Package 'tadaatoolbox'

December 15, 2017

Type Package

Date 2017-12-15

Title Helpers for Data Analysis and Presentation Focused on Undergrad Psychology

Version 0.15.0

Description Contains functions for the easy display of statistical tests as well as some convenience functions for data cleanup. It is meant to ease existing workflows with packages like 'sjPlot', 'dplyr', and 'ggplot2'. The primary components are the functions prefixed with 'tadaa_', which are built to work in an interactive environment, but also print tidy markdown tables powered by 'pixiedust' for the creation of 'RMarkdown' reports.

License MIT + file LICENSE

LazyData TRUE

Encoding UTF-8

Depends R (>= 3.2.1)

Suggests testthat, knitr, rmarkdown, cowplot

Imports stats, methods, broom, magrittr, dplyr, pwr, pixiedust, car, ggplot2, lazyeval, sjlabelled, sjmisc, haven, ryouready, vcd, nortest, lsr, viridis, DescTools

RoxygenNote 6.0.1

URL https://github.com/tadaadata/tadaatoolbox

BugReports https://github.com/tadaadata/tadaatoolbox/issues

VignetteBuilder knitr

NeedsCompilation no

Author Lukas Burk [aut, cre], Tobias Anton [aut], Daniel Lüdecke [ctb]

Maintainer Lukas Burk <lukas@quantenbrot.de>

Repository CRAN

Date/Publication 2017-12-15 17:56:23 UTC

R topics documented:

confint_t *Confidence Intervals*

Description

Confidence Intervals

Usage

```
confint_t(x, alpha = 0.05, na.rm = TRUE)

confint_norm(x, alpha = 0.05, na.rm = TRUE)
```

Arguments

x	A Numeric vector.
alpha	Alpha, default is 0.05.
na.rm	If TRUE (default), missing values are dropped.

Value

numeric of length one (size of CI in one direction).

Examples

```
set.seed(42)
df <- data.frame(x = runif(100), y = sample(c("A", "B"), 100, TRUE))
confint_t(df$x)
set.seed(42)
df <- data.frame(x = runif(100), y = sample(c("A", "B"), 100, TRUE))
confint_norm(df$x)
```

delete_na *Delete cases with set amount of missing values*

Description

Delete cases with set amount of missing values

Usage

```
delete_na(df, n = ncol(df) - 1)
```

Arguments

df	A data.frame,
n	Number of NAs allowed, defaults to ncol(df) - 1.

Value

A filtered version of the input `data.frame`.

Note

Adapted from http://stackoverflow.com/a/30461945/409362.

Examples

```
## Not run:
df <- data.frame(x = sample(c(1:10, NA), 10),
                 y = sample(c(1:10, NA), 10),
                 z = sample(c(1:10, NA), 10))

df <- delete_na(df, 1)

# Or with magrittr syntax sugar
df %<>% delete_na

## End(Not run)
```

`drop_labels` *Re-label a vector after subsetting*

Description

Re-label a vector after subsetting

Usage

```
drop_labels(x)
```

Arguments

`x`	A vector with now unused labels

Value

Identical vector with appropriate labels

Examples

```
## Not run:
x <- drop_labels(x)

## End(Not run)
```

`effect_size_t` *Simple Effect Size Calculation for t-Tests*

Description

Calculates Cohen's d for two sample comparisons.

Usage

```
effect_size_t(data, response, group, absolute = FALSE, paired = FALSE,
  na.rm = TRUE)
```

Arguments

`data`	A `data.frame`.
`response`	The response variable (dependent).
`group`	The group variable, usually a `factor`.
`absolute`	If set to `TRUE`, the absolute effect size is returned.
`paired`	Whether the effect should be calculated for a paired t-test, default is `FALSE`.
`na.rm`	If `TRUE` (default), missing values are dropped.

Details

The effect size here is Cohen's d as calculated by $d = \frac{m_{diff}}{S_p}$, where $m_{diff} = \bar{x}_1 - \bar{x}_2$ and $S_p = \sqrt{\frac{n_1 - 1 \cdot {s_{x_1}}^2 + n_2 - 1 \cdot {s_{x_2}}^2}{n_1 + n_2 - 2}}$.

For `paired = TRUE`, S_p is substituted by $S_D = S_{x_1 - x_2}$ via `sd(x1 - x2)`.

Value

`numeric` of length 1.

Examples

```
set.seed(42)
df <- data.frame(x = runif(100), group = sample(c("A", "B"), 100, TRUE))
effect_size_t(df, "x", "group")
```

generate_recodes *Convenience functions for interval recodes*

Description

Get recode assigments for even intervals of discrete numeric values compatible with car::recode.

