Package: apa7 (via r-universe)

November 22, 2026

Title Writing Documents in American Psychological Association Style, Seventh Edition

Version 0.2.0

Description Create American Psychological Association Style, Seventh Edition documents. Format numbers and text consistent with APA style. Create tables that comply with APA style by extending 'flextable' functions.

License CC0

URL https://github.com/wjschne/apa7, https://wjschne.github.io/apa7/

BugReports https://github.com/wjschne/apa7/issues

Depends R (>= 4.1.0)

Imports cli, dplyr, effectsize, flextable, ftExtra, glue, knitr, parameters, performance, psych, purrr, rlang, S7, scales, shiny, signs, snakecase, stringr, systemfonts, tibble, tidyr, tidyselect

Suggests afex, bsicons, bslib, car, carData, conflicted, forcats, fresh, ggplot2, lme4, MASS, ordinal, pbkrtest, quarto, ragg, rclipboard, readr, rmarkdown, sandwich, shinyWidgets, spelling, svglite, testthat (>= 3.0.0), tippy, toastui, withr, yaml

VignetteBuilder knitr, quarto

Config/Needs/website quarto

Config/testthat/edition 3

Encoding UTF-8

Language en-US

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author W. Joel Schneider [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-8393-5316>)

Maintainer W. Joel Schneider <w.joel.schneider@gmail.com>

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

Date/Publication 2026-11-21 20:40:09 UTC

RemoteUrl https://github.com/cran/apa7

RemoteRef HEAD

RemoteSha 8cccb8177c1dc025bcf84888edd7d9241019b6e9

Contents

add_break_columns . . . 3
add_list_column . . . 4
add_star_column . . . 5
align_chr . . . 6
apa_anova . . . 7
apa_chisq . . . 11
apa_cor . . . 13
apa_factor_cor . . . 14
apa_flextable . . . 15
apa_format_columns . . . 19
apa_loadings . . . 20
apa_note . . . 21
apa_p . . . 24
apa_p_star_note . . . 25
apa_parameters.aov . . . 26
apa_performance . . . 30
apa_performance_comparison . . . 32
apa_style . . . 33
apa7_defaults . . . 35
apa7_tbl . . . 36
column_format . . . 37
column_formats . . . 38
column_spanner_label . . . 39
hanging_indent . . . 40
install_apaquarto . . . 41
is_numeric_like . . . 42
make_apaquarto . . . 42
matrix_display . . . 43
num_pad . . . 44
p2stars . . . 45
pivot_wider_name_first . . . 46
pretty_widths . . . 48
sci_md . . . 49
separate_star_column . . . 49
star_balance . . . 50
str_wrap_equal . . . 51
tagger . . . 51

Index **53**

add_break_columns *Add break columns*

Description

Add break columns

Usage

```
add_break_columns(
  d,
  ...,
  .before = FALSE,
  omit_first = FALSE,
  omit_last = FALSE
)
```

Arguments

d	data.frame or tibble
...	Column name or tidyselect function. Select columns
.before	insert break columns before selected columns (defaults to FALSE)
omit_first	omit the first break column
omit_last	omit the last break column

Value

data.frame or tibble

Examples

```
d <- data.frame(x_n = 3, x_mean = 4,
            y_n = 5, y_mean = 6,
            z_n = 4, z_mean = 4)
# Unquoted variable names
add_break_columns(d, x_mean)

# Character vector
add_break_columns(d, c("y_n", "z_n"),  .before = TRUE)

# Tidyselect function (contains, starts_with, ends_with,
# matches, num_range, all_of, any_of)
# Insert columns after all columns
# ending with "_mean" except the last instance
add_break_columns(d,
                  dplyr::ends_with("_mean"),
                  omit_last = TRUE)
```

add_list_column *Make a column into a list column*

Description

Make a column into a list column

Usage

```
add_list_column(
  data,
  ...,
  type = c("1", "a", "A", "I", "i"),
  sep = ". ",
  merge = FALSE
)
```

Arguments

data	data.frame or tibble
...	Column name or tidyselect function. Select columns. Default is first column
type	list type. Can be "1" (numeric), "a" (lowercase alphabetical), or "ABC" (uppercase alphabetical), "i" (lowercase Roman numerals), "I" (uppercase Roman numerals)
sep	separator
merge	If TRUE, list columns will be united with their respective text columns.

Value

data.frame

Examples

```
d <- data.frame(x = letters[1:5], y = letters[2:6])
# default is first column
add_list_column(d) |>
  apa_flextable()
# select any column
add_list_column(d, y) |>
  apa_flextable()
add_list_column(d, type = "a", sep = ") ") |>
 apa_flextable()
add_list_column(d, merge = TRUE)
```

`add_star_column` *Adds stars next to a column based on p-values*

Description

Adds stars next to a column based on p-values

Usage

```
add_star_column(
  data,
  ...,
  p = "p",
  merge = FALSE,
  superscript = TRUE,
  star = "\\*",
  alpha = c(0.05, 0.01, 0.001),
  first_alpha_marginal = FALSE,
  add_trailing_space = FALSE,
  prefix = "\\"
)
```

Arguments

`data`	data.frame or tibble
`...`	Column name or tidyselect function. Select columns
`p`	Column name or tidyselect function. Select p-value column name
`merge`	merge and balance columns (default: `FALSE`)
`superscript`	make as superscript
`star`	text for making stars
`alpha`	vector of thresholds
`first_alpha_marginal`	if TRUE, the first alpha value is treated as marginal and gets a dagger instead of a star
`add_trailing_space`	if TRUE, adds a trailing space after the stars (default: FALSE)
`prefix`	usually backslashes to prevent markdown from interpreting asterisks as bullets or italics

Value

data.frame

Examples

```
data.frame(b = c(1.4,2.2),
           p = c(.54, .02)) |>
 add_star_column(b, p)
```

`align_chr` *Align text on center text (default is decimal)*

Description

Align text on center text (default is decimal)

Usage

```
align_chr(
  x,
  accuracy = NULL,
  trim_leading_zeros = FALSE,
  drop0trailing = FALSE,
  add_plusses = FALSE,
  padding_character = NULL,
  center = ".",
  format_integers = FALSE,
  side = c("both", "left", "right"),
  NA_value = "",
  format_numeric_character = FALSE,
  ...
)
```

Arguments

`x`	vector (numeric or character)
`accuracy`	number to round to. If NULL, the current default accuracy set with `apa7_defaults()` will be used.
`trim_leading_zeros`	if TRUE (default), trims leading zeros, otherwise keeps them
`drop0trailing`	Drop trailing zeros
`add_plusses`	if TRUE (default), adds a plus to positive numbers
`padding_character`	character to use for padding, default is ` ` (figure space). With the default, text is padded to equal rendered width in the default font family (see `num_pad()` and the `width_aware_padding` argument of `apa7_defaults()`).
`center`	text on which to align text. Center on decimal by default, but can be any text.
`format_integers`	If TRUE, integers will be formatted with digits

side	side on which to make text of equal width
NA_value	value to replace NA
format_numeric_character	format character variables with numeric content
...	additional arguments passed to signs::signs()

Value

character vector

Examples

```
align_chr(c(1, 10, 100))
```

---

apa_anova *Format ANOVA tables in APA style*

Description

Makes an analysis of variance source table with sums of squares, degrees of freedom, mean squares, *F* tests, *p*-values, and effect sizes. Values come from parameters::model_parameters(), and effect sizes come from the effectsize package (e.g., effectsize::eta_squared()).

