# Package 'reporttools'

October 14, 2022

**Type** Package

**Title** Generate ``LaTeX"" Tables of Descriptive Statistics

**Version** 1.1.3

**Date** 2021-10-10

**Author** Kaspar Rufibach

**Maintainer** Kaspar Rufibach <kaspar.rufibach@gmail.com>

**Depends** xtable

**Imports** stats

**Suggests** survival

**Description**

These functions are especially helpful when writing reports of data analysis using ``Sweave".

**License** GPL (>= 2)

**URL** http://www.kasparrufibach.ch

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2021-10-12 16:10:02 UTC

## R topics documented:

---

`reporttools-package` *Generate LaTeX Tables of Descriptive Statistics*

---

## Description

Provides functions to generate tables of descriptive statistics for continuous and nominal variables, as well as some further data manipulation functions. These functions are especially helpful when writing reports of data analysis using Sweave.

## Details

| | |
|---|---|
| Package: | reporttools |
| Type: | Package |
| Version: | 1.1.3 |
| Date: | 2021-10-10 |
| Depends: | xtable, survival |
| License: | GPL (>=2) |

At the beginning of data analysis, it is often useful to have tables of descriptive values for continuous and nominal variables available. This package provides such functions, where the output is a LaTeX table. The functions are most efficiently used when generating reports combining LaTeX with R via Sweave.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

I thank Daniel Sabanes-Bove, Sarah Haile, Philipp Muri, Patrich McCormick, and Sina Rueeger for helpful discussions and remarks.

## References

Rufibach, K. (2009) reporttools: R-Functions to Generate LaTeX Tables of Descriptive Statistics. Journal of Statistical Software, Code Snippets, 31(1).
doi: 10.18637/jss.v031.c01.

---

`addLineBreak` *Break lines in a text column of a dataframe.*

---

## Description

Given a dataframe with a column containing character string, generate a new dataframe where these strings have a maximal length. Useful when embedding dataframes in a Sweave document, without having it overlapping page width.

## Usage

```
addLineBreak(tab, length, col)
```

## Arguments

| | |
|---|---|
| `tab` | Dataframe containing the data. |
| `length` | Maximal length to which strings should be broken. |
| `col` | Column of `tab` that contains strings. |

## Value

List with two elements: The resulting dataframe with lines broken, and a vector that gives row where each entry in the new dataframe ends. The latter is useful when horizontal lines should be added when using xtable.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## Examples

```
tab <- data.frame(cbind(1:4))
tab[1, 2] <- paste(letters, sep = "", collapse = "")
tab[3, 2] <- paste(LETTERS, sep = "", collapse = "")
tab[c(2, 4), 2] <- ""
colnames(tab) <- c("nr", "text")

tab
addLineBreak(tab, length = 12, col = 2)
```

## attachPresAbs *Attach levels absent and present to a 0-1 vector.*

### Description

Attach levels "absent" and "present" to a 0-1 vector and turn it into a factor.

### Usage

```
attachPresAbs(v)
```

### Arguments

| | |
|---|---|
| v | Vector. |

### Value

Factor with the corresponding levels.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
set.seed(1977)
vec <- round(runif(10, 0, 1))
attachPresAbs(vec)
```

## attachYesNo *Attach levels no and yes to a 0-1 vector.*

### Description

Attach levels "no" and "yes" to a 0-1 vector and turn it into a factor.

### Usage

```
attachYesNo(v)
```

### Arguments

| | |
|---|---|
| v | Vector. |

## Value

Factor with the corresponding levels.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## Examples

```
set.seed(1977)
vec <- round(runif(10, 0, 1))
attachYesNo(vec)
```

---

# checkDateSuccession *Check whether dates in two vectors are elementwise consecutive*

---

## Description

Given two vectors $d_1$ and $d_2$ of date type, this function outputs all entries $d_{1j}$ and $d_{2j}$ such that $d_{1j} \geq d_{2j}$.

## Usage

```
checkDateSuccession(d1, d2, pat, names = NA, lab = "", typ = c("R", "tex")[2])
```

## Arguments

| | |
|---|---|
| d1 | Supposedly earlier dates. |
| d2 | Supposedly later dates. |
| pat | Corresponding list of patient (observation) numbers. |
| names | Names of date vectors, of length 3. |
| lab | Label of the generated latex table. |
| typ | Type of output. |

## Value

A latex table is output.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## Examples

