Package 'gtstats'

September 21, 2026

Type Package

Title Beginner-Friendly Statistics and Publication-Ready Tables

Version 1.0.0

Description Provides beginner-friendly functions for common descriptive and inferential statistical analyses, together with tools for creating publication-ready tables. The package includes data description, summary statistics, distribution assessment, group comparisons, effect sizes, correlations, proportions and rates with confidence intervals, crosstabs for epidemiological measures, plots, and modular table-building workflows. Statistical methods include unequal-variance comparisons described by Welch (1947) <doi:10.1093/biomet/34.1-2.28>, score confidence intervals described by Wilson (1927) <doi:10.1080/01621459.1927.10502953>, and robust variance assessment described by Brown and Forsythe (1974) <doi:10.1080/01621459.1974.10482955>.

URL https://gtstats.thinkdenominator.com/, https://github.com/ThinkDenominator/gtstats

BugReports https://github.com/thinkdenominator/gtstats/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports dplyr, tibble, gt, flextable, officer, stats, ggplot2, tidyselect

Suggests shiny, rstudioapi, rio, testthat (>= 3.0.0), knitr, rmarkdown, pkgdown, webshot2

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Rubeshkumar Polani [aut, cre] (ORCID:
<https://orcid.org/0000-0002-0418-7592>),
Mogan Kaviprawin [aut] (ORCID: <https://orcid.org/0000-0002-1659-059X>),
Manikandanesan Sakthivel [aut] (ORCID:
<https://orcid.org/0000-0002-5438-3970>),
Salin K Eliyas [aut] (ORCID: <https://orcid.org/0000-0002-8020-5860>),
Yuvaraj Krishnamoorthy [aut] (ORCID:
<https://orcid.org/0000-0003-4688-510X>)

Maintainer Rubeshkumar Polani <rubesh@thinkdenominator.com>

Repository CRAN

Date/Publication 2026-09-21 21:50:02 UTC

Contents

`add_ci` *Add confidence intervals to a summary table*

Description

Add confidence intervals to all eligible summaries, or only to selected variables, without rebuilding the descriptive table. Categorical variables receive binomial confidence intervals for their displayed proportions. Continuous variables displayed with a mean receive a t-based confidence interval for the mean. Median-only summaries are left unchanged because a distribution-free median interval is not implied by the displayed IQR. Compact tables show the interval after the estimate without repeating the confidence level in every cell. Separate tables use concise `CI` child columns. The confidence level and interval method are stated once in the publication footnote.

Usage

```
add_ci(
  x,
  vars = NULL,
  conf.level = 0.95,
  method = c("wilson", "exact"),
  digits = NULL,
  type = NULL,
  estimate = NULL,
  numerator = NULL,
  denominator = NULL,
  sd = NULL,
  n = NULL,
  se = NULL,
  multiplier = 1,
  ci_name = NULL
)
```

Arguments

x	A table created by summary_table() or as_stats_table().
vars	Variables that should receive confidence intervals. NULL (default) selects every eligible variable already in the table. Variables may be supplied as bare names, for example c(age, sex), or as a character vector.
conf.level	Confidence level. Default is 0.95.
method	Binomial interval method for categorical proportions: "wilson" (default) or "exact". Continuous mean intervals use the usual t interval.
digits	Decimal places for confidence limits. NULL inherits the confidence-interval precision from the table.
type	For as_stats_table() input, the explicit aggregate-data calculation: "proportion", "rate", "mean", or "normal".
estimate, numerator, denominator, sd, n, se	Columns containing the required aggregate inputs. Supply bare column names or single character names. Proportions and rates require numerator and denominator; means require estimate, sd, and n; normal intervals require estimate and se.
multiplier	Positive rate multiplier, such as 1000 person-years.
ci_name	Optional name for the added interval column. The default is the confidence-level label, for example "95% CI".

Value

The updated gtstats_summary object.

Examples

```
summary_table(mtcars, by = am, include = c(mpg, cyl), layout = "separate") |>
  add_ci()

summary_table(mtcars, by = am, include = c(mpg, cyl, vs)) |>
  add_ci(vars = c(mpg, vs), conf.level = 0.90)

aggregate_rates <- data.frame(
  Group = c("A", "B"), Events = c(8, 14), PersonYears = c(420, 510)
)
as_stats_table(aggregate_rates) |>
  add_ci(type = "rate", numerator = Events, denominator = PersonYears,
         multiplier = 1000)
```

add_p *Add p-values to a descriptive table*

Description

Add a p-value column to a descriptive table by comparing each displayed variable across the grouping variable. P-values are calculated using `compare_groups()` and inserted once per variable, with optional superscript markers indicating which statistical test was used.

Usage

```
add_p(
  x,
  test = "auto",
  include = tidyselect::everything(),
  paired = FALSE,
  id = NULL,
  distribution_check = TRUE,
  var_equal = FALSE,
  correction = TRUE,
  fisher_seed = 1049L,
  p_adjust = c("none", setdiff(stats::p.adjust.methods, "none")),
  digits = 3
)
```

Arguments

`x`	A `gtstats_summary` object created with `summary_table()`.
`test`	Statistical test to use. Either a single test string, or a named character vector/list specifying methods for individual variables. Names may match either displayed variable labels or underlying variable names.
`include`	Variables in the descriptive table for which p-values should be calculated. Uses tidy-select syntax and defaults to all summarized variables. For example, `include = -bwt` keeps birth weight in the descriptive table but omits its p-value when the grouping variable was derived from birth weight.
`paired`	Logical; whether comparisons should be treated as paired.
`id`	Pair or participant identifier required when `paired = TRUE`.
`distribution_check`	Logical; when `test = "auto"`, use distribution guidance to choose parametric or rank-based tests. This guidance is based on skewness; Shapiro-Wilk is supporting information only. For paired analyses the check is applied to within-pair differences.
`var_equal`	Logical; for independent, non-skewed continuous comparisons in `test = "auto"`, use Student's t-test or classical ANOVA. The default `FALSE` uses Welch methods. This is a user-specified assumption, not a variance test, and does not affect paired, categorical, ordinal, or rank-based comparisons.

correction	Logical; apply continuity correction to chi-square and McNemar tests where applicable.
fisher_seed	Integer seed for simulated Fisher exact tests on tables larger than 2 x 2. Use NULL to use the current random-number state.
p_adjust	Multiplicity adjustment applied across displayed variable tests. One of stats::p.adjust.methods; default "none".
digits	Number of decimal places used when formatting p-values.

Details

This function works only for descriptive tables created in mode = "summary" and requires a grouping variable supplied via summary_table().

Supported methods include:

- "auto"
- "welch_t"
- "t_test"
- "wilcox"
- "anova"
- "welch_anova"
- "kruskal"
- "chisq"
- "fisher"
- "mcnemar"

You may also provide a named character vector or named list to specify different methods for individual variables.

Automatic selection and audit trail:

With test = "auto", add_p() delegates each comparison to compare_groups() using the same fixed selection policy: Welch t-test or Welch ANOVA by default when distribution guidance does not flag skewness; Student's t-test or classical ANOVA when var_equal = TRUE; rank-based tests when marked skewness is flagged; and chi-square or Fisher's exact test according to expected cell counts. For independent ordered factors, automatic mode uses the same chi-square/Fisher distribution comparison as other categorical variables; select "wilcox" or "kruskal" explicitly when a rank-based ordinal comparison is wanted. Shapiro-Wilk is supporting information only and does not itself select a test.

The publication table contains only the p-value and compact test markers. The full per-variable audit trail is retained in $assumptions, $diagnostics, $p_values, and $denominators. In particular, diagnostics_stats(x) records automatic selection, distribution guidance, expected cell counts where relevant, and observed group spread for independent continuous comparisons. Observed spread is descriptive context, not a variance-test gatekeeper; Welch methods do not require equal variances.

Value

An updated `gtstats_summary` object with a `p-value` column added. When `paired = TRUE`, `$paired_p_notes` records the complete-pair denominator for each displayed p-value; `to_gt()` displays this as a concise p-value footnote.

Examples

```
summary_table(mtcars, by = am, include = c(mpg, wt, cyl)) |>
  add_p()

summary_table(mtcars, by = am, include = c(mpg, wt, cyl)) |>
  add_p(test = c(mpg = "welch_t", wt = "wilcox", cyl = "chisq"))

summary_table(mtcars, by = am, include = c(mpg, wt, cyl)) |>
  add_p(include = c(mpg, wt))
```

---

add_proportion *Add a proportion row to a descriptive table*

Description

Add a row showing the proportion of a selected level of a binary, categorical, or ordinal variable within a `gtstats` descriptive table.

Usage

```
add_proportion(
  x,
  var,
  level = NULL,
  ci = TRUE,
  conf.level = NULL,
  ci_method = NULL,
  display = NULL,
  layout = NULL,
  digits = NULL,
  label = NULL
)
```

Arguments

`x`	A `gtstats_summary` object created with `summary_table()`.
`var`	Variable to summarise as a proportion. Can be supplied as a bare name or as a character string.
`level`	Optional level to count. If `NULL`, a default level is selected automatically.
`ci`	Logical; whether to display a confidence interval.

conf.level	Confidence level for the interval. NULL inherits the parent table setting, usually 0.95.
ci_method	Confidence-interval method: "wilson" or "exact". NULL inherits the parent table setting.
display	Cell display: "n_percent", "percent", or "n_over_N_percent". NULL inherits the categorical display used by the parent table.
layout	Table layout. NULL inherits the parent table layout; "compact" keeps the estimate and CI together and "separate" places them in separate columns beneath each cohort header.
digits	Number of decimal places used when formatting percentages. NULL inherits the parent table precision.
label	Optional row label. Defaults to the variable label if available, otherwise the variable name.

