Package: TernTables (via r-universe)

October 26, 2026

Type Package

Title Automated Statistical Analysis and Table Generation for Biomedical Research

Version 1.7.2.9004

Description Generates publication-ready summary tables for clinical research, supporting descriptive summaries and comparisons across two or three groups. The package streamlines the analytical workflow by detecting variable types and applying appropriate statistical tests (Welch t-test, Wilcoxon rank-sum, Welch ANOVA, Kruskal-Wallis, Chi-squared, or Fisher's exact test). Results are formatted as 'tibble' objects and can be exported to 'Word' or 'Excel' using the 'officer', 'flextable', and 'writexl' packages. Optional pairwise post-hoc testing for three-group comparisons (Games-Howell and Dunn's test) is available via the 'rstatix' package. Example data are derived from the landmark adjuvant colon cancer trial described in Moertel et al. (1990) <doi:10.1056/NEJM199002083220602>.

License MIT + file LICENSE

URL https://cran.r-project.org/package=TernTables, https://github.com/jdpreston30/TernTables, https://tern-tables.com/

BugReports https://github.com/jdpreston30/TernTables/issues

Encoding UTF-8

RoxygenNote 7.3.3

Imports cli, dplyr (>= 1.0.0), epitools, flextable (>= 0.9.0), magrittr, multcompView, officer (>= 0.4.6), rlang, rstatix, stats, stringr, tibble, withr, writexl

Suggests knitr, rmarkdown, survival, testthat (>= 3.0.0)

VignetteBuilder knitr

Depends R (>= 4.1.0)

LazyData true

Config/testthat/edition 3

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev zlib1g-dev

Repository https://jdpreston30.r-universe.dev

Date/Publication 2026-07-28 18:54:25 UTC

RemoteUrl https://github.com/jdpreston30/TernTables

RemoteRef HEAD

RemoteSha 02f9e99f86dd1607d6735b7dc8a51d8aabea2ae8

Contents

classify_normality . . . 2
print.ternP_result . . . 4
tern_colon . . . 5
tern_estimates . . . 6
tern_stats . . . 7
ternB . . . 8
ternD . . . 10
ternG . . . 16
ternP . . . 23
ternStyle . . . 26
val_format . . . 30
val_p_format . . . 31
word_export . . . 31
write_cleaning_doc . . . 36
write_methods_doc . . . 37

Index **40**

`classify_normality` *Classify variables by normality and routing decision*

Description

Applies the same normality assessment logic used internally by `ternG()` and `ternD()` and returns a tidy tibble showing per-variable (and per-group) statistics, the gate that triggered the routing decision, and the final parametric / non-parametric routing outcome.

Usage

```
classify_normality(
  data,
  vars = NULL,
  exclude_vars = NULL,
  group_var = NULL,
  consider_normality = "ROBUST"
)
```

Arguments

data	A data frame or tibble.
vars	Optional character vector of variable names to assess. If NULL (default), all assessable numeric variables are included (excluding exclude_vars and group_var).
exclude_vars	Optional character vector of variable names to exclude.
group_var	Optional name of the grouping variable (as used in ternG()). When provided, the normality assessment is performed across groups simultaneously — exactly as ternG() does. When NULL, each variable is assessed as a single vector (matching ternD()).
consider_normality	Normality assessment mode — must match what was (or will be) passed to ternG() / ternD() to guarantee identical routing. "ROBUST" (default), TRUE, or FALSE.

Details

Useful for:

- Answering reviewer questions about normality testing ("was Age normally distributed?").
- Verifying that a given variable's routing matches your expectation before running ternG() or ternD().
- Generating a supplemental normality audit table for a manuscript.

Value

A tibble with one row per variable × group (or one row per variable when group_var = NULL), containing:

variable Variable name.

group Group level, or "[all]" when no group_var is supplied.

n Non-missing sample size in this group.

skewness Sample skewness (population moments).

kurtosis Excess kurtosis (population moments; 0 for a normal distribution).

sw_p Shapiro-Wilk p-value for this group. NA when the routing decision was made at Gates 1–3 under "ROBUST", when n is outside the valid range (3–5000), or when consider_normality = FALSE.

gate Integer 1–4 indicating which gate made the routing decision under `consider_normality = "ROBUST"`, or `NA` for `TRUE` / `FALSE` modes.

gate_reason Plain-language explanation of the gate decision, naming which group(s) triggered the rule where relevant.

is_normal Logical; `TRUE` = routed to parametric (mean ± SD, t-test / ANOVA); `FALSE` = non-parametric (median [IQR], Wilcoxon / Kruskal-Wallis).

routing Human-readable routing summary: `"Parametric (mean \u00b1 SD)"` or `"Non-parametric (median [IQR])"`.

Examples

```
data(tern_colon)

# Single-group audit (ternD-style)
classify_normality(tern_colon, exclude_vars = "ID")

# Grouped audit matching a ternG call
classify_normality(tern_colon, exclude_vars = "ID", group_var = "Recurrence")

# Specific variables only
classify_normality(tern_colon,
                   vars      = c("Age", "Positive_Lymph_Nodes_n"),
                   group_var = "Recurrence")

# Using Shapiro-Wilk only (matches consider_normality = TRUE in ternG/ternD)
classify_normality(tern_colon, exclude_vars = "ID",
                   group_var          = "Recurrence",
                   consider_normality = TRUE)
```

---

print.ternP_result *Print method for ternP_result objects*

Description

Re-displays the preprocessing summary for a `ternP_result` object. Note that `ternP` already emits this summary automatically at the time it is called, so this method is most useful for reviewing the summary after the fact (e.g. typing `result` at the console later in a session).

Usage

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

Arguments

`x`	A `ternP_result` object returned by `ternP`.
`...`	Currently unused; included for S3-method compatibility.

Value

Invisibly returns `x`.

`tern_colon` *Colon Cancer Recurrence Data (Example Dataset)*

Description

A processed subset of the colon dataset restricted to the recurrence endpoint (`etype == 1`), providing one row per patient. Variables have been relabelled with clinically descriptive names and factor levels suitable for direct use in TernTables functions. This dataset is provided as a ready-to-use example for demonstrating `ternD()` and `ternG()` functionality.

Usage

```
tern_colon
```

Format

A tibble with 929 rows and 12 variables:

ID Integer patient identifier.

Age_Years Age at study entry (years).

Sex Patient sex: `"Female"` or `"Male"`.

Colonic_Obstruction Colonic obstruction present: `"N"` or `"Y"`.

Bowel_Perforation Bowel perforation present: `"N"` or `"Y"`.

Positive_Lymph_Nodes_n Number of positive lymph nodes detected.

Over_4_Positive_Nodes More than 4 positive lymph nodes: `"N"` or `"Y"`.

Tumor_Adherence Tumour adherence to surrounding organs: `"N"` or `"Y"`.

Tumor_Differentiation Tumour differentiation grade: `"Well"`, `"Moderate"`, or `"Poor"`.

Extent_of_Local_Spread Depth of tumour penetration: `"Submucosa"`, `"Muscle"`, `"Serosa"`, or `"Contiguous Structures"`.

Recurrence Recurrence status: `"No Recurrence"` or `"Recurrence"`.

Treatment_Arm Randomised treatment: `"Levamisole + 5FU"`, `"Levamisole"`, or `"Observation"`.

Source

Derived from `colon` (Laurie et al., 1989). See `colon` for full provenance. Pre-processing script: `data-raw/tern_colon.R`.

Examples

```
data(tern_colon)
head(tern_colon)
```

`tern_estimates` *Extract the tidy per-group summary estimates from a TernTables result*

Description

Long-format companion to `tern_stats`: one row per variable x group x level, holding the raw numeric summary values (counts, percentages, means, SDs, medians, quartiles) behind each displayed cell, plus the exact formatted string that was rendered.

Usage

```
tern_estimates(x)
```

Arguments

`x`	A tibble returned by `ternG` or `ternD`.

Details

For categorical variables, every level is reported — including levels the display collapses away (a binary Y/N variable shows only the "Y" row in the table, but both levels appear here).

Value

A tibble with columns `variable`, `label`, `group` (the group level, or `"Total"` for the aggregate column), `level` (factor level for categorical variables, `NA` for continuous), `n`, `pct`, `mean`, `sd`, `median`, `q1`, `q3`, and `value_fmt`. Columns not applicable to a given variable type are `NA`.

See Also

`tern_stats`

Examples

```
data(tern_colon)

tbl <- ternG(tern_colon, exclude_vars = "ID", group_var = "Recurrence",
             methods_doc = FALSE)
est <- tern_estimates(tbl)
est[est$variable == "Age_Years", ]
```

`tern_stats` *Extract the tidy per-variable statistics from a TernTables result*

Description

Returns the machine-readable record of everything `ternG` or `ternD` computed while building the display table: one row per variable, with raw un-rounded statistics in numeric columns and the matching publication-formatted strings alongside them.

Usage

```
tern_stats(x)
```

Arguments

`x`	A tibble returned by `ternG` or `ternD`.

Details

This is the supported way to consume TernTables results programmatically. It removes any need to parse the rendered table — no walking the internal `.indent` column, no allowing for the fact that continuous and categorical variables park their test results on differently-shaped rows, and no fuzzy-matching around `smart_rename`'s display-name cleaning (rows are keyed by `variable`, the original data column name).

The frame is attached to every `ternG()` / `ternD()` result as the `"tern_stats"` attribute, so a single call yields both the formatted table and the tidy data. Values are never recomputed: each row is populated from the same test object that produced the corresponding display cell, so the tidy frame cannot drift from the rendered table.

