Package ‘GimmeMyStats’

January 23, 2026

Title Statistics Utilities

Version 1.0.0

Description Facilitate reporting for regression and correlation modeling, hypothesis testing, variance analysis, outlier detection, and detailed descriptive statistics.

License GPL-3

URL `https://github.com/ecamenen/GimmeMyStats`, `https://ecamenen.github.io/GimmeMyStats/`

BugReports `https://github.com/ecamenen/GimmeMyStats/issues`

Depends magrittr, R (>= 3.8), tidyverse

Imports dplyr, e1071, forcats, lme4, lmerTest, rstatix, stats, stringi, stringr, tidyr, tidyselect, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData false

RoxygenNote 7.3.2

NeedsCompilation no

Author Etienne Camenen [aut, cre]

Maintainer Etienne Camenen <`etienne.camenen@gmail.com`>

Repository CRAN

Date/Publication 2026-01-23 14:10:11 UTC

Contents

`add_significance0` *Add P-value Significance Symbols*

Description

Redefine the default parameters of `rstatix::add_significance()` by adding p-value significance symbols to a data frame.

Usage

```
add_significance0(data, p.col = NULL, output.col = NULL)
```

Arguments

`data`	a data frame containing a p-value column.
`p.col`	column name containing p-values.
`output.col`	the output column name to hold the adjusted p-values.

Value

a data frame

Examples

```
library(magrittr)
library(rstatix, warn.conflicts = FALSE)
data("ToothGrowth")
ToothGrowth %>%
    t_test(len ~ dose) %>%
    adjust_pvalue() %>%
    add_significance0("p.adj")
```

`count_category` *Frequency of categorical variables*

Description

Formats a data frame or vector containing categorical variables and calculates the frequency of each category.

Usage

```
count_category(x, width = 15, collapse = FALSE, sort = TRUE, format = TRUE)
```

Arguments

`x`	Data frame or vector containing categorical variables.
`width`	Integer specifying the maximum width for wrapping text.
`collapse`	Logical specifying whether to merge categories with identical proportions.
`sort`	Logical or character vector. If `TRUE`, orders categories by frequency. If `FALSE`, orders by names. If a character vector, renames and orders categories accordingly.
`format`	Logical specifying whether to format category names if the input is a vector.

Value

A tibble with one row per category and the following columns:

f Factor specifying the category labels, possibly wrapped to the specified width. When `collapse = TRUE`, multiple categories with identical frequencies are merged into a single label separated by commas.

n Integer specifying the frequency count for each category.

Examples

```
# Vector of categorical variable
k <- 5
n <- runif(k, 1, 10) %>% round()
x <- paste("Level", seq(k)) %>%
    mapply(function(x, y) rep(x, y), ., n) %>%
    unlist()
count_category(x)

# Data frame of categorical variable
df <- sapply(seq(k), function(x) runif(10) %>% round()) %>% as.data.frame()
colnames(df) <- paste("Level", seq(k))
count_category(df)
count_category(x, sort = FALSE, width = 5)
count_category(x, sort = seq(k), format = FALSE)
x2 <- c(x, rep("Level 6", n[1]))
count_category(x2, collapse = TRUE)
```

`housetasks` *Household tasks distribution by gender and arrangement*

Description

A dataset containing the distribution of household tasks among different arrangements: Wife, Alternating, Husband, and Jointly. The data represents the frequency of each task performed by each arrangement.

Usage

```
data(housetasks)
```

Format

A `data.frame` with 13 rows (tasks) and 4 columns (arrangements):

Wife Numeric, the frequency of the task performed primarily by the wife.

Alternating Numeric, the frequency of the task performed in an alternating manner.

Husband Numeric, the frequency of the task performed primarily by the husband.

Jointly Numeric, the frequency of the task performed jointly by both partners.

Source

The dataset was downloaded from the ggpubr GitHub repository: `https://raw.githubusercontent.com/kassambara/ggpubr/refs/heads/master/inst/demo-data/housetasks.txt`

Examples

```
data(housetasks)
head(housetasks)
```

`identify_outliers` *Identifies outliers in a numeric vector*

Description

Detects outliers using methods like IQR, percentiles, Hampel, MAD, or SD.

Usage

```
identify_outliers(
  x,
  probabilities = c(0.25, 0.75),
  method = "iqr",
  weight = 1.5,
  replace = FALSE
)
```

Arguments

`x`	Vector containing numerical values.
`probabilities`	Numeric vector specifying probabilities for percentiles.
`method`	Character specifying the method: `iqr`, `percentiles`, `hampel`, `mad`, or `sd`.
`weight`	Double specifying the multiplier for the detection threshold.
`replace`	Logical specifying whether to replace outliers with `NA`.

