

Package ‘Deducer’

November 2, 2009

Version 0.2-0

Date 2009-05-13

Title Deducer

Author Ian Fellows

Maintainer Ian Fellows <thefell@gmail.com>

Description An intuitive, cross-platform graphical data analysis system. It uses menus and dialogs to guide the user efficiently through the data manipulation and analysis process, and has an excel like spreadsheet for easy data frame visualization and editing. Deducer works best when used with the Java based R GUI JGR, but the dialogs can be called from the command line. Some integration with the Windows Rgui has been done.

Depends R (>= 2.7.0), rJava (>= 0.7),JavaGD, ggplot2, JGR(>= 1.7), car, multcomp,effects,foreign,lawstat

SystemRequirements Java (>= 5.0)

License GPL-2

URL <http://www.deducer.org/pmwiki/index.php?n=Main.DeducerManual>

Repository CRAN

Date/Publication 2009-11-02 07:46:03

R topics documented:

add.cross.strata.test	2
add.mantel.haenszel	3
add.test	4
as.matrix.cor.matrix	6
chi.noncentral.conf	6
chi.squared.test	7
contin.tests.to.table	9
contingency.tables	10

cor.matrix	11
deducer	12
define.groups	12
descriptive.table	13
extract.counts	14
fishers.exact.test	15
frequencies	16
ggcorplot	17
k.sample.test	18
kurtosis	19
likelihood.test	20
multi.test	21
one.sample.test	21
onesample.plot	22
oneway.plot	23
perm	23
perm.t.test	24
plot.cor.matrix	25
print.contin.table	26
print.contin.tests	27
print.contingency.tables	27
print.cor.matrix	28
print.freq.table	28
print.multi.test	29
qscatter_array	29
recode.variables	30
rocplot	31
skewness	32
sort.data.frame	33
summary.lm	34
table.to.data	35
two.sample.test	36
Index	37

add.cross.strata.test

Apply a Stratified test to a Contingency Table

Description

Applies and adds a hypothesis test to a contingency.tables object.

Usage

```
add.cross.strata.test(tables, name, htests, types=c("asymptotic", "monte.carlo", "exact"))
```

Arguments

tables	An object of class <code>contingency.tables</code>
name	The name of the hypothesis test
htests	A function or list of functions which take a three dimensional array as it's argument and returns an object of class <code>htest</code>
types	A character vector with the same number of items as <code>htests</code> , indicating what type of test was done

Value

A `contingency.tables` object identical to `tables`, but with the test applied to each table.

See Also

[add.mantel.haenszel](#) [add.test](#)

Examples

```
dat<-data.frame(a=rnorm(100)>.5,b=rnorm(100)>0,c=rnorm(100)>(-.5))
tables<-contingency.tables(
  row.vars=c("a"),
  col.vars=c("b"),
  stratum.var=c("c"),data=dat)
add.cross.strata.test(tables,"Mantel-Haenszel",list(function(x) mantelhaen.test(x,correct=FALSE)
  "asymptotic"))
tables
```

```
add.mantel.haenszel
```

Apply the Mantel-Haenszel test to a Contingency Table

Description

Applies and adds the Cochran-Mantel-Haenszel test to a `contingency.tables` object. The Cochran-Mantel-Haenszel tests the independence of two nominal variables, stratified by a third nominal variable, assuming no three way interaction.

Usage

```
add.mantel.haenszel(tables,conservative=FALSE)
```

Arguments

tables	An object of class <code>contingency.tables</code>
conservative	Should a continuity 'correction' be applied

Details

This is a convenience function wrapping `mantelhaen.test` in a `add.cross.strata.test` call. See [mantelhaen.test](#) for further details.

Value

A `contingency.tables` object identical to `tables`, but with the test applied to each table.

See Also

[add.cross.strata.test](#) [add.test](#) [mantelhaen.test](#)

Examples

```
dat<-data.frame(a=rnorm(100)>.5,b=rnorm(100)>0,c=rnorm(100)>(-.5))
tables1<-contingency.tables(
  row.vars=c("a"),
  col.vars=c("b"),
  stratum.var=c("c"),data=dat)
tables1<-add.mantel.haenszel(tables1)
print(tables1,prop.r=TRUE,prop.c=TRUE,prop.t=FALSE)
```

add.test

Apply a test to a Contingency Tables object

Description

Applies and adds a test to a `contingency.tables` object.

Usage

```
add.test(tables,name,htests,types=c("asymptotic","monte.carlo","exact"))
add.chi.squared(tables, conservative = FALSE, simulate.p.value = FALSE, B = 10000)
add.likelihood.ratio(tables, conservative = FALSE, simulate.p.value = FALSE, B = 10000)
add.fishers.exact(tables, conservative = FALSE, simulate.p.value = FALSE, B = 10000)
add.correlation(tables,method=c("spearman","kendall"))
add.kruskal(tables,nominal=c("both","rows","cols"))
```

Arguments

<code>tables</code>	An object of class <code>contingency.tables</code>
<code>name</code>	Name of the test
<code>htests</code>	A function or list of functions which take a matrix as it's argument and returns an object of class <code>htest</code>
<code>types</code>	A character vector with the same number of items as <code>htests</code> , indicating what type of test was done
<code>conservative</code>	Should a conservative p-value be computed. i.e. One with a continuity correction for asymptotic tests and not using the mid p-value for exact and approximate tests
<code>simulate.p.value</code>	If TRUE p-values will be computed via monte carlo simulation
<code>B</code>	the number of samples for the monte carlo simulation
<code>method</code>	the type of correlation
<code>nominal</code>	Should the rows or columns be considered nominal.