Value

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

variable Original column name in `data` (the join key).

label Display name as it appears in the rendered table, after `smart_rename` cleaning when enabled.

type `"continuous"` or `"categorical"`.

stat_type How the variable was summarized: `"mean_sd"`, `"median_iqr"`, or `"n_pct"`.

n, n_missing Observations used by the test, and observations dropped as missing.

n_levels Number of levels for categorical variables; `NA` for continuous.

test Name of the statistical test applied (`NA` for `ternD`, which performs no group comparison).

statistic, df, df2 Test statistic and degrees of freedom, un-rounded. `df2` holds the denominator df for Welch ANOVA. `NA` where the test reports none (e.g. Fisher's exact).

p_value The raw numeric P value, un-rounded.

p_fmt The same P value formatted exactly as displayed.

p_adjusted, p_adjusted_fmt Benjamini-Hochberg corrected P values, populated only when p_adjust = TRUE.

test_note Reason the test could not be run (e.g. "insufficient variation"), otherwise NA.

is_normal Normality routing decision for continuous variables; NA for categorical variables and for variables whose distribution was never assessed (e.g. force_ordinal).

sw_p Minimum Shapiro-Wilk P across groups, where computed.

or_value, or_lcl, or_ucl Odds ratio and 95% CI bounds, populated when OR_col = TRUE.

or_fmt, or_method Formatted OR string and the method used ("Fisher" or "Wald").

Statistical fields are populated whenever the underlying test ran, including when show_p = FALSE or show_test = FALSE suppressed them from the display table.

See Also

tern_estimates for the per-group, per-level summary values; ternG and ternD, whose plain_tibble argument returns this frame directly.

Examples

```
data(tern_colon)

tbl <- ternG(tern_colon, exclude_vars = "ID", group_var = "Recurrence",
             methods_doc = FALSE)

# Pretty table returned as always; tidy data one accessor away
stats <- tern_stats(tbl)
stats[, c("variable", "test", "p_value", "p_fmt")]

# Harvest specific variables without touching the rendered layout
stats[stats$variable %in% c("Age_Years", "Sex"), ]
```

---

ternB *Combine multiple ternD/ternG tables into a single Word document*

Description

Takes a list of tibbles previously created by ternD() or ternG() and writes them all into one .docx file, one table per page, preserving the exact formatting settings that were used when each table was built.

Usage

```
ternB(
  tables,
  output_docx,
  page_break = TRUE,
```

```
  methods_doc = FALSE,
  methods_filename = "TernTables_methods.docx",
  open_doc = TRUE,
  citation = TRUE,
  font_family = getOption("TernTables.font_family", "Arial")
)
```

Arguments

tables	A **list** of tibbles created by `ternD()` or `ternG()`. Must be constructed with `list()`, not `c()` (e.g. `list(T1, T2, T3)`). Each tibble must have been produced in the *current R session*; the metadata is stored in memory, not in the tibble columns.
output_docx	Output file path ending in `.docx`.
page_break	Logical; if `TRUE` (default), inserts a page break between each consecutive table.
methods_doc	Logical; if `TRUE`, writes a single methods section Word document that covers all tables in the list. Statistical test details are pooled across all tables. Default is `FALSE`.
methods_filename	Output file path for the methods document. Defaults to `"TernTables_methods.docx"` in the working directory.
open_doc	Logical; if `TRUE` (default), automatically opens each written `.docx` in the system default application after saving. Set to `FALSE` to suppress.
citation	Logical; if `TRUE` (default), appends a citation line at the bottom of each table footnote block and methods document: package version, authors, and links to the GitHub repository and web interface. Set to `FALSE` to suppress.
font_family	Character; font family for all Word output. Any font name accepted by the rendering system is valid. Can also be set via `options(TernTables.font_family = "Garamond")`. Default `"Arial"`.

Details

`ternB()` works by replaying the exact `word_export()` call that `ternD()` / `ternG()` would have made – using stored metadata attached as an attribute to each returned tibble – but directing all output into a single combined document instead of separate files.

Table captions (`table_caption`) and footnotes (`table_footnote`) specified in the original `ternD()` / `ternG()` call are reproduced automatically. You can override them by modifying the `"ternB_meta"` attribute before calling `ternB()`, though in practice it is easier to set captions and footnotes when you first build each table.

Value

Invisibly returns the path to the written Word file.

Examples

```
data(tern_colon)

T1 <- ternD(tern_colon,
            exclude_vars  = "ID",
            table_caption = "Table 1. Overall patient characteristics.",
            methods_doc   = FALSE,
            open_doc      = FALSE)

T2 <- ternG(tern_colon,
            group_var     = "Recurrence",
            exclude_vars  = "ID",
            table_caption = "Table 2. Characteristics by recurrence status.",
            methods_doc   = FALSE,
            open_doc      = FALSE)

ternB(list(T1, T2),
      output_docx = file.path(tempdir(), "combined_tables.docx"),
      open_doc    = FALSE)
```

---

ternD *Generate descriptive summary table (optionally normality-aware)*

Description

Creates a descriptive summary table with a single "Total" column format. By default (`consider_normality = "ROBUST"`), continuous variables are shown as mean +/- SD or median [IQR] based on a four-gate decision (n < 3 fail-safe, skewness/kurtosis, CLT, and Shapiro-Wilk). This can be overridden via `consider_normality` and `force_ordinal`.

Usage

```
ternD(
  data,
  vars = NULL,
  exclude_vars = NULL,
  force_ordinal = NULL,
  force_normal = NULL,
  force_continuous = NULL,
  output_xlsx = NULL,
  output_docx = NULL,
  consider_normality = "ROBUST",
  print_normality = FALSE,
  round_intg = FALSE,
  round_decimal = NULL,
  smart_rename = TRUE,
  insert_subheads = TRUE,
```

```
  factor_order = "mixed",
  methods_doc = TRUE,
  methods_filename = "TernTables_methods.docx",
  category_start = NULL,
  plain_header = NULL,
  table_font_size = 9,
  manual_italic_indent = NULL,
  manual_underline = NULL,
  table_caption = NULL,
  table_footnote = NULL,
  abbreviation_footnote = NULL,
  variable_footnote = NULL,
  index_style = "symbols",
  line_break_header = getOption("TernTables.line_break_header", TRUE),
  open_doc = TRUE,
  citation = TRUE,
  font_family = getOption("TernTables.font_family", "Arial"),
  show_missing = FALSE,
  zero_to_dash = FALSE,
  show_missingness = FALSE,
  missing_indicators = NULL,
  plain_tibble = FALSE
)
```

Arguments

`data`	Tibble with variables.
`vars`	Character vector of variables to summarize. Defaults to all except `exclude_vars`.
`exclude_vars`	Character vector to exclude from the summary.
`force_ordinal`	Character vector of variables to treat as ordinal (i.e., use median [IQR]) regardless of the `consider_normality` setting. This parameter takes priority over normality testing when `consider_normality = "ROBUST"` or `TRUE`.
`force_normal`	Character vector of variable names to treat as normally distributed, bypassing all normality assessment. Listed variables are summarized as mean ± SD regardless of the `consider_normality` setting. Takes priority over `consider_normality` but not over `force_ordinal` (if a variable appears in both, `force_ordinal` wins). Default is `NULL`.
`force_continuous`	Character vector of variables to force treatment as continuous (mean ± SD), bypassing the automatic binary `0/1` detection that would otherwise convert them to categorical Y/N. Useful when a numeric variable with only two unique values (e.g. `0/1` dose levels) should be analysed as a continuous measurement rather than a dichotomous category. Default is `NULL`.
`output_xlsx`	Optional Excel filename to export the table.
`output_docx`	Optional Word filename to export the table.

consider_normality
: Character or logical; controls routing of continuous variables to mean ± SD vs median [IQR]. "ROBUST" (default) applies a four-gate decision: (1) n < 3 → non-parametric (conservative fail-safe); (2) absolute skewness > 2 or excess kurtosis > 7 → non-parametric regardless of n; (3) n ≥ 30 → parametric via the Central Limit Theorem; (4) otherwise Shapiro-Wilk p > 0.05 → parametric. If TRUE, uses Shapiro-Wilk alone (can be over-sensitive at large n). If FALSE, defaults to mean ± SD for all numeric variables unless specified in force_ordinal.

print_normality
: Logical; if TRUE, includes Shapiro-Wilk P values as an additional column in the output. Default is FALSE.

round_intg
: Logical; if TRUE, rounds all means, medians, IQRs, and standard deviations to nearest integer (0.5 rounds up). Default is FALSE.

round_decimal
: Integer or NULL; number of decimal places for all continuous summary values (means, SDs, medians, IQRs). Overrides the default of 1 decimal place when set. Ignored when round_intg = TRUE. Default is NULL (1 decimal place).

smart_rename
: Logical; if TRUE, automatically cleans variable names and subheadings for publication-ready output using built-in rule-based pattern matching for common medical abbreviations and prefixes. Default is TRUE.

insert_subheads
: Logical; if TRUE (default), creates a hierarchical structure with a header row and indented sub-category rows for categorical variables with 3 or more levels. Binary variables (Y/N, YES/NO, or numeric 1/0 – which are auto-detected and treated as Y/N) are always displayed as a single row showing the positive/yes count regardless of this setting. Two-level categorical variables whose values are not Y/N, YES/NO, or 1/0 (e.g. Male/Female) use the hierarchical sub-row format, showing both levels as indented rows. If FALSE, all categorical variables use a single-row flat format. Default is TRUE.

