

# Package ‘gtsummary’

June 13, 2020

**Title** Presentation-Ready Data Summary and Analytic Result  
Tables

**Version** 1.3.2

**Description** Creates presentation-ready tables summarizing data sets, regression models, and more. The code to create the tables is concise and highly customizable. Data frames can be summarized with any function, e.g. `mean()`, `median()`, even user-written functions. Regression models are summarized and include the reference rows for categorical variables. Common regression models, such as logistic regression and Cox proportional hazards regression, are automatically identified and the tables are pre-filled with appropriate column headers.

**License** MIT + file LICENSE

**URL** <https://github.com/ddsjoberg/gtsummary>,  
<http://www.danieldsjoberg.com/gtsummary/>

**BugReports** <https://github.com/ddsjoberg/gtsummary/issues>

**Depends** R (>= 3.4)

**Imports** broom (>= 0.5.6),  
broom.mixed (>= 0.2.6),  
crayon (>= 1.3.4),  
dplyr (>= 0.8.5),  
forcats (>= 0.5.0),  
glue (>= 1.4.0),  
gt (>= 0.2.1),  
knitr (>= 1.28),  
lifecycle (>= 0.2.0),  
magrittr (>= 1.5),  
purrr (>= 0.3.4),  
rlang (>= 0.4.6),  
stringr (>= 1.4.0),  
survival,  
tibble (>= 3.0.1),  
tidyr (>= 1.0.3),  
tidyselect (>= 1.1.0),  
usethis (>= 1.6.1)

**Suggests** car,  
covr,

flextable,  
geepack,  
Hmisc,  
kableExtra,  
lme4,  
officer,  
pkgdown,  
rmarkdown,  
scales,  
spelling,  
testthat

**VignetteBuilder** knitr

**RdMacros** lifecycle

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.0

## R topics documented:

add_global_p . . . . .	3
add_global_p.tbl_regression . . . . .	4
add_global_p.tbl_uvregression . . . . .	5
add_n . . . . .	6
add_nevent . . . . .	7
add_nevent.tbl_regression . . . . .	8
add_nevent.tbl_uvregression . . . . .	9
add_overall . . . . .	10
add_p . . . . .	11
add_p.tbl_cross . . . . .	11
add_p.tbl_summary . . . . .	12
add_q . . . . .	14
add_stat . . . . .	15
add_stat_label . . . . .	17
as_flextable . . . . .	18
as_gt . . . . .	20
as_kable . . . . .	21
as_kable_extra . . . . .	22
as_tibble.gtsummary . . . . .	23
bold_italicize_labels_levels . . . . .	24
bold_p . . . . .	25
combine_terms . . . . .	26
gtsummary_logo . . . . .	28
inline_text . . . . .	28
inline_text.tbl_cross . . . . .	29
inline_text.tbl_regression . . . . .	30
inline_text.tbl_summary . . . . .	31
inline_text.tbl_survfit . . . . .	32
inline_text.tbl_uvregression . . . . .	34

<i>add_global_p</i>	3
modify_footnote . . . . .	35
modify_header . . . . .	37
modify_spanning_header . . . . .	38
print_gtsummary . . . . .	39
select_helpers . . . . .	40
set_gtsummary_theme . . . . .	41
sort_p . . . . .	42
style_percent . . . . .	43
style_pvalue . . . . .	43
style_ratio . . . . .	44
style_sigfig . . . . .	45
tbl_cross . . . . .	46
tbl_merge . . . . .	47
tbl_regression . . . . .	49
tbl_stack . . . . .	51
tbl_summary . . . . .	53
tbl_survfit . . . . .	56
tbl_uvregression . . . . .	58
theme_gtsummary . . . . .	61
trial . . . . .	62
<b>Index</b>	<b>63</b>

---

<i>add_global_p</i>	<i>Adds the global p-value for a categorical variables</i>
---------------------	--

---

**Description**

This function uses `car::Anova` with argument `type = "III"` to calculate global p-values for categorical variables. Output from `tbl_regression` and `tbl_uvregression` objects supported.

**Usage**

```
add_global_p(x, ...)
```

**Arguments**

- `x` `tbl_regression` or `tbl_uvregression` object
- `...` Further arguments passed to or from other methods.

**Note**

If a needed class of model is not supported by `car::Anova`, please create a [GitHub Issue](#) to request support.

**Author(s)**

Daniel D. Sjoberg

**See Also**

[add\\_global\\_p.tbl\\_regression](#), [add\\_global\\_p.tbl\\_uvregression](#)

---

`add_global_p.tbl_regression`*Adds the global p-value for categorical variables*

---

### Description

This function uses `car::Anova` with argument `type = "III"` to calculate global p-values for categorical variables.

### Usage

```
## S3 method for class 'tbl_regression'  
add_global_p(  
  x,  
  include = x$table_body$variable[x$table_body$var_type %in% c("categorical",  
    "interaction")],  
  keep = FALSE,  
  terms = NULL,  
  quiet = NULL,  
  ...  
)
```

### Arguments

<code>x</code>	Object with class <code>tbl_regression</code> from the <code>tbl_regression</code> function
<code>include</code>	Variables to calculate global p-value for. Input may be a vector of quoted or unquoted variable names. <code>tidyselect</code> and <code>gtsummary</code> select helper functions are also accepted. Default is <code>NULL</code> , which adds global p-values for all categorical and interaction terms.
<code>keep</code>	Logical argument indicating whether to also retain the individual p-values in the table output for each level of the categorical variable. Default is <code>FALSE</code>
<code>terms</code>	DEPRECATED. Use <code>include=</code> argument instead.
<code>quiet</code>	Logical indicating whether to print messages in console. Default is <code>FALSE</code>
<code>...</code>	Additional arguments to be passed to <code>car::Anova</code>

### Value

A `tbl_regression` object

### Note

If a needed class of model is not supported by `car::Anova`, please create a [GitHub Issue](#) to request support.

### Example Output

### Author(s)

Daniel D. Sjoberg

**See Also**

Other `tbl_regression` tools: `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

**Examples**

```
tbl_lm_global_ex1 <-  
  lm(marker ~ age + grade, trial) %>%  
  tbl_regression() %>%  
  add_global_p()
```

---

`add_global_p.tbl_uvregression`

*Adds the global p-value for categorical variables*

---

**Description**

This function uses `car::Anova` with argument `type = "III"` to calculate global p-values for categorical variables.

**Usage**

```
## S3 method for class 'tbl_uvregression'  
add_global_p(x, quiet = NULL, ...)
```

**Arguments**

<code>x</code>	Object with class <code>tbl_uvregression</code> from the <code>tbl_uvregression</code> function
<code>quiet</code>	Logical indicating whether to print messages in console. Default is <code>FALSE</code>
<code>...</code>	Additional arguments to be passed to <code>car::Anova</code> .

**Value**

A `tbl_uvregression` object

**Example Output****Author(s)**

Daniel D. Sjoberg

**See Also**

Other `tbl_uvregression` tools: `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

## Examples

```
tbl_uv_global_ex2 <-
  trial[c("response", "trt", "age", "grade")] %>%
  tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial),
    exponentiate = TRUE
  ) %>%
  add_global_p()
```

---

add_n	<i>Add column with N</i>
-------	--------------------------

---

## Description

For each variable in a `tbl_summary` table, the `add_n` function adds a column with the total number of non-missing (or missing) observations

## Usage

```
add_n(
  x,
  statistic = "{n}",
  col_label = "***N**",
  footnote = FALSE,
  last = FALSE,
  missing = NULL
)
```

## Arguments

<code>x</code>	Object with class <code>tbl_summary</code> from the <a href="#">tbl_summary</a> function
<code>statistic</code>	String indicating the statistic to report. Default is the number of non-missing observation for each variable, <code>statistic = "{n}"</code> . Other statistics available to report include: <ul style="list-style-type: none"> <li>"{N}" total number of observations,</li> <li>"{n}" number of non-missing observations,</li> <li>"{n_miss}" number of missing observations,</li> <li>"{p}" percent non-missing data,</li> <li>"{p_miss}" percent missing data The argument uses <a href="#">glue::glue</a> syntax and multiple statistics may be reported, e.g. <code>statistic = "{n} / {N} ({p}%)"</code></li> </ul>
<code>col_label</code>	String indicating the column label. Default is <code>"***N**"</code>
<code>footnote</code>	Logical argument indicating whether to print a footnote clarifying the statistics presented. Default is <code>FALSE</code>
<code>last</code>	Logical indicator to include N column last in table. Default is <code>FALSE</code> , which will display N column first.
<code>missing</code>	DEPRECATED. Logical argument indicating whether to print N ( <code>missing = FALSE</code> ), or N missing ( <code>missing = TRUE</code> ). Default is <code>FALSE</code>

**Value**

A `tbl_summary` object

**Example Output****Author(s)**

Daniel D. Sjöberg

**See Also**

Other `tbl_summary` tools: [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

**Examples**

```
tbl_n_ex <-
  trial[c("trt", "age", "grade", "response")] %>%
  tbl_summary(by = trt) %>%
  add_n()
```

---

add\_nevent

*Add number of events to a regression table*

---

**Description**

Adds a column of the number of events to tables created with [tbl\\_regression](#) or [tbl\\_uvregression](#). Supported model types include GLMs with binomial distribution family (e.g. [stats::glm](#), [lme4::glmer](#), and [geepack::geeglm](#)) and Cox Proportion Hazards regression models ([survival::coxph](#)).

**Usage**

```
add_nevent(x, ...)
```

**Arguments**

`x` `tbl_regeression` or `tbl_uvregression` object  
`...` Additional arguments passed to or from other methods.

**Author(s)**

Daniel D. Sjöberg

**See Also**

[add\\_nevent.tbl\\_regression](#), [add\\_nevent.tbl\\_uvregression](#), [tbl\\_regression](#), [tbl\\_uvregression](#)

---

`add_nevent.tbl_regression`*Add number of events to a regression table*

---

## Description

This function adds a column of the number of events to tables created with `tbl_regression`. Supported model types include GLMs with binomial distribution family (e.g. `stats::glm`, `lme4::glmer`, and `geepack::geeglm`) and Cox Proportion Hazards regression models (`survival::coxph`).

The number of events is added to the internal `.$table_body` tibble, and not printed in the default output table (similar to `N`). The number of events is accessible via the `inline_text` function for printing in a report.

## Usage

```
## S3 method for class 'tbl_regression'  
add_nevent(x, ...)
```

## Arguments

<code>x</code>	tbl_regression object
<code>...</code>	Not used

## Value

A `tbl_regression` object

## Example Output

## Author(s)

Daniel D. Sjoberg

## See Also

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

## Examples

```
add_nevent_ex <-  
  glm(response ~ trt, trial, family = binomial) %>%  
  tbl_regression() %>%  
  add_nevent()
```



---

`add_nevent.tbl_uvregression`*Add number of events to a regression table*

---

## Description

Adds a column of the number of events to tables created with `tbl_uvregression`. Supported model types include GLMs with binomial distribution family (e.g. `stats::glm`, `lme4::glmer`, and `geepack::geeglm`) and Cox Proportion Hazards regression models (`survival::coxph`).

## Usage

```
## S3 method for class 'tbl_uvregression'  
add_nevent(x, ...)
```

## Arguments

<code>x</code>	tbl_uvregression object
<code>...</code>	Not used

## Value

A `tbl_uvregression` object

## Reporting Event N

The number of events is added to the internal `.$table_body` tibble, and printed to the right of the N column. The number of events is also accessible via the `inline_text` function for printing in a report.

## Example Output

## Author(s)

Daniel D. Sjoberg

## See Also

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

## Examples

```
tbl_uv_nevent_ex <-  
  trial[c("response", "trt", "age", "grade")] %>%  
  tbl_uvregression(  
    method = glm,  
    y = response,  
    method.args = list(family = binomial)  
  ) %>%  
  add_nevent()
```

---

add_overall	<i>Add column with overall summary statistics</i>
-------------	---

---

### Description

Adds a column with overall summary statistics to tables created by `tbl_summary`.