Details

`add.test` applies a supplied list of tests to all of the tables in `tables`.

`add.chi.squared` is a wrapper function applying the `chi.squared.test` function to each table. `add.likelihood.ratio` is a wrapper function applying the `likelihood.test` function to each table. `add.fishers.exact` is a wrapper function applying the `fishers.exact.test` function to each table. `add.correlation` is a wrapper function applying the `cor.test` function to each table. `add.kruskal` is a wrapper function applying the `kruskal.test` function to each table.

Value

A `contingency.tables` object identical to `tables`, but with the test applied to each table.

See Also

`add.cross.strata.test` `chi.squared.test` `likelihood.test` `fishers.exact.test` `cor.test` `kruskal.test`

Examples

```
dat<-data.frame(a=rnorm(100)>.5,b=rnorm(100)>0,c=rnorm(100)>(-.5))
tables<-contingency.tables(
  row.vars=c("a"),
  col.vars=c("b"),
  stratum.var=c("c"),data=dat)
tables<-add.chi.squared(tables,simulate.p.value=TRUE,B=10000)
tables<-add.likelihood.ratio(tables)
tables<-add.fishers.exact(tables)
tables<-add.correlation(tables,method='kendall')
tables<-add.kruskal(tables)
tables<-add.mantel.haenszel(tables)
```

```
print (tables)
remove (tables)
```

```
as.matrix.cor.matrix
      as.matrix method
```

Description

as matrix

Usage

```
## S3 method for class 'cor.matrix':
as.matrix(x, ...)
```

Arguments

x Object of class "cor.matrix".
 ... further arguments. unused

Value

a matrix

```
chi.noncentral.conf
      Non-central Chi-Squared Confidence Interval
```

Description

Confidence interval for the Non-centrality parameter of Non-central chi-squared distribution

Usage

```
chi.noncentral.conf (chival, df, conf, prec=.00001)
```

Arguments

chival The observed Chi-Squared value
 conf The confidence level (e.g. .95)
 df Degrees of freedom
 prec Precision of estimate

Value

A 2X2 matrix whose rows represent the upper and lower bounds, and whose columns represent the parameter value and upper tail percentiles.

References

Smithson, M.J. (2003). Confidence Intervals, Quantitative Applications in the Social Sciences Series, No. 140. Thousand Oaks, CA: Sage.

See Also

[Chisquare](#) [chi.squared.test](#)

Examples

```
chi.noncentral.conf(6,1,.95)
#           Result:

#           Non-Central
#Lower    0.2089385 0.97500899
#Upper    19.4443359 0.02499302
```

`chi.squared.test` *Pearson's Chi-squared Test for Contingency Tables*

Description

Performs performs the chi-squared contingency table test

Usage

```
chi.squared.test(x, y = NULL, conservative = FALSE, cramers.v.conf=.95, simulate.p.
```

Arguments

<code>x</code>	A vector or a matrix
<code>y</code>	A vector that is ignored if <code>x</code> is a matrix and required if <code>x</code> is a vector
<code>conservative</code>	If <code>FALSE</code> , the usual chi-squared test is performed and if <code>simulate.p.value=TRUE</code> , the monte carlo mid p-value is returned. Otherwise, Yates' continuity correction is applied, and if <code>simulate.p.value=TRUE</code> the monte carlo conservative p-value is returned
<code>cramers.v.conf</code>	The confidence level for the confidence interval around the Cramer's V effect size
<code>simulate.p.value</code>	a logical indicating whether to compute p-values by monte carlo simulation
<code>B</code>	An integer specifying the number of replicates used in the monte carlo test

Details

If `x` is a matrix with at least two rows and columns, it is taken as a two-dimensional contingency table. The entries of `x` must be non-negative integers. Otherwise, `x` and `y` must be vectors or factors of the same length; incomplete cases are removed, the objects are coerced into factor objects, and the contingency table is computed from these. Then, Pearson's chi-squared test of the null hypothesis that the joint distribution of the cell counts in a 2-dimensional contingency table is the product of the row and column marginals is performed.

If `simulate.p.value` is `FALSE` and `conservative` is `FALSE` the p-value is computed from the asymptotic chi-squared distribution of the test statistic; continuity correction is not applied.

If `simulate.p.value` is `TRUE` and `conservative` is `FALSE` the p-value is computed via for a Monte Carlo test (Hope, 1968) with `B` replicates. The mid p-value is returned. The mid p-value is defined as the proportion of replicates less than the observed chi-squared value plus one half times the proportion of replicates equal to the observed chi-squared value.

If `simulate.p.value` is `FALSE` and `conservative` is `TRUE` the p-value is computed from the asymptotic chi-squared distribution of the test statistic with the Yates continuity correction applied only in the case of a 2-by-2 table.

If `simulate.p.value` is `TRUE` and `conservative` is `TRUE` the p-value is computed via for a Monte Carlo test (Hope, 1968) with `B` replicates. The conservative p-value is returned, which is defined as the proportion of replicates less than or equal to the observed chi-squared value.

In the contingency table case simulation is done by random sampling from the set of all contingency tables with given marginals, and works only if the marginals are strictly positive. (A C translation of the algorithm of Patefield (1981) is used.) Continuity correction is never used, and the statistic is quoted without it. Note that this is not the usual sampling situation for the chi-squared test but rather that for Fisher's exact test.