Usage

```
generate_recodes(from, to, width = 5)
```

Arguments

from, to	A numeric value for the beginning and the end of the interval.
width	The width of the interval, e.g. 5 (default) for intervals 0-5.

Value

A character vector of recode assignments compatible with car::recode.

Examples

```
## Not run:
x       <- round(runif(100, 0, 100), 0)
recodes <- generate_recodes(0, 100, 10)

library(car)
recode(x, recodes = recodes)

## End(Not run)
```

interval_labels *Convenience functions for interval recodes*

Description

Get interval labels for even intervals of discrete numeric values compatible with base::cut.

Usage

```
interval_labels(from, to, width = 5)
```

Arguments

from, to	A numeric value for the beginning and the end of the interval.
width	The width of the interval, e.g. 5 (default) for intervals 0-5.

Value

A `character` vector of interval labels compatible with base::cut.

Examples

```
## Not run:
x       <- round(runif(100, 0, 100), 0)
labels  <- interval_labels(0, 100, 10)

cut(x, breaks = seq(0, 100, 10), labels = labels)

## End(Not run)
```

`mean_ci_sem` *Standard Error of the Mean with CI*

Description

Standard Error of the Mean with CI

Usage

```
mean_ci_sem(x, conf.level = 0.95)
```

Arguments

`x`	a numeric vector or R object which is coercible to one
`conf.level`	the confidence level (alpha) of the Interval

Value

a `data.frame` with the mean, SEM and its Confidence Interval

Examples

```
set.seed(42)
iq <- rnorm(100, 100, 15)

mean_ci_sem(iq)
```

mean_ci_t *Get mean and CI for a numeric vector*

Description

Suitable for use within ggplot's ggplot2::stat_summary.

Usage

```
mean_ci_t(x, alpha = 0.05, na.rm = TRUE)
```

Arguments

x	A Numeric vector.
alpha	Alpha, default is 0.05.
na.rm	If TRUE (default), missing values are dropped.

Value

A data.frame with y (mean), ymin and ymax values.

Examples

```
set.seed(42)
df <- data.frame(x = runif(100), y = sample(c("A", "B"), 100, TRUE))
mean_ci_t(df$x)
```

modus *Modus*

Description

Calculate the mode of a numeric vector. German name kept to avoid confusion.

Usage

```
modus(x, as_character = TRUE, reduce = TRUE)
```

Arguments

x	A vector with numeric data.
as_character	Always return a character. TRUE by default, or dplyr::summarize. will be very unpleased.
reduce	Since mode can be of length > 1, this option pastes the result into a single character value.

Value

A `vector` of length 1 of type `numeric` or `character`, depending on input.

Examples

```
## Not run:
x <- c(1, 2, 6, 2, 1, 5, 7, 8, 4, 3, 2, 2, 2)
modus(x)

# Or for nominal data
x <- structure(c(2L, 1L, 2L, 2L, 2L, 1L), .Label = c("Ja", "Nein"), class = "factor")
modus(x)

## End(Not run)
```

`ngo` *The ngo dataset*

Description

Sample data for teaching purposes used by Kähler (2008)

Usage

```
ngo
```

Format

A `data.frame` containing numerical and factor data.

Note

`ryouready::d.ngo` is the source of this data, but with some recoding done.

Source

`ryouready::d.ngo()`

References

Kähler, W.-M. (2008). *Statistische Datenanalyse: Verfahren verstehen und mit SPSS gekonnt einsetzen*. Wiesbaden: Vieweg.

nom_c *Contingency Coefficient C*

Description

Very simple wrapper for vcd::assocstats.

Usage

```
nom_c(x, y = NULL)
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable

Value

numeric value

Examples

```
nom_c(ngo$abschalt, ngo$geschl)
```

nom_chisqu *Simple Chi^2*

Description

This is a very simple wrapper for stats::chisq.test.

Usage

```
nom_chisqu(x, y = NULL, correct = FALSE)
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable
correct	Apply correction, passed to chisq.test.

Value

A numeric value

Note

The warning message in case of low samples size and possibly incorrect approximation is suppressed silently.

Examples

```
nom_chisqu(ngo$abschalt, ngo$geschl)
```

`nom_lambda` *Lambda*

Description

Very simple wrapper for ryouready::nom.lambda.

Usage

```
nom_lambda(x, y = NULL, symmetric = FALSE, reverse = FALSE)
```

Arguments

`x`	Dependent variable. Alternatively a `table`.
`y`	Independent variable
`symmetric`	If `TRUE`, symmetric lambda is returned. Default is `FALSE`.
`reverse`	If `TRUE`, row and column variable are switched.