```
set.seed(1977)
diagnosis <- as.Date(round(runif(10, min = 35000, max = 40000)),
    origin = "1899-12-30")
death <- as.Date(round(runif(10, min = 35000, max = 40000)),
    origin = "1899-12-30")

## check whether diagnosis was before death
checkDateSuccession(diagnosis, death, 1:10, names = c("Pat",
    "diagnosis", "death"), lab = "tab: diag --> death")

checkDateSuccession(diagnosis, death, 1:10, names = c("Pat",
    "diagnosis", "death"), lab = "tab: diag --> death", typ = "R")
```

---

`colToMat` *Break a n * p data frame in a data frame with ceiling(n / cols) rows and cols * p columns*

---

## Description

Often, one does not want to span a data frame over several pages. This function breaks a $n \times p$ data frame in a data frame with `ceiling(n / cols)` rows and `cols * p` columns.

## Usage

```
colToMat(tab, cols)
```

## Arguments

| | |
|---|---|
| `tab` | The data frame to be reformatted. |
| `cols` | Number of columns of the reformatted data.frame. |

## Value

Returns the reformatted data frame.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

`correctVarNames` *Modify all entries in a data frame such that xtable has no problems displaying them*

## Description

Replace all relevant characters in the entries and row- and colnames of a data frame such that `xtable` does not complain displaying them.

## Usage

```
correctVarNames(tab, rowcol = TRUE, cols = 1:ncol(tab))
```

## Arguments

| | |
|---|---|
| `tab` | The data frame to be formatted. |
| `rowcol` | If `TRUE`, row- and colnames are reformatted. |
| `cols` | Provide a vector of column indices of columns whose entries are to be reformatted. If `NA`, none of the entries of the initial data frame is formatted. |

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

`disp` *Display vectors of numbers, especially targeted to vectors of p-values*

## Description

This function serves to display numbers in plain text, using a given number of digits after the comma.

## Usage

```
disp(n, d1 = 2, d2 = 1)
```

## Arguments

| | |
|---|---|
| `n` | Vector of real numbers to be displayed. |
| `d1` | Number of digits numbers are basically rounded to. |
| `d2` | If numbers in $n$ are smaller than $10^{-\texttt{d1}}$, then `d2` significant digits are given. |

### Value

| | |
|---|---|
| t | A vector of character strings containing the input number n formatted as specified by d1 and d2. |

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
r <- c(0.23445, 0.000089)
disp(r)
```

---

## displayCI *Generate strings of a confidence interval from a matrix*

---

### Description

This function serves to display a confidence interval in plain text, taking a vector of length 2 or a $d \times 2$-matrix containing the confidence limits and given number of digits after the comma. A unit can be additionally supplied.

### Usage

```
displayCI(ci, digit = 2, unit = "", text = "none")
```

### Arguments

| | |
|---|---|
| ci | Vector of length 2 or matrix of size $d \times 2$, containing the confidence limits. |
| digit | Number of digits after the comma. |
| unit | Character string denoting a unit of measurement. |
| text | Specifies the way how the confidence interval should be displayed. |

### Value

A character string to be inserted in plain text.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## Examples

```
a <- 0.05
k <- qnorm(p = 1 - a / 2)
x <- 50
n <- 100
wilson.ci <- (x + k ^ 2 / 2) / (n + k ^ 2) + c(-1, 1) * (k * n ^ 0.5) /
    (n + k ^ 2) * sqrt(x / n * (1 - x / n) + k ^ 2 / (4 * n))
displayCI(wilson.ci)
displayCI(wilson.ci, digit = 1, unit = "cm", text = "none")
displayCI(wilson.ci, digit = 1, unit = "cm", text = "english")
```

---

`displayCoxPH` *Function to display a coxph() object*

---

## Description

Generate a LaTeX table of a coxph object. To be used in a Sweave document.

## Usage

```
displayCoxPH(mod, cap = "", lab = "mod", dig.coef = 2, dig.p = 1)
```

## Arguments

| | |
|---|---|
| `mod` | coxph object. |
| `cap` | The function provides an automatic caption displaying the number of observations and events in `mod`. If `cap != ""` this string is added to the default caption. |
| `lab` | The LaTeX label for the generated table. |
| `dig.coef` | The number of significant digits for the estimated coefficients and the hazard ratios. |
| `dig.p` | The number of significant digits for $p$-values. |