Details

This is useful when you want to highlight a specific category such as "Yes", "1", or "TRUE" within a Table 1 workflow. The row can be added overall, by groups, or both, depending on how the descriptive table was created.

If level = NULL, the function chooses a default level using the following order:

- "1"
- "Yes" / "yes"
- "TRUE" / "True" / "true"
- the second available level for binary variables
- otherwise the first available non-missing level

Wilson confidence intervals are used by default. Exact binomial intervals are available with ci_method = "exact".

Value

An updated gtstats_summary object with a proportion row appended.

Examples

```
summary_table(mtcars, by = am, overall = TRUE) |>
  add_proportion(var = vs)

summary_table(mtcars, by = am, overall = TRUE) |>
  add_proportion(var = vs, level = "1", ci = TRUE)

summary_table(mtcars) |>
  add_proportion(var = vs, ci = FALSE)
```

add_rate *Add an event-rate row*

Description

Add an event rate to a table created with summary_table(). Event counts and accumulated time are calculated from complete event-time pairs only. Event counts and time values must be finite. Exact Poisson confidence intervals are shown by default. A row with zero accumulated time is retained as \u2014 and recorded as not estimable in the audit.

Usage

```
add_rate(
  x,
  event,
  time,
  label = NULL,
  multiplier = 1000,
  time_label = NULL,
  ci = TRUE,
  conf.level = 0.95,
  digits = 1,
  layout = NULL
)
```

Arguments

x	A gtstats_summary created with summary_table().
event	Non-negative integer event count, supplied as a bare name or character string. Logical values are accepted as binary event indicators.
time	Non-negative numeric person-time or exposure-time variable.
label	Optional row label.
multiplier	Positive rate multiplier. Default is 1000.
time_label	Optional readable time unit such as "person-years" or "catheter-days". Defaults to "person-time".
ci	Logical; display an exact Poisson confidence interval.
conf.level	Confidence level.
digits	Number of decimal places.
layout	Table layout. NULL inherits the parent table layout; "compact" keeps rate and CI together and "separate" places them in separate columns beneath each cohort header.

Details

Rate denominators are tracked separately from ordinary descriptive rows and are stated in the table audit and footnote.

Value

The updated `gtstats_summary`.

Examples

```
summary_table(mtcars, by = am, overall = TRUE) |>
  add_rate(
    event = carb,
    time = cyl,
    label = "Carburettor rate",
    multiplier = 1000
  )

# Rates may be added alongside ordinary summaries
summary_table(mtcars, by = am, include = mpg) |>
  add_rate(event = carb, time = cyl, multiplier = 1000)
```

---

add_row *Add a custom row to a descriptive table*

Description

Add a user-defined row to a `gtstats` descriptive table. This is useful for inserting contextual information such as study period, data source, setting, or other custom annotations that should appear alongside the main table content.

Usage

```
add_row(x, label, overall = NULL, values = NULL, level = "")
```

Arguments

`x`	A `gtstats_summary` object created with `summary_table()`.
`label`	A single character string giving the row label to display in the `Variable` column.
`overall`	Optional value to display in the `Overall` column when the descriptive table includes `overall = TRUE`.
`values`	Optional named character vector or named list giving values for displayed group columns. Names must match the displayed group column names, for example `c("am = 1" = "2020-2024", "am = 0" = "2020-2024")`.
`level`	Optional text to display in the `Level` column. Defaults to an empty string.

Details

The row is matched to the current table structure automatically:

- if `overall = TRUE`, an `Overall` column is supported
- if by is used, values must be named using displayed group column names
- if neither `overall` nor by is used, a single `Value` column is used

Value

An updated `gtstats_summary` object with the custom row appended.

Examples

```
res <- summary_table(mtcars, by = am, include = c(mpg, wt), overall = TRUE) |>
  add_row(
    label = "Study period",
    overall = "2020-2024",
    values = c("am = 1" = "2020-2024", "am = 0" = "2020-2024")
  )
```

`add_summary` *Add summary rows to a descriptive table*

Description

Add summary statistics to a `gtstats` descriptive table builder.

Usage

```
add_summary(
  x,
  vars,
  statistic = "recommended",
  percent = c("column", "row", "overall", "none"),
  categorical = c("n_percent", "n_over_N_percent", "n", "percent"),
  categorical_layout = c("combined", "separate"),
 overall_categorical = c("auto", "n_percent", "n_over_N_percent", "n", "percent"),
  show_dichotomous = c("all_levels", "single_row"),
  value = NULL,
  layout = NULL,
  missing = c("ifany", "always", "no", "as_category"),
  digits = 1
)
```

Arguments

`x`	A `gtstats_summary` object created with `summary_table()`.
`vars`	Variables to summarise. Can be supplied as bare names or as a character vector.
`statistic`	Continuous summary selection. A single value applies to all selected continuous variables. A named character vector can select a different summary for each variable, for example `c(age = "mean_sd", bmi = "median_iqr")`. `"auto"` is accepted as an alias for `"recommended"`.

`percent` Denominator for categorical percentages: `"column"` uses the non-missing denominator within each group, `"row"` distributes each level across groups, `"overall"` uses the overall non-missing denominator, and `"none"` displays counts only.

`categorical` Display for categorical values: `"n_percent"`, `"n_over_N_percent"`, `"n"`, or `"percent"`.

`categorical_layout` Categorical display layout. `"combined"` keeps n and % together. `"separate"` creates distinct n and % child columns and is available for categorical-only tables without confidence intervals.

`overall_categorical` Categorical display used only in the Overall column. `"auto"` uses counts only when `percent = "row"` and otherwise follows `categorical`. Other choices are `"n_percent"`, `"n_over_N_percent"`, `"n"`, and `"percent"`.

`show_dichotomous` How binary variables are displayed. `"all_levels"` (default) shows both levels. `"single_row"` shows one event level as a compact row using the variable label.

`value` Optional named character vector or named list selecting the event level used when `show_dichotomous = "single_row"`, for example `c(smoke = "Yes", hypertension = "Yes")`. When omitted, the second declared factor level (or the second sorted observed value) is used.

`layout` Table layout. `"compact"` keeps each summary in one cell; `"separate"` places summaries and confidence intervals in separate columns once intervals are added. It does not create empty CI columns. When omitted, the layout chosen in `summary_table()` is used.

`missing` Missing-value display and percentage handling. `"ifany"` shows a missing row only when needed, `"always"` always shows it, and `"no"` hides it; these three use non-missing categorical denominators. `"as_category"` displays missing values as a category and includes them when calculating categorical percentages.

`digits` One number applied throughout, or a named numeric vector using `continuous`, `percent`, and `ci`.

Details

This function is the main way to populate a descriptive table with variable summaries. It supports both grouped and ungrouped tables and can optionally add an `Overall` column when the descriptive table was created with `overall = TRUE`.

Continuous variables can be displayed in one of four formats:

- `"recommended"`: mean (SD) or median (IQR) as appropriate
- `"mean_sd"`: mean (SD)
- `"mean_se"`: mean (standard error)
- `"mean_ci"`: mean with a t confidence interval
- `"median_iqr"`: median (IQR)
- `"both"`: mean (SD) and median (IQR)

Variable names may be supplied either as bare names, for example `c(age, sex, bmi)`, or as a character vector, for example `c("age", "sex", "bmi")`.

Value

An updated `gtstats_summary` object with summary rows added.

Examples

```
summary_table(mtcars, by = am) |>
  add_summary(vars = c(mpg, wt, cyl))

summary_table(mtcars, by = am, overall = TRUE) |>
  add_summary(vars = c("mpg", "wt", "cyl"))

summary_table(mtcars) |>
  add_summary(vars = c(mpg, wt), statistic = "mean_sd")

missing_example <- mtcars
missing_example$vs[1:3] <- NA
summary_table(missing_example) |>
  add_summary(vars = vs, missing = "as_category")
```

---

add_total *Add total counts to a descriptive table*

Description

Add a total row to a `gtstats` descriptive table showing the number of observations overall or within each displayed group.

Usage

```
add_total(x, label = "Total (N)", position = c("last", "first"))
```

Arguments

`x`	A `gtstats_summary` object created with `summary_table()`.
`label`	Row label to display in the `Variable` column. Defaults to `"Total (N)"`.
`position`	Position of the total row. Use `"first"` to place sample sizes at the top of the table or `"last"` to append them.

Details

This helper is useful in Table 1 workflows where a final row is needed to show the number of observations contributing to each column. When the table includes an `Overall` column, the total number of rows in the source data is shown there. When the table is grouped, totals are calculated within each displayed group. Publication-table headers already display these cohort denominators automatically, so this row is optional.

This helper can be used only with descriptive tables created in `mode = "summary"`.

Value

An updated `gtstats_summary` object with a total row appended.

Examples

```
summary_table(mtcars, by = am, include = c(mpg, wt, cyl)) |>
  add_total()

summary_table(mtcars, by = am, include = c(mpg, wt), overall = TRUE) |>
  add_total()

summary_table(mtcars) |>
  add_total()
```

---

assess_distribution *Assess the empirical distribution of continuous variables*

Description

Assess the empirical distribution of continuous numeric variables to support descriptive reporting. The function describes missingness, finite sample size, skewness, and (optionally) Shapiro-Wilk results. It provides guidance about presenting a variable; it does **not** select an inferential test.

Usage

```
assess_distribution(
  data,
  vars = NULL,
  by = NULL,
  normality_test = TRUE,
  skew_cutoff = 1,
  min_n = 3,
  plots = FALSE,
  digits = 2,
  format = c("table", "tibble")
)
```

Arguments

`data`	A data frame.
`vars`	Continuous numeric variables to assess. Bare names or a character vector are accepted. When omitted, all detected continuous variables are assessed. Categorical, ordinal, logical, date-time, and binary variables are rejected when explicitly selected.
`by`	Optional grouping variable, supplied as a bare name or character string. Factors, characters, logical variables, binary variables, and ordinal variables are supported.

normality_test	Logical; run Shapiro-Wilk when 3 to 5000 finite observations are available. Default is TRUE.
skew_cutoff	Positive absolute-skewness threshold for marked skew.
min_n	Minimum finite observations required for a shape assessment.
plots	Logical; create histogram, density, Q-Q, and box plots. Plots are stored in $plots (or attr(result, "plots") for tibble output).
digits	Number of decimal places.
format	Output format: "table" (default) or "tibble".

