts pairwise permutation tests across groups for percentiles, medians, and proportion below a threshold value.

Usage

```
pairwisePercentileTest(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  test = "median",
  tau = 0.5,
  type = 7,
  threshold = NA,
  comparison = "<",
  r = 1000,
  digits = 4,
  progress = "TRUE",
  method = "fdr"
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`test`	The statistic to compare between groups. Can be `"median"`, `"percentile"`, `"iqr"`, `"proportion"`, `"mean"`, or `"variance"`.
`tau`	If `"percentile"` is chosen as the `test`, `tau` indicates the percentile to test. Expressed as a quantile. That is, 0.5 indicates a test for medians. 0.75 indicates a test for 75th percentiles.
`type`	The `type` value passed to the `quantile` function.
`threshold`	If `"proportion"` is chosen as the `test`, `threshold` indicates the value of the dependent variable to use as the threshold. For example, to test if there is a different in the proportion of observations below $10,000, `threshold = 10000` would be used.
`comparison`	If `"proportion"` is chosen as the `test`, `comparison` indicates the inequality to use. Options are `"<"`, `"<="`, `">"`, `">="`, or , `"=="`
`r`	The number of replicates in the permutation test.
`digits`	The number of significant digits in the output.
`progress`	If `TRUE`, prints a dot for every 1 percent of the progress while conducting the test.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.

Details

The function conducts pairwise tests using the `percentileTest` function. The user can consult the documentation for that function for additional details.

The input should include either `formula` and `data`; or `x`, and `y`.

Value

A dataframe of the groups being compared, the p-values, and the adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_15.html`

See Also

`percentileTest`, `groupwisePercentile`

Examples

```
## Not run:
data(BrendonSmall)
PT = pairwisePercentileTest(Sodium ~ Instructor,
                            data = BrendonSmall,
                            test = "percentile",
                            tau  = 0.75)
PT
cldList(p.adjust ~ Comparison,
        data       = PT,
        threshold  = 0.05)

data(BrendonSmall)
PT = pairwisePercentileTest(Sodium ~ Instructor,
                            data       = BrendonSmall,
                            test       = "proportion",
                            threshold  = 1300)
PT
cldList(p.adjust ~ Comparison,
        data       = PT,
        threshold  = 0.05)

## End(Not run)
```

`pairwisePermutationMatrix` *Pairwise two-sample independence tests with matrix output*

Description

Conducts pairwise two-sample independence tests across groups.

Usage

```
pairwisePermutationMatrix(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  method = "fdr",
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	The response variable as a vector.
`g`	The grouping variable as a vector.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.
`...`	Additional arguments passed to `coin::independence_test`.

Details

The input should include either `formula` and `data`; or `x`, and `g`.

This function is a wrapper for `coin::independence_test`, passing pairwise groups to the function. It's critical to read and understand the documentation for this function to understand its use and options.

For some options for common tests, see Horthorn et al., 2008.

Value

A list consisting of: A matrix of p-values; the p-value adjustment method; a matrix of adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/K_02.html`

Hothorn, T., K. Hornik, M.A. van de Wiel, and A. Zeileis. 2008. Implementing a Class of Permutation Tests: The coin Package. Journal of Statistical Software, 28(8), 1–23.

See Also

`pairwisePermutationTest`

Examples

```
### Fisher-Pitman test

data(BrendonSmall)

library(coin)

independence_test(Sodium ~ Instructor, data = BrendonSmall,
                  teststat = "quadratic")

PT = pairwisePermutationMatrix(Sodium ~ Instructor,
                               data     = BrendonSmall,
                               teststat = "quadratic",
                               method   = "fdr")
PT

PA = PT$Adjusted
library(multcompView)
multcompLetters(PA,
                compare="<",
                threshold=0.05,
                Letters=letters)
```

`pairwisePermutationSymmetry` *Pairwise two-sample symmetry tests*

Description

Conducts pairwise two-sample symmetry tests across groups.

Usage

```
pairwisePermutationSymmetry(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  b = NULL,
  method = "fdr",
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group | block.
`data`	The data frame to use.
`x`	The response variable as a vector.
`g`	The grouping variable as a vector.
`b`	The blocking variable as a vector.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.
`...`	Additional arguments passed to `coin::symmetry_test`.

Details

The input should include either `formula` and `data`; or `x`, `g`, and `b`.

This function is a wrapper for `coin::symmetry_test`, passing pairwise groups to the function. It's critical to read and understand the documentation for this function to understand its use and options.

Value

A dataframe of the groups being compared, the p-values, and the adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable. The second variable on the right side is used for the blocking variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/K_03.html`

See Also

pairwisePermutationSymmetryMatrix

Examples

```
data(BobBelcher)

BobBelcher$Instructor = factor( BobBelcher$Instructor,
                                levels = c("Linda Belcher", "Louise Belcher",
                                           "Tina Belcher", "Bob Belcher",
                                           "Gene Belcher"))

library(coin)

symmetry_test(Likert ~ Instructor | Rater, data= BobBelcher,
              ytrafo   = rank_trafo,
              teststat = "quadratic")

PT = pairwisePermutationSymmetry(Likert ~ Instructor | Rater,
                                 data     = BobBelcher,
                                 ytrafo   = rank_trafo,
                                 teststat = "quadratic",
                                 method   = "fdr")
PT

cldList(comparison = PT$Comparison,
        p.value    = PT$p.adjust,
       threshold  = 0.05)
```

---

pairwisePermutationSymmetryMatrix

Pairwise two-sample symmetry tests with matrix output

Description

Conducts pairwise two-sample symmetry tests across groups.

Usage

```
pairwisePermutationSymmetryMatrix(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  b = NULL,
  method = "fdr",
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	The response variable as a vector.
`g`	The grouping variable as a vector.
`b`	The blocking variable as a vector.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.
`...`	Additional arguments passed to `coin::symmetry_test`.

Details

The input should include either `formula` and `data`; or `x`, `g`, and `b`.

This function is a wrapper for `coin::symmetry_test`, passing pairwise groups to the function. It's critical to read and understand the documentation for this function to understand its use and options.