Value

`numeric` value

Examples

```
nom_lambda(ngo$abschalt, ngo$geschl)
```

nom_phi *Phi coefficient*

Description

Very simple wrapper for vcd::assocstats.

Usage

```
nom_phi(x, y = NULL)
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable

Value

numeric value

Examples

```
nom_phi(ngo$abschalt, ngo$geschl)
```

nom_v *Cramer's V*

Description

Very simple wrapper for vcd::assocstats.

Usage

```
nom_v(x, y = NULL)
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable

Value

numeric value

Examples

```
nom_v(ngo$abschalt, ngo$geschl)
```

ord_gamma *Gamma*

Description

Simple wrapper for ryouready::ord.gamma.

Usage

```
ord_gamma(x, y = NULL)
```

Arguments

x	A table or dependent numeric variable.
y	Empty or independent grouping variable

Value

numeric of length 1.

Examples

```
df <- data.frame(rating = round(runif(50, 1, 5)),
                 group  = sample(c("A", "B", "C"), 50, TRUE))
tbl <- table(df)
ord_gamma(tbl)
```

ord_somers_d *Somers' D*

Description

Very simple wrapper for ryouready::ord.somers.d.

Usage

```
ord_somers_d(x, y = NULL, symmetric = FALSE, reverse = FALSE)
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable
symmetric	If TRUE, symmetric D is returned. Default is FALSE.
reverse	If TRUE, row and column variable are switched.

Value

numeric value

Examples

```
ord_somers_d(ngo$abschalt, ngo$geschl)
```

---

ord_tau *Various Tau Statistics*

Description

A wrapper for the appropriate functions from DescTools to calculate Tau A, B and C.

Usage

```
ord_tau(x, y = NULL, tau = "b", reverse = FALSE)
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable
tau	Which of the Taus to return. Default is "b".
reverse	If TRUE, row and column variable are switched.

Value

numeric value

Examples

```
ord_tau(ngo$urteil, ngo$leistung, tau = "a")
```

pval_string *Easy p-value formatting*

Description

Easy p-value formatting

Usage

```
pval_string(pv)
```

Arguments

pv	A p-value in numeric form.

Value

A formatted `character` representation of the input value.

Note

Simplified version of pixiedust::pvalString which considers `< 0.05`.

Examples

```
pv <- c(.9, .2, .049, .009, .000003)
pval_string(pv)
```

tadaa_aov *Tadaa, ANOVA!*

Description

Performs one-, two-way or factorial ANOVA with adjustable sums of squares method and optionally displays effect sizes ((partial) η^2, Cohen's f) and power (calculated via pwr::pwr.f2.test to work with unbalanced designs).

Usage

```
tadaa_aov(formula, data = NULL, show_effect_size = TRUE,
  show_power = TRUE, factorize = TRUE, type = 3, check_contrasts = TRUE,
  print = c("df", "console", "html", "markdown"))
```

Arguments

formula	Formula for model, passed to aov.
data	Data for model.
show_effect_size	If TRUE (default), effect sizes partial eta^2 and Cohen's f are appended as columns.
show_power	(Experimental) If TRUE (default), power is calculated via pwr::pwr.f2.test and appended as a column.
factorize	If TRUE (default), non-factor independent variables will automatically converted via as.factor, so beware of your inputs.
type	Which type of SS to use. Default is 3, can also be 1 or 2.
check_contrasts	Only applies to type = 3. If TRUE (default), the contrasts of each non-ordered factor are set to "contr.sum".
print	Print method, default df: A regular data.frame. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.

Details

If a specified independent variable is not properly encoded as a factor, it is automatically converted if factorize = TRUE to ensure valid results.

If type = 3 and check_contrasts = TRUE, the "contrasts" of each non-ordered factor will be checked and set to contr.sum to ensure the function yields usable results. It is highly recommended to only use check_contrasts = FALSE for debugging or educational purposes.

Value

A data.frame by default, otherwise dust object, depending on print.