Details

When by is supplied, diagnostics are calculated within every group and one consistent, variable-level recommendation is also returned in $recommendations. Shapiro-Wilk is supporting information only: it is sensitive to sample size and never determines the recommendation by itself.

Value

With format = "table", a gt_distribution object that prints as a publication-ready table. $summary contains group-level diagnostics and $recommendations contains one descriptive recommendation per variable. With format = "tibble", the group-level summary tibble is returned.

Examples

```
assess_distribution(mtcars, vars = c(mpg, wt))
assess_distribution(mtcars, vars = c(mpg, wt), by = am)
assess_distribution(mtcars, vars = "mpg", normality_test = FALSE)
assess_distribution(mtcars, vars = c("mpg", "wt"), plots = TRUE)$plots
```

---

assess_variance *Assess variation of continuous variables across groups*

Description

Describe the spread of continuous numeric variables within groups. This is a diagnostic companion to assess_distribution() and is intended to make variation visible before a group comparison is interpreted.

Usage

```
assess_variance(
  data,
  vars = NULL,
  by,
  digits = 2,
  test = c("levene", "none", "bartlett"),
  format = c("table", "tibble")
)
```

Arguments

data	A data frame.
vars	Continuous numeric variables to assess. Bare names or a character vector are accepted. When omitted, all detected continuous variables are assessed. Categorical, ordinal, logical, date-time, and binary variables are rejected when explicitly selected.
by	Grouping variable, supplied as a bare name or character string. It must be categorical, binary, logical, or ordinal and contain at least two observed groups.
digits	Number of decimal places.
test	Variance hypothesis test to display: "levene" (default), "none", or "bartlett". Levene's test is median-centred (the robust Brown-Forsythe form). Both are supporting diagnostics, not gatekeepers for ANOVA or Welch methods.
format	Output format: "table" (default) or "tibble".

Details

assess_variance() reports group sample sizes, standard deviations, variances, and the ratio of the largest to the smallest group SD and variance. These ratios are descriptive diagnostics, not pass/fail tests. The function deliberately does not run a variance hypothesis test by default, and it does not choose an inferential test. Welch t-tests and Welch ANOVA do not require equal variances for *independent* groups; this function does not assess pairing, repeated-measures sphericity, or select a repeated-measures method. The default is the median-centred Levene test (often called the Brown-Forsythe modification) as supporting information. It is less sensitive to non-normality than Bartlett's test. Set test = "none" for descriptive spread only, or test = "bartlett" when the normal-distribution assumption is justified. Neither test is used to select a test in compare_groups().

Value

With format = "table", a gt_variance object that prints as one readable row per variable: each group's n and SD, the observed SD ratio, the requested test p-value, and a plain-language interpretation. $summary contains the full group-level values and $diagnostics retains technical test metadata. With format = "tibble", the detailed summary tibble is returned.

Examples

```
assess_variance(mtcars, vars = c(mpg, wt), by = am)
assess_variance(mtcars, vars = "mpg", by = am, digits = 1)
assess_variance(mtcars, vars = "mpg", by = am, test = "bartlett")
assess_variance(mtcars, vars = "mpg", by = am, test = "levene")
```

assumptions_stats *Inspect statistical assumptions*

Description

Return a plain-language checklist of items to confirm before reporting a result. Use `view = "audit"` to retrieve the underlying technical status and result codes retained by the analysis object.

Usage

```
assumptions_stats(
  x,
  format = c("table", "tibble"),
  title = "Checks before reporting",
  subtitle = NULL,
  view = c("checklist", "audit")
)
```

Arguments

`x`	A `gtstats` result.
`format`	Output format: `"table"` (default) or `"tibble"`.
`title, subtitle`	Optional table heading used for `format = "table"`.
`view`	Either `"checklist"` (the default, plain-language view) or `"audit"` (technical status and result codes).

Value

A tibble or `gt_tbl`.

Examples

```
result <- compare_groups(mtcars, variable = mpg, group = am)
assumptions_stats(result)
assumptions_stats(result, format = "tibble")
```

as_stats_table *Convert an already summarised data frame into a publication table*

Description

Wrap a data frame or tibble containing values that have already been calculated so it can be rendered, styled, and exported with gtstats. No descriptive statistics, confidence intervals, or p-values are calculated or checked by this function.

Usage

```
as_stats_table(data, notes = NULL)
```

Arguments

data	A data frame or tibble containing one row per intended table row.
notes	Optional character vector of explanatory notes to display below the table.

Details

Use summary_table() instead when each row represents a participant or observation and descriptive statistics still need to be calculated. Use epi_table() when events and denominators need to be calculated from a line list or aggregate outbreak/surveillance data.

Value

A gt_data_table object. Use customise_table(), to_flextable(), to_gt(), or save_output() to present or export it.

Examples

```
summarised <- mtcars |>
  dplyr::summarise(
    Cars = dplyr::n(),
    `Mean mpg` = mean(mpg),
    `Mean weight` = mean(wt)
  )

table <- as_stats_table(
  summarised,
  notes = "Values were calculated before table formatting."
)
customise_table(table, title = "Vehicle summary")
to_flextable(table)
to_gt(table)
```

---

birthwt *Low birth weight data*

Description

A labelled and analysis-ready version of the low birth weight study data. It contains 189 observations and is derived from the dataset distributed in MASS, originally reported by Hosmer and Lemeshow. Numeric clinical codes have been converted to readable factors; antenatal_visits is an ordered factor and the original numeric variables are retained.

Usage

```
birthwt
```

Format

A data frame with 189 rows and 12 variables:

low Birth-weight outcome: Normal birth weight or Low birth weight.

age Maternal age in years.

lwt Maternal weight in pounds.

race Maternal race.

smoke Smoking during pregnancy.

ptl Number of previous premature labours.

ht History of hypertension.

ui Uterine irritability.

ftv Number of first-trimester physician visits.

bwt Birth weight in grams.

previous_preterm Any previous premature labour.

antenatal_visits Ordered visit category.

Source

Hosmer, D. W. and Lemeshow, S. (1989). *Applied Logistic Regression*. Derived from `MASS::birthwt`.

Examples

```
describe_data(birthwt)
summary_table(birthwt, by = low, include = c(age, lwt, smoke), overall = TRUE)
```

---

compare_groups *Compare groups using common inferential tests*

Description

Compare a variable across groups using a practical set of common inferential tests.

Usage

```
compare_groups(
  data,
  variable,
  group,
  paired = FALSE,
  id = NULL,
 test = c("auto", "t_test", "welch_t", "wilcox", "anova", "welch_anova", "kruskal",
    "chisq", "fisher", "mcnemar", "rm_anova", "friedman", "cochran_q"),
  effect_size = FALSE,
  conf.level = 0.95,
  digits = 2,
  var_equal = FALSE,
  fisher_seed = 1049L,
  format = c("table", "tibble")
)
```

Arguments

data	A data.frame.
variable	Variable to compare. Can be supplied as a bare name or as a character string.
group	Grouping variable. Can be supplied as a bare name or as a character string.
paired	Logical; whether the comparison is paired. If TRUE, paired t-test, Wilcoxon signed-rank test, or McNemar test will be used where appropriate.
id	Pair or participant identifier required when paired = TRUE. Each identifier must occur at most once in each group.
test	Test to use. One of "auto", "t_test", "welch_t", "wilcox", "anova", "welch_anova", "kruskal", "chisq", "fisher", "mcnemar", "rm_anova", "friedman", or "cochran_q". See **Automatic selection policy** for the exact rules used by "auto".
effect_size	Logical; calculate and display the effect size selected for the comparison structure. Default is FALSE.
conf.level	Confidence level for intervals.
digits	Number of decimal places for formatting.
var_equal	Logical; for independent, non-skewed continuous outcomes with test = "auto", use equal-variance Student's t-test (two groups) or classical one-way ANOVA (three or more groups). The default FALSE uses Welch methods. This is a pre-specified user choice and is not tested or inferred from the observed variances. It does not affect paired, categorical, ordinal, or rank-based comparisons.
fisher_seed	Integer seed used only for simulated Fisher exact tests on tables larger than 2 x 2. The default makes results reproducible. Set to NULL to use the current random-number state.
format	Output format: "table" (default) or a plain console "tibble".

Details

This function is designed for beginner-friendly and teaching-focused workflows. It combines:

- descriptive summaries by group
- automatic or user-specified test selection
- effect size calculation where supported
- a simple display-ready results table

Supported outcome types are:

- continuous
- binary
- categorical
- ordinal

Supported tests include:

- `"auto"`
- `"t_test"`
- `"welch_t"`
- `"wilcox"`
- `"anova"`
- `"welch_anova"`
- `"kruskal"`
- `"chisq"`
- `"fisher"`
- `"mcnemar"`
- `"rm_anova"`
- `"friedman"`
- `"cochran_q"`

Automatic selection policy:

`test = "auto"` uses the following fixed, data-driven rules. These rules are intended as transparent defaults, not a substitute for a prespecified analysis plan.

