Package 'ggforestplotR'

September 14, 2026

Title Publication-Ready Forest Plots with 'ggplot2'

Version 0.5.0

Description Transform model coefficients into flexible forest plots using 'ggplot2'. Provides helpers to standardize coefficient data from a range of modelling workflows and render publication-ready forest plots with a consistent interface.

License MIT + file LICENSE

Encoding UTF-8

Imports ggplot2, patchwork, rlang

Suggests broom, broom.mixed, dplyr, knitr, lme4, marginaleffects, nlme, rmarkdown, survival, testthat (>= 3.0.0), tibble

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website pkgdown

URL https://thatoneguy006.github.io/ggforestplotR/, https://github.com/thatoneguy006/ggforestplotR

BugReports https://github.com/thatoneguy006/ggforestplotR/issues

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Carson Richardson [aut, cre, cph] (ORCID: <https://orcid.org/0009-0004-8710-1097>)

Maintainer Carson Richardson <carson.richardson@outlook.com>

Repository CRAN

Date/Publication 2026-09-14 06:50:14 UTC

Contents

`add_favors` *Add directional favors labels beneath a forest plot*

Description

Compose a two-sided arrow annotation beneath the trained forest-plot x panel. The annotation is a separate footer plot, so it does not alter the forest plot's scales, limits, confidence intervals, or reference line.

Usage

```
add_favors(
  plot = NULL,
  left,
  right,
  reference = NULL,
  gap = 0.02,
  footer_height = 0.4,
  text_size = 3.2,
  linewidth = 0.5,
  arrow_length = 0.08,
  arrow_type = c("closed", "open")
)
```

Arguments

`plot`	A plot created by `ggforestplot()` or a supported forest-table composition. Leave as `NULL` to use `+ add_favors(...)` syntax.
`left, right`	Single strings shown beneath the left and right arrows.
`reference`	Optional numeric reference value. `NULL` uses the reference line resolved by `ggforestplot()`, falling back to the null value stored in the forest-data metadata.
`gap`	Gap on each side of the trained reference position, expressed as a fraction of the forest panel width.
`footer_height`	Footer height in inches.
`text_size`	Text size passed to `ggplot2::geom_text()`.
`linewidth`	Line width passed to `ggplot2::geom_segment()`.
`arrow_length`	Arrowhead length in inches.
`arrow_type`	Whether arrowheads are `"closed"` or `"open"`.

Details

add_favors() is designed as the final composition step. It works with a bare ggforestplot() result and with layouts returned by add_forest_table() and add_split_table().

Value

A patchwork-composed plot with a footer beneath only the forest-plot column, or a ggplot add-on object when plot = NULL.

Examples

```
coefs <- data.frame(
  term = c("Age", "BMI", "Treatment"),
  estimate = c(0.10, -0.08, 0.34),
  conf.low = c(0.02, -0.16, 0.12),
  conf.high = c(0.18, 0.00, 0.56)
)

ggforestplot(coefs) +
  add_favors(
    left = "Treatment A better",
    right = "Treatment B better"
  )

add_favors(
  ggforestplot(coefs),
  left = "Treatment A better",
  right = "Treatment B better"
)
```

---

add_forest_table *Add a summary table to a forest plot*

Description

Compose a summary table onto a forest plot.

Usage

```
add_forest_table(
  plot = NULL,
  position = c("left", "right"),
  columns = NULL,
  term_header = "Term",
  n_header = "N",
  events_header = "Events",
  estimate_label = NULL,
  p_header = "P-value",
```

```
  column_labels = NULL,
  digits = NULL,
  estimate_digits = NULL,
  interval_digits = NULL,
  p_digits = NULL,
  estimate_fmt = NULL,
  ci_fmt = NULL,
  text_size = NULL,
  header_text_size = NULL,
  header_fontface = "bold",
  header_family = NULL,
  striped_rows = NULL,
  stripe_fill = NULL,
  stripe_colour = NULL,
  stripe_alpha = NULL,
  grid_lines = FALSE,
  grid_line_colour = "black",
  grid_line_size = 0.3,
  grid_line_linetype = 1,
  table_width = NULL,
  plot_width = NULL
)
```

