

Package ‘forestploter’

July 22, 2025

Title Create a Flexible Forest Plot

Version 1.1.3

Description Create a forest plot based on the layout of the data. Confidence intervals in multiple columns by groups can be done easily. Editing the plot, inserting/adding text, applying a theme to the plot, and much more.

License MIT + file LICENSE

URL <https://github.com/adayim/forestploter>

BugReports <https://github.com/adayim/forestploter/issues>

Encoding UTF-8

RoxygenNote 7.3.2

Imports grid, gridExtra, gtable

Suggests gridtext, rmarkdown, knitr, vdiffr, testthat (>= 3.0.0), covr

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Alimu Dayimu [aut, cre] (ORCID:
<<https://orcid.org/0000-0001-9998-7463>>)

Maintainer Alimu Dayimu <ad938@cam.ac.uk>

Repository CRAN

Date/Publication 2025-04-13 16:00:07 UTC

Contents

add_border	2
add_grob	3
add_text	4
edit_plot	5
forest	6
forest_theme	9
get_wh	12

insert_text	13
makeci	14
make_boxplot	15
make_summary	17
print.forestplot	18

Index	19
--------------	-----------

add_border	<i>Add border to cells</i>
------------	----------------------------

Description

Add border to any cells at any side.
[add_underline](#) is a wrapper of [add_border](#) can be used to add underline to cells.

Usage

```
add_border(  
  plot,  
  row = NULL,  
  col = NULL,  
  part = c("body", "header"),  
  where = c("bottom", "left", "top", "right"),  
  gp = gpar(lwd = 2)  
)  
  
add_underline(  
  plot,  
  row = NULL,  
  col = NULL,  
  part = c("body", "header"),  
  gp = gpar(lwd = 2)  
)
```

Arguments

plot	A forest plot object.
row	A numeric value or vector indicating row number to add border. This is corresponding to the data row number. Remember to account for any text inserted. A border will be drawn to all rows if this is omitted.
col	A numeric value or vector indicating the columns to add border. A border will be drawn to all columns if this is omitted.
part	The border will be added to "body" (default) or "header".
where	Where to draw the border of the cell, possible values are "bottom" (default), "left", "top" and "right"
gp	An object of class "gpar", graphical parameter to be passed to segmentsGrob .

Value

A [gtable](#) object.

See Also

[gpar segmentsGrob gtable_add_grob](#)

add_grob	<i>Add grob in cells</i>
----------	--------------------------

Description

Draw grobs in any cells.

Usage

```
add_grob(
  plot,
  row = NULL,
  col = NULL,
  part = c("body", "header"),
  order = c("top", "text", "background", "bottom"),
  gb_fn,
  ...
)
```

Arguments

plot	A forest plot object.
row	A numeric value or vector indicating row(s) to draw a grob.
col	A numeric value or vector indicating the columns to draw a grob.
part	The border will be added to "body" (default) or "header".
order	Order in which the grobs should be plotted. Use 'top' (default) to draw the grob above everything, 'text' on the top of text given by plot data but below everything else, 'background' plot on the top of background but below everything else, 'bottom' below everything.
gb_fn	Grob function
...	Other parameters to be passed to gb_fn.

Value

A [gtable](#) object.

See Also

[gtable_add_grob](#)

add_text	<i>Add text to forest plot</i>
----------	--------------------------------

Description

This function can be used to add text to forest plot. The text can be span to multiple rows and columns. The height of the row will be changed accordingly if the text is added to only one row. The width of the text may exceeds the columns provided if the text is too long.

Usage

```
add_text(
  plot,
  text,
  row = NULL,
  col = NULL,
  part = c("body", "header"),
  just = c("center", "left", "right"),
  gp = gpar(),
  padding = unit(1, "mm"),
  parse = FALSE
)
```

Arguments

plot	A forest plot object.
text	A character or expression vector, see textGrob .
row	Row to add the text, this will be ignored if the part is "header".
col	A numeric value or vector indicating the columns the text will be added. The text will span over the column if a vector is given.
part	Part to add text, body (default) or header.
just	The justification of the text, "center" (default), "left" or "right".
gp	An object of class "gpar", this is the graphical parameter settings of the text. See gpar .
padding	Padding of the text, default is <code>unit(1, "mm")</code>
parse	Logical, behaviour for parsing text as plotmath, see plotmath

Value

A [gtable](#) object.