- **Continuous outcome, two independent groups:** Welch t-test by default, or Student's t-test when `var_equal = TRUE`, unless marked skewness is flagged in either group, then Wilcoxon rank-sum test.
- **Continuous outcome, three or more independent groups:** Welch ANOVA by default, or classical one-way ANOVA when `var_equal = TRUE`, unless marked skewness is flagged in any group, then Kruskal-Wallis test.
- **Paired continuous outcome:** for two occasions, paired t-test unless marked skewness in within-pair differences is flagged, then Wilcoxon signed-rank; for three or more occasions, repeated-measures ANOVA unless marked skewness is flagged, then Friedman test. Repeated-measures ANOVA reports a conservative Greenhouse-Geisser-corrected p-value.

- **Independent ordinal, binary, or nominal categorical outcome:** Pearson chi-square test when no expected count is below 1 and no more than 20% are below 5; Fisher's exact test otherwise (Monte Carlo p-value for larger tables). This compares the distribution of all levels, which is the usual Table 1 question. Use an explicit rank test (`"wilcox"` or `"kruskal"`) when the ordered scale itself is the intended estimand.
- **Paired ordinal outcome:** Wilcoxon signed-rank for two occasions; Friedman for three or more paired occasions.
- **Paired binary outcome:** McNemar test for two occasions; Cochran's Q test for three or more occasions.

Distribution guidance uses the package's skewness assessment within each group (or within-pair differences). Shapiro-Wilk is supporting information; it does not by itself change the selected test. Automatic decisions, the values used, and the selected method are retained in `$method`, `$diagnostics`, and `$notes`.

For independent continuous comparisons, `$diagnostics` also reports the observed standard deviation and variance ratios across groups. These are descriptive context only: they have no pass/fail threshold and do not alter automatic test selection. `var_equal` is a user-specified analytical assumption, not a variance hypothesis test: gtstats never infers it using Levene, Bartlett, or F tests. Welch t-tests and Welch ANOVA are the conservative defaults because they do not require equal variances.

When `effect_size = TRUE`, the function selects an effect size from the comparison structure:

- Hedges' g for two-group parametric comparisons
- rank-biserial correlation for two-group rank comparisons
- omega-squared for ANOVA or Welch ANOVA
- epsilon-squared for Kruskal-Wallis comparisons
- Cramer's V for categorical contingency tables

Hedges' g is accompanied by a large-sample confidence interval. Other effect-size intervals are omitted unless a supported interval method is available. Conventional magnitude labels are retained in `inferential` for teaching but are not displayed as clinical importance thresholds.

Value

A `gt_compare` object containing:

- `inputs` — function inputs and settings
- `descriptives` — descriptive summaries by group
- `inferential` — inferential test results
- `table` — display-ready results table
- `method` — metadata on detected variable types
- `notes` — explanatory notes
- `call` — matched function call

For paired or repeated analyses, only complete, uniquely matched identifiers are analysed. The number retained and excluded is available in `$denominators` and `$notes`; rendered tables also identify the complete-pair denominator. Friedman and Cochran's Q require within-participant variation and fail clearly when it is absent.

Examples

```
compare_groups(mtcars, variable = mpg, group = am)

compare_groups(
  mtcars,
  variable = mpg,
  group = am,
  effect_size = TRUE
)

compare_groups(
  mtcars,
  variable = vs,
  group = am,
  test = "chisq",
  effect_size = TRUE
)

compare_groups(
  mtcars,
  variable = mpg,
  group = am,
  paired = FALSE,
  test = "welch_t"
)

compare_groups(mtcars, variable = mpg, group = am, var_equal = TRUE)

to_gt(compare_groups(mtcars, variable = mpg, group = am))
```

---

correlation *Correlation analysis for one pair or several continuous variables*

Description

`correlation()` analyses either one prespecified pair (`x` and `y`) or a correlation matrix (`vars`). Matrix mode uses one method throughout, retains pairwise sample sizes and inferential results in `$summary`, and prints a compact publication-ready matrix. Use `plot_correlation()` for a shaded heatmap of a matrix result.

Usage

```
correlation(
  data,
  x = NULL,
  y = NULL,
  method = c("auto", "pearson", "spearman"),
  conf.level = 0.95,
```

```
  digits = 2,
  vars = NULL,
  triangle = c("lower", "upper", "full"),
  order = c("input", "alphabetical", "cluster"),
  show_diagonal = TRUE,
 display = c("estimate", "estimate_p", "estimate_n", "estimate_p_n", "estimate_ci"),
  shade = TRUE,
  missing = c("pairwise"),
  adjust = c("none", "holm", "bonferroni", "BH"),
  format = c("table", "tibble")
)
```

Arguments

`data`	A data frame.
`x, y`	Two continuous variables supplied as bare names or character strings. Omit these when using `vars`.
`method`	Correlation method: `"auto"`, `"pearson"`, or `"spearman"`.
`conf.level`	Confidence level for intervals.
`digits`	Number of decimal places used for display.
`vars`	Optional vector of at least two continuous variables, supplied as `c(age, weight, outcome)` or a character vector.
`triangle`	Matrix display: `"lower"`, `"upper"`, or `"full"`.
`order`	Variable order in matrix mode: `"input"` preserves the order in `vars`, `"alphabetical"` orders display labels, and `"cluster"` places variables with similar absolute correlation patterns together.
`show_diagonal`	Logical; show self-correlations on the diagonal.
`display`	Matrix cell content: correlation `"estimate"`, `"estimate_p"`, `"estimate_n"`, `"estimate_p_n"`, or `"estimate_ci"`. Confidence intervals unavailable from the selected method are shown as an em dash in the tidy result and omitted from the matrix cell.
`shade`	Logical; apply coefficient-based shading to the publication matrix. This affects rendering, not `$summary`.
`missing`	Matrix missing-data rule. Currently `"pairwise"`: each coefficient uses all complete finite observations for that pair.
`adjust`	Multiplicity adjustment for matrix p-values: `"none"`, `"holm"`, `"bonferroni"`, or `"BH"`.
`format`	Output format: `"table"` (default) or a plain console `"tibble"`.

Details

In automatic matrix mode, Pearson correlation is used only when every selected variable has absolute sample skewness below 1; otherwise Spearman correlation is used throughout. This is transparent descriptive guidance, not proof of linearity or monotonicity. Inspect the matrix heatmap and relevant pairwise plots before interpretation.

Value

A gt_correlation object. Matrix results additionally inherit from gt_correlation_matrix and contain a tidy pair-level $summary.

Examples

```
correlation(mtcars, x = mpg, y = wt)
correlation(mtcars, vars = c(mpg, disp, hp, wt))
plot_correlation(correlation(mtcars, vars = c(mpg, disp, hp, wt)))
```

---

crosstabs *Cross-tabulations with optional 2x2 epidemiological measures*

Description

Create a publication-ready cross-tabulation for any two categorical variables. Counts, selected row/column/total percentages, and margins are displayed. For a binary 2x2 table, risks, risk ratios, odds ratios, and risk differences are additionally available because their direction is defined. For a 2x2 table, the selected exposed and event levels are always retained in the result, including when they were chosen automatically. Association diagnostics, zero-cell handling, and complete-pair denominators remain in the audit components rather than cluttering the displayed table.

Usage

```
crosstabs(
  data,
  row,
  col,
  percent = "column",
  totals = TRUE,
  row_level = NULL,
  col_level = NULL,
  measures = c("rr", "or", "rd"),
  conf.level = 0.95,
  risk_ci = c("wilson", "exact"),
  test = c("auto", "none", "chisq", "fisher"),
  zero_correction = c("haldane_anscombe", "none"),
  simulate_B = 10000,
  digits = 2,
  format = c("table", "tibble")
)
```

Arguments

data	A data frame.
row	Categorical row variable. For a binary 2x2 table, this is the exposure/reference axis.

col	Categorical column variable. For a binary 2x2 table, this is the outcome/event axis.
percent	Percentages to show in each cell: "column" (default), "row", "total", or "none". Supply c("row", "column") to show more than one denominator.
totals	Logical; include row, column, and grand totals.
row_level	Level of row treated as exposed. A sensible event-like level is selected when omitted; set this explicitly for reporting.
col_level	Level of col treated as the event. A sensible event-like level is selected when omitted; set this explicitly for reporting.
measures	Measures to display: risk, risk ratio ("rr"), odds ratio ("or"), and/or risk difference ("rd").
conf.level	Confidence level for intervals.
risk_ci	Risk confidence-interval method: "wilson" or "exact".
test	Association test: "auto", "none", "chisq", or "fisher".
zero_correction	Zero-cell strategy: "haldane_anscombe" or "none".
simulate_B	Number of simulations for the automatic Fisher test in a sparse table larger than 2x2.
digits	Number of decimal places.
format	Output format: "table" (default) or a plain console "tibble".

Value

A gt_twobytwo object.

Examples

```
crosstabs(mtcars, row = am, col = vs)
```

---

customise_table *Customize a gtstats table*

Description

Apply titles, labels, alignment, emphasis, colours, and a predefined visual theme to a table produced by gtstats.

