Package 'kim'

April 25, 2021

Title Behavioral Scientists' Analysis Toolkit

Version 0.3.13

Description Miscellaneous functions to simplify and expedite analyses of experimental data. Examples include a function that plots sample means of groups in a factorial experimental design, a function that conducts robust regressions with bootstrapped samples, and a function that conducts robust two-way analysis of variance. Many of the functions will require installing package(s) listed in the Selected References.
Selected References:
Canty & Ripley (2021) <https://CRAN.R-project.org/package=boot>.
Cohen (1988) <doi:10.4324/9780203771587>.
DeCarlo (1997) <doi:10.1037/1082-989X.2.3.292>.
Dinno (2018) <https://CRAN.R-project.org/package=paran>.
Doane & Seward (2011) <doi:10.1080/10691898.2011.11889611>.
Dowle et al. (2021) <https://CRAN.R-project.org/package=data.table>.
Edwards et al. (2020) <https://CRAN.R-project.org/package=lemon>.
Fox et al. (2020) <https://CRAN.R-project.org/package=car>.
Hester et al. (2020) <https://CRAN.R-project.org/package=remotes>.
Ioannidis (2005) <doi:10.1371/journal.pmed.0020124>
Kim (2021) <doi:10.5281/zenodo.4445388>.
Kim (2020) <https://CRAN.R-project.org/package=ezr>.
Long (2020) <https://CRAN.R-project.org/package=interactions>.
Mair & Wilcox (2021) <https://CRAN.R-project.org/package=WRS2>.
Pasek et al. (2020) <https://CRAN.R-project.org/package=weights>.
Simmons et al. (2011) <doi:10.1177/0956797611417632>.
Tingley et al. (2019) <https://CRAN.R-project.org/package=mediation>.
Torchiano (2020) <https://CRAN.R-project.org/package=effsize>.
Wickham et al. (2020) <https://CRAN.R-project.org/package=ggplot2>.
Wilke (2021) <https://CRAN.R-project.org/package=ggridges>.

License GPL-3

URL https://github.com/jinkim3/kim, https://jinkim.science

BugReports https://github.com/jinkim3/kim/issues

Imports data.table, remotes

Suggests boot, ggplot2, moments

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Jin Kim [aut, cre] (<https://orcid.org/0000-0002-5013-3958>)

Maintainer Jin Kim <jin.m.kim@yale.edu>

Repository CRAN

Date/Publication 2021-04-24 22:40:09 UTC

R topics documented:

barplot_for_counts . . . 3
capitalize . . . 4
change_var_names . . . 5
check_req_pkg . . . 6
chi_square_test_pairwise . . . 6
clean_data_from_qualtrics . . . 7
coefficent_of_variation . . . 8
cohen_d . . . 9
cohen_d_from_cohen_textbook . . . 10
comma_sep_string_to_numbers . . . 11
compare_datasets . . . 11
compare_groups . . . 12
correlation_matrix . . . 13
cum_percent_plot . . . 14
desc_stats . . . 15
desc_stats_by_group . . . 16
find_duplicates . . . 16
floodlight_2_by_continuous . . . 17
ggsave_quick . . . 19
histogram . . . 20
histogram_by_group . . . 22
histogram_w_outlier_bins . . . 23
id_across_datasets . . . 25
install_all_dependencies . . . 26
kurtosis . . . 26
mann_whitney . . . 27
matrix_prep_dt . . . 28
mediation_analysis . . . 28
merge_data_tables . . . 30
merge_data_table_list . . . 31
multiple_regression . . . 32
order_rows_specifically_in_dt . . . 33
parallel_analysis . . . 34
percentile_rank . . . 35

plot_group_means . . . 35
population_variance . . . 37
prep . . . 37
pretty_round_p_value . . . 38
print_loop_progress . . . 39
proportion_of_values_in_vector . . . 40
read_csv . . . 41
read_sole_csv . . . 42
regex_match . . . 43
rel_pos_of_value_in_vector . . . 43
rel_value_of_pos_in_vector . . . 44
remove_user_installed_pkgs . . . 44
robust_regression . . . 45
round_flexibly . . . 46
scatterplot . . . 47
score_scale_items . . . 49
setup_r_env . . . 50
setwd_to_active_doc . . . 51
se_of_mean . . . 51
skewness . . . 52
start_kim . . . 53
su . . . 54
tabulate_vector . . . 54
theme_kim . . . 56
top_median_or_bottom . . . 57
two_way_anova . . . 58
t_test_pairwise . . . 60
update_kim . . . 61
wilcoxon_rank_sum_test . . . 62
write_csv . . . 62

Index **64**

`barplot_for_counts` *Barplot for counts*

Description

Barplot for counts

Usage

```
barplot_for_counts(data = NULL, x, y)
```

Arguments

`data`	a data object (a data frame or a data.table)
`x`	name of the variable that will be on the x axis of the barplot
`y`	name of the variable that will be on the y axis of the barplot

Examples

```
barplot_for_counts(
  data = data.frame(
    cyl = names(table(mtcars$cyl)),
    count = as.vector(table(mtcars$cyl))
  ),
  x = "cyl", y = "count"
)
```

`capitalize` *Capitalize a substring*

Description

Capitalizes the first letter (by default) or a substring of a given character string or each element of the character vector

Usage

```
capitalize(x, start = 1, end = 1)
```

Arguments

`x`	a character string or a character vector
`start`	starting position of the susbtring (default = 1)
`end`	ending position of the susbtring (default = 1)

Value

a character string or a character vector

Examples

```
capitalize("abc")
capitalize(c("abc", "xyx"), start = 2, end = 3)
```

`change_var_names` *Change variable names in a data set*

Description

Change variable names in a data set

Usage

```
change_var_names(
  data = NULL,
  old_var_names = NULL,
  new_var_names = NULL,
  skip_absent = FALSE,
  print_summary = TRUE,
  output_type = "dt"
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`old_var_names`	a vector of old variable names (i.e., variable names to change)
`new_var_names`	a vector of new variable names
`skip_absent`	If `skip_absent = TRUE`, old variable names that do not exist in the data set will be skipped (default = TRUE).
`print_summary`	If `print_summary = TRUE`, a summary of old and new variable names will printed. (default = TRUE)
`output_type`	type of the output. If `output_type = "dt"`, the function's output will be a data.table with changed names. If `output_type = "summary"`, the function's output will be a data.table listing old and new variable names. By default, `output_type = "dt"`.

Value

a data.table object with changed variable names

Examples

```
change_var_names(
mtcars, old = c("mpg", "cyl"), new = c("mpg_new", "cyl_new"))
```

check_req_pkg *Check for required packages*

Description

Check whether required packages are installed.

Usage

```
check_req_pkg(pkg = NULL)
```

Arguments

pkg	a character vector containing names of packages to check

Value

there will be no output from this function. Rather, the function will check whether the packages given as inputs are installed.

Examples

```
check_req_pkg("data.table")
check_req_pkg(c("base", "utils", "ggplot2", "data.table"))
```

chi_square_test_pairwise *Chi square test, pairwise*

Description

Conducts a chi-square test for every possible pairwise comparison with Bonferroni correction

Usage

```
chi_square_test_pairwise(
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  focal_dv_value = NULL,
  percentages_only = NULL,
  counts_only = NULL,
  sigfigs = 3,
  chi_sq_test_stats = FALSE
)
```

Arguments

data	a data object (a data frame or a data.table)
iv_name	name of the independent variable (must be a categorical variable)
dv_name	name of the dependent variable (must be a binary variable)
focal_dv_value	focal value of the dependent variable whose frequencies will be calculated (i.e., the value of the dependent variable that will be considered a "success" or a result of interest)
percentages_only	tabulate percentages of the focal DV value only
counts_only	tabulate counts of the focal DV value only
sigfigs	number of significant digits to round to
chi_sq_test_stats	if chi_sq_test_stats = TRUE, chi-square test statistic and degrees of freedom will be included in the pairwise comparison data.table.

Examples

```
chi_square_test_pairwise(data = mtcars, iv_name = "vs", dv_name = "am")
chi_square_test_pairwise(data = mtcars, iv_name = "vs", dv_name = "am",
percentages_only = TRUE)
```

---

clean_data_from_qualtrics

Clean data from Qualtrics

Description

Clean a data set downloaded from Qualtrics

Usage

```
clean_data_from_qualtrics(
  data = NULL,
  remove_survey_preview_data = TRUE,
  remove_test_respose_data = TRUE,
  default_cols_by_qualtrics = NULL,
  default_cols_by_qualtrics_new = NULL,
  warn_accuracy_loss = FALSE,
  click_data_cols = "rm",
  page_submit_cols = "move_to_right"
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`remove_survey_preview_data`	logical. Whether to remove data from survey preview (default = TRUE)
`remove_test_respose_data`	logical. Whether to remove data from test response (default = TRUE)
`default_cols_by_qualtrics`	names of columns that Qualtrics includes in the data set by default (e.g., "StartDate", "Finished"). Accepting the default value `default_cols_by_qualtrics = NULL` will set the names to be those that Qualtrics uses as of Dec 25, 2020.
`default_cols_by_qualtrics_new`	new names for columns that Qualtrics includes in the data set by default (e.g., "StartDate", "Finished"). Accepting the default value `default_cols_by_qualtrics_new = NULL` will set the names to be those that Qualtrics uses as of Dec 25, 2020 converted to snake_case (e.g., "start_date", "finished").
`warn_accuracy_loss`	logical. whether to warn the user if converting character to numeric leads to loss of accuracy. (default = FALSE)
`click_data_cols`	if `click_data_cols = "rm"`, columns containing click data (e.g., "_First Click") will be removed. If `click_data_cols = "move_to_right"`, the columns will be moved to the right (end) of the data set.
`page_submit_cols`	if `page_submit_cols = "rm"`, columns containing page submit data (e.g., "_Page Submit"; "response time" data) will be removed. If `page_submit_cols = "move_to_right"`, the columns will be moved to the right (end) of the data set.

