Package: descsuppRplots (via r-universe)

October 11, 2026

Title Generate Plots for All Variables in Descriptive Tables

Version 1.0

Description Visualizes variables from descriptive tables produced by 'descsuppR::buildDescrTbl()' using 'ggstatsplot'. It automatically maps each variable to a suitable 'ggstatsplot' plotting function based on the applied or suggested statistical test. Users can override the automatic mapping via a named list of plot specifications. The package supports grouped and ungrouped tables, and forwards additional arguments to the underlying 'ggstatsplot' functions, providing quick, reproducible, and customizable default visualizations for descriptive summaries.

Depends descsuppR (>= 1.1)

Imports dplyr, descutils, rlang, tibble, ggplot2, ggstatsplot (>= 0.12.0), ggsignif, zoo

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Andreas Leha [aut], Fabian Kück [aut, cre]

Maintainer Fabian Kück <fabian.kueck@med.uni-goettingen.de>

Config/pak/sysreqs cmake libgmp3-dev make libicu-dev libuv1-dev libmpfr-dev libssl-dev pari-gp libx11-dev zlib1g-dev

Repository https://fkueck.r-universe.dev

Date/Publication 2025-10-16 11:40:13 UTC

RemoteUrl https://github.com/cran/descsuppRplots

RemoteRef HEAD

RemoteSha 06eda4e5dc21791d2112e62fdd3fc1e934fa5e91

Contents

get_plotfun_defaults 2
plotDescTbl 3
set_plotfun_defaults 4
suggestPlot 5
w.emptyplot 6
w.ggbarstats 7
w.ggbetweenstats 8
w.ggscatterstats 10

Index **12**

`get_plotfun_defaults` *Report default plot functions*

Description

Report default plot functions

Usage

```
get_plotfun_defaults()
```

Value

`tibble` with the currently set default suggestions for plotting function

Author(s)

Dr. Andreas Leha

Examples

```
get_plotfun_defaults()
```

plotDescTbl *Variable Visualizations for Descriptive Tables*

Description

Using as underlying functions the plot functions from ggstatsplot this package visualizes the variables in a descriptive table generated via buildDescrTbl.

Usage

```
plotDescTbl(
  dtab,
  plotfuns = NULL,
  verbose = 0,
  pairwise.display = "none",
  pval.display = FALSE,
  ...
)
```

Arguments

dtab	data.frame with attributes descrtbl_full, data, and groupby as generated by buildDescrTbl
plotfuns	named list of tibbles. names must be in dtab$parameter. entries must be of the form tibble(plotfun = "betweenstats", args = list(list(messages = FALSE, type = "nonparametric"))). For missing entries plotDescTbl tries to guess a suitable plot function. Defaults to NULL meaning that the plot functions for all parameters are guessed.
verbose	logical or integer. Level of verbosity. If TRUE or > 0, print some messages. Defaults to 0.
pairwise.display	Decides if and which pairwise comparisons are shown in the plots. Available options are: - ‘"none"‘ - ‘"significant"‘ (abbreviation accepted: ‘"s"‘) - ‘"non-significant"‘ (abbreviation accepted: ‘"ns"‘) - ‘"all"‘ Defaults to "none".
pval.display	logical. If TRUE p-values are shown in the plots. If dtab contains adjusted p-values, these are shown in the plots. Defaults to FALSE.
...	additional parameters will be passed to the plotting functions.

Value

tibble with column p containing the plots.