factor_order
: Character; controls the ordering of factor levels in the output. "mixed" (default) applies level-aware ordering for two-level categorical variables and frequency ordering for variables with three or more levels: for any factor, factor level order is always respected regardless of the number of levels; for non-factor two-level variables, levels are sorted alphabetically; for non-factor variables with three or more levels, levels are sorted by decreasing frequency. "levels" respects the original factor level ordering for all variables; if the variable is not a factor, falls back to frequency ordering. "frequency" orders all levels by decreasing frequency (most common first).

methods_doc
: Logical; if TRUE (default), generates a methods document describing the statistical presentation used. The document contains boilerplate text for all three table types so the relevant section can be copied directly into a manuscript.

methods_filename
: Character; filename for the methods document. Default is "TernTables_methods.docx".

category_start
: Named character vector specifying where to insert category headers. Names are the header label text to display; values are the anchor variable – either the original column name (e.g. "Age_Years") or the cleaned display name (e.g. "Age (yr)"). Both forms are accepted. Example: c("Demographics" =

"Age_Years", "Clinical Measures" = "bmi"). Default is NULL (no category headers).

plain_header Named character vector, same interface as category_start. Names are the label text; values are the anchor variable to insert before. Inserts a label-only row with underline formatting and no bold, merge, or border treatments. Default NULL.

table_font_size
Numeric; font size for Word document output tables. Default is 9.

manual_italic_indent
Character vector of display variable names (post-cleaning) that should be formatted as italicized and indented in Word output – matching the appearance of factor sub-category rows. Has no effect on the returned tibble; only applies when output_docx is specified. Default is NULL.

manual_underline
Character vector of display variable names (post-cleaning) that should be formatted as underlined in Word output – matching the appearance of multi-category variable headers. Has no effect on the returned tibble; only applies when output_docx is specified. Default is NULL.

table_caption Optional character string for a table caption to display above the table in the Word document. Rendered as size 11 Arial bold, single-spaced with a small gap before the table. Default is NULL (no caption). Example: "Table 1. Patient demographics."

table_footnote Optional character string for a footnote to display below the table in the Word document. Rendered as size 6 Arial italic with a double-bar border above and below. Default is NULL (no footnote).

abbreviation_footnote
Optional character string listing abbreviations. Always printed first in the footnote block. Default NULL.

variable_footnote
Optional named character vector. Names are display variable names (case-insensitive); values are the footnote definition text. Each variable gets the next symbol appended to its name in the table, and the footnote block lists each definition below the abbreviation line. To share one footnote between multiple variables, separate their names with a pipe: c("Var A|Var B" = "Shared note text."). Default NULL.

index_style Character; "symbols" (default) uses *, dagger, double-dagger ... "alphabet" uses Unicode superscript letters. See word_export for details.

line_break_header
Logical; if TRUE (default), column headers are wrapped with \n – the first column header includes a category hierarchy label, and the sample size appears on a second line. Set to FALSE to suppress all header line breaks. Can also be set package-wide via options(TernTables.line_break_header = FALSE).

open_doc Logical; if TRUE (default), automatically opens the written Word document in the system default application after saving. Set to FALSE to suppress. Has no effect when output_docx is NULL.

`citation` Logical; if `TRUE` (default), appends a citation line at the bottom of the table footnote block and at the end of the methods document: package version, authors, and links to the GitHub repository and web interface. Set to `FALSE` to suppress.

`font_family` Character; font family name used for all Word output (table, captions, footnotes, methods document). Any font installed on the system that renders the document may be used. Popular options include `"Arial"`, `"Helvetica"`, `"Times New Roman"`, `"Garamond"`, and `"Calibri"`. Defaults to `getOption("TernTables.font_family", "Arial")`.

`show_missing` Logical; if `TRUE`, appends a `"Missing"` row after each variable's data rows showing the count and percentage of missing observations (denominator is the total N). Only emitted when at least one observation is missing. A footnote is automatically appended noting that missing values are reported. Default is `FALSE`.

`zero_to_dash` Logical; if `TRUE`, replaces any categorical cell that would display `"0 (0%)"` with `"-"` in the output table. Useful when zero counts are not meaningful to report numerically. Default is `FALSE`.

`show_missingness`
Controls whether a `"Missing, n (%)"` column is appended to the table after the `Total` column. Options:
`FALSE` (default) — no missingness column added.
`"total"` — one column appended showing the count and percentage of missing observations across all rows for each variable.
`"group"` is not supported by `ternD()` (which has no group structure); use `ternG()` with `show_missingness = "group"` instead. Missingness is computed on the raw data column so both `NA` values and string representations of missing data (e.g., `"Unknown"`, `"N/A"`) are counted. See `missing_indicators`.

`missing_indicators`
Optional character vector of string values to treat as missing in addition to (or instead of) the built-in ternP defaults. When `NULL` (default), the ternP canonical list is used. When supplied, the custom list **replaces** the ternP defaults. Matching is case-insensitive and trims whitespace.

`plain_tibble` Logical; if `TRUE`, returns the tidy per-variable statistics frame (see `tern_stats`) instead of the formatted display table — one un-indented row per variable with raw numeric statistics and their formatted counterparts in adjacent columns. Word/Excel exports and the methods document are still written if requested. Setting this is never required to reach the tidy data: the same frame is always attached to the normal return value as the `"tern_stats"` attribute, so `tern_stats(ternD(...))` yields both the pretty table and the tidy data from one call. Since `ternD()` performs no group comparison, the test-related columns (`test`, `statistic`, `df`, `p_value`) are `NA`. Default is `FALSE`.

Details

The function always returns a tibble with a single `Total (N = n)` column format, regardless of the `consider_normality` setting. The behavior for numeric variables follows this priority:

1. Variables in `force_ordinal`: Always use median [IQR]
2. When `consider_normality = "ROBUST"`: Four-gate decision (n<3 fail-safe, skewness/kurtosis, CLT, Shapiro-Wilk)

3. When consider_normality = TRUE: Use Shapiro-Wilk test to choose format
4. When consider_normality = FALSE: Default to mean +/- SD

For categorical variables, the function shows frequencies and percentages. When insert_subheads = TRUE, categorical variables with 3 or more levels are displayed with hierarchical formatting (main variable as header, levels as indented sub-rows). Binary variables (Y/N, YES/NO, or numeric 1/0 auto-detected as Y/N) always use a single-row format showing only the positive/yes count, regardless of this setting. Two-level categorical variables whose values are not Y/N, YES/NO, or 1/0 (e.g. Male/Female) also use the hierarchical sub-row format.

Value

A tibble with one row per variable (multi-row for factors), containing:

Variable Variable names with appropriate indentation

Total (N = n) Summary statistics (mean +/- SD, median [IQR], or n (%) as appropriate)

SW_p Shapiro-Wilk P values (only if print_normality = TRUE)

Two tidy, machine-readable frames are always attached as attributes: "tern_stats" (one row per variable — summary type, n, normality decision, Shapiro-Wilk P) and "tern_estimates" (long format: the underlying counts, percentages, means, SDs, medians and quartiles behind each displayed cell). Access them with tern_stats and tern_estimates rather than parsing the display table.

See Also

tern_stats, tern_estimates

Examples

```
data(tern_colon)

# Basic descriptive summary
ternD(tern_colon, exclude_vars = c("ID"), methods_doc = FALSE)

# With normality-aware formatting and category section headers
ternD(tern_colon, exclude_vars = c("ID"), methods_doc = FALSE,
      category_start = c("Patient Demographics"  = "Age (yr)",
                         "Tumor Characteristics" = "Positive Lymph Nodes (n)"))

# Force specific variables to ordinal (median [IQR]) display
ternD(tern_colon, exclude_vars = c("ID"), methods_doc = FALSE,
      force_ordinal = c("Positive_Lymph_Nodes_n"))

# Export to Word (writes a file to tempdir)

ternD(tern_colon,
      exclude_vars     = c("ID"),
      methods_doc      = FALSE,
      open_doc         = FALSE,
      output_docx      = file.path(tempdir(), "descriptive.docx"),
```

```
  category_start   = c("Patient Demographics"  = "Age (yr)",
                       "Surgical Findings"     = "Colonic Obstruction",
                       "Tumor Characteristics" = "Positive Lymph Nodes (n)",
                       "Outcomes"              = "Recurrence"))
```

`ternG` *Generate grouped summary table with appropriate statistical tests*

Description

Creates a grouped summary table with optional statistical testing for group comparisons. Supports numeric and categorical variables; numeric variables can be treated as ordinal via `force_ordinal`. Includes options to calculate P values and odds ratios. For descriptive (ungrouped) tables, use `ternD`.