Value

a data.table object

Examples

```
clean_data_from_qualtrics(mtcars)
clean_data_from_qualtrics(mtcars, default_cols_by_qualtrics = "mpg",
default_cols_by_qualtrics_new = "mpg2")
```

---

coefficent_of_variation

Coefficient of variation

Description

Calculates the (population or sample) coefficient of variation of a given numeric vector

Usage

```
coefficent_of_variation(vector, pop_or_sample = "pop")
```

Arguments

vector	a numeric vector
pop_or_sample	should coefficient of variation be calculated for a "population" or a "sample"?

Value

a numeric value

Examples

```
coefficent_of_variation(1:4, pop_or_sample = "sample")
coefficent_of_variation(1:4, pop_or_sample = "pop")
```

---

cohen_d *Cohen's d with confidence interval*

Description

Calculates Cohen's d and its confidence interval.

Usage

```
cohen_d(
  sample_1 = NULL,
  sample_2 = NULL,
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  ci_range = 0.95
)
```

Arguments

sample_1	a vector of values in the first of two samples
sample_2	a vector of values in the second of two samples
data	a data object (a data frame or a data.table)
iv_name	name of the independent variable
dv_name	name of the dependent variable
ci_range	range of the confidence interval for Cohen's d (default = 0.95)

Details

To construct confidence interval around Cohen's d with this function, the following package(s) must be installed prior to running this function: Package 'effsize' v0.8.1 (or possibly a higher version) by Marco Torchiano (2020), https://cran.r-project.org/package=effsize

Examples

```
cohen_d(1:10, 3:12)
cohen_d(data = mtcars, iv_name = "vs", dv_name = "mpg", ci_range = 0.99)
```

---

cohen_d_from_cohen_textbook

Cohen's d from Jacob Cohen's textbook (1988)

Description

Calculates Cohen's d as described in Jacob Cohen's textbook (1988), Statistical Power Analysis for the Behavioral Sciences, 2nd Edition Cohen, J. (1988) doi: 10.4324/9780203771587

Usage

```
cohen_d_from_cohen_textbook(
  sample_1 = NULL,
  sample_2 = NULL,
  data = NULL,
  iv_name = NULL,
  dv_name = NULL
)
```

Arguments

sample_1	a vector of values in the first of two samples
sample_2	a vector of values in the second of two samples
data	a data object (a data frame or a data.table)
iv_name	name of the independent variable
dv_name	name of the dependent variable

Value

the output will be a Cohen's d value (a numeric vector of length one)

Examples

```
cohen_d_from_cohen_textbook(1:10, 3:12)
cohen_d_from_cohen_textbook(
  data = mtcars, iv_name = "vs", dv_name = "mpg"
)
```

comma_sep_string_to_numbers *Convert a comma-separated string of numbers*

Description

Convert a comma-separated string of numbers

Usage

```
comma_sep_string_to_numbers(string)
```

Arguments

string	a character string consisting of numbers separated by commas

Value

a character string

Examples

```
comma_sep_string_to_numbers("1, 2, 3,4,  5  6")
```

compare_datasets *Compare data sets*

Description

Compares whether or not data sets are identical

Usage

```
compare_datasets(dataset_1 = NULL, dataset_2 = NULL, dataset_list = NULL)
```

Arguments

dataset_1	a data object (a data frame or a data.table)
dataset_2	another data object (a data frame or a data.table)
dataset_list	list of data objects (data.frame or data.table)

Value

the output will be a data.table showing differences in data sets

Examples

```
# catch differences in class attributes of the data sets
compare_datasets(
dataset_1 = data.frame(a = 1:2, b = 3:4),
dataset_2 = data.table::data.table(a = 1:2, b = 3:4))
# catch differences in number of columns
compare_datasets(
dataset_1 = data.frame(a = 1:2, b = 3:4, c = 5:6),
dataset_2 = data.frame(a = 1:2, b = 3:4))
# catch differences in number of rows
compare_datasets(
dataset_1 = data.frame(a = 1:2, b = 3:4),
dataset_2 = data.frame(a = 1:10, b = 11:20))
# catch differences in column names
compare_datasets(
dataset_1 = data.frame(A = 1:2, B = 3:4),
dataset_2 = data.frame(a = 1:2, b = 3:4))
# catch differences in values within corresponding columns
compare_datasets(
dataset_1 = data.frame(a = 1:2, b = c(3, 400)),
dataset_2 = data.frame(a = 1:2, b = 3:4))
compare_datasets(
dataset_1 = data.frame(a = 1:2, b = 3:4, c = 5:6),
dataset_2 = data.frame(a = 1:2, b = c(3, 4), c = c(5, 6)))
# check if data sets in a list are identical
compare_datasets(
dataset_list = list(
dt1 = data.frame(a = 1:2, b = 3:4, c = 5:6),
dt2 = data.frame(a = 1:2, b = 3:4),
dt3 = data.frame(a = 1:2, b = 3:4, c = 5:6)))
```

---

compare_groups *Compare groups*

Description

Compares groups by (1) creating histogram by group; (2) summarizing descriptive statistics by group; and (3) conducting pairwise comparisons (t-tests and Mann-Whitney tests).

Usage

```
compare_groups(
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  sigfigs = 3,
  mann_whitney = TRUE,
  t_test_stats = FALSE
)
```

Arguments

data	a data object (a data frame or a data.table)
iv_name	name of the independent variable (grouping variable)
dv_name	name of the dependent variable (measure variable of interest)
sigfigs	number of significant digits to round to
mann_whitney	if mann_whitney = TRUE, Mann-Whitney test results will be included in the pairwise comparison data.table. If mann_whitney = FALSE, Mann-Whitney tests will not be performed.
t_test_stats	if t_test_stats = TRUE, t-test statistic and degrees of freedom will be included in the pairwise comparison data.table.

Value

the output will be a list of (1) ggplot object (histogram by group) (2) a data.table with descriptive statistics by group; and (3) a data.table with pairwise comparison results

Examples

```
compare_groups(data = iris, iv_name = "Species", dv_name = "Sepal.Length")
```

---

correlation_matrix *correlation matrix*

Description

Creates a correlation matrix

Usage

```
correlation_matrix(
  data = NULL,
  var_names = NULL,
  row_var_names = NULL,
  col_var_names = NULL,
  round_r = 2,
  round_p = 3,
  output_type = "r"
)
```

Arguments

data	a data object (a data frame or a data.table)
var_names	names of the variables for which to calculate all pairwise correlations
row_var_names	names of the variables that will go on the rows of the correlation matrix
col_var_names	names of the variables that will go on the columns of the correlation matrix

round_r	number of decimal places to which to round correlation coefficients (default = 2)
round_p	number of decimal places to which to round p-values (default = 3)
output_type	which value should be filled in cells of the correlation matrix? If output_type = "r", correlation coefficients; if output_type = "p", p-values; if output_type = "rp", correlation coefficients with significance symbols based on p-values; if output_type = "n", sizes of the samples used to calculate the correlation coefficients

Value

the output will be a correlation matrix in a data.table format

Examples

```
correlation_matrix(data = mtcars, var_names = c("mpg", "cyl", "wt"))
correlation_matrix(data = mtcars,
row_var_names = c("mpg", "cyl", "hp"), col_var_names = c("wt", "am"))
```

---

cum_percent_plot *Cumulative percentage plot*

Description

Plots or tabulates cumulative percentages associated with elements in a vector

Usage

```
cum_percent_plot(vector, output_type = "plot")
```

Arguments

vector	a numeric vector
output_type	if output_type = "plot", return a cumulative percentage plot; if output_type = "dt", return a data.table with cumulative percentages. By default, output_type = "plot"

Examples

```
cum_percent_plot(c(1:100, NA, NA))
cum_percent_plot(mtcars$mpg)
cum_percent_plot(vector= mtcars$mpg, output_type = "dt")
```

desc_stats *Descriptive statistics*

Description

Returns descriptive statistics for a numeric vector.

Usage

```
desc_stats(
  vector = NULL,
  output_type = "vector",
  sigfigs = 3,
  ci = TRUE,
  pi = TRUE,
  notify_na_count = NULL,
  print_dt = TRUE
)
```

Arguments

vector	a numeric vector
output_type	if output_type = "vector", return a vector of descriptive statistics; if output_type = "dt", return a data.table of descriptive statistics (default = "vector")
sigfigs	number of significant digits to round to (default = 3)
ci	logical. Should 95% CI be included in the descriptive stats? (default = TRUE)
pi	logical. Should 95% PI be included in the descriptive stats? (default = TRUE)
notify_na_count	if TRUE, notify how many observations were removed due to missing values. By default, NA count will be printed only if there are any NA values.
print_dt	if TRUE, print the descriptive stats data.table

Value

if output_type = "vector", the output will be a named numeric vector of descriptive statistics; if output_type = "dt", the output will be data.table of descriptive statistics.

Examples

```
desc_stats(1:100)
desc_stats(1:100, ci = TRUE, pi = TRUE, sigfigs = 2)
desc_stats(c(1:100, NA))
desc_stats(vector = c(1:100, NA), output_type = "dt")
```

desc_stats_by_group *Descriptive statistics by group*

Description

Returns descriptive statistics by group

Usage

```
desc_stats_by_group(
  data = NULL,
  var_for_stats = NULL,
  grouping_vars = NULL,
  sigfigs = NULL,
  cols_to_round = NULL
)
```

Arguments

data	a data object (a data frame or a data.table)
var_for_stats	name of the variable for which descriptive statistics will be calculated
grouping_vars	name(s) of grouping variables
sigfigs	number of significant digits to round to
cols_to_round	names of columns whose values will be rounded

Value

the output will be a data.table showing descriptive statistics of the variable for each of the groups formed by the grouping variables.

Examples

```
desc_stats_by_group(data = mtcars, var_for_stats = "mpg",
grouping_vars = c("vs", "am"))
```

find_duplicates *Find duplicated values in a vector*

Description

Find duplicated values in a vector

Usage

```
find_duplicates(vector = NULL, na.rm = TRUE, sigfigs = 2, output = "summary")
```

Arguments

vector	a vector whose elements will be checked for duplicates
na.rm	logical. If na.rm = TRUE, NA values in the vector will be removed before searching for duplicates. If na.rm = FALSE, NA values will be included in the search as potentially duplicated values.
sigfigs	number of significant digits to round to in the percent column of the summary (default = 2)
output	type of output. If output = "summary", the function's output will be a data.table summarizing duplicated values and their counts. If output = "duplicated_values", the function's output will be a vector of duplicated values. If output = "non_duplicated_values", the function's output will be a vector of non-duplicated values (default = "summary")

Value

the output will be a data.table object (summary), a vector of duplicated values, or a vector non-duplicated values.