Arguments

plot	A plot created by ggforestplot(). Leave as NULL to use + add_forest_table(...) syntax.
position	Whether to place the table on the left or right of the forest plot.
columns	Optional explicit columns to display in the side table, in the order they should appear. Accepts built-in names such as "term", "group", "n", "events", "estimate", "ci", and "p", arbitrary original dataframe columns, or numeric positions in the supplied data. "conf.low" and "conf.high" are accepted as aliases for "ci". The default grouped table includes "group"; omit it from an explicit selection to hide it. Its header defaults to the source column supplied through group = for data frames and to "Model" for bind_forest_models() output.
term_header	Deprecated. Use the corresponding entry in column_labels, such as column_labels = c(term = "Variable").
n_header	Deprecated. Use column_labels = c(n = "N") instead.
events_header	Deprecated. Use column_labels = c(events = "Events") instead.
estimate_label	Header label for the estimate column. Defaults to the model-derived label when available.
p_header	Deprecated. Use column_labels = c(p = "P-value") instead.
column_labels	Optional named vector used to relabel table column headers. Names should match values supplied to columns after column resolution, such as "term", "group", "estimate", "ci", "p", or an arbitrary original dataframe column. For grouped data frames, either "group" or the source column supplied through

group = can relabel the group column. Use this argument instead of the deprecated dedicated header arguments.

digits — Deprecated. Number of digits used when formatting estimates and p-values. Defaults to 2. Use estimate_digits, interval_digits, and p_digits for separate control.

estimate_digits — Number of digits used for point estimates.

interval_digits — Number of digits used for confidence interval bounds.

p_digits — Number of digits used for p-values.

estimate_fmt — Format string for the estimate column. Use {estimate}, {conf.low}, and {conf.high} as placeholders. The shorthand {conf.low, conf.high} is also supported. Defaults to "{estimate} ({conf.low}, {conf.high})", or "{estimate}" when columns includes "ci".

ci_fmt — Format string for the confidence interval column when columns includes "ci". Use {conf.low} and {conf.high} as placeholders. The shorthand {conf.low, conf.high} is also supported. Defaults to "({conf.low}, {conf.high})".

text_size — Text size for table contents. Defaults to 3.2.

header_text_size — Header text size for table column labels. Defaults to 11.

header_fontface — Font face used for table column labels. Defaults to "bold".

header_family — Optional font family used for table column labels.

striped_rows — Whether to draw alternating row stripes behind the table. Defaults to the stripe setting used in ggforestplot().

stripe_fill — Fill colour used for striped rows. Defaults to the stripe fill used in ggforestplot().

stripe_colour — Outline colour for striped rows. Defaults to the stripe outline used in ggforestplot().

stripe_alpha — Transparency for striped rows. Defaults to the stripe alpha used in ggforestplot().

grid_lines — Whether to draw black horizontal grid lines in the table.

grid_line_colour — Colour used for the table grid lines.

grid_line_size — Line width used for the table grid lines.

grid_line_linetype — Line type used for the table grid lines.

table_width — Optional relative width allocated to the table panel. By default this is calculated from the displayed table content.

plot_width — Optional relative width allocated to the forest-plot panel. Defaults to 2.4.

Value

A patchwork-composed plot containing the forest plot and side table, or a ggplot add-on object when plot = NULL.