### Usage

```
add_overall(x, last = FALSE)
```

### Arguments

<code>x</code>	Object with class <code>tbl_summary</code> from the <a href="#">tbl_summary</a> function
<code>last</code>	Logical indicator to display overall column last in table. Default is <code>FALSE</code> , which will display overall column first.

### Value

A `tbl_summary` object

### Example Output

### Author(s)

Daniel D. Sjoberg

### See Also

Other `tbl_summary` tools: [add\\_n\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

### Examples

```
tbl_overall_ex <-  
  trial[c("age", "grade", "trt")] %>%  
  tbl_summary(by = trt) %>%  
  add_overall()
```

---

add_p	<i>Adds p-values to gtsummary table</i>
-------	---

---

**Description**

Adds p-values to gtsummary table

**Usage**

```
add_p(x, ...)
```

**Arguments**

x	Object created from a gtsummary function
...	Additional arguments passed to other methods.

**Author(s)**

Daniel D. Sjoberg

**See Also**

[add\\_p.tbl\\_summary](#), [add\\_p.tbl\\_cross](#)

---

add_p.tbl_cross	<i>Adds p-value to crosstab table</i>
-----------------	---------------------------------------

---

**Description**

**Experimental** Calculate and add a p-value comparing the two variables in the cross table. Missing values are included in p-value calculations.

**Usage**

```
## S3 method for class 'tbl_cross'
add_p(x, test = NULL, pvalue_fun = NULL, source_note = FALSE, ...)
```

**Arguments**

x	Object with class <code>tbl_cross</code> from the <a href="#">tbl_cross</a> function
test	A string specifying statistical test to perform. Default is "chisq.test" when expected cell counts $\geq 5$ and "fisher.test" when expected cell counts $< 5$ .
pvalue_fun	Function to round and format p-value. Default is <a href="#">style_pvalue</a> , except when <code>source_note = TRUE</code> when the default is <code>style_pvalue(x, prepend_p = TRUE)</code>
source_note	Logical value indicating whether to show p-value in the {gt} table source notes rather than a column.
...	Not used

## Example Output

### Author(s)

Karissa Whiting

### See Also

Other `tbl_cross` tools: [inline\\_text.tbl\\_cross\(\)](#), [tbl\\_cross\(\)](#)

### Examples

```
add_p_cross_ex1 <-
  trial %>%
  tbl_cross(row = stage, col = trt) %>%
  add_p()

add_p_cross_ex2 <-
  trial %>%
  tbl_cross(row = stage, col = trt) %>%
  add_p(source_note = TRUE)
```

---

`add_p.tbl_summary`      *Adds p-values to summary tables*

---

### Description

Adds p-values to tables created by `tbl_summary` by comparing values across groups.

### Usage

```
## S3 method for class 'tbl_summary'
add_p(
  x,
  test = NULL,
  pvalue_fun = NULL,
  group = NULL,
  include = everything(),
  exclude = NULL,
  ...
)
```

### Arguments

`x`                      Object with class `tbl_summary` from the [tbl\\_summary](#) function

`test`                    List of formulas specifying statistical tests to perform, e.g. `list(all_continuous() ~ "t.test", all_categorical() ~ "fisher.test")`. Options include

- "t.test" for a t-test,
- "aov" for a one-way ANOVA test,

- "wilcox.test" for a Wilcoxon rank-sum test,
- "kruskal.test" for a Kruskal-Wallis rank-sum test,
- "chisq.test" for a chi-squared test of independence,
- "chisq.test.no.correct" for a chi-squared test of independence without continuity correction,
- "fisher.test" for a Fisher's exact test,
- "lme4" for a random intercept logistic regression model to account for clustered data, `lme4::glmer(by ~ variable + (1 | group), family = binomial)`. The `by` argument must be binary for this option.

Tests default to "kruskal.test" for continuous variables, "chisq.test" for categorical variables with all expected cell counts  $\geq 5$ , and "fisher.test" for categorical variables with any expected cell count  $< 5$ . A custom test function can be added for all or some variables. See below for an example.

pvalue_fun	Function to round and format p-values. Default is <code>style_pvalue</code> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code> ).
group	Column name (unquoted or quoted) of an ID or grouping variable. The column can be used to calculate p-values with correlated data (e.g. when the test argument is "lme4"). Default is NULL. If specified, the row associated with this variable is omitted from the summary table.
include	Variables to include in output. Input may be a vector of quoted variable names, unquoted variable names, or <code>tidyselect</code> select helper functions. Default is <code>everything()</code> .
exclude	DEPRECATED
...	Not used

## Value

A `tbl_summary` object

## Setting Defaults

If you like to consistently use a different function to format p-values or estimates, you can set options in the script or in the user- or project-level startup file, '.Rprofile'. The default confidence level can also be set. Please note the default option for the estimate is the same as it is for `tbl_regression()`.

- `options(gtsummary.pvalue_fun = new_function)`

## Example Output

## Author(s)

Emily C. Zabor, Daniel D. Sjoberg

**See Also**

See `tbl_summary` [vignette](#) for detailed examples

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

**Examples**

```
# Example 1 -----
add_p_ex1 <-
  trial[c("age", "grade", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p()

# Example 2 -----
# Conduct a custom McNemar test for response,
# Function must return a named list of the p-value and the
# test name: list(p = 0.123, test = "McNemar's test")
# The '...' must be included as input
# This feature is experimental, and the API may change in the future
my_mcnemar <- function(data, variable, by, ...) {
  result <- list()
  result$p <- stats::mcnemar.test(data[[variable]], data[[by]])$p.value
  result$test <- "McNemar's test"
  result
}

add_p_ex2 <-
  trial[c("response", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p(test = response ~ "my_mcnemar")
```

---

 add\_q

---

*Add a column of q-values to account for multiple comparisons*


---

**Description**

Adjustments to p-values are performed with `stats::p.adjust`.

**Usage**

```
add_q(x, method = "fdr", pvalue_fun = NULL, quiet = NULL)
```

**Arguments**

<code>x</code>	a <code>gtsummary</code> object
<code>method</code>	String indicating method to be used for p-value adjustment. Methods from <code>stats::p.adjust</code> are accepted. Default is <code>method = "fdr"</code> .
<code>pvalue_fun</code>	Function to round and format p-values. Default is <code>style_pvalue</code> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code> ).

quiet Logical indicating whether to print messages in console. Default is FALSE

## Example Output

### Author(s)

Esther Drill, Daniel D. Sjoberg

### See Also

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

## Examples

```
# Example 1 -----
add_q_ex1 <-
  trial[c("trt", "age", "grade", "response")] %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  add_q()

# Example 2 -----
add_q_ex2 <-
  trial[c("trt", "age", "grade", "response")] %>%
  tbl_uvregression(
    y = response,
    method = glm,
    method.args = list(family = binomial),
    exponentiate = TRUE
  ) %>%
  add_global_p() %>%
  add_q()
```

---

add\_stat

*Add a custom statistic column*

---

## Description

**Experimental** The function allows a user to add a new column with a custom, user-defined statistic.

**Usage**

```
add_stat(
  x,
  fns,
  fmt_fun = NULL,
  header = "**Statistic**",
  footnote = NULL,
  new_col_name = NULL
)
```

**Arguments**

x	tbl_summary object
fns	list of formulas indicating the functions that create the statistic
fmt_fun	for numeric statistics, fmt_fun= is the styling/formatting function. Default is NULL
header	Column header of new column. Default is "**Statistic**"
footnote	Footnote associated with new column. Default is no footnote (i.e. NULL)
new_col_name	name of new column to be created in .stable_body. Default is "add_stat_1", unless that column exists then it is "add_stat_2", etc.

**Details**

The custom functions passed in fns= are required to follow a specified format. Each of these function will execute on a single variable from tbl\_summary().

1. Each function must return a single scalar or character value of length one.
2. Each function may take the following arguments: foo(data, variable, by, tbl)
  - data= is the input data frame passed to tbl\_summary()
  - variable= is a string indicating the variable to perform the calculation on
  - by= is a string indicating the by variable from tbl\_summary=, if present
  - tbl= the original tbl\_summary() object is also available to utilize

The user-defined does not need to utilize each of these inputs. It's encouraged the user-defined function accept ... as each of the arguments *will* be passed to the function, even if not all inputs are utilized by the user's function, e.g. foo(data, variable, by, ...)

**Example Output****Examples**

```
# Example 1 -----
# this example replicates `add_p()`

# fn returns t-test pvalue
my_ttest <- function(data, variable, by, ...) {
  t.test(data[[variable]] ~ as.factor(data[[by]]))$p.value
}
```



```

add_stat_ex1 <-
  trial %>%
  select(trt, age, marker) %>%
  tbl_summary(by = trt, missing = "no") %>%
  add_p(test = everything() ~ t.test) %>%
  # replicating result of `add_p()` with `add_stat()`
  add_stat(
    fns = everything() ~ my_ttest, # all variables compared with with t-test
    fmt_fun = style_pvalue,      # format result with style_pvalue()
    header = "**My p-value**"     # new column header
  )

# Example 2 -----
# fn returns t-test test statistic and pvalue
my_ttest2 <- function(data, variable, by, ...) {
  tt <- t.test(data[[variable]] ~ as.factor(data[[by]]))

  # returning test statistic and pvalue
  stringr::str_glue(
    "t={style_sigfig(tt$statistic)}, {style_pvalue(tt$p.value, prepend_p = TRUE)}"
  )
}

add_stat_ex2 <-
  trial %>%
  select(trt, age, marker) %>%
  tbl_summary(by = trt, missing = "no") %>%
  add_stat(
    fns = everything() ~ my_ttest2, # all variables will be compared by t-test
    fmt_fun = NULL, # fn returns and chr, so no formatting function needed
    header = "**Treatment Comparison**", # column header
    footnote = "T-test statistic and p-value" # footnote
  )
# Example 1 -----

```

---

add\_stat\_label

*Add statistic labels*


---

## Description

Adds labels describing the summary statistics presented for each variable in the [tbl\\_summary](#) table.

## Usage

```
add_stat_label(x, location = NULL, label = NULL)
```

## Arguments

x	Object with class <code>tbl_summary</code> from the <a href="#">tbl_summary</a> function
location	location where statistic label will be included. "row" (the default) to add the statistic label to the variable label row, and "column" adds a column with the statistic label.
label	a list of formulas or a single formula updating the statistic label, e.g. <code>label = all_categorical() ~ "No. (%)"</code>

**Value**

A tbl\_summary object

**Example Output****Author(s)**

Daniel D. Sjoberg

**See Also**

Other tbl\_summary tools: [add\\_n\(\)](#), [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [bold\\_italicize\\_labels\\_level\(\)](#), [inline\\_text.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

**Examples**

```
tbl <- trial %>%
  dplyr::select(trt, age, grade, response) %>%
  tbl_summary(by = trt)

# Example 1 -----
# Add statistic presented to the variable label row
add_stat_label_ex1 <-
  tbl %>%
  add_stat_label(
    # update default statistic label for continuous variables
    label = all_continuous() ~ "med. (iqr)"
  )

# Example 2 -----
add_stat_label_ex2 <-
  tbl %>%
  add_stat_label(
    # add a new column with statistic labels
    location = "column"
  )
```

---

as\_flextable

---

*Convert gtsummary object to a flextable object*


---

**Description**

**Experimental** Function converts a gtsummary object to a flextable object. A user can use this function if they wish to add customized formatting available via the flextable functions. The flextable output is particularly useful when combined with R markdown with Word output, since the gt package does not support Word.