See Also

Other Tadaa-functions: tadaa_chisq, tadaa_kruskal, tadaa_levene, tadaa_nom, tadaa_normtest, tadaa_one_sample, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
tadaa_aov(stunzahl ~ jahrgang, data = ngo)
tadaa_aov(stunzahl ~ jahrgang * geschl, data = ngo)

# Other types of sums and print options
## Not run:
tadaa_aov(stunzahl ~ jahrgang * geschl, data = ngo, type = 1, print = "console")
tadaa_aov(stunzahl ~ jahrgang * geschl, data = ngo, type = 3, print = "console")
tadaa_aov(stunzahl ~ jahrgang * geschl, data = ngo,
          type = 3, check_contrasts = FALSE, print = "console")

## End(Not run)
```

tadaa_balance *Grouping design balance*

Description

Easily generate heatmaps to show how well balanced groups are designed, e.g. for ANOVA.

Usage

```
tadaa_balance(data, group1, group2, palette = "D", annotate = TRUE)
```

Arguments

data	A data.frame
group1	The grouping variable on the x-axis
group2	The grouping variable on the y-axis
palette	The viridis::viridis color palette to use; c("A", "B", "C", "D"), defaults to "D"
annotate	Should the n of each group be displayed in each cell of the heatmap?

Value

A ggplot2 object

See Also

Other Tadaa-plot functions: tadaa_int, tadaa_mean_ci, tadaa_plot_tukey

Examples

```
tadaa_balance(ngo, jahrgang, geschl)
```

tadaa_chisq *Tadaa, Chi-Square Test!*

Description

A comfortable wrapper of stats::chisq.test with pretty output and effect sizes depending on the size of the contingency table: Phi coefficient and Odds Ratios in case of a 2x2 table, Cramer's V otherwise. The result is either returned as a broom::tidy data.frame or prettified using various pixiedust::sprinkle shenanigans.

Usage

```
tadaa_chisq(data, x, y, correct = TRUE, print = c("df", "console", "html",
  "markdown"))
```

Arguments

data	A data.frame.
x	A vector of categorial data (factor or character).
y	Another vector of categorial data (also factor or character).
correct	Apply Yate's continuity correction for 2x2 tables, passed to stats::chisq.test. Defaults to TRUE.
print	Print method, default df: A regular data.frame. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.

Value

A data.frame by default, otherwise dust object, depending on print.

Note

The warning message in case of low samples size and possibly incorrect approximation is suppressed silently.

See Also

Other Tadaa-functions: tadaa_aov, tadaa_kruskal, tadaa_levene, tadaa_nom, tadaa_normtest, tadaa_one_sample, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
tadaa_chisq(ngo, abschalt, geschl)

tadaa_chisq(ngo, abschalt, jahrgang)
```

---

tadaa_int *Interaction plots*

Description

Easily generate interaction plots of two nominal grouping variables and a numeric response variable.

Usage

```
tadaa_int(data, response, group1, group2, grid = FALSE,
  brewer_palette = "Set1", labels = c("A", "B"), show_n = FALSE,
  print = TRUE)
```

Arguments

data	A data.frame.
response	Response variable.
group1	First grouping variable.
group2	Second grouping variable.
grid	If TRUE, the resulting graphs will be arranged in a grid via cowplot::plot_grid.
brewer_palette	The name of the RColorBrewer palette to use, defaults to Set1.
labels	Labels used for the plots when printed in a grid (grid = TRUE), defaults to c("A", "B").
show_n	If TRUE, displays N in plot subtitle.
print	Default is TRUE, set FALSE to suppress automatic printing. Useful if you intend to further modify the outpur plots.

Value

Invisible: A list with two ggplot2 objects named p1 and p2. If print = TRUE: Printed: The one or two ggplot2 objects, depending on grid.

See Also

Other Tadaa-plot functions: tadaa_balance, tadaa_mean_ci, tadaa_plot_tukey

Examples

```
## Not run:
tadaa_int(ngo, stunzahl, jahrgang, geschl)

# As grid, if cowplot is installed
tadaa_int(ngo, stunzahl, jahrgang, geschl, grid = TRUE)

## End(Not run)
```

---

tadaa_kruskal *Tadaa, Kruskal-Wallis!*

Description

Tadaa, Kruskal-Wallis!

Usage

```
tadaa_kruskal(formula, data = NULL, print = c("df", "console", "html",
  "markdown"))
```

Arguments

formula	Formula for model, passed to kruskal.test.
data	Data for model.
print	Print method, per default a regular data.frame. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.

Value

A data.frame by default, otherwise dust object, depending on print.