Usage

```
ternG(
  data,
  vars = NULL,
  exclude_vars = NULL,
  group_var,
  force_ordinal = NULL,
  force_normal = NULL,
  force_continuous = NULL,
  group_order = NULL,
  output_xlsx = NULL,
  output_docx = NULL,
  OR_col = FALSE,
  OR_method = "dynamic",
  consider_normality = "ROBUST",
  print_normality = FALSE,
  show_test = FALSE,
  p_digits = 3,
  round_intg = FALSE,
  round_decimal = NULL,
  smart_rename = TRUE,
  insert_subheads = TRUE,
  factor_order = "mixed",
  table_font_size = 9,
  methods_doc = TRUE,
  methods_filename = "TernTables_methods.docx",
  category_start = NULL,
  plain_header = NULL,
  manual_italic_indent = NULL,
  manual_underline = NULL,
```

```
  indent_info_column = FALSE,
  show_total = TRUE,
  table_caption = NULL,
  table_footnote = NULL,
  abbreviation_footnote = NULL,
  variable_footnote = NULL,
  index_style = "symbols",
  line_break_header = getOption("TernTables.line_break_header", TRUE),
  post_hoc = FALSE,
  p_adjust = FALSE,
  p_adjust_display = "fdr_only",
  open_doc = TRUE,
  citation = TRUE,
  font_family = getOption("TernTables.font_family", "Arial"),
  show_missing = FALSE,
  show_p = TRUE,
  zero_to_dash = FALSE,
  percentage_compute = "column",
  categorical_posthoc = FALSE,
  show_missingness = FALSE,
  missing_indicators = NULL,
  plain_tibble = FALSE
)
```

Arguments

`data`	Tibble containing all variables.
`vars`	Character vector of variables to summarize. Defaults to all except `group_var` and `exclude_vars`.
`exclude_vars`	Character vector of variable(s) to exclude. `group_var` is automatically excluded.
`group_var`	Character, the grouping variable (factor or character with >=2 levels).
`force_ordinal`	Character vector of variables to treat as ordinal (i.e., use medians/IQR and non-parametric tests).
`force_normal`	Character vector of variable names to treat as normally distributed, bypassing all normality assessment (Gates 1–4 under `"ROBUST"`, or Shapiro-Wilk under `TRUE`). Listed variables are summarized as mean ± SD and compared with Welch tests regardless of the `consider_normality` setting. Takes priority over `consider_normality` but not over `force_ordinal` (if a variable appears in both, `force_ordinal` wins). Default is `NULL`.
`force_continuous`	Character vector of variables to force treatment as continuous (mean ± SD and parametric tests), bypassing the automatic binary `0/1` detection that would otherwise convert them to categorical Y/N. Useful when a numeric variable with only two unique values (e.g. `0/1` dose levels) should be analysed as a continuous measurement rather than a dichotomous category. Takes priority over automatic type detection but does not override `force_ordinal` (if a variable appears in both, `force_ordinal` wins). Default is `NULL`.

`group_order` Optional character vector to specify a custom group level order.

`output_xlsx` Optional filename to export the table as an Excel file.

`output_docx` Optional filename to export the table as a Word document.

`OR_col` Logical; if `TRUE`, adds unadjusted odds ratios with 95% CI for binary categorical variables (Y/N, YES/NO, or numeric 0/1) and two-level categorical variables (e.g. Male/Female). For two-level categoricals displayed with sub-rows, the reference level (factor level 1, or alphabetical first for non-factors) shows `"1.00 (ref.)"`; the non-reference level shows the computed OR with 95% CI. Variables with three or more levels show `"-"`. Only valid when `group_var` has exactly 2 levels; an error is raised for 3+ group comparisons. Default is `FALSE`.

`OR_method` Character; controls how odds ratios are calculated when `OR_col = TRUE`. If `"dynamic"` (default), uses Fisher's exact method when any expected cell count is < 5 (Cochran criterion), otherwise uses the Wald method. If `"wald"`, forces the Wald method regardless of expected cell counts.

`consider_normality` Character or logical; controls how continuous variables are routed to parametric vs. non-parametric tests. `"ROBUST"` (default) applies a four-gate decision consistent with standard biostatistical practice: (1) any group $n < 3$ is a conservative fail-safe to non-parametric; (2) absolute skewness > 2 or excess kurtosis > 7 in any group routes to non-parametric regardless of sample size (catches heavy-tailed distributions and LOS/count-style skews in which the CLT guarantee is compromised); (3) all groups $n \geq 30$ routes to parametric via the Central Limit Theorem; (4) otherwise Shapiro-Wilk $p > 0.05$ in all groups routes to parametric. Normal variables use mean $\pm$ SD and Welch t-test (2 groups) or Welch ANOVA (3+ groups); non-normal variables use median [IQR] and Wilcoxon rank-sum (2 groups) or Kruskal-Wallis (3+ groups). If `TRUE`, uses Shapiro-Wilk alone ($p > 0.05$ in all groups = normal). Conservative at large n. If `FALSE`, all numeric variables are treated as normally distributed regardless of distribution. If `"FORCE"`, all numeric variables are treated as non-normal (median [IQR], nonparametric tests).

`print_normality` Logical; if `TRUE`, includes Shapiro-Wilk P values in the output. Default is `FALSE`.

`show_test` Logical; if `TRUE`, includes the statistical test name as a column in the output. Default is `FALSE`.

`p_digits` Integer; number of decimal places for P values (default 3).

`round_intg` Logical; if `TRUE`, rounds all means, medians, IQRs, and standard deviations to nearest integer (0.5 rounds up). Default is `FALSE`.

`round_decimal` Integer or `NULL`; number of decimal places for all continuous summary values (means, SDs, medians, IQRs). Overrides the default of 1 decimal place when set. Ignored when `round_intg = TRUE`. Default is `NULL` (1 decimal place).

`smart_rename` Logical; if `TRUE`, automatically cleans variable names and subheadings for publication-ready output using built-in rule-based pattern matching for common medical abbreviations and prefixes. Default is `TRUE`.

`insert_subheads` Logical; if `TRUE` (default), creates a hierarchical structure with a header row and indented sub-category rows for categorical variables with 3 or more levels.

Binary variables (Y/N, YES/NO, or numeric 1/0 – which are auto-detected and treated as Y/N) are always displayed as a single row showing the positive/yes count regardless of this setting. Two-level categorical variables whose values are not Y/N, YES/NO, or 1/0 (e.g. Male/Female) use the hierarchical sub-row format, showing both levels as indented rows. If FALSE, all categorical variables use a single-row flat format. Default is TRUE.

`factor_order` Character; controls the ordering of factor levels in the output. "mixed" (default) applies level-aware ordering for two-level categorical variables and frequency ordering for variables with three or more levels: for any factor, factor level order is always respected regardless of the number of levels; for non-factor two-level variables (e.g. Male/Female), levels are sorted alphabetically; for non-factor variables with three or more levels, levels are sorted by decreasing frequency. "levels" respects the original factor level ordering for all variables; if the variable is not a factor, falls back to frequency ordering. "frequency" orders all levels by decreasing frequency (most common first).

`table_font_size`
Numeric; font size for Word document output tables. Default is 9.

`methods_doc` Logical; if TRUE (default), generates a methods document describing the statistical tests used.

`methods_filename`
Character; filename for the methods document. Default is "TernTables_methods.docx".

`category_start` Named character vector specifying where to insert category headers. Names are the header label text to display; values are the anchor variable – either the original column name (e.g. "Age_Years") or the cleaned display name (e.g. "Age (yr)"). Both forms are accepted. Example: c("Demographics" = "Age_Years", "Clinical" = "bmi"). Default is NULL (no category headers).

`plain_header` Named character vector, same interface as category_start. Names are the label text; values are the anchor variable to insert before. Inserts a label-only row with underline formatting and no bold, merge, or border treatments. Default NULL.

`manual_italic_indent`
Character vector of display variable names (post-cleaning) that should be formatted as italicized and indented in Word output – matching the appearance of factor sub-category rows. Has no effect on the returned tibble; only applies when output_docx is specified or when the tibble is passed to word_export.

`manual_underline`
Character vector of display variable names (post-cleaning) that should be formatted as underlined in Word output – matching the appearance of multi-category variable headers. Has no effect on the returned tibble; only applies when output_docx is specified or when the tibble is passed to word_export.

`indent_info_column`
Logical; if FALSE (default), the internal .indent helper column is dropped from the returned tibble. Set to TRUE to retain it – this is necessary when you intend to post-process the tibble and later pass it to word_export directly, as word_export uses the .indent column to apply correct indentation and italic formatting in the Word table.

show_total Logical; if TRUE, adds a "Total" column showing the aggregate summary statistic across all groups (e.g., for a publication Table 1 that includes both per-group and overall columns). Default is TRUE.

table_caption Optional character string for a table caption to display above the table in the Word document. Rendered as size 11 Arial bold, single-spaced with a small gap before the table. Default is NULL (no caption). Example: "Table 2. Comparison of recurrence vs. no recurrence."

table_footnote Optional character string for a footnote to display below the table in the Word document. Rendered as size 6 Arial italic with a double-bar border above and below. Default is NULL (no footnote).

abbreviation_footnote
Optional character string listing abbreviations. Always printed first in the footnote block. Default NULL.

variable_footnote
Optional named character vector. Names are display variable names (case-insensitive); values are the footnote definition text. Each variable gets the next symbol appended to its name in the table, and the footnote block lists each definition below the abbreviation line. To share one footnote between multiple variables, separate their names with a pipe: c("Var A|Var B" = "Shared note text."). Default NULL.

index_style Character; "symbols" (default) uses *, dagger, double-dagger ... "alphabet" uses Unicode superscript letters. See word_export for details.

line_break_header
Logical; if TRUE (default), column headers are wrapped with \n – group names break on spaces, sample size counts move to a second line, and the first column header reads "Category / Variable". Set to FALSE to suppress all header line breaks. Can also be set package-wide via options(TernTables.line_break_header = FALSE).

post_hoc Logical; if TRUE, runs pairwise post-hoc tests for continuous and ordinal variables in three or more group comparisons and annotates each group column value with a compact letter display (CLD) superscript. Groups sharing a letter are not significantly different at $\alpha = 0.05$. For normally distributed variables (Welch ANOVA path), Games-Howell pairwise tests are used. For non-normal and ordinal variables (Kruskal-Wallis path), Dunn's test with Holm correction is used. Post-hoc testing is never applied to categorical variables. Only valid when group_var has three or more levels; silently ignored for two-group comparisons. Requires the rstatix package. Default is FALSE.