Examples

```
find_duplicates(mtcars$cyl)
find_duplicates(mtcars$cyl, output = "duplicated_values")
find_duplicates(vector = c(mtcars$cyl, 11:20, NA, NA))
find_duplicates(vector = c(mtcars$cyl, 11:20, NA, NA), na.rm = FALSE)
find_duplicates(vector = c(mtcars$cyl, 11:20, NA, NA),
na.rm = FALSE, sigfigs = 4, output = "duplicated_values")
```

---

floodlight_2_by_continuous

Floodlight 2 by Continuous

Description

Conduct a floodlight analysis for 2 x Continuous design.

Usage

```
floodlight_2_by_continuous(
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  mod_name = NULL,
  interaction_p_include = TRUE,
  iv_level_order = NULL,
  output = "reg_lines_plot",
  jitter_x_percent = 0,
```

```
  jitter_y_percent = 0,
  dot_alpha = 0.5,
  dot_size = 4,
  legend_position = "right",
  reg_line_types = c("solid", "dashed"),
  jn_line_types = c("solid", "solid"),
  sig_region_color = "green",
  sig_region_alpha = 0.08,
  nonsig_region_color = "gray",
  nonsig_region_alpha = 0.08,
  x_axis_title = NULL,
  y_axis_title = NULL,
  legend_title = NULL,
  round_decimals_int_p_value = 3
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`iv_name`	name of the binary independent variable
`dv_name`	name of the dependent variable
`mod_name`	name of the continuous moderator variable
`interaction_p_include`	logical. Should the plot include a p-value for the interaction term?
`iv_level_order`	order of levels in the independent variable for legend. By default, it will be set as levels of the independent variable ordered using R's base function `sort`.
`output`	type of output (default = "reg_lines_plot").
`jitter_x_percent`	horizontally jitter dots by a percentage of the range of x values
`jitter_y_percent`	vertically jitter dots by a percentage of the range of y values#'
`dot_alpha`	opacity of the dots (0 = completely transparent, 1 = completely opaque). By default, `dot_alpha = 0.5`
`dot_size`	size of the dots (default = 4)
`legend_position`	position of the legend (default = "right"). If `legend_position = "none"`, the legend will be removed.
`reg_line_types`	types of the regression lines for the two levels of the independent variable. By default, `reg_line_types = c("solid","dashed")`
`jn_line_types`	types of the lines for Johnson-Neyman points. By default, `jn_line_types = c("solid","solid")`
`sig_region_color`	color of the significant region, i.e., range(s) of the moderator variable for which simple effect of the independent variable on the dependent variable is statistically significant.

`sig_region_alpha`
: opacity for `sig_region_color`. (0 = completely transparent, 1 = completely opaque). By default, `sig_region_alpha = 0.08`

`nonsig_region_color`
: color of the non-significant region, i.e., range(s) of the moderator variable for which simple effect of the independent variable on the dependent variable is not statistically significant.

`nonsig_region_alpha`
: opacity for `nonsig_region_color`. (0 = completely transparent, 1 = completely opaque). By default, `nonsig_region_alpha = 0.08`

`x_axis_title`
: title of the x axis. By default, it will be set as input for `mod_name`. If `x_axis_title = FALSE`, it will be removed.

`y_axis_title`
: title of the y axis. By default, it will be set as input for `dv_name`. If `y_axis_title = FALSE`, it will be removed.

`legend_title`
: title of the legend. By default, it will be set as input for `iv_name`. If `legend_title = FALSE`, it will be removed.

`round_decimals_int_p_value`
: To how many digits after the decimal point should the p value for the interaction term be rounded? (default = 3)

Details

The following package(s) must be installed prior to running this function: Package 'interactions' v1.1.1 (or possibly a higher version) by Jacob A. Long (2019), https://cran.r-project.org/package=interactions

Examples

```
floodlight_2_by_continuous(
data = mtcars,
iv_name = "am",
dv_name = "mpg",
mod_name = "qsec")
```

---

ggsave_quick *ggsave quick*

Description

quickly save the current plot with a timestamp

Usage

```
ggsave_quick(
  name = NULL,
  timestamp = NULL,
  file_name_extension = ".png",
  width = 16,
  height = 9
)
```

Arguments

name	a character string of the png file name. By default, if no input is given (name = NULL), the file name will begin with "ggplot". If the desired output file name is "myplot.png", enter name = "myplot",timestamp = FALSE
timestamp	if timestamp = TRUE, a timestamp of the current time will be appended to the file name. The timestamp will be in the format, jan_01_2021_1300_10_000001, where "jan_01_2021" would indicate January 01, 2021; 1300 would indicate 13:00 (i.e., 1 PM); and 10_000001 would indicate 10.000001 seconds after the hour. By default, timestamp will be set as TRUE, if no input is given for the name argument, and as FALSE, if an input is given for the name argument.
file_name_extension	file name extension (default = ".png")
width	width of the plot to be saved. This argument will be directly entered as the width argument for the ggsave function within ggplot2 package (default = 16)
height	height of the plot to be saved. This argument will be directly entered as the height argument for the ggsave function within ggplot2 package (default = 9)

Value

the output will be a .png image file in the working directory.

Examples

```
## Not run:
kim::histogram(rep(1:30, 3))
ggsave_quick()

## End(Not run)
```

---

histogram *Histogram*

Description

Create a histogram

Usage

```
histogram(
  vector = NULL,
  number_of_bins = 30,
  x_tick_marks = NULL,
  y_tick_marks = NULL,
  fill_color = "cyan4",
  border_color = "black",
  y_axis_title_vjust = 0.85,
  x_axis_title = NULL,
  y_axis_title = NULL,
  cap_axis_lines = FALSE,
  notify_na_count = NULL
)
```

Arguments

vector	a numeric vector
number_of_bins	number of bins for the histogram (default = 30)
x_tick_marks	a vector of values at which to place tick marks on the x axis (e.g., setting x_tick_marks = seq(0,10,5) will put tick marks at 0, 5, and 10.)
y_tick_marks	a vector of values at which to place tick marks on the y axis (e.g., setting y_tick_marks = seq(0,10,5) will put tick marks at 0, 5, and 10.)
fill_color	color for inside of the bins (default = "cyan4")
border_color	color for borders of the bins (default = "black")
y_axis_title_vjust	position of the y axis title (default = 0.85).
x_axis_title	title for x axis (default = "Value")
y_axis_title	title for y axis (default = "Count")
cap_axis_lines	logical. Should the axis lines be capped at the outer tick marks? (default = FALSE)
notify_na_count	if TRUE, notify how many observations were removed due to missing values. By default, NA count will be printed only if there are any NA values.

Value

the output will be a histogram, a ggplot object.

Examples

```
histogram(1:100)
histogram(c(1:100, NA))
histogram(vector = mtcars[["mpg"]])
histogram(vector = mtcars[["mpg"]], x_tick_marks = seq(10, 36, 2))
histogram(vector = mtcars[["mpg"]], x_tick_marks = seq(10, 36, 2),
```

```
y_tick_marks = seq(0, 8, 2), y_axis_title_vjust = 0.5,
y_axis_title = "Freq", x_axis_title = "Values of mpg")
```

`histogram_by_group` *Histogram by group*

Description

Creates histograms by group to compare distributions.

Usage

```
histogram_by_group(
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  order_of_groups_top_to_bot = NULL,
  number_of_bins = 40,
  space_between_histograms = 0.15,
  draw_baseline = FALSE
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`iv_name`	name of the independent variable
`dv_name`	name of the dependent variable
`order_of_groups_top_to_bot`	a character vector indicating the desired presentation order of levels in the independent variable (from the top to bottom). Omitting a group in this argument will remove the group in the set of histograms.
`number_of_bins`	number of bins for the histograms (default = 40)
`space_between_histograms`	space between histograms (minimum = 0, maximum = 1, default = 0.15)
`draw_baseline`	logical. Should the baseline and the trailing lines to either side of the histogram be drawn? (default = FALSE)

Details

The following package(s) must be installed prior to running this function: Package 'ggridges' v0.5.3 (or possibly a higher version) by Claus O. Wilke (2021), https://cran.r-project.org/package=ggridges

Value

the output will be a set of vertically arranged histograms (a ggplot object), i.e., one histogram for each level of the independent variable.

Examples

```
histogram_by_group(data = mtcars, iv_name = "cyl", dv_name = "mpg")
histogram_by_group(
  data = mtcars, iv_name = "cyl", dv_name = "mpg",
  order_of_groups_top_to_bot = c("8", "4"), number_of_bins = 10,
  space_between_histograms = 0.5
)
histogram_by_group(
data = iris, iv_name = "Species", dv_name = "Sepal.Length")
```

---

histogram_w_outlier_bins

Histogram with outlier bins

Description

Create a histogram with outlier bins

Usage

```
histogram_w_outlier_bins(
  vector = NULL,
  bin_cutoffs = NULL,
  outlier_bin_left = TRUE,
  outlier_bin_right = TRUE,
  x_tick_marks = NULL,
  x_tick_mark_labels = NULL,
  y_tick_marks = NULL,
  outlier_bin_fill_color = "coral",
  non_outlier_bin_fill_color = "cyan4",
  border_color = "black",
  y_axis_title_vjust = 0.85,
  x_axis_title = NULL,
  y_axis_title = NULL,
  notify_na_count = NULL,
  plot_proportion = TRUE,
  plot_frequency = FALSE,
  mean = TRUE,
  ci = TRUE,
  median = TRUE,
  median_position = 15,
  error_bar_size = 3
)
```