Examples

```
coefs <- data.frame(
  term = c("Age", "BMI", "Treatment"),
  estimate = c(0.3, -0.2, 0.4),
  conf.low = c(0.1, -0.4, 0.2),
  conf.high = c(0.5, 0.0, 0.6),
  sample_size = c(120, 115, 98),
  p_value = c(0.012, 0.031, 0.004)
)

p <- ggforestplot(coefs, n = "sample_size", p.value = "p_value")
add_forest_table(
  p,
  position = "left",
  columns = c("term", "n", "estimate", "p"),
  estimate_label = "Beta"
)

ggforestplot(coefs, n = "sample_size", p.value = "p_value") +
  add_forest_table(
    position = "right",
    columns = c("term", "n", "estimate", "p"),
    estimate_label = "Beta"
  )
```

---

add_split_table *Add split tables around a forest plot*

Description

Compose split table blocks around a forest plot so that summary data appear on both sides of the plotting panel.

Usage

```
add_split_table(
  plot = NULL,
  left_columns = NULL,
  right_columns = NULL,
  term_header = "Term",
  n_header = "N",
  events_header = "Events",
  estimate_label = NULL,
  p_header = "P-value",
  column_labels = NULL,
  digits = NULL,
  estimate_digits = NULL,
  interval_digits = NULL,
```

```
  p_digits = NULL,
  estimate_fmt = NULL,
  ci_fmt = NULL,
  text_size = NULL,
  header_text_size = NULL,
  header_fontface = "bold",
  header_family = NULL,
  striped_rows = NULL,
  stripe_fill = NULL,
  stripe_colour = NULL,
  stripe_alpha = NULL,
  left_width = NULL,
  plot_width = NULL,
  right_width = NULL
)
```

Arguments

plot	A plot created by ggforestplot(). Leave as NULL to use + add_split_table(...) syntax.
left_columns	Optional explicit columns to place on the left side of the forest plot. Accepts built-in names such as "term", "group", "n", "events", "estimate", "ci", and "p", arbitrary original dataframe columns, or numeric positions in the supplied data. "conf.low" and "conf.high" are accepted as aliases for "ci". The default grouped layout includes "group" on the left. Its header defaults to the source column supplied through group = for data frames and to "Model" for bind_forest_models() output.
right_columns	Optional explicit columns to place on the right side of the forest plot. Accepts built-in names such as "group", "estimate", "ci", and "p", arbitrary original dataframe columns, or numeric positions in the supplied data. "conf.low" and "conf.high" are accepted as aliases for "ci".
term_header	Deprecated. Use the corresponding entry in column_labels, such as column_labels = c(term = "Variable").
n_header	Deprecated. Use column_labels = c(n = "N") instead.
events_header	Deprecated. Use column_labels = c(events = "Events") instead.
estimate_label	Header label for the estimate column. Defaults to the model-derived label when available.
p_header	Deprecated. Use column_labels = c(p = "P-value") instead.
column_labels	Optional named vector used to relabel table column headers. Names should match values supplied to left_columns or right_columns after column resolution, such as "term", "group", "estimate", "ci", "p", or an arbitrary original dataframe column. For grouped data frames, either "group" or the source column supplied through group = can relabel the group column. Use this argument instead of the deprecated dedicated header arguments.
digits	Deprecated. Number of digits used when formatting estimates and p-values. Defaults to 2. Use estimate_digits, interval_digits, and p_digits for separate control.

estimate_digits
: Number of digits used for point estimates.

interval_digits
: Number of digits used for confidence interval bounds.

p_digits
: Number of digits used for p-values.

estimate_fmt
: Format string for the estimate column. Use {estimate}, {conf.low}, and {conf.high} as placeholders. The shorthand {conf.low, conf.high} is also supported. Defaults to "{estimate} ({conf.low}, {conf.high})", or "{estimate}" when table columns include "ci".

ci_fmt
: Format string for the confidence interval column when table columns include "ci". Use {conf.low} and {conf.high} as placeholders. The shorthand {conf.low, conf.high} is also supported. Defaults to "({conf.low}, {conf.high})".

text_size
: Text size for table contents. Defaults to 3.2.

header_text_size
: Header text size for table column labels. Defaults to 11.

header_fontface
: Font face used for table column labels. Defaults to "bold".