## Value

Returns a LaTeX table containing columns with the estimated coefficients, hazard ratios, 95 percent confidence intervals for the hazard ratios and the $p$-values.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## Examples

```
## Not run:
# use example from coxph() in library 'survival'
test1 <- list(time = c(4, 3, 1, 1, 2, 2, 3),
              status = c(1, 1, 1, 0, 1, 1, 0),
              x = c(0, 2, 1, 1, 1, 0, 0),
              sex = c(0, 0, 0, 0, 1, 1, 1))

# fit a coxph() model
mod1 <- coxph(Surv(time, status) ~ x + sex, data = test1)

# generate table to insert in Sweave file
m1 <- displayCoxPH(mod1)

## End(Not run)
```

---

`displayCrossTabs` *Function to display a set of K x C frequency tables, including p-value*

---

## Description

For each column of a dataframe, generate a LaTeX table against a given variable using `displayKbyC` and add a suitable $p$-value: If the expected frequencies are all $> 5$ then a $\chi^2$-test is computed, otherwise Fisher's exact test.

## Usage

```
displayCrossTabs(vars, v0, nam0, lab0,
    percentage = c("none", "row", "col", "total")[1],
    add.p = TRUE)
```

## Arguments

| | |
|---|---|
| `vars` | Dataframe of nominal variables. |
| `v0` | Nominal variable to tabulate all columns of `vars` against. |
| `nam0` | Name of `v0`. |
| `lab0` | Initial string for table label. The column number of `vars` will be added, so that each table has a unique label. |
| `percentage` | Add percentages with respect to row, column, or table total. |
| `add.p` | Logical. If true, add $p$-value as described above. |

## Value

Displays LaTeX K x C tables and returns a list containing all the information.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
set.seed(1977)
v0 <- round(runif(20, 0, 5))
v1 <- round(runif(20, 0, 3))
v2 <- round(runif(20, 0, 4))
displayCrossTabs(vars = data.frame(v1, v2), v0, nam0 = "v0", lab0 = "Q1")
```

---

## displayKbyC — *Function to display a K x C frequency table including col- and row-names and totals*

---

### Description

Generate a LaTeX table of a $K \times C$ frequency table that contains not only the cell frequencies, but also pre-specified row- and col-names as well as totals of rows and cols.

### Usage

```
displayKbyC(v1, v2, percentage = c("none", "row", "col",
    "total")[1], names = c("v1", "v2"), cap = "",
    lab = "", row.nam = NA, col.nam = NA)
```

### Arguments

| | |
|---|---|
| v1 | Vector with integer entries. |
| v2 | Vector with integer entries. |
| percentage | Add percentages with respect to row, column, or table total. |
| names | Names of the vectors under consideration. |
| cap | Caption of the LaTeX table to be generated. |
| lab | Label of the LaTeX table to be generated. |
| row.nam | Labels of v1 to be given as row names. |
| col.nam | Labels of v2 to be given as column names. |

### Value

Returns a LaTeX K x C table, together with the resulting computations. If you use this function in an .rnw file, you need to assign it to a (dummy) variable name in order for the results beyond the LaTeX table not to appear in the .tex file.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## Examples

```
set.seed(1977)
v1 <- round(runif(20, 0, 3))
v2 <- round(runif(20, 0, 5))
displayKbyC(v1, v2, percentage = "row", names = c("v1", "v2"),
    cap = "", lab = "", row.nam = NA, col.nam = NA)
```

---

# eliminateNA *Eliminate all observations with at least one NA in a data frame*

---

## Description

Generates two matrices: One with complete observations and one with all observations containing at least one missing value.

## Usage

```
eliminateNA(dat)
```

## Arguments

| | |
|---|---|
| dat | Dataframe with observations in rows. |

## Value

| | |
|---|---|
| complete | Dataframe containing complete observations. |
| incomplete | Dataframe containing observations with at least one NA. |

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## See Also

complete.cases

## Examples

```
pat <- 1:10; var1 <- rnorm(10); var2 <- factor(round(rgamma(10, 2, 1)))
dat <- data.frame(cbind(pat, var1, var2))
dat[c(2, 8), 3] <- NA
eliminateNA(dat)
```

## formatPercent *Format a numeric proportion.*

### Description

Takes a number and formats it as a percentage.

### Author(s)

Leo Held
<leonhard.held@ifspm.uzh.ch>

## formatPval *Format P Values*

### Description

`formatPval` is intended for formatting $p$-values, and is based on the function `format.pval` in the base R-package.

### Usage