**Usage**

```
as_flextable(x, ...)

## S3 method for class 'gtsummary'
as_flextable(
  x,
  include = everything(),
  return_calls = FALSE,
  strip_md_bold = TRUE,
  ...
)
```

**Arguments**

x	Object created by a function from the gtsummary package (e.g. <a href="#">tbl_summary</a> or <a href="#">tbl_regression</a> )
...	Not used
include	Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is everything().
return_calls	Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
strip_md_bold	When TRUE, all double asterisk (markdown language for bold weight) in column labels and spanning headers are removed. Default is TRUE

**Value**

A flextable object

**Details**

The `as_flextable()` takes the data frame that will be printed and converts it to a flextable and formats the table with the following flextable functions.

1. `flextable::flextable()`
2. `flextable::set_header_labels()` to set column labels
3. `flextable::add_header_row()`, if applicable, to set spanning column header
4. `flextable::align()` to set column alignment
5. `flextable::padding()` to indent variable levels
6. `flextable::fontsize()` to set font size
7. `flextable::autofit()` to estimate the column widths
8. `flextable::footnote()` to add table footnotes and source notes
9. `flextable::bold()` to bold cells in data frame
10. `flextable::italic()` to italicize cells in data frame

Any one of these commands may be omitted using the `include=` argument.

Pro tip: Use the `flextable::width()` function for exacting control over column width after calling `as_flextable()`.

**Author(s)**

Daniel D. Sjoberg

**See Also**Other gtsummary output types: [as\\_gt\(\)](#), [as\\_kable\\_extra\(\)](#), [as\\_kable\(\)](#), [as\\_tibble.gtsummary\(\)](#)**Examples**

```
trial %>%
  select(trt, age, grade) %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  as_flextable()
```

as\_gt

*Convert gtsummary object to a gt object***Description**

Function converts a gtsummary object to a gt\_tbl object. Function is used in the background when the results are printed or knit. A user can use this function if they wish to add customized formatting available via the [gt package](#).

Review the [tbl\\_summary vignette](#) or [tbl\\_regression vignette](#) for detailed examples in the 'Advanced Customization' section.

**Usage**

```
as_gt(
  x,
  include = everything(),
  return_calls = FALSE,
  exclude = NULL,
  omit = NULL
)
```

**Arguments**

x	Object created by a function from the gtsummary package (e.g. <a href="#">tbl_summary</a> or <a href="#">tbl_regression</a> )
include	Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is everything().
return_calls	Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
exclude	DEPRECATED.
omit	DEPRECATED.

**Value**

A gt\_tbl object

## Example Output

### Author(s)

Daniel D. Sjöberg

### See Also

Other gtsummary output types: [as\\_flextable\(\)](#), [as\\_kable\\_extra\(\)](#), [as\\_kable\(\)](#), [as\\_tibble.gtsummary\(\)](#)

### Examples

```
as_gt_ex <-
  trial[c("trt", "age", "response", "grade")] %>%
  tbl_summary(by = trt) %>%
  as_gt()
```

---

as\_kable

*Convert gtsummary object to a kable object*

---

### Description

Function converts a gtsummary object to a knitr\_kable object. This function is used in the background when the results are printed or knit. A user can use this function if they wish to add customized formatting available via [knitr::kable](#).

Output from [knitr::kable](#) is less full featured compared to summary tables produced with [gt](#). For example, kable summary tables do not include indentation, footnotes, or spanning header rows.

### Usage

```
as_kable(x, include = everything(), return_calls = FALSE, exclude = NULL, ...)
```

### Arguments

x	Object created by a function from the gtsummary package (e.g. <a href="#">tbl_summary</a> or <a href="#">tbl_regression</a> )
include	Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is <a href="#">everything()</a> , which includes all commands in x\$kable_calls.
return_calls	Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
exclude	DEPRECATED
...	Additional arguments passed to <a href="#">knitr::kable</a>

### Details

Tip: To better distinguish variable labels and level labels when indenting is not supported, try [bold\\_labels\(\)](#) or [italicize\\_levels\(\)](#).

**Value**

A knitr\_kable object

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other gtsummary output types: [as\\_flextable\(\)](#), [as\\_gt\(\)](#), [as\\_kable\\_extra\(\)](#), [as\\_tibble.gtsummary\(\)](#)

**Examples**

```
trial %>%
  tbl_summary(by = trt) %>%
  bold_labels() %>%
  as_kable()
```

---

as\_kable\_extra

*Convert gtsummary object to a kableExtra object*

---

**Description**

**Experimental** Function converts a gtsummary object to a knitr\_kable + kableExtra object. A user can use this function if they wish to add customized formatting available via [knitr::kable](#) and kableExtra. Note that gtsummary uses the standard markdown **\*\*** to bold headers, and they may need to be changed manually with kableExtra output.

**Usage**

```
as_kable_extra(
  x,
  include = everything(),
  return_calls = FALSE,
  strip_md_bold = TRUE,
  ...
)
```

**Arguments**

x	Object created by a function from the gtsummary package (e.g. <a href="#">tbl_summary</a> or <a href="#">tbl_regression</a> )
include	Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is everything(), which includes all commands in x\$kable_calls.
return_calls	Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
strip_md_bold	When TRUE, all double asterisk (markdown language for bold weight) in column labels and spanning headers are removed. Default is TRUE
...	Additional arguments passed to <a href="#">knitr::kable</a>

**Value**

A kableExtra object

**Author(s)**

Daniel D. Sjöberg

**See Also**

Other gtsummary output types: [as\\_flextable\(\)](#), [as\\_gt\(\)](#), [as\\_kable\(\)](#), [as\\_tibble.gtsummary\(\)](#)

**Examples**

```
tbl <-
  trial %>%
  tbl_summary(by = trt) %>%
  as_kable_extra()
```

---

as\_tibble.gtsummary    *Convert gtsummary object to a tibble*

---

**Description**

Function converts gtsummary objects tibbles. The formatting stored in x\$kable\_calls is applied.

**Usage**

```
## S3 method for class 'gtsummary'
as_tibble(
  x,
  include = everything(),
  col_labels = TRUE,
  return_calls = FALSE,
  exclude = NULL,
  ...
)
```

**Arguments**

x	Object created by a function from the gtsummary package (e.g. <a href="#">tbl_summary</a> or <a href="#">tbl_regression</a> )
include	Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is everything(), which includes all commands in x\$kable_calls.
col_labels	Logical argument adding column labels to output tibble. Default is TRUE.
return_calls	Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
exclude	DEPRECATED
...	Not used

**Value**

a [tibble](#)

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other gtsummary output types: [as\\_flextable\(\)](#), [as\\_gt\(\)](#), [as\\_kable\\_extra\(\)](#), [as\\_kable\(\)](#)

**Examples**

```
tbl <-  
  trial %>%  
  select(trt, age, grade, response) %>%  
  tbl_summary(by = trt)  
  
as_tibble(tbl)  
  
# without column labels  
as_tibble(tbl, col_labels = FALSE)
```

---

bold\_italicize\_labels\_levels

*Bold or Italicize labels or levels in gtsummary tables*

---

**Description**

Bold or Italicize labels or levels in gtsummary tables

**Usage**

```
bold_labels(x)
```

```
bold_levels(x)
```

```
italicize_labels(x)
```

```
italicize_levels(x)
```

**Arguments**

x                      Object created using gtsummary functions

**Value**

Functions return the same class of gtsummary object supplied



**Functions**

- `bold_labels`: Bold labels in gtsummary tables
- `bold_levels`: Bold levels in gtsummary tables
- `italicize_labels`: Italicize labels in gtsummary tables
- `italicize_levels`: Italicize levels in gtsummary tables

**Example Output****Author(s)**

Daniel D. Sjoberg

**See Also**

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

**Examples**

```
tbl_bold_ital_ex <-
  trial[c("trt", "age", "grade")] %>%
  tbl_summary() %>%
  bold_labels() %>%
  bold_levels() %>%
  italicize_labels() %>%
  italicize_levels()
```

---

`bold_p`

*Bold significant p-values or q-values*

---

**Description**

Bold values below a chosen threshold (e.g. <0.05) in a gtsummary tables.

**Usage**

```
bold_p(x, t = 0.05, q = FALSE)
```

**Arguments**

x	Object created using gtsummary functions
t	Threshold below which values will be bold. Default is 0.05.
q	Logical argument. When TRUE will bold the q-value column rather than the p-values. Default is FALSE.

**Example Output****Author(s)**

Daniel D. Sjoberg, Esther Drill

**Examples**

```
# Example 1 -----
bold_p_ex1 <-
  trial[c("age", "grade", "response", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  bold_p(t = 0.65)

# Example 2 -----
bold_p_ex2 <-
  glm(response ~ trt + grade, trial, family = binomial(link = "logit")) %>%
  tbl_regression(exponentiate = TRUE) %>%
  bold_p(t = 0.65)
```

---

combine\_terms

*Combine terms in a regression model*

---

**Description**

**Experimental** The function combines terms from a regression model, and replaces the terms with a single row in the output table. The p-value is calculated using `stats::anova()`.

**Usage**

```
combine_terms(x, formula_update, label = NULL, quiet = NULL, ...)
```

**Arguments**

x	a <code>tbl_regression</code> object
formula_update	formula update passed to the <code>stats::update</code> . This updated formula is used to construct a reduced model, and is subsequently passed to <code>stats::anova()</code> to calculate the p-value for the group of removed terms. See the <code>stats::update</code> help file for proper syntax. function's <code>formula.=</code> argument
label	Option string argument labeling the combined rows
quiet	Logical indicating whether to print messages in console. Default is FALSE
...	Additional arguments passed to <code>stats::anova</code>

**Value**

tbl\_regression object

**Example Output****Author(s)**

Daniel D. Sjoberg

**See Also**

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

**Examples**

```
# Example 1 -----
# fit model with nonlinear terms for marker
nlmod1 <- lm(
  age ~ marker + I(marker^2) + grade,
  trial[c("age", "marker", "grade")] %>% na.omit() # keep complete cases only!
)

combine_terms_ex1 <-
tbl_regression(nlmod1, label = grade ~ "Grade") %>%
# collapse non-linear terms to a single row in output using anova
combine_terms(
  formula_update = . ~ . - marker - I(marker^2),
  label = "Marker (non-linear terms)"
)

# Example 2 -----
# Example with Cubic Splines
library(Hmisc, warn.conflicts = FALSE, quietly = TRUE)
mod2 <- lm(
  age ~ rcspline.eval(marker, inclx = TRUE) + grade,
  trial[c("age", "marker", "grade")] %>% na.omit() # keep complete cases only!
)

combine_terms_ex2 <-
tbl_regression(mod2, label = grade ~ "Grade") %>%
combine_terms(
  formula_update = . ~ . -rcspline.eval(marker, inclx = TRUE),
  label = "Marker (non-linear terms)"
)

# Example 3 -----
# Logistic Regression Example, LRT p-value
combine_terms_ex3 <-
glm(
  response ~ marker + I(marker^2) + grade,
  trial[c("response", "marker", "grade")] %>% na.omit(), # keep complete cases only!
  family = binomial
```

```

) %>%
tbl_regression(label = grade ~ "Grade", exponentiate = TRUE) %>%
# collapse non-linear terms to a single row in output using anova
combine_terms(
  formula_update = . ~ . - marker - I(marker^2),
  label = "Marker (non-linear terms)",
  test = "LRT"
)

```

---

gtsummary\_logo

*The gtsummary logo, using ASCII or Unicode characters*


---

### Description

Use `crayon::strip_style()` to get rid of the colors.

### Usage

```
gtsummary_logo(unicode = 110n_info()$`UTF-8`)
```

### Arguments

unicode            Whether to use Unicode symbols. Default is TRUE on UTF-8 platforms.

### Examples

```
gtsummary_logo()
```

---

inline\_text

*Report statistics from gtsummary tables inline*


---

### Description

Report statistics from gtsummary tables inline

### Usage

```
inline_text(x, ...)
```

### Arguments

x                    Object created from a gtsummary function  
...                   Additional arguments passed to other methods.