Value

A list consisting of: A matrix of p-values; the p-value adjustment method; a matrix of adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable. The second variable on the right side is used for the blocking variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/K_03.html`

See Also

`pairwisePermutationSymmetry`

Examples

```
data(BobBelcher)

BobBelcher$Instructor = factor( BobBelcher$Instructor,
                                levels = c("Linda Belcher", "Louise Belcher",
                                           "Tina Belcher", "Bob Belcher",
                                           "Gene Belcher"))
```

```
library(coin)

symmetry_test(Likert ~ Instructor | Rater, data= BobBelcher,
              ytrafo   = rank_trafo,
              teststat = "quadratic")

PT = pairwisePermutationSymmetryMatrix(Likert ~ Instructor | Rater,
                                 data     = BobBelcher,
                                 ytrafo   = rank_trafo,
                                 teststat = "quadratic",
                                 method   = "fdr")
PT

PA = PT$Adjusted
library(multcompView)
multcompLetters(PA,
                compare="<",
                threshold=0.05,
                Letters=letters)
```

`pairwisePermutationTest`

Pairwise two-sample independence tests

Description

Conducts pairwise two-sample independence tests across groups.

Usage

```
pairwisePermutationTest(
  formula = NULL,
  data = NULL,
  x = NULL,
  g = NULL,
  method = "fdr",
  ...
)
```

Arguments

`formula`	A formula indicating the measurement variable and the grouping variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	The response variable as a vector.
`g`	The grouping variable as a vector.
`method`	The p-value adjustment method to use for multiple tests. See `stats::p.adjust`.
`...`	Additional arguments passed to `coin::independence_test`.

Details

The input should include either `formula` and `data`; or `x`, and `g`.

This function is a wrapper for `coin::independence_test`, passing pairwise groups to the function. It's critical to read and understand the documentation for this function to understand its use and options.

For some options for common tests, see Horthorn et al., 2008.

Value

A dataframe of the groups being compared, the p-values, and the adjusted p-values.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/K_02.html`

Hothorn, T., K. Hornik, M.A. van de Wiel, and A. Zeileis. 2008. Implementing a Class of Permutation Tests: The coin Package. Journal of Statistical Software, 28(8), 1–23.

See Also

`pairwisePermutationMatrix`

Examples

```
### Fisher-Pitman test

data(BrendonSmall)

library(coin)

independence_test(Sodium ~ Instructor, data = BrendonSmall,
                  teststat="quadratic")

PT = pairwisePermutationTest(Sodium ~ Instructor,
                             data   = BrendonSmall,
                             teststat="quadratic",
                             method = "fdr")
PT

cldList(comparison = PT$Comparison,
        p.value    = PT$p.adjust,
        threshold  = 0.05)
```

`pairwiseRobustMatrix` *[Defunct!] Pairwise two-sample robust tests with matrix output*

Description

Defunct. Performs pairwise two-sample robust tests across groups with matrix output.

Usage

```
pairwiseRobustMatrix(...)
```

Arguments

`...`	Anything.

`pairwiseRobustTest` *[Defunct!] Pairwise two-sample robust tests*

Description

Defunct. Performs pairwise two-sample robust tests across groups.

Usage

```
pairwiseRobustTest(...)
```

Arguments

`...`	Anything.

`pairwiseSignMatrix` *[Defunct!] Pairwise sign tests with matrix output*

Description

Defunct. Performs pairwise sign tests.

Usage

```
pairwiseSignMatrix(...)
```

Arguments

`...`	Anything.

`pairwiseSignTest` *[Defunct!] Pairwise sign tests*

Description

Defunct. Performs pairwise sign tests.

Usage

```
pairwiseSignTest(...)
```

Arguments

`...`	Anything.

`Pennsylvania18` *Votes for the Democratic candidate in Pennsylvania 18 in 2016 and 2018*

Description

A two-by-two matrix with the proportion of votes for the Democratic candidate in two races, in 2016 and 2018. 2016 is the Presidential election with Hilary Clinton as the Democratic candidate. 2018 is a House of Representatives election with Conor Lamb. These data are for Pennsylvania's 18th Congressional District.

Usage

```
Pennsylvania18
```

Format

An object of class `matrix` (inherits from `array`) with 2 rows and 2 columns.

Source

`https://rcompanion.org/handbook/H_10.html`

`percentileTest` *Test of percentiles by permutation test*

Description

Conducts a permutation test to compare two groups for medians, percentiles, or proportion below a threshold value.

Usage

```
percentileTest(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  test = "median",
  tau = 0.5,
  type = 7,
  threshold = NA,
  comparison = "<",
  r = 1000,
  digits = 4,
  progress = "TRUE"
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`test`	The statistic to compare between groups. Can be `"median"`, `"percentile"`, `"iqr"`, `"proportion"`, `"mean"`, or `"variance"`.
`tau`	If `"percentile"` is chosen as the `test`, `tau` indicates the percentile to test. Expressed as a quantile. That is, 0.5 indicates a test for medians. 0.75 indicates a test for 75th percentiles.
`type`	The `type` value passed to the `quantile` function.
`threshold`	If `"proportion"` is chosen as the `test`, `threshold` indicates the value of the dependent variable to use as the threshold. For example, to test if there is a different in the proportion of observations below $10,000, `threshold = 10000` would be used.
`comparison`	If `"proportion"` is chosen as the `test`, `comparison` indicates the inequality to use. Options are `"<"`, `"<="`, `">"`, `">="`, or , `"=="`

`r`	The number of replicates in the permutation test.
`digits`	The number of significant digits in the output.
`progress`	If `TRUE`, prints a dot for every 1 percent of progress while conducting the test.

Details

The function will test for a difference in medians, percentiles, interquartile ranges, proportion of observations above or below some threshold value, means, or variances between two groups by permutation test.

The permutation test simply permutes the observed values over the two groups and counts how often the calculated statistic is at least as extreme as the original observed statistic.

The input should include either `formula` and `data`; or `x` and `y`.

The function removes cases with NA in any of the variables.

If the independent variable has more than two groups, only the first two levels of the factor variable will be used.

The p-value returned is a two-sided test.

Value

A list of three data frames with the data used, a summary for each group, and the p-value from the test.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the independent variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

https://rcompanion.org/handbook/F_15.html

Examples

```
data(BrendonSmall)
percentileTest(Sodium ~ Instructor,
               data=BrendonSmall,
               test="median")

percentileTest(Sodium ~ Instructor,
               data=BrendonSmall,
               test="percentile",
               tau = 0.75)

percentileTest(Sodium ~ Instructor,
               data=BrendonSmall,
```

```
                test="proportion",
                threshold = 1300)
```

---

phi *phi*

Description

Calculates phi for a 2 x 2 table of nominal variables; confidence intervals by bootstrap.

Usage