Usage

```
apa_anova(
  fit,
  anova_parameters = c("Parameter", "Sum_Squares", "df", "Mean_Square", "F", "p",
    "Effect_Size"),
  effect_size = c("eta", "omega", "epsilon"),
  partial = TRUE,
  generalized = FALSE,
  type = NULL,
  ci = 0.95,
  residual_label = "Error",
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  ...
)

## S3 method for class 'lm'
apa_anova(
  fit,
  anova_parameters = c("Parameter", "Sum_Squares", "df", "Mean_Square", "F", "p",
    "Effect_Size"),
  effect_size = c("eta", "omega", "epsilon"),
```

```
  partial = TRUE,
  generalized = FALSE,
  type = NULL,
  ci = 0.95,
  residual_label = "Error",
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  ...
)

## S3 method for class 'aovlist'
apa_anova(
  fit,
  anova_parameters = c("Parameter", "Sum_Squares", "df", "Mean_Square", "F", "p",
    "Effect_Size"),
  effect_size = c("eta", "omega", "epsilon"),
  partial = TRUE,
  generalized = FALSE,
  type = NULL,
  ci = 0.95,
  residual_label = "Error",
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  ...
)

## S3 method for class 'anova'
apa_anova(
  fit,
  anova_parameters = c("Parameter", "Sum_Squares", "df", "Mean_Square", "F", "p",
    "Effect_Size"),
  effect_size = c("eta", "omega", "epsilon"),
  partial = TRUE,
  generalized = FALSE,
  type = NULL,
  ci = 0.95,
  residual_label = "Error",
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  ...
)

## S3 method for class 'afex_aov'
apa_anova(
  fit,
```

```
  anova_parameters = c("Parameter", "df", "df_error", "Mean_Square", "F", "p",
    "Effect_Size"),
  effect_size = c("eta", "omega", "epsilon"),
  partial = TRUE,
  generalized = TRUE,
  type = NULL,
  ci = 0.95,
  residual_label = "Error",
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  ...
)

## Default S3 method:
apa_anova(fit, ...)
```

Arguments

`fit`	an `aov` or `lm` model, a repeated-measures `aov` model with an `Error()` term (class `aovlist`), an `afex` model (class `afex_aov`), or an ANOVA table from `stats::anova()` or `car::Anova()`.
`anova_parameters`	columns to display. `"Effect_Size"` stands for whichever effect size `effect_size`, `partial`, and `generalized` select, and `"CI"` adds its confidence interval as [*LL*, *UL*]. Columns not in the ANOVA table are silently dropped.
`effect_size`	effect size to display: `"eta"` (η^2, default), `"omega"` (ω^2), or `"epsilon"` (ϵ^2). If `NULL`, no effect size is computed.
`partial`	if `TRUE` (default), partial effect sizes (e.g., η_p^2) are displayed. For one-way designs, partial and non-partial effect sizes are the same.
`generalized`	if `TRUE`, generalized η^2 (η_G^2) is displayed instead, which is comparable across between- and within-subjects designs. Only for `effect_size = "eta"`.
`type`	type of sums of squares for `aov` and `lm` models: `1` (sequential, default, as in `stats::anova()`), `2`, or `3`. Types 2 and 3 are computed with `car::Anova()`. Type 3 tests are interpretable only when factors have sum-to-zero contrasts (e.g., `contr.sum`) and covariates are centered. Ignored for ANOVA tables, which already have a type.
`ci`	confidence level for the effect size interval, shown if `anova_parameters` includes `"CI"`. Intervals are two-sided, so a 90% interval excludes 0 when the *F* test is significant at $\alpha = .05$, and its lower bound is 0 otherwise. Intervals are not shown for generalized η^2, because those computed by `effectsize` can exclude the point estimate.
`residual_label`	label for residual (error) rows
`starred`	columns to star with significant *p*-values (e.g., `"F"`)
`bolded`	columns to bold, if significant

column_formats	column_formats object to format columns. If NULL, the default column_formats is used.
...	further arguments passed to parameters::model_parameters()

Details

For repeated-measures models fit with aov(y ~ a * b + Error(id / b)), each error stratum is introduced by a heading row ("Between subjects" or "Within subjects"), and each stratum's error term follows its effects.

Model comparisons (e.g., anova(m1, m2)) are not ANOVA source tables. Use apa_performance_comparison() for them.

For afex models (afex::aov_car(), afex::aov_ez(), afex::aov_4()), each effect has its own error term, so the table shows each effect's degrees of freedom (*df*~effect~, *df*~error~), mean squared error (*MSE*), *F*, *p*, and effect size, as in afex's own nice() tables. Degrees of freedom and *p*-values reflect the sphericity correction set in afex (Greenhouse-Geisser by default), and generalized η^2 is the default effect size, as in afex.

Value

apa7_tbl (tibble)

Examples

```
d <- mtcars
d$cyl <- factor(d$cyl)
d$am <- factor(d$am)
fit <- aov(mpg ~ cyl * am, data = d)
apa_anova(fit)

# Omega squared with a 90% confidence interval
apa_anova(
  fit,
  anova_parameters = c(
    "Parameter", "Sum_Squares", "df", "Mean_Square", "F", "p",
    "Effect_Size", "CI"
  ),
  effect_size = "omega",
  ci = .90
)

# Type 2 sums of squares (requires the car package)
if (requireNamespace("car", quietly = TRUE)) {
  apa_anova(fit, type = 2)
}
if (requireNamespace("afex", quietly = TRUE)) {
  data(obk.long, package = "afex")
  fit_afex <- afex::aov_ez(
    "id", "value", obk.long,
    between = "treatment", within = "phase", fun_aggregate = mean
  )
  apa_anova(fit_afex)
```

```
}
```

---

apa_chisq *Make contingency table with chi-square test of independence*

Description

Make contingency table with chi-square test of independence

Usage

```
apa_chisq(
  data,
  note = NULL,
  row_title_column = NULL,
  row_title_prefix = "",
  row_title_sep = " ",
  row_title_align = "center",
  row_title_border = list(color = "gray20", style = "solid", width = 1),
  left_column_padding = 20,
  cwidth = 0.75,
  cheight = 0.25,
  separate_headers = TRUE,
  apa_style = TRUE,
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  horizontal_padding = 3,
  table_align = "left",
  layout = "autofit",
  table_width = 1,
  markdown = TRUE,
  markdown_header = markdown,
  markdown_body = markdown,
  auto_format_columns = TRUE,
  column_formats = NULL,
  pretty_widths = TRUE,
  suppress_warnings = TRUE,
  ...
)
```

Arguments

data	A two-column data.frame or tibble

note Custom note (overrides automatic note.)

row_title_column
Column name or tidyselect function. column to group rows

row_title_prefix
text to be added to each title

row_title_sep separator for prefix

row_title_align
alignment of row title ("left", "center", "right")

row_title_border
list of flextable styles

left_column_padding
Number of points the left column is padded (only relevant when there is a row_title_column and row_title_align = "left")

cwidth initial cell width in inches

cheight initial cell height in inches

separate_headers
separate header rows (default: TRUE)

apa_style apply apa_style function (default: TRUE)

font_family font family

font_size font size

text_color text color

border_color border color

border_width border width in pixels

line_spacing spacing between lines

horizontal_padding
horizontal padding (in pixels)

table_align table alignment ("left", "center", "right")

layout table layout ("autofit", "fixed")

table_width table width (in pixels, 0 for auto)

markdown apply markdown formatting to header and body

markdown_header
apply markdown formatting to header

markdown_body apply markdown formatting to body

auto_format_columns
if true, will attempt to format some columns automatically

column_formats a column_formats object

pretty_widths apply pretty_widths function

suppress_warnings
Suppress any warnings if true.

... arguments passed to apa_style

Value

flextable::flextable

Examples

```
apa_chisq(mtcars[, c("am", "gear")])
```

---

apa_cor *APA-formatted correlation table*

Description

APA-formatted correlation table

Usage

```
apa_cor(
  data,
  note = NULL,
  p_value = c(0.05, 0.01, 0.001),
  digits = 2,
  bold_significant = FALSE,
  star_significant = TRUE,
  significance_note = TRUE,
  output = c("flextable", "tibble"),
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  table_width = 6.5,
  keep_empty_star_columns = TRUE,
  summary_functions = list(M = mean, SD = stats::sd),
  column_formats = NULL,
  ...
)
```

Arguments

data	data.frame or tibble with variables to be
note	Custom note to appear below table. (Overrides automatic note.)
p_value	p-value needed to be flagged as significant
digits	Number of digits for rounding
bold_significant	bold significant correlations

star_significant
: start significant correlations

significance_note
: If TRUE, place note at bottom of table that significant correlations are bolded.

output
: output type. Can be "flextable" or "tibble"

font_family
: font family

font_size
: font size

text_color
: text color

border_color
: border color

border_width
: border width in pixels

line_spacing
: spacing between lines

table_width
: table width (in pixels, 0 for auto)

keep_empty_star_columns
: Keep remove empty star columns (Default: TRUE)

summary_functions
: Any named list of functions that summarize data columns (e.g., list(Mean = mean, SD = sd)). Can also be a character vector of function names (e.g., c("n", "M", "SD")). Functions available in a character vector: IQR, Interquartile Range, Kurtosis, MAD, Median Absolute Deviation, M, Mean, Med, Median, n, N, Quantile, Range, SD, Skewness, Var, Variance

column_formats
: column_formats object

...
: <data-masking> parameters passed to psych::corTest

Value

flextable::flextable

Examples

```
apa_cor(mtcars[, c("mpg", "am", "gear", "carb")])
apa_cor(mtcars[, c("mpg", "am", "gear", "carb")],
        output = "tibble",
        star_significant = FALSE)
```

---

apa_factor_cor *Factor correlation table*

Description

Makes an APA-style table of the correlations among factors (or components) from an obliquely rotated solution: a psych::fa() or psych::principal() object, or the output of parameters::factor_analysis() or parameters::principal_components().