### Value

A string reporting results from a gtsummary table

### Author(s)

Daniel D. Sjoberg

**See Also**

[inline\\_text.tbl\\_summary](#), [inline\\_text.tbl\\_regression](#), [inline\\_text.tbl\\_uvregression](#), [inline\\_text.tbl\\_survfit](#)

---

inline\_text.tbl\_cross *Report statistics from cross table inline*

---

**Description**

**Experimental** Extracts and returns statistics from a `tbl_cross` object for inline reporting in an R markdown document. Detailed examples in the [inline\\_text vignette](#)

**Usage**

```
## S3 method for class 'tbl_cross'
inline_text(x, col_level = NULL, row_level = NULL, pvalue_fun = NULL, ...)
```

**Arguments**

<code>x</code>	a <code>tbl_cross</code> object
<code>col_level</code>	Level of the column variable to display. Default is <code>NULL</code> . Can also specify "p.value" for the p-value and "stat_0" for Total column.
<code>row_level</code>	Level of the row variable to display. Can also specify the 'Unknown' row. Default is <code>NULL</code> .
<code>pvalue_fun</code>	Function to round and format p-values. Default is <a href="#">style_pvalue</a> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code> ).
<code>...</code>	Not used

**Value**

A string reporting results from a `gtsummary` table

**See Also**

Other `tbl_cross` tools: [add\\_p.tbl\\_cross\(\)](#), [tbl\\_cross\(\)](#)

**Examples**

```
tbl_cross <-
  tbl_cross(trial, row = trt, col = response) %>%
  add_p()

inline_text(tbl_cross, row_level = "Drug A", col_level = "1")
inline_text(tbl_cross, row_level = "Total", col_level = "1")
inline_text(tbl_cross, col_level = "p.value")
```

---

```
inline_text.tbl_regression
```

*Report statistics from regression summary tables inline*

---

## Description

Takes an object with class `tbl_regression`, and the location of the statistic to report and returns statistics for reporting inline in an R markdown document. Detailed examples in the [inline\\_text vignette](#)

## Usage

```
## S3 method for class 'tbl_regression'
inline_text(
  x,
  variable,
  level = NULL,
  pattern = "{estimate} ({conf.level*100})% CI {conf.low}, {conf.high}; {p.value})",
  estimate_fun = x$fmt_fun$estimate,
  pvalue_fun = NULL,
  ...
)
```

## Arguments

<code>x</code>	Object created from <a href="#">tbl_regression</a>
<code>variable</code>	Variable name of statistics to present
<code>level</code>	Level of the variable to display for categorical variables. Default is <code>NULL</code> , returning the top row in the table for the variable.
<code>pattern</code>	String indicating the statistics to return. Uses <a href="#">glue::glue</a> formatting. Default is <code>"{estimate} ({conf.level }% CI {conf.low}, {conf.high}; {p.value})"</code> . All columns from <code>x\$table_body</code> are available to print as well as the confidence level ( <code>conf.level</code> ). See below for details.
<code>estimate_fun</code>	function to style model coefficient estimates. Columns <code>'estimate'</code> , <code>'conf.low'</code> , and <code>'conf.high'</code> are formatted. Default is <code>x\$inputs\$estimate_fun</code>
<code>pvalue_fun</code>	function to style p-values and/or q-values. Default is <code>function(x) style_pvalue(x, prepend_p = TRUE)</code>
<code>...</code>	Not used

## Value

A string reporting results from a `gtsummary` table

## pattern argument

The following items are available to print. Use `print(x$table_body)` to print the table the estimates are extracted from.

- `{estimate}` coefficient estimate formatted with `'estimate_fun'`

- {conf.low} lower limit of confidence interval formatted with 'estimate\_fun'
- {conf.high} upper limit of confidence interval formatted with 'estimate\_fun'
- {ci} confidence interval formatted with x\$estimate\_fun
- {p.value} p-value formatted with 'pvalue\_fun'
- {N} number of observations in model
- {label} variable/variable level label

### Author(s)

Daniel D. Sjoberg

### See Also

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

### Examples

```
inline_text_ex1 <-
  glm(response ~ age + grade, trial, family = binomial(link = "logit")) %>%
  tbl_regression(exponentiate = TRUE)

inline_text(inline_text_ex1, variable = age)
inline_text(inline_text_ex1, variable = grade, level = "III")
```

---

```
inline_text.tbl_summary
```

*Report statistics from summary tables inline*

---

### Description

Extracts and returns statistics from a `tbl_summary` object for inline reporting in an R markdown document. Detailed examples in the [inline\\_text vignette](#)

### Usage

```
## S3 method for class 'tbl_summary'
inline_text(
  x,
  variable,
  column = NULL,
  level = NULL,
  pattern = NULL,
  pvalue_fun = NULL,
  ...
)
```

**Arguments**

x	Object created from <a href="#">tbl_summary</a>
variable	Variable name of statistic to present
column	Column name to return from x\$table_body. Can also pass the level of a by variable.
level	Level of the variable to display for categorical variables. Can also specify the 'Unknown' row. Default is NULL
pattern	String indicating the statistics to return. Uses <a href="#">glue::glue</a> formatting. Default is pattern shown in <code>tbl_summary()</code> output
pvalue_fun	Function to round and format p-values. Default is <a href="#">style_pvalue</a> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code> ).
...	Not used

**Value**

A string reporting results from a gtsummary table

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other `tbl_summary` tools: [add\\_n\(\)](#), [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

**Examples**

```
t1 <- trial[c("trt", "grade")] %>% tbl_summary(by = trt) %>% add_p()

inline_text(t1, variable = grade, level = "I", column = "Drug A", pattern = "{n}/{N} ({p})%")
inline_text(t1, variable = grade, column = "p.value")
```

---

inline\_text.tbl\_survfit

*Report statistics from survfit tables inline*

---

**Description**

**Experimental** Extracts and returns statistics from a `tbl_survfit` object for inline reporting in an R markdown document. Detailed examples in the [inline\\_text vignette](#)



**Usage**

```
## S3 method for class 'tbl_survfit'
inline_text(
  x,
  time = NULL,
  prob = NULL,
  level = NULL,
  estimate_fun = NULL,
  pvalue_fun = NULL,
  ...
)
```

**Arguments**

x	Object created from <a href="#">tbl_survfit</a>
time	time for which to return survival probabilities.
prob	probability with values in (0,1)
level	Level of the variable to display for categorical variables. Can also specify the 'Unknown' row. Default is NULL
estimate_fun	Function to round and format coefficient estimates. Default is <a href="#">style_sigfig</a> when the coefficients are not transformed, and <a href="#">style_ratio</a> when the coefficients have been exponentiated.
pvalue_fun	Function to round and format p-values. Default is <a href="#">style_pvalue</a> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code> ).
...	tbl_survfit used

**Value**

A string reporting results from a gtsummary table

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other `tbl_summary` tools: [add\\_n\(\)](#), [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_summary\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

**Examples**

```
library(survival)
# fit survfit
fit1 <- survfit(Surv(ttdeath, death) ~ trt, trial)
fit2 <- survfit(Surv(ttdeath, death) ~ 1, trial)

# summarize survfit objects
tbl1 <- tbl_survfit(
```

```

fit1,
times = c(12, 24),
label = "Treatment",
label_header = "**{time} Month**"
)

tbl2 <- tbl_survfit(
  fit2,
  probs = 0.5,
  label_header = "**Median Survival**"
)

# report results inline
inline_text(tbl1, time = 24, level = "Drug B")
inline_text(tbl2, prob = 0.5)

```

---

```
inline_text.tbl_uvregression
```

*Report statistics from regression summary tables inline*

---

## Description

Extracts and returns statistics from a table created by the `tbl_uvregression` function for inline reporting in an R markdown document. Detailed examples in the [inline\\_text vignette](#)

## Usage

```

## S3 method for class 'tbl_uvregression'
inline_text(
  x,
  variable,
  level = NULL,
  pattern = "{estimate} ({{conf.level*100}}% CI {conf.low}, {conf.high}); {p.value})",
  estimate_fun = x$fmt_fun$estimate,
  pvalue_fun = NULL,
  ...
)

```

## Arguments

<code>x</code>	Object created from <a href="#">tbl_uvregression</a>
<code>variable</code>	Variable name of statistics to present
<code>level</code>	Level of the variable to display for categorical variables. Default is <code>NULL</code> , returning the top row in the table for the variable.
<code>pattern</code>	String indicating the statistics to return. Uses <a href="#">glue::glue</a> formatting. Default is <code>"{estimate} ({{conf.level }}% CI {conf.low}, {conf.high}); {p.value})"</code> . All columns from <code>x\$table_body</code> are available to print as well as the confidence level ( <code>conf.level</code> ). See below for details.
<code>estimate_fun</code>	function to style model coefficient estimates. Columns <code>'estimate'</code> , <code>'conf.low'</code> , and <code>'conf.high'</code> are formatted. Default is <code>x\$inputs\$estimate_fun</code>
<code>pvalue_fun</code>	function to style p-values and/or q-values. Default is <code>function(x) style_pvalue(x, prepend_p = TRUE)</code>
<code>...</code>	Not used

**Value**

A string reporting results from a gtsummary table

**pattern argument**

The following items are available to print. Use `print(x$table_body)` to print the table the estimates are extracted from.

- `{estimate}` coefficient estimate formatted with `'estimate_fun'`
- `{conf.low}` lower limit of confidence interval formatted with `'estimate_fun'`
- `{conf.high}` upper limit of confidence interval formatted with `'estimate_fun'`
- `{ci}` confidence interval formatted with `x$estimate_fun`
- `{p.value}` p-value formatted with `'pvalue_fun'`
- `{N}` number of observations in model
- `{label}` variable/variable level label

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other `tbl_uvregression` tools: [add\\_global\\_p.tbl\\_uvregression\(\)](#), [add\\_nevent.tbl\\_uvregression\(\)](#), [add\\_q\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_uvregression\(\)](#)

**Examples**

```
inline_text_ex1 <-
  trial[c("response", "age", "grade")] %>%
  tbl_uvregression(
    method = glm,
    method.args = list(family = binomial),
    y = response,
    exponentiate = TRUE
  )

inline_text(inline_text_ex1, variable = age)
inline_text(inline_text_ex1, variable = grade, level = "III")
```

---

modify\_footnote

*Update gtsummary table footnote*

---

**Description**

Update gtsummary table footnote

**Usage**

```
modify_footnote(x, update, abbreviation = FALSE)
```

**Arguments**

x	a gtsummary object
update	list of formulas or a single formula specifying the footnote update. The LHS selects the columns from x\$table_body whose footnote will be updated, and the RHS is the new footnote. For example, update = stat_0 ~ "New footnote!" or update = starts_with("stat_") ~ "New footnote!". To delete the footnote, update the text to NA.
abbreviation	Logical indicating if an abbreviation is being updated. Abbreviation footnotes are handled differently. See examples below.

**Value**

gtsummary object

**Example Output****See Also**

Other tbl\_summary tools: [add\\_n\(\)](#), [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

Other tbl\_regression tools: [add\\_global\\_p.tbl\\_regression\(\)](#), [add\\_nevent.tbl\\_regression\(\)](#), [add\\_q\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [combine\\_terms\(\)](#), [inline\\_text.tbl\\_regression\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_regression\(\)](#), [tbl\\_stack\(\)](#)

Other tbl\_uvregression tools: [add\\_global\\_p.tbl\\_uvregression\(\)](#), [add\\_nevent.tbl\\_uvregression\(\)](#), [add\\_q\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_uvregression\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_uvregression\(\)](#)

**Examples**

```
tbl_summary <-
  trial %>%
  select(trt, age, grade) %>%
  tbl_summary(by = trt)

# Example 1 -----
# update footnote
modify_footnote_ex1 <-
  tbl_summary %>%
  modify_footnote(
    update = starts_with("stat_") ~
      "median (IQR) for continuous variables; n (%) categorical variables"
  )

# Example 2 -----
# delete all footnotes - except abbreviations
# use `modify_footnote(everything() ~ NA, abbreviation = TRUE)` to delete abbrev. footnotes
modify_footnote_ex2 <-
  tbl_summary %>%
  modify_footnote(update = everything() ~ NA)
```

```
# Example 3 -----
# updating the footnote abbreviation for CI
modify_footnote_ex3 <-
  glm(response ~ age + grade, trial, family = binomial) %>%
  tbl_regression(exponentiate = TRUE) %>%
  modify_footnote(ci ~ "CI = Credible Interval", abbreviation = TRUE)
```

---

modify_header	<i>Modify column headers in gtsummary tables</i>
---------------	--

---

## Description

Column labels can be modified to include calculated statistics; e.g. the N can be dynamically included by wrapping it in curly brackets (following `glue::glue` syntax).