p_adjust Logical; if TRUE, applies the Benjamini-Hochberg (BH) false discovery rate correction to all omnibus P values after testing. The correction pool is one P value per variable — sub-rows of multi-level categoricals share the parent P value and are not double-counted. Post-hoc pairwise P values (which already carry within-variable Holm correction) are excluded from the correction pool. Default is FALSE.

p_adjust_display
Character; controls how BH-corrected P values appear in the output when p_adjust = TRUE. "fdr_only" (default) replaces the standard P value column with the corrected values, renaming the column to "P value (FDR corrected)". "both"

retains the original P values in a column named "P value" and adds FDR-corrected values immediately to its right in a column named "P value (FDR corrected)". Ignored when p_adjust = FALSE.

open_doc
: Logical; if TRUE (default), automatically opens the written Word document in the system default application after saving. Set to FALSE to suppress. Has no effect when output_docx is NULL.

citation
: Logical; if TRUE (default), appends a citation line at the bottom of the table footnote block and at the end of the methods document: package version, authors, and links to the GitHub repository and web interface. Set to FALSE to suppress.

font_family
: Character; font family name used for all Word output (table, captions, footnotes, methods document). Any font installed on the system that renders the document may be used. Popular options include "Arial", "Helvetica", "Times New Roman", "Garamond", and "Calibri". Defaults to getOption("TernTables.font_family", "Arial").

show_missing
: Logical; if TRUE, appends a "Missing" row after each variable's data rows showing the count and percentage of missing observations per group (denominator is each group's total N). Missing rows display "-" in the P and OR columns. A footnote is automatically appended noting that missing values are reported. Default is FALSE.

show_p
: Logical; if TRUE (default), the P value column is included in the output and Excel/Word exports. Set to FALSE to produce a descriptive-only grouped table — the output will contain only the Variable column, one column per group level, and the Total column (if show_total = TRUE). When FALSE, OR_col, show_test, print_normality, post_hoc, categorical_posthoc, and p_adjust are all suppressed automatically.

zero_to_dash
: Logical; if TRUE, replaces any categorical cell that would display "0 (0%)" with "-" in the output table. Useful when zero counts in a group are not meaningful to report numerically (e.g. no patients with a condition in one arm). Default is FALSE.

percentage_compute
: Character; controls the denominator used when computing percentages for categorical variables. "column" (default) divides each cell count by the column (group) total, so percentages describe the composition of each group – the standard Table 1 interpretation (e.g. "60% of the Recurrence group is Male"). "row" divides each cell count by the row total (the number of subjects with that category level across all groups), so percentages describe how each category level is distributed across groups (e.g. "30% of Males had Recurrence"). When "row", the Total column is automatically suppressed (a Total column would show 100% for every level, which is uninformative). Applies to both binary and multinomial categorical variables in both two- and three-group comparisons.

categorical_posthoc
: Logical; if TRUE, computes adjusted standardized residuals from the global contingency table for categorical variables following a significant omnibus test ($p < 0.05$). Cells whose adjusted standardized residual exceeds ± 1.96 are marked with an asterisk (*), indicating a significant deviation from expected frequencies ($\alpha = 0.05$). This method is equivalent to Haberman's adjusted residuals and does not require a separate multiple-comparisons correction, as the ± 1.96

threshold is derived directly from the omnibus test distribution. Only applied when group_var has three or more levels; silently ignored for two-group comparisons. Default is FALSE.

show_missingness
: Controls whether a column of missing-value percentages is appended to the table. Options:
FALSE (default) — no missingness columns added.
"total" — one column ("Missing, n (%)") appended at the far right showing the number and percentage of missing observations across *all* rows for each variable.
"group" — one column per group level ("Miss. [level]") interleaved immediately after each group's data column, showing per-group missingness for each variable.
Missingness is computed on the *raw* data column (before ternG's internal NA filtering), so both NA values and string representations of missing data (e.g., "Unknown", "N/A") are counted. See missing_indicators to customise which strings count.

missing_indicators
: Optional character vector of string values to treat as missing in addition to (or instead of) the built-in ternP defaults. When NULL (default), the ternP canonical list is used ("na", "n/a", "unknown", etc.). When supplied, the custom list **replaces** the ternP defaults — only the values in missing_indicators (plus true NA) are counted as missing. Matching is always case-insensitive and ignores leading/trailing whitespace.

plain_tibble
: Logical; if TRUE, returns the tidy per-variable statistics frame (see tern_stats) instead of the formatted display table — one un-indented row per variable with raw numeric statistics and their formatted counterparts in adjacent columns. Word/Excel exports and the methods document are still written if requested. Setting this is never required to reach the tidy data: the same frame is always attached to the normal return value as the "tern_stats" attribute, so tern_stats(ternG(...)) yields both the pretty table and the tidy data from one call. Default is FALSE.

Details

Independence assumption: all statistical tests applied by this function (Welch's *t*-test, Wilcoxon rank-sum, Welch ANOVA, Kruskal-Wallis, chi-squared, and Fisher's exact) assume that observations are independent — each row must represent a distinct, unrelated subject. ternG is not appropriate for repeated-measures, longitudinal, or clustered data (e.g. pre/post measurements, matched pairs, or patients nested within sites).

Value

A tibble with one row per variable (multi-row for multi-level factors), showing summary statistics by group, P values, test type, and optionally odds ratios and total summary column. Two tidy, machine-readable frames are always attached as attributes: "tern_stats" (one row per variable — test, statistic, df, raw numeric P value, normality decision, odds ratio) and "tern_estimates" (long format: one row per variable x group x level, with the underlying counts, percentages, means, SDs,

medians and quartiles). Access them with tern_stats and tern_estimates rather than parsing the display table.

See Also

tern_stats, tern_estimates

Examples

```
data(tern_colon)

# 2-group comparison
ternG(tern_colon, exclude_vars = c("ID"), group_var = "Recurrence",
      methods_doc = FALSE)

# 2-group comparison with odds ratios
ternG(tern_colon, exclude_vars = c("ID"), group_var = "Recurrence",
      OR_col = TRUE, methods_doc = FALSE)

# 3-group comparison
ternG(tern_colon, exclude_vars = c("ID"), group_var = "Treatment_Arm",
      group_order = c("Observation", "Levamisole", "Levamisole + 5FU"),
      methods_doc = FALSE)

# 2-group comparison with BH FDR correction (fdr_only — default display)
ternG(tern_colon, exclude_vars = c("ID"), group_var = "Recurrence",
      p_adjust = TRUE, methods_doc = FALSE)

# 2-group comparison with BH FDR correction (show raw + corrected side by side)
ternG(tern_colon, exclude_vars = c("ID"), group_var = "Recurrence",
      p_adjust = TRUE, p_adjust_display = "both", methods_doc = FALSE)

# Export to Word (writes a file to tempdir)

ternG(tern_colon,
      exclude_vars   = c("ID"),
      group_var      = "Recurrence",
      OR_col         = TRUE,
      methods_doc    = FALSE,
      open_doc       = FALSE,
      output_docx    = file.path(tempdir(), "comparison.docx"),
      category_start = c("Patient Demographics"  = "Age (yr)",
                         "Tumor Characteristics" = "Positive Lymph Nodes (n)"))
```

---

ternP *Preprocess a raw data frame for use with ternG or ternD*

Description

`ternP()` cleans a raw data frame loaded from a CSV or XLSX file, applying a standardized set of transformations and performing validation checks before the data is passed to `ternG` or `ternD`.

Usage

```
ternP(data, mode = "auto", extra_na = NULL, drop_cols = NULL)
```

Arguments

`data`	A data frame or tibble as loaded from a CSV or XLSX file (e.g. via `readr::read_csv()` or `readxl::read_excel()`). All character columns are processed; numeric and logical columns are passed through unchanged by the string-cleaning steps.
`mode`	Preprocessing mode. One of `"auto"` (default) or `"manual"`.
	`"auto"` Default behaviour. PHI column-name detection and unnamed-column checks run as hard stops before any cleaning. All other transformations (string-NA conversion, whitespace trimming, empty-column removal, blank-row removal, case normalisation) run automatically.
	`"manual"` PHI detection is **skipped**. You take full responsibility for ensuring no patient identifiers are present. A prominent warning is emitted. All other cleaning transformations still run. Use `drop_cols` to explicitly remove any columns you do not want in the cleaned data.
`extra_na`	Optional character vector of additional string values to treat as missing (converted to `NA`). These are **appended** to the built-in list (na, n/a, missing, unknown, etc.) — not a replacement. Matching is case-insensitive and whitespace-trimmed. Works in both `"auto"` and `"manual"` modes. Example: `extra_na = c("9999", "Not Done", "PENDING")`.
`drop_cols`	Optional character vector of column names to drop from the data before cleaning begins. Intended for use in `"manual"` mode to explicitly remove identifier or unwanted columns without triggering the PHI check. Any names not found in `data` are silently ignored. Works in both modes.

Value

A named list with three elements:

`clean_data` A tibble containing the fully cleaned dataset, ready to pass to `ternG()` or `ternD()`.

`sparse_rows` A tibble of rows from `clean_data` where more than 50% of values are `NA`. These rows are *retained* in `clean_data` but extracted here for optional review or download. An empty tibble if no sparse rows exist.

`feedback` A named list of feedback items. Each element is `NULL` if the corresponding transformation was not triggered, or a value describing what changed:

- `string_na_converted` A named list with elements `total` (integer count of values converted) and `cols` (character vector of affected column names), or `NULL` if no string NA values were found.
- `blank_rows_removed` A named list with elements `count` (integer) and `row_indices` (integer vector of original row positions removed), or `NULL` if none.

`sparse_rows_flagged` A named list with elements `count` (integer) and `row_indices` (integer vector of row positions in `clean_data` with >50% missingness), or `NULL` if none.