Usage

```
apa_factor_cor(fit, factor_names = NULL, digits = 2)
```

Arguments

fit	a psych::fa, psych::principal, parameters::factor_analysis, or parameters::principal_comp object
factor_names	names for the factors. Either an unnamed character vector in the order the factors appear in the table (psych orders factors by variance explained), or a named vector that maps the original names to new names (e.g., c(PA1 = "Verbal")). The unnamed form is safer: after an oblimin rotation, the labels psych assigns (PA1, PA2, ...) can change from one run to the next, but the order of the factors does not.
digits	number of decimal places

Details

The factors are numbered in the first column, and the numbers are used as column headers. Only the lower triangle is shown. The factors are in the same order as in apa_loadings().

Value

apa7_tbl (tibble)

Examples

```
fit <- psych::fa(Harman74.cor$cov, 4, fm = "pa", rotate = "oblimin")
apa_factor_cor(
  fit,
  factor_names = c("Verbal", "Spatial", "Memory", "Speed")
)
```

---

apa_flextable *Convert data to flextable consistent with APA style*

Description

The apa_flextable function performs a number of formatting operations on the data before and after the data are sent to flextable. See Details.

Usage

```
apa_flextable(
  data,
  row_title_column = NULL,
  row_title_align = "left",
  row_title_prefix = "",
```

```
  row_title_sep = " ",
  row_title_border = list(color = "gray20", style = "solid", width = 1),
  left_column_padding = 20,
  col_keys = colnames(data),
  cwidth = 0.75,
  cheight = 0.25,
  header_align_vertical = c("top", "middle", "bottom"),
  separate_headers = TRUE,
  apa_style = TRUE,
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  horizontal_padding = 3,
  table_align = "left",
  layout = "autofit",
  table_width = 1,
  markdown = TRUE,
  markdown_header = markdown,
  markdown_body = markdown,
  no_markdown_columns = NULL,
  no_markdown_columns_header = NULL,
  no_format_columns = NULL,
  auto_format_columns = TRUE,
  column_formats = NULL,
  pretty_widths = TRUE,
  add_breaks_between_spanners = TRUE,
  note = NULL,
  ...
)
```

Arguments

`data`	data.frame or tibble
`row_title_column`	Column name or tidyselect function. column to group rows
`row_title_align`	alignment of row title ("left", "center", "right")
`row_title_prefix`	text to be added to each title
`row_title_sep`	separator for prefix
`row_title_border`	list of flextable styles
`left_column_padding`	Number of points the left column is padded (only relevant when there is a `row_title_column` and `row_title_align = "left"`)

col_keys column keys passed to flextable (defaults data column names)

cwidth initial cell width in inches

cheight initial cell height in inches

header_align_vertical
vertical alignment of headers. Can be "top", "middle", or "bottom"

separate_headers
separate header rows (default: TRUE)

apa_style apply apa_style function (default: TRUE)

font_family font family

font_size font size

text_color text color

border_color border color

border_width border width in pixels

line_spacing spacing between lines

horizontal_padding
horizontal padding (in pixels)

table_align table alignment ("left", "center", "right")

layout table layout ("autofit", "fixed")

table_width table width (in pixels, 0 for auto)

markdown apply markdown formatting to header and body

markdown_header
apply markdown formatting to header

markdown_body apply markdown formatting to body

no_markdown_columns
body columns that should not be treated as markdown

no_markdown_columns_header
column headers that should not be treated as markdown

no_format_columns
Column name or tidyselect function. selected columns are not formatted. Columns named in a "no_format_columns" attribute of data (set by functions such as apa_loadings()) are not formatted either. Likewise, a rule is drawn above the first body row whose first-column text matches a "rule_above" attribute of data.

auto_format_columns
if true, will attempt to format some columns automatically

column_formats a column_formats object

pretty_widths apply pretty_widths function

add_breaks_between_spanners
add breaks between spanners if TRUE

note — table note placed below the table, after "*Note*.". If TRUE, the note is written by apa_note() from what a table function such as apa_anova(), apa_parameters(), or apa_loadings() recorded. A character string is used as the note (markdown is allowed), and NULL or FALSE (default) adds no note. Use this for documents that are not made with apaquarto. With apaquarto, use the apa-note chunk option instead (e.g., #| apa-note: !expr apa7::apa_note(tbl)), so that the note is placed and styled by apaquarto and is not added twice.

... — arguments passed to apa_style

Details

Roughly speaking, apa_flextable performs these operations by default:

1. Apply as_grouped_data and restructure row titles, if row_title is specified.
2. Format data with apa_format_columns if auto_format_columns = TRUE
3. Separate headers into multiple header rows if separate_headers = TRUE
4. Apply flextable::flextable
5. Apply flextable::surround to make borders to separate row groups, if any.
6. Apply the apa_style function (table formatting and markdown conversion) if apa_style = TRUE
7. Apply pretty_widths if pretty_widths = TRUE

Value

flextable::flextable

Examples

```
library(dplyr)
library(tidyr)
library(flextable)
mtcars %>%
  dplyr::select(vs, am, gear, carb) |>
  tidyr::pivot_longer(-vs,  names_to = "Variable") |>
  dplyr::summarise(Mean = round(mean(value), 2),
                   SD = round(sd(value), 2),
                   .by = c(Variable,vs)) |>
  dplyr::mutate(vs = factor(vs, levels = 0:1, labels = c("Automatic", "Manual"))) |>
  apa_flextable(row_title_column= vs,  row_title_align = "center") |>
  align(j = 2:3, align = "center")
```

apa_format_columns *Format data columns*

Description

Format data columns

Usage

```
apa_format_columns(
  data,
  column_formats = NULL,
  no_format_columns = NULL,
  rename_headers = TRUE,
  latex_headers = FALSE,
  format_separated_headers = TRUE,
  sep = "_",
  accuracy = NULL
)
```

Arguments

data	data set (data.frame or tibble)
column_formats	column_formats object. If NULL, the current default formatter set with apa7_defaults() will be used.
no_format_columns	Column name or tidyselect function. selected columns are not formatted
rename_headers	if TRUE, rename headers with markdown or latex
latex_headers	if TRUE, rename headers with latex instead of markdown
format_separated_headers	if TRUE, format headers with separated names. For example, if the formatter formats column R2 as *R*^2^, then Model 1_R2 becomes Model 1_*R*^2^)
sep	separator for separated headers (default is "_")
accuracy	numeric (default: NULL, uses the current default accuracy set with apa7_defaults()). If not NULL, sets the accuracy for the formatter.

Value

tibble

Examples

```
lm(mpg ~ cyl + wt, data = mtcars) |>
  parameters::parameters() |>
  apa_format_columns() |>
  apa_flextable()
```

apa_loadings *Factor loadings table*

Description

Makes an APA-style table of factor loadings from an exploratory factor analysis or a principal components analysis.