Value

A numeric vector whose content depends on the value of `replace`:

replace = FALSE A numeric vector containing only the detected outlier values. The vector is named with the original indices or names of `x`.

replace = TRUE A numeric vector of the same length as `x`, where detected outliers are replaced by `NA`.

Examples

```
x <- rnorm(100)
identify_outliers(x, method = "iqr")
identify_outliers(x, method = "percentiles", probabilities = c(0.1, 0.9))
identify_outliers(x, method = "sd", weight = 3)
identify_outliers(x, method = "mad", replace = TRUE)
```

`mcor_test` *Multiple correlation test*

Description

Calculates correlations between multiple variables.

Usage

```
mcor_test(
  x,
  y = NULL,
  estimate = TRUE,
  p.value = FALSE,
  method = "spearman",
  method_adjust = "BH"
)
```

Arguments

`x`	Data frame containing numerical variables.
`y`	Data frame containing numerical variables. If `NULL`, correlations are calculated within `x`.
`estimate`	Logical specifying whether to return correlation coefficients.
`p.value`	Logical specifying whether to return adjusted p-values.
`method`	Character specifying the correlation method: `pearson`, `kendall`, or `spearman`.
`method_adjust`	Character specifying the p-value adjustment method.

Value

Depending on the values of `estimate` and `p.value`, one of the following:

estimate = TRUE, p.value = FALSE A numeric matrix of correlation coefficients, with columns corresponding to variables in `x` and rows to variables in `y`.

estimate = FALSE, p.value = TRUE A numeric matrix of adjusted p-values, with columns corresponding to variables in `x` and rows to variables in `y`.

estimate = TRUE, p.value = TRUE A named list with two elements:

- **estimate** Numeric matrix of correlation coefficients.
- **p.value** Numeric matrix of adjusted p-values.

Examples

```
library(magrittr)
x0 <- runif(20)
x <- lapply(
    c(1, -1),
    function(i) sapply(seq(10), function(j) x0 * i + runif(10, max = 1))
) %>%
    Reduce(cbind, .) %>%
    set_colnames(paste("Variable", seq(20)))
y <- lapply(
    c(1, -1),
    function(i) sapply(seq(10), function(j) x0 * i + runif(10, max = 1))
) %>%
    Reduce(cbind, .) %>%
    set_colnames(paste("Variable", seq(20))) %>%
    .[, seq(5)]
mcor_test(x)
mcor_test(
    x,
    y,
    p.value = TRUE,
    method = "pearson",
    method_adjust = "bonferroni"
)
```

`post_hoc_chi2` *Performs post hoc analysis for chi-squared or Fisher's exact test*

Description

Identifies pairwise differences between categories following a chi-squared or Fisher's exact test.

Usage

```
post_hoc_chi2(
  x,
  method = "fisher",
  method_adjust = "BH",
  digits = 3,
  count = FALSE,
  ...
)
```

Arguments

`x`	Data frame, vector, or table. If numeric, treated as a contingency table and the names are considered as categories; otherwise, the levels of the factor or the characters are used.
`method`	Character specifying the statistical test: `chisq` for chi-squared or `fisher` for Fisher's exact test.
`method_adjust`	Character specifying the p-value adjustment method.
`digits`	Integer specifying the number of decimal places for the test statistic.
`count`	Logical specifying if `x` is a contingency table.
`...`	Additional arguments passed to `chisq.test` or `fisher.test`.

Details

If `x` is numeric, it is treated as a contingency table and the names are considered as categories; otherwise, the levels of the factor or the characters are used.

Value

A tibble with pairwise test results containing the following columns:

group1, group2 Character vectors specifying the pair of groups being compared.

n Numeric vector specifying the total count or sample size for the comparison.

statistic Numeric vector specifying the test statistic (for chi-squared tests only).

df Numeric vector specifying the degrees of freedom (for chi-squared tests only).

p Raw p-value for the pairwise comparison, formatted as numeric or character ("< 0.001" for very small p-values).

p.signif Character vectors specifying the significance codes for raw p-values: 'ns' (not significant).