Arguments

vector	a numeric vector
bin_cutoffs	cutoff points for bins
outlier_bin_left	logical. Should the leftmost bin treated as an outlier bin? (default = TRUE)
outlier_bin_right	logical. Should the rightmost bin treated as an outlier bin? (default = TRUE)
x_tick_marks	a vector of values at which to place tick marks on the x axis. Note that the first bar spans from 0.5 to 1.5, second bar from 1.5 to 2.5, ... nth bar from n - 0.5 to n + 0.5. See the example. By default, tick marks will be placed at every cutoff point for bins
x_tick_mark_labels	a character vector to label tick marks. By default, the vector of cutoff points for bins will also be used as labels.
y_tick_marks	a vector of values at which to place tick marks on the y axis (e.g., setting `y_tick_marks = seq(0,10,5)` will put tick marks at 0, 5, and 10.)
outlier_bin_fill_color	color to fill inside of the outlier bins (default = "coral")
non_outlier_bin_fill_color	color to fill inside of the non-outlier bins (default = "cyan4")
border_color	color for borders of the bins (default = "black")
y_axis_title_vjust	position of the y axis title (default = 0.85).
x_axis_title	title for x axis (default = "Value"). If `x_axis_title = FALSE`, x axis title will be removed from the plot.
y_axis_title	title for y axis. By default, it will be either "Proportion" or "Count".
notify_na_count	if `TRUE`, notify how many observations were removed due to missing values. By default, NA count will be printed only if there are any NA values.
plot_proportion	logical. Should proportions be plotted, as opposed to frequencies? (default = TRUE)
plot_frequency	logical. Should frequencies be plotted, as opposed to proportions? (default = FALSE). If `plot_frequency = TRUE`, `plot_proportion` will switch to be FALSE.
mean	logical. Should mean marked on the histogram? (default = TRUE)
ci	logical. Should 95% confidence interval marked on the histogram? (default = TRUE)
median	logical. Should median marked on the histogram? (default = TRUE)
median_position	position of the median label as a percentage of height of the tallest bin (default = 15)
error_bar_size	size of the error bars (default = 3)

Value

a ggplot object

Examples

```
histogram_w_outlier_bins(vector = 1:100, bin_cutoffs = seq(0, 100, 10))
histogram_w_outlier_bins(vector = 0:89, bin_cutoffs = seq(0, 90, 10),
x_tick_marks = seq(0.5, 9.5, 3), x_tick_mark_labels = seq(0, 90, 30))
histogram_w_outlier_bins(vector = 1:10, bin_cutoffs = seq(0, 10, 2.5))
histogram_w_outlier_bins(vector = 1:5, bin_cutoffs = seq(0, 10, 2.5))
histogram_w_outlier_bins(vector = 1:15, bin_cutoffs = c(5.52, 10.5))
```

---

id_across_datasets *ID across datasets*

Description

Create an ID column in each of the data sets. The ID values will span across the data sets.

Usage

```
id_across_datasets(
  dt_list = NULL,
  id_col_name = "id",
  id_col_position = "first",
  silent = FALSE
)
```

Arguments

dt_list	a list of data.table objects
id_col_name	name of the column that will contain ID values
id_col_position	position of the newly created ID column. If id_col_position = "first", the new ID column will be placed as the first column in respective data sets. If id_col_position = "last", the new ID column will be placed as the last column in respective data sets.
silent	If silent = TRUE, a summary of starting and ending ID values in each data set will not be printed. If silent = FALSE, a summary of starting and ending ID values in each data set will be printed. (default = FALSE)

Value

the output will be a list of data.table objects.

Examples

```
# running the examples below requires importing the data.table package.
prep(data.table)
id_across_datasets(
dt_list = list(setDT(copy(mtcars)), setDT(copy(iris))))
id_across_datasets(
dt_list = list(setDT(copy(mtcars)), setDT(copy(iris)), setDT(copy(women))),
id_col_name = "newly_created_id_col",
id_col_position = "last")
```

`install_all_dependencies` *Install all dependencies for all functions*

Description

Install all dependencies for all functions in Package 'kim'.

Usage

```
install_all_dependencies()
```

Value

there will be no output from this function. Rather, dependencies of all functions in Package 'kim' will be installed.

Examples

```
## Not run:
install_all_dependencies()

## End(Not run)
```

`kurtosis` *Kurtosis*

Description

Calculate kurtosis of the sample using a formula for either the (1) biased estimator or (2) an unbiased estimator of the population kurtosis. Formulas were taken from DeCarlo (1997), doi: 10.1037/1082-989X.2.3.292

Usage

```
kurtosis(vector = NULL, unbiased = TRUE)
```

Arguments

vector	a numeric vector
unbiased	logical. If unbiased = TRUE, the unbiased estimate of the population kurtosis will be calculated. If unbiased = FALSE, the biased estimate of the population kurtosis will be calculated. By default, unbiase = TRUE.

Value

a numeric value, i.e., kurtosis of the given vector

Examples

```
# calculate the unbiased estimator (e.g., kurtosis value that
# Excel 2016 will produce)
kim::kurtosis(c(1, 2, 3, 4, 5, 10))
# calculate the biased estimator (e.g., kurtosis value that
# R Package 'moments' will produce)
kim::kurtosis(c(1, 2, 3, 4, 5, 10), unbiased = FALSE)
# compare with kurtosis from 'moments' package
moments::kurtosis(c(1, 2, 3, 4, 5, 10))
```

---

mann_whitney *Mann-Whitney U Test (Also called Wilcoxon Rank-Sum Test)*

Description

A nonparametric equivalent of the independent t-test

Usage

```
mann_whitney(data = NULL, iv_name = NULL, dv_name = NULL, sigfigs = 3)
```

Arguments

data	a data object (a data frame or a data.table)
iv_name	name of the independent variable (grouping variable)
dv_name	name of the dependent variable (measure variable of interest)
sigfigs	number of significant digits to round to

Value

the output will be a data.table object with all pairwise Mann-Whitney test results

Examples

```
mann_whitney(data = iris, iv_name = "Species", dv_name = "Sepal.Length")
```

matrix_prep_dt *Prepare a two-column data.table that will be used to fill values in a matrix*

Description

Prepare a two-column data.table that will be used to fill values in a matrix

Usage

```
matrix_prep_dt(row_var_names = NULL, col_var_names = NULL)
```

Arguments

row_var_names	a vector of variable names, each of which will be header of a row in the eventual matrix
col_var_names	a vector of variable names, each of which will be header of a column in the eventual matrix

Examples

```
matrix_prep_dt(
  row_var_names = c("mpg", "cyl"),
  col_var_names = c("hp", "gear")
)
```

mediation_analysis *Mediation analysis*

Description

Conducts a mediation analysis to estimate an independent variable's indirect effect on dependent variable through a mediator variable. The current version of the package only supports a simple mediation model consisting of one independent variable, one mediator variable, and one dependent variable.

Usage

```
mediation_analysis(
  data = NULL,
  iv_name = NULL,
  mediator_name = NULL,
  dv_name = NULL,
  covariates_names = NULL,
  robust_se = TRUE,
  iterations = 1000,
```

```
  sigfigs = 3,
  output_type = "summary_dt",
  silent = FALSE
)
```

Arguments

data	a data object (a data frame or a data.table)
iv_name	name of the independent variable
mediator_name	name of the mediator variable
dv_name	name of the dependent variable
covariates_names	names of covariates to control for
robust_se	if TRUE, heteroskedasticity-consistent standard errors will be used in quasi-Bayesian simulations. By default, it will be set as FALSE if nonparametric bootstrap is used and as TRUE if quasi-Bayesian approximation is used.
iterations	number of bootstrap samples. The default is set at 1000, but consider increasing the number of samples to 5000, 10000, or an even larger number, if slower handling time is not an issue.
sigfigs	number of significant digits to round to
output_type	if output_type = "summary_dt", return the summary data.table; if output_type = "mediate_output", return the output from the mediate function in the 'mediate' package; if output_type = "indirect_effect_p", return the p value associated with the indirect effect estimated in the mediation model (default = "summary_dt")
silent	if silent = FALSE, mediation analysis summary, estimation method, sample size, and number of simulations will be printed; if silent = TRUE, nothing will be printed. (default = FALSE)

Details

This function requires installing Package 'mediation' v4.5.0 (or possibly a higher version) by Tingley et al. (2019), and uses the source code from a function in the package. https://cran.r-project.org/package=mediation

Value

if output_type = "summary_dt", which is the default, the output will be a data.table showing a summary of mediation analysis results; if output_type = "mediate_output", the output will be the output from the mediate function in the 'mediate' package; if output_type = "indirect_effect_p", the output will be the p-value associated with the indirect effect estimated in the mediation model (a numeric vector of length one).

Examples

```
mediation_analysis(
  data = mtcars, iv_name = "cyl",
  mediator_name = "disp", dv_name = "mpg", iterations = 100
)
mediation_analysis(
  data = iris, iv_name = "Sepal.Length",
  mediator_name = "Sepal.Width", dv_name = "Petal.Length",
  iterations = 100
)
```

---

merge_data_tables *Merge data tables*

Description

Merge two data.table objects. If there are any duplicated ID values and column names across the two data tables, the cell values in the first data.table will remain intact and the cell values in the second data.table will be discarded for the resulting merged data table.

Usage

```
merge_data_tables(dt1 = NULL, dt2 = NULL, id = NULL, silent = TRUE)
```

Arguments

dt1	the first data.table which will remain intact
dt2	the second data.table which will be joined outside of (around) the first data.table. If there are any duplicated ID values and column names across the two data tables, the cell values in the first data.table will remain intact and the cell values in the second data.table will be discarded for the resulting merged data table.
id	name of the column that will contain the ID values in the two data tables. The name of the ID column must be identical in the two data tables.
silent	If silent = TRUE, no message will be printed regarding how many ID values and column names were duplicated. If silent = FALSE, messages will be printed regarding how many ID values and column names were duplicated. (default = FALSE)

Value

a data.table object, which merges (joins) the second data.table around the first data.table.

Examples

```
data_1 <- data.table::data.table(
id_col = c(4, 2, 1, 3),
a = 3:6,
b = 5:8,
c = c("w", "x", "y", "z"))
data_2 <- data.table::data.table(
id_col = c(1, 4, 99),
d = 6:8,
b = c("p", "q", "r"),
e = c(TRUE, FALSE, FALSE))
merge_data_tables(dt1 = data_1, dt2 = data_2, id = "id_col")
```

---

merge_data_table_list *Merge a list of data tables*

Description

Successively merge a list of data.table objects in a recursive fashion. That is, merge the (second data table in the list) around the first data table in the list; then, around this resulting data table, merge the third data table in the list; and so on.