See Also

Other Tadaa-functions: tadaa_aov, tadaa_chisq, tadaa_levene, tadaa_nom, tadaa_normtest, tadaa_one_sample, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
tadaa_kruskal(stunzahl ~ jahrgang, data = ngo)
```

---

tadaa_levene *Levene's Test for Homoskedasticity*

Description

A thin wrapper around car::leveneTest with some formatting done.

Usage

```
tadaa_levene(data, formula, center = "median", print = c("df", "console",
  "html", "markdown"))
```

Arguments

data	Data for the test
formula	Formula specifiyng groups, passed to leveneTest.
center	Method to use, either median (default for robustness) or mean.
print	Print method, default df: A regular data.frame. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.

Value

A data.frame by default, otherwise dust object, depending on print.

Note

The case of `center = "median"` is technically called *Brown–Forsythe test*, so if that's selected the header for non-`df` returns will reflect that.

See Also

Other Tadaa-functions: `tadaa_aov`, `tadaa_chisq`, `tadaa_kruskal`, `tadaa_nom`, `tadaa_normtest`, `tadaa_one_sample`, `tadaa_ord`, `tadaa_pairwise_gh`, `tadaa_pairwise_tukey`, `tadaa_pairwise_t`, `tadaa_t.test`, `tadaa_wilcoxon`

Examples

```
tadaa_levene(ngo, deutsch ~ jahrgang, print = "console")
tadaa_levene(ngo, deutsch ~ jahrgang * geschl, print = "console")
```

---

tadaa_mean_ci *Means with Errorbars*

Description

Means with Errorbars

Usage

```
tadaa_mean_ci(data, response, group, brewer_palette = "Set1")
```

Arguments

`data`	A `data.frame`
`response`	Response variable, numeric.
`group`	Grouping variable, ideally a `factor`.
`brewer_palette`	Optional: The name of the `RColorBrewer`` palette to use, defaults to`Set1`. Use`NULL` for no brewer palette.

Value

A ggplot2 object.

See Also

Other Tadaa-plot functions: `tadaa_balance`, `tadaa_int`, `tadaa_plot_tukey`

Examples

```
tadaa_mean_ci(ngo, deutsch, jahrgang, brewer_palette = "Set1")
```

tadaa_nom *Get all the nominal stats*

Description

Get all the nominal stats

Usage

```
tadaa_nom(x, y = NULL, round = 2, print = "console")
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable
round	Ho many digits should be rounded. Default is 2.
print	Print method. Passed to pixiedust::sprinkle_print_method as of now.

Value

A dust object, depending on print.

See Also

Other Tadaa-functions: tadaa_aov, tadaa_chisq, tadaa_kruskal, tadaa_levene, tadaa_normtest, tadaa_one_sample, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
tadaa_nom(ngo$abschalt, ngo$geschl)
```

tadaa_normtest *Tadaa, test for normality!*

Description

Tadaa, test for normality!

Usage

```
tadaa_normtest(data, method = "ad", print = c("df", "console", "html",
  "markdown"), ...)
```

Arguments

data	A data.frame.
method	The type of test to perform. Either ad for Anderson Darling, shapiro for Shapiro-Wilk, pearson for Pearson's chi-square test or ks for Kolmogorov-Smirnov (not recommended).
print	Print method, default df: A regular data.frame. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.
...	Further arguments passed to test functions where applicable, see nortest::pearson.test and stats::ks.test.

Value

A data.frame by default, otherwise dust object, depending on print.

See Also

Other Tadaa-functions: tadaa_aov, tadaa_chisq, tadaa_kruskal, tadaa_levene, tadaa_nom, tadaa_one_sample, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
## Not run:
library(dplyr)
ngo %>%
select(englisch, deutsch, mathe) %>%
tadaa_normtest(method = "shapiro")

ngo %>%
select(englisch, deutsch, mathe) %>%
tadaa_normtest(method = "pearson", print = "console")

## End(Not run)
```

---

tadaa_one_sample *Tadaa, one-sample tests!*

Description

If sigma is omitted, the function will just perform a one-sample stats::t.test, but if sigma is provided, a z-test is performed. It basically works the same way, except that we pretend we know the population sigma and use the normal distribution for comparison.

Usage

```
tadaa_one_sample(data = NULL, x, mu, sigma = NULL,
  direction = "two.sided", na.rm = FALSE, conf.level = 0.95,
  print = c("df", "console", "html", "markdown"))
```

Arguments

data	A data.frame (optional).
x	A numeric vector or bare column name of data.
mu	The true mean (μ) to test for.
sigma	Population sigma. If supplied, a z-test is performed, otherwise a one-sample stats::t.test is performed.
direction	Test direction, like alternative in t.test.
na.rm	Whether to drop NA values. Default is FALSE.
conf.level	Confidence level used for power and CI, default is 0.95.
print	Print method, default df: A regular data.frame. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.

Value

A data.frame by default, otherwise dust object, depending on print.