```
phi(
  x,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  verbose = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  ...
)
```

Arguments

`x`	Either a 2 x 2 table or a 2 x 2 matrix. Can also be a vector of observations for one dimension of a 2 x 2 table.
`y`	If `x` is a vector, `y` is the vector of observations for the second dimension of a 2 x2 table.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`verbose`	If `TRUE`, prints the table of counts.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`...`	Additional arguments. (Ignored.)

Details

phi is used as a measure of association between two binomial variables, or as an effect size for a chi-square test of association for a 2 x 2 table. The absolute value of the phi statistic is the same as Cramer's V for a 2 x 2 table.

Unlike Cramer's V, phi can be positive or negative (or zero), and ranges from -1 to 1.

When phi is close to its extremes, or with small counts, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, phi. Or a small data frame consisting of phi, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/H_10.html`

See Also

`cramerV`

Examples

```
### Example with table
Matrix = matrix(c(13, 26, 26, 13), ncol=2)
phi(Matrix)

### Example with two vectors
Species = c(rep("Species1", 16), rep("Species2", 16))
Color   = c(rep(c("blue", "blue", "blue", "green"),4),
            rep(c("green", "green", "green", "blue"),4))
phi(Species, Color)
```

`plotDensityHistogram` *Histogram with a density curve*

Description

Produces a histogram for a vector of values and adds a density curve of the distribution.

Usage

```
plotDensityHistogram(
  x,
  prob = FALSE,
  col = "gray",
  main = "",
  linecol = "black",
  lwd = 2,
  adjust = 1,
  bw = "nrd0",
  kernel = "gaussian",
  ...
)
```

Arguments

`x`	A vector of values.
`prob`	If `FALSE`, then counts are displayed in the histogram. If `TRUE`, then the density is shown.
`col`	The color of the histogram bars.
`main`	The title displayed for the plot.
`linecol`	The color of the line in the plot.
`lwd`	The width of the line in the plot.
`adjust`	Passed to `density`. A lower value makes the density plot smoother.
`bw`	Passed to `density`.
`kernel`	Passed to `density`.
`...`	Other arguments passed to `hist`.

Details

The function relies on the `hist` function. The density curve relies on the `density` function.

Value

Produces a plot. Returns nothing.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/C_04.html`

See Also

`plotNormalHistogram`, `plotNormalDensity`

Examples

```
### Plot of residuals from a model fit with lm
data(Catbus)
model = lm(Steps ~ Gender + Teacher,
           data = Catbus)
plotDensityHistogram(residuals(model))
```

`plotNormalDensity` *Density plot with a normal curve*

Description

Produces a density plot for a vector of values and adds a normal curve with the same mean and standard deviation. The plot can be used to quickly compare the distribution of data to a normal distribution.

Usage

```
plotNormalDensity(
  x,
  col1 = "white",
  col2 = "gray",
  col3 = "blue",
  border = NA,
  main = "",
  lwd = 2,
  length = 1000,
  adjust = 1,
  bw = "nrd0",
  kernel = "gaussian",
  ...
)
```

Arguments

`x`	A vector of values.
`col1`	The color of the density plot. Usually not visible.
`col2`	The color of the density polygon.
`col3`	The color of the normal line.
`border`	The color of the border around the density polygon.
`main`	The title displayed for the plot.
`lwd`	The width of the line in the plot.
`length`	The number of points in the line in the plot.
`adjust`	Passed to `density`. A lower value makes the density plot smoother.

`bw`	Passed to `density`.
`kernel`	Passed to `density`.
`...`	Other arguments passed to `plot`.

Details

The function plots a polygon based on the `density` function. The normal curve has the same mean and standard deviation as the values in the vector.

Value

Produces a plot. Returns nothing.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/I_01.html`

See Also

`plotNormalHistogram`, `plotDensityHistogram`

Examples

```
### Plot of residuals from a model fit with lm
data(Catbus)
model = lm(Steps ~ Gender + Teacher,
           data = Catbus)
 plotNormalDensity(residuals(model))
```

`plotNormalHistogram` *Histogram with a normal curve*

Description

Produces a histogram for a vector of values and adds a normal curve with the same mean and standard deviation. The plot can be used to quickly compare the distribution of data to a normal distribution.

Usage

```
plotNormalHistogram(
  x,
  prob = FALSE,
  col = "gray",
  main = "",
  linecol = "blue",
  lwd = 2,
  length = 1000,
  ...
)
```

Arguments

`x`	A vector of values.
`prob`	If `FALSE`, then counts are displayed in the histogram. If `TRUE`, then the density is shown.
`col`	The color of the histogram bars.
`main`	The title displayed for the plot.
`linecol`	The color of the line in the plot.
`lwd`	The width of the line in the plot.
`length`	The number of points in the line in the plot.
`...`	Other arguments passed to `hist`.

Details

The function relies on the `hist` function. The normal curve has the same mean and standard deviation as the values in the vector.

Value

Produces a plot. Returns nothing.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/I_01.html`

See Also

`plotNormalDensity`, `plotDensityHistogram`

Examples

```
### Plot of residuals from a model fit with lm
data(Catbus)
model = lm(Steps ~ Gender + Teacher,
           data = Catbus)
 plotNormalHistogram(residuals(model))
```

`plotPredy` *Plot a predicted line from a bivariate model*

Description

Plots the best fit line for a model with one y variable and one x variable, or with one y variable and polynomial x variables.

Usage

```
plotPredy(
  data,
  x,
  y,
  model,
  order = 1,
  x2 = NULL,
  x3 = NULL,
  x4 = NULL,
  x5 = NULL,
  pch = 16,
  xlab = "X",
  ylab = "Y",
  length = 1000,
  lty = 1,
  lwd = 2,
  col = "blue",
  type = NULL,
  ...
)
```

Arguments

`data`	The name of the data frame.
`x`	The name of the x variable.
`y`	The name of the y variable.
`model`	The name of the model object.

`order`	If plotting a polynomial function, the order of the polynomial. Otherwise can be left as `1`.
`x2`	If applicable, the name of the second order polynomial x variable.
`x3`	If applicable, the name of the third order polynomial x variable.
`x4`	If applicable, the name of the fourth order polynomial x variable.
`x5`	If applicable, the name of the fifth order polynomial x variable.
`pch`	The shape of the plotted data points.
`xlab`	The label for the x-axis.
`ylab`	The label for the y-axis.
`length`	The number of points used to draw the line.
`lty`	The style of the plotted line.
`lwd`	The width of the plotted line.
`col`	The col of the plotted line.
`type`	Passed to `predict`. Required for certain models.
`...`	Other arguments passed to `plot`.

Details

Any model for which `predict()` is defined can be used.

Value

Produces a plot. Returns nothing.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/I_10.html`

Examples