## Usage

```
modify_header(
  x,
  update = NULL,
  stat_by = NULL,
  text_interpret = c("md", "html"),
  ...
)
```

## Arguments

x	gtsummary object, e.g. <code>tbl_summary</code> or <code>tbl_regression</code>
update	list of formulas or a single formula specifying the updated column label. Columns from <code>x\$table_body</code> may be selected.
stat_by	Used with <code>tbl_summary(by=)</code> objects with a <code>by=</code> argument. String specifying text to include above the summary statistics. The following fields are available for use in the headers: <ul style="list-style-type: none"> <li>• {n} number of observations in each group,</li> <li>• {N} total number of observations,</li> <li>• {p} percentage in each group,</li> <li>• {level} the 'by' variable level,</li> </ul> Syntax follows <code>glue::glue()</code> , e.g. <code>stat_by = "**{level}**, N = {n} ({style_percent(p)})"</code> .
text_interpret	String indicates whether text will be interpreted with <code>gt::md()</code> or <code>gt::html()</code> . Must be "md" (default) or "html".
...	Specify a column and updated column label, e.g. <code>modify_header(p.value = "Model P-values")</code> . This is provided as an alternative to the <code>update=</code> argument. They accomplish the same goal of updating column headers.

## Value

Function return the same class of gtsummary object supplied

## Example Output

**Author(s)**

Daniel D. Sjöberg

**See Also**

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_footnote()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

**Examples**

```
# create summary table
tbl <- trial[c("age", "grade", "trt")] %>%
  tbl_summary(by = trt, missing = "no") %>%
  add_p()

# print `.$table_body` to show column names and update headers
tbl$table_body

# Example 1 -----
# updating column headers
modify_header_ex1 <-
  tbl %>%
  modify_header(
    update = list(
      label ~ "**Variable**",
      p.value ~ "**P**"
    )
  )

# Example 2 -----
# using `stat_by=` argument to update headers
modify_header_ex2 <-
  tbl %>%
  modify_header(
    stat_by = "**{level}**", N = {n} ({style_percent(p)}%)"
  )
```

---

modify\_spanning\_header

*Update gsummary table spanning header*

---

**Description**

Update gsummary table spanning header

**Usage**

```
modify_spanning_header(x, update)
```

**Arguments**

x	a gtsummary object
update	list of formulas or a single formula specifying the update. The LHS selects the variables whose spanning header will be updated, and the RHS is the new spanning header. For example, update = starts_with("stat_") ~ "New spanning header!". Columns from x\$table_body may be selected. To remove all spanning headers, use update = everything() ~ NA.

**Value**

gtsummary object

**Example Output****See Also**

Other tbl\_summary tools: [add\\_n\(\)](#), [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_summary\(\)](#)

Other tbl\_regression tools: [add\\_global\\_p.tbl\\_regression\(\)](#), [add\\_nevent.tbl\\_regression\(\)](#), [add\\_q\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [combine\\_terms\(\)](#), [inline\\_text.tbl\\_regression\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_regression\(\)](#), [tbl\\_stack\(\)](#)

Other tbl\_uvregression tools: [add\\_global\\_p.tbl\\_uvregression\(\)](#), [add\\_nevent.tbl\\_uvregression\(\)](#), [add\\_q\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_uvregression\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#), [tbl\\_uvregression\(\)](#)

**Examples**

```
# Example 1 -----
# add header above summary statistics
spanning_header_ex1 <-
  trial %>%
  select(trt, age, grade) %>%
  tbl_summary(by = trt) %>%
  modify_spanning_header(starts_with("stat_") ~ "***Randomization Assignment**")
```

---

print\_gtsummary

*print and knit\_print methods for gtsummary objects*


---

**Description**

print and knit\_print methods for gtsummary objects

**Usage**

```
## S3 method for class 'gtsummary'  
print(x, print_engine = NULL, ...)  
  
## S3 method for class 'gtsummary'  
knit_print(x, ...)
```

**Arguments**

x	An object created using gtsummary functions
print_engine	String indicating the print method. Must be one of "gt", "kable", "kable_extra", "flextable", "tibble"
...	Not used

**Author(s)**

Daniel D. Sjoberg

**See Also**

[tbl\\_summary](#) [tbl\\_regression](#) [tbl\\_uvregression](#) [tbl\\_merge](#) [tbl\\_stack](#)

---

select\_helpers

*Select helper functions*

---

**Description**

Set of functions to supplement the tidyselct set of functions for selecting columns of data frames. `all_continuous()`, `all_categorical()`, and `all_dichotomous()` may only be used with `tbl_summary()`, where each variable has been classified into one of these three groups. All other helpers are available throughout the package.

**Usage**

```
all_continuous()  
  
all_categorical(dichotomous = TRUE)  
  
all_dichotomous()  
  
all_numeric()  
  
all_character()  
  
all_integer()  
  
all_double()  
  
all_logical()  
  
all_factor()
```



**Arguments**

dichotomous Logical indicating whether to include dichotomous variables. Default is TRUE

**Value**

A character vector of column names selected

**Examples**

```
select_ex1 <-  
  trial %>%  
  select(age, response, grade) %>%  
  tbl_summary(  
    statistic = all_continuous() ~ "{mean} ({sd})",  
    type = all_dichotomous() ~ "categorical"  
  )
```

---

set\_gtsummary\_theme *Set a gtsummary theme*

---

**Description**

**Experimental** Use this function to set preferences for the display of gtsummary tables. The default formatting and styling throughout the gtsummary package are taken from the published reporting guidelines of the top four urology journals: European Urology, The Journal of Urology, Urology and the British Journal of Urology International. Use this function to change the default reporting style to match another journal, or your own personal style.

**Usage**

```
set_gtsummary_theme(x)  
  
reset_gtsummary_theme()
```

**Arguments**

x A gtsummary theme function, e.g. theme\_gtsummary\_journal(), or a named list defining a gtsummary theme. See details below.

**Themes**

Review the [themes vignette](#) to create your own themes.

**Example Output****See Also**

Available [gtsummary themes](#)

**Examples**

```
# Setting JAMA theme for gtsummary
set_gtsummary_theme(theme_gtsummary_journal("jama"))
# Themes can be combined by including more than one
set_gtsummary_theme(theme_gtsummary_compact())

set_gtsummary_theme_ex1 <-
  trial %>%
  dplyr::select(age, grade, trt) %>%
  tbl_summary(by = trt) %>%
  add_stat_label() %>%
  as_gt()

# reset gtsummary theme
reset_gtsummary_theme()
```

---

 sort\_p

---

*Sort variables in table by ascending p-values*


---

**Description**

Sort tables created by gtsummary by p-values

**Usage**

```
sort_p(x, q = FALSE)
```

**Arguments**

x                    An object created using gtsummary functions  
 q                    Logical argument. When TRUE will sort by the q-value column

**Example Output****Author(s)**

Karissa Whiting

**Examples**

```
# Example 1 -----
sort_p_ex1 <-
  trial[c("age", "grade", "response", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  sort_p()

# Example 2 -----
sort_p_ex2 <-
  glm(response ~ trt + grade, trial, family = binomial(link = "logit")) %>%
  tbl_regression(exponentiate = TRUE) %>%
  sort_p()
```

---

style_percent	<i>Style percentages to be displayed in tables or text</i>
---------------	--

---

**Description**

Style percentages to be displayed in tables or text

**Usage**

```
style_percent(x, symbol = FALSE)
```

**Arguments**

x	numeric vector of percentages
symbol	Logical indicator to include percent symbol in output. Default is FALSE.

**Value**

A character vector of styled percentages

**Author(s)**

Daniel D. Sjoberg

**See Also**

See Table Gallery [vignette](#) for example

Other style tools: [style\\_pvalue\(\)](#), [style\\_ratio\(\)](#), [style\\_sigfig\(\)](#)

**Examples**

```
percent_vals <- c(-1, 0, 0.0001, 0.005, 0.01, 0.10, 0.45356, 0.99, 1.45)
style_percent(percent_vals)
style_percent(percent_vals, symbol = TRUE)
```

---

style_pvalue	<i>Style p-values to be displayed in tables or text</i>
--------------	---

---

**Description**

Style p-values to be displayed in tables or text

**Usage**

```
style_pvalue(x, digits = 1, prepend_p = FALSE)
```

**Arguments**

x	Numeric vector of p-values.
digits	Number of digits large p-values are rounded. Must be 1 or 2. Default is 1.
prepend_p	Logical. Should 'p=' be prepended to formatted p-value. Default is FALSE

**Value**

A character vector of styled p-values

**Author(s)**

Daniel D. Sjoberg

**See Also**

See `tbl_summary` vignette for examples

Other style tools: `style_percent()`, `style_ratio()`, `style_sigfig()`

**Examples**

```
pvals <- c(
  1.5, 1, 0.999, 0.5, 0.25, 0.2, 0.197, 0.12, 0.10, 0.0999, 0.06,
  0.03, 0.002, 0.001, 0.00099, 0.0002, 0.00002, -1
)
style_pvalue(pvals)
style_pvalue(pvals, digits = 2, prepend_p = TRUE)
```

---

style\_ratio

*Implement significant figure-like rounding for ratios*

---

**Description**

When reporting ratios, such as relative risk or an odds ratio, we'll often want the rounding to be similar on each side of the number 1. For example, if we report an odds ratio of 0.95 with a confidence interval of 0.70 to 1.24, we would want to round to two decimal places for all values. In other words, 2 significant figures for numbers less than 1 and 3 significant figures 1 and larger. `style_ratio()` performs significant figure-like rounding in this manner.

**Usage**

```
style_ratio(x, digits = 2)
```

**Arguments**

<code>x</code>	Numeric vector
<code>digits</code>	Integer specifying the number of significant digits to display for numbers below 1. Numbers larger than 1 will be be <code>digits + 1</code> . Default is <code>digits = 2</code> .

**Value**

A character vector of styled ratios

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other style tools: [style\\_percent\(\)](#), [style\\_pvalue\(\)](#), [style\\_sigfig\(\)](#)

**Examples**

```
c(
  0.123, 0.9, 1.1234, 12.345, 101.234, -0.123,
  -0.9, -1.1234, -12.345, -101.234
) %>%
  style_ratio()
```

---

style_sigfig	<i>Implement significant figure-like rounding</i>
--------------	---

---

**Description**

Converts a numeric argument into a string that has been rounded to a significant figure-like number. Scientific notation output is avoided, however, and additional significant figures may be displayed for large numbers. For example, if the number of significant digits requested is 2, 123 will be displayed (rather than 120 or  $1.2 \times 10^2$ ).

**Usage**

```
style_sigfig(x, digits = 2)
```

**Arguments**

x	Numeric vector
digits	Integer specifying the minimum number of significant digits to display

**Details**

If 2 sig figs are input, the number is rounded to 2 decimal places when  $\text{abs}(x) < 1$ , 1 decimal place when  $\text{abs}(x) \geq 1$  &  $\text{abs}(x) < 10$ , and to the nearest integer when  $\text{abs}(x) \geq 10$ .