See Also

Other Tadaa-functions: tadaa_aov, tadaa_chisq, tadaa_kruskal, tadaa_levene, tadaa_nom, tadaa_normtest, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
set.seed(42)
df <- data.frame(x = rnorm(n = 20, mean = 100, sd = 1))

tadaa_one_sample(df, x, mu = 101, sigma = 1)

# No data.frame, just a vector
tadaa_one_sample(x = rnorm(20), mu = 0)
```

---

tadaa_ord *Get all the ordinal stats*

Description

As of now, only Gamma and Somers D are supported. But let's be honest: Everybody hates Tau.

Usage

```
tadaa_ord(x, y = NULL, round = 2, print = "console")
```

Arguments

x	Dependent variable. Alternatively a table.
y	Independent variable
round	Ho many digits should be rounded. Default is 2.
print	Print method. Passed to pixiedust::sprinkle_print_method as of now.

Value

A dust object, depending on print.

See Also

Other Tadaa-functions: tadaa_aov, tadaa_chisq, tadaa_kruskal, tadaa_levene, tadaa_nom, tadaa_normtest, tadaa_one_sample, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_pairwise_t, tadaa_t.test, tadaa_wilcoxon

Examples

```
tadaa_ord(ngo$abschalt, ngo$geschl)
```

---

tadaa_pairwise_gh *Games Howell Post-Hoc Test*

Description

An implementation of the Games Howell procedure for pairwise comparisons. The workhorse of this function is adapted from this gist: https://gist.github.com/aschleg/ea7942efc6108aedfa9ec98aeb6c2096

Usage

```
tadaa_pairwise_gh(data, response, group1, group2 = NULL, print = "df")
```

Arguments

data	A data.frame containing the variables.
response	The response variable, i.e. the dependent numeric vector.
group1	The grouping variables, typically a factor.
group2	(Optional) second grouping variable.
print	Print method, defaults to df for data.frame output, otherwise passed to pixiedust::sprinkle_print_method.

Value

A data.frame or pixiedust::dust object depending on print.

Note

This function is really, really slow for large comparisons (`k > 50`). Sorry about that.

Author(s)

`gitub.com/aschleg`, Lukas Burk

Source

https://gist.github.com/aschleg/ea7942efc6108aedfa9ec98aeb6c2096

References

https://rpubs.com/aaronsc32/games-howell-test

See Also

`tadaa_pairwise_t()`, `tadaa_pairwise_tukey()`

Other Tadaa-functions: `tadaa_aov`, `tadaa_chisq`, `tadaa_kruskal`, `tadaa_levene`, `tadaa_nom`, `tadaa_normtest`, `tadaa_one_sample`, `tadaa_ord`, `tadaa_pairwise_tukey`, `tadaa_pairwise_t`, `tadaa_t.test`, `tadaa_wilcoxon`

Examples

```
tadaa_pairwise_gh(ngo, deutsch, jahrgang)
tadaa_pairwise_gh(ngo, deutsch, jahrgang, geschl)
```

---

tadaa_pairwise_t *Extended Pairwise t-Tests*

Description

This is an extension of stats::pairwise.t.test that's meant to deal with interactions our of the box, while also performing pairwise tests for the primary terms. The output of the function is modeled after stats::TukeyHSD, unfortunately without confidence intervals or test statistic though.

Usage

```
tadaa_pairwise_t(data, response, group1, group2 = NULL, p.adjust = "bonf",
  paired = FALSE, pool.sd = !paired, alternative = "two.sided",
  print = "df")
```

Arguments

data	A data.frame containing the variables.
response	The response variable, i.e. the dependent numeric vector.
group1	The grouping variables, typically a factor.
group2	(Optional) second grouping variable.
p.adjust	The p-adjustment method, see stats::p.adjust.methods, passed to stats::pairwise.t.test. Additionally, sidak is supported as a method, which is not the case with stats::p.adjust, as is sidakSD for the Sidak step-down procedure.
paired	Defaults to FALSE, also passed to stats::pairwise.t.test.
pool.sd	Defaults to the inverse of paired, passed to stats::pairwise.t.test.
alternative	Defaults to two.sided, also passed to stats::pairwise.t.test.
print	Print method, defaults to df for data.frame output, otherwise passed to pixiedust::sprinkle_print_method.

Value

A data.frame with columns term, comparison and adj.p.value.

Note

The adjustment method is applied within each term, meaning that the number of pairwise t-tests counted for the adjustment is only equal to the number of rows per term of the output. The additional Sidak adjustment method uses the following method: p_adj <- 1 - pbinom(q = 0, size = length(p_values), pr And is sometimes preferred over Bonferroni. The Sidak-like (1987) step-down procedure (sidakSD) is an improvement over the Holm's (1979) step-down procedure.

References

https://stats.stackexchange.com/questions/20825/sidak-or-bonferroni

https://rdrr.io/rforge/mutoss/man/SidakSD.html

See Also

tadaa_pairwise_tukey(), tadaa_pairwise_gh()

Other Tadaa-functions: tadaa_aov, tadaa_chisq, tadaa_kruskal, tadaa_levene, tadaa_nom, tadaa_normtest, tadaa_one_sample, tadaa_ord, tadaa_pairwise_gh, tadaa_pairwise_tukey, tadaa_t.test, tadaa_wilcoxon

Examples