See Also

[gtable](#) [gpar](#) [textGrob](#) [gtable_add_grob](#)

edit_plot	<i>Edit forest plot</i>
-----------	-------------------------

Description

This function is used to edit the graphical parameters of text and background of the forest plot.

Usage

```
edit_plot(
  plot,
  row = NULL,
  col = NULL,
  part = c("body", "header"),
  which = c("text", "background", "ci"),
  gp = gpar(),
  ...
)
```

Arguments

plot	A forest plot object.
row	A numeric value or vector indicating row number to edit in the dataset. Will edit the whole row if left blank for the body. This will be ignored if the part is "header".
col	A numeric value or vector indicating column to edit in the dataset. Will edit the whole column if left blank.
part	Part to edit, "body" (default) or "header".
which	Which element to edit, "text", "background" or "ci" (confidence interval). This will not edit diamond shaped summary CI, please change it with forest_theme . Also, change in ci will not have any impact on the legend.
gp	Pass gpar parameters, see gpar . It should be passed as gpar(col = "red"). For which = "ci", please refer to forest_theme ci_* parameters for the editable elements.
...	Other parameters to be passed to the grobs. For the "text", please see textGrob for details and rectGrob for "background". This will be ignored if which = "ci" as changing parameters for the confidence interval are not allowed except for the graphical parameters.

Value

A [gtable](#) object.

See Also

[gpar](#) [editGrob](#) [forest_theme](#) [textGrob](#) [rectGrob](#)

forest	<i>Forest plot</i>
--------	--------------------

Description

A data frame will be used for the basic layout of the forest plot. Graphical parameters can be set using the `forest_theme` function.

Usage

```
forest(  
  data,  
  est,  
  lower,  
  upper,  
  sizes = 0.4,  
  ref_line = ifelse(x_trans %in% c("log", "log2", "log10"), 1, 0),  
  vert_line = NULL,  
  ci_column,  
  is_summary = NULL,  
  xlim = NULL,  
  ticks_at = NULL,  
  ticks_digits = NULL,  
  ticks_minor = NULL,  
  arrow_lab = NULL,  
  x_trans = "none",  
  xlab = NULL,  
  footnote = NULL,  
  title = NULL,  
  nudge_y = 0,  
  fn_ci = makeci,  
  fn_summary = make_summary,  
  index_args = NULL,  
  theme = NULL,  
  ...  
)
```

Arguments

data	Data to be displayed in the forest plot
est	Point estimation. Can be a list for multiple columns and/or multiple groups. If the length of the list is larger than then length of ci_column, then the values reused for each column and considered as different groups.
lower	Lower bound of the confidence interval, same as est.
upper	Upper bound of the confidence interval, same as est.

sizes	Size of the point estimation box, can be a unit, vector or a list. Values will be used as it is, no transformation will be applied.
ref_line	X-axis coordinates of zero line, default is 1. Provide an atomic vector if different reference line for each ci_column is desired.
vert_line	Numerical vector, add additional vertical line at given value. Provide a list of numerical vector element if different vertical line for each ci_column is desired.
ci_column	Column number of the data the CI will be displayed.
is_summary	A logical vector indicating if the value is a summary value, which will have a diamond shape for the estimate. Can not be used with multiple group forestplot.
xlim	Limits for the x axis as a vector of length 2, i.e. c(low, high). It will take the minimum and maximum of the lower and upper value if not provided. This will apply to all CI columns if provided, and will be calculated automatically for each column if not provided. This should be a list with the same length of ci_column if different xlim for different column is desired.
ticks_at	Set X-axis tick-marks point. This will apply to all CI columns if provided, and will be calculated automatically for each column if not provided. This should be a list if different ticks_at for different column is desired. Although many efforts have been made to automatically get a pretty ticks break, it will not give a perfect solution, especially if 'log2' and 'log10' defined for x_trans. Please provide this value if possible.
ticks_digits	Number of digits for the x-axis, default is NULL to calculate an integer based on ticks_at if provided or lower and upper if not. This should be a numerical vector if different rounding will be applied to different column. If an integer is specified, for example 1L, trailing zeros after the decimal mark will be dropped. Specify numeric, for example 1, to keep the trailing zero after the decimal mark.
ticks_minor	A numeric vector of positions to draw ticks without labels (optional). The values provided here can be superset or disjoint sets of ticks_at.
arrow_lab	Labels for the arrows, string vector of length two (left and right). The theme of arrow will inherit from the x-axis. This should be a list if different arrow labels for each column is desired.
x_trans	Change axis scale, Allowed values are one of c("none", "log", "log2", "log10"). Default is "none", no transformation will be applied. The formatted label will be used for scale = "log2" or "log10", change this in x_trans. Set this to "log" if x-axis tick marks assume values are exponential, e.g. for logistic regression (OR), survival estimates (HR), Poisson regression etc.
xlab	X-axis labels, it will be put under the x-axis. An atomic vector should be provided if different xlab for different column is desired.
footnote	Footnote for the forest plot, will be aligned at left bottom of the plot. Please adjust the line length with line break to avoid the overlap with the arrow and/or x-axis.
title	The text for the title.
nudge_y	Horizontal adjustment to nudge groups by, must be within 0 to 1.
fn_ci	Name of the function to draw confidence interval, default is makeci . You can specify your own drawing function to draw the confidence interval, but the function needs to accept arguments "est", "lower", "upper", "sizes", "xlim",