**Value**

A character vector of styled numbers

**Author(s)**

Daniel D. Sjoberg

**See Also**

Other style tools: [style\\_percent\(\)](#), [style\\_pvalue\(\)](#), [style\\_ratio\(\)](#)

**Examples**

```
c(0.123, 0.9, 1.1234, 12.345, -0.123, -0.9, -1.1234, -12.345, NA, -0.001) %>%
  style_sigfig()
```

tbl\_cross

*Create a cross table of summary statistics***Description**

**Experimental** The function creates a cross table of two categorical variables.

**Usage**

```
tbl_cross(
  data,
  row = NULL,
  col = NULL,
  label = NULL,
  statistic = NULL,
  percent = c("none", "column", "row", "cell"),
  margin = c("column", "row"),
  missing = c("ifany", "always", "no"),
  missing_text = "Unknown",
  margin_text = "Total"
)
```

**Arguments**

data	A data frame
row	A column name in data to be used for columns of cross table.
col	A column name in data to be used for rows of cross table.
label	List of formulas specifying variables labels, e.g. <code>list(age ~ "Age, yrs", stage ~ "Path T Stage")</code> . If a variable's label is not specified here, the label attribute ( <code>attr(data\$age, "label")</code> ) is used. If attribute label is NULL, the variable name will be used.
statistic	A string with the statistic name in curly brackets to be replaced with the numeric statistic (see <code>glue::glue</code> ). The default is <code>{n}</code> . If percent argument is "column", "row", or "cell", default is <code>{n} ({p}%)</code> .
percent	Indicates the type of percentage to return. Must be one of "none", "column", "row", or "cell". Default is "cell" when <code>{N}</code> or <code>{p}</code> is used in statistic.
margin	Indicates which margins to add to the table. Default is <code>c("row", "column")</code>
missing	Indicates whether to include counts of NA values in the table. Allowed values are "no" (never display NA values), "ifany" (only display if any NA values), and "always" (includes NA count row for all variables). Default is "ifany".
missing_text	String to display for count of missing observations. Default is "Unknown".
margin_text	Text to display for margin totals. Default is "Total"

**Value**

A `tbl_cross` object

**Example Output****Author(s)**

Karissa Whiting, Daniel D. Sjoberg

**See Also**

Other tbl\_cross tools: [add\\_p.tbl\\_cross\(\)](#), [inline\\_text.tbl\\_cross\(\)](#)

**Examples**

```
# Example 1 -----
tbl_cross_ex1 <-
  trial %>%
  tbl_cross(row = trt, col = response)

# Example 2 -----
tbl_cross_ex2 <-
  trial %>%
  tbl_cross(row = stage, col = trt, percent = "cell") %>%
  add_p()
```

---

tbl_merge	<i>Merge two or more gtsummary objects</i>
-----------	--

---

**Description**

Merges two or more tbl\_regression, tbl\_uvregression, tbl\_stack, or tbl\_summary objects and adds appropriate spanning headers.

**Usage**

```
tbl_merge(tbls, tab_spanner = NULL)
```

**Arguments**

tbls	List of gtsummary objects to merge
tab_spanner	Character vector specifying the spanning headers. Must be the same length as tbls. The strings are interpreted with <code>gt::md</code> . Must be same length as tbls argument

**Value**

A tbl\_merge object

**Example Output**

**Author(s)**

Daniel D. Sjöberg

**See Also**[tbl\\_stack](#)

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_regression()`, `tbl_stack()`

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_stack()`, `tbl_uvregression()`

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_stack()`, `tbl_summary()`

**Examples**

```
# Example 1 -----
# Side-by-side Regression Models
library(survival)
t1 <-
  glm(response ~ trt + grade + age, trial, family = binomial) %>%
  tbl_regression(exponentiate = TRUE)
t2 <-
  coxph(Surv(ttdeath, death) ~ trt + grade + age, trial) %>%
  tbl_regression(exponentiate = TRUE)
tbl_merge_ex1 <-
  tbl_merge(
    tbls = list(t1, t2),
    tab_spanner = c("**Tumor Response**", "**Time to Death**")
  )

# Example 2 -----
# Descriptive statistics alongside univariate regression, with no spanning header
t3 <-
  trial[c("age", "grade", "response")] %>%
  tbl_summary(missing = "no") %>%
  add_n %>%
  modify_header(stat_0 ~ "**Summary Statistics**")
t4 <-
  tbl_uvregression(
    trial[c("ttdeath", "death", "age", "grade", "response")],
    method = coxph,
    y = Surv(ttdeath, death),
    exponentiate = TRUE,
    hide_n = TRUE
  )

tbl_merge_ex2 <-
  tbl_merge(tbls = list(t3, t4)) %>%
  modify_spanning_header(everything() ~ NA_character_)
```



---

tbl\_regression      *Display regression model results in table*


---

## Description

This function takes a regression model object and returns a formatted table that is publication-ready. The function is highly customizable allowing the user to obtain a bespoke summary table of the regression model results. Review the [tbl\\_regression vignette](#) for detailed examples.

## Usage

```
tbl_regression(
  x,
  label = NULL,
  exponentiate = FALSE,
  include = everything(),
  show_single_row = NULL,
  conf.level = NULL,
  intercept = FALSE,
  estimate_fun = NULL,
  pvalue_fun = NULL,
  tidy_fun = NULL,
  show_yesno = NULL,
  exclude = NULL
)
```

## Arguments

x	Regression model object
label	List of formulas specifying variables labels, e.g. <code>list(age ~ "Age, yrs", stage ~ "Path T Stage")</code>
exponentiate	Logical indicating whether to exponentiate the coefficient estimates. Default is FALSE.
include	Variables to include in output. Input may be a vector of quoted variable names, unquoted variable names, or <code>tidyselect</code> select helper functions. Default is <code>everything()</code> .
show_single_row	By default categorical variables are printed on multiple rows. If a variable is dichotomous (e.g. Yes/No) and you wish to print the regression coefficient on a single row, include the variable name(s) here—quoted and unquoted variable name accepted.
conf.level	Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
intercept	Logical argument indicating whether to include the intercept in the output. Default is FALSE
estimate_fun	Function to round and format coefficient estimates. Default is <a href="#">style_sigfig</a> when the coefficients are not transformed, and <a href="#">style_ratio</a> when the coefficients have been exponentiated.

pvalue_fun	Function to round and format p-values. Default is <code>style_pvalue</code> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code> ).
tidy_fun	Option to specify a particular tidier function if the model is not a <code>vetted model</code> or you need to implement a custom method. Default is NULL
show_yesno	DEPRECATED
exclude	DEPRECATED

### Value

A `tbl_regression` object

### Setting Defaults

If you prefer to consistently use a different function to format p-values or estimates, you can set options in the script or in the user- or project-level startup file, `’.Rprofile’`. The default confidence level can also be set.

- `options(gtsummary.pvalue_fun = new_function)`
- `options(gtsummary.tbl_regression.estimate_fun = new_function)`
- `options(gtsummary.conf.level = 0.90)`

### Note

The N reported in the output is the number of observations in the data frame `model.frame(x)`. Depending on the model input, this N may represent different quantities. In most cases, it is the number of people or units in your model. Here are some common exceptions.

1. Survival regression models including time dependent covariates.
2. Random- or mixed-effects regression models with clustered data.
3. GEE regression models with clustered data.

This list is not exhaustive, and care should be taken for each number reported.

### Example Output

#### Author(s)

Daniel D. Sjoberg

#### See Also

See `tbl_regression` [vignette](#) for detailed examples

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`

**Examples**

```

# Example 1 -----
library(survival)
tbl_regression_ex1 <-
  coxph(Surv(ttdeath, death) ~ age + marker, trial) %>%
  tbl_regression(exponentiate = TRUE)

# Example 2 -----
tbl_regression_ex2 <-
  glm(response ~ age + grade, trial, family = binomial(link = "logit")) %>%
  tbl_regression(exponentiate = TRUE)

# Example 3 -----
library(lme4, warn.conflicts = FALSE, quietly = TRUE)
tbl_regression_ex3 <-
  glmer(am ~ hp + (1 | gear), mtcars, family = binomial) %>%
  tbl_regression(exponentiate = TRUE)

```

tbl\_stack

*Stacks two or more gtsummary objects***Description**

Assists in patching together more complex tables. `tbl_stack()` appends two or more `tbl_regression`, `tbl_summary`, or `tbl_merge` objects. `gt` attributes from the first regression object are utilized for output table.

**Usage**

```
tbl_stack(tbls, group_header = NULL)
```

**Arguments**

<code>tbls</code>	List of <code>gtsummary</code> objects
<code>group_header</code>	Character vector with table headers where length matches the length of <code>tbls</code>

**Value**

A `tbl_stack` object

**Example Output****Author(s)**

Daniel D. Sjoberg

**See Also****tbl\_merge**

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_summary()`

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_regression()`

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_uvregression()`

**Examples**

```
# Example 1 -----
# stacking two tbl_regression objects
t1 <-
  glm(response ~ trt, trial, family = binomial) %>%
  tbl_regression(
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (unadjusted)")
  )

t2 <-
  glm(response ~ trt + grade + stage + marker, trial, family = binomial) %>%
  tbl_regression(
    include = "trt",
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (adjusted)")
  )

tbl_stack_ex1 <- tbl_stack(list(t1, t2))

# Example 2 -----
# stacking two tbl_merge objects
library(survival)
t3 <-
  coxph(Surv(ttdeath, death) ~ trt, trial) %>%
  tbl_regression(
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (unadjusted)")
  )

t4 <-
  coxph(Surv(ttdeath, death) ~ trt + grade + stage + marker, trial) %>%
  tbl_regression(
    include = "trt",
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (adjusted)")
  )

# first merging, then stacking
row1 <- tbl_merge(list(t1, t3), tab_spanner = c("Tumor Response", "Death"))
row2 <- tbl_merge(list(t2, t4))
```

```
tbl_stack_ex2 <-
  tbl_stack(list(row1, row2), group_header = c("Unadjusted Analysis", "Adjusted Analysis"))
```

---

tbl_summary	<i>Create a table of summary statistics</i>
-------------	---

---

## Description

The `tbl_summary` function calculates descriptive statistics for continuous, categorical, and dichotomous variables. Review the [tbl\\_summary vignette](#) for detailed examples.

## Usage

```
tbl_summary(
  data,
  by = NULL,
  label = NULL,
  statistic = NULL,
  digits = NULL,
  type = NULL,
  value = NULL,
  missing = NULL,
  missing_text = NULL,
  sort = NULL,
  percent = NULL,
  include = everything(),
  group = NULL
)
```

## Arguments

<code>data</code>	A data frame
<code>by</code>	A column name (quoted or unquoted) in <code>data</code> . Summary statistics will be calculated separately for each level of the <code>by</code> variable (e.g. <code>by = trt</code> ). If <code>NULL</code> , summary statistics are calculated using all observations.
<code>label</code>	List of formulas specifying variables labels, e.g. <code>list(age ~ "Age, yrs", stage ~ "Path T Stage")</code> . If a variable's label is not specified here, the <code>label</code> attribute ( <code>attr(data\$age, "label")</code> ) is used. If attribute <code>label</code> is <code>NULL</code> , the variable name will be used.
<code>statistic</code>	List of formulas specifying types of summary statistics to display for each variable. The default is <code>list(all_continuous() ~ "{median} ({p25},{p75})", all_categorical() ~ "{n} ({p}%")</code> . See below for details.
<code>digits</code>	List of formulas specifying the number of decimal places to round continuous summary statistics. If not specified, <code>tbl_summary</code> guesses an appropriate number of decimals to round statistics. When multiple statistics are displayed for a single variable, supply a vector rather than an integer. For example, if the statistic being calculated is <code>"{mean} ({sd})"</code> and you want the mean rounded to 1 decimal place, and the SD to 2 use <code>digits = list(age ~ c(1, 2))</code> .

type	List of formulas specifying variable types. Accepted values are <code>c("continuous", "categorical", "dichotomous")</code> , e.g. <code>type = list(age ~ "continuous", female ~ "dichotomous")</code> . If type not specified for a variable, the function will default to an appropriate summary type. See below for details.
value	List of formulas specifying the value to display for dichotomous variables. See below for details.
missing	Indicates whether to include counts of NA values in the table. Allowed values are "no" (never display NA values), "ifany" (only display if any NA values), and "always" (includes NA count row for all variables). Default is "ifany".
missing_text	String to display for count of missing observations. Default is "Unknown".
sort	List of formulas specifying the type of sorting to perform for categorical data. Options are frequency where results are sorted in descending order of frequency and alphanumeric, e.g. <code>sort = list(everything() ~ "frequency")</code>
percent	Indicates the type of percentage to return. Must be one of "column", "row", or "cell". Default is "column".
include	variables to include in the summary table. Default is <code>everything()</code>
group	DEPRECATED. Migrated to <a href="#">add_p</a>

### Value

A `tbl_summary` object

### select helpers

**Select helpers** from the `\tidyselect\` package and `\gtsummary\` package are available to modify default behavior for groups of variables. For example, by default continuous variables are reported with the median and IQR. To change all continuous variables to mean and standard deviation use `statistic = list(all_continuous() ~ "{mean} ({sd})")`.