```
tadaa_pairwise_t(ngo, deutsch, jahrgang, geschl, p.adjust = "none", print = "console")
tadaa_pairwise_t(ngo, deutsch, jahrgang, geschl, p.adjust = "bonf", print = "console")
tadaa_pairwise_t(ngo, deutsch, jahrgang, geschl, p.adjust = "sidak", print = "console")
```

`tadaa_pairwise_tukey` *Tukey HSD pairwise comparisons*

Description

This function is merely a thin wrapper around stats::TukeyHSD with tidying done by broom::tidy and optional formatting via pixiedust::sprinkle. Its input is not a `aov` model like in the original function, but instead the `aov` model is fit internally based on the arguments given. This is meant to enable a consistent usage between the `tadaa_pairwise`-functions.

Usage

```
tadaa_pairwise_tukey(data, response, group1, group2 = NULL, print = "df",
  ...)
```

Arguments

`data`	A `data.frame` containing the variables.
`response`	The response variable, i.e. the dependent numeric vector.
`group1`	The grouping variables, typically a `factor`.
`group2`	(Optional) second grouping variable.
`print`	Print method, defaults to `df` for `data.frame` output, otherwise passed to pixiedust::sprinkle_print_method.
`...`	Further arguments passed to stats::TukeyHSD

Value

A `data.frame` or pixiedust::dust object depending on `print`.

See Also

`tadaa_pairwise_t()`, `tadaa_pairwise_gh()`

Other Tadaa-functions: `tadaa_aov`, `tadaa_chisq`, `tadaa_kruskal`, `tadaa_levene`, `tadaa_nom`, `tadaa_normtest`, `tadaa_one_sample`, `tadaa_ord`, `tadaa_pairwise_gh`, `tadaa_pairwise_t`, `tadaa_t.test`, `tadaa_wilcoxon`

Examples

```
tadaa_pairwise_tukey(data = ngo, deutsch, jahrgang, geschl)
tadaa_pairwise_tukey(data = ngo, deutsch, jahrgang, print = "console")
```

`tadaa_plot_tukey` *Plot TukeyHSD Results as Errorbars*

Description

This is a simple plotting template that takes the broom::tidy'd output of stats::TukeyHSD or alternatively the `print = "df"` output of tadaa_pairwise_tukey and plots it nicely with error bars.

Usage

```
tadaa_plot_tukey(data, brewer_palette = "Set1")
```

Arguments

`data`	The broom::tidy'd output of stats::TukeyHSD.
`brewer_palette`	Optional: The name of the `RColorBrewer`` palette to use, defaults toSet1. UseNULL‘ for no brewer palette.

Value

A ggplot2 object.

Note

The `alpha` of the error bars is set to `0.25` if the comparison is not significant, and `1` otherwise. That's neat.

See Also

Other Tadaa-plot functions: `tadaa_balance`, `tadaa_int`, `tadaa_mean_ci`

Examples

```
tests <- tadaa_pairwise_tukey(data = ngo, deutsch, jahrgang, geschl, print = "df")
tadaa_plot_tukey(tests)
```

`tadaa_t.test` *Tadaa, t-Test!*

Description

An extension for stats::t.test with added boni and tidy and/or pretty output. Before a t-test is performed, car::leveneTest is consulted as to wether heteroskedasticity is present (using the default `center = "mean"` method for a more robust test), and sets `var.equal` accordingly. Afterwards, the effect size is calculated and pwr::pwr.t.test or pwr::pwr.t2n.test are used to calculate the test's power accordingly. The result is either returned as a broom::tidy `data.frame` or prettified using various pixiedust::sprinkle shenanigans.

Usage