`case_normalized_vars` A named list with elements `cols` (character vector of affected column names) and `detail` (a named list per column, each with `changed_from` and `changed_to` character vectors showing the exact value changes), or `NULL` if none. #'

`dropped_user_cols` Character vector of column names explicitly dropped via the `drop_cols` parameter, or `NULL` if `drop_cols` was not used.

`manual_mode` Logical. `TRUE` when `mode = "manual"` was used (PHI check skipped), `FALSE` otherwise.

`dropped_empty_cols` Character vector of column names (or `""` for unnamed columns) that were dropped because they were 100% empty, or `NULL` if none.

`date_cols_detected` Character vector of column names that appear to contain date values — either R `Date`/`POSIXct` types (from Excel) or character columns where $\geq$80% of non-NA values match a common date pattern (from CSV). These columns are *not* dropped automatically; the caller should decide whether to exclude them or keep them as categorical variables.

Cleaning pipeline (in order)

1. Date columns are detected (R `Date`/`POSIXct` types, or character columns where $\geq$80% of values match a common date pattern) and reported in `feedback$date_cols_detected`. They are not dropped automatically — the caller decides whether to exclude or keep them.
2. String NA values (`"NA"`, `"na"`, `"N/A"`, `"NaN"`, `"missing"`, `"unknown"`, `"unk"`, `"not available"`, `"not applicable"`, `"none"`, `"null"`, `"nil"`, `"-"`, `"."`, `"?"`) are converted to `NA` (matching is case-insensitive).
3. Leading and trailing whitespace is trimmed from all character columns.
4. Columns that are 100% empty (all `NA`) are silently dropped.
5. Rows where every cell is `NA` are removed.
6. Character columns where values differ only by capitalization (e.g. `"Male"` vs `"MAle"`) are standardized to title case.

Validation hard stops

`ternP()` stops with a descriptive error if:

- Any column name matches a protected health information (PHI) pattern (e.g. `MRN`, `DOB`, `FirstName`). De-identified research identifiers such as `patient_id`, `subject_id`, and `participant_id` are explicitly excluded, as are clinical-event dates (admission date, discharge date, visit date, etc.). Only personal-identity dates such as DOB and DOD are flagged.
- Any column with a blank or whitespace-only header contains data. Completely empty unnamed columns are silently dropped and do not trigger this error.

See Also

`ternG` for grouped comparisons, `ternD` for descriptive statistics.

Examples

```
# Load a messy CSV and preprocess it
path   <- system.file("extdata/csv", "tern_colon_messy.csv",
                      package = "TernTables")
raw    <- read.csv(path, stringsAsFactors = FALSE)
result <- ternP(raw)

# Access cleaned data
result$clean_data

# Review preprocessing feedback
result$feedback

# Sparse rows flagged (>50% missing), retained but not removed
result$sparse_rows
```

`ternStyle` *Export a custom tibble to Word with TernTables formatting*

Description

`ternStyle()` renders any user-built tibble into a Word document with the exact same visual style as tables produced by `ternG()`, `ternD()`, and `word_export()` – Arial font, grey header, double-bar footer, caption/footnote block, and citation footer.

Usage

```
ternStyle(
  tbl,
  filename = NULL,
  col1_name = NULL,
  subheader_rows = NULL,
  bold_rows = NULL,
  bold_sig = NULL,
  italic_rows = NULL,
  bold_cols = NULL,
  italic_cols = NULL,
  header_format_follow = FALSE,
  round_intg = FALSE,
  round_decimal = NULL,
  font_size = 9,
  category_start = NULL,
  plain_header = NULL,
  manual_italic_indent = NULL,
  manual_underline = NULL,
  table_caption = NULL,
```

```
  table_footnote = NULL,
  abbreviation_footnote = NULL,
  variable_footnote = NULL,
  index_style = "symbols",
  col1_header = NULL,
  line_break_header = FALSE,
  open_doc = TRUE,
  citation = TRUE,
  font_family = getOption("TernTables.font_family", "Arial"),
  spanner = NULL,
  col1_spanner_pos = c("top", "bottom")
)
```

Arguments

`tbl`	A data frame or tibble. The first column is used as the row-label column (rendered as "Variable" unless renamed via `col1_name`). All columns are coerced to character before rendering; `NA` values become empty strings.
`filename`	Output file path ending in `.docx`. Pass `NULL` (default) to write to a temporary file and suppress auto-opening – useful when the result will be passed directly to `ternB` for bundling.
`col1_name`	Optional character string. If supplied, the first column is renamed to this label in the rendered table. The column need not be named `"Variable"` in the input; any name is accepted and renamed here. Default `NULL` (use the tibble's existing first column name).
`subheader_rows`	Character vector of labels that already exist as rows in `tbl` and should be formatted as full section-header rows: cells merged across all columns, bold, with a bottom border line – identical to the treatment applied by `category_start`. No row is inserted; the matching existing row is formatted in place. Matching is case-insensitive. Default `NULL`.
`bold_rows`	Integer vector of body row indices (1-based, final rendered table) to bold across every column. Applied after all structural formatting so it always wins. Default `NULL`.
`bold_sig`	Optional named list for cell-level p-value-based bolding. Use this when your tibble has pre-formatted p-value strings in columns that are not named `"P value"` (e.g. `"Uni p"`, `"Multi p"`). Supply a list with: • `p_cols` — character vector of column names containing p-value strings. • `hr_cols` — optional character vector of column names (same length and order as `p_cols`) to also bold when the paired p-value is significant (e.g. the corresponding HR or coefficient column). Pass `NULL` or omit to skip paired-column bolding. • `threshold` — numeric significance threshold. Default `0.05`. The Variable column is never modified by `bold_sig`; use `bold_rows` to bold entire rows (e.g., predictor-level rows where the p-value represents an omnibus LRT). Example:

```
bold_sig = list(
  p_cols    = c("Uni p", "Multi p"),
  hr_cols   = c("Uni HR (95% CI)", "Multi HR (95% CI)"),
  threshold = 0.05
)
```

Default NULL.

italic_rows Integer vector of body row indices to italicize across every column. Default NULL.

bold_cols Integer vector of column indices (1-based) to bold across all body rows. Default NULL.

italic_cols Integer vector of column indices to italicize across all body rows. Default NULL.

header_format_follow
Logical; if TRUE, columns listed in bold_cols or italic_cols also have their header cell bolded or italicized. Default FALSE.

round_intg Logical; passed to word_export. Default FALSE.

round_decimal Integer or NULL; if provided, rounds all numeric values in the table to this many decimal places before rendering. Passed to word_export. Default NULL (no rounding).

font_size Numeric; font size for table body. Default 9.

category_start Named character vector; same as in word_export. Insert new section-header rows at anchor variable positions, in addition to any rows already in the tibble. Default NULL.

plain_header Named character vector; same as in word_export. Insert underline-only (no bold, no merge) label rows at anchor positions. Default NULL.

manual_italic_indent
Character vector of row labels to italicize and indent (sub-item appearance). Default NULL.

manual_underline
Character vector of row labels to underline (multi- category header appearance without the full subheader treatment). Default NULL.

table_caption Optional character string for the caption above the table. Default NULL.

table_footnote Optional character string for a footnote below the table. Default NULL.

abbreviation_footnote
Optional character string (or character vector) of abbreviations. Always printed first in the footnote block. Default NULL.

variable_footnote
Optional named character vector of per-variable footnote definitions (case-insensitive name match). To share one footnote symbol between multiple variables, separate their names with a pipe: c("Var A|Var B" = "Shared note text."). Default NULL.

index_style Character; "symbols" (default) or "alphabet". Controls the footnote symbol sequence. See word_export for details.

col1_header Optional character string. Overrides the top-left header cell. When NULL (default), the standard "Category\n Variable" label is used. Example: "Variable\n Index Management Strategy".

line_break_header
: Logical; if TRUE, column headers are wrapped with \n and the first column header shows the two-line "Category / Variable" label. For custom tibbles the column names are typically already formatted, so this defaults to FALSE here (unlike word_export where it defaults to TRUE).

open_doc
: Logical; if TRUE (default), opens the written document after saving. Default TRUE.

citation
: Logical; if TRUE (default), appends the TernTables citation line in the page footer. Default TRUE.

font_family
: Character; font family name used for all Word output. Defaults to getOption("TernTables.font_famil "Arial"). See word_export for details.

spanner
: Optional named list of column spanners (decked header). Each element name is the spanner label; the value is a character vector of column names from tbl that fall under that spanner. Columns not listed receive an empty spanner cell. The spanner row sits above the column-header row and shares the same grey background and bold styling. Supply the column names exactly as they appear in tbl.

  To strip redundant group prefixes from column headers (e.g. when the spanner already provides the group label), use a *named* inner vector where the **names** are the display labels and the **values** are the actual column names in tbl:

```
spanner = list(
    "SLAM"    = c("r" = "SLAM r",    "p" = "SLAM p"),
    "Control" = c("r" = "Control r", "p" = "Control p"),
    "NMN"     = c("r" = "NMN r",     "p" = "NMN p")
  )
```

  Unnamed inner vectors leave column headers unchanged:

```
spanner = list(
    "Univariate"   = c("Uni HR (95% CI)", "Uni p"),
    "Multivariate" = c("Multi HR (95% CI)", "Multi p")
  )
```

  Default NULL (no spanner row).

col1_spanner_pos
: Character; "top" (default) places the first-column header label in the spanner row for visual prominence; "bottom" keeps the classic layout with the label in the column-names row. Has no effect when spanner is NULL.