Usage

```
merge_data_table_list(dt_list = NULL, id = NULL, silent = TRUE)
```

Arguments

dt_list	a list of data.table objects
id	name of the column that will contain the ID values in the data tables. The name of the ID column must be identical in the all data tables.
silent	If silent = TRUE, no message will be printed regarding how many ID values and column names were duplicated. If silent = FALSE, messages will be printed regarding how many ID values and column names were duplicated. (default = FALSE)

Details

If there are any duplicated ID values and column names across the data tables, the cell values in the earlier data table will remain intact and the cell values in the later data table will be discarded for the resulting merged data table in each recursion.

Value

a data.table object, which successively merges (joins) a data table around (i.e., outside) the previous data table in the list of data tables.

Examples

```
data_1 <- data.table::data.table(
id_col = c(4, 2, 1, 3),
a = 3:6,
b = 5:8,
c = c("w", "x", "y", "z"))
data_2 <- data.table::data.table(
id_col = c(1, 4, 99),
d = 6:8,
b = c("p", "q", "r"),
e = c(TRUE, FALSE, FALSE))
data_3 <- data.table::data.table(
id_col = c(200, 3),
f = 11:12,
b = c(300, "abc"))
merge_data_table_list(
dt_list = list(data_1, data_2, data_3), id = "id_col")
```

---

multiple_regression *Multiple regression*

Description

Conduct multiple regression analysis and summarize the results in a data.table.

Usage

```
multiple_regression(
  data = NULL,
  formula = NULL,
  sigfigs = NULL,
  round_digits_after_decimal = NULL
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`formula`	a formula object for the regression equation
`sigfigs`	number of significant digits to round to
`round_digits_after_decimal`	round to nth digit after decimal (alternative to `sigfigs`)

Details

To include standardized beta(s) in the regression results table, the following package(s) must be installed prior to running the function: Package 'lm.beta' v1.5-1 (or possibly a higher version) by Stefan Behrendt (2014), https://cran.r-project.org/package=lm.beta

Value

the output will be a data.table showing multiple regression results.

Examples

```
multiple_regression(data = mtcars, formula = mpg ~ gear * cyl)
```

`order_rows_specifically_in_dt` *Order rows specifically in a data table*

Description

Order rows in a data.table in a specific order

Usage

```
order_rows_specifically_in_dt(
  dt = NULL,
  col_to_order_by = NULL,
  specific_order = NULL
)
```

Arguments

`dt`	a data.table object
`col_to_order_by`	a character value indicating the name of the column by which to order the data.table
`specific_order`	a vector indicating a specific order of the values in the column by which to order the data.table.

Value

the output will be a data.table object whose rows will be ordered as specified.

Examples

```
order_rows_specifically_in_dt(mtcars, "carb", c(3, 2, 1, 4, 8, 6))
```

`parallel_analysis` *Parallel analysis*

Description

Conducts a parallel analysis to determine how many factors to retain in a factor analysis.

Usage

```
parallel_analysis(
  data = NULL,
  names_of_vars = NULL,
  iterations = NULL,
  percentile_for_eigenvalue = 95
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`names_of_vars`	names of the variables
`iterations`	number of random data sets. If no input is entered, this value will be set as 30 * number of variables.
`percentile_for_eigenvalue`	percentile used in estimating bias (default = 95).

Details

The following package(s) must be installed prior to running the function: Package 'paran' v1.5.2 (or possibly a higher version) by Alexis Dinno (2018), https://cran.r-project.org/package=paran

Examples

```
parallel_analysis(
  data = mtcars, names_of_vars = c("disp", "hp", "drat")
)
# parallel_analysis(
# data = mtcars, names_of_vars = c("carb", "vs", "gear", "am"))
```

percentile_rank *Percentile rank*

Description

Calculate percentile rank of each value in a vector

Usage

```
percentile_rank(vector)
```

Arguments

vector	a numeric vector

Examples

```
percentile_rank(1:5)
percentile_rank(1:10)
percentile_rank(1:100)
```

plot_group_means *Plot group means*

Description

Creates a plot of sample means and error bars by group.

Usage

```
plot_group_means(
  data = NULL,
  dv_name = NULL,
  iv_name = NULL,
  na.rm = TRUE,
  error_bar = "ci",
  error_bar_range = 0.95,
  lines_connecting_means = TRUE,
  line_size = 1,
  dot_size = 3,
  error_bar_tip_width = 0.13,
  error_bar_thickness = 1,
  position_dodge = 0.13,
  legend_position = "right",
  y_axis_title_vjust = 0.85
)
```

Arguments

data	a data object (a data frame or a data.table)
dv_name	name of the dependent variable
iv_name	name(s) of the independent variable(s). Up to two independent variables can be supplied.
na.rm	logical. If na.rm = TRUE, NA values in independent and dependent variables will be removed before calculating group means.
error_bar	if error_bar = "se"; error bars will be +/-1 standard error, if error_bar = "ci" error bars will be a confidence interval; if error_bar = "pi", error bars will be a prediction interval
error_bar_range	width of the confidence or prediction interval (default = 0.95 for 95 percent confidence or prediction interval). This argument will not apply when error_bar = "se"
lines_connecting_means	logical. Should lines connecting means within each group be drawn? (default = TRUE)
line_size	thickness of the lines connecting group means, (default = 1)
dot_size	size of the dots indicating group means (default = 3)
error_bar_tip_width	graphically, width of the segments at the end of error bars (default = 0.13)
error_bar_thickness	thickness of the error bars (default = 1)
position_dodge	by how much should the group means and error bars be horizontally offset from each other so as not to overlap? (default = 0.13)
legend_position	position of the legend: "none","top","right","bottom","left","none" (default = "right")
y_axis_title_vjust	position of the y axis title (default = 0.85). If default is used, y_axis_title_vjust = 0.85, the y axis title will be positioned at 85% of the way up from the bottom of the plot.

Value

by default, the output will be a ggplot object. If output = "table", the output will be a data.table object.

Examples

```
plot_group_means(data = mtcars, dv_name = "mpg", iv_name = c("vs", "am"))
plot_group_means(
  data = mtcars, dv_name = "mpg", iv_name = c("vs", "am"),
  error_bar = "se"
)
plot_group_means(
```

```
  data = mtcars, dv_name = "mpg", iv_name = c("vs", "am"),
  error_bar = "pi", error_bar_range = 0.99
)
```

`population_variance` *Population variance of a vector*

Description

Calculates the population variance, rather than the sample variance, of a vector

Usage

```
population_variance(vector, na.rm = TRUE)
```

Arguments

`vector`	a numeric vector
`na.rm`	if `TRUE`, NA values will be removed before calculation

Examples

```
population_variance(1:4)
var(1:4)
```

`prep` *Prepare package(s) for use*

Description

Installs, loads, and attaches package(s). If package(s) are not installed, installs them prior to loading and attaching.

Usage

```
prep(
  ...,
  pkg_names_as_object = FALSE,
  silent_if_successful = FALSE,
  silent_load_pkgs = NULL
)
```

Arguments

`...`	names of packages to load and attach, separated by commas, e.g., `"ggplot2",data.table`. The input can be any number of packages, whose names may or may not be wrapped in quotes.
`pkg_names_as_object`	logical. If `pkg_names_as_object = TRUE`, the input will be evaluated as one object containing package names. If `pkg_names_as_object = FALSE`, the input will be considered as literal packages names (default = FALSE).
`silent_if_successful`	logical. If `silent_if_successful = TRUE`, no message will be printed if preparation of package(s) is successful. If `silent_if_successful = FALSE`, a message indicating which package(s) were successfully loaded and attached will be printed (default = FALSE).
`silent_load_pkgs`	a character vector indicating names of packages to load silently (i.e., suppress messages that get printed when loading the packaged). By default, `silent_load_pkgs = NULL`

Value

there will be no output from this function. Rather, packages given as inputs to the function will be installed, loaded, and attached.

Examples

```
prep(data.table)
prep("data.table", silent_if_successful = TRUE)
prep("base", utils, ggplot2, "data.table")
pkgs <- c("ggplot2", "data.table")
prep(pkgs, pkg_names_as_object = TRUE)
prep("data.table", silent_load_pkgs = "data.table")
```

`pretty_round_p_value` *Pretty round p-value*

Description

Pretty round p-value

Usage

```
pretty_round_p_value(
  p_value_vector = NULL,
  round_digits_after_decimal = 3,
  include_p_equals = FALSE
)
```

Arguments

p_value_vector one number or a numeric vector

round_digits_after_decimal
: how many digits after the decimal point should the p-value be rounded to?

include_p_equals
: if TRUE, output will be a string of mathematical expression including "p", e.g., "p < .01" (default = FALSE)

Value

the output will be a character vector with p values, e.g., a vector of strings like "< .001" (or "p < .001").