```
tadaa_t.test(data, response, group, direction = "two.sided", paired = FALSE,
  var.equal = NULL, conf.level = 0.95, print = c("df", "console", "html",
  "markdown"))
```

Arguments

`data`	A `data.frame`.
`response`	The response variable (dependent).
`group`	The group variable, usually a `factor`.
`direction`	Test direction, like `alternative` in t.test.
`paired`	If `TRUE`, a paired test is performed, defaults to `FALSE`.
`var.equal`	If set, passed to stats::t.test to decide whether to use a Welch-correction. Default is `NULL` to automatically determine heteroskedasticity.
`conf.level`	Confidence level used for power and CI, default is `0.95`.
`print`	Print method, default `df`: A regular `data.frame`. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.

Value

A `data.frame` by default, otherwise `dust` object, depending on `print`.

Note

The cutoff for the interal Levene's test to decided whether or not to perform a Welch-corrected t-test is set to `0.05` by default. To override the internal tests and decide whether to use a Welch test, set `var.equal` as you would with stats::t.test.

See Also

Other Tadaa-functions: `tadaa_aov`, `tadaa_chisq`, `tadaa_kruskal`, `tadaa_levene`, `tadaa_nom`, `tadaa_normtest`, `tadaa_one_sample`, `tadaa_ord`, `tadaa_pairwise_gh`, `tadaa_pairwise_tukey`, `tadaa_pairwise_t`, `tadaa_wilcoxon`

Examples

```
set.seed(42)
df <- data.frame(x = runif(100), y = sample(c("A", "B"), 100, TRUE))
tadaa_t.test(df, x, y)

df <- data.frame(x = runif(100), y = c(rep("A", 50), rep("B", 50)))
tadaa_t.test(df, x, y, paired = TRUE)

tadaa_t.test(ngo, deutsch, geschl, print = "console")
```

`tadaa_wilcoxon` *Tadaa, Wilcoxon!*

Description

Tadaa, Wilcoxon!

Usage

```
tadaa_wilcoxon(data, response, group, direction = "two.sided",
  paired = FALSE, print = c("df", "console", "html", "markdown"), ...)
```

Arguments

`data`	A `data.frame`.
`response`	The response variable (dependent).
`group`	The group variable, usually a `factor`.
`direction`	Test direction, like `alternative` in t.test.
`paired`	If `TRUE`, a paired test is performed, defaults to `FALSE`.
`print`	Print method, default `df`: A regular `data.frame`. Otherwise passed to pixiedust::sprinkle_print_method for fancyness.
`...`	Further arguments passed to stats::wilcox.test, e.g. `correct = FALSE`.

Value

A `data.frame` by default, otherwise `dust` object, depending on `print`.

See Also

Other Tadaa-functions: `tadaa_aov`, `tadaa_chisq`, `tadaa_kruskal`, `tadaa_levene`, `tadaa_nom`, `tadaa_normtest`, `tadaa_one_sample`, `tadaa_ord`, `tadaa_pairwise_gh`, `tadaa_pairwise_tukey`, `tadaa_pairwise_t`, `tadaa_t.test`

Examples

```
set.seed(42)
df <- data.frame(x = runif(100), y = sample(c("A", "B"), 100, TRUE))
tadaa_wilcoxon(df, x, y)

df <- data.frame(x = runif(100), y = c(rep("A", 50), rep("B", 50)))
tadaa_wilcoxon(df, x, y, paired = TRUE)
```

`theme_readthedown` *ggplot2 theme to fit the readthedown Rmd format*

Description

A ggplot theme to fit `rmdformats::readthedown` in terms of background color and dark grid lines.

Usage

```
theme_readthedown(base_size = 12, base_family = "", bg = "#fcfcfc",
  axis_emph = "xy", ...)

theme_tadaa(base_size = 12, base_family = "", bg = "#fcfcfc",
  axis_emph = "xy", ...)
```

Arguments

`base_size`	Base text size, defaults to 12.
`base_family`	Base text family. Use `"Roboto Slab"` to match the `readthedown` headers, or `"Lato"` for the body style.
`bg`	Background color, defaults to `rmdformats::readthedown`'s background, `#fcfcfc`
`axis_emph`	Which axis to emphasize visually (black lines). One of `"x"`, `"y"`, `"xy"`, `NULL`.
`...`	Other arguments passed to `ggplot2::theme()`

Value

A ggplot2 theme

Examples

```
## Not run:
library(ggplot2)
p <- qplot(1:10, 1:10, geom = "point")

p + theme_readthedown()
p + theme_readthedown(base_family = "Lato")
p + theme_readthedown(base_family = "Roboto Slab", axis_emph = "x")

## End(Not run)
```

z *Convert numeric vector to z-values*

Description

A trivial scaling function. You might as well use base::scale, which allows arbitrary centers and scales, but returns a `matrix` by default.

Usage

```
z(x)
```

Arguments

x	A numeric vector.

Value

A vector of z-values of the same length as `x`.

Examples

```
x      <- rnorm(500, mean = 10, sd = 5)
z_vals <- z(x)
round(c(mean = mean(z_vals), sd = sd(z_vals)), 2)
```

Index