Details

Use this function when you have pre-computed summary statistics in a tibble (e.g. a custom cross-tab or manually assembled output table) and want it to match the rest of your TernTables document without running it through the full ternG/ternD pipeline.

Value

Invisibly returns the input tibble (after renaming and coercion) with a "ternB_meta" attribute attached. This makes the result directly passable to ternB for bundling with other tables into a com-

bined Word document. Any spanner definition is stored in the metadata and replayed automatically by ternB().

Examples

```
library(tibble)
# Basic usage with section headers
my_tbl <- tibble(
  Variable      = c("Section A", "Row 1", "Row 2", "Section B", "Row 3"),
  `Group 1`     = c("",          "12 (40%)", "18 (60%)", "", "9 (30%)"),
  `Group 2`     = c("",          "15 (50%)", "15 (50%)", "", "21 (70%)")
)
ternStyle(
  tbl             = my_tbl,
  filename        = file.path(tempdir(), "custom_table.docx"),
  subheader_rows  = c("Section A", "Section B"),
  open_doc        = FALSE,
  citation        = FALSE
)

# Spanner (decked) header grouping related columns
reg_tbl <- tibble(
  Variable            = c("Age (yr)", "BMI"),
  `Uni HR (95% CI)`   = c("1.02 [0.98-1.06]", "1.11 [1.03-1.19]"),
  `Uni p`             = c("0.31", "0.006"),
  `Multi HR (95% CI)` = c("1.01 [0.97-1.05]", "1.08 [1.00-1.17]"),
  `Multi p`           = c("0.64", "0.047")
)
ternStyle(
  tbl         = reg_tbl,
  filename    = file.path(tempdir(), "regression_table.docx"),
  col1_header = "Variable",
  spanner     = list(
    "Univariate"   = c("Uni HR (95% CI)", "Uni p"),
    "Multivariate" = c("Multi HR (95% CI)", "Multi p")
  ),
  open_doc    = FALSE,
  citation    = FALSE
)
```

---

val_format *Format a mean +/- SD string*

Description

Format a mean +/- SD string

Usage

```
val_format(mean, sd)
```

Arguments

mean	Numeric mean value. Formatted to 1 decimal place.
sd	Numeric standard deviation. Formatted to 1 decimal place.

Value

A character string of the form "X.X \u00b1 Y.Y" where both values are rendered to 1 decimal place using fixed-point notation.

---

val_p_format *Format a P value for reporting*

Description

Format a P value for reporting

Usage

```
val_p_format(p, digits = 3)
```

Arguments

p	Numeric P value in the range [0, 1]. NA values are returned as NA_character_. Values >= 1 (or rounding to >= 1) are returned as e.g. ">0.999".
digits	Integer; number of decimal places for reported P values. Default is 3. Note: for p < 0.001, the value is reported in scientific notation with 1 significant figure regardless of digits (e.g., 8E-4).

Value

A character string. Values < 0.001 are formatted in scientific notation with 1 significant figure (e.g., "8E-4"). All other values use fixed-point notation rounded to digits decimal places.

---

word_export *Export TernTables output to a formatted Word document*

Description

Export TernTables output to a formatted Word document

Usage

```
word_export(
  tbl,
  filename,
  round_intg = FALSE,
  round_decimal = NULL,
  font_size = 9,
  category_start = NULL,
  plain_header = NULL,
  subheader_rows = NULL,
  bold_rows = NULL,
  bold_sig = NULL,
  italic_rows = NULL,
  bold_cols = NULL,
  italic_cols = NULL,
  header_format_follow = FALSE,
  manual_italic_indent = NULL,
  manual_underline = NULL,
  table_caption = NULL,
  table_footnote = NULL,
  abbreviation_footnote = NULL,
  posthoc_footnote = NULL,
  variable_footnote = NULL,
  index_style = "symbols",
  page_break_after = FALSE,
  col1_header = NULL,
  line_break_header = getOption("TernTables.line_break_header", TRUE),
  open_doc = TRUE,
  citation = TRUE,
  font_family = getOption("TernTables.font_family", "Arial"),
  spanner = NULL,
  col1_spanner_pos = c("top", "bottom")
)
```

Arguments

`tbl`	A tibble created by ternG or ternD
`filename`	Output file path ending in .docx
`round_intg`	Logical; if TRUE, adds note about integer rounding. Default is FALSE.
`round_decimal`	Integer or NULL; if provided, rounds all numeric values in the table to this many decimal places before rendering. Default is `NULL` (no rounding).
`font_size`	Numeric; font size for table body. Default is 9.
`category_start`	Named character vector specifying category headers. Names are header label text; values are anchor variable names – either the original column name or the cleaned display name (both forms accepted).
`plain_header`	Named character vector, same interface as `category_start`. Names are the label text to display; values are the anchor variable to insert before. The inserted

row has text only in column 1 (all other cells blank) and receives *underline* formatting – identical to `manual_underline` – but no bold, merge, or border treatments. Default is `NULL` (none).

`subheader_rows` Character vector of labels that already exist as rows in the table and should be formatted as full category section headers (merged across all columns, bold, with a bottom border line). Unlike `category_start`, no new row is inserted – the matching existing row is formatted in place. Intended for use with `ternStyle()` where section- divider rows are pre-built into the tibble. Case-insensitive match. Default `NULL`.

`bold_rows` Integer vector of body row indices (1-based, in the final rendered table) to bold across every column. Applied as the last formatting pass so it overrides structural formatting. Default `NULL`.

`bold_sig` Optional named list for cell-level conditional bolding based on parsed p-values. Intended for use with `ternStyle()` when columns contain pre-formatted p-values that are not named `"P value"` (the name `ternG` uses internally). Supply a list with:

- `p_cols` — character vector of column names containing p-value strings.
- `hr_cols` — optional character vector of column names to also bold when the paired p-value is significant (must be the same length and order as `p_cols`, or `NULL` to skip paired HR bolding).
- `threshold` — numeric significance threshold; default `0.05`.

For each p-value cell where the parsed numeric value is below `threshold`, that cell is bolded. If `hr_cols` is supplied, the corresponding HR cell in the same row is also bolded. The Variable column is never touched by this argument — use `bold_rows` to bold entire rows (e.g., predictor header rows). Default `NULL`.

`italic_rows` Integer vector of body row indices to italicize across every column. Default `NULL`.

`bold_cols` Integer vector of column indices (1-based) to bold across all body rows. Default `NULL`.

`italic_cols` Integer vector of column indices to italicize across all body rows. Default `NULL`.

`header_format_follow`
Logical; if `TRUE`, any columns listed in `bold_cols` or `italic_cols` also have their header cell bolded or italicized, respectively. Default `FALSE`.

`manual_italic_indent`
Character vector of display variable names (post-cleaning) to force into italicized and indented formatting, matching the appearance of factor sub-category rows (e.g., levels of a multi-category variable). Use this for rows that should visually appear as sub-items but are not automatically detected as such.

`manual_underline`
Character vector of display variable names (post-cleaning) to force into underlined formatting, matching the appearance of multi-category variable header rows. Use this for rows that should visually appear as section headers but are not automatically detected as such.

`table_caption` Optional character string to display as a caption above the table in the Word document. Rendered as size 11 Arial bold, single-spaced with a small gap before the table. Default is `NULL` (no caption).

table_footnote Optional character string to display as a footnote below the table in the Word document. Rendered as size 6 Arial italic. A double-bar border is applied above and below the footnote row. Default is NULL (no footnote).

abbreviation_footnote
Optional character string (or character vector, which will be collapsed with spaces) listing abbreviations to display at the top of the footnote block. Always printed first, before any variable-specific footnote lines. Default NULL.

posthoc_footnote
Optional character string describing post-hoc CLD superscript conventions. When supplied by ternG(), it is inserted after the abbreviations footnote and before the variable symbol footnotes. Default NULL.

variable_footnote
Optional named character vector. Names are display variable names as they appear in the table (case-insensitive match); values are the footnote definition text for that variable. Each entry is assigned the next symbol in the sequence (*, dagger, double-dagger, ...) and the symbol is appended to the variable name in column 1. The footnote block lists each as "* Definition text." below the abbreviations. To map multiple variables to the same footnote symbol and note, separate the variable names with a pipe character in the key: c("Var A|Var B" = "Shared note text."). Default NULL.

index_style Character; controls the footnote symbol sequence. "symbols" (default) uses *, dagger, double-dagger, section, pilcrow, double-vertical-bar, then doubled forms. "*" is appended as plain text; all others are rendered as true Word superscripts. "alphabet" uses Unicode superscript letters (a, b, c, ...) which render as raised glyphs without explicit superscript formatting.

page_break_after
Logical; if TRUE, a page break is appended at the end of the Word document after the table. Used internally by ternB() to embed page breaks inside each table's temp file rather than injecting them into the combined document body, which avoids double-break artifacts when tables do not fill the page. Default is FALSE.

col1_header Optional character string. Overrides the top-left header cell text. When NULL (default), the cell shows "Category\n Variable" (the standard two-line label). Supply any string, including "\n" line breaks, to customise. Example: "Variable\n Index Management Strategy".

line_break_header
Logical; if TRUE (default), column headers are wrapped with \n – group names break on spaces, sample size counts move to a second line, and the first column header includes a category hierarchy label. Set to FALSE to suppress all header line breaks. Can also be set package-wide via options(TernTables.line_break_header = FALSE).

open_doc Logical; if TRUE (default), automatically opens the written Word document in the system default application after saving. Set to FALSE to suppress.

citation Logical; if TRUE (default), appends a citation line at the bottom of the table footnote block: package version, authors, and links to the GitHub repository and web interface. Set to FALSE to suppress.

font_family
: Character; font family used for the entire Word table and its caption, footnote, and citation. Any font name accepted by the rendering system is valid (Word will fall back to its default if the font is not installed). Can also be set package-wide via `options(TernTables.font_family = "Garamond")`. Default is `"Arial"`.

spanner
: Optional named list of column spanners (decked header). Each element name is the spanner label text; the value is a character vector of column names from `tbl` that fall under that spanner. Columns not listed receive an empty spanner cell. The spanner row is rendered above the normal column-header row, inheriting the same grey background, bold font, and border treatment. Supply either the original column names or their post-rename equivalents – both are resolved.