Usage

```
apa_loadings(
  fit,
  sort_loading = TRUE,
  min_loading = 0.2,
  factor_names = NULL,
  variance = FALSE,
  communality = FALSE,
  uniqueness = FALSE,
  complexity = FALSE,
  bold_max = FALSE,
  format_variable_names = TRUE,
  digits = 2
)
```

Arguments

fit	a psych::fa, psych::principal, parameters::factor_analysis, or parameters::principal_comp object
sort_loading	sort the rows so that variables are grouped by the factor with their largest loading, and by the size of that loading within each factor (as in psych::fa.sort())
min_loading	minimum absolute loading to display. Use NA or 0 to show all loadings.
factor_names	names for the factors. Either an unnamed character vector in the order the factors appear in the table (psych orders factors by variance explained), or a named vector that maps the original names to new names (e.g., c(PA1 = "Verbal")). The unnamed form is safer: after an oblimin rotation, the labels psych assigns (PA1, PA2, ...) can change from one run to the next, but the order of the factors does not.
variance	add rows for the variance explained by each factor
communality	add a communality (h^2^) column
uniqueness	add a uniqueness (u^2^) column
complexity	add a column with Hofmann's complexity index
bold_max	bold the largest displayed loading (in absolute value) in each row
format_variable_names	format the variable names with the Variable column format. Set to FALSE to keep item text exactly as it is.
digits	number of decimal places for loadings

Details

`fit` can be a `psych::fa()` or `psych::principal()` object, or the output of `parameters::factor_analysis()` or `parameters::principal_components()`. For the parameters functions, the loadings come from the underlying model, so their `sort` and `threshold` settings do not matter: use `sort_loading` and `min_loading` instead. Unrotated `parameters::principal_components()` output wraps a `stats::prcomp()` model. Its loadings are the correlations between the variables and the components when the variables are standardized (the default, `standardize = TRUE`).

Loadings smaller in absolute value than `min_loading` are left blank. With `bold_max = TRUE`, the largest displayed loading in each row is bolded.

Rows for variance explained (`variance = TRUE`) are added below the loadings: the sum of squared loadings, the proportion of total variance, and the cumulative proportion (the last only when there is more than one factor). `apa_flextable()` draws a rule above these rows. With an oblique rotation the factors are correlated, so the variance explained by each factor overlaps with the others and the cumulative proportion is only approximate. Report the factor correlations with `apa_factor_cor()`.

The variable names are formatted (e.g., title case) when `format_variable_names = TRUE`. Either way, `apa_flextable()` leaves the `Variable` column as it is, so item text is never changed again when the table is rendered.

Value

`apa7_tbl` (tibble)

Examples

```
fit <- psych::fa(Harman74.cor$cov, 4, fm = "pa", rotate = "varimax")
apa_loadings(
  fit,
  factor_names = c("Verbal", "Spatial", "Speed", "Memory")
)

# With rows for variance explained and bolded primary loadings
apa_loadings(fit, variance = TRUE, communality = TRUE, bold_max = TRUE)

# Principal components
psych::principal(Harman74.cor$cov, 4) |>
  apa_loadings()

# parameters output
parameters::principal_components(mtcars[, 1:7], n = 2) |>
  apa_loadings()
```

---

apa_note *Write a table note*

Description

Makes the text of an APA-style table note from what a table function recorded about how the table was made. The text is a starting point: edit it or add to it as needed.

Usage

```
apa_note(x, collapse = " ")
```

Arguments

x	a table made by apa_anova(), apa_parameters(), apa_loadings(), or apa_factor_cor()
collapse	string that joins the sentences. Use NULL to get a character vector with one sentence per element.

Details

For apa_anova() tables, the note can say

- which type of sums of squares was used,
- which effects have sphericity-corrected degrees of freedom (for afex models, only effects involving a within-subjects factor with more than two levels are corrected),
- what the effect size (and *MSE*, for afex models) stands for, and
- what stars or bold values mean.

For apa_loadings() tables, the note can say

- how the factors were extracted and rotated (e.g., "Principal axis factoring with oblimin rotation"), and that loadings after an oblique rotation are pattern coefficients,
- which loadings are omitted (min_loading) or bolded (bold_max),
- what the *h*^2^, *u*^2^, and Complexity columns stand for, and
- that the cumulative proportion of variance is approximate when the factors are correlated.

For apa_factor_cor() tables, the note says how the factors were extracted and rotated.

For apa_parameters() tables of regression models (lm, glm, polr, clm, and lme4 models, or a list of models), the note can say

- which robust standard errors were used (e.g., vcov = "HC3"),
- how the degrees of freedom of mixed-model *t* tests were approximated (Kenward-Roger or Satterthwaite),
- that ordinal-model thresholds are not tested,
- what the standardized coefficient (β), exponentiated coefficient (e.g., *OR*), confidence interval, and model-comparison (e.g., ΔR^2) columns stand for, and
- what stars or bold values mean.

Abbreviations are defined from the abbreviation slot of each column's column_format(), so a note defines every abbreviated header that has a definition. To define another header, or to change a definition, set the slot in the column_formats passed to the table function (or set the default with apa7_defaults()):

```
cf <- column_formats()
cf$Sum_Squares@abbreviation <- "sum of squares"
apa_anova(fit, anova_parameters = c("Parameter", "Sum_Squares", "df", "F", "p"),
          column_formats = cf)
```

The sentences are in APA order: general notes first, then probability notes. The text does not start with "*Note*." because apaquarto adds it. With apaquarto, make the table in one chunk and display it in another, using the note as the `apa-note` chunk option. Write `apa7::apa_note()` there, because in documents with cached chunks, chunk options can be evaluated before `library(apa7)` takes effect:

````
```{r}
#| label: anova-fit
tbl <- apa_anova(fit, type = 3)
```

```{r}
#| label: tbl-anova
#| tbl-cap: Analysis of Variance
#| apa-note: !expr apa7::apa_note(tbl)
tbl
```
````

The information is stored in an `"apa_note"` attribute of the table. dplyr functions such as `mutate()` and `filter()` keep it, but some operations (e.g., `bind_rows()` and joins) drop it, so call `apa_note()` on the table that the table function returned.

Value

character

Examples

```
d <- mtcars
d$cyl <- factor(d$cyl)
d$am <- factor(d$am)
fit <- aov(mpg ~ cyl * am, data = d)
tbl <- apa_anova(fit, starred = "F")
apa_note(tbl)

if (requireNamespace("afex", quietly = TRUE)) {
  data(obk.long, package = "afex")
  fit_afex <- afex::aov_ez(
    "id", "value", obk.long,
    between = "treatment", within = "phase", fun_aggregate = mean
  )
  apa_note(apa_anova(fit_afex))
}

# Factor analysis
fit_fa <- psych::fa(Harman74.cor$cov, 4, fm = "pa", rotate = "varimax")
apa_loadings(fit_fa, min_loading = .3, communality = TRUE) |>
  apa_note()

# Regression
lm(mpg ~ cyl + wt, data = mtcars) |>
```

```
apa_parameters(starred = "p") |>
apa_note()
```

---

apa_p *p-value in APA format*

Description

p-value in APA format

Usage

```
apa_p(
  p,
  inline = FALSE,
  markdown = TRUE,
  min_digits = 2,
  max_digits = 3,
  align = FALSE
)
```

Arguments

p	probability
inline	If TRUE (default), returns statistic (e.g., p = .04), otherwise just the number (e.g., .04)
markdown	By default, outputs text compatible with markdown if TRUE, otherwise prints plain text compatible with latex.
min_digits	minimum number of digits to round to. Default is 2.
max_digits	maximum number of digits to round to. Default is 3.
align	decimal alignment if TRUE

Value

character vector

Examples

```
# Values less than .001 are <.001
apa_p(.0002)
# Values between .001 and .01 are rounded to 3 digits
apa_p(.002)
# Values between .01 and .995 are rounded to 2 digits
apa_p(.02)#'
apa_p(.22)
apa_p(.994)
# Values above .995 are >.99
```

```
apa_p(.999)
# Rounding to 3 digits
apa_p(.2341, min_digits = 3)
apa_p(.0123, min_digits = 3)
apa_p(.00123, min_digits = 3)
apa_p(.000123, min_digits = 3)
apa_p(.991, min_digits = 3)
apa_p(.9991, min_digits = 3)
apa_p(.9995, min_digits = 3)
```

`apa_p_star_note` *Make star notes for p-values*

Description

Make star notes for p-values

Usage

```
apa_p_star_note(x = c(0.05, 0.01, 0.001), first_alpha_marginal = FALSE)
```

Arguments

`x`	vector of alpha values (p-value thresholds)
`first_alpha_marginal`	if TRUE, the first alpha value is treated as marginal and gets a dagger instead of a star