```
### Plot of linear model fit with lm
data(BrendonSmall)
model = lm(Weight ~ Calories, data = BrendonSmall)
plotPredy(data  = BrendonSmall,
          y     = Weight,
          x     = Calories,
          model = model,
          xlab  = "Calories per day",
          ylab  = "Weight in kilograms")

### Plot of polynomial model fit with lm
data(BrendonSmall)
BrendonSmall$Calories2 = BrendonSmall$Calories * BrendonSmall$Calories
```

```
model = lm(Sodium ~ Calories + Calories2, data = BrendonSmall)
plotPredy(data  = BrendonSmall,
          y     = Sodium,
          x     = Calories,
          x2    = Calories2,
          model = model,
          order = 2,
          xlab  = "Calories per day",
          ylab  = "Sodium intake per day")

### Plot of quadratic plateau model fit with nls
data(BrendonSmall)
quadplat = function(x, a, b, clx) {
          ifelse(x  < clx, a + b * x   + (-0.5*b/clx) * x   * x,
                           a + b * clx + (-0.5*b/clx) * clx * clx)}
model = nls(Sodium ~ quadplat(Calories, a, b, clx),
            data = BrendonSmall,
            start = list(a   = 519,
                         b   = 0.359,
                         clx = 2304))
plotPredy(data  = BrendonSmall,
          y     = Sodium,
          x     = Calories,
          model = model,
          xlab  = "Calories per day",
          ylab  = "Sodium intake per day")

### Logistic regression example requires type option
data(BullyHill)
Trials = cbind(BullyHill$Pass, BullyHill$Fail)
model.log = glm(Trials ~ Grade, data = BullyHill,
                family = binomial(link="logit"))
plotPredy(data  = BullyHill,
          y     = Percent,
          x     = Grade,
          model = model.log,
          type  = "response",
          xlab  = "Grade",
          ylab  = "Proportion passing")
```

`PMCMRTable` *Convert PMCMR Objects to a Data Frame*

Description

Extracts a data frame of comparisons and p-values from an PMCMR object from the PMCMRplus package

Usage

```
PMCMRTable(PMCMR, reverse = TRUE, digits = 3)
```

Arguments

`PMCMR`	A PMCMR object
`reverse`	If `TRUE`, reports the comparison as e.g. (B - A = 0). This will more closely match the output of `PMCMRplus::summary.PMCMR` for all-pairs comparisons. If `FALSE`, reports the comparison as e.g. (A - B = 0). This will result in the output from `rcompanion::cldList` matching the output of `PMCMRplus::summaryGroup`
`digits`	The significant digits in the output

Details

Should produce meaningful output for all-pairs and many-to-one comparisons.

Value

A data frame of comparisons and p-values

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_08.html`

`Pooh` *Hypothetical data for paired ratings of Pooh Bear*

Description

A data frame of Likert responses for instructor Pooh Bear for each of 10 respondents, paired before and after. Hypothetical data.

Usage

```
Pooh
```

Format

An object of class `data.frame` with 20 rows and 4 columns.

Source

`https://rcompanion.org/handbook/F_06.html`

`PoohPiglet` *Hypothetical data for ratings of Pooh, Piglet, and Tigger*

Description

A data frame of Likert responses for instructors Pooh Bear, Piglet, and Tigger. Hypothetical data.

Usage

```
PoohPiglet
```

Format

An object of class `data.frame` with 30 rows and 2 columns.

Source

https://rcompanion.org/handbook/F_08.html

`quantileCI` *Quantiles and confidence intervals*

Description

Calculates an estimate for a quantile and confidence intervals for a vector of discrete or continuous values

Usage

```
quantileCI(
  x,
  tau = 0.5,
  level = 0.95,
  method = "binomial",
  type = 3,
  digits = 3,
  ...
)
```

Arguments

`x`	The vector of observations. Can be an ordered factor as long as `type` is 1 or 3.
`tau`	The quantile to use, e.g. 0.5 for median, 0.25 for 25th percentile.
`level`	The confidence interval to use, e.g. 0.95 for 95 percent confidence interval.
`method`	If `"binomial"`, uses the binomial distribution the confidence limits. If `"normal"`, uses the normal approximation to the binomial distribution.
`type`	The `type` value passed to the `quantile` function.
`digits`	The number of significant figures to use in output.
`...`	Other arguments, ignored.

Details

Conover recommends the `"binomial"` method for sample sizes less than or equal to 20. With the current implementation, this method can be used also for larger sample sizes.

Value

A data frame of summary statistics, quantile estimate, and confidence limits.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/E_04.html`

Conover, W.J., Practical Nonparametric Statistics, 3rd.

See Also

`groupwisePercentile`, `groupwiseMedian`

Examples

```
### From Conover, Practical Nonparametric Statistics, 3rd
Hours = c(46.9, 47.2, 49.1, 56.5, 56.8, 59.2, 59.9, 63.2,
          63.3, 63.4, 63.7, 64.1, 67.1, 67.7, 73.3, 78.5)
quantileCI(Hours)

### Example with ordered factor
set.seed(12345)
Pool = factor(c("smallest", "small", "medium", "large", "largest"),
             ordered=TRUE,
             levels=c("smallest", "small", "medium", "large", "largest"))
Sample = sample(Pool, 24, replace=TRUE)
quantileCI(Sample)
```

`Religion` *Hypothetical data for change in religion after a caucusing event*

Description

A matrix of paired counts for religion of people before and after an event. Hypothetical data.

Usage

```
Religion
```

Format

An object of class `matrix` (inherits from `array`) with 4 rows and 4 columns.

Source

https://rcompanion.org/handbook/H_05.html

`scheirerRayHare` *Scheirer Ray Hare test*

Description

Conducts Scheirer Ray Hare test.

Usage