FDR False Discovery Rate adjusted p-value using the specified method, formatted as numeric or character ("< 0.001" for very small values).

fdr.signif Character vectors specifying the significance codes for FDR-adjusted p-values: 'ns' (not significant), '' *(p < 0.05),* '' **(p < 0.01),** '' (p < 0.001).

For Fisher's exact tests, the `statistic` and `df` columns are not included..

Examples

```
x <- c(rep("A", 100), rep("B", 78), rep("C", 25))
post_hoc_chi2(x)

x <- data.frame(G1 = c(Yes = 100, No = 78), G2 = c(Yes = 75, No = 23))
post_hoc_chi2(x, count = TRUE, method = "chisq")

data("housetasks")
housetasks[, c("Wife", "Husband")] %>%
    t() %>%
    post_hoc_chi2(count = TRUE, workspace = 1e6)

x <- cbind(
    mapply(function(x, y) rep(x, y), letters[seq(3)], c(7, 5, 8)) %>% unlist(),
    mapply(function(x, y) rep(x, y), LETTERS[seq(3)], c(6, 6, 8)) %>% unlist()
)
post_hoc_chi2(x)
```

`print_binomial` *Prints descriptive statistics for binomial variables*

Description

Calculates and prints frequency counts and percentages for binomial (two-level) categorical variables.

Usage

```
print_binomial(x, digits = 1, width = 15)
```

Arguments

`x`	Data frame, matrix, or vector containing binomial variables.
`digits`	Integer specifying the number of decimal places for the test statistic.
`width`	Integer specifying the maximum width for wrapping text.

Value

A tibble with one row per level for each categorical level containing the following columns:

Variables Character vector specifying the name of each variable.

Levels Character vector specifying the category level for each variable.

Statistics Character vector combining the frequency count and the percentage for each level.

Examples

```
x <- data.frame(A = sample(c("X", "Y"), 100, replace = TRUE))
print_binomial(x)
print_binomial(x, digits = 2, width = 5)
```

`print_chi2_test` *Prints the results of a Chi2*

Description

Formats the results of a Chi-squared or Fisher's exact test.

Usage

```
print_chi2_test(x, digits = 3)
```

Arguments

`x`	Test object from `rstatix` among `chisq_test` or `fisher_test`.
`digits`	Integer specifying the number of decimal places for the test statistic.

Value

A character string containing the formatted test results with:

Test statistic For Chi-squared test.

P-value Formatted p-value with significance stars.

Sample size Total count for sample size.

For Fisher's exact test, only the P-value and sample size are included.

Examples

```
x <- c(A = 100, B = 78, C = 25)
library(rstatix)
print_chi2_test(chisq_test(x))

xtab <- as.table(rbind(c(490, 10), c(400, 100)))
dimnames(xtab) <- list(
    group = c("grp1", "grp2"),
    smoker = c("yes", "no")
)
print_chi2_test(fisher_test(xtab))
```

`print_dispersion` *Prints the dispersion of a numeric vector*

Description

Calculates and prints the median and interquartile range (IQR) or the mean and standard deviation (SD).

Usage

```
print_dispersion(x, digits = 1, width = 15, method = "median")
```

Arguments

`x`	Vector containing numerical values.
`digits`	Integer specifying the number of decimal places for the test statistic.
`width`	Integer specifying the maximum width for wrapping text.
`method`	Character specifying the method: `median` for median and IQR, or `mean` for mean and SD.

Value

A character string containing a measure of central tendency and dispersion. Depending on `method`, this is either the median and interquartile range or the mean and standard deviation.

Examples

```
print_dispersion(runif(10))
print_dispersion(runif(10), method = "mean", digits = 2, width = 5)
```

`print_multinomial` *Prints descriptive statistics for multinomial variables*

Description

Calculates and prints frequency counts and percentages for multinomial (multi-level) categorical variables.

Usage

```
print_multinomial(x, label = NULL, digits = 1, width = 15, n = nrow(x), ...)
```

Arguments

`x`	Data frame, matrix, or vector containing multinomial variables.
`label`	Character vector specifying the names of the categorical variables.
`digits`	Integer specifying the number of decimal places for the test statistic.
`width`	Integer specifying the maximum width for wrapping text.
`n`	Integer specifying the total number of observations.
`...`	Additional arguments passed to `count_category`.

Value

A tibble with one row per level for each categorical level containing the following columns:

Variables Character vector specifying the name of each variable.

Levels Character vector specifying the category level for each variable.

Statistics Character vector combining the frequency count and the percentage for each level.