```
formatPval(pv, digits = max(1, getOption("digits") - 2),
           eps = 0.0001, na.form = "NA", scientific = FALSE,
           includeEquality=FALSE)
```

### Arguments

| | |
|---|---|
| `pv` | a numeric vector. |
| `digits` | how many significant digits are to be used. |
| `eps` | a numerical tolerance: see 'Details'. |
| `na.form` | character representation of `NA`s. |
| `scientific` | use scientific number format (not by default) |
| `includeEquality` | include equality signs in front of the large $p$-values? (not by default) |

### Details

`formatPval` is mainly an auxiliary function for the family of table functions, but can also be useful on its own. If a $p$-value is smaller than `eps`, we return just that it is smaller than the threshold but no longer the exact value. This function is more general than `format.pval` the behaviour of which can (almost) be obtained by using the options `eps = .Machine$double.eps` and `scientific = TRUE`.

## Value

A character vector.

## Examples

```
## include equality signs?
formatPval(c(stats::runif(5), pi^-100, NA))
formatPval(c(stats::runif(5), pi^-100, NA), include=TRUE)

## try another eps argument
formatPval(c(0.1, 0.0001, 1e-7))
formatPval(c(0.1, 0.0001, 1e-7), eps=1e-7)

## only the white space can differ with the base function result:
(a <- formatPval(c(0.1, 0.0001, 1e-27),
                 eps = .Machine$double.eps, scientific = TRUE))
(b <- format.pval(c(0.1, 0.0001, 1e-27)))
all.equal(a, b)
```

---

# getFonts *Used by the tabulating functions to format column titles*

---

## Description

Used by the tabulating functions to format column titles.

## Usage

```
getFonts(font)
```

## Arguments

| | |
|---|---|
| font | Provide font type. |

## Value

Returns function to format column titles.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## math *Enclose a string in math dollars*

### Description

Enclose a string in math dollars.

### Usage

```
math(x)
```

### Arguments

| | |
|---|---|
| x | Character string. |

### Value

Returns x as a string within math dollars.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## NAtoCategory *Change NAs in a factor into a category*

### Description

Extract all the missing values in a factor variable and turn them into a separate category.

### Usage

```
NAtoCategory(fact, label = "missing")
```

### Arguments

| | |
|---|---|
| fact | Factor variable. |
| label | Label to be given to the missing valus. |

### Value

Updated factor variable.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
set.seed(1977)
fact <- factor(sample(c(round(runif(10, 1, 3)), rep(NA, 10))), levels = 1:3,
    labels = c("no", "maybe", "yes"))
NAtoCategory(fact)
```

---

## NAtoZero *Change NAs in a vector into a given value.*

---

### Description

Extract all the missing values in a vector and turn them into a given value.

### Usage

```
NAtoZero(v, value = 0)
```

### Arguments

| | |
|---|---|
| v | Vector. |
| value | Value to be given to the missing valus. |

### Value

Updated vector.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
set.seed(1977)
vec <- sample(c(round(runif(10, 1, 3)), rep(NA, 10)))
NAtoZero(vec)
```

## nominalTest *Compute Chi square or Fisher's exact test*

### Description

Depending on the value of the smallest expected count compute either a $\chi^2$ or Fisher's exact test.

### Usage

```
nominalTest(tab, limit.exp = 5)
```

### Arguments

| | |
|---|---|
| tab | Frequency table, received by applying `table()` to two nominal variables. |
| limit.exp | If the smallest expected count is at most `limit.exp` the $p$-value of a Fisher test is returned. Otherwise, a $\chi^2$ test is computed. |

### Value

A list containing:

| | |
|---|---|
| p | The computed $p$-value. |
| test | A string indicating the test that was used. |

### Examples

```
v1 <- as.factor(round(runif(40, 0, 3)))
v2 <- as.factor(round(runif(40, 2, 3)))
tab <- table(v1, v2)
nominalTest(tab)
```

## pairwise.fisher.test *Pairwise Fisher's exact test*

### Description

Similar to `pairwise.wilcox.test` and `pairwise.t.test`, calculate pairwise comparisons of a nominal variable between group levels with corrections for multiple testing.

### Usage

```
pairwise.fisher.test(x, g, p.adjust.method, ...)
```

### Arguments

| | |
|---|---|
| x | Response vector, nominal (or ordinal). |
| g | Grouping vector or factor. |
| p.adjust.method | Method for adjusting $p$-values (see p.adjust). |
| ... | Additional arguments to pass to fisher.test. |

### Value

Object of class "pairwise.htest"

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### See Also

fisher.test, p.adjust, pairwise.wilcox.test, pairwise.t.test

### Examples

