Package 'kim'

December 3, 2020

Title Functions for Behavioral Science Researchers

Version 0.2.20

Description Miscellaneous functions designed to facilitate and simplify 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 two-way analysis of variance.

License GPL-3

URL https://github.com/jinkim3/kim

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

Imports boot, car, data.table, devtools, dplyr, effsize, ggplot2, ggridges, lm.beta, mediation, moments, paran, rstudioapi, stats, utils, weights, WRS2

Encoding UTF-8

LazyData true

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 2020-12-03 07:50:03 UTC

R topics documented:

`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_first_letter` *Capitalize the first letter*

Description

Capitalizes the first letter of a given character string

Usage

```
capitalize_first_letter(x)
```

Arguments

`x`	a character string

Value

a character string

Examples

```
capitalize_first_letter("abc")
```

`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
)
```

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

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)
```

---

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 using the 'effsize' package v0.8.1 by Torchiano (2020) https://github.com/mtorchiano/effsize/

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)

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

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

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_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)
```

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 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")
```

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.

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, notify_na_count = NULL)
```

Arguments

vector	a numeric vector
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

a named numeric vector

Examples

```
desc_stats(1:100)
desc_stats(c(1:100, NA))
```

`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

a data.frame

Examples

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

`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
)
```

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)

Value

a ggplot object

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")
```

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. Uses the source code from 'mediation' package v4.5.0, Tingley et al. (2019) https://cran.r-project.org/package=mediation

Usage

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

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.

Examples

```
mediation_analysis(
  data = mtcars, iv_name = "cyl",
  mediator_name = "disp", dv_name = "mpg", iterations = 100
)
```

---

multiple_regression *Summarize multiple regression results in a data.table*

Description

Summarize multiple regression 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)

Examples

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

`parallel_analysis` *Parallel analysis*

Description

Conducts a parallel analysis to determine how many factors to retain in a factor analysis. Uses the 'paran' package v1.5.2 by Dinno (2018) https://cran.r-project.org/package=paran

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).

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,
  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"
)
```

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.
`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"`)

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(...)
```

Arguments

...	names of packages to load and attach, separated by commas, e.g., "ggplot2",data.table. The arguments can be any number of packages, and they may or may not be wrapped in quotes.

Examples

```
prep(data.table)
prep("base", utils, ggplot2, "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	round to nth digit after decimal
include_p_equals	if TRUE, output will be a string of mathematical expression including "p", e.g., "p < .01"

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)
```

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, vector, na.exclude = TRUE)
```

Arguments

values	a set of values
vector	a numeric or character vector
na.exclude	if TRUE, NA values will be removed both from vector and values before calculation

Examples

```
proportion_of_values_in_vector(
  values = 2:3, vector = c(1:3, NA)
)
proportion_of_values_in_vector(
  values = 2:3, vector = c(1:3, NA), na.exclude = FALSE
)
proportion_of_values_in_vector(
  values = c(2:3, NA), vector = c(1:3, NA), na.exclude = FALSE
)
```

---

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.

Examples

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

`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,
  line_of_fit_type = "lm",
  ci_for_line_of_fit = FALSE,
  x_axis_label = NULL,
  y_axis_label = NULL,
  point_labels_size_range = c(3, 12),
  jitter_x_percent = 0,
  jitter_y_percent = 0
)
```

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

`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_labels_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

Value

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",
  annotate_stats = TRUE
)
```

`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 current file; (4) unloading and loading the kim package

Usage

```
setup_r_env(
  clear_console = TRUE,
  clear_global_env = TRUE,
  set_wd_to_current_file = 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)
set_wd_to_current_file	if TRUE, set the working directory to the current file (default = TRUE)
prep_kim	if TRUE, unload and load the kim package (default = TRUE)

Examples

```
## Not run:
setup_r_env()

## 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.

Examples

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

`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

a data.table or data.frame

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"
)
```

---

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

Two-way 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,
  robust = TRUE,
  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
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"

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)
```

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

Value

the output will be a data.table

Examples

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

`update_kim` *Update 'kim' package*

Description

Updates the current package 'kim' by installing the most recent version

Usage

```
update_kim(source = "github")
```

Arguments

`source`	location of the most recent version of the package (default = "github")

Examples

```
update_kim()
```

Index