Author(s)

Dr. Andreas Leha

Examples

```
ttt <- data.frame(data="training set",
                  age=runif(100, 0, 100),
                 sex=as.factor(sample(c("m","f"), 100, replace=TRUE, prob=c(0.3, 0.7))),
                  score=factor(sample(1:5, 100, replace=TRUE),
                    ordered=TRUE,
                    levels=1:5))
ttt2 <- data.frame(data="test set",
                   age=runif(100, 0, 100),
                  sex=as.factor(sample(c("m","f"), 100, replace=TRUE, prob=c(0.5,0.5))),
                   score=factor(sample(1:5, 100, replace=TRUE),
                     ordered=TRUE,
                     levels=1:5))

## ungrouped table and plots
dtab <- buildDescrTbl(dplyr::mutate(rbind(ttt, ttt2), data = as.factor(data)),
                      includeNAs=TRUE,
                      dopvals = TRUE)

plotDescTbl(dtab)

## grouped table and plots
dtab <- buildDescrTbl(dplyr::mutate(rbind(ttt, ttt2), data = as.factor(data)),
                      groupby="data",
                      includeNAs=TRUE,
                      dopvals = TRUE)

plotDescTbl(dtab)
```

---

set_plotfun_defaults *Change default plot functions*

Description

Change default plot functions

Usage

```
set_plotfun_defaults(df)
```

Arguments

df	`data.frame` with default choices for plotting functions. See `get_plotfun_defaults` for the structure.

Value

invisible. `tibble` with the old (prior to the change) set default suggestions for plotting function

Author(s)

Dr. Andreas Leha

Examples

```
current <- get_plotfun_defaults()
current$plotfun[current$test == "Fisher's Exact Test for Count Data"] <- "emptyplot"
current$args[current$test == "Fisher's Exact Test for Count Data"] <- list(NULL)
set_plotfun_defaults(current)
```

---

suggestPlot *Function suggesting a default plot for a given parameter in a descriptive table*

Description

The plot suggestion is based on the test that was used for that variable.

Usage

```
suggestPlot(ltest)
```

Arguments

`ltest`	character vector of tests. This normally filled by `buildDescrTbl` which in turn usually uses the `method` element of the test result.

Details

The default choice is taken from `plotDescTbl_options$plotfun_defaults` with the current associations.

If no test was performed than numeric variables are treated as if tested via t-tests and non-numeric variables as if tested via Fisher test.

Value

tibble with columns `plotfun` an `args` where `plotfun` is a string with the name of the function to be called to generate the plot and `args` is a list of additional (besides the data) arguments

Author(s)

Dr. Andreas Leha

Examples

```
suggestPlot("Fisher's Exact Test for Count Data")

suggestPlot("foobar")
```

`w.emptyplot` *Wrapper for ggplot2::ggplot()*

Description

Wrapper for ggplot2::ggplot()

Usage

```
w.emptyplot(data, parameter, groupby = NULL, pval.display = FALSE, ldat, ...)
```

Arguments

`data`	A data frame with the data to be plotted. Ignored in this case.
`parameter`	Character. Name of the column in `data` to the values to be plotted. Ignored in this case.
`groupby`	Character (default: NULL). Name of the grouping (categorical) variable in `data`. Ignored in this case.
`pval.display`	Logical. If `TRUE` and there are exactly two groups in `groupby`, a p-value annotation is added. Ignored in this case.
`ldat`	A data frame containing the p-value(s) to annotate on the plot. Expected columns: ‘"p value"‘ and/or ‘"adjusted p value"‘. Ignored in this case.
`...`	Additional arguments Ignored in this case.

Value

A `ggplot` object.

Author(s)

Dr. Andreas Leha

Examples

```
w.emptyplot()
```

`w.ggbarstats` *Wrapper for ggstatsplot::ggbarstats with optional p-value annotation*

Description

This function creates a ‘ggbarstats‘ plot from ‘ggstatsplot‘, and optionally adds a p-value annotation using ‘geom_signif‘ for two-group comparisons based on values in ‘ldat‘.