  To override the display label shown in the column-header row (e.g. to strip a redundant group prefix when the spanner already provides context), use a *named* inner vector where the **names** are the display labels and the **values** are the actual column names in `tbl`:

```
spanner = list(
    "SLAM"    = c("r" = "SLAM r",    "p" = "SLAM p"),
    "Control" = c("r" = "Control r", "p" = "Control p"),
    "NMN"     = c("r" = "NMN r",     "p" = "NMN p")
  )
```

  Unnamed inner vectors (the simple form) leave column headers unchanged:

```
spanner = list(
    "Univariate"    = c("Uni HR (95% CI)", "Uni p"),
    "Multivariate"  = c("Multi HR (95% CI)", "Multi p")
  )
```

  Default `NULL` (no spanner row).

col1_spanner_pos
: Character; controls where the first-column header label appears when `spanner` is active. `"top"` (default) places the label in the spanner row (top-left cell) and leaves the column-names row cell below it empty, giving the label more visual prominence. `"bottom"` keeps the classic layout where the label stays in the column-names row and the spanner row cell above it is blank. Has no effect when `spanner` is `NULL`.

Value

Invisibly returns the path to the written Word file.

Examples

```
data(tern_colon)
tbl <- ternD(tern_colon, exclude_vars = c("ID"), methods_doc = FALSE, open_doc = FALSE)
word_export(
  tbl      = tbl,
  filename = file.path(tempdir(), "descriptive.docx"),
  open_doc = FALSE,
  category_start = c(
    "Patient Demographics"  = "Age (yr)",
```

```
    "Tumor Characteristics" = "Positive Lymph Nodes (n)"
  )
)
```

`write_cleaning_doc` *Write a cleaning summary document for ternP output*

Description

Generates a Word document summarising the preprocessing transformations applied by `ternP`. Only sections for triggered transformations are written; if the data required no preprocessing, a single sentence stating that is produced instead. The document can be attached to a data-management log or supplemental materials.

Usage

```
write_cleaning_doc(
  result,
  filename = "cleaning_summary.docx",
  font_family = getOption("TernTables.font_family", "Arial"),
  open_doc = TRUE,
  citation = TRUE
)
```

Arguments

`result`	A `ternP_result` object returned by `ternP`.
`filename`	Output file path ending in `.docx`. Default is `"cleaning_summary.docx"` in the current working directory.
`font_family`	Character; font family for the Word document. Default `"Arial"`. Can also be set via `options(TernTables.font_family = ...)`.
`open_doc`	Logical; if `TRUE` (default), automatically opens the written Word document in the system default application after saving. Set to `FALSE` to suppress.
`citation`	Logical; if `TRUE` (default), appends the TernTables citation as a page footer in the written document. Set to `FALSE` to suppress.

Value

Invisibly returns the path to the written Word file.

See Also

`ternP`, `write_methods_doc`

Examples

```
path   <- system.file("extdata/csv", "tern_colon_messy.csv",
                      package = "TernTables")
raw    <- read.csv(path, stringsAsFactors = FALSE)
result <- ternP(raw)
write_cleaning_doc(result, filename = file.path(tempdir(), "cleaning_summary.docx"),
                   open_doc = FALSE)
```

`write_methods_doc` *Write a methods section Word document for TernTables output*

Description

Generates a Word document containing a methods paragraph describing the statistical approach used in a specific `ternG` or `ternD` run. The paragraph is fully dynamic: it reflects the tests that were actually used, the number of comparison groups, whether odds ratios were calculated, and whether post-hoc testing was performed. It is headed by a bold **Statistical Methods** label and followed by a brief attribution footer.

Usage

```
write_methods_doc(
  tbl,
  filename,
  n_levels = 2,
  OR_col = FALSE,
  OR_method = "dynamic",
  source = "ternG",
  post_hoc = FALSE,
  categorical_posthoc = FALSE,
  cat_posthoc_fisher_vars = character(0),
  show_missingness = FALSE,
  missing_indicators = NULL,
  boilerplate = FALSE,
  p_adjust = FALSE,
  open_doc = TRUE,
  citation = TRUE,
  font_family = getOption("TernTables.font_family", "Arial")
)
```

Arguments

`tbl`	A tibble created by `ternG` or `ternD`, or `NULL` when generating a generic document.
`filename`	Output file path ending in `.docx`.

`n_levels` Number of group levels used in `ternG` (2 for two-group, 3+ for multi-group). Ignored when called from `ternD`.

`OR_col` Logical; whether odds ratios were calculated. Default `FALSE`.

`OR_method` Character; the OR calculation method used in `ternG`. `"dynamic"` (default) means Wald when all expected cells >= 5, Fisher's exact otherwise. `"wald"` means Wald was forced regardless of cell counts. Controls the OR description in the generated methods paragraph.

`source` Character; `"ternG"` or `"ternD"`. Controls which section is populated with dynamic test information. Default `"ternG"`.

`post_hoc` Logical; whether pairwise post-hoc testing was requested (`post_hoc = TRUE` in `ternG`). When `TRUE` and `n_levels >= 3`, the three-group methods paragraph is updated to describe the post-hoc test pairing (Games-Howell or Dunn's + Holm). Default `FALSE`.

`categorical_posthoc` Logical; whether adjusted standardized residuals were requested (`categorical_posthoc = TRUE` in `ternG`). When `TRUE` and `n_levels >= 3`, the methods paragraph notes that cells with adjusted standardized residuals exceeding ± 1.96 are marked with an asterisk following a significant omnibus test. Default `FALSE`.

`cat_posthoc_fisher_vars` Character vector of variable names for which Fisher's exact test was the omnibus test while `categorical_posthoc = TRUE`. When non-empty, a caveat sentence is appended noting that Haberman's adjusted residuals were derived from the chi-squared contingency table in the absence of a Fisher's exact equivalent. Populated automatically when called from `ternG()`. Default `character(0)`.

`show_missingness` Logical or character; whether missingness columns were added to the table (`FALSE`, `"total"`, or `"group"`). When non-`FALSE`, a sentence is appended describing the missingness reporting approach and the string representations used to flag missing values. Should match the `show_missingness` argument passed to `ternG()` or `ternD()`. Default `FALSE`.

`missing_indicators` Character vector of string values treated as missing in addition to R `NA`, or `NULL` to use TernTables defaults. Should match the `missing_indicators` argument passed to `ternG()` or `ternD()`. Default `NULL`.

`boilerplate` Logical; if `TRUE`, ignores all other arguments and writes a single comprehensive Word document covering every possible TernTables configuration (descriptive, two-group with and without odds ratios, three-or-more-group with and without post-hoc testing), using package-default phrasing throughout. Output is written to `filename` if supplied; otherwise falls back to `comprehensive_boilerplate_methods.docx` in the current working directory. Intended as a reference document, not for inclusion in a manuscript. Default `FALSE`.

`p_adjust` Logical; if `TRUE`, prepends a sentence to the methods paragraph stating that P values were corrected using the Benjamini-Hochberg FDR procedure, and updates the significance threshold wording accordingly. Should match the `p_adjust` argument passed to `ternG()`. Default `FALSE`.

`open_doc` Logical; if `TRUE` (default), automatically opens the written Word document in the system default application after saving. Set to `FALSE` to suppress.

citation	Logical; if TRUE (default), appends a citation line after the document footer: package version, authors, and links to the GitHub repository and web interface. Set to FALSE to suppress.
font_family	Character; font family for the Word document. Default "Arial". Can also be set via options(TernTables.font_family = ...).

Details

When boilerplate = TRUE, all run-specific arguments are ignored and a comprehensive reference document is written instead, covering all five standard TernTables configurations with package-default phrasing. See the boilerplate parameter for details.

Value

Invisibly returns the methods paragraph text as a character string (or, when boilerplate = TRUE, invisibly returns the output file path). Useful for programmatic inspection or testing without opening the Word file.

Examples

```
data(tern_colon)
tbl <- ternG(tern_colon, exclude_vars = c("ID"), group_var = "Recurrence",
            methods_doc = FALSE, open_doc = FALSE)
write_methods_doc(tbl, filename = file.path(tempdir(), "methods.docx"),
                  open_doc = FALSE)


# Write a comprehensive reference document covering all configurations.
write_methods_doc(tbl = NULL,
                  filename = file.path(tempdir(), "boilerplate_methods.docx"),
                  boilerplate = TRUE, open_doc = FALSE)
```

Index

∗ **datasets**
tern_colon, 5

classify_normality, 2
colon, *5*

print.ternP_result, 4

tern_colon, 5
tern_estimates, 6, *8*, *15*, *23*
tern_stats, *6*, 7, *14, 15*, *22, 23*
ternB, 8, *27*, *29*
ternD, *6–8*, 10, *24, 25*
ternG, *6–8*, 16, *24, 25*
ternP, *4*, 23, *36*
ternStyle, 26

val_format, 30
val_p_format, 31

word_export, *29*, 31
write_cleaning_doc, 36
write_methods_doc, *36*, 37