Examples

```
pretty_round_p_value(
  p_value_vector = 0.049,
  round_digits_after_decimal = 2, include_p_equals = FALSE
)
pretty_round_p_value(c(0.0015, 0.0014), include_p_equals = TRUE)
```

---

print_loop_progress *print loop progress*

Description

Print current progress inside a loop (e.g., for loop or lapply)

Usage

```
print_loop_progress(
  iteration_number = NULL,
  iteration_start = 1,
  iteration_end = NULL,
  text_before = "",
  percent = 1,
  output_method = "cat"
)
```

Arguments

iteration_number
: current number of iteration

iteration_start
: iteration number at which the loop begins (default = 1)

iteration_end iteration number at which the loop ends.

text_before	text to add before "Loop Progress..." By default, it is set to be blank, i.e., text_before = ""
percent	if percent = 1, progress level will be printed at every 1 percent progress (default = 1)
output_method	if output_method = "cat", progress level will be printed using the 'cat' function; if output_method = "return", progress level will be returned as the output of the function (default = "cat")

Examples

```
for (i in seq_len(250)) {
  Sys.sleep(0.001)
  print_loop_progress(
    iteration_number = i,
    iteration_end = 250)
}
unlist(lapply(seq_len(7), function (i) {
  Sys.sleep(0.1)
  print_loop_progress(
    iteration_number = i,
    iteration_end = 7)
  return(i)
}))
```

---

proportion_of_values_in_vector

Proportion of given values in a vector

Description

Proportion of given values in a vector

Usage

```
proportion_of_values_in_vector(
  values = NULL,
  vector = NULL,
  na.exclude = TRUE,
  output_type = "proportion",
  silent = FALSE,
  conf.level = 0.95,
  correct_yates = TRUE
)
```

Arguments

values	a set of values that will count as successes (hits)
vector	a numeric or character vector containing successes (hits) and failures (misses)
na.exclude	if TRUE, NA values will be removed both from vector and values before calculation (default = TRUE).
output_type	By default, output_type = "proportion". If output_type = "proportion", the function will return the calculated proportion; if output_type = "se", the function will return the standard error of the sample proportion; if output_type = "dt", the function will return the the data table of proportion and confidence intervals.
silent	If silent = TRUE, no message will be printed regarding number of NA values or confidence interval. (default = FALSE)
conf.level	confidence level of the returned confidence interval. Input to this argument will be passed onto the conf.level argument in the prop.test function from the default stats package.
correct_yates	a logical indicating whether Yates' continuity correction should be applied where possible (default = TRUE). Input to this argument will be passed onto the correct argument in the prop.test function from the default stats package.

Examples

```
proportion_of_values_in_vector(
  values = 2:3, vector = c(rep(1:3, each = 10), rep(NA, 10))
)
proportion_of_values_in_vector(
  values = 2:3, vector = c(rep(1:3, each = 10), rep(NA, 10)),
  output_type = "se"
)
proportion_of_values_in_vector(
  values = 2:3, vector = c(rep(1:3, each = 10), rep(NA, 10)),
  conf.level = 0.99
)
proportion_of_values_in_vector(
  values = c(2:3, NA), vector = c(rep(1:3, each = 10), rep(NA, 10)),
  na.exclude = FALSE
)
```

---

read_csv *Read a csv file*

Description

Read a csv file

Usage

```
read_csv(name = NULL, head = FALSE, ...)
```

Arguments

name	a character string of the csv file name without the ".csv" extension. For example, if the csv file to read is "myfile.csv", enter name = "myfile"
head	logical. if head = TRUE, prints the first five rows of the data set.
...	optional arguments for the fread function from the data.table package. Any arguments for data.table's fread function can be used, e.g., fill = TRUE, nrows = 100

Value

the output will be a data.table object, that is, an output from the data.table function, fread

Examples

```
## Not run:
mydata <- read_csv("myfile")

## End(Not run)
```

---

read_sole_csv *Read the sole csv file in the working directory*

Description

Read the sole csv file in the working directory

Usage

```
read_sole_csv(head = FALSE, ...)
```

Arguments

head	logical. if head = TRUE, prints the first five rows of the data set.
...	optional arguments for the fread function from the data.table package. Any arguments for data.table's fread function can be used, e.g., fill = TRUE, nrows = 100

Value

the output will be a data.table object, that is, an output from the data.table function, fread

Examples

```
mydata <- read_sole_csv()
mydata <- read_sole_csv(head = TRUE)
mydata <- read_sole_csv(fill = TRUE, nrows = 5)
```

regex_match *Regular expression matches*

Description

Returns elements of a character vector that match the given regular expression

Usage

```
regex_match(regex = NULL, vector = NULL, mute_report = FALSE, perl = FALSE)
```

Arguments

regex	a regular expression provided, a default theme will be used.
vector	a character vector in which to search for regular expression matches
mute_report	logical. Should the report on regular expression matches be printed?
perl	logical. Should Perl-compatible regexps be used?

Examples

```
regex_match("p$", names(mtcars))

colnames_ending_with_p <- regex_match("p$", names(mtcars))
```

rel_pos_of_value_in_vector *Find relative position of a value in a vector*

Description

Find relative position of a value in a vector that may or may not contain the value

Usage

```
rel_pos_of_value_in_vector(value = NULL, vector = NULL)
```

Arguments

value	a value whose relative position is to be searched in a vector
vector	a numeric vector

Value

a number indicating the relative position of the value in the vector

Examples

```
rel_pos_of_value_in_vector(value = 3, vector = c(2, 4))
rel_pos_of_value_in_vector(value = 3, vector = c(2, 6))
rel_pos_of_value_in_vector(value = 3, vector = 1:3)
```

`rel_value_of_pos_in_vector`

Find relative value of a position in a vector

Description

Find relative value of a position in a vector

Usage

```
rel_value_of_pos_in_vector(vector = NULL, position = NULL)
```

Arguments

`vector`	a numeric vector
`position`	position of a vector

Value

a number indicating the relative value of the position in the vector

Examples

```
rel_value_of_pos_in_vector(vector = c(0, 100), position = 1.5)
rel_value_of_pos_in_vector(vector = 2:4, position = 2)
rel_value_of_pos_in_vector(vector = c(2, 4, 6), position = 2.5)
```

`remove_user_installed_pkgs`

Remove all user installed packages

Description

Remove all user installed packages

Usage

```
remove_user_installed_pkgs(
  exceptions = NULL,
  type_of_pkg_to_keep = c("base", "recommended"),
  keep_kim = FALSE
)
```

Arguments

exceptions	a character vector of names of packages to keep
type_of_pkg_to_keep	a character vector indicating types of packages to keep. The default, type_of_pkg_to_keep = c("base","recommended"), keeps all base and recommended packages that come with R when R is installed.
keep_kim	logical. If keep_kim = FALSE, Package 'kim' will be removed along with all other user-installed packages. If keep_kim = TRUE, Package 'kim' will not be removed. By default, keep_kim = FALSE

Examples

```
## Not run:
remove_user_installed_pkgs()

## End(Not run)
```

---

robust_regression *Robust regression (bootstrapped regression)*

Description

Estimate coefficients in a multiple regression model by bootstrapping.

Usage

```
robust_regression(
  data = NULL,
  formula = NULL,
  sigfigs = NULL,
  round_digits_after_decimal = NULL,
  iterations = 1000
)
```

Arguments

data	a data object (a data frame or a data.table)
formula	a formula object for the regression equation
sigfigs	number of significant digits to round to
round_digits_after_decimal	round to nth digit after decimal (alternative to sigfigs)
iterations	number of bootstrap samples. The default is set at 1000, but consider increasing the number of samples to 5000, 10000, or an even larger number, if slower handling time is not an issue.

Details

The following package(s) must be installed prior to running this function: Package 'boot' v1.3-26 (or possibly a higher version) by Canty & Ripley (2021), https://cran.r-project.org/package=boot

Examples

```
robust_regression(
  data = mtcars, formula = mpg ~ cyl * hp,
  iterations = 100
)
```

---

round_flexibly *Round flexibly*

Description

Round numbers to a flexible number of significant digits. "Flexible" rounding refers to rounding all numbers to the highest level of precision seen among numbers that would have resulted from the 'signif()' function in base R. The examples of this function demonstrate flexible rounding.

Usage

```
round_flexibly(x = NULL, sigfigs = 3)
```

Arguments

x	a numeric vector
sigfigs	number of significant digits to flexibly round to. By default, sigfigs = 3.

Value

the output will be a numeric vector with values rounded to the highest level of precision seen among numbers that result from the 'signif()' function in base R.

Examples

```
# Example 1
# First, observe results from the 'signif' function:
c(0.00012345, pi)
signif(c(0.00012345, pi), 3)
# In the result above, notice how info is lost on some digits
# (e.g., 3.14159265 becomes 3.140000).
# In contrast, flexible rounding retains the lost info in the digits
round_flexibly(x = c(0.00012345, pi), sigfigs = 3)

# Example 2
# Again, first observe results from the 'signif' function:
```

```
c(0.12345, 1234, 0.12, 1.23, .01)
signif(c(0.12345, 1234, 0.12, 1.23, .01), 3)
# In the result above, notice how info is lost on some digits
# (e.g., 1234 becomes 1230.000).
# In contrast, flexible rounding retains the lost info in the digits.
# Specifically, in the example below, 0.12345 rounded to 3 significant
# digits (default) is signif(0.12345, 3) = 0.123 (3 decimal places).
# Because this 3 decimal places is the highest precision seen among
# all numbers, all other numbers will also be rounded to 3 decimal places.
round_flexibly(
c(0.12345, 1234, 0.12, 1.23, .01))
```

`scatterplot` *Scatterplot*

Description

Creates a scatter plot and calculates a correlation between two variables.

Usage

```
scatterplot(
  data = NULL,
  x_var_name = NULL,
  y_var_name = NULL,
  point_label_var_name = NULL,
  weight_var_name = NULL,
  alpha = 1,
  annotate_stats = FALSE,
  annotate_y_pos = 5,
  line_of_fit_type = "lm",
  ci_for_line_of_fit = FALSE,
  x_axis_label = NULL,
  y_axis_label = NULL,
  point_label_size = NULL,
  point_size_range = c(3, 12),
  jitter_x_percent = 0,
  jitter_y_percent = 0,
  cap_axis_lines = FALSE
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`x_var_name`	name of the variable that will go on the x axis
`y_var_name`	name of the variable that will go on the y axis
`point_label_var_name`	name of the variable that will be used to label individual observations

`weight_var_name`
: name of the variable by which to weight the individual observations for calculating correlation and plotting the line of fit

`alpha`
: opacity of the dots (0 = completely transparent, 1 = completely opaque)

`annotate_stats`
: if `TRUE`, the correlation and p-value will be annotated at the top of the plot

`annotate_y_pos`
: position of the annotated stats, expressed as a percentage of the range of y values by which the annotated stats will be placed above the maximum value of y in the data set (default = 5). If `annotate_y_pos = 5`, and the minimum and maximum y values in the data set are 0 and 100, respectively, the annotated stats will be placed at 5% of the y range (100 - 0) above the maximum y value, y = 0.05 * (100 - 0) + 100 = 105.

`line_of_fit_type`
: if `line_of_fit_type = "lm"`, a regression line will be fit; if `line_of_fit_type = "loess"`, a local regression line will be fit; if `line_of_fit_type = "none"`, no line will be fit

`ci_for_line_of_fit`
: if `ci_for_line_of_fit = TRUE`, confidence interval for the line of fit will be shaded

`x_axis_label`
: alternative label for the x axis

`y_axis_label`
: alternative label for the y axis

`point_label_size`
: size for dots' labels on the plot. If no input is entered for this argument, it will be set as `point_label_size = 5` by default. If the plot is to be weighted by some variable, this argument will be ignored, and dot sizes will be determined by the argument `point_size_range`

`point_size_range`
: minimum and maximum size for dots on the plot when they are weighted

`jitter_x_percent`
: horizontally jitter dots by a percentage of the range of x values

`jitter_y_percent`
: vertically jitter dots by a percentage of the range of y values

`cap_axis_lines`
: logical. Should the axis lines be capped at the outer tick marks? (default = TRUE)

Details

If a weighted correlation is to be calculated, the following package(s) must be installed prior to running the function: Package 'weights' v1.0 (or possibly a higher version) by John Pasek (2018), https://cran.r-project.org/package=weights

Value

the output will be a scatter plot, a ggplot object.

Examples