header_family
: Optional font family used for table column labels.

striped_rows
: Whether to draw alternating row stripes behind the split table layout. Defaults to the stripe setting used in ggforestplot().

stripe_fill
: Fill colour used for striped rows. Defaults to the stripe fill used in ggforestplot().

stripe_colour
: Outline colour for striped rows. Defaults to the stripe outline used in ggforestplot().

stripe_alpha
: Transparency for striped rows. Defaults to the stripe alpha used in ggforestplot().

left_width
: Optional width allocated to the left table block. By default this is derived from the number of displayed left-side columns relative to plot_width.

plot_width
: Optional width allocated to the forest plot panel. Defaults to 2.5.

right_width
: Optional width allocated to the right table block. By default this is derived from the number of displayed right-side columns relative to plot_width.

Value

A patchwork-composed plot containing a left table, the forest plot, and a right table, or a ggplot add-on object when plot = NULL.

Examples

```
coefs <- data.frame(
  term = c("Age", "BMI", "Treatment"),
  estimate = c(0.3, -0.2, 0.4),
  conf.low = c(0.1, -0.4, 0.2),
  conf.high = c(0.5, 0.0, 0.6),
  sample_size = c(120, 115, 98),
  p_value = c(0.012, 0.031, 0.004)
)

p <- ggforestplot(coefs, n = "sample_size", p.value = "p_value")
```

```
add_split_table(
  p,
  left_columns = c("term", "n"),
  right_columns = c("estimate", "p"),
  estimate_label = "HR"
)

ggforestplot(coefs, n = "sample_size", p.value = "p_value") +
  add_split_table(
    left_columns = c(1, 5),
    right_columns = c(2, 6),
    estimate_label = "HR"
  )
```

`as_forest_data` *Standardize coefficient data for forest plots*

Description

Standardizes a coefficient table into the internal forest-plot data structure used throughout `ggforestplotR`.

Usage

```
as_forest_data(data, ...)

## S3 method for class 'forest_data'
as_forest_data(
  data,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  exponentiate = NULL,
  p_method = NULL,
  ...
)

## S3 method for class 'data.frame'
as_forest_data(
  data,
  term,
  estimate,
  conf.low,
  conf.high,
  label = term,
  term_labels = NULL,
  group = NULL,
  grouping = NULL,
  separate_groups = NULL,
  n = NULL,
```

```
  events = NULL,
  p.value = NULL,
  exponentiate = NULL,
  estimate_scale = NULL,
  axis_transform = NULL,
  effect_label = NULL,
  conf.level = 0.95,
  reference_value = NULL,
  source_model = NULL,
  source_package = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  p_method = c("overall", "level"),
  ...
)

## S3 method for class 'lm'
as_forest_data(
  data,
  ...,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)

## S3 method for class 'glm'
as_forest_data(
  data,
  ...,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)

## S3 method for class 'coxph'
as_forest_data(
```

```
  data,
  ...,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)

## S3 method for class 'merMod'
as_forest_data(
  data,
  ...,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)

## S3 method for class 'lme'
as_forest_data(
  data,
  ...,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)

## S3 method for class 'glmmTMB'
as_forest_data(
  data,
  ...,
  conf.int = TRUE,
```

```
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)

## Default S3 method:
as_forest_data(
  data,
  ...,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)
```

Arguments

data	A data frame or data-frame subclass containing coefficient estimates and intervals. Tibbles and `data.table` objects are supported.
...	Arguments passed to an `as_forest_data()` method.
term_labels	Optional named vector used to relabel displayed terms. Names should match values in the term column and values are the labels to display.
sort_terms	How to sort rows: `"none"`, `"descending"`, or `"ascending"`. Subgroup hierarchies require `"none"` so their source order is preserved.
exponentiate	Compatibility argument. `TRUE` is equivalent to `estimate_scale = "ratio"`; `FALSE` is equivalent to `estimate_scale = "identity"` when `estimate_scale` is not supplied.
p_method	Subgroup p-value placement. `"overall"` displays one omnibus or block-level p-value on the subgroup header; `"level"` keeps p-values on the individual subgroup estimate rows. For fitted-model methods, this also selects whether `p.value` contains the omnibus interaction test or the post-estimation test for each derived effect.
term	Column name holding the model term identifier.
estimate	Column name holding the point estimate.
conf.low	Column name holding the lower confidence bound.
conf.high	Column name holding the upper confidence bound.