Value

A list with class "htest" containing the following components:

<code>statistic</code>	the value the chi-squared test statistic.
<code>parameter</code>	the degrees of freedom of the approximate chi-squared distribution of the test statistic.
<code>p.value</code>	the p-value for the test.
<code>method</code>	a character string indicating the type of test performed, and whether Monte Carlo simulation or continuity correction was used.
<code>data.name</code>	a character string giving the name(s) of the data.
<code>observed</code>	the observed counts.
<code>expected</code>	the expected counts under the null hypothesis.
<code>residuals</code>	the Pearson residuals, $(\text{observed} - \text{expected}) / \sqrt{\text{expected}}$.
<code>estimate</code>	Cramer's V
<code>conf.int</code>	A confidence interval for Cramer's V

References

Hope, A. C. A. (1968) A simplified Monte Carlo significance test procedure. J. Roy, Statist. Soc. B 30.

Patefield, W. M. (1981) Algorithm AS159. An efficient method of generating r x c tables with given row and column totals. Applied Statistics 30.

See Also

[chisq.test](#) [likelihood.test](#)

Examples

```
data(InsectSprays)
chi.squared.test(InsectSprays$count>7, InsectSprays$spray)
```

```
contin.tests.to.table
      contin.tests.to.table
```

Description

Makes a nice table out of a `contin.tests` object

Usage

```
contin.tests.to.table(tests, test.digits=3, ...)
```

Arguments

<code>tests</code>	a <code>contin.tests</code> object
<code>test.digits</code>	The number of digits to round to
<code>...</code>	other paramaters

Value

A nice table

contingency.tables *Contingency Tables*

Description

Creates a contingency.tables object

Usage

```
contingency.tables(row.vars, col.vars, stratum.var, data, missing.include=FALSE )
```

Arguments

row.vars	A character vector of variable names in data
col.vars	A character vector of variable names in data
stratum.var	A variable name in data
data	A data.frame
missing.include	A logical indicating whether a missing category should be included in the table

Value

A list with class "contingency.tables." Each element of the list is a single contingency table of class "contin.table" corresponding to each combination of elements of row.vars and col.vars stratified by stratum.var

See Also

[extract.counts](#)

Examples

```
temp.data<-data.frame(a=rnorm(100)>0,b=rnorm(100)>0,gender=rep(c("male","female"),50))
#a vs. b stratified by gender
tab<-contingency.tables("a","b","gender",data=temp.data)
tab

##add in chi-squared tests
tab<-add.chi.squared(tab)
tab
```

<code>cor.matrix</code>	<i>cor.matrix</i>
-------------------------	-------------------

Description

Creates a correlation matrix

Usage

```
cor.matrix(variables, with.variables, data, test=cor.test, ...)
```

Arguments

<code>variables</code>	An expression denoting a set of variable.
<code>with.variables</code>	An optional expression denoting a set of variables to correlate with <code>variables</code> . If nothing is specified, all variables in <code>variables</code> are correlated with themselves.
<code>data</code>	A <code>data.frame</code> from which the variables and factor will be selected.
<code>test</code>	A function whose first two arguments are the variables upon which the correlation will be calculated, and whose result is an object of class <code>htest</code> .
<code>...</code>	further arguments for <code>test</code> .

Details

The `variables` and `with.variables` arguments work by first replacing column names in the expression with the corresponding column numbers in the data frame and then using the resulting integer vector to index the columns. This allows the use of the standard indexing conventions so that for example ranges of columns can be specified easily, or single columns can be dropped (see the examples).

Value

A `multi.test` object, representing a table of the results of `func` applied to each of the variables.

See Also

[t.test](#) [ks.test](#) [wilcox.test](#)

Examples

```
dat<-data.frame(aa=rnorm(100),bb=rnorm(100),cc=rnorm(100),dd=rnorm(100))
dat$aa<-dat$aa+dat$dd
dat$cc<-dat$cc+dat$aa
cor.matrix(aa:dd, , data=dat, test=cor.test)
cor.matrix(-dd, dd, data=dat, test=cor.test, method="kendall")
cor.matrix(c(aa, cc), c(dd, bb), data=dat, test=cor.test, method="spearman")
```

deducer *GUI Access functions*

Description

splits a variable into two groups

Usage

```
deducer (cmd=NULL)
```

```
data.viewer()
```

Arguments

cmd The command to be executed

define.groups *define.groups*

Description

splits a variable into two groups

Usage

```
define.groups (factor.var, data, cut=NULL, group1=NULL, group2=NULL)
```

Arguments

factor.var An expression indicating a two-level variable dividing variable into two groups.

data A data.frame from which the variables and factor will be selected.

cut An optional cut point dividing factor into two groups.

group1 An optional vector of levels of factor defining group 1.

group2 An optional vector of levels of factor defining group 2.

Value

data, with factor.var recoded into two groups

descriptive.table *Table of Descriptives*

Description

Table of descriptive statistics, possibly stratified

Usage

```
descriptive.table(dat, func.names = c("Mean", "St. Deviation", "Median",
  "25th Percentile", "75th Percentile", "Minimum", "Maximum",
  "Skew", "Kurtosis", "Valid N"), strata, func.additional)
```

Arguments

dat	A data.frame containing the variables on which to run descriptive statistics. Variables to be stratified on should also be included.
func.names	A character vector of built-in statistics
strata	A character vector of dat variables to stratify on. Statistics be calculated within each level (or combination of levels).
func.additional	A named list of functions. Each function should take a numeric vector as its argument, and return a single value

Value

Returns a list of matrix objects containing descriptive information on all variables in dat. One for each level or combination of levels in strata.