Usage

```
customise_table(
  x,
  engine = c("flextable", "gt"),
  theme = c("default", "journal", "classic", "minimal", "compact"),
  title = NULL,
  subtitle = NULL,
  source_note = NULL,
  col_labels = NULL,
  row_labels = NULL,
  level_labels = NULL,
  align = NULL,
  hide_cols = NULL,
  bold_cols = NULL,
  italic_cols = NULL,
  font_size = NULL,
  font = NULL,
  width = NULL,
  row_striping = NULL,
  accent_color = NULL,
  stripe_color = NULL,
  bold_labels = TRUE,
  show_footnotes = TRUE,
  spanning_header = NULL,
  footnotes = NULL,
  borders = c("horizontal", "all", "minimal"),
  density = c("standard", "compact", "spacious"),
  column_widths = NULL,
  pvalue_style = c("threshold", "fixed", "scientific"),
  pvalue_digits = 3,
  pvalue_threshold = 0.001,
  pvalue_prefix = FALSE
)
```

Arguments

x	A supported gtstats result, rendered flextable, or rendered gt_tbl.
engine	Rendering engine used when x is an unrendered result. "flextable" is the default; use "gt" for HTML-oriented workflows.
theme	Visual theme: "default", "journal", "classic", "minimal", or "compact".
title, subtitle	Optional title and subtitle.
source_note	Optional note below the table.
col_labels, row_labels, level_labels	Named character vectors for relabelling.
align	Named list of left-, right-, or centre-aligned columns.
hide_cols	Columns to hide.

bold_cols, italic_cols
: Columns to emphasise.

font_size
: Font size in pixels.

font
: Optional font family.

width
: Table width as a percentage from 0 to 100.

row_striping
: Logical; apply alternating row shading.

accent_color, stripe_color
: Optional table colours.

bold_labels
: Logical; bold variable labels when rendering a raw result.

show_footnotes
: Logical; retain explanatory footnotes when rendering a raw result.

spanning_header
: Optional spanning heading. Supply one character value to span all result columns, or a named list/vector mapping displayed headings to completed column names.

footnotes
: Optional additional footer notes.

borders
: Border style: "horizontal", "all", or "minimal".

density
: Cell density: "standard", "compact", or "spacious".

column_widths
: Optional named numeric vector of column widths in inches for flextable output.

pvalue_style
: P-value style for summary-table results: "threshold", "fixed", or "scientific".

pvalue_digits
: Number of displayed p-value digits.

pvalue_threshold
: Threshold displayed using a less-than sign.

pvalue_prefix
: Logical; prepend p = to ordinary p-values.

Value

A styled flextable by default, or a gt_tbl when engine = "gt" or x is already a gt table.

Examples

```
result <- summary_table(mtcars, include = c(mpg, wt))
customise_table(result, title = "Vehicle characteristics")
```

---

denominators_stats *Inspect statistical denominators*

Description

Return a transparent audit of the observations, missing values, numerators, denominators, and denominator rules used in a gtstats result.

Usage

```
denominators_stats(
  x,
  format = c("table", "tibble"),
  title = "Denominator audit",
  subtitle = NULL,
  view = c("readable", "audit")
)
```

Arguments

x	A gtstats result.
format	Output format: "table" (default) or "tibble".
title, subtitle	Optional table heading used for format = "table".
view	Either "readable" (the default, plain-language headings) or "audit" (raw field names retained by the analysis object).

Value

A tibble or gt_tbl.

Examples

```
result <- proportion_stats(mtcars, var = vs, by = am)
denominators_stats(result)
```

---

describe_data *Understand a dataset before analysis*

Description

Create a concise, clinically oriented first look at a dataset. One row is returned per variable, combining its label, detected type, completeness, cardinality, a type-specific overview, and range or levels. Potential data-quality findings are kept separately in $issues.

Usage

```
describe_data(data, vars = NULL, digits = 2, format = c("table", "tibble"))
```

Arguments

data	A data.frame.
vars	Optional character vector of variables. Default is all variables.
digits	Number of decimal places in concise numeric summaries.
format	Output format: "table" (default) or "tibble".

Details

`describe_data()` deliberately does not assess distributional assumptions or recommend inferential tests. Use `assess_distribution()` for the shape of selected continuous variables, `summary_table()` for detailed descriptive statistics, and `compare_groups()` for inferential comparisons.

Value

With `format = "table"`, a `gt_describe` object that prints as a publication-ready table. `$summary` is the concise variable overview and `$issues` contains only findings requiring review. With `format = "tibble"`, the concise summary tibble is returned directly.

Examples

```
describe_data(mtcars)
describe_data(mtcars, vars = c("mpg", "cyl", "am"))
to_gt(describe_data(mtcars))
```

---

diagnostics_stats *Inspect statistical diagnostics*

Description

Return diagnostic checks, observed values, thresholds, and interpretations retained by a `gtstats` result.

Usage

```
diagnostics_stats(
  x,
  format = c("table", "tibble"),
  title = "Diagnostics",
  subtitle = NULL,
  view = c("readable", "audit")
)
```

Arguments

`x`	A `gtstats` result.
`format`	Output format: `"table"` (default) or `"tibble"`.
`title, subtitle`	Optional table heading used for `format = "table"`.
`view`	Either `"readable"` (the default, plain-language headings) or `"audit"` (raw diagnostic codes retained by the analysis object).

Value

A tibble or `gt_tbl`.

Examples

```
result <- compare_groups(mtcars, variable = vs, group = am)
diagnostics_stats(result)
```

`effect_size` *Estimate an effect size*

Description

Quantify the magnitude of a group difference or association without adding the full hypothesis-test output produced by `compare_groups()`. For directional two-group measures, the Contrast column names the grouping variable and reports first group minus second group. Factor order is therefore meaningful. Cramer's V and omnibus measures have no direction.

Usage

```
effect_size(
  data,
  variable,
  group,
  method = c("auto", "hedges_g", "rank_biserial", "omega_squared", "epsilon_squared",
    "cramers_v"),
  paired = FALSE,
  id = NULL,
  conf.level = 0.95,
  interpretation = FALSE,
  digits = 2,
  format = c("table", "tibble")
)
```

Arguments

`data`	A data frame.
`variable`	Outcome or response variable.
`group`	Grouping variable.
`method`	Effect-size method: `"auto"`, `"hedges_g"`, `"rank_biserial"`, `"omega_squared"`, `"epsilon_squared"`, or `"cramers_v"`.
`paired`	Logical; whether the two-group comparison is paired.
`id`	Pair or participant identifier required when `paired = TRUE`.
`conf.level`	Confidence level for supported intervals.
`interpretation`	Logical; display a conventional magnitude label. These labels are generic teaching aids and are not clinical importance thresholds.
`digits`	Number of decimal places.
`format`	Output format: `"table"` (default) or a plain console `"tibble"`.

Details

The default `method = "auto"` selects one measure from the outcome and comparison structure:

- Hedges' g for two-group parametric comparisons
- rank-biserial correlation for two-group rank comparisons
- omega-squared for comparisons involving more than two continuous groups
- epsilon-squared when a multi-group rank method is requested
- Cramer's V for categorical associations

Risk ratios, odds ratios, and risk differences are intentionally not duplicated here; use `crosstabs()` for those epidemiological measures.

Value

A publication-ready `gt_effect` object containing `summary`, `table`, `inputs`, `method`, `assumptions`, `diagnostics`, `denominators`, and `notes`.

Examples

```
effect_size(mtcars, variable = mpg, group = am)

effect_size(
  mtcars,
  variable = mpg,
  group = am,
  method = "hedges_g",
  interpretation = TRUE
)
```

---

epi_table *Outbreak and surveillance summary table*

Description

Build a publication-ready epidemiology table from either individual line-list records or aggregate numerator/denominator data. Unlike `summary_table()`, this function always makes the event and denominator explicit and always reports a confidence interval.

Usage

```
epi_table(
  data,
  outcomes = NULL,
  by = NULL,
  event = NULL,
  numerator = NULL,
```

```
  denominator = NULL,
  label = NULL,
  person_time = NULL,
  measure = c("proportion", "prevalence", "attack_rate", "incidence_rate"),
  multiplier = NULL,
  ci_method = c("wilson", "exact"),
  conf.level = 0.95,
  p_value = FALSE,
  p_adjust = c("none", "holm", "bonferroni", "BH", "fdr"),
  effects = "none",
  layout = c("auto", "wide", "long"),
  digits = 1,
  format = c("table", "tibble")
)
```

Arguments

`data`	A data frame.
`outcomes`	Line-list outcome variables selected with tidyselect syntax.
`by`	Optional grouping variable.
`event`	Event level to count. Supply one value for all outcomes or a named vector, for example `c(infected = "Yes", admitted = "Yes")`.
`numerator, denominator`	Aggregate count and denominator columns. For an incidence rate, `denominator` is accumulated person-time.
`label`	Optional aggregate outcome-label column or a single text label.
`person_time`	Optional line-list person-time column. Required when `measure = "incidence_rate"`.
`measure`	One of `"proportion"`, `"prevalence"`, `"attack_rate"`, or `"incidence_rate"`.
`multiplier`	Scale used for estimates. Defaults to 100 for proportions, prevalence and attack rates, and 1,000 for incidence rates.
`ci_method`	Binomial interval method: `"wilson"` or `"exact"`. Incidence rates always use an exact Poisson interval.
`conf.level`	Confidence level.
`p_value`	Add an association/rate-comparison p-value when by is used.
`p_adjust`	Optional multiplicity adjustment passed to `stats::p.adjust()`.
`effects`	Optional two-group effect measures. Use `"none"` (default), `"all"`, or any of `"rr"`, `"rd"`, and `"or"`. Incidence-rate tables support `"irr"`.
`layout`	`"auto"`, `"wide"`, or `"long"`. Auto uses wide output for four or fewer groups and long output otherwise.
`digits`	Number of decimal places.
`format`	`"table"` (default) or `"tibble"`.

Value

A `gt_epi_table` object containing `$summary`, `$table`, `$denominators`, `$p_values`, `$effects`, `$inputs`, and `$notes`.

Examples

```
epi_table(
  birthwt, outcomes = low, by = smoke,
  event = "Low birth weight", measure = "prevalence"
)

aggregate_data <- data.frame(
  ward = c("A", "B"), cases = c(12, 7), population = c(80, 65)
)
epi_table(
  aggregate_data, numerator = cases, denominator = population,
  by = ward, label = "Influenza", measure = "attack_rate"
)
```

`gtstats_app` *Launch the gtstats graphical interface*

Description

Open a guided Shiny interface for the most common **gtstats** workflows: inspecting and preparing a dataset, assessing continuous-variable distributions and spread, building a summary table, creating outbreak or surveillance tables, comparing groups, producing correlation tables, and producing a cross-tabulation. Data can be selected from the current R environment, loaded from a teaching dataset, or uploaded. Results, plots, and generated R code can be downloaded. The table customiser can also wrap an already calculated results data frame with `as_stats_table()` without recalculating its values. Excel uploads are available when the suggested **rio** package is installed.