All columns with class logical are displayed as dichotomous variables showing the proportion of events that are TRUE on a single row. To show both rows (i.e. a row for TRUE and a row for FALSE) use `type = list(all_logical() ~ "categorical")`.

The select helpers are available for use in any argument that accepts a list of formulas (e.g. `statistic`, `type`, `digits`, `value`, `sort`, etc.)

### statistic argument

The statistic argument specifies the statistics presented in the table. The input is a list of formulas that specify the statistics to report. For example, `statistic = list(age ~ "{mean} ({sd})")` would report the mean and standard deviation for age; `statistic = list(all_continuous() ~ "{mean} ({sd})")` would report the mean and standard deviation for all continuous variables. A statistic name that appears between curly brackets will be replaced with the numeric statistic (see [glue::glue](#)).

For categorical variables the following statistics are available to display.

- `{n}` frequency
- `{N}` denominator, or cohort size
- `{p}` formatted percentage

For continuous variables the following statistics are available to display.

- `{median}` median

- {mean} mean
- {sd} standard deviation
- {var} variance
- {min} minimum
- {max} maximum
- {p##} any integer percentile, where ## is an integer from 0 to 100
- {foo} any function of the form foo(x) is accepted where x is a numeric vector

For both categorical and continuous variables, statistics on the number of missing and non-missing observations and their proportions are available to display.

- {N\_obs} total number of observations
- {N\_miss} number of missing observations
- {N\_nonmiss} number of non-missing observations
- {p\_miss} percentage of observations missing
- {p\_nonmiss} percentage of observations not missing

Note that for categorical variables, {N\_obs}, {N\_miss} and {N\_nonmiss} refer to the total number, number missing and number non missing observations in the denominator, not at each level of the categorical variable.

### type argument

tbl\_summary displays summary statistics for three types of data: continuous, categorical, and dichotomous. If the type is not specified, tbl\_summary will do its best to guess the type. Dichotomous variables are categorical variables that are displayed on a single row in the output table, rather than one row per level of the variable. Variables coded as TRUE/FALSE, 0/1, or yes/no are assumed to be dichotomous, and the TRUE, 1, and yes rows are displayed. Otherwise, the value to display must be specified in the value argument, e.g. value = list(varname ~ "level to show")

### Example Output

#### Author(s)

Daniel D. Sjoberg

#### See Also

See [tbl\\_summary vignette](#) for detailed tutorial

See [table gallery](#) for additional examples

Other tbl\_summary tools: [add\\_n\(\)](#), [add\\_overall\(\)](#), [add\\_p.tbl\\_summary\(\)](#), [add\\_q\(\)](#), [add\\_stat\\_label\(\)](#), [bold\\_italicize\\_labels\\_levels](#), [inline\\_text.tbl\\_summary\(\)](#), [inline\\_text.tbl\\_survfit\(\)](#), [modify\\_footnote\(\)](#), [modify\\_header\(\)](#), [modify\\_spanning\\_header\(\)](#), [tbl\\_merge\(\)](#), [tbl\\_stack\(\)](#)

**Examples**

```
# Example 1 -----
tbl_summary_ex1 <-
  trial[c("age", "grade", "response")] %>%
  tbl_summary()

# Example 2 -----
tbl_summary_ex2 <-
  trial[c("age", "grade", "response", "trt")] %>%
  tbl_summary(
    by = trt,
    label = list(age ~ "Patient Age"),
    statistic = list(all_continuous() ~ "{mean} ({sd})"),
    digits = list(age ~ c(0, 1))
  )

# Example 3 -----
# for convenience, you can also pass named lists to any arguments
# that accept formulas (e.g label, digits, etc.)
tbl_summary_ex3 <-
  trial[c("age", "trt")] %>%
  tbl_summary(
    by = trt,
    label = list(age = "Patient Age")
  )
```

tbl\_survfit

*Creates table of survival probabilities***Description**

**Experimental** Function takes a `survfit` object as an argument, and provides a formatted summary table of the results

**Usage**

```
tbl_survfit(
  x,
  times = NULL,
  probs = NULL,
  statistic = "{estimate} ({conf.low}, {conf.high})",
  label = NULL,
  label_header = NULL,
  estimate_fun = NULL,
  missing = "--",
  conf.level = 0.95,
  failure = FALSE
)
```

**Arguments**

`x` `survfit` object. Object may have no stratification (e.g. `survfit(Surv(ttdeath, death) ~ 1, trial)`), or a single stratifying variable (e.g. `survfit(Surv(ttdeath, death) ~ trt, trial)`)



times	numeric vector of times for which to return survival probabilities.
probs	numeric vector of probabilities with values in (0,1) specifying the survival quantiles to return
statistic	string defining the statistics to present in the table. Default is "{estimate} ({conf.low},{conf.high})"
label	string specifying variable or overall label. Default is stratifying variable name or "Overall" when no stratifying variable present
label_header	string specifying column labels above statistics. Default is "{prob} Percentile" for survival percentiles, and "Time {time}" for n-year survival estimates
estimate_fun	function to format the Kaplan-Meier estimates. Default is <code>style_percent</code> for survival probabilities and <code>style_sigfig</code> for survival times
missing	text to fill when estimate is not estimable. Default is "--"
conf.level	Confidence level for confidence intervals. Default is 0.95
failure	Calculate failure probabilities rather than survival probabilities. Default is FALSE. Does not apply to survival quantile requests

## Example Output

### Author(s)

Daniel D. Sjoberg

### Examples

```
library(survival)
fit1 <- survfit(Surv(ttdeath, death) ~ trt, trial)
fit2 <- survfit(Surv(ttdeath, death) ~ 1, trial)

# Example 1 -----
tbl_survfit_ex1 <- tbl_survfit(
  fit1,
  times = c(12, 24),
  label = "Treatment",
  label_header = "**{time} Month**"
)

# Example 2 -----
tbl_survfit_ex2 <- tbl_survfit(
  fit2,
  probs = 0.5,
  label_header = "**Median Survival**"
)
```

---

tbl\_uvregression      *Display univariate regression model results in table*


---

### Description

This function estimates univariate regression models and returns them in a publication-ready table. It can create univariate regression models holding either a covariate or outcome constant.

For models holding outcome constant, the function takes as arguments a data frame, the type of regression model, and the outcome variable  $y$ . Each column in the data frame is regressed on the specified outcome. The `tbl_uvregression` function arguments are similar to the [tbl\\_regression](#) arguments. Review the [tbl\\_uvregression vignette](#) for detailed examples.

You may alternatively hold a single covariate constant. For this, pass a data frame, the type of regression model, and a single covariate in the  $x$  argument. Each column of the data frame will serve as the outcome in a univariate regression model. Take care using the  $x$  argument that each of the columns in the data frame are appropriate for the same type of model, e.g. they are all continuous variables appropriate for `lm`, or dichotomous variables appropriate for logistic regression with `glm`.

### Usage

```
tbl_uvregression(
  data,
  method,
  y = NULL,
  x = NULL,
  method.args = NULL,
  formula = "{y} ~ {x}",
  exponentiate = FALSE,
  label = NULL,
  include = everything(),
  exclude = NULL,
  hide_n = FALSE,
  show_single_row = NULL,
  conf.level = NULL,
  estimate_fun = NULL,
  pvalue_fun = NULL,
  show_yesno = NULL,
  tidy_fun = NULL
)
```

### Arguments

<code>data</code>	Data frame to be used in univariate regression modeling. Data frame includes the outcome variable(s) and the independent variables.
<code>method</code>	Regression method (e.g. <code>lm</code> , <code>glm</code> , <code>survival::coxph</code> , and more).
<code>y</code>	Model outcome (e.g. <code>y = recurrence</code> or <code>y = Surv(time, recur)</code> ). All other column in data will be regressed on $y$ . Specify one and only one of $y$ or $x$
<code>x</code>	Model covariate (e.g. <code>x = trt</code> ). All other columns in data will serve as the outcome in a regression model with $x$ as a covariate. Output table is best when $x$ is a continuous or dichotomous variable displayed on a single row. Specify one and only one of $y$ or $x$

method.args	List of additional arguments passed on to the regression function defined by method.
formula	String of the model formula. Uses <a href="#">glue::glue</a> syntax. Default is "{y} ~ {x}", where {y} is the dependent variable, and {x} represents a single covariate. For a random intercept model, the formula may be formula = "{y} ~ {x} + (1   gear)".
exponentiate	Logical indicating whether to exponentiate the coefficient estimates. Default is FALSE.
label	List of formulas specifying variables labels, e.g. list(age ~ "Age, yrs", stage ~ "Path T Stage")
include	Variables to include in output. Input may be a vector of quoted variable names, unquoted variable names, or tidyselect select helper functions. Default is everything().
exclude	DEPRECATED
hide_n	Hide N column. Default is FALSE
show_single_row	By default categorical variables are printed on multiple rows. If a variable is dichotomous (e.g. Yes/No) and you wish to print the regression coefficient on a single row, include the variable name(s) here—quoted and unquoted variable name accepted.
conf.level	Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
estimate_fun	Function to round and format coefficient estimates. Default is <a href="#">style_sigfig</a> when the coefficients are not transformed, and <a href="#">style_ratio</a> when the coefficients have been exponentiated.
pvalue_fun	Function to round and format p-values. Default is <a href="#">style_pvalue</a> . The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. pvalue_fun = function(x) style_pvalue(x, digits = 2) or equivalently, purrr::partial(style_pvalue, digits = 2)).
show_yesno	DEPRECATED
tidy_fun	Option to specify a particular tidier function if the model is not a <a href="#">vetted model</a> or you need to implement a custom method. Default is NULL

## Value

A tbl\_uvregression object

## Example Output

## Setting Defaults

If you prefer to consistently use a different function to format p-values or estimates, you can set options in the script or in the user- or project-level startup file, '.Rprofile'. The default confidence level can also be set.

- options(gtsummary.pvalue\_fun = new\_function)
- options(gtsummary.tbl\_regression.estimate\_fun = new\_function)
- options(gtsummary.conf.level = 0.90)

**Note**

The N reported in the output is the number of observations in the data frame `model.frame(x)`. Depending on the model input, this N may represent different quantities. In most cases, it is the number of people or units in your model. Here are some common exceptions.

1. Survival regression models including time dependent covariates.
2. Random- or mixed-effects regression models with clustered data.
3. GEE regression models with clustered data.

This list is not exhaustive, and care should be taken for each number reported.

**Author(s)**

Daniel D. Sjoberg

**See Also**

See `tbl_regression` [vignette](#) for detailed examples

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_footnote()`, `modify_header()`, `modify_spanning_header()`, `tbl_merge()`, `tbl_stack()`

**Examples**

```
# Example 1 -----
tbl_uv_ex1 <-
  tbl_uvregression(
    trial[c("response", "age", "grade")],
    method = glm,
    y = response,
    method.args = list(family = binomial),
    exponentiate = TRUE
  )

# Example 2 -----
# rounding pvalues to 2 decimal places
library(survival)
tbl_uv_ex2 <-
  tbl_uvregression(
    trial[c("ttdeath", "death", "age", "grade", "response")],
    method = coxph,
    y = Surv(ttdeath, death),
    exponentiate = TRUE,
    pvalue_fun = function(x) style_pvalue(x, digits = 2)
  )
```

---

theme_gtsummary	<i>Available gtsummary themes</i>
-----------------	-----------------------------------

---

## Description

**Experimental** The following themes are available to use within the gtsummary package. Use the `set_gtsummary_theme()` function to set a theme.