See Also

[frequencies mean by](#)

Examples

```
data<-data.frame(a=rnorm(100),b=rnorm(100),male=rnorm(100)>0)
##means and standard deviations
descriptive.table(data[,1:2],func.names=c("Mean","St. Deviation","Valid N"))
##stratifying by gender
descriptive.table(data,func.names=c("Mean","St. Deviation","Valid N"),strata="male")
func.list=list(mean.deviance=function(x) mean(abs(x-mean(x))))

##Adding deviance as a statistic
descriptive.table(data,func.names=c("Mean","St. Deviation","Valid N"),strata="male",func.add
```

extract.counts *Extract Contingency Table Arrays*

Description

Extracts the counts of a contingency.tables object

Usage

```
extract.counts(tables)
```

Arguments

tables A contingency.table object

Value

A named list of three dimensional arrays. One for each contin.table in tables

See Also

[contingency.tables](#)

Examples

```
temp.data<-data.frame(a=rnorm(100)>0,b=rnorm(100)>0,gender=rep(c("male","female"),50))
#a vs. b stratified by gender
tab<-contingency.tables("a","b","gender",data=temp.data)
tab

##extract counts
extract.counts(tab)

##Yields something like the following:
#`a by b`
# , , female
#
#      FALSE TRUE
#FALSE     11   9
#TRUE      15  15
#
# , , male
#
#      FALSE TRUE
#FALSE     10  10
#TRUE      22   8
```

`fishers.exact.test` *Fisher's Exact Test of independence*

Description

Performs Fisher's exact test of independence in a contingency table conditional on the marginal totals

Usage

```
fishers.exact.test(x, y = NULL, conservative=FALSE, simulate.p.value = FALSE, B = 1000,
                  workspace = 2e+05, control = list())
```

Arguments

<code>x</code>	A vector or a matrix
<code>y</code>	A vector that is ignored if <code>x</code> is a matrix and required if <code>x</code> is a vector
<code>conservative</code>	If <code>FALSE</code> , the exact mid p-value value is returned; Otherwise the exact conservative p-value is used.
<code>simulate.p.value</code>	a logical indicating whether to compute p-values by Monte Carlo simulation
<code>B</code>	The number of replications to use in the simulation
<code>workspace</code>	an integer specifying the size of the workspace used in the network algorithm. In units of 4 bytes. Only used for non-simulated p-values larger than 2 by 2 tables.
<code>control</code>	a list with named components for low level algorithm control. At present the only one used is "mult", a positive integer ≥ 2 with default 30 used only for larger than 2 by 2 tables. This says how many times as much space should be allocated to paths as to keys: see file <code>fexact.c</code> in the sources of the package 'stats'.

Details

The same as `fisher.test` except that the mid p-value is supported, and is the default.

Value

A list with class "htest" containing the following components:

<code>p.value</code>	the p-value for the test.
<code>method</code>	a character string indicating the type of test performed, and whether Monte Carlo simulation or continuity correction was used.
<code>data.name</code>	a character string giving the name(s) of the data.
<code>statistic</code>	currently always NA.
<code>parameter</code>	currently always NA.

See Also

[fisher.test](#) [chi.squared.test](#) [likelihood.test](#)

Examples

```
## A r x c table Agresti (2002, p. 57) Job Satisfaction
Job <- matrix(c(1,2,1,0, 3,3,6,1, 10,10,14,9, 6,7,12,11), 4, 4,
dimnames = list(income=c("< 15k", "15-25k", "25-40k", "> 40k"),
satisfaction=c("VeryD", "LittleD", "ModerateS", "VeryS")))
fishers.exact.test(Job)
```

frequencies

Frequency Tables

Description

Creates a set of frequency tables.

Usage

```
frequencies(data, r.digits=1)
```

Arguments

`data` A data.frame containing the variables on which to run frequencies
`r.digits` how many digits should the percentages be rounded to

Value

Returns a list of `freq.table` objects. One for each variable in `data`.

See Also

[table](#) [xtabs](#) [descriptive.table](#) [prop.table](#)

Examples

```
dat<-data.frame(rnorm(100)>0, trunc(runif(100,0,5)))
##rounding to 1
frequencies(dat)
##rounding to 4
frequencies(dat,4)
```

ggcorplot *Correlation matrix*

Description

Plots a correlation matrix

Usage

```
ggcorplot (cor.mat, data, lines=TRUE, line.method=c("lm", "loess"), type="points",
           alpha=.25, main="auto", var_text_size=5,
           cor_text_limits=c(5, 25), level=.05)
```

Arguments

cor.mat	a cor.matrix object to plot
data	the data.frame used to compute the correlation matrix
lines	Logical. Should regression lines be drawn.
type	type of plot. "points" or "bins"
line.method	Character. Type of regression line.
alpha	numeric. level of alpha transparency for the points.
main	Title of the plot. defaults to the method of cor.mat.
var_text_size	size of the diagonal variable names.
cor_text_limits	lower and upper bounds for the size of the correlation text.
level	the size of the test differentiated by text color.