label	Optional column name used for the displayed row label.
group	Optional column name used for color-grouping multiple estimates per row. If this column is a factor, its levels control the group legend and vertical dodge order.
grouping	Optional column name used to split rows into grouped plot sections.
separate_groups	Optional column name used to identify labeled variable blocks that can be outlined with separator lines.
n	Optional column name holding sample sizes or other N labels for table helpers.
events	Optional column name holding event counts or event labels for table helpers.
p.value	Optional column name holding p-values.
estimate_scale	Semantic scale of the stored estimates. One of "identity", "log", "ratio", "probability", or "risk_difference".
axis_transform	Transformation used for the plotting axis. Defaults to "log10" for ratios and "identity" otherwise.
effect_label	Short label for the effect measure, such as "Beta", "OR", "HR", "RR", or "RD".
conf.level	Confidence level represented by the interval columns, or NA when it is unknown.
reference_value	Numeric null/reference value, or NULL when the effect measure has no universal reference value.
source_model	Optional character vector identifying the source model class. The complete fitted model is not retained.
source_package	Optional package name identifying the model source.
subgroup	For data frames, an optional column name defining presentation-only hierarchical subgroup blocks. Missing or empty values identify ordinary standalone estimates. Rows with the same non-empty value must form one contiguous block within each facet. When p.value is mapped, the first nonmissing value for each subgroup and estimate group, when applicable, is displayed on its parent row; child p-value cells are suppressed. Data-frame subgroups are never inferred and do not calculate model contrasts. For fitted-model methods, use "auto" to detect one unambiguous continuous-by-factor interaction, or supply a factor predictor name together with focal to derive covariance-aware post-estimation subgroup effects through marginaleffects. Fitted-model subgroup rows use the canonical p.value column for either the omnibus interaction test or row-level effect tests, according to p_method, so they can share a table column with ordinary coefficient p-values.
conf.int	Logical; model methods require TRUE because forest data include confidence-interval columns.
intercept	Logical; for model methods, whether to retain the intercept term.
focal	For fitted-model methods, the predictor whose conditional effect is estimated within each subgroup level. It may be continuous or a factor. Factor effects compare each non-reference level with the first level. Ignored for data-frame methods.

Value

A forest_data data-frame subclass ready for ggforestplot() and the table composition helpers. Original data-frame columns are retained for table helpers so they can be displayed with add_forest_table(columns = ...).

Examples

```
raw <- data.frame(
  variable = c("Age", "BMI", "Treatment"),
  beta = c(0.10, -0.08, 0.34),
  lower = c(0.02, -0.16, 0.12),
  upper = c(0.18, 0.00, 0.56)
)

as_forest_data(
  data = raw,
  term = "variable",
  estimate = "beta",
  conf.low = "lower",
  conf.high = "upper"
)
```

---

bind_forest_models *Bind multiple model summaries for a grouped forest plot*

Description

Tidies multiple fitted models and stacks their fixed-effect coefficient tables into a single forest-plot data frame. The resulting data can be passed directly to ggforestplot(), where model labels are used as the grouping variable for dodged, color-coded estimates.

Usage

```
bind_forest_models(models, model_labels = NULL, exponentiate = NULL, ...)
```

Arguments

models	A non-empty list of fitted model objects supported by an as_forest_data() method.
model_labels	Optional labels used to identify each model in the forest plot. Defaults to list names when present, otherwise "Model 1", "Model 2", and so on.
exponentiate	NULL, a single logical value, or one logical value per model. NULL uses the canonical scale inferred by as_forest_data() for each model.
...	Additional arguments passed to as_forest_data(), such as conf.level, intercept, term_labels, or sort_terms.