Value

character vector

Examples

```
apa_p_star_note()
apa_p_star_note(x = c(.10, .05, .01, .001), first_alpha_marginal = TRUE)
```

apa_parameters.aov *format model parameters in APA style*

Description

format model parameters in APA style

Usage

```
## S3 method for class 'aov'
apa_parameters(
  fit,
  predictor_parameters = NULL,
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  ...
)

## S3 method for class 'aovlist'
apa_parameters(
  fit,
  predictor_parameters = NULL,
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  ...
)

## S3 method for class 'afex_aov'
apa_parameters(
  fit,
  predictor_parameters = NULL,
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  ...
)

apa_parameters(
  fit,
 predictor_parameters = c("Coefficient", "SE", "Std_Coefficient", "t", "df_error", "p"),
  starred = NULL,
  bolded = NULL,
```

```
  column_formats = NULL,
  t_with_df = TRUE,
  ...
)

## S3 method for class 'lm'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Std_Coefficient", "t",
    "df_error", "p"),
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  ...
)

## S3 method for class 'glm'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Exp_Coefficient", "t", "z",
    "df_error", "p"),
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  exponentiate = NULL,
  ...
)

## S3 method for class 'polr'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Exp_Coefficient", "z", "p"),
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  exponentiate = NULL,
  thresholds = TRUE,
  ...
)

## S3 method for class 'clm'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Exp_Coefficient", "z", "p"),
  starred = NA,
```

```
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  exponentiate = NULL,
  thresholds = TRUE,
  ...
)

## S3 method for class 'merMod'
apa_parameters(
  fit,
 predictor_parameters = c("Parameter", "Coefficient", "SE", "Exp_Coefficient", "t", "z",
    "df_error", "p"),
  starred = NA,
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  exponentiate = NULL,
  random = TRUE,
  ...
)

## S3 method for class 'list'
apa_parameters(
  fit,
  predictor_parameters = NULL,
  starred = "deltaR2",
  bolded = NA,
  column_formats = NULL,
  t_with_df = TRUE,
  performance_parameters = c("R2", "deltaR2"),
  ...
)
```

Arguments

`fit`	model fit object
`predictor_parameters`	predictor parameters to display. If named vector, column names will be vector names. Columns not present in the model output (e.g., `t` for a model tested with *z*) are silently dropped. For lists of models, `NULL` (default) uses the default for each model's class. Add `"CI"` to show confidence intervals as [*LL*, *UL*] (headed, e.g., "95% CI"), and, for `glm`, `polr`, and `clm` models, `"Exp_CI"` to show confidence intervals of exponentiated coefficients (headed, e.g., "95% CI for *OR*"). Confidence intervals are not shown by default.
`starred`	columns to star with significant p_values
`bolded`	columns to bold, if significant

`column_formats` column_formats object to format columns. If NULL, the default column_formats is used.

`t_with_df` if TRUE, the t column will be displayed with degrees of freedom in parentheses. If FALSE, only the t value is displayed.

`...` arguments passed to methods. For `lm`, `glm`, `polr`, and `clm` models, further arguments are passed to `parameters::model_parameters()`. For example, `vcov = "HC3"` gives heteroskedasticity-consistent (robust) standard errors, and the test statistics, *p*-values, and confidence intervals are based on them. `ci = .90` sets the confidence level.

`exponentiate` if `NULL` (default), exponentiated coefficients are added when the model's link function is logit or log (i.e., odds ratios, risk ratios, or incidence rate ratios). If `TRUE`, exponentiated coefficients are always added. If `FALSE`, they are never added.

`thresholds` for ordinal models, if `TRUE` (default), threshold (cutpoint) rows are displayed below the predictors.

`random` for mixed models, if `TRUE` (default), random-effect standard deviations and correlations (and the residual standard deviation, for linear mixed models) are displayed below the fixed effects.

`performance_parameters` for lists of models, model comparison metrics passed to `apa_performance_comparison()`. If `NULL`, no metrics are added. By default, `R2` and `deltaR2`; for lists of mixed models (`lme4`), the default also includes the conditional R^2 and its change (`R2_conditional`, `deltaR2_conditional`), and `R2` is the marginal R^2.

Details

For `aov` models (including repeated-measures models with an `Error()` term) and `afex` models, `apa_parameters()` returns the ANOVA table from `apa_anova()`. `predictor_parameters` and `t_with_df` are ignored, and other arguments in `...` (e.g., `effect_size`, `type`) are passed to `apa_anova()`. To display regression coefficients instead, refit the model with `stats::lm()`.

For `glm` objects (including `MASS::glm.nb`), the test statistic is z for families with a fixed dispersion parameter (binomial, Poisson, negative binomial) and t for families with an estimated dispersion parameter (Gaussian, Gamma, inverse Gaussian, quasi families). Whichever statistic applies is displayed. The exponentiated coefficient column (`Exp_Coefficient`) is labeled *OR* for logit links, *IRR* for log links in count models (Poisson, quasi-Poisson, negative binomial), *RR* for log links in binomial models, and Exp(*B*) otherwise.

For ordinal regression models (`MASS::polr` and `ordinal::clm`), coefficients are tested with Wald z tests. (`parameters` reports t for `polr`, but the test is asymptotically normal, and `clm` reports z for the same model.) With a logit link, exponentiated coefficients are cumulative odds ratios, labeled *OR*. Threshold rows show only the estimate, its standard error, and its confidence interval, because their exponentiated values are not odds ratios, and testing whether a threshold equals 0 is not meaningful. Confidence intervals (`"CI"`, `"Exp_CI"`) are Wald z intervals, consistent with the z tests.

For mixed models fit with `lme4` (`lmer` and `glmer`), fixed effects are displayed as for `lm` and `glm` models. For linear mixed models, t tests and confidence intervals use Kenward-Roger degrees of freedom by default (`ci_method = "kenward"`, which requires the pbkrtest package); pass another `ci_method` (e.g., `"satterthwaite"`, which requires lmerTest, or `"wald"`) to change this. When the

degrees of freedom differ across fixed effects, they are shown in a *df* column rather than in the *t* column header. Random-effect rows show only the estimated standard deviation (*SD*) or correlation (*r*), labeled by grouping factor.

Value

tibble

Examples

```
lm(mpg ~ cyl + wt, data = mtcars) |>
   apa_parameters() |>
   apa_flextable()

# Confidence intervals with robust (HC3) standard errors
if (requireNamespace("sandwich", quietly = TRUE)) {
  lm(mpg ~ cyl + wt, data = mtcars) |>
    apa_parameters(
    predictor_parameters = c("Parameter", "Coefficient", "SE", "CI", "t", "df_error", "p"),
      vcov = "HC3"
    ) |>
    apa_flextable()
}
glm(am ~ wt + hp, data = mtcars, family = binomial) |>
   apa_parameters() |>
   apa_flextable()

if (requireNamespace("MASS", quietly = TRUE) &&
    requireNamespace("carData", quietly = TRUE)) {
  MASS::polr(poverty ~ gender + age, data = carData::WVS, Hess = TRUE) |>
    apa_parameters()
}


if (requireNamespace("lme4", quietly = TRUE) &&
    requireNamespace("pbkrtest", quietly = TRUE)) {
  lme4::lmer(Reaction ~ Days + (Days | Subject), data = lme4::sleepstudy) |>
    apa_parameters()
}
```

---

apa_performance *format model performance metrics in APA style*

Description

format model performance metrics in APA style

Usage

```
apa_performance(fit, metrics = c("R2", "Sigma"), column_formats = NULL)

## S3 method for class 'lm'
apa_performance(fit, metrics = c("R2", "Sigma"), column_formats = NULL)

## S3 method for class 'glm'
apa_performance(fit, metrics = c("R2", "AIC"), column_formats = NULL)

## S3 method for class 'polr'
apa_performance(fit, metrics = c("R2", "AIC"), column_formats = NULL)

## S3 method for class 'clm'
apa_performance(fit, metrics = c("R2", "AIC"), column_formats = NULL)

## S3 method for class 'merMod'
apa_performance(
  fit,
  metrics = c("R2_marginal", "R2_conditional", "ICC"),
  column_formats = NULL
)

## S3 method for class 'list'
apa_performance(fit, metrics = NULL, column_formats = NULL)
```

Arguments

`fit`	model fit object
`metrics`	performance metrics. Default is R2 and Sigma
`column_formats`	column_formats object to format columns. If NULL, the default column_formats is used.