Author(s)

Mike Lawrence and Ian Fellows

See Also

[cor.matrix](#) [qscatter_array](#)

Examples

```
data(mtcars)
corr.mat1<-cor.matrix(variables=mpg:carb,,
                      data=mtcars,
                      test=cor.test,
                      method='spearman',
                      alternative="two.sided", exact=FALSE)

ggcorplot(corr.mat1, data = mtcars)
```

`k.sample.test` *K Sample Test*

Description

Performs a K independent sample test.

Usage

```
k.sample.test(variables, factor.var, data, test=oneway.test, ...)
```

Arguments

<code>variables</code>	An expression indicating the variables upon which the test will be done.
<code>factor.var</code>	An expression indicating a two-level variable dividing variable into two groups.
<code>data</code>	A <code>data.frame</code> from which the variables and factor will be selected.
<code>test</code>	A function whose first argument is a formula with the outcome on the lhs and the factor on the rhs. The second argument should be the data to be used for the formula. The result of the function should be an object of class <code>htest</code> .
<code>...</code>	further arguments for <code>func</code>

Details

The `variables` and `factor` arguments work by first replacing column names in the expression with the corresponding column numbers in the data frame and then using the resulting integer vector to index the columns. This allows the use of the standard indexing conventions so that for example ranges of columns can be specified easily, or single columns can be dropped (see the examples).

Value

A `multi.test` object, representing a table of the results of `func` applied to each of the variables.

See Also

[oneway.test](#) [kruskal.test](#) [wilcox.test](#)

Examples

```
dat<-data.frame(a=rnorm(100),b=rnorm(100),c=rnorm(100),d=cut(rnorm(100),4))
k.sample.test(a:b,d,dat)
k.sample.test(-d,d,dat,,var.equal=FALSE)
k.sample.test(c(a,c),d,dat,kruskal.test)
```

kurtosis	<i>Sample Kurtosis</i>
----------	------------------------

Description

Computes the kurtosis

Usage

```
kurtosis(x, na.rm = FALSE, type = 3)
```

Arguments

<code>x</code>	a numeric vector containing the values whose kurtosis is to be computed.
<code>na.rm</code>	a logical value indicating whether NA values should be stripped before the computation proceeds.
<code>type</code>	an integer between 1 and 3 selecting one of the algorithms for computing kurtosis detailed below.

Details

See `kurtosis` in package `e1071`. This function is under development, and in the future will perform a hypothesis test.

Value

The estimated kurtosis of `x`.

See Also

[skewness](#) [var](#) [mean](#)

Examples

```
x <- rnorm(100)
kurtosis(x)
```

likelihood.test *Likelihood Ratio (G test) for contingency tables*

Description

Performs a likelihood ratio test of independence

Usage

```
likelihood.test(x, y=NULL, conservative=FALSE)
```

Arguments

`x` A vector or a matrix
`y` A vector that is ignored if `x` is a matrix and required if `x` is a vector
`conservative` If TRUE, the Williams' continuity correction is used

Value

A list with class "hstest" containing the following components:

`statistic` the value the chi-squared test statistic.
`parameter` the degrees of freedom of the approximate chi-squared distribution of the test statistic.
`p.value` the p-value for the test.
`method` a character string indicating the type of test performed, and whether the continuity correction was used.
`data.name` a character string giving the name(s) of the data.

See Also

[chisq.test](#) [chi.squared.test](#)

Examples

```
data(InsectSprays)
likelihood.test(InsectSprays$count>7, InsectSprays$spray)
```

multi.test	<i>multi.test</i>
------------	-------------------

Description

Creates a table from a list of htests

Usage

```
multi.test(tests)
```

Arguments

tests	A named list of htest objects representing the same test applied to a number of different conditions or variables.
-------	--

Value

A multi.test object, representing a table of the htest objects.

one.sample.test	<i>One Sample Test</i>
-----------------	------------------------

Description

Performs a one sample test.

Usage

```
one.sample.test(variables, data, test=t.test, ...)
```

Arguments

variables	An expression indicating the variables upon which the test will be done.
data	A data.frame from which the variables and factor will be selected.
test	A function whose first two arguments are the two-samples to be tested, and whose result is an object of class htest.
...	further arguments for func

Details

The variables arguments work by first replacing column names in the expression with the corresponding column numbers in the data frame and then using the resulting integer vector to index the columns. This allows the use of the standard indexing conventions so that for example ranges of columns can be specified easily, or single columns can be dropped (see the examples).

Value

A `multi.test` object, representing a table of the results of `test` applied to each of the variables.

See Also

`t.test` `shapiro.test`

Examples

```
dat<-data.frame(a=rnorm(100),b=runif(100),c=rnorm(100),d=rnorm(100)>(-.5))
one.sample.test(a:d,dat,t.test)
one.sample.test(a:c,dat,shapiro.test)
```

`onesample.plot` *onesample.plot*

Description

plots for one sample tests

Usage

```
onesample.plot(variables,data,test.value=0,scale=TRUE,type="hist",alpha=.2)
```

Arguments

<code>variables</code>	An expression denoting a set of variable.
<code>data</code>	A <code>data.frame</code> from which the variables will be selected.
<code>test.value</code>	null hypothesis test value
<code>scale</code>	scale variables
<code>type</code>	type of plot. 'hist' or 'box' are allowed
<code>alpha</code>	transparency of points for box plot