Value

A standardized forest-plot data frame with one row per model term and a `group` column containing the model labels.

Examples

```
if (requireNamespace("broom", quietly = TRUE)) {
  fit1 <- lm(mpg ~ wt + hp, data = mtcars)
  fit2 <- lm(mpg ~ wt + qsec, data = mtcars)

  bound <- bind_forest_models(
    list(Base = fit1, Adjusted = fit2)
  )

  ggforestplot(bound)
}
```

`forest_metadata` *Inspect forest-data metadata*

Description

Returns the semantic and provenance metadata attached to a `forest_data` object. Plotting behavior is determined by this metadata rather than by the class of the original fitted model.

Usage

```
forest_metadata(x)
```

Arguments

`x`	A `forest_data` object.

Value

A named metadata list, including the effect scale, reference value, source mappings, and subgroup `p_method` display contract.

`ggforestplot` *Draw a ggplot2 forest plot*

Description

Builds a forest plot from standardized coefficient data or directly from a fitted model.

Usage

```
ggforestplot(
  data,
  term = "term",
  estimate = "estimate",
  conf.low = "conf.low",
  conf.high = "conf.high",
  label = term,
  term_labels = NULL,
  group = NULL,
  facet = NULL,
  facet_strip_position = c("left", "right"),
  separate_groups = NULL,
  n = NULL,
  events = NULL,
  p.value = NULL,
  exponentiate = NULL,
  conf.level = 0.95,
  sort_terms = c("none", "descending", "ascending"),
  point_size = 2.3,
  point_shape = 19,
  linewidth = 0.5,
  line_size = NULL,
  staple_width = 0.2,
  ci_limits = NULL,
  ci_arrows = TRUE,
  ci_arrow_length = 0.08,
  ci_arrow_type = c("closed", "open"),
  dodge_width = 0.6,
  separate_lines = FALSE,
  separator_line_linetype = 2,
  separator_line_colour = "black",
  separator_line_size = 0.4,
  striped_rows = FALSE,
  stripe_fill = "grey95",
  stripe_colour = NA,
  stripe_alpha = 1,
  ref_line = NULL,
  ref_label = NULL,
```

```
  ref_linetype = 2,
  ref_color = "grey60",
  subgroup = NULL,
  p_method = c("overall", "level")
)
```

Arguments

data	Either a tidy coefficient data frame or a model object supported by broom::tidy().
term	Column name holding the model term identifiers.
estimate	Column name holding the point estimates.
conf.low	Column name holding the lower confidence bounds.
conf.high	Column name holding the upper confidence bounds.
label	Optional column name used for the displayed row labels.
term_labels	Optional named vector used to relabel displayed terms. Names should match values in the term column and values are the labels to display.
group	Optional column name used for color-grouping estimates. If this column is a factor, its levels control the group legend and vertical dodge order.
facet	Optional column name used to split rows into faceted plot sections. If this column is a factor, its levels control facet order.
facet_strip_position	Positioning for facet strip labels.
separate_groups	Optional column name used to identify labeled variable blocks that can be outlined with grid lines.
n	Optional column name holding sample sizes or other N labels for table helpers.
events	Optional column name holding event counts or event labels for table helpers.
p.value	Optional column name holding p-values.
exponentiate	Logical; if TRUE, transform the estimates and draw the axis on the log scale with the reference line at 1. For model objects, NULL uses the canonical scale when it can be inferred, such as hazard ratios for Cox models.
conf.level	Confidence level represented by model or data-frame interval columns. Defaults to 0.95.
sort_terms	How to sort rows: "none", "descending", or "ascending". Subgroup hierarchies require "none" so their source order is preserved.
point_size	Point size for coefficient markers.
point_shape	Shape used for coefficient markers.
linewidth	Line width for confidence intervals.
line_size	Deprecated. Use linewidth instead.
staple_width	Width of the terminal staples on confidence interval lines.
ci_limits	Optional numeric vector of length 2 used to truncate displayed confidence intervals. Intervals extending beyond these limits are clipped to the boundary and marked with arrows when ci_arrows is TRUE.