```
scatterplot(data = mtcars, x_var_name = "wt", y_var_name = "mpg")
scatterplot(
  data = mtcars, x_var_name = "wt", y_var_name = "mpg",
  point_label_var_name = "hp", weight_var_name = "drat",
  annotate_stats = TRUE
)
scatterplot(
  data = mtcars, x_var_name = "wt", y_var_name = "mpg",
  point_label_var_name = "hp", weight_var_name = "cyl",
  point_label_size = 7, annotate_stats = TRUE
)
```

---

score_scale_items *Score scale items*

Description

Score items in a scale (e.g., Likert scale items) by computing the sum or mean of the items.

Usage

```
score_scale_items(
  items = NULL,
  reverse = NULL,
  operation = "mean",
  na_summary = TRUE,
  reverse_code_minuend = NULL
)
```

Arguments

items	list of scale items (i.e., list of vectors of ratings) to code normally (as opposed to reverse coding).
reverse	list of scale items to reverse code.
operation	if operation = "mean", mean of the scale items will be calculated; if operation = "sum", sum of the scale items will be calculated (default = "mean").
na_summary	logical. If na_summary = TRUE a summary of NA values will be printed; if na_summary = FALSE the summary will not be printed (default = TRUE).
reverse_code_minuend	required for reverse coding; the number from which to subtract item ratings when reverse-coding. For example, if the items to reverse code are measured on a 7-point scale, enter reverse_code_minuend = 8.

Examples

```
score_scale_items(items = list(1:5, rep(3, 5)),
reverse = list(rep(5, 5)), reverse_code_minuend = 6)
score_scale_items(items = list(c(1, 1), c(1, 5)),
reverse = list(c(5, 3)), reverse_code_minuend = 6, na_summary = FALSE)
score_scale_items(items = list(c(1, 1), c(1, 5)),
reverse = list(c(5, 1)), reverse_code_minuend = 6, operation = "sum")
```

---

setup_r_env *Set up R environment*

Description

Set up R environment by (1) clearing the console; (2) removing all objects in the global environment; (3) setting the working directory to the active document (in RStudio only); (4) unloading and loading the kim package.

Usage

```
setup_r_env(
  clear_console = TRUE,
  clear_global_env = TRUE,
  setwd_to_active_doc = TRUE,
  prep_kim = TRUE
)
```

Arguments

clear_console	if TRUE, clear the console (default = TRUE)
clear_global_env	if TRUE, remove all objects in the global environment (default = TRUE)
setwd_to_active_doc	if TRUE, set the working directory to the active document in RStudio (default = TRUE)
prep_kim	if TRUE, unload and load the kim package (default = TRUE)

Examples

```
## Not run:
setup_r_env()

## End(Not run)
```

setwd_to_active_doc *Set working directory to active document in RStudio*

Description

Set working directory to location of the active document in RStudio

Usage

```
setwd_to_active_doc()
```

Value

there will be no output from this function. Rather, the working directory will be set as location of the active document.

Examples

```
## Not run:
setwd_to_active_doc()

## End(Not run)
```

se_of_mean *Standard error of the mean*

Description

Standard error of the mean

Usage

```
se_of_mean(vector, na.rm = TRUE, notify_na_count = NULL)
```

Arguments

vector	a numeric vector
na.rm	if TRUE, NA values will be removed before calculation
notify_na_count	if TRUE, notify how many observations were removed due to missing values. By default, NA count will be printed only if there are any NA values.

Value

the output will be a numeric vector of length one, which will be the standard error of the mean for the given numeric vector.

Examples

```
se_of_mean(c(1:10, NA))
```

`skewness` *Skewness*

Description

Calculate skewness using one of three formulas: (1) the traditional Fisher-Pearson coefficient of skewness; (2) the adjusted Fisher-Pearson standardized moment coefficient; (3) the Pearson 2 skewness coefficient. Formulas were taken from Doane & Seward (2011), doi: 10.1080/10691898.2011.11889611

Usage

```
skewness(vector = NULL, type = "adjusted")
```

Arguments

`vector`	a numeric vector
`type`	a character string indicating the type of skewness to calculate. If `type = "adjusted"`, the adjusted Fisher-Pearson standardized moment coefficient will be calculated. If `type = "traditional"`, the traditional Fisher-Pearson coefficient of skewness will be calculated. If `type = "pearson_2"`, the Pearson 2 skewness coefficient will be calculated. By default, `type = "adjusted"`.

Value

a numeric value, i.e., skewness of the given vector

Examples

```
# calculate the adjusted Fisher-Pearson standardized moment coefficient
kim::skewness(c(1, 2, 3, 4, 5, 10))
# calculate the traditional Fisher-Pearson coefficient of skewness
kim::skewness(c(1, 2, 3, 4, 5, 10), type = "traditional")
# compare with skewness from 'moments' package
moments::skewness(c(1, 2, 3, 4, 5, 10))
# calculate the Pearson 2 skewness coefficient
kim::skewness(c(1, 2, 3, 4, 5, 10), type = "pearson_2")
```

start_kim *Start kim*

Description

Start kim (update kim; attach default packages; set working directory, etc.)

Usage

```
start_kim(
  update = TRUE,
  upgrade_other_pkg = FALSE,
  setup_r_env = TRUE,
  default_packages = c("data.table", "ggplot2"),
  silent_load_pkgs = c("data.table", "ggplot2")
)
```

Arguments

update	If update = "force", force updating the package 'kim'. If update = TRUE, compares the currently installed package 'kim' with the most recent version on GitHub and, if the version on GitHub is more recent, ask the user to confirm the update. If confirmed, then update the package. If update = FALSE, skip updating the package. By default, update = "force"
upgrade_other_pkg	input for the upgrade argument to be passed on to remotes::install_github. One of "default", "ask", "always", "never", TRUE, or FALSE. "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE correspond to "always" and "never" respectively. By default, upgrade_other_pkg = FALSE.
setup_r_env	logical. If update = TRUE, runs the function setup_r_env in the package "kim". Type "?kim::setup_r_env" to learn more. By default, setup_r_env = TRUE
default_packages	a vector of names of packages to load and attach. By default, default_packages = c("data.table","ggplot2")
silent_load_pkgs	a character vector indicating names of packages to load silently (i.e., suppress messages that get printed when loading the packages). By default, silent_load_pkgs = c("data.table","ggplot2")

Examples

```
## Not run:
start_kim()
```

```
start_kim(default_packages = c("dplyr", "ggplot2"))
start_kim(update = TRUE, setup_r_env = FALSE)

## End(Not run)
```

---

su *su: Sorted unique values*

Description

Extract unique elements and sort them

Usage

```
su(x = NULL)
```

Arguments

x	a vector or a data frame or an array or NULL.

Value

a vector, data frame, or array-like 'x' but with duplicate elements/rows removed.

Examples

```
su(c(10, 3, 7, 10))
su(c("b", "z", "b", "a"))
```

---

tabulate_vector *Tabulate vector*

Description

Shows frequency and proportion of unique values in a table format

Usage

```
tabulate_vector(
  vector = NULL,
  na.rm = TRUE,
  sort_by_decreasing_count = NULL,
  sort_by_increasing_count = NULL,
  sort_by_decreasing_value = NULL,
  sort_by_increasing_value = NULL,
  total_included = TRUE,
  sigfigs = NULL,
```

```
  round_digits_after_decimal = NULL,
  output_type = "dt"
)
```

Arguments

vector	a character or numeric vector
na.rm	if TRUE, NA values will be removed before calculating frequencies and proportions.
sort_by_decreasing_count	if TRUE, the output table will be sorted in the order of decreasing frequency.
sort_by_increasing_count	if TRUE, the output table will be sorted in the order of increasing frequency.
sort_by_decreasing_value	if TRUE, the output table will be sorted in the order of decreasing value.
sort_by_increasing_value	if TRUE, the output table will be sorted in the order of increasing value.
total_included	if TRUE, the output table will include a row for total counts.
sigfigs	number of significant digits to round to
round_digits_after_decimal	round to nth digit after decimal (alternative to sigfigs)
output_type	if output_type = "df", return a data.frame. By default, output_type = "dt", which will return a data.table.

Value

if output_type = "dt", which is the default, the output will be a data.table showing the count and proportion (percent) of each element in the given vector; if output_type = "df", the output will be a data.frame showing the count and proportion (percent) of each value in the given vector.

Examples

```
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA))
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA),
  sort_by_increasing_count = TRUE
)
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA),
  sort_by_decreasing_value = TRUE
)
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA),
  sort_by_increasing_value = TRUE
)
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA),
  sigfigs = 4
)
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA),
  round_digits_after_decimal = 1
)
```

```
tabulate_vector(c("a", "b", "b", "c", "c", "c", NA),
  output_type = "df"
)
```

`theme_kim` *Theme Kim*

Description

A custom ggplot theme

Usage