Examples

```
n<-50
dat<-data.frame(a=rnorm(n),b=rnorm(n,1),c=rnorm(n,0,30))
onesample.plot(variables=c(a,b,c),data=dat,test.value=0.0,type='box',alpha=0.2)
onesample.plot(variables=c(a,b,c),data=dat,test.value=0.0,type='hist')
```

`oneway.plot`*One Way PLOT*

Description

plots a categorical variable against a series of continuous variables

Usage

```
oneway.plot(variables, factor.var, data, alpha=.2, box=TRUE, points=TRUE, scale=TRUE)
```

Arguments

<code>variables</code>	an expression indicating the continuous variables of data
<code>factor.var</code>	an expression for the categorical variable
<code>data</code>	the data
<code>alpha</code>	alpha transparency level for the points.
<code>box</code>	prints boxplot
<code>points</code>	prints jitter plot
<code>scale</code>	standardize the variables prior to plotting

Value

a ggplot object

Examples

```
oneway.plot(DriversKilled:PetrolPrice, law, as.data.frame(Seatbelts))
```

`perm`*Vector Permutations*

Description

Enumerates all permutations of a vector

Usage

```
perm(vec, duplicates=FALSE)
```

Arguments

vec The vector to permute
duplicates Should duplicate permutations be listed

Value

Returns a matrix where each row is a permutation of vec. All possible permutations are listed, and if `duplicates=TRUE` non-unique permutations are also listed.

See Also

[sample](#)

Examples

```
perm(1:4)
perm(LETTERS[4:8])
```

perm.t.test

Permutation t-test

Description

Two Sample t-test via monte-carlo permutation

Usage

```
perm.t.test(x,y,statistic=c("t","mean"),
            alternative=c("two.sided", "less", "greater"), midp=TRUE, B)
```

Arguments

x a numeric vector containing the first sample
y a numeric vector containing the second sample
statistic The statistic to be permuted. See details
alternative The alternative hypothesis
midp should the mid p-value be used
B The number of monte-carlo samples to be generated

Details

This function performs a two sample permutation test. If the mean is permuted, then the test assumes exchangeability between the two samples. if the t-statistic is used, the test assumes either exchangeability or a sufficiently large sample size. Because there is little lost in the way of power, and the assumptions are weaker, the t-statistic is used by default.

Value

A list with class "htest" containing the following components:

statistic	The observed value of the statistic.
p.value	the p-value for the test.
method	a character string indicating the type of test performed.
data.name	a character string giving the name(s) of the data.
B	The number of samples generated
alternative	the direction of the test

See Also

[t.test](#)

Examples

```
perm.t.test(rnorm(100), runif(100, -.5, .5))
```

plot.cor.matrix *Plot method*

Description

Produces a circle plot for an object of class "plot.cor.matrix"

Usage

```
## S3 method for class 'cor.matrix':
plot(x, y=NULL, size=10, ...)
```

Arguments

x	Object of class "cor.matrix".
y	unused
size	maximum radius size
...	further arguments. unused

Value

a ggplot object

print.contin.table *Print method*

Description

Print object of class "contin.table" in nice layout.

Usage

```
## S3 method for class 'contin.table':
print(x,digits=3,prop.r=TRUE,prop.c=TRUE,prop.t=TRUE,
      expected.n=FALSE,residuals=FALSE,std.residuals=FALSE,
      adj.residuals=FALSE,no.tables=FALSE)
```

Arguments

x	Object of class "contin.table".
digits	Number of digits to round to.
prop.r	Logical. print row proportions.
prop.c	Logical. print column proportions.
prop.t	Logical. print proportions.
expected.n	Logical print expected cell counts.
residuals	Logical. print residuals.
std.residuals	Logical. print standardized residuals.
adj.residuals	Logical. Print Adjusted residuals
no.tables	Logical. Suppress tables
...	further arguments

Value

none

```
print.contin.tests Print method
```

Description

Print object of class "contin.tests" in nice layout.

Usage

```
## S3 method for class 'contin.tests':
print(x, test.digits, ...)
```

Arguments

x	Object of class "contin.tests".
test.digits	Number of digits to be printed
...	further arguments to be passed to or from methods.

Value

none

```
print.contingency.tables
      Print method
```

Description

Print object of class "contingency.tables" in nice layout.

Usage

```
## S3 method for class 'contingency.tables':
print(x, digits=3, prop.r=TRUE, prop.c=TRUE, prop.t=TRUE,
      expected.n=FALSE, no.tables=FALSE, ...)
```

Arguments

x	Object of class "contin.table".
digits	Number of digits to round to.
prop.r	Logical. print row proportions.
prop.c	Logical. print column proportions.
prop.t	Logical. print proportions.
expected.n	Logical print expected cell counts.
no.tables	Logical. Suppress tables
...	further arguments

Value

none

print.cor.matrix *Print method*

Description

Print object of class "cor.matrix" in nice layout.

Usage

```
## S3 method for class 'cor.matrix':  
print(x, digits=4, N=TRUE, CI=TRUE, stat=TRUE, p.value=TRUE, ...)
```

Arguments

x	Object of class "cor.matrix".
digits	Number of digits to round to.
N	Logical. print a row for sample size.
CI	Logical. print a row for confidence intervals if they exist.
stat	Logical. print a row for test statistics.
p.value	Logical. print a row for p-values.
...	further arguments

Value

none

print.freq.table *Print method*

Description

Print object of class "freq.table" in nice layout.