`ci_arrows` Logical; if `TRUE`, draw outward-facing arrows for confidence intervals truncated by `ci_limits`.

`ci_arrow_length` Length of CI truncation arrows in inches.

`ci_arrow_type` Arrowhead type for truncated confidence intervals. Passed to `grid::arrow()`.

`dodge_width` Horizontal dodging used for grouped estimates.

`separate_lines` Logical; if `TRUE`, draw grid lines around each labeled block identified by `separate_groups`.

`separator_line_linetype` Line type used for separator lines.

`separator_line_colour` Colour used for separator lines.

`separator_line_size` Line width used for separator lines.

`striped_rows` Logical; if `TRUE`, shade alternating rows.

`stripe_fill` Fill color used for shaded rows.

`stripe_colour` Border color for shaded rows.

`stripe_alpha` Transparency for shaded rows.

`ref_line` Numeric x-value where the reference line is drawn, or `NULL` to hide it. When omitted, the value is taken from the `forest_data` metadata, such as `0` for additive effects and `1` for ratios.

`ref_label` Optional label drawn alongside the reference line.

`ref_linetype` Line type for the reference line.

`ref_color` Color for the reference line.

`subgroup` Optional column name defining hierarchical subgroup blocks. Missing or empty values identify ordinary standalone estimates. Each non-empty subgroup must form one contiguous block within a facet. This is a presentation-only mapping; subgroups and contrasts are not inferred. With the default `p_method = "overall"` forest-data metadata, the first nonmissing p-value for each subgroup and estimate group, when applicable, is displayed on its parent row and child p-value cells are suppressed. `p_method = "level"` retains p-values on child rows. For fitted models, first call `tidy_forest_model()` with `subgroup`, `focal`, and the desired `p_method`; direct model calls do not perform post-estimation subgroup inference inside the plotting layer.

`p_method` Subgroup p-value placement for data-frame input. `"overall"` displays one p-value on each subgroup header; `"level"` displays p-values on the subgroup estimate rows. Fitted-model subgroup data inherit this choice from `tidy_forest_model()`.

Value

A `ggplot` object. Use standard `ggplot2` functions such as `ggplot2::labs()` for plot labels, and add composition helpers after styling the main plot.

Examples

```
coefs <- data.frame(
  term = c("Age", "BMI", "Treatment"),
  estimate = c(0.10, -0.08, 0.34),
  conf.low = c(0.02, -0.16, 0.12),
  conf.high = c(0.18, 0.00, 0.56)
)

ggforestplot(coefs)

ggforestplot(coefs, striped_rows = TRUE, point_shape = 17)
```

---

tidy_forest_model *Tidy a model object for forest plotting*

Description

Uses `broom::tidy()` to convert a fitted model into forest-plot data. When subgroup effects are requested, `marginaleffects::avg_slopes()` or `marginaleffects::avg_comparisons()` derives conditional average effects from the original fitted model and its covariance matrix. Mixed models are supported through `broom.mixed` tidy methods when that package is installed.