```
scheirerRayHare(
  formula = NULL,
  data = NULL,
  y = NULL,
  x1 = NULL,
  x2 = NULL,
  type = 2,
  tie.correct = TRUE,
  ss = TRUE,
  verbose = TRUE
)
```

Arguments

`formula`	A formula indicating the response variable and two independent variables. e.g. y ~ x1 + x2.
`data`	The data frame to use.
`y`	If no formula is given, the response variable.
`x1`	If no formula is given, the first independent variable.
`x2`	If no formula is given, the second independent variable.
`type`	The type of sum of squares to be used. Acceptable options are `1`, `2`, `"I"`, or `"II"`.
`tie.correct`	If `"TRUE"`, applies a correction for ties in the response variable.
`ss`	If `"TRUE"`, includes the sums of squares in the output.
`verbose`	If `"TRUE"`, outputs statistics used in the analysis by direct print.

Details

The Scheirer Ray Hare test is a nonparametric test used for a two-way factorial experiment. It is described by Sokal and Rohlf (1995).

It is sometimes recommended that the design should be balanced, and that there should be at least five observations for each cell in the interaction.

One might consider using aligned ranks transformation anova instead of the Scheirer Ray Hare test.

Note that for unbalanced designs, by default, a type-II sum-of-squares approach is used.

The input should include either `formula` and `data`; or `y`, `x1`, and `x2`.

The function removes cases with NA in any of the variables.

Value

A data frame of results similar to an anova table. Output from the `verbose` option is printed directly and not returned with the data frame.

Acknowledgments

Thanks to Guillaume Loignon for the suggestion to include type-II sum-of-squares.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the first independent variable. The second variable on the right side is used for the second independent variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

Sokal, R.R. and F.J. Rohlf. 1995. Biometry. 3rd ed. W.H. Freeman, New York.

https://rcompanion.org/handbook/F_14.html

Examples

```
### Example from Sokal and Rohlf, 1995.
Value = c(709,679,699,657,594,677,592,538,476,508,505,539)
Sex   = c(rep("Male",3), rep("Female",3), rep("Male",3), rep("Female",3))
Fat   = c(rep("Fresh", 6), rep("Rancid", 6))
Sokal = data.frame(Value, Sex, Fat)

scheirerRayHare(Value ~ Sex + Fat, data=Sokal)
```

`spearmanRho` *Spearman's rho, Kendall's tau, Pearson's r*

Description

Calculates Spearmans's rho, Kendall's tau, or Pearson's r with confidence intervals by bootstrap

Usage

```
spearmanRho(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  method = "spearman",
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  ...
)
```

Arguments

`formula`	A formula indicating the two paired variables, e.g. `~ x + y`. The variables should be vectors of the same length.
`data`	The data frame to use.
`x`	If no formula is given, the values for one variable.

`y`	The values for the other variable.
`method`	One of "spearman", "kendall", or "pearson". Passed to `cor`.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`...`	Additional arguments passed to the `cor` function.

Details

This function is a wrapper for `stats::cor` with the addition of confidence intervals.

The input should include either `formula` and `data`; or `x`, and `y`.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the returned statistic is close to -1 or close to 1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, rho, tau, or r. Or a small data frame consisting of rho, tau, or r, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/I_10.html`

Examples

```
data(Catbus)
spearmanRho( ~ Steps + Rating, data=Catbus)
```

`transformTukey` *Tukey's Ladder of Powers*

Description

Conducts Tukey's Ladder of Powers on a vector of values to produce a more-normally distributed vector of values.

Usage

```
transformTukey(
  x,
  start = -10,
  end = 10,
  int = 0.025,
  plotit = TRUE,
  verbose = FALSE,
  quiet = FALSE,
  statistic = 1,
  returnLambda = FALSE
)
```

Arguments

`x`	A vector of values.
`start`	The starting value of lambda to try.
`end`	The ending value of lambda to try.
`int`	The interval between lambda values to try.
`plotit`	If `TRUE`, produces plots of Shapiro-Wilks W or Anderson-Darling A vs. lambda, a histogram of transformed values, and a quantile-quantile plot of transformed values.
`verbose`	If `TRUE`, prints extra output for Shapiro-Wilks W or Anderson-Darling A vs. lambda.
`quiet`	If `TRUE`, doesn't print any output to the screen.
`statistic`	If `1`, uses Shapiro-Wilks test. Will report `NA` if the sample size is greater than 5000. If `2`, uses Anderson-Darling test.
`returnLambda`	If `TRUE`, returns only the lambda value, not the vector of transformed values.

Details

The function simply loops through lamdba values from `start` to `end` at an interval of `int`.

The function then chooses the lambda which maximizes the Shapiro-Wilks W statistic or minimizes the Anderson-Darling A statistic.

It may be beneficial to add a constant to the input vector so that all values are posititive. For left-skewed data, a (Constant - X) transformation may be helpful. Large values may need to be scaled.

Value

The transformed vector of values. The chosen lambda value is printed directly.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/I_12.html

Examples

```
### Log-normal distribution example
Conc = rlnorm(100)
Conc.trans = transformTukey(Conc)
```

---

vda *Vargha and Delaney's A*

Description

Calculates Vargha and Delaney's A (VDA) with confidence intervals by bootstrap

Usage

```
vda(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  reportIncomplete = FALSE,
  brute = FALSE,
  verbose = FALSE,
  digits = 3,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`brute`	If `FALSE`, the default, the statistic is based on the U statistic from the `wilcox.test` function. If `TRUE`, the function will compare values in the two samples directly.
`verbose`	If `TRUE`, reports the proportion of ties and the proportions of (Ya > Yb) and (Ya < Yb).
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `wilcox.test` function.

Details

VDA is an effect size statistic appropriate in cases where a Wilcoxon-Mann-Whitney test might be used. It ranges from 0 to 1, with 0.5 indicating stochastic equality, and 1 indicating that the first group dominates the second.

By default, the function calculates VDA from the "W" U statistic from the `wilcox.test` function. Specifically, `VDA = U/(n1*n2)`.

The input should include either `formula` and `data`; or `x`, and `y`. If there are more than two groups, only the first two groups are used.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, VDA is greater than 0.5. When the data in the second group are greater than in the first group, VDA is less than 0.5.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

When VDA is close to 0 or close to 1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, VDA. Or a small data frame consisting of VDA, and the lower and upper confidence limits.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

`https://rcompanion.org/handbook/F_04.html`

See Also

`cliffDelta`, `multiVDA`

Examples

```
data(Catbus)
vda(Steps ~ Gender, data=Catbus)
```

`wilcoxonOneSampleR` *r effect size for Wilcoxon one-sample signed-rank test*

Description

Calculates r effect size for a Wilcoxon one-sample signed-rank test; confidence intervals by bootstrap.