	"pch", "gp", "t_height", "nudge_y". Please refer to the makeci function for the details of these parameters.
fn_summary	Name of the function to draw summary confidence interval, default is make_summary . You can specify your own drawing function to draw the summary confidence interval, but the function needs to accept arguments "est", "lower", "upper", "sizes", "xlim", "gp". Please refer to the make_summary function for the details of these parameters.
index_args	A character vector, name of the arguments used for indexing the row and column. This should be the name of the arguments that is working the same way as est, lower and upper. Check out the examples in the make_boxplot .
theme	Theme of the forest plot, see forest_theme for details.
...	Other arguments passed on to the fn_ci and fn_summary.

Value

A [gtable](#) object.

See Also

[gtable](#) [tableGrob](#) [forest_theme](#) [make_boxplot](#) [makeci](#) [make_summary](#)

Examples

```
library(grid)
# Read provided sample example data
dt <- read.csv(system.file("extdata", "example_data.csv", package = "forestploter"))

# Keep needed columns
dt <- dt[,1:6]

# indent the subgroup if there is a number in the placebo column
dt$Subgroup <- ifelse(is.na(dt$Placebo),
                     dt$Subgroup,
                     paste0("  ", dt$Subgroup))

# NA to blank or NA will be transformed to character.
dt$Treatment <- ifelse(is.na(dt$Treatment), "", dt$Treatment)
dt$Placebo <- ifelse(is.na(dt$Placebo), "", dt$Placebo)
dt$se <- (log(dt$hi) - log(dt$est))/1.96

# Add blank column for the forest plot to display CI.
# Adjust the column width with space.
dt$` ` <- paste(rep(" ", 20), collapse = " ")

# Create confidence interval column to display
dt$`HR (95% CI)` <- ifelse(is.na(dt$se), "",
                          sprintf("%.2f (%.2f to %.2f)",
                                  dt$est, dt$low, dt$hi))

# Define theme
```

```

tm <- forest_theme(base_size = 10,
                   refline_gp = gpar(col = "red"),
                   footnote_gp = gpar(col = "#636363", fontface = "italic"))

p <- forest(dt[,c(1:3, 8:9)],
            est = dt$est,
            lower = dt$low,
            upper = dt$hi,
            sizes = dt$se,
            ci_column = 4,
            ref_line = 1,
            arrow_lab = c("Placebo Better", "Treatment Better"),
            xlim = c(0, 4),
            ticks_at = c(0.5, 1, 2, 3),
            footnote = "This is the demo data. Please feel free to change\nanything you want.",
            theme = tm)

# Print plot
plot(p)

```

forest_theme

Forest plot default theme

Description

Default theme for the forest plot, but can pass other parameters. The parameters will be passed to corresponding elements of the forest plot.