Usage

```
w.ggbarstats(
  data,
  parameter,
  groupby = NULL,
  pval.display = FALSE,
  ldat,
  tip_length = 4e-04,
  pval_y_position = 1.05,
  ...
)
```

Arguments

`data`	A data frame with the data to be plotted.
`parameter`	Character. Name of the column in `data` to the values to be plotted.
`groupby`	Character (default: NULL). Name of the grouping (categorical) variable in `data`.
`pval.display`	Logical. If `TRUE` and there are exactly two groups in `x`, a p-value annotation is added.
`ldat`	A data frame containing the p-value(s) to annotate on the plot. Expected columns: ‘"p value"‘ and/or ‘"adjusted p value"‘.
`tip_length`	Numeric. Length of the annotation tips for `geom_signif` (default: 0.0004).
`pval_y_position`	Numeric. Vertical position for the p-value annotation (default: 1.05).
`...`	Additional arguments passed to `ggbarstats`.

Details

If `groupby` is not `NULL` and there are exactly two levels in the grouping variable, and `pval.display = TRUE`, a p-value from `ldat` is added between the two groups using `geom_signif`. If `"adjusted p value"` is present in `ldat`, it is used; otherwise, `"p value"` is used.

Value

A `ggplot` object.

Author(s)

Fabian Kück

Examples

```
## ungrouped plot
p <- w.ggbarstats(mtcars, "gear",
                  bf.message = FALSE,
                  proportion.test = FALSE)
if (FALSE) print(p)

## grouped plot
p <- w.ggbarstats(mtcars, "gear", groupby = "cyl",
                  bf.message = FALSE,
                  proportion.test = FALSE)
if (FALSE) print(p)

## grouped plot with p value annotation for >2 groups
pval <- fisher.test(mtcars$gear, mtcars$cyl)$p.value
pval <- descutils::prettyPvalues(pval, digits = 3, orgbold = FALSE, lhs = "p")
ldat <- data.frame(`p value` = pval, check.names = FALSE)
p <- w.ggbarstats(mtcars, "gear", groupby = "cyl", pval.display = TRUE, ldat = ldat,
                  bf.message = FALSE,
                  proportion.test = FALSE)
if (FALSE) print(p)

## grouped plot with p value annotation for two groups
pval <- fisher.test(mtcars$gear, mtcars$vs)$p.value
pval <- descutils::prettyPvalues(pval, digits = 3, orgbold = FALSE, lhs = "p")
ldat <- data.frame(`p value` = pval, check.names = FALSE)
p <- w.ggbarstats(mtcars, "gear", groupby = "vs", pval.display = TRUE, ldat = ldat,
                  bf.message = FALSE,
                  proportion.test = FALSE)
if (FALSE) print(p)
```

`w.ggbetweenstats` *Wrapper for ggstatsplot::ggbetweenstats with optional p-value annotation and date formatting*

Description

This function creates a 'ggbetweenstats' plot from 'ggstatsplot', and optionally adds a p-value annotation using 'geom_signif' for two-group comparisons based on values in 'ldat'. If requested, it can also format the y-axis as dates.

Usage

```
w.ggbetweenstats(
  data,
```

```
  parameter,
  groupby = NULL,
  pval.display = FALSE,
  ldat,
  ...
)
```

Arguments

data	A data frame with the data to be plotted.
parameter	Character. Name of the column in data to the values to be plotted.
groupby	Character (default: NULL). Name of the grouping (categorical) variable in data.
pval.display	Logical. If TRUE and there are exactly two groups in groupby, a p-value annotation is added.
ldat	A data frame containing the p-value(s) to annotate on the plot. Expected columns: ‘"p value"‘ and/or ‘"adjusted p value"‘.
...	Additional arguments passed to ggbetweenstats.

Details

If groupby is not NULL and there are exactly two levels in the grouping variable, and pval.display = TRUE, a p-value from ldat is added between the two groups using geom_signif. If "adjusted p value" is present in ldat, it is used; otherwise, "p value" is used. If wasDate is TRUE, the y-axis labels are formatted as dates.

Value

A ggplot object.