Usage

```
wilcoxonOneSampleR(
  x,
  mu = NULL,
  adjustn = TRUE,
  coin = FALSE,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  ...
)
```

Arguments

`x`	A vector of observations.
`mu`	The value to compare `x` to, as in `wilcox.test`
`adjustn`	If `TRUE`, reduces the sample size in the calculation of `r` by the number of observations equal to `mu`.
`coin`	If `FALSE`, the default, the Z value is extracted from a function similar to the `wilcox.test` function in the stats package. If `TRUE`, the Z value is extracted from the `wilcox_test` function in the coin package. This method may be much slower, especially if a confidence interval is produced.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `wilcoxsign_test` function.

Details

r is calculated as Z divided by square root of the number of observations.

The calculated statistic is equivalent to the statistic returned by the `wilcoxPairedR` function with one group equal to a vector of `mu`. The author knows of no reference for this technique.

This statistic typically reports a smaller effect size (in absolute value) than does the matched-pairs rank biserial correlation coefficient (`wilcoxonOneSampleRC`), and may not reach a value of -1 or 1 if there are values tied with `mu`.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data are greater than `mu`, r is positive. When the data are less than `mu`, r is negative.

When r is close to extremes, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, r. Or a small data frame consisting of r, and the lower and upper confidence limits.

Acknowledgments

My thanks to Peter Stikker for the suggestion to adjust the sample size for ties with `mu`.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/F_02.html

Examples

```
X = c(1,2,3,3,3,3,4,4,4,4,4,5,5,5,5,5)
wilcox.test(X, mu=3, exact=FALSE)
wilcoxonOneSampleR(X, mu=3)
```

`wilcoxonOneSampleRC` *Rank biserial correlation coefficient for one-sample Wilcoxon test*

Description

Calculates rank biserial correlation coefficient effect size for one-sample Wilcoxon signed-rank test; confidence intervals by bootstrap.

Usage

```
wilcoxonOneSampleRC(
  x,
  mu = NULL,
  zero.method = "Wilcoxon",
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  verbose = FALSE,
  ...
)
```

Arguments

`x`	A vector of observations.
`mu`	The value to compare `x` to, as in `wilcox.test`
`zero.method`	If `"Wilcoxon"`, differences of zero are discarded and then ranks are determined. If `"Pratt"`, ranks are determined, and then differences of zero are discarded. If `"none"`, differences of zero are not discarded.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.

`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`verbose`	If `TRUE`, prints information on sample size and ranks.
`...`	Additional arguments passed to the `wilcoxsign_test` function.

Details

It is recommended that `NA`s be removed beforehand.

When rc is close to extremes, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, rc. Or a small data frame consisting of rc, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_02.html`

See Also

`wilcoxonPairedRC`

Examples

```
### Example with one zero difference
A = c(11,12,13,14,15,16,17,18,19,20)
#' wilcoxonOneSampleRC(x = A, mu=15)
wilcoxonOneSampleRC(x = A, mu=15, verbose=TRUE, zero.method="Wilcoxon")
wilcoxonOneSampleRC(x = A, mu=15, verbose=TRUE, zero.method="Pratt")
wilcoxonOneSampleRC(x = A, mu=15, verbose=TRUE, zero.method="none")
```

`wilcoxonOR` *Agresti's Generalized Odds Ratio for Stochastic Dominance*

Description

Calculates Agresti's Generalized Odds Ratio for Stochastic Dominance (OR) with confidence intervals by bootstrap

Usage

```
wilcoxonOR(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`verbose`	If `TRUE`, reports the proportion of ties and the proportions of (Ya > Yb) and (Ya < Yb).
`...`	Additional arguments, not used.

Details

OR is an effect size statistic appropriate in cases where a Wilcoxon-Mann-Whitney test might be used.

OR is defined as P(Ya > Yb) / P(Ya < Yb).

OR can range from 0 to infinity. An OR of 1 indicates stochastic equality between the two groups. An OR greater than 1 indicates that the first group dominates the second group. An OR less than 1 indicates that the second group dominates the first.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

The input should include either `formula` and `data`; or `x`, and `y`. If there are more than two groups, only the first two groups are used.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

With a small sample size, or with an OR near its extremes, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, OR. Or a small data frame consisting of OR, and the lower and upper confidence limits.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

Grissom, R.J. and J.J. Kim. 2012. Effect Sizes for Research. 2nd ed. Routledge, New York.

https://rcompanion.org/handbook/F_04.html

See Also

`wilcoxonPS`

Examples

```
data(Catbus)
wilcoxonOR(Steps ~ Gender, data=Catbus, verbose=TRUE)
```

`wilcoxonPairedR` *r effect size for Wilcoxon two-sample paired signed-rank test*

Description

Calculates r effect size for a Wilcoxon two-sample paired signed-rank test; confidence intervals by bootstrap.

Usage

```
wilcoxonPairedR(
  x,
  g = NULL,
  adjustn = TRUE,
  coin = FALSE,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  cases = TRUE,
  digits = 3,
  ...
)
```

Arguments

`x`	A vector of observations.
`g`	The vector of observations for the grouping, nominal variable. Only the first two levels of the nominal variable are used. The data must be ordered so that the first observation of the of the first group is paired with the first observation of the second group.
`adjustn`	If `TRUE`, reduces the sample size in the calculation of `r` by the number of tied pairs.
`coin`	If `FALSE`, the default, the Z value is extracted from a function similar to the `wilcox.test` function in the stats package. If `TRUE`, the Z value is extracted from the `wilcox_test` function in the coin package. This method may be much slower, especially if a confidence interval is produced.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.

`cases`	By default the `N` used in the formula for `r` is the number of pairs. If `cases=FALSE`, the `N` used in the formula for `r` is the total number of observations, as some sources suggest.
`digits`	The number of significant digits in the output.
`...`	Additional arguments passed to the `wilcoxsign_test` function.

Details

r is calculated as Z divided by square root of the number of observations in one group. This results in a statistic that ranges from -1 to 1. This range doesn't hold if `cases=FALSE`.

This statistic typically reports a smaller effect size (in absolute value) than does the matched-pairs rank biserial correlation coefficient (`wilcoxonPairedRC`), and may not reach a value of -1 or 1 if there are ties in the paired differences.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, r is positive. When the data in the second group are greater than in the first group, r is negative. Be cautious with this interpretation, as R will alphabetize groups if g is not already a factor.

When r is close to extremes, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, r. Or a small data frame consisting of r, and the lower and upper confidence limits.

Acknowledgments

My thanks to Peter Stikker for the suggestion to adjust the sample size for ties.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

https://rcompanion.org/handbook/F_06.html

See Also

`wilcoxonPairedRC`

Examples