Details

For `glm` objects, `R2` is the pseudo-*R*^2^ that `performance::r2()` selects for the model family (e.g., Tjur's *R*^2^ for logistic regression and Nagelkerke's *R*^2^ for count models). The header names the pseudo-*R*^2^ that was used. Because the residual standard deviation is not meaningful for most GLMs, the default metrics are `R2` and `AIC`.

For mixed models (`lme4::lmer` and `lme4::glmer`), the default metrics are the marginal *R*^2^ (variance explained by the fixed effects), the conditional *R*^2^ (fixed and random effects), and the intraclass correlation (ICC), as computed by `performance::model_performance()`.

For a list of models, each model's metrics are displayed in a row, labeled in a `Model` column by the list's names (or "Model 1", "Model 2", ... for unnamed elements). If `metrics` is `NULL` (default), each model's default metrics are used.

Value

tibble

Examples

```
lm(mpg ~ cyl + wt, data = mtcars) |>
   apa_performance() |>
   apa_flextable()
list(
  `Cylinders` = lm(mpg ~ cyl, data = mtcars),
  `Cylinders and weight` = lm(mpg ~ cyl + wt, data = mtcars)
) |>
  apa_performance()
```

---

apa_performance_comparison

format model comparison metrics in APA style

Description

format model comparison metrics in APA style

Usage

```
apa_performance_comparison(
  ...,
  metrics = NULL,
  starred = NA,
  column_formats = NULL
)
```

Arguments

`...`	model fit objects, or a list of them. Models are labeled by the list's names (or the argument names), and "Model 1", "Model 2", ... for unnamed models.
`metrics`	performance metrics. If `NULL` (default), R2, deltaR2, F, and p for models with an estimated dispersion parameter (e.g., `lm`, Gaussian or quasi-family `glm`), and R2, deltaR2, Chi2, df, and p for GLMs with a fixed dispersion parameter (binomial, Poisson, negative binomial). Use "all" for all available metrics.
`starred`	columns to star with significant p_values
`column_formats`	column_formats object to format columns. If NULL, the default column_formats is used.

Details

Nested models are compared with *F* tests (`performance::test_wald()`) unless every model is a GLM with a fixed dispersion parameter, in which case likelihood ratio tests (`performance::test_likelihoodratio()`) are used. For GLMs, `R2` is the pseudo-*R*^2^ that `performance::r2()` selects for the model family, and the header names it (e.g., *R*^2^~Tjur~).

Mixed models (`lme4::lmer` and `lme4::glmer`) are compared with likelihood ratio tests. `R2` is the marginal *R*^2^ (variance explained by the fixed effects), and by default the conditional *R*^2^ (fixed and random effects) and its change (`R2_conditional`, `deltaR2_conditional`) are shown too, because adding random effects changes the conditional but not the marginal *R*^2^.

Value

tibble

Examples

```
m1 <- lm(mpg ~ cyl, data = mtcars)
m2 <- lm(mpg ~ cyl + wt, data = mtcars)
apa_performance_comparison(list(`Model 1` =m1, `Model 3` =m2)) |>
   apa_flextable()
```

---

apa_style *Style* `flextable::flextable` *object according to APA style*

Description

Style `flextable::flextable` object according to APA style

Usage

```
apa_style(
  x,
  font_family = NULL,
  font_size = 12,
  text_color = "black",
  border_color = "black",
  border_width = 0.5,
  line_spacing = 2,
  horizontal_padding = 3,
  table_align = "left",
  header_align_vertical = c("top", "middle", "bottom"),
  layout = "autofit",
  table_width = 0,
  markdown = TRUE,
  markdown_header = markdown,
  markdown_body = markdown,
  no_markdown_columns = NULL,
  no_markdown_columns_header = no_markdown_columns,
  separate_headers = TRUE
)
```

Arguments

x	object
font_family	font family
font_size	font size
text_color	text color
border_color	border color
border_width	border width in pixels
line_spacing	spacing between lines
horizontal_padding	horizontal padding (in pixels)
table_align	table alignment ("left", "center", "right")
header_align_vertical	vertical alignment of headers. Can be "top", "middle", or "bottom"
layout	table layout ("autofit", "fixed")
table_width	table width (in pixels, 0 for auto)
markdown	apply markdown formatting to header and body
markdown_header	apply markdown formatting to header
markdown_body	apply markdown formatting to body
no_markdown_columns	body columns that should not be treated as markdown
no_markdown_columns_header	column headers that should not be treated as markdown
separate_headers	separate headers into column spanner labels

Value

object

Examples

```
d <- data.frame(x = 1:3, y = 4:6)
flextable::flextable(d) |>
  apa_style()
```

apa7_defaults *Set defaults for apa7 package*

Description

Set defaults for apa7 package

Usage

```
apa7_defaults(
  accuracy = NULL,
  font_family = NULL,
  intercept_text = NULL,
  column_formats = NULL,
  number_formatter = NULL,
  trim_leading_zero = NULL,
  width_aware_padding = NULL,
  reset = FALSE
)
```

Arguments

accuracy	numeric (default: .01)
font_family	font family
intercept_text	what to call the intercept
column_formats	column formatting functions
number_formatter	default function to format numbers
trim_leading_zero	default function to trim leading zeros from numbers
width_aware_padding	if TRUE (default), align_chr() pads text to equal rendered width in font_family (when the font is installed). If FALSE, text is padded with one figure space per missing character.
reset	if TRUE, reset all defaults (except as specified)

Value

previous defaults

Examples

```
apa7_defaults(accuracy = .001)
# Reset to package defaults
apa7_defaults(reset = TRUE)
```

apa7_tbl *Tables ready for* apa_flextable

Description

The tibbles returned by apa_anova(), apa_factor_cor(), apa_loadings(), apa_parameters(), apa_performance(), and apa_performance_comparison() have class apa7_tbl. They are ordinary tibbles for post-processing with dplyr, but in knitr and Quarto documents they are rendered as an apa_flextable() automatically. In the console they print as tibbles.

Usage

```
## S3 method for class 'apa7_tbl'
print(x, ...)

## S3 method for class 'apa7_tbl'
knit_print(x, ...)
```

Arguments

x	an apa7_tbl
...	arguments passed to the next method

Details

The apa7.print option controls this behavior:

- "auto" (default): flextable in knitr/Quarto, tibble in the console.
- "flextable": flextable everywhere.
- "tibble": tibble everywhere.

To render with non-default settings, pass the table to apa_flextable() explicitly, which always returns a flextable.

Value

x, invisibly, for print. A knit_asis or flextable output for knit_print.

Examples

```
fit <- lm(mpg ~ cyl + wt, data = mtcars)
# Prints as a tibble in the console
apa_parameters(fit)

# It is still a tibble, so it can be post-processed
apa_parameters(fit) |>
  dplyr::filter(Variable != "Constant") |>
  apa_flextable()
```

column_format *Column format class*

Description

This class is used to define the format of columns in tables, including the name, header, latex representation, a formatter function, and what the header abbreviates.

Usage

```
column_format(
  name = character(0),
  header = character(0),
  latex = character(0),
  formatter = function() NULL,
  abbreviation = character(0)
)
```

Arguments

name	name of column
header	markdown representation of header name
latex	latex representation of header name
formatter	function that formats the column values. It should take a vector of values and return a character vector of formatted values.
abbreviation	what the header stands for, used by apa_note() to define the abbreviations in a table's note (e.g., "standardized coefficient" defines the β header as "β = standardized coefficient"). The term defined is the header, unless the string is named (e.g., c(CI = "confidence interval") defines "CI" for a header such as "95% CI"). Leave empty (the default) for symbols that need no definition, such as *M*, *SD*, *t*, and *p*.

Value

column_format object

Examples

```
R2 <- column_format(
         "R2",
         header = "*R*^2^",
         latex = "$R^2$",
         formatter = \(x, accuracy = the$accuracy, ...) {
                      align_chr(x,
                                accuracy = accuracy,
                                trim_leading_zeros = TRUE,
                                ...)
```

```
                      })
R2
R2@header
R2@formatter

# Define an abbreviation in table notes
SS <- column_formats()$Sum_Squares
SS@abbreviation <- "sum of squares"
```

`column_formats` *Create a set of column formats*

Description

Returns an S7 object that contains a list of `column_format` objects that can be used to format parameters in APA style.