Author(s)

Fabian Kück

Examples

```
## ungrouped plot
p <- w.ggbetweenstats(iris, "Sepal.Length")
if (FALSE) print(p)

## grouped plot
p <- w.ggbetweenstats(iris, "Sepal.Length", groupby = "Species",
                      bf.message = FALSE, pairwise.display = "none")
if (FALSE) print(p)

## grouped plot with p value annotation for >2 groups
pval <- anova(lm(Sepal.Length ~ Species, data = iris))['Species', 'Pr(>F)']
pval <- descutils::prettyPvalues(pval, digits = 3, orgbold = FALSE, lhs = "p")
ldat <- data.frame(`p value` = pval, check.names = FALSE)
p <- w.ggbetweenstats(iris, "Sepal.Length", groupby = "Species", pval.display = TRUE, ldat = ldat,
```

```
                      bf.message = FALSE, pairwise.display = "none")
if (FALSE) print(p)

## setup simple data with only two groups
iris2 <- iris[iris$Species %in% c("setosa", "virginica"),]

## grouped plot with p value annotation for two groups
pval <- t.test(Sepal.Length ~ Species, data = iris2)$p.value
pval <- descutils::prettyPvalues(pval, digits = 3, orgbold = FALSE, lhs = "p")
ldat <- data.frame(`p value` = pval, check.names = FALSE)
p <- w.ggbetweenstats(iris2, "Sepal.Length", groupby = "Species", pval.display = TRUE, ldat = ldat,
                      bf.message = FALSE, pairwise.display = "none")
if (FALSE) print(p)
```

`w.ggscatterstats` *Wrapper for ggstatsplot::ggscatterstats with optional p-value annotation*

Description

This function creates a ‘ggscatterstats‘ plot from ‘ggstatsplot‘, and optionally adds a p-value annotation based on values in ‘ldat‘. It can also format the y-axis as dates.

Usage

```
w.ggscatterstats(data, parameter, groupby, pval.display = FALSE, ldat, ...)
```

Arguments

`data`	A data frame with the data to be plotted.
`parameter`	Character. Name of the column in `data` to the values to be plotted.
`groupby`	Character. Name of the grouping (categorical) variable in `data`.
`pval.display`	Logical. If `TRUE` and there are exactly two groups in `groupby`, a p-value annotation is added.
`ldat`	A data frame containing the p-value(s) to annotate on the plot. Expected columns: ‘"p value"‘ and/or ‘"adjusted p value"‘.
`...`	Additional arguments passed to `ggscatterstats`.

Details

If `groupby` is not `NULL` and there are exactly two levels in the grouping variable, and `pval.display = TRUE`, a p-value from `ldat` is added between the two groups using `geom_signif`. If `"adjusted p value"` is present in `ldat`, it is used; otherwise, `"p value"` is used. If `wasDate` is `TRUE`, the y-axis labels are formatted as dates.

Value

A `ggplot` object.

Author(s)

Fabian Kück

Examples

```
## plot
p <- w.ggscatterstats(mtcars, parameter = "cyl", groupby = "carb",
                      bf.message = FALSE,)
if (FALSE) print(p)

## plot with p value annotation
pval <- cor.test(~ cyl + carb, data = mtcars, method = "kendall")$p.value
pval <- descutils::prettyPvalues(pval, digits = 3, orgbold = FALSE, lhs = "p")
ldat <- data.frame(`p value` = pval, check.names = FALSE)
p <- w.ggscatterstats(mtcars, parameter = "cyl", groupby = "carb", pval.display = TRUE, ldat = ldat,
                      bf.message = FALSE,)
if (FALSE) print(p)
```

Index

buildDescrTbl, *3*, *5*

get_plotfun_defaults, 2, *4*
ggbarstats, *7*
ggbetweenstats, *9*
ggscatterstats, *10*

plotDescTbl, 3

set_plotfun_defaults, 4
suggestPlot, 5

w.emptyplot, 6
w.ggbarstats, 7
w.ggbetweenstats, 8
w.ggscatterstats, 10