Usage

```
## S3 method for class 'freq.table':  
print(x, ...)
```


Arguments

<code>row.vars</code>	An expression denoting a set of variable.
<code>col.vars</code>	An expression denoting a set of variable.
<code>data</code>	A data.frame from which the variables will be selected.
<code>x.lab</code>	A label for the x axis
<code>y.lab</code>	A label for the y axis
<code>main</code>	A label for the plot
<code>common.scales</code>	should common x and y scales be used.
<code>alpha</code>	alpha transparency

Details

The `row.vars` and `col.vars` arguments work by first replacing column names in the expression with the corresponding column numbers in the data frame and then using the resulting integer vector to index the columns. This allows the use of the standard indexing conventions so that for example ranges of columns can be specified easily, or single columns can be dropped.

Examples

```
data(mtcars)
qscatter_array(cyl:drat,wt:carb,data=mtcars,common.scales=FALSE)
qscatter_array(cyl:drat,wt:carb,data=mtcars,common.scales=TRUE)
```

`recode.variables` *Recode*

Description

Recodes a set of variables according to a set of rules

Usage

```
recode.variables(data, recodes)
```

Arguments

<code>data</code>	A data.frame to be recoded
<code>recodes</code>	Definition of the recoding rules. See details

Details

recodes contains a set of recoding rules separated by ";". There are three different types of recoding rules:

1. The simplest codes one value to another. If we wish to recode 1 into 2, we could use the rule "1->2;".
2. A range of values can be coded to a single value using "1:3->4;". This rule would code all values between 1 and 3 inclusive into 4. For factors, a value is between two levels if it is between them in the factor ordering. One sided ranges can be specified using the Lo and Hi key words (e.g."Lo:3->0; 4:Hi->1")
3. Default conditions can be coded using "else." For example, if we wish to recode all values >=0 to 1 and all values <0 to missing, we could use ("0:Hi->1; else->NA")

Value

returns a recoded `data.frame`

See Also

`cut` recode in package 'Hmisc'

Examples

```
data<-data.frame(a=rnorm(100),b=rnorm(100),male=rnorm(100)>0)
recode.variables(data[c("a","b")] , "Lo:0 -> 0;0:Hi -> 1;")
data[c("male")] <- recode.variables(data[c("male")] , "1 -> 'Male';0 -> 'Female';else -> NA;")
```

rocplot

ROC Plot for a logistic regression model

Description

Plots the ROC Curve

Usage

```
rocplot(logistic.model,diag=TRUE,pred.prob.labels=FALSE,prob.label.digits=3,AUC=TRUE)
```

Arguments

`logistic.model`
a glm object with binomial link function.

`diag`
a logical value indicating whether a diagonal reference line should be displayed.

`pred.prob.labels`
a logical value indicating whether the predictive probabilities should be displayed

`prob.label.digits` The number of digits of the predictive probabilities to be displayed.

`AUC` a logical value indicating whether the estimated area under the curve should be displayed

Value

a ggplot object

Examples

```
model.glm <- glm(formula=income>5930.5 ~ education + women + type,family=binomial(),data=Pre)
rocplot(model.glm)
```

<code>skewness</code>	<i>Sample Skewness</i>
-----------------------	------------------------

Description

Computes the skewness

Usage

```
skewness(x, na.rm = FALSE, type = 3)
```

Arguments

`x` a numeric vector containing the values whose skewness is to be computed.

`na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.

`type` an integer between 1 and 3 selecting one of the algorithms for computing skewness detailed below.

Details

See `skewness` in package `e1071`. This function is under development, and in the future will perform a hypothesis test.

Value

The estimated skewness of `x`.

See Also

[kurtosis](#)

Examples

```
x <- rnorm(100)
skewness(x)
```

`sort.data.frame` *Sort Data*

Description

Sorts a data frame

Usage

```
## S3 method for class 'data.frame':  
sort(x, decreasing, by, ...)
```

Arguments

<code>x</code>	A <code>data.frame</code> to be sorted
<code>decreasing</code>	unused
<code>by</code>	A character, a one sided formula, or an expression indicating the sorting order
<code>...</code>	further arguments

Details

If `by` is a formula, or a character vector coerce-able into a formula, `x` is sorted by each element of the formula, with ties broken by subsequent elements. Elements preceded by a '-' indicate descending order, otherwise ascending order is used. Parentheses or any formula operator other than + and - are ignored, so sorting by `a*b` will sort based on the product of `a` and `b`.

If `by` is not a formula, a `~` is appended to the left hand side of the call, and coerced into a formula.

The `decreasing` argument is included for generic method consistency, and is not used.

Value

returns `x`, sorted.

See Also

[sort order](#)

Examples

```
data(mtcars)  
  
#sort by the number of cylenders  
sort(mtcars, by= ~cyl)  
sort(mtcars, by= cyl) #identical: no need for ~  
  
#sort in descending order  
sort(mtcars, by= -cyl)
```

```

#break ties with horse power
sort(mtcars,by= cyl +hp )
sort(mtcars,by= cyl -hp )

#randomly permute the data
sort(mtcars,by= rnorm(nrow(mtcars)) )

#reverse order
sort(mtcars,by= nrow(mtcars):1 )

#sort by squared deviation from mean hp
sort(mtcars,by= -(hp-mean(hp))^2 )
sort(mtcars,by= "-(hp-mean(hp))^2" ) #identical

```

summary.lm

Summary table for a linear model

Description

Computes the coefficients, std. errors, t values, and p-values for a linear model in the presence of possible heteroskedasticity.