Usage

```
gtstats_app(..., launch.browser = NULL)
```

Arguments

`...`	Additional arguments passed to `shiny::runApp()`.
`launch.browser`	Logical or a function passed to `shiny::runApp()`. When called from RStudio, the default opens in the RStudio Viewer; otherwise it opens a browser in an interactive R session.

Details

The app is a companion to the package's code-first workflow. It creates reproducible R code for every analysis, so users can begin in the interface and continue in an R script when they are ready.

Shiny is a suggested dependency and is loaded only when `gtstats_app()` is called. It is therefore not required for ordinary use of gtstats. While the app is open, R runs the local Shiny session and the console shows a `Listening on ...` message. This is expected. Click **Close app** in the bottom-right corner to stop the session cleanly and return to the R prompt.

Value

Invisibly returns the result of `shiny::runApp()`.

Examples

```
if (interactive()) {
  gtstats_app()
}
```

`outbreak_data` *Oswego foodborne-outbreak line list*

Description

A beginner-friendly copy of the 1940 Oswego County church-supper outbreak line list supplied by the US Centers for Disease Control and Prevention (CDC) with Epi Info. The 75 interview records are unchanged. Variable names use snake case, `Y/N` values are labelled `Yes`/`No`, and variable labels have been added.

Usage

```
outbreak_data
```

Format

A data frame with 75 rows and 21 variables. It contains participant age and sex, illness status and onset information, meal time, and indicators for foods and drinks consumed. `ill` is the outcome; food variables such as `vanilla_ice_cream` can be used as exposures.

Source

CDC Epi Info teaching data, documented in the Epi Info 6 User's Guide. The source material is a US Government work available without charge. CDC does not endorse gtstats. See the CDC agency materials policy.

Examples

```
epi_table(
  outbreak_data,
  outcomes = ill,
  by = vanilla_ice_cream,
  event = "Yes",
  measure = "attack_rate",
  p_value = TRUE,
  effects = "all"
)
```

paired_data *Paired follow-up teaching data*

Description

Simulated long-format data for paired analyses. Each participant has a Baseline and Follow-up record. `pain_score` supports a paired t-test example, `days_off_work` is skewed for a signed-rank example, and `symptom_present` supports McNemar's test.

Usage

```
paired_data
```

Format

A data frame with 180 rows (90 complete pairs) and 5 variables: `id`, `visit`, `pain_score`, `days_off_work`, and `symptom_present`.

Examples

```
compare_groups(
  paired_data, variable = pain_score, group = visit,
  paired = TRUE, id = id
)
compare_groups(
  paired_data, variable = symptom_present, group = visit,
  paired = TRUE, id = id
)
```

plot_compare *Plot a group comparison*

Description

Create a publication-ready comparison plot using a sensible visual selected from the outcome type and study design.

Usage

```
plot_compare(
  data,
  variable,
  group,
  paired = FALSE,
  id = NULL,
  type = c("auto", "box", "bar"),
```

```
  display = c("proportion", "count"),
  show_points = TRUE,
  show_p = FALSE,
  test = c("auto", "t_test", "welch_t", "wilcox", "anova", "welch_anova", "kruskal",
    "chisq", "fisher", "mcnemar", "rm_anova", "friedman", "cochran_q"),
  var_equal = FALSE,
  palette = NULL,
  base_size = 14,
  title = NULL,
  caption = NULL,
  xlab = NULL,
  ylab = NULL,
  legend_title = NULL
)
```

Arguments

data	A data frame.
variable	Outcome or response variable, supplied as a bare name or character string.
group	Categorical grouping variable, supplied as a bare name or character string.
paired	Logical; whether observations are paired or repeatedly measured.
id	Participant identifier required when paired = TRUE.
type	Plot type: "auto", "box", or "bar".
display	Categorical display: within-group "proportion" or "count".
show_points	Logical; show individual observations for continuous data.
show_p	Logical; add the selected test and p-value as a plot caption.
test	Test passed to compare_groups() when show_p = TRUE.
var_equal	Logical; passed to compare_groups() when show_p = TRUE. With test = "auto", TRUE selects the equal-variance parametric route for independent, non-skewed continuous outcomes. Default FALSE retains Welch methods; no variance test is performed.
palette	Optional character vector of colours. It must contain at least one colour per displayed group or outcome level.
base_size	Base font size.
title, caption	Optional plot title and caption.
xlab, ylab	Optional axis labels.
legend_title	Optional legend title.

Details

The minimal call is plot_compare(data, variable, group). Continuous outcomes are shown as boxplots with individual observations, categorical outcomes as stacked within-group proportions, and paired continuous outcomes as participant-level connected observations. The returned object is a standard ggplot and can be customized with ordinary ggplot2 layers.

When show_p = TRUE, the annotation is obtained from compare_groups() with the same test, var_equal, paired, and id settings. The test name is always shown with the p-value. The annotation and plotted denominators use the same complete observations; non-finite continuous values are excluded.

Value

A ggplot object.

Examples

```
plot_compare(mtcars, variable = mpg, group = am)

plot_compare(
  mtcars,
  variable = vs,
  group = am,
  display = "proportion",
  show_p = TRUE
)
```

---

plot_correlation *Plot the relationship between two continuous variables*

Description

Create a publication-ready scatterplot that is aligned with correlation(). The minimal call is plot_correlation(data, x, y). Incomplete pairs are excluded and the analysed number of complete pairs is shown in the caption.

Usage

```
plot_correlation(
  data,
  x = NULL,
  y = NULL,
  method = c("auto", "pearson", "spearman"),
  trend = c("auto", "linear", "smooth", "none"),
  show_ci = TRUE,
  show_correlation = FALSE,
  conf.level = 0.95,
  digits = 2,
  point_color = "#4472C4",
  line_color = "#ED7D31",
  base_size = 14,
  title = NULL,
  caption = NULL,
```

```
  xlab = NULL,
  ylab = NULL,
  triangle = NULL,
  show_diagonal = NULL,
  show_values = TRUE,
  low_color = "#355C7D",
  mid_color = "#FFFFFF",
  high_color = "#C06C5B"
)
```

Arguments

`data`	A data frame, or a matrix result returned by `correlation()`.
`x`, `y`	Continuous variables, supplied as bare names or character strings.
`method`	Correlation method: `"auto"`, `"pearson"`, or `"spearman"`.
`trend`	Fitted trend: `"auto"`, `"linear"`, `"smooth"`, or `"none"`.
`show_ci`	Logical; display the confidence band around a fitted trend.
`show_correlation`	Logical; report the correlation result in the caption.
`conf.level`	Confidence level passed to `correlation()`.
`digits`	Number of decimal places used in the correlation annotation.
`point_color`, `line_color`	Colours used for observations and the trend.
`base_size`	Base font size.
`title`, `caption`	Optional plot title and caption.
`xlab`, `ylab`	Optional axis labels.
`triangle`	Matrix cells to show when `data` is a correlation-matrix result: `"lower"`, `"upper"`, or `"full"`. The matrix object's setting is inherited when `NULL`.
`show_diagonal`	Logical; show self-correlations in a matrix heatmap. The matrix object's setting is inherited when `NULL`.
`show_values`	Logical; print coefficients inside heatmap cells.
`low_color`, `mid_color`, `high_color`	Colours used by the matrix heatmap.

Details

With `trend = "auto"`, a linear trend is used for Pearson correlation and a smooth trend for Spearman correlation. Set `trend = "none"` to display the observations alone. When `show_correlation = TRUE`, the caption reports the same method, coefficient, confidence interval when available, and p-value as `correlation()`. The returned object is a standard `ggplot`, so ordinary ggplot2 layers can be added.

Value

A `ggplot` object.

Examples

```
plot_correlation(mtcars, x = mpg, y = wt)

plot_correlation(
  mtcars,
  x = mpg,
  y = wt,
  show_correlation = TRUE
)

matrix_result <- correlation(mtcars, vars = c(mpg, disp, hp, wt))
plot_correlation(matrix_result)
```

---

print.gtstats_summary *Print a descriptive table*

Description

Print a completed `gtstats_summary` as a publication-ready `flextable`. An empty builder instead prints a short instruction explaining how to add rows.

Usage

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

Arguments

`x`	A `gtstats_summary` object.
`...`	Further arguments passed to methods.

Details

Use `format = "tibble"` in `summary_table()` for a plain console table, or call `to_gt()` when an HTML-oriented `gt` table is required.

Value

The input object, invisibly.

Examples

```
x <- summary_table(mtcars, by = am, overall = TRUE)
print(x)
```

print.gt_compare *Print a gtstats compare object*

Description

Print a publication-ready comparison table.

Usage

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

Arguments

x	A gt_compare object.
...	Further arguments passed to methods.

Details

The print method renders the concise publication table. Detailed numerical results and audit information remain available in the object components.

Value

The input object, invisibly.

Examples

```
x <- compare_groups(mtcars, variable = mpg, group = am)
print(x)
```

print.gt_correlation *Print a gtstats correlation object*

Description

Print a publication-ready correlation table.

Usage

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

Arguments

x	A gt_correlation object.
...	Further arguments passed to methods.

Details

Detailed numerical results and audit information remain available in the object components.

Value

The input object, invisibly.

Examples

```
x <- correlation(mtcars, x = mpg, y = wt)
print(x)
```

---

print.gt_data_table *Print an already summarised gtstats table*

Description

Print an already summarised gtstats table

Usage

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

Arguments

x	A gt_data_table object.
...	Further arguments passed to to_flextable().