```
theme_kim(
  legend_position = "none",
  base_size = 20,
  axis_tick_font_size = 20,
  axis_title_font_size = 24,
  y_axis_title_vjust = 0.85,
  axis_title_margin_size = 24,
  cap_axis_lines = FALSE
)
```

Arguments

`legend_position`	position of the legend (default = "none")
`base_size`	base font size
`axis_tick_font_size`	font size for axis tick marks
`axis_title_font_size`	font size for axis title
`y_axis_title_vjust`	position of the y axis title (default = 0.85). If default is used, `y_axis_title_vjust = 0.85`, the y axis title will be positioned at 85% of the way up from the bottom of the plot.
`axis_title_margin_size`	size of the margin between axis title and the axis line
`cap_axis_lines`	logical. Should the axis lines be capped at the outer tick marks? (default = FALSE)

Details

If a axis lines are to be capped at the ends, the following package(s) must be installed prior to running the function: Package 'lemon' v0.4.4 (or possibly a higher version) by Edwards et al. (2020), https://cran.r-project.org/package=lemon

Value

a ggplot object; there will be no meaningful output from this function. Instead, this function should be used with another ggplot object, e.g., `ggplot(mtcars ,aes(x = disp,y = mpg)) + theme_kim()`

Examples

```
prep(ggplot2)
ggplot2::ggplot(mtcars, aes(x = cyl, y = mpg)) + geom_point() + theme_kim()
```

---

top_median_or_bottom *Top, median, or bottom*

Description

Indicates whether each value in a vector belongs to top, median, or bottom

Usage

```
top_median_or_bottom(vector)
```

Arguments

`vector`	a numeric vector

Value

a character vector indicating whether each element in a vector belongs to "top", "median", or "bottom"

Examples

```
top_median_or_bottom(c(1, 2, 3, NA))
top_median_or_bottom(c(1, 2, 2, NA))
top_median_or_bottom(c(1, 1, 2, NA))
```

two_way_anova *Two-way ANOVA*

Description

Conduct a two-way analysis of variance (ANOVA).

Usage

```
two_way_anova(
  data = NULL,
  dv_name = NULL,
  iv_1_name = NULL,
  iv_2_name = NULL,
  iv_1_values = NULL,
  iv_2_values = NULL,
  sigfigs = 3,
  robust = FALSE,
  iterations = 2000,
  plot = FALSE,
  error_bar = "ci",
  error_bar_range = 0.95,
  line_size = 1,
  dot_size = 3,
  error_bar_tip_width = 0.13,
  position_dodge = 0.13,
  legend_position = "right",
  output = NULL
)
```

Arguments

data	a data object (a data frame or a data.table)
dv_name	name of the dependent variable
iv_1_name	name of the first independent variable
iv_2_name	name of the second independent variable
iv_1_values	restrict all analyses to observations having these values for the first independent variable
iv_2_values	restrict all analyses to observations having these values for the second independent variable
sigfigs	number of significant digits to which to round values in anova table (default = 3)
robust	if TRUE, conduct a robust ANOAVA in addition.

iterations	number of bootstrap samples for robust ANOVA. The default is set at 2000, but consider increasing the number of samples to 5000, 10000, or an even larger number, if slower handling time is not an issue.
plot	if TRUE, print a plot and enable returning an output.
error_bar	if error_bar = "se"; error bars will be +/-1 standard error, if error_bar = "ci" error bars will be a confidence interval; if error_bar = "pi", error bars will be a prediction interval
error_bar_range	width of the confidence or prediction interval (default = 0.95 for 95 percent confidence or prediction interval). This argument will not apply when error_bar = "se"
line_size	thickness of the lines connecting group means, (default = 1)
dot_size	size of the dots indicating group means (default = 3)
error_bar_tip_width	graphically, width of the segments at the end of error bars (default = 0.13)
position_dodge	by how much should the group means and error bars be horizontally offset from each other so as not to overlap? (default = 0.13)
legend_position	position of the legend: "none","top","right","bottom","left","none" (default = "right")
output	output type can be one of the following: "anova_table", "group_stats", "plot","levene_test_result","robust_anova_results","robust_anova_post_hoc_results", "robust_anova_post_hoc_contrast"

Details

The following package(s) must be installed prior to running this function: Package 'car' v3.0.9 (or possibly a higher version) by Fox et al. (2020), https://cran.r-project.org/package=car

If robust ANOVA is to be conducted, the following package(s) must be installed prior to running the function: Package 'WRS2' v1.1-1 (or possibly a higher version) by Mair & Wilcox (2021), https://cran.r-project.org/package=WRS2

Value

by default, the output will be "anova_table"

Examples

```
two_way_anova(
  data = mtcars, dv_name = "mpg", iv_1_name = "vs",
  iv_2_name = "am", iterations = 100
)
```

`t_test_pairwise` *t test, pairwise*

Description

Conducts a t-test for every possible pairwise comparison with Bonferroni correction

Usage

```
t_test_pairwise(
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  sigfigs = 3,
  mann_whitney = TRUE,
  t_test_stats = FALSE,
  t_test_df_decimals = 1,
  sd = FALSE
)
```

Arguments

`data`	a data object (a data frame or a data.table)
`iv_name`	name of the independent variable
`dv_name`	name of the dependent variable
`sigfigs`	number of significant digits to round to
`mann_whitney`	if `TRUE`, Mann-Whitney test results will be included in the output data.table. If `TRUE`, Mann-Whitney tests will not be performed.
`t_test_stats`	if `t_test_stats = TRUE`, t-test statistic and degrees of freedom will be included in the output data.table.
`t_test_df_decimals`	number of decimals for the degrees of freedom in t-tests (default = 1)
`sd`	if `sd = TRUE`, standard deviations will be included in the output data.table.

Value

the output will be a data.table showing results of all pairwise comparisons between levels of the independent variable.

Examples

```
t_test_pairwise(data = iris, iv_name = "Species", dv_name = "Sepal.Length")
t_test_pairwise(data = iris, iv_name = "Species",
dv_name = "Sepal.Length", t_test_stats = TRUE, sd = TRUE)
t_test_pairwise(data = iris, iv_name = "Species", dv_name = "Sepal.Length",
mann_whitney = FALSE)
```

update_kim *Update the package 'kim'*

Description

Updates the current package 'kim' by installing the most recent version of the package from GitHub

Usage

```
update_kim(force = TRUE, upgrade_other_pkg = FALSE, confirm = TRUE)
```

Arguments

force	logical. If force = TRUE, force installing the update. If force = FALSE, do not force installing the update. By default, force = TRUE.
upgrade_other_pkg	input for the upgrade argument to be passed on to remotes::install_github. One of "default", "ask", "always", "never", TRUE, or FALSE. "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE correspond to "always" and "never" respectively. By default, upgrade_other_pkg = FALSE.
confirm	logical. If confirm = TRUE, the user will need to confirm the update. If confirm = FALSE, the confirmation step will be skipped. By default, confirm = TRUE.

Value

there will be no output from this function. Rather, executing this function will update the current 'kim' package by installing the most recent version of the package from GitHub.

Examples

```
## Not run:
if (interactive()) {update_kim()}

## End(Not run)
```

wilcoxon_rank_sum_test *Wilcoxon Rank-Sum Test (Also called the Mann-Whitney U Test)*

Description

A nonparametric equivalent of the independent t-test

Usage

```
wilcoxon_rank_sum_test(
  data = NULL,
  iv_name = NULL,
  dv_name = NULL,
  sigfigs = 3
)
```

Arguments

data	a data object (a data frame or a data.table)
iv_name	name of the independent variable (grouping variable)
dv_name	name of the dependent variable (measure variable of interest)
sigfigs	number of significant digits to round to

Value

the output will be a data.table object with all pairwise Wilcoxon rank-sum test results

Examples

```
wilcoxon_rank_sum_test(
data = iris, iv_name = "Species", dv_name = "Sepal.Length")
```

write_csv *Write to a csv file*

Description

Write to a csv file

Usage

```
write_csv(data = NULL, name = NULL, timestamp = NULL)
```

Arguments

data	a data object (a data frame or a data.table)
name	a character string of the csv file name without the ".csv" extension. For example, if the csv file to write to is "myfile.csv", enter name = "myfile"
timestamp	logical. Should the timestamp be appended to the file name?

Value

the output will be a .csv file in the working directory, that is, an output from the data.table function, fwrite

Examples

```
## Not run:
write_csv(mtcars, "mtcars_from_write_csv")
write_csv(mtcars)

## End(Not run)
```

Index

barplot_for_counts, 3

capitalize, 4
change_var_names, 5
check_req_pkg, 6
chi_square_test_pairwise, 6
clean_data_from_qualtrics, 7
coefficent_of_variation, 8
cohen_d, 9
cohen_d_from_cohen_textbook, 10
comma_sep_string_to_numbers, 11
compare_datasets, 11
compare_groups, 12
correlation_matrix, 13
cum_percent_plot, 14

desc_stats, 15
desc_stats_by_group, 16

find_duplicates, 16
floodlight_2_by_continuous, 17

ggsave_quick, 19

histogram, 20
histogram_by_group, 22
histogram_w_outlier_bins, 23

id_across_datasets, 25
install_all_dependencies, 26

kurtosis, 26

mann_whitney, 27
matrix_prep_dt, 28
mediation_analysis, 28
merge_data_table_list, 31
merge_data_tables, 30
multiple_regression, 32

order_rows_specifically_in_dt, 33

parallel_analysis, 34
percentile_rank, 35
plot_group_means, 35
population_variance, 37
prep, 37
pretty_round_p_value, 38
print_loop_progress, 39
proportion_of_values_in_vector, 40

read_csv, 41
read_sole_csv, 42
regex_match, 43
rel_pos_of_value_in_vector, 43
rel_value_of_pos_in_vector, 44
remove_user_installed_pkgs, 44
robust_regression, 45
round_flexibly, 46

scatterplot, 47
score_scale_items, 49
se_of_mean, 51
setup_r_env, 50
setwd_to_active_doc, 51
skewness, 52
start_kim, 53
su, 54

t_test_pairwise, 60
tabulate_vector, 54
theme_kim, 56
top_median_or_bottom, 57
two_way_anova, 58

update_kim, 61

wilcoxon_rank_sum_test, 62
write_csv, 62