Usage

```

## S3 method for class 'lm':
summary(object,correlation=FALSE,symbolic.cor = FALSE,white.adjust=FALSE,...)

```

Arguments

object	an object of class lm.
correlation	a logical value indicating whether parameter correlations should be printed.
symbolic.cor	logical. If TRUE, print the correlations in a symbolic form (see symnum) rather than as numbers. Effective only if white.adjust is FALSE.
white.adjust	value passed to hccm indicating the type of robust adjustment to be used. If TRUE, type is assumed to be 'hc3'
...	additional parameters passed to stats::summary.lm

Details

If white.adjust is false, the function returns a value identical to stats::summary.lm. Otherwise, robust summaries are computed

Value

A summary table

Examples

```
ctl <- c(4.17,5.58,5.18,6.11,4.50,4.61,5.17,4.53,5.33,5.14)
trt <- c(4.81,4.17,4.41,3.59,5.87,3.83,6.03,4.89,4.32,4.69)
group <- gl(2,10,20, labels=c("Ctl","Trt"))
weight <- c((ctl-mean(ctl))*10+mean(ctl), trt)
lm.D9 <- lm(weight ~ group)
summary(lm.D9,white.adjust=TRUE)
```

table.to.data *Table -> data.frame*

Description

Creates a data.frame from a table

Usage

```
table.to.data(x)
```

Arguments

x A matrix or table representing the cross tabulation of two variables

Value

A two column data.frame where each row is an observation and each column is a variable.

See Also

[xtabs](#)

Examples

```
tab<-matrix(c(4,5,6,9,7,3),ncol=3)
tab
table.to.data(tab)
```

two.sample.test *Two Sample Test*

Description

Performs a two independent sample test.

Usage

```
two.sample.test(variables, factor.var, data, test=t.test, ..., cut=NULL, group1=NULL, group2=NULL)
```

Arguments

variables	An expression indicating the variables upon which the test will be done.
factor.var	An expression indicating a two-level variable dividing variable into two groups.
data	A data.frame from which the variables and factor will be selected.
test	A function whose first two arguments are the two-samples to be tested, and whose result is an object of class htest.
...	further arguments for func
cut	An optional cut point dividing factor into two groups.
group1	An optional vector of levels of factor defining group 1.
group2	An optional vector of levels of factor defining group 2.

Details

The variables and factor arguments work by first replacing column names in the expression with the corresponding column numbers in the data frame and then using the resulting integer vector to index the columns. This allows the use of the standard indexing conventions so that for example ranges of columns can be specified easily, or single columns can be dropped (see the examples).

Value

A multi.test object, representing a table of the results of func applied to each of the variables.

See Also

[t.test](#) [ks.test](#) [wilcox.test](#)

Examples

```
dat<-data.frame(a=rnorm(100),b=rnorm(100),c=rnorm(100),d=rnorm(100)>(-.5))
two.sample.test(a:b,d,dat,t.test)
two.sample.test(-d,d,dat,ks.test)
two.sample.test(c(a,c),d,dat,wilcox.test)
```

Index

`add.chi.squared` (*add.test*), 4
`add.correlation` (*add.test*), 4
`add.cross.strata.test`, 2, 3, 5
`add.fishers.exact` (*add.test*), 4
`add.kruskal` (*add.test*), 4
`add.likelihood.ratio` (*add.test*), 4
`add.mantel.haenszel`, 2, 3
`add.test`, 2, 3, 4
`as.matrix.cor.matrix`, 5

`by`, 13

`chi.noncentral.conf`, 6
`chi.squared.test`, 4–6, 7, 15, 19
`chisq.test`, 8, 19
Chisquare, 6
`contin.tests.to.table`, 9
`contingency.tables`, 9, 13
`cor.matrix`, 10, 17
`cor.test`, 4, 5
`cut`, 30

`data.viewer` (*deducer*), 11
`deducer`, 11
`define.groups`, 12
`descriptive.table`, 12, 16

`extract.counts`, 10, 13

`fisher.test`, 15
`fishers.exact.test`, 4, 5, 14
`frequencies`, 13, 15

`ggcorplot`, 16

`k.sample.test`, 17
`kruskal.test`, 4, 5, 18
`ks.test`, 11, 35
`kurtosis`, 18, 31

`likelihood.test`, 4, 5, 8, 15, 19

`mantelhaen.test`, 3
`mean`, 13, 18
`multi.test`, 20

`one.sample.test`, 20
`onesample.plot`, 21
`oneway.plot`, 22
`oneway.test`, 18
`order`, 32

`perm`, 22
`perm.t.test`, 23
`plot.cor.matrix`, 24
`print.contin.table`, 25
`print.contin.tests`, 26
`print.contingency.tables`, 26
`print.cor.matrix`, 27
`print.freq.table`, 27
`print.multi.test`, 28
`prop.table`, 16

`qscatter_array`, 17, 28

`recode.variables`, 29
`rocplot`, 30

`sample`, 23
`shapiro.test`, 21
`skewness`, 18, 31
`sort`, 32
`sort.data.frame`, 32
`summary.lm`, 33

`t.test`, 11, 21, 24, 35
`table`, 16
`table.to.data`, 34
`two.sample.test`, 35

`var`, 18

`wilcox.test`, 11, 18, 35

`xtabs`, 16, 34