Value

The input object, invisibly.

print.gt_describe *Print a gtstats describe object*

Description

Print the publication-ready table stored by a `gt_describe` object.

Usage

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

Arguments

`x`	A `gt_describe` object.
`...`	Further arguments passed to methods.

Details

The underlying concise tibble remains available in `$summary`, and focused data-quality findings are available in `$issues`.

Value

The input object, invisibly.

Examples

```
x <- describe_data(mtcars)
print(x)
```

print.gt_distribution *Print a gtstats distribution object*

Description

Print publication-ready distribution diagnostics and recommendations.

Usage

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

Arguments

x	A gt_distribution object.
...	Further arguments passed to methods.

Details

The detailed diagnostic table includes the suggested presentation beside the numerical diagnostics. With groups, the common variable-level suggestion is shown once beside the first group row. Machine-readable results remain available in $summary and $recommendations.

Value

The input object, invisibly.

Examples

```
x <- assess_distribution(mtcars, vars = c("mpg", "wt"))
print(x)
```

---

print.gt_effect *Print a gtstats effect-size object*

Description

Print the publication-ready table stored by a gt_effect object.

Usage

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

Arguments

x	A gt_effect object.
...	Further arguments passed to to_gt().

Value

The input object, invisibly.

Examples

```
x <- effect_size(mtcars, variable = mpg, group = am)
print(x)
```

print.gt_epi_table *Print an outbreak and surveillance table*

Description

Print an outbreak and surveillance table

Usage

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

Arguments

x	A gt_epi_table object.
...	Further arguments passed to to_flextable().

Value

The input object, invisibly.

print.gt_prop *Print a gtstats proportion object*

Description

Print a compact console preview of a gt_prop object.

Usage

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

Arguments

x	A gt_prop object.
...	Further arguments passed to methods.

Details

The print method shows the dataset name, variable, selected level, optional grouping variable, and the display-ready proportion table.

Value

The input object, invisibly.

Examples

```
x <- proportion_stats(mtcars, var = vs, by = am)
print(x)
```

`print.gt_rate` *Print a gtstats rate object*

Description

Print a compact console preview of a `gt_rate` object.

Usage

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

Arguments

`x`	A `gt_rate` object.
`...`	Further arguments passed to methods.

Details

The print method shows the dataset name, event variable, person-time variable, optional grouping variable, and the display-ready rate table.

Value

The input object, invisibly.

Examples

```
df <- data.frame(
  event = c(1, 0, 1, 0, 1, 1),
  ptime = c(10, 12, 8, 9, 11, 7),
  arm = c("A", "A", "A", "B", "B", "B")
)
x <- rate_stats(df, event = event, time = ptime, by = arm)
print(x)
```

print.gt_twobytwo *Print a gtstats 2x2 table object*

Description

Print a compact console preview of a gt_twobytwo object.

Usage

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

Arguments

x	A gt_twobytwo object.
...	Further arguments passed to methods.

Details

The print method shows the dataset name, row/reference definition, column/event definition, and the display-ready 2x2 epidemiology table.

Value

The input object, invisibly.

Examples

```
x <- crosstabs(mtcars, row = am, col = vs)
print(x)
```

print.gt_variance *Print a gtstats variance object*

Description

Print publication-ready variance diagnostics. Group-level sample sizes, standard deviations, variances, and observed spread ratios are displayed; the underlying values and explanatory metadata remain available in $summary and $diagnostics.

Usage

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

Arguments

x	A gt_variance object.
...	Further arguments passed to methods.

Value

The input object, invisibly.

Examples

```
x <- assess_variance(mtcars, vars = c(mpg, wt), by = am)
print(x)
```

---

proportion_stats *Proportion statistics*

Description

Calculate a proportion and confidence interval, optionally within groups. This is the descriptive-statistics counterpart to rate_stats().

Usage

```
proportion_stats(
  data,
  var,
  by = NULL,
  level = NULL,
  conf.level = 0.95,
  ci_method = c("wilson", "exact"),
  display = c("n_percent", "percent", "n_over_N_percent"),
  digits = 1,
  format = c("table", "tibble")
)
```

Arguments

data	A data frame.
var	Binary or categorical variable whose selected level is counted.
by	Optional categorical grouping variable.
level	Outcome level to count. A sensible event level is selected when omitted.
conf.level	Confidence level for the interval.
ci_method	Confidence-interval method: "wilson" (default) or "exact".

display	Estimate display: "n_percent", "percent", or "n_over_N_percent". The publication table places the confidence interval in a separate column; with by, each group spans its estimate and interval columns.
digits	Number of decimal places.
format	Output format: "table" (default) or a plain console "tibble".

Value

A gt_prop object containing numeric results and a display table.

Examples

```
proportion_stats(mtcars, var = vs)
proportion_stats(mtcars, var = vs, by = am)
```

---

rate_stats *Incidence rate with exact Poisson confidence interval*

Description

Compute an incidence or event rate with exact Poisson confidence intervals, either overall or within groups defined by a categorical variable.

Usage

```
rate_stats(
  data,
  event,
  time,
  by = NULL,
  multiplier = 1000,
  time_label = NULL,
  conf.level = 0.95,
  digits = 1,
  format = c("table", "tibble")
)
```

Arguments

data	A data.frame.
event	Event variable. Can be supplied as a bare name or as a character string. May be binary (0/1, TRUE/FALSE) or a count variable.
time	Person-time variable. Can be supplied as a bare name or as a character string. Must be numeric and non-negative.
by	Optional grouping variable. Can be supplied as a bare name or as a character string.

multiplier	Numeric multiplier used to scale the rate, for example 1000 or 100000. Default is 1000.
time_label	Optional readable unit for accumulated time, such as "person-years" or "catheter-days". Defaults to "person-time".
conf.level	Confidence level for the interval. Default is 0.95.
digits	Number of decimal places used when formatting rates. Default is 1.
format	Output format: "table" (default) or a plain console "tibble".

Details

This function is designed for simple epidemiological summaries where events are counted over a denominator of person-time. The event variable may be a binary indicator, a logical variable, or a count variable. The time variable must be numeric, finite, and non-negative. A zero accumulated person-time denominator is retained and clearly marked as not estimable rather than silently converted to a rate.

When a grouping variable is supplied, rates are calculated separately within each group. Confidence intervals are calculated using stats::poisson.test(). The publication table uses each group as a spanning header, with separate columns for events, accumulated time, rate, and confidence interval. The tidy long-form numerical results remain available in $summary.

Value

A gt_rate object containing:

- inputs — function inputs and settings
- summary — detailed summary table
- table — display-ready table
- notes — explanatory note
- call — matched function call

Examples

```
df <- data.frame(
  event = c(1, 0, 1, 0, 1, 1),
  ptime = c(10, 12, 8, 9, 11, 7),
  arm = c("A", "A", "A", "B", "B", "B")
)

rate_stats(df, event = event, time = ptime)

rate_stats(df, event = event, time = ptime, by = arm)

to_gt(rate_stats(df, event = event, time = ptime, by = arm))
```

save_output *Save a gtstats table or plot*

Description

Save a gtstats result, rendered flextable, rendered gt_tbl, or ggplot2 plot. A named list of gtstats results, flextables, and plots can be combined into one Word document. The object determines the export route automatically. The file type is inferred from filename.

Usage

```
save_output(
  x,
  filename,
  path = NULL,
  title = NULL,
  subtitle = NULL,
  bold_labels = TRUE,
  show_footnotes = TRUE,
  zoom = 2,
  expand = 5,
  vwidth = 992,
  vheight = 744,
  width = 8,
  height = 6,
  units = "in",
  dpi = 300,
  bg = "white",
  page_break = TRUE,
  quiet = FALSE,
  ...
)
```

Arguments

x	A gtstats result, rendered flextable, rendered gt_tbl, ggplot2 plot, or a named list of tables and plots for a combined Word report.
filename	Output filename including a supported extension.
path	Optional output directory. A bare filename requires an explicit path; use tempdir() for temporary output. Alternatively, supply a full or relative path in filename.
title, subtitle	Optional table title and subtitle.
bold_labels	Logical; bold variable labels in tables.
show_footnotes	Logical; include explanatory table footnotes.
zoom, expand	Image-export controls for tables.

vwidth, vheight	Browser viewport dimensions for table image export.
width, height	Plot dimensions.
units	Dimension units passed to ggplot2::ggsave() for plots.
dpi	Output resolution for plots.
bg	Plot background colour.
page_break	Logical; when saving a list to Word, start each output after the first on a new page.
quiet	Logical; suppress the saved-path message.
...	Additional arguments passed to the relevant underlying save method.

Value

The normalized saved path, invisibly.

Examples

```
table <- summary_table(mtcars) |> add_summary(vars = c(mpg, wt))
save_output(table, "summary.html", path = tempdir())

plot <- plot_compare(mtcars, variable = mpg, group = am)
save_output(plot, "comparison.png", path = tempdir())

save_output(
  list("Table 1" = table, "Comparison plot" = plot),
  "statistical-report.docx",
  path = tempdir()
)
```

---

summary_table *Create a summary table builder*

Description

Create the descriptive foundation of a publication-ready table. Add further layers only when needed: add_ci() for confidence intervals, add_p() for statistical comparisons, and specialist helpers such as add_proportion(), add_rate(), add_total(), and add_row().