Usage

```
column_formats(
  .data = NULL,
  accuracy = NULL,
  intercept_text = NULL,
  starred_columns = character(0),
  variable_labels = character(0),
  custom_columns = NULL
)
```

Arguments

`.data`	list of `column_format` objects
`accuracy`	numeric (passed to scales::number)
`intercept_text`	describe intercept
`starred_columns`	which columns get p-value stars
`variable_labels`	named vector of variable names (with vector names as labels). For example, c(`Parental Income` = "parental_income", `Number of Siblings` = "n_siblings")
`custom_columns`	named list of column_formats to add or replace existing columns

Value

column_formats

Slots

`get_column_names` getter for column names

`get_headers` getter for column headers

`get_latex` getter for column latex headers

`get_formatters` getter for column formatters

`get_header_rename` getter for column names with headers as names

`get_header_rename_latex` getter for column names with latex headers as names

`get_tibble` getter for tibble with column names, headers, latex headers, and formatters

`get_abbreviations` getter for the abbreviations defined by the column formats, as a character vector of definitions named by the term defined

Examples

```
my_formatter <- column_formats()
my_formatter$Coefficient@formatter <- \(x) round(x, 2)
my_formatter$Coefficient@formatter(2.214)
```

---

column_spanner_label *Prepend column spanner labels to data column labels*

Description

Prepend column spanner labels to data column labels

Usage

```
column_spanner_label(data, label, ..., relocate = TRUE)
```

Arguments

`data`	data.frame or tibble
`label`	character of column spanner
`...`	columns (i.e., one or more tidyselect functions and/or a vector of quoted or unquoted variable names)
`relocate`	relocate columns with same spanner label to be adjacent

Value

data.frame or tibble

Examples

```
d <- data.frame(y = 1:3, x1 = 2:4, x2 = 3:5)

# Unquoted variable names
column_spanner_label(d, "Label", c(x1, x2))
# Character values (quoted variable names)
column_spanner_label(d, "Label", c("x1", "x2"))
# Tidyselect function (e.g., starts_with, ends_with, contains)
column_spanner_label(d, "Label", dplyr::starts_with("x"))
# Tidyselect range
column_spanner_label(d, "Label", x1:x2)
# Selected variables are relocated after the first selected variable
column_spanner_label(d, "Label", c(x2, y))
```

---

hanging_indent *Return markdown text with hanging indent*

Description

Return markdown text with hanging indent

Usage

```
hanging_indent(
  x,
  indent = 4,
  width = 30,
  space = NULL,
  newline = "\\\\\n",
  whitespace_only = FALSE,
  wrap_equal_width = FALSE
)
```

Arguments

x	text
indent	number of spaces to indent
width	number of characters to break lines
space	indenting space character (defaults to non-breaking space)
newline	text for creating new line
whitespace_only	wrapping spaces only
wrap_equal_width	Attempts to split lines to make them of approximately equal width

Value

character vector

Examples

```
hanging_indent("Hello Darkness, my old friend. I've come to talk with you again.")
```

`install_apaquarto` *Installs the apaquarto extension.*

Description

A wrapper for `quarto::quarto_add_extension`

Usage

```
install_apaquarto(
  no_prompt = FALSE,
  quiet = FALSE,
  quarto_args = NULL,
  ask = TRUE
)
```

Arguments

`no_prompt`	Do not prompt to confirm approval to download external extension.
`quiet`	Suppress warning and other messages
`quarto_args`	Character vector of other quarto CLI arguments to append to the Quarto command executed by this function.
`ask`	if `TRUE` (default), ask before installing missing packages. If `FALSE`, install missing packages without asking.

Value

installs the apaquarto Quarto extension

Examples

```
## Not run:
install_apaquarto()

## End(Not run)
```

is_numeric_like *Tests if a character vector contains numeric-like values*

Description

Tests if a character vector contains numeric-like values

Usage

```
is_numeric_like(x, elementwise = FALSE)
```

Arguments

x	character vector
elementwise	if TRUE, returns a logical vector for each element, otherwise returns a single logical value indicating if all elements are numeric-like (default: FALSE)

Value

logical vector

Examples

```
is_numeric_like(c("-9", " 2.0", "-1.0 "))
is_numeric_like(c("9-", -1, "10"))
is_numeric_like(c("9", -1.2, "10"))
```

make_apaquarto *Run shiny app to make a document in APA style via Quarto*

Description

A wrapper for shiny::runGitHub Note that running this function will install any missing packages needed to run the app: bsicons, bslib, conflicted, dplyr, fresh, purrr, ragg, rclipboard, readr, shiny, shinyWidgets, snakecase, svglite, tibble, tidyr, tippy, toastui, yaml

Usage

```
make_apaquarto(launch.browser = TRUE, ask = TRUE)
```

Arguments

launch.browser	run shiny app in default browser
ask	if TRUE (default), ask before installing missing packages. If FALSE, install missing packages without asking.

Value

Runs a shiny app that creates apaquarto documents

Examples

```
## Not run:
make_apaquarto()

## End(Not run)
```

`matrix_display` *Display latex matrices*

Description

Display latex matrices

Usage

```
matrix_display(x, bracket = c("[", "(", "|", ""), include_names = TRUE)
```

Arguments

`x`	a matrix
`bracket`	type of bracket: bmatrix, Bmatrix, pmatrix, vmatrix, Vmatrix, , matrix
`include_names`	if `TRUE` (default), the matrix's row and column names are displayed above and to the left of the matrix

Value

character

Examples

```
matrix_display(diag(2))
```

num_pad *Pads text on the left or right so that the width is the same for each element of the vector*

Description

By default, text is padded with one `padding_character` per missing visible character. Markdown markup (e.g., `**`, `*`, `^`, `~`, and backslash escapes) is not counted. If `font_family` names an installed font, text is instead padded to equal *rendered* width in that font, using a combination of figure, punctuation, thin, and hair spaces (`padding_character` is then ignored). Superscripts and subscripts are measured at 2/3 of the font size.

Usage

```
num_pad(
  x,
  pad_left = TRUE,
  padding_character = " ",
  NA_value = "",
  font_family = NULL
)
```

Arguments

`x`	vector of text
`pad_left`	if TRUE (default), pads on the left, otherwise pads on the right
`padding_character`	character to use for padding, default is ` ` (figure space)
`NA_value`	value to replace NA
`font_family`	font in which to measure text widths. If `NULL` (default) or `NA`, or if the font is not installed, widths are counted in characters.

Value

character vector

Examples

```
num_pad(c("a", "bb"))
num_pad(c(paste0(intToUtf8(8722), "1"), "1"), font_family = "Times New Roman")
```

p2stars *Convert p-values to stars*

Description

Convert p-values to stars

Usage

```
p2stars(
  p,
  alpha = c(0.05, 0.01, 0.001),
  first_alpha_marginal = FALSE,
  superscript = FALSE,
  add_trailing_space = FALSE,
  prefix = "\\"
)
```

Arguments

p	vector of numbers
alpha	vector of thresholds
first_alpha_marginal	if TRUE, the first alpha value is treated as marginal and gets a dagger instead of a star
superscript	make as superscript
add_trailing_space	if TRUE, adds a trailing space after the stars (default: FALSE)
prefix	usually backslashes to prevent markdown from interpreting asterisks as bullets or italics

Value

character vector

Examples

```
p2stars(c(.32, .02, .005),
        alpha = c(.05, .01))
```

pivot_wider_name_first *A wrapper for tidyr::pivot_wider that creates columns names as* name_variable *instead of* variable_name

Description

The default for names_vary is "slowest" instead of the usual "fastest".

Usage

```
pivot_wider_name_first(
  data,
  ...,
  id_cols = NULL,
  id_expand = FALSE,
  names_from = name,
  names_prefix = "",
  names_sep = "_",
  names_sort = FALSE,
  names_vary = "slowest",
  names_expand = FALSE,
  names_repair = "check_unique",
  values_from = value,
  values_fill = NULL,
  values_fn = NULL,
  unused_fn = NULL
)
```

Arguments

data	A data frame to pivot.
...	Additional arguments passed on to methods.
id_cols	<tidy-select> A set of columns that uniquely identify each observation. Typically used when you have redundant variables, i.e. variables whose values are perfectly correlated with existing variables. Defaults to all columns in data except for the columns specified through names_from and values_from. If a tidyselect expression is supplied, it will be evaluated on data after removing the columns specified through names_from and values_from.
id_expand	Should the values in the id_cols columns be expanded by expand() before pivoting? This results in more rows, the output will contain a complete expansion of all possible values in id_cols. Implicit factor levels that aren't represented in the data will become explicit. Additionally, the row values corresponding to the expanded id_cols will be sorted.

names_from, values_from
: <tidy-select> A pair of arguments describing which column (or columns) to get the name of the output column (names_from), and which column (or columns) to get the cell values from (values_from).