```
data(Pooh)
Time1 = Pooh$Likert[Pooh$Time==1]
Time2 = Pooh$Likert[Pooh$Time==2]
wilcox.test(x = Time1, y = Time2, paired=TRUE, exact=FALSE)
wilcoxonPairedR(x = Pooh$Likert, g = Pooh$Time)
```

`wilcoxonPairedRC` *Matched-pairs rank biserial correlation coefficient*

Description

Calculates matched-pairs rank biserial correlation coefficient effect size for paired Wilcoxon signed-rank test; confidence intervals by bootstrap.

Usage

```
wilcoxonPairedRC(
  x,
  g = NULL,
  zero.method = "Wilcoxon",
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  verbose = FALSE,
  ...
)
```

Arguments

`x`	A vector of observations.
`g`	The vector of observations for the grouping, nominal variable. Only the first two levels of the nominal variable are used.
`zero.method`	If `"Wilcoxon"`, differences of zero are discarded and then ranks are determined. If `"Pratt"`, ranks are determined, and then differences of zero are discarded. If `"none"`, differences of zero are not discarded.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`verbose`	If `TRUE`, prints information on sample size and ranks.
`...`	Additional arguments passed to `rank`

Details

It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, rc is positive. When the data in the second group are greater than in the first group, rc is negative.

Be cautious with this interpretation, as R will alphabetize groups if g is not already a factor.

When rc is close to extremes, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, rc. Or a small data frame consisting of rc, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

King, B.M., P.J. Rosopa, and E.W. Minium. 2011. Statistical Reasoning in the Behavioral Sciences, 6th ed.

`https://rcompanion.org/handbook/F_06.html`

See Also

`wilcoxonPairedR`

Examples

```
data(Pooh)
Time1 = Pooh$Likert[Pooh$Time==1]
Time2 = Pooh$Likert[Pooh$Time==2]
wilcox.test(x = Time1, y = Time2, paired=TRUE, exact=FALSE)
wilcoxonPairedRC(x = Pooh$Likert, g = Pooh$Time)

### Example from King, Rosopa, and Minium
Placebo = c(24,39,29,28,25,32,31,33,31,22)
Drug    = c(28,29,34,21,28,15,17,28,16,12)
Y = c(Placebo, Drug)
Group = factor(c(rep("Placebo", length(Placebo)),
                 rep("Drug", length(Drug))),
                 levels=c("Placebo", "Drug"))
wilcoxonPairedRC(x = Y, g = Group)

### Example with some zero differences
A = c(11,12,13,14,15,16,17,18,19,20)
B = c(12,14,16,18,20,22,12,10,19,20)
Y = c(A, B)
Group = factor(c(rep("A", length(A)),
```

```
                rep("B", length(B))))
wilcoxonPairedRC(x = Y, g = Group, verbose=TRUE, zero.method="Wilcoxon")
wilcoxonPairedRC(x = Y, g = Group, verbose=TRUE, zero.method="Pratt")
wilcoxonPairedRC(x = Y, g = Group, verbose=TRUE, zero.method="none")
```

`wilcoxonPS` *Grissom and Kim's Probability of Superiority (PS)*

Description

Calculates Grissom and Kim's Probability of Superiority (PS) with confidence intervals by bootstrap

Usage

```
wilcoxonPS(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`formula`	A formula indicating the response variable and the independent variable. e.g. y ~ group.
`data`	The data frame to use.
`x`	If no formula is given, the response variable for one group.
`y`	The response variable for the other group.
`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of "`norm`", "`basic`", "`perc`", or "`bca`". Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.

`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`verbose`	If `TRUE`, reports the proportion of ties and the proportions of (Ya > Yb) and (Ya < Yb).
`...`	Additional arguments, not used.

Details

PS is an effect size statistic appropriate in cases where a Wilcoxon-Mann-Whitney test might be used. It ranges from 0 to 1, with 0.5 indicating stochastic equality, and 1 indicating that the first group dominates the second.

PS is defined as P(Ya > Yb), with no provision made for tied values across groups.

If there are no tied values, PS will be equal to VDA.

The input should include either `formula` and `data`; or `x`, and `y`. If there are more than two groups, only the first two groups are used.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, PS is greater than 0.5. When the data in the second group are greater than in the first group, PS is less than 0.5.

Be cautious with this interpretation, as R will alphabetize groups in the formula interface if the grouping variable is not already a factor.

When PS is close to 0 or close to 1, or with small sample size, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, PS. Or a small data frame consisting of PS, and the lower and upper confidence limits.

Note

The parsing of the formula is simplistic. The first variable on the left side is used as the measurement variable. The first variable on the right side is used for the grouping variable.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

Grissom, R.J. and J.J. Kim. 2012. Effect Sizes for Research. 2nd ed. Routledge, New York.

`https://rcompanion.org/handbook/F_04.html`

See Also

`cliffDelta`, `vda`

Examples

```
data(Catbus)
wilcoxonPS(Steps ~ Gender, data=Catbus, verbose=TRUE)
```

`wilcoxonR` *r effect size for Wilcoxon two-sample rank-sum test*

Description

Calculates r effect size for Mann-Whitney two-sample rank-sum test, or a table with an ordinal variable and a nominal variable with two levels; confidence intervals by bootstrap.