Usage

```
summary_table(
  data,
  by = NULL,
  include = NULL,
  overall = FALSE,
  statistic = "recommended",
```

```
  categorical = c("n_percent", "n_over_N_percent", "n", "percent"),
  categorical_layout = c("combined", "separate"),
  show_dichotomous = c("all_levels", "single_row"),
  value = NULL,
  percent = c("column", "row", "overall"),
 overall_categorical = c("auto", "n_percent", "n_over_N_percent", "n", "percent"),
  digits = 1,
  missing = c("ifany", "always", "no", "as_category"),
  layout = c("compact", "separate"),
  label = NULL,
  conf.level = 0.95,
  format = c("table", "tibble")
)
```

Arguments

`data`	A data.frame.
`by`	Optional grouping variable. Can be supplied as a bare name or as a character string. The grouping variable must be categorical, binary, or ordinal.
`include`	Optional variables to summarise immediately. Supply bare names, such as `c(age, sex, bmi)`, or a character vector. Mixed variable types can be selected together. When omitted, an empty advanced builder is returned.
`overall`	Overall-column setting. Use `FALSE` to omit it, `"first"` to place it before the grouped columns, or `"last"` to place it after them. `TRUE` is accepted as a shorthand for `"first"`.
`statistic`	Continuous summary format: `"recommended"`, `"mean_sd"`, `"mean_se"`, `"mean_ci"`, `"median_iqr"`, or `"both"`. A single value applies to all continuous variables. In a named vector, `continuous` supplies a fallback for every continuous variable and variable names supply exceptions, for example `c(continuous = "mean_sd", lwt = "median_iqr")`. Without a `continuous` fallback, unnamed variables use the recommended summary.
`categorical`	Categorical display: `"n_percent"`, `"n_over_N_percent"`, `"n"`, or `"percent"`.
`categorical_layout`	Categorical column layout. `"combined"` (default) displays n (%). `"separate"` places n and % in distinct child columns for categorical-only tables without confidence intervals.
`show_dichotomous`	Binary-variable display. `"all_levels"` (default) shows both levels; `"single_row"` shows one selected event level as a compact row.
`value`	Optional named character vector or list selecting the event level for compact binary rows, for example `c(smoke = "Yes")`. Unspecified binary variables use their second declared or sorted level.
`percent`	Percentage denominator: `"column"`, `"row"`, or `"overall"`.
`overall_categorical`	Categorical display in the Overall column. `"auto"` (default) uses counts only with row percentages, because the Overall percentage would be redundant, and

	otherwise follows `categorical`. Explicit choices are `"n_percent"`, `"n_over_N_percent"`, `"n"`, or `"percent"`.
`digits`	One number applied throughout, or a named numeric vector using `continuous`, `percent`, and `ci`.
`missing`	Missing-value display and percentage handling. `"ifany"` shows a missing row only when needed, `"always"` always shows it, and `"no"` hides it; these three calculate observed-category percentages from non-missing values. `"as_category"` displays missing values as a category and includes them when calculating categorical percentages.
`layout`	Table layout. `"compact"` keeps each summary in one cell. `"separate"` requests summary and CI child columns beneath each cohort header. Those child columns appear only after confidence intervals are added; choosing the layout alone does not create empty CI columns.
`label`	Optional named character vector overriding variable labels.
`conf.level`	Confidence level used when `statistic = "mean_ci"`.
`format`	Display format: `"table"` (default) or `"tibble"`. The builder remains composable; this option changes how the completed object prints without discarding its audit components.

Details

For the usual Table 1 workflow, select all variables together with `include`. Continuous, binary, categorical, and ordinal variables are detected automatically and added using beginner-friendly defaults. There is no need to add continuous and categorical variables separately.

When `include = NULL`, an empty builder is returned for specialist row-only workflows. Printing a completed object automatically displays a publication-ready `flextable`; call `to_gt()` when an HTML-oriented `gt` table is required.

A grouping variable may be supplied to create one column per group. An optional `overall` column can also be requested for later use.

Value

A `gtstats_summary` object containing the source data, structural settings, and placeholders for table components.

Examples

```
summary_table(
  mtcars,
  by = am,
  include = c(mpg, wt, cyl, vs),
  overall = TRUE
)

summary_table(
  mtcars,
  by = am,
  include = c(mpg, wt, cyl, vs),
```

```
  overall = TRUE
) |>
  add_p()

# Percentages without decimals and Overall displayed last
summary_table(
  mtcars,
  by = am,
  include = c(mpg, wt, cyl, vs),
  overall = "last",
  digits = c(continuous = 1, percent = 0)
)

# Add confidence intervals as a visible layer
summary_table(
  mtcars,
  include = c(cyl, vs),
  categorical = "percent",
  layout = "separate"
) |>
  add_ci()

# Compact binary rows, with an explicit event where required
summary_table(
  mtcars,
  include = c(mpg, vs, am),
  show_dichotomous = "single_row",
  value = c(vs = "1", am = "1")
)

# Include Missing in a categorical percentage denominator
missing_example <- mtcars
missing_example$vs[1:3] <- NA
summary_table(
  missing_example,
  include = vs,
  missing = "as_category"
)
```

`surveillance_data` *Archived weekly US hospital-admission surveillance data*

Description

A teaching extract from CDC's archived weekly United States COVID-19 hospital-admission surveillance dataset. It contains one record per health service area for the report dated 12 January 2024. The CDC county file repeats the same area-level metrics for each constituent county; this copy retains one exact metric record per health service area to prevent double counting. Names and labels were simplified, dates were parsed, and the source spelling `Insuficient Data` was corrected in the factor label.

Usage

```
surveillance_data
```

Format

A data frame with 808 rows and 9 variables:

health_service_area_id CDC health service area identifier.

population Health service area population used as denominator.

report_date CDC report date.

week_end_date End date of the surveillance week.

mmwr_week MMWR epidemiological week.

mmwr_year MMWR year.

admissions Weekly confirmed COVID-19 hospital admissions.

admissions_per_100k CDC-published weekly rate per 100,000.

admission_level CDC admission-level category.

Two areas have insufficient admission data and therefore missing counts and rates. The archived values are provisional teaching data and should not be used to describe current disease activity.

Source

CDC, Weekly United States COVID-19 Hospitalization Metrics by County - ARCHIVED, dataset `akn2-qxic`. The metadata identifies this dataset as `USGOV_WORKS`; it is available without charge. CDC does not endorse gtstats. See the CDC agency materials policy.

Examples

```
complete_surveillance <- subset(surveillance_data, !is.na(admissions))
epi_table(
  complete_surveillance,
  numerator = admissions,
  denominator = population,
  label = admission_level,
  measure = "incidence_rate",
  multiplier = 100000
)
```

`to_flextable` *Convert a gtstats object to flextable*

Description

Convert a supported `gtstats` object into a `flextable` for use in Word, PowerPoint, and other Office-style outputs.

Usage

```
to_flextable(
  x,
  font_size = 10,
  font = NULL,
  autofit = TRUE,
  show_footnotes = TRUE,
  title = NULL,
  subtitle = NULL
)
```

Arguments

`x`	A supported `gtstats` object containing a `$table` component, such as an object returned by `summary_table()`, `compare_groups()`, `proportion_stats()`, `as_stats_table()`, or related functions.
`font_size`	Font size applied to the whole table. Default is `10`.
`font`	Optional font family applied to the whole table.
`autofit`	Logical; whether column widths should be adjusted automatically using `flextable::autofit()`.
`show_footnotes`	Logical; include concise explanatory footnotes.
`title, subtitle`	Optional title and subtitle placed above the table.

Details

This function is intended for export workflows where a `gtstats` object needs to be inserted into a report or presentation. The input should be the original analytical object, not the rendered `gt` table returned by `to_gt()`.

Text columns are left-aligned and value columns are right-aligned. Notes, footnotes, and p-value method footnotes are appended to the footer when available.

Value

A `flextable` object.

Examples

```
res <- summary_table(mtcars, by = am, overall = TRUE) |>
  add_summary(vars = c(mpg, wt, cyl)) |>
  add_total() |>
  add_p()

to_flextable(res)
```

---

to_gt *Convert a gtstats result to a gt table*

Description

Render supported `gtstats` objects as formatted `gt` tables.

Usage

```
to_gt(
  x,
  title = NULL,
  subtitle = NULL,
  bold_labels = TRUE,
  show_footnotes = TRUE
)
```

Arguments

`x`	A supported `gtstats` object.
`title`	Optional table title.
`subtitle`	Optional table subtitle.
`bold_labels`	Logical; whether to bold variable labels where appropriate.
`show_footnotes`	Logical; whether explanatory footnotes should be displayed.

Details

This function is the explicit rendering bridge between analytical `gtstats` objects and presentation-ready table output. It supports descriptive, inferential, epidemiological, and table-builder objects created by the package and applies a consistent visual style using `gt`.

Supported inputs include:

- `gt_describe`
- `gt_distribution`
- `gt_variance`
- `gt_compare`

- gt_correlation
- gt_effect
- gt_prop
- gt_rate
- gt_twobytwo
- gtstats_summary
- gt_data_table

Value

A gt_tbl object.

Examples

```
to_gt(describe_data(mtcars))

to_gt(summary_table(mtcars, by = am, include = c(mpg, wt)))

to_gt(compare_groups(mtcars, variable = mpg, group = am))

to_gt(
  summary_table(
    mtcars,
    by = am,
    include = c(mpg, wt, cyl),
    overall = TRUE
  ) |>
    add_total() |>
    add_p()
)
```

---

trial_data *Three-arm clinical trial teaching data*

Description

Simulated data for demonstrating independent-group analyses. The dataset is intentionally constructed to include approximately symmetric continuous outcomes, a right-skewed outcome, an ordinal outcome, common and rare binary outcomes, correlated biomarkers, and person-time with event counts.

Usage

```
trial_data
```

Format

A data frame with 180 rows and 13 variables. `arm` has three groups; `change_score` is approximately symmetric; `hospital_days` is right-skewed; `response` is ordered; `rare_event` is sparse; and `followup_years` with `infection_events` supports rate examples.

Examples

```
compare_groups(trial_data, variable = change_score, group = arm)
compare_groups(trial_data, variable = hospital_days, group = arm)
compare_groups(trial_data, variable = rare_event, group = arm)
```

Index