```
set.seed(1977)
x <- factor(abs(round(rnorm(99, 0, 1))))
g <- factor(round(runif(99, 0, 2)))
pairwise.fisher.test(x, g, p.adjust.method = "holm")
```

---

## tableContinuous *Generate a LaTeX table of descriptive statistics for continuous variables*

---

### Description

Many data analyses start with a display of descriptive statistics of important variables. This function takes a data frame of continuous variables and possible grouping (such as e.g. treatment), weighting, and subset variables and provides a LaTeX table of descriptive statistics separately per group and jointly for all observations, per variable. User-defined statistics can be provided.

### Usage

```
tableContinuous(vars, weights = NA, subset = NA, group = NA,
     stats = c("n", "min", "q1", "median", "mean", "q3", "max",
     "s", "iqr", "na"), prec = 1, col.tit = NA,
     col.tit.font = c("bf", "", "sf", "it", "rm"), print.pval =
     c("none", "anova", "kruskal"), pval.bound = 10^-4,
     declare.zero = 10^-10, cap = "", lab = "",
     font.size = "footnotesize", longtable = TRUE,
     disp.cols = NA, nams = NA, ...)
```

## Arguments

| | |
|---|---|
| vars | A data frame containing continuous variables. See nams for an alternative way of specifying the variables to be displayed. |
| weights | Optional vector of weights of each observation. |
| subset | Optional logical vector, indicates subset of observations to be used. |
| group | Optional grouping variable. |
| stats | Specify which descriptive statistics should be displayed in the table, by either directly providing one or more of the default character strings (in arbitrary order) or a user-defined function. A user-defined function must bear a name, take a vector as an argument (NA's are removed by default) and return a single number (the desired statistic). For details see the examples below. |
| prec | Specify number of decimals to be displayed. |
| col.tit | Specify titles of columns. Note that the length of this vector must be equal to the length of stats plus the number of potential user-defined functions added to stats. |
| col.tit.font | If col.tit has not been specified, choose the font for the column titles here (default: no special font face). |
| print.pval | If print.pval == "anova", $p$-values for an analysis of variance for a location difference between groups are added to the table. If print.pval == "kruskal", $p$-values of a Kruskal-Wallis test are given. If group has only two levels, the respective $p$-values of a $t$- or Mann-Whitney test are provided. Only applies if group is provided. Note that by default, any missing values are removed for computation of $p$-values. If missings should be considered a separate level, define the input variables accordingly. |
| pval.bound | $p$-values below pval.bound are formatted as < pval.bound. |
| declare.zero | Computed descriptive statistics (not $p$-values) below that constant are set to 0. Yields nicer tables, especially when displaying centered or standardized variables. |
| cap | The caption of the resulting LaTeX table. |
| lab | The label of the resulting LaTeX table. |
| font.size | Font size for the generated table in LaTeX. |
| longtable | If TRUE, function makes use of package longtable in LaTex to generate tables that span more than one page. If FALSE, generates a table in tabular environment. |
| disp.cols | Only included for backward compatibility. Needs to be a vector built of (some of) the default statistics character strings if not equal to NA. From package version 1.0.2 on use of stats is recommended. |
| nams | A vector of strings, containing the names corresponding to the variables in vars, if vars is not a data frame but a list of variables. These are then the names that appear in the LaTeX table. This option is only kept for backward compatibility. |
| ... | Arguments pass through to print.xtable. |

## Value

Outputs the LaTeX table.

## Warning

If either one of the arguments `group`, `weights`, or `subset` is different from `NA` and if `vars` is a list, then it is assumed that all variables in `vars` are of *equal length*.

## Note

If `longtable = TRUE` (which is the default), the function generates a table that may be more than one page long, you need to include the package longtable in the LaTeX source.

If a list of variables is given to `vars`, not all of these variables need to be of the same length. However, note the Warning above.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## References

Rufibach, K. (2009) reporttools: R-Functions to Generate LaTeX Tables of Descriptive Statistics. Journal of Statistical Software, Code Snippets, 31(1).
doi: 10.18637/jss.v031.c01.

## Examples

```
data(CO2)
vars <- CO2[, 4:5]
group <- CO2[, "Treatment"]
weights <- c(rep(1, 60), rep(0, 10), rep(2, 14))