## Usage

```
theme_gtsummary_journal(journal = "jama")

theme_gtsummary_compact()

theme_gtsummary_printer(
  print_engine = c("gt", "kable", "kable_extra", "flextable", "tibble")
)
```

## Arguments

journal	String indicating the journal theme to follow. <ul style="list-style-type: none"> <li>• "jama" Journal of the American Medical Association</li> </ul>
print_engine	String indicating the print engine. Default is "gt"

## Themes

- theme\_gtsummary\_journal(journal=)
  - journal = "jama"
    - \* sets theme to align with the JAMA reporting guidelines
    - \* large p-values are rounded to two decimal places
    - \* in tbl\_summary() the IQR is separated with a dash, rather than comma
    - \* in tbl\_summary() the percent symbol is not printed next to percentages
- theme\_gtsummary\_compact()
  - tables printed with gt or flextable will be compact with smaller font size and reduced cell padding
- theme\_gtsummary\_printer(print\_engine=)
  - "gt" sets the gt package as the default print engine
  - "kable" sets the knitr::kable() function as the default print engine
  - "flextable" sets the flextable package as the default print engine
  - "kable\_extra" sets the kableExtra package as the default print engine

Use `reset_gtsummary_theme()` to restore the default settings

Review the [themes vignette](#) to create your own themes.

## Example Output

**See Also**[set\\_gtsummary\\_theme\(\)](#)**Examples**

```
# Setting JAMA theme for gtsummary
set_gtsummary_theme(theme_gtsummary_journal("jama"))
# Themes can be combined by including more than one
set_gtsummary_theme(theme_gtsummary_compact())

set_gtsummary_theme_ex1 <-
  trial %>%
  dplyr::select(age, grade, trt) %>%
  tbl_summary(by = trt) %>%
  add_stat_label() %>%
  as_gt()

# reset gtsummary theme
reset_gtsummary_theme()
```

---

trial

*Results from a simulated study of two chemotherapy agents*

---

**Description**

A dataset containing the baseline characteristics of 200 patients who received Drug A or Drug B. Dataset also contains the outcome of tumor response to the treatment.

**Usage**

```
trial
```

**Format**

A data frame with 200 rows—one row per patient

**trt** Chemotherapy Treatment

**age** Age, yrs

**marker** Marker Level, ng/mL

**stage** T Stage

**grade** Grade

**response** Tumor Response

**death** Patient Died

**ttdeath** Months to Death/Censor

# Index

- \* **datasets**
    - trial, 62
  - \* **gtsummary output types**
    - as\_flextable, 18
    - as\_gt, 20
    - as\_kable, 21
    - as\_kable\_extra, 22
    - as\_tibble.gtsummary, 23
  - \* **style tools**
    - style\_percent, 43
    - style\_pvalue, 43
    - style\_ratio, 44
    - style\_sigfig, 45
  - \* **tbl\_cross tools**
    - add\_p.tbl\_cross, 11
    - inline\_text.tbl\_cross, 29
    - tbl\_cross, 46
  - \* **tbl\_regression tools**
    - add\_global\_p.tbl\_regression, 4
    - add\_nevent.tbl\_regression, 8
    - add\_q, 14
    - bold\_italicize\_labels\_levels, 24
    - combine\_terms, 26
    - inline\_text.tbl\_regression, 30
    - modify\_footnote, 35
    - modify\_header, 37
    - modify\_spanning\_header, 38
    - tbl\_merge, 47
    - tbl\_regression, 49
    - tbl\_stack, 51
  - \* **tbl\_summary tools**
    - add\_n, 6
    - add\_overall, 10
    - add\_p.tbl\_summary, 12
    - add\_q, 14
    - add\_stat\_label, 17
    - bold\_italicize\_labels\_levels, 24
    - inline\_text.tbl\_summary, 31
    - inline\_text.tbl\_survfit, 32
    - modify\_footnote, 35
    - modify\_header, 37
    - modify\_spanning\_header, 38
    - tbl\_merge, 47
    - tbl\_stack, 51
    - tbl\_summary, 53
  - \* **tbl\_uvregression tools**
    - add\_global\_p.tbl\_uvregression, 5
    - add\_nevent.tbl\_uvregression, 9
    - add\_q, 14
    - bold\_italicize\_labels\_levels, 24
    - inline\_text.tbl\_uvregression, 34
    - modify\_footnote, 35
    - modify\_header, 37
    - modify\_spanning\_header, 38
    - tbl\_merge, 47
    - tbl\_stack, 51
    - tbl\_uvregression, 58
- add\_global\_p, 3
  - add\_global\_p.tbl\_regression, 3, 4, 8, 15, 25, 27, 31, 36, 38, 39, 48, 50, 52
  - add\_global\_p.tbl\_uvregression, 3, 5, 9, 15, 25, 35, 36, 38, 39, 48, 52, 60
  - add\_n, 6, 10, 14, 15, 18, 25, 32, 33, 36, 38, 39, 48, 52, 55
  - add\_nevent, 7
  - add\_nevent.tbl\_regression, 5, 7, 8, 15, 25, 27, 31, 36, 38, 39, 48, 50, 52
  - add\_nevent.tbl\_uvregression, 5, 7, 9, 15, 25, 35, 36, 38, 39, 48, 52, 60
  - add\_overall, 7, 10, 14, 15, 18, 25, 32, 33, 36, 38, 39, 48, 52, 55
  - add\_p, 11, 54
  - add\_p.tbl\_cross, 11, 11, 29, 47
  - add\_p.tbl\_summary, 7, 10, 11, 12, 15, 18, 25, 32, 33, 36, 38, 39, 48, 52, 55
  - add\_q, 5, 7–10, 14, 14, 18, 25, 27, 31–33, 35, 36, 38, 39, 48, 50, 52, 55, 60
  - add\_stat, 15
  - add\_stat\_label, 7, 10, 14, 15, 17, 25, 32, 33, 36, 38, 39, 48, 52, 55
  - all\_categorical (select\_helpers), 40
  - all\_character (select\_helpers), 40
  - all\_continuous (select\_helpers), 40
  - all\_dichotomous (select\_helpers), 40
  - all\_double (select\_helpers), 40
  - all\_factor (select\_helpers), 40

- all\_integer (select\_helpers), 40
- all\_logical (select\_helpers), 40
- all\_numeric (select\_helpers), 40
- as\_flextable, 18, 21–24
- as\_flextable(), 19
- as\_gt, 20, 20, 22–24
- as\_kable, 20, 21, 21, 23, 24
- as\_kable\_extra, 20–22, 22, 24
- as\_tibble.gtsummary, 20–23, 23
- bold\_italicize\_labels\_levels, 5, 7–10, 14, 15, 18, 24, 27, 31–33, 35, 36, 38, 39, 48, 50, 52, 55, 60
- bold\_labels
  - (bold\_italicize\_labels\_levels), 24
- bold\_labels(), 21
- bold\_levels
  - (bold\_italicize\_labels\_levels), 24
- bold\_p, 25
- car::Anova, 3–5
- combine\_terms, 5, 8, 15, 25, 26, 31, 36, 38, 39, 48, 50, 52
- crayon::strip\_style(), 28
- flextable::add\_header\_row(), 19
- flextable::align(), 19
- flextable::autofit(), 19
- flextable::bold(), 19
- flextable::flextable(), 19
- flextable::fontsize(), 19
- flextable::footnote(), 19
- flextable::italic(), 19
- flextable::padding(), 19
- flextable::set\_header\_labels(), 19
- flextable::width(), 19
- geepack::geeglm, 7–9
- glm, 58
- glue::glue, 6, 30, 32, 34, 37, 54, 59
- glue::glue(), 37
- gt::html(), 37
- gt::md(), 37
- gtsummary themes, 41
- gtsummary\_logo, 28
- inline\_text, 8, 9, 28
- inline\_text.tbl\_cross, 12, 29, 47
- inline\_text.tbl\_regression, 5, 8, 15, 25, 27, 29, 30, 36, 38, 39, 48, 50, 52
- inline\_text.tbl\_summary, 7, 10, 14, 15, 18, 25, 29, 31, 33, 36, 38, 39, 48, 52, 55
- inline\_text.tbl\_survfit, 7, 10, 14, 15, 18, 25, 29, 32, 32, 36, 38, 39, 48, 52, 55
- inline\_text.tbl\_uvregression, 5, 9, 15, 25, 29, 34, 36, 38, 39, 48, 52, 60
- italicize\_labels
  - (bold\_italicize\_labels\_levels), 24
- italicize\_levels
  - (bold\_italicize\_labels\_levels), 24
- italicize\_levels(), 21
- knit\_print.gtsummary (print\_gtsummary), 39
- knitr::kable, 21, 22
- lm, 58
- lme4::glmer, 7–9
- modify\_footnote, 5, 7–10, 14, 15, 18, 25, 27, 31–33, 35, 35, 38, 39, 48, 50, 52, 55, 60
- modify\_header, 5, 7–10, 14, 15, 18, 25, 27, 31–33, 35, 36, 37, 39, 48, 50, 52, 55, 60
- modify\_spanning\_header, 5, 7–10, 14, 15, 18, 25, 27, 31–33, 35, 36, 38, 38, 48, 50, 52, 55, 60
- print.gtsummary (print\_gtsummary), 39
- print\_gtsummary, 39
- reset\_gtsummary\_theme
  - (set\_gtsummary\_theme), 41
- select\_helpers, 40
- set\_gtsummary\_theme, 41
- set\_gtsummary\_theme(), 61, 62
- sort\_p, 42
- stats::anova, 26
- stats::anova(), 26
- stats::glm, 7–9
- stats::p.adjust, 14
- stats::update, 26
- style\_percent, 43, 44, 45, 57
- style\_pvalue, 11, 13, 14, 29, 32, 33, 43, 43, 45, 50, 59
- style\_ratio, 33, 43, 44, 44, 45, 49, 59
- style\_sigfig, 33, 43–45, 45, 49, 57, 59
- survival::coxph, 7–9, 58
- tbl\_cross, 11, 12, 29, 46
- tbl\_merge, 5, 7–10, 14, 15, 18, 25, 27, 31–33, 35, 36, 38–40, 47, 50, 52, 55, 60



tbl\_regression, [4](#), [5](#), [7](#), [8](#), [15](#), [19–23](#), [25](#), [27](#),  
[30](#), [31](#), [36](#), [38–40](#), [48](#), [49](#), [52](#), [58](#)  
tbl\_stack, [5](#), [7–10](#), [14](#), [15](#), [18](#), [25](#), [27](#), [31–33](#),  
[35](#), [36](#), [38–40](#), [48](#), [50](#), [51](#), [55](#), [60](#)  
tbl\_summary, [6](#), [7](#), [10](#), [12](#), [14](#), [15](#), [17–23](#), [25](#),  
[32](#), [33](#), [36](#), [38–40](#), [48](#), [52](#), [53](#)  
tbl\_survfit, [33](#), [56](#)  
tbl\_uvregression, [5](#), [7](#), [9](#), [15](#), [25](#), [34–36](#),  
[38–40](#), [48](#), [52](#), [58](#)  
theme\_gtsummary, [61](#)  
theme\_gtsummary\_compact  
    (theme\_gtsummary), [61](#)  
theme\_gtsummary\_journal  
    (theme\_gtsummary), [61](#)  
theme\_gtsummary\_printer  
    (theme\_gtsummary), [61](#)  
tibble, [24](#)  
trial, [62](#)  
  
vetted model, [50](#), [59](#)