Usage

```
wilcoxonR(
  x,
  g = NULL,
  group = "row",
  coin = FALSE,
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  ...
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations.
`g`	If `x` is a vector, `g` is the vector of observations for the grouping, nominal variable. Only the first two levels of the nominal variable are used.
`group`	If `x` is a table or matrix, `group` indicates whether the `"row"` or the `"column"` variable is the nominal, grouping variable.
`coin`	If `FALSE`, the default, the Z value is extracted from a function similar to the `wilcox.test` function in the stats package. If `TRUE`, the Z value is extracted from the `wilcox_test` function in the coin package. This method may be much slower, especially if a confidence interval is produced.

`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of `"norm"`, `"basic"`, `"perc"`, or `"bca"`. Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`...`	Additional arguments passed to the `wilcox_test` function.

Details

r is calculated as Z divided by square root of the total observations.

This statistic reports a smaller effect size than does Glass rank biserial correlation coefficient (`wilcoxonRG`), and cannot reach -1 or 1. This effect is exaserbated when sample sizes are not equal.

Currently, the function makes no provisions for `NA` values in the data. It is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, r is positive. When the data in the second group are greater than in the first group, r is negative. Be cautious with this interpretation, as R will alphabetize groups if g is not already a factor.

When r is close to extremes, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, r. Or a small data frame consisting of r, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>

References

`https://rcompanion.org/handbook/F_04.html`

See Also

`freemanTheta`, `wilcoxonRG`

Examples

```
data(Breakfast)
Table = Breakfast[1:2,]
library(coin)
chisq_test(Table, scores = list("Breakfast" = c(-2, -1, 0, 1, 2)))
wilcoxonR(Table)

data(Catbus)
wilcox.test(Steps ~ Gender, data = Catbus)
wilcoxonR(x = Catbus$Steps, g = Catbus$Gender)
```

`wilcoxonRG` *Glass rank biserial correlation coefficient*

Description

Calculates Glass rank biserial correlation coefficient effect size for Mann-Whitney two-sample rank-sum test, or a table with an ordinal variable and a nominal variable with two levels; confidence intervals by bootstrap.

Usage

```
wilcoxonRG(
  x,
  g = NULL,
  group = "row",
  ci = FALSE,
  conf = 0.95,
  type = "perc",
  R = 1000,
  histogram = FALSE,
  digits = 3,
  reportIncomplete = FALSE,
  verbose = FALSE,
  na.last = NA,
  ...
)
```

Arguments

`x`	Either a two-way table or a two-way matrix. Can also be a vector of observations.
`g`	If `x` is a vector, `g` is the vector of observations for the grouping, nominal variable. Only the first two levels of the nominal variable are used.
`group`	If `x` is a table or matrix, `group` indicates whether the `"row"` or the `"column"` variable is the nominal, grouping variable.

`ci`	If `TRUE`, returns confidence intervals by bootstrap. May be slow.
`conf`	The level for the confidence interval.
`type`	The type of confidence interval to use. Can be any of `"norm"`, `"basic"`, `"perc"`, or `"bca"`. Passed to `boot.ci`.
`R`	The number of replications to use for bootstrap.
`histogram`	If `TRUE`, produces a histogram of bootstrapped values.
`digits`	The number of significant digits in the output.
`reportIncomplete`	If `FALSE` (the default), `NA` will be reported in cases where there are instances of the calculation of the statistic failing during the bootstrap procedure.
`verbose`	If `TRUE`, prints information on factor levels and ranks.
`na.last`	Passed to `rank`. For example, can be set to `TRUE` to assign `NA` values a minimum rank.
`...`	Additional arguments passed to `rank`

Details

rg is calculated as 2 times the difference of mean of ranks for each group divided by the total sample size. It appears that rg is equivalent to Cliff's delta.

`NA` values can be handled by the `rank` function. In this case, using `verbose=TRUE` is helpful to understand how the `rg` statistic is calculated. Otherwise, it is recommended that `NA`s be removed beforehand.

When the data in the first group are greater than in the second group, rg is positive. When the data in the second group are greater than in the first group, rg is negative.

Be cautious with this interpretation, as R will alphabetize groups if `g` is not already a factor.

When rg is close to extremes, or with small counts in some cells, the confidence intervals determined by this method may not be reliable, or the procedure may fail.

Value

A single statistic, rg. Or a small data frame consisting of rg, and the lower and upper confidence limits.

Author(s)

Salvatore Mangiafico, `<mangiafico@njaes.rutgers.edu>`

References

King, B.M., P.J. Rosopa, and E.W. Minium. 2011. Statistical Reasoning in the Behavioral Sciences, 6th ed.

`https://rcompanion.org/handbook/F_04.html`

See Also

`wilcoxonR`

Examples

```
data(Breakfast)
Table = Breakfast[1:2,]
library(coin)
chisq_test(Table, scores = list("Breakfast" = c(-2, -1, 0, 1, 2)))
wilcoxonRG(Table)

data(Catbus)
wilcox.test(Steps ~ Gender, data = Catbus)
wilcoxonRG(x = Catbus$Steps, g = Catbus$Gender)

### Example from King, Rosopa, and Minium
Criticism = c(-3, -2, 0, 0, 2, 5, 7, 9)
Praise = c(0, 2, 3, 4, 10, 12, 14, 19, 21)
Y = c(Criticism, Praise)
Group = factor(c(rep("Criticism", length(Criticism)),
              rep("Praise", length(Praise))))
wilcoxonRG(x = Y, g = Group, verbose=TRUE)
```

`wilcoxonZ` *Wilcoxon z statistic*

Description

Calculates the z statistic for a Wilcoxon two-sample, paired, or one-sample test.

Usage

```
wilcoxonZ(
  x,
  y = NULL,
  mu = 0,
  paired = FALSE,
  exact = FALSE,
  correct = FALSE,
  digits = 3
)
```

Arguments

`x`	A vector of observations.
`y`	For the two-sample and paired cases, a second vector of observations.
`mu`	For the one-sample case, the value to compare `x` to, as in `wilcox.test`
`paired`	As used in `wilcox.test`.
`exact`	As used in `wilcox.test`, default here is `FALSE`.
`correct`	As used in `wilcox.test`, default here is `FALSE`.
`digits`	The number of significant digits in the output.

Details

This function uses code from `wilcox.test`, and reports the `z` statistic, which is calculated by the original function but isn't returned.

The returned value will be NA if the function attempts an exact test.

For the paired case, the observations in `x` and and `y` should be ordered such that the first observation in `x` is paired with the first observation in `y`, and so on.

Value

A single statistic, `z`.

Author(s)

Salvatore Mangiafico, <`mangiafico@njaes.rutgers.edu`>, R Core Team

Examples

```
data(Pooh)
wilcoxonZ(x = Pooh$Likert[Pooh$Time==1], y = Pooh$Likert[Pooh$Time==2],
          paired=TRUE, exact=FALSE, correct=FALSE)
```

Index