## display default statistics, provide neither group nor weights
tableContinuous(vars = vars, stats = c("n", "min", "mean", "median",
    "max", "iqr", "na"), print.pval = "kruskal",
    cap = "Table of continuous variables.", lab = "tab: descr stat")

## display default statistics, only use a subset of observations, grouped analysis
tableContinuous(vars = vars, weights = weights, subset =
    c(rep(TRUE, 57), rep(FALSE, 100 - 57)), group = group, prec = 3, print.pval =
    "kruskal", cap = "Table of continuous variables.", lab = "tab: descr stat")

## supply user-defined statistics: trimmed mean and IQR as an unbiased estimate
## of the population standard deviation in case of normal data
my.stats <- list("n", "na", "mean", "$\\bar{x}_{trim}$" = function(x){return(mean(x,
    trim = .05))}, "iqr", "IQR.unbiased" = function(x){return(IQR(x) /
    (2 * qnorm(3 / 4)))})
tableContinuous(vars = vars, weights = weights, group = group, stats = my.stats,
    prec = 3, print.pval = "none", cap = "Table of continuous variables.",
    lab = "tab: descr stat")
```

```
## disp.cols and nams can still be used, for backward compatibility.
## If a list is given to vars, the variables can be of different length. However,
## then weights, subset, and group must be set to NA (the default).
tableContinuous(vars = list(CO2$conc, CO2$uptake, rnorm(1111), runif(2222)),
    nams = c("conc", "uptake", "random1", "random2"), disp.cols =
   c("n", "min", "median", "max", "iqr", "na"), cap = "Table of continuous variables.", lab =
    "tab: descr stat")
```

---

`tableDate` *Display descriptive statistics for date variables*

---

## Description

Many data analyses start with a display of descriptive statistics of important variables. This function takes a data frame of date variables and possible grouping (such as e.g. treatment), weighting, and subset variables and provides a LaTeX table of descriptive statistics separately per group and jointly for all observations, per variable.

## Usage

```
tableDate(vars, weights = NA, subset = NA, group = NA,
     stats = c("n", "min", "q1", "median", "mean", "q3", "max", "na"),
     col.tit = NA, col.tit.font = c("bf", "", "sf", "it", "rm"),
     print.pval = TRUE, pval.bound = 10^-4, cap = "", lab = "",
     font.size = "footnotesize", longtable = TRUE, disp.cols = NA,
     nams = NA, ...)
```

## Arguments

| | |
|---|---|
| `vars` | A data frame of date variables. See `nams` for an alternative way of specifying the variables to be displayed. |
| `weights` | Optional vector of weights of each observation. |
| `subset` | Optional logical vector, indicates subset of observations to be used. |
| `group` | Optional grouping variable. |
| `stats` | Specify which descriptive statistics should be displayed in the table, by either directly providing one or more of the default character strings (in arbitrary order). |
| `col.tit` | Specify titles of columns. |
| `col.tit.font` | If `col.tit` has not been specified, choose the font for the column titles here (default: no special font face). |
| `print.pval` | If `print.pval == TRUE`, $p$-values of a Mann-Whitney or Kruskal-Wallis test for a difference between groups are provided. |
| `pval.bound` | $p$-values below `pval.bound` are formatted as < `pval.bound`. |
| `cap` | The caption of the resulting LaTeX table. |

| | |
|---|---|
| lab | The label of the resulting LaTeX table. |
| font.size | Font size for the generated table in LaTeX. |
| longtable | If TRUE, function makes use of package longtable in LaTex to generate tables that span more than one page. If FALSE, generates a table in tabular environment. |
| disp.cols | Only included for backward compatibility. Needs to be a vector of (some of) the default statistics character strings if not equal to NA. From package version 1.0.2 use of stats is recommended. |
| nams | A vector of strings, containing the names corresponding to the variables in vars, if vars is not a data frame but a list of variables. These are then the names that appear in the LaTeX table. This option is only kept for backward compatibility. |
| ... | Arguments pass through to print.xtable. |

## Value

Outputs the LaTeX table.

## Warning

If either one of the arguments group, weights, or subset is different from NA and if vars is a list, then it is assumed that all variables in vars are of *equal length*.

## Note

If longtable = TRUE (which is the default), the function generates a table that may be more than one page long, you need to include the package longtable in the LaTeX source.

If a list of variables is given to vars, not all of these variables need to be of the same length. However, note the Warning below.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## References

Rufibach, K. (2009) reporttools: R-Functions to Generate LaTeX Tables of Descriptive Statistics. Journal of Statistical Software, Code Snippets, 31(1).
doi: 10.18637/jss.v031.c01.

## Examples