  If values_from contains multiple values, the value will be added to the front of the output column.

names_prefix
: String added to the start of every variable name. This is particularly useful if names_from is a numeric vector and you want to create syntactic variable names.

names_sep
: If names_from or values_from contains multiple variables, this will be used to join their values together into a single string to use as a column name.

names_sort
: Should the column names be sorted? If FALSE, the default, column names are ordered by first appearance.

names_vary
: When names_from identifies a column (or columns) with multiple unique values, and multiple values_from columns are provided, in what order should the resulting column names be combined?

  - "fastest" varies names_from values fastest, resulting in a column naming scheme of the form: value1_name1, value1_name2, value2_name1, value2_name2. This is the default.
  - "slowest" varies names_from values slowest, resulting in a column naming scheme of the form: value1_name1, value2_name1, value1_name2, value2_name2.

names_expand
: Should the values in the names_from columns be expanded by expand() before pivoting? This results in more columns, the output will contain column names corresponding to a complete expansion of all possible values in names_from. Implicit factor levels that aren't represented in the data will become explicit. Additionally, the column names will be sorted, identical to what names_sort would produce.

names_repair
: What happens if the output has invalid column names? The default, "check_unique" is to error if the columns are duplicated. Use "minimal" to allow duplicates in the output, or "unique" to de-duplicated by adding numeric suffixes. See vctrs::vec_as_names() for more options.

values_fill
: Optionally, a (scalar) value that specifies what each value should be filled in with when missing.

  This can be a named list if you want to apply different fill values to different value columns.

values_fn
: Optionally, a function applied to the value in each cell in the output. You will typically use this when the combination of id_cols and names_from columns does not uniquely identify an observation.

  This can be a named list if you want to apply different aggregations to different values_from columns.

unused_fn
: Optionally, a function applied to summarize the values from the unused columns (i.e. columns not identified by id_cols, names_from, or values_from).

  The default drops all unused columns from the result.

  This can be a named list if you want to apply different aggregations to different unused columns.

id_cols must be supplied for unused_fn to be useful, since otherwise all unspecified columns will be considered id_cols.

This is similar to grouping by the id_cols then summarizing the unused columns using unused_fn.

Value

data.frame

Examples

```
d <- data.frame(Model = paste0("Model","_", 1:2), t = 1:2, p = c(.05, .45))
pivot_wider_name_first(d, names_from = Model, values_from = c(t, p))
# For comparison:
tidyr::pivot_wider(d, names_from = Model, values_from = c(t, p))
```

---

pretty_widths *Use flextable::dim_pretty to fit column widths*

Description

Use flextable::dim_pretty to fit column widths

Usage

```
pretty_widths(
  x,
  min_width = 0.05,
  unit = c("in", "cm", "mm"),
  table_width = 6.5
)
```

Arguments

x	flextable
min_width	minimum width of columns
unit	Can be in, cm, or mm
table_width	width of table

Value

flextable::flextable

sci_md *Scientific notation for markdown*

Description

Scientific notation for markdown

Usage

```
sci_md(x, digits = 2, lb = 0.0095, ub = 99.5, align = FALSE)
```

Arguments

x	Numeric vector
digits	number of digits of precision
lb	lower bound where scientific notation is turned on
ub	upper bound where scientific notation is turned on
align	decimal align

Value

A character vector

Examples

```
sci_md(c(.0045,1.44, 24567857.45))
```

separate_star_column *Add columns that separate significance stars from numbers*

Description

Add columns that separate significance stars from numbers

Usage

```
separate_star_column(
  data,
  ...,
  superscript = TRUE,
  star = "\\*",
  star_replace = "\\\\*"
)
```

Arguments

data	data.frame or tibble
...	Column name or tidyselect function. Select columns
superscript	make stars superscript
star	character to use for stars (default: "*")
star_replace	character to replace stars with (default: "*")

Value

data.frame or tibble

Examples

```
tibble::tibble(x = c(".45", ".58*", ".68**"),
               y = c(1,2,3),
               z = 4:6) |>
               separate_star_column(x)
```

---

star_balance *Prefix text with figure spaces to balance star text*

Description

Prefix text with figure spaces to balance star text

Usage

```
star_balance(x, star = "\\*", superscript = TRUE)
```

Arguments

x	character vector
star	star text
superscript	Place superscript text if TRUE

Value

character vector

Examples

```
star_balance(".05\\^\\*\\*\\^")
```

str_wrap_equal *Like* stringr::str_wrap*, but attempts to create lines of equal width*

Description

Like stringr::str_wrap, but attempts to create lines of equal width

Usage

```
str_wrap_equal(x, max_width = 30L, sep = "\n")
```

Arguments

x	character
max_width	maximum line width
sep	separation character

Value

character

Examples

```
str_wrap_equal("This function attempts to split the string into lines with roughly equal width.")
```

tagger *Surrounds text with tags unless empty*

Description

Surrounds text with tags unless empty

Usage

```
tagger(x, tag = "<span>", right_tag = gsub("^<", "</", tag))

bold_md(x)

italic_md(x)

superscript_md(x)

subscript_md(x)

header_md(x, level = 1)
```

Arguments

x	character vector
tag	opening tag, e.g.,
right_tag	closing tag, e.g., . Defaults to the same value as the opening tag.
level	heading level

Value

character vector

Examples

```
x <- c("hello", "", NA)
tagger(x, "<span>")
bold_md(x)
italic_md(x)
superscript_md(x)
subscript_md(x)
header_md("Level 1")
header_md("Level 2", 2)
```

Index

add_break_columns, 3
add_list_column, 4
add_star_column, 5
align_chr, 6
align_chr(), *35*
apa7_defaults, 35
apa7_defaults(), *6*, *19*, *22*
apa7_tbl, 36
apa_anova, 7
apa_anova(), *18*, *22*, *29*, *36*
apa_chisq, 11
apa_cor, 13
apa_factor_cor, 14
apa_factor_cor(), *21*, *22*, *36*
apa_flextable, 15
apa_flextable(), *21*, *36*
apa_format_columns, 19
apa_loadings, 20
apa_loadings(), *15*, *17*, *18*, *22*, *36*
apa_note, 21
apa_note(), *18*, *37*
apa_p, 24
apa_p_star_note, 25
apa_parameters *(*apa_parameters.aov*)*, 26
apa_parameters(), *18*, *22*, *36*
apa_parameters.aov, 26
apa_performance, 30
apa_performance(), *36*
apa_performance_comparison, 32
apa_performance_comparison(), *10*, *29*, *36*
apa_style, 33

bold_md *(*tagger*)*, 51

column_format, 37
column_format(), *22*
column_formats, 38
column_spanner_label, 39

effectsize::eta_squared(), *7*
expand(), *46*, *47*

hanging_indent, 40
header_md *(*tagger*)*, 51

install_apaquarto, 41
is_numeric_like, 42
italic_md *(*tagger*)*, 51

knit_print.apa7_tbl *(*apa7_tbl*)*, 36

make_apaquarto, 42
matrix_display, 43

num_pad, 44
num_pad(), *6*

p2stars, 45
parameters::factor_analysis(), *14*, *21*
parameters::model_parameters(), *7*, *10*, *29*
parameters::principal_components(), *14*, *21*
performance::model_performance(), *31*
performance::r2(), *31*, *32*
performance::test_likelihoodratio(), *32*
performance::test_wald(), *32*
pivot_wider_name_first, 46
pretty_widths, 48
print.apa7_tbl *(*apa7_tbl*)*, 36
psych::fa(), *14*, *21*
psych::fa.sort(), *20*
psych::principal(), *14*, *21*

sci_md, 49
separate_star_column, 49
star_balance, 50
stats::anova(), *9*
stats::lm(), *29*
stats::prcomp(), *21*

str_wrap_equal, 51
subscript_md *(*tagger*)*, 51
superscript_md *(*tagger*)*, 51

tagger, 51

vctrs::vec_as_names(), *47*