Examples

```
x <- data.frame(A = sample(c("X", "Y", "Z"), 100, replace = TRUE))
print_multinomial(x, label = "A")
x2 <- rbind(x, data.frame(A = rep("Level A", length(x[x == "Level X", ]))))
print_multinomial(
    x,
    label = "Variable A",
    sort = FALSE,
    n = 90,
    digits = 2,
    width = 5
)
```

`print_numeric` *Prints descriptive statistics for numeric variables*

Description

Prints summary statistics (mean, median, quartiles, range, etc.) for numeric variables.

Usage

```
print_numeric(x, digits = 1, width = 15)
```

Arguments

`x`	Data frame, matrix, or vector containing numerical variables.
`digits`	Integer specifying the number of decimal places for the test statistic.
`width`	Integer specifying the maximum width for wrapping text.

Value

A tibble with one row per numeric variable and the following columns:

Variables Character specifying the variable name.

Mean+/-SD Character specifying the mean and standard deviation.

Median+/-IQR Character specifying the median and interquartile range.

Q1-Q3 Character specifying the first and third quartiles.

Range Character specifying the minimum and maximum values.

Kurtosis Numeric specifying the kurtosis coefficient.

Skewness Numeric specifying the skewness coefficient.

Normality Character specifying the Shapiro-Wilk normality test significance code.

Zeros Integer specifying the number of zero values.

NAs Integer specifying the number of missing values.

Examples

```
x <- data.frame(A = rnorm(100), B = rnorm(100))
print_numeric(x)
print_numeric(x, digits = 2, width = 5)
```

`print_test` *Prints a hypothesis test*

Description

Formats the results of a hypothesis test (ANOVA, Kruskal-Wallis, or Wilcoxon).

Usage

```
print_test(x, digits = 0, digits_p = 2)
```

Arguments

`x`	Test object from `rstatix` among `anova_test`, `kruskal_test`, or `wilcox_test`.
`digits`	Integer specifying the number of decimal places for the test statistic.
`digits_p`	Integer specifying the number of decimal places for the p-value.

Value

A character string containing the formatted test results with:

Test name Name of the statistical test (ANOVA, Kruskal-Wallis, Wilcoxon, t-test, Friedman, or mixed-effects model).

Test statistic Test statistic (F, K, W, T, or χ^2) with degrees of freedom when applicable.

P-value P-value with significance stars.

Examples

```
library(rstatix)
data("ToothGrowth")
res <- anova_test(ToothGrowth, len ~ dose)
print_test(res)

res <- kruskal_test(ToothGrowth, len ~ dose)
print_test(res)

res <- wilcox_test(ToothGrowth, len ~ supp)
print_test(res)

library(lmerTest)
data("sleepstudy", package = "lme4")
res <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
print_test(res)
```

`summary_binomial` *Summarizes descriptive statistics for binomial variables*

Description

Summarizes descriptive statistics for binomial variables

Usage

```
summary_binomial(x, ...)
```

Arguments

`x`	Data frame, matrix, or vector containing binomial variables.
`...`	Additional arguments passed to `print_binomial`.

Value

A tibble with descriptive statistics containing the following columns:

Variables Character vector specifying the name of each variable.

Statistics Character vector combining the reference level of a variable with its frequency count and its percentage.

Examples

```
x <- data.frame(A = sample(c("X", "Y"), 100, replace = TRUE))
summary_binomial(x)
summary_binomial(x, digits = 2, width = 5)
```

`summary_numeric` *Summarizes descriptive statistics for numeric variables*

Description

Formats the output of `print_numeric` into a concise summary.

Usage

```
summary_numeric(x, ...)
```

Arguments

`x`	Data frame, matrix, or vector containing numerical variables.
`...`	Additional arguments passed to `print_numeric`.

Value

A tibble with one row per numeric variable and the following columns:

Variables Character specifying the variable name.

Median+/-IQR Character specifying the median and interquartile range.

Examples

```
x <- data.frame(A = rnorm(100), B = rnorm(100))
summary_numeric(x)
summary_numeric(x, digits = 2, width = 5)
```

`to_title` *Convert Strings to Title Case*

Description

Converts the first character of each string to uppercase and the rest to lowercase.

Usage

```
to_title(x)
```

Arguments

`x`	A character vector or a list containing strings to convert to title case.

Value

A character vector with the same length as `x`, where each element has its first character converted to uppercase and remaining characters are preserved as-is.

Examples

```
to_title(c("hELLO", "WoRLD", "R"))
# Returns: "Hello" "World" "R"
```

Index