```
set.seed(1977)
diagnosis <- as.Date(round(runif(10, min = 35000, max = 40000)),
    origin = "1899-12-30")
death <- as.Date(round(runif(10, min = 35000, max = 40000)),
    origin = "1899-12-30")
vars <- data.frame(diagnosis, death)
```

```
group <- sample(c(rep("A", 5), rep("B", 5)))
tableDate(vars = vars, group = group, stats = c("n", "min", "median", "max", "na"),
    cap = "Table of date variables.", lab = "tab: descr stat date")

## suppose we have weighted observations
weights <- c(2, 3, 1, 4, rep(1, 6))
subset <- 1:5
tableDate(vars = vars, weights = weights, subset = subset,
    cap = "Table of date variables.", lab = "tab: descr stat date")

## For backward compatibility, disp.cols and nams are still working.
## If a list is given to vars, the variables can be of different length.
## However, then weights, subset, and group must be set to NA (the default).
tableDate(vars = list(diagnosis, death), nams = c("Diagnosis", "Death"),
    disp.cols = c("n", "na", "min", "max"), print.pval = FALSE, cap =
    "Table of date variables.", lab = "tab: descr stat date")
```

---

`tableNominal` *Display descriptive statistics for nominal variables*

---

## Description

Many data analyses start with a display of descriptive statistics of important variables. This function takes a data frame of nominal variables and possible grouping (such as e.g. treatment), weighting, and subset variables and provides a LaTeX table of descriptive statistics separately per group and jointly for all observations, per variable.

## Usage

```
tableNominal(vars, weights = NA, subset = NA,
    group = NA, miss.cat = NA, print.pval = c("none", "fisher",
    "chi2"), pval.bound = 10^-4, fisher.B = 2000, vertical = TRUE,
    cap = "", lab = "", col.tit.font = c("bf", "", "sf", "it", "rm"),
    font.size = "footnotesize", longtable = TRUE, nams = NA,
    cumsum = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `vars` | A data frame of nominal variables. See `nams` for an alternative way of specifying the variables to be displayed. |
| `weights` | Optional vector of weights of each observation. |
| `subset` | Optional logical vector, indicates subset of observations to be used. |
| `group` | Optional grouping variable. |
| `miss.cat` | Vector specifying the factors in `vars` that should have their `NA`s transformed to a separate category. |

| | |
|---|---|
| print.pval | Add $p$-values of Fisher's exact or $\chi^2$ test for a difference of distributions between groups to the table, if there is more than one group. Note that by default, any missing values are removed for computation of $p$-values. If missings should be considered a separate level, define the input variables accordingly. |
| pval.bound | $p$-values below pval.bound are formatted as < pval.bound. |
| fisher.B | Number of simulations to compute $p$-value for Fisher's exact test. Note that in the function fisher.test the option simulate.p.value is set to TRUE, unless fisher.B == Inf which asks for the exact computation. |
| vertical | If TRUE, add vertical lines to the table, separating labels and groups, if applicable. |
| cap | The caption of the resulting LaTeX table. |
| lab | The label of the resulting LaTeX table. |
| col.tit.font | Choose the font for the column titles here (default: boldface). |
| font.size | Font size for the generated table in LaTeX. |
| longtable | If TRUE, function makes use of package longtable in LaTex to generate tables that span more than one page. If FALSE, generates a table in tabular environment. |
| nams | A vector of strings, containing the names corresponding to the variables in vars, if vars is not a data frame but a list of variables. These are then the names that appear in the LaTeX table. This option is only kept for backward compatibility. |
| cumsum | If TRUE, the cumulative sums of the percentages are included for every level of the grouping variable. |
| ... | Arguments pass through to print.xtable. |

## Value

Outputs the LaTeX table.

## Warning

If either one of the arguments group, weights, or subset is different from NA and if vars is a list, then it is assumed that all variables in vars are of *equal length*.

## Note

If longtable = TRUE (which is the default), the function generates a table that may be more than one page long, you need to include the package longtable in the LaTeX source.

If a list of variables is given to vars, not all of these variables need to be of the same length. However, note the Warning above.

## Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

## References

Rufibach, K. (2009) reporttools: R-Functions to Generate LaTeX Tables of Descriptive Statistics. Journal of Statistical Software, Code Snippets, 31(1).
doi: 10.18637/jss.v031.c01.