Usage

```
tidy_forest_model(
  model,
  conf.int = TRUE,
  conf.level = 0.95,
  exponentiate = NULL,
  intercept = FALSE,
  term_labels = NULL,
  sort_terms = c("none", "descending", "ascending"),
  subgroup = NULL,
  focal = NULL,
  p_method = c("overall", "level")
)
```

Arguments

`model`	A fitted model object supported by `broom::tidy()` or, for mixed models, a `broom.mixed` tidy method.
`conf.int`	Logical; if `TRUE`, request confidence intervals from `broom::tidy()`.
`conf.level`	Confidence level for intervals.
`exponentiate`	`NULL` uses the model's conventional coefficient scale, such as odds ratios for logistic models and hazard ratios for Cox models. `TRUE` or `FALSE` overrides that behavior.

intercept	Logical; if FALSE, drop the intercept term.
term_labels	Optional named vector used to relabel displayed terms. Names should match model term names and values are the labels to display.
sort_terms	How to sort rows: "none", "descending", or "ascending".
subgroup	NULL for ordinary coefficient rows, "auto" to detect one unambiguous continuous-by-factor interaction, or the name of a factor defining subgroup levels. Explicit selection also requires focal.
focal	Optional predictor whose conditional effect is estimated within each subgroup level. It may be continuous or a factor. For factors, each non-reference level is contrasted with the first factor level.
p_method	Subgroup p-value method. "overall" uses an omnibus Wald test of the selected interaction on the subgroup header. "level" uses the test returned for each subgroup-specific slope or comparison.

Details

With subgroup = NULL, the function retains its ordinary coefficient-tidy behavior. When subgroup effects are requested, interaction selection uses the fitted model's terms and model frame rather than parsing coefficient names. The selected focal main effect, subgroup main-effect coefficients, and raw interaction coefficients are replaced at their original position by one hierarchical subgroup block. Unrelated coefficient rows stay in formula order.

Continuous focal predictors use marginaleffects::avg_slopes() within each observed subgroup. Factor focal predictors use marginaleffects::avg_comparisons() and compare each non-reference level with the first factor level. Both functions use the original fitted model and its variance-covariance matrix; no subgroup models are refitted.

Automatic selection is deliberately conservative. It accepts one unambiguous continuous-by-factor interaction. Factor-by-factor interactions require explicit focal and subgroup names. Continuous subgroups, transformed focal terms, multiple interactions involving the selected predictors, and three-way interactions are rejected.

Linear and identity-link effects remain additive. Logit and log-link effects are estimated on the link scale and use the existing exponentiate semantics to return odds ratios or ratios by default. Cox effects are estimated on the linear-predictor scale and returned as hazard ratios by default. Other links fail rather than silently returning a response-scale estimand with a different interpretation.

The canonical p.value column always contains both ordinary-covariate and interaction-related tests. With p_method = "overall", subgroup rows store an omnibus Wald test of the selected interaction, which the display layer promotes to the parent header. With p_method = "level", subgroup rows retain the post-estimation p-value for each slope or comparison and display it alongside that estimate.

Value

A forest_data object ready for ggforestplot(). Derived rows add subgroup_level, focal, model_term, contrast, estimand, and effect_scale columns. Their canonical p.value follows p_method and shares one table column with ordinary covariate p-values.

Examples

```
if (requireNamespace("broom", quietly = TRUE)) {
  fit <- lm(mpg ~ wt + hp + qsec, data = mtcars)
  tidy_forest_model(fit)

  if (requireNamespace("marginaleffects", quietly = TRUE)) {
    interaction_fit <- lm(wt ~ mpg * factor(cyl), data = mtcars)
    tidy_forest_model(
      interaction_fit,
      subgroup = "auto",
      focal = "mpg"
    )
  }

  set.seed(123)
  logit_data <- data.frame(
    age = rnorm(250, mean = 62, sd = 8),
    bmi = rnorm(250, mean = 28, sd = 4),
    treatment = factor(rbinom(250, 1, 0.45), labels = c("Control", "Treatment"))
  )
  linpred <- -9 + 0.09 * logit_data$age + 0.11 * logit_data$bmi +
    0.9 * (logit_data$treatment == "Treatment")
  logit_data$event <- rbinom(250, 1, plogis(linpred))
  logit_fit <- glm(event ~ age + bmi + treatment, data = logit_data, family = binomial())

  tidy_forest_model(logit_fit, exponentiate = TRUE)
}
```

Index