## Examples

```
data(CO2)
vars <- CO2[, 1:2]
group <- CO2[, "Treatment"]
weights <- c(rep(1, 60), rep(0, 10), rep(2, 14))

## first all observations
tableNominal(vars = vars, weights = weights, group = group, cap =
    "Table of nominal variables.", lab = "tab: nominal")

## do not include cumulative percentages
tableNominal(vars = vars, weights = weights, group = group, cap =
    "Table of nominal variables.", lab = "tab: nominal", cumsum = FALSE)

## but include p-value for Fisher's exact test
tableNominal(vars = vars, weights = weights, group = group, cap =
    "Table of nominal variables.", lab = "tab: nominal",
    print.pval = "fisher", cumsum = FALSE)

## Fisher's exact test without simulated p-value
tableNominal(vars = vars, weights = weights, group = group, cap =
    "Table of nominal variables.", lab = "tab: nominal",
    print.pval = "fisher", fisher.B = Inf, cumsum = FALSE)

## then only consider a subset of observations
subset <- c(1:50, 60:70)
tableNominal(vars = vars, weights = weights, subset = subset, group = group,
    cap = "Table of nominal variables.", lab = "tab: nominal")

## do not include cumulative percentages
tableNominal(vars = vars, weights = weights, subset = subset, group = group,
    cap = "Table of nominal variables.", lab = "tab: nominal", cumsum = FALSE)

## Not run:
## caption placement at the top and repeat column headings on top of each page
## in the longtable format. Have to manually add another backslash to hline and endhead
## below (they are removed when compiling the help file)!
tableNominal(vars = vars, cap = "Table of nominal variables.", cumsum = FALSE,
caption.placement = "top", longtable = TRUE, add.to.row = list(pos = list(0),
command = "\hline \endhead ")

## End(Not run)
```

## transformVarNames — *Generate R-code assigning each variable in a data frame to its name*

### Description

This function generates a one-column matrix, containing strings of assignments of the variables in a data frame.

### Usage

```
transformVarNames(dat, name)
```

### Arguments

| | |
|---|---|
| dat | Dataframe. |
| name | Name of data frame. |

### Value

One-column matrix of strings containing the assignments.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
labpar1 <- rnorm(50)
labor.param2 <- rgamma(50, 2, 1)
dat <- data.frame(labpar1, labor.param2)
transformVarNames(dat, name = "dat")
```

## transformVarNames2 — *Generate R-code assigning each variable in a data frame to its name*

### Description

This function generates a one-column matrix, containing strings of assignments of the variables in a data frame, to be used with `with` in **plyr**, e.g.

### Usage

```
transformVarNames2(nams)
```

### Arguments

| | |
|---|---|
| nams | Variable names, typically colnames applied to a data.frame. |

### Value

One-column matrix of strings containing the assignments.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
labpar1 <- rnorm(50)
labor.param2 <- rgamma(50, 2, 1)
dat <- data.frame(labpar1, labor.param2)
transformVarNames2(colnames(dat))
```

---

## twoGroupComparisons *Compute a table with analysis of two groups comparisons*

---

### Description

For each column of a dataframe, generate a row in a resulting table that contains basic descriptive statistics, effect size, $p$-value, and confidence intervals for a two group comparions, where the grouping variable is separately given.

### Usage

```
twoGroupComparisons(vars, v0, conf.level = 0.95, paired = FALSE)
```

### Arguments

| | |
|---|---|
| vars | Dataframe of continuous variables. |
| v0 | Binary variable that builds the two groups. |
| conf.level | Confidence level used in computation of confidence intervals. |
| paired | Logical, indicate whether comparisons are paired or not. |

### Value

A list consisting of the following elements:

| | |
|---|---|
| raw | Matrix that contains the above as raw numbers. |
| formatted | The same table where numbers are formatted and confidence intervals are given as character string. |

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
set.seed(1977)
v0 <- round(runif(200, 0, 1))
v1 <- rnorm(200)
v2 <- rgamma(200, 2, 1)
twoGroupComparisons(vars = data.frame(v1, v2), v0)
```

---

## varNamesToChar *Split a character string into variable names*

---

### Description

Transform a given string of variable names, separated by ", ", into a vector of corresponding variable names.

### Usage

```
varNamesToChar(varnam)
```

### Arguments

| | |
|---|---|
| varnam | Character string, where variable names are separated by commas. |

### Value

Vector of variable names.

### Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

### Examples

```
nams <- "var1, var2, var3"
varNamesToChar(nams)
```

# Index