- `ci_*` Control the graphical parameters of confidence intervals
- `legend_*` Control the graphical parameters of legend
- `xaxis_*` Control the graphical parameters of x-axis
- `refline_*` Control the graphical parameters of reference line
- `vertline_*` Control the graphical parameters of vertical line
- `summary_*` Control the graphical parameters of diamond shaped summary CI
- `footnote_*` Control the graphical parameters of footnote
- `title_*` Control the graphical parameters of title
- `arrow_*` Control the graphical parameters of arrow

See [gpar](#) for more details.

Usage

```

forest_theme(
  base_size = 12,
  base_family = "",
  ci_pch = 15,

```

```

ci_col = "black",
ci_alpha = 1,
ci_fill = NULL,
ci_lty = 1,
ci_lwd = 1,
ci_Theight = NULL,
legend_name = "Group",
legend_position = "right",
legend_value = "",
legend_gp = gpar(fontsize = base_size, fontfamily = base_family, cex = 1),
legend_ncol = 1,
legend_byrow = TRUE,
xaxis_gp = gpar(fontsize = base_size, fontfamily = base_family, lwd = 0.6, cex = 1),
refline_gp = gpar(lwd = 1, lty = "dashed", col = "grey20"),
vertline_lwd = 1,
vertline_lty = "dashed",
vertline_col = "grey20",
summary_col = "#4575b4",
summary_fill = summary_col,
footnote_gp = gpar(fontsize = base_size, fontfamily = base_family, cex = 0.6, fontface
  = "plain", col = "black"),
footnote_parse = TRUE,
title_just = c("left", "right", "center"),
title_gp = gpar(cex = 1.2, fontface = "bold", col = "black", fontfamily = base_family),
arrow_type = c("open", "closed"),
arrow_label_just = c("start", "end"),
arrow_length = 0.05,
arrow_gp = gpar(fontsize = base_size, fontfamily = base_family, lwd = 0.6),
xlab_adjust = c("refline", "center"),
xlab_gp = gpar(fontsize = base_size, fontfamily = base_family, cex = 1, fontface =
  "plain"),
...
)

```

Arguments

<code>base_size</code>	The size of text
<code>base_family</code>	The font family
<code>ci_pch</code>	Shape of the point estimation. It will be reused if the forest plot is grouped.
<code>ci_col</code>	Color of the CI. A vector of color should be provided for the grouped forest plot. An internal color set will be if only not.
<code>ci_alpha</code>	Scalar value, alpha channel for transparency of point estimation. A small vertical line will be added to indicate the point estimation if this is not equals to 1.
<code>ci_fill</code>	Color fill the point estimation. A vector of color should be provided for the grouped forest plot. If this is NULL (default), the value will inherit from " <code>ci_col</code> ". This is valid only if <code>ci_pch</code> within 15:25.

ci_lty	Line type of the CI. A vector of line type should be provided for the grouped forest plot.
ci_lwd	Line width of the CI. A vector of line type should be provided for the grouped forest plot.
ci_Theight	A unit specifying the height of the T end of CI. If set to NULL (default), no T end will be drawn.
legend_name	Title of the legend.
legend_position	Position of the legend, "right", "top", "bottom" or "none" to suppress the legend.
legend_value	Legend labels (expressions). A vector should be provided for the grouped forest plot. A "Group 1" etc will be created if not a vector for a grouped forest plot.
legend_gp	gpar graphical parameters of legend, see gpar .
legend_ncol	integer; the number of columns, see legendGrob .
legend_byrow	logical indicating whether rows of the legend are filled first, see legendGrob .
xaxis_gp	gpar graphical parameters of x-axis, see gpar .
refline_gp	gpar graphical parameters of reference line, see gpar .
vertline_lwd	Line width for extra vertical line. A vector can be provided for each vertical line, and the values will be recycled if no enough values are given.
vertline_lty	Line type for extra vertical line. Works same as vertline_lwd.
vertline_col	Line color for the extra vertical line. Works same as vertline_lwd.
summary_col	Color for borders of the summary diamond shape.
summary_fill	Color for filling the summary diamond shape.
footnote_gp	gpar graphical parameters of footnote, see gpar .
footnote_parse	Parse footnote text (default).
title_just	The justification of the title, default is 'left'.
title_gp	gpar graphical parameters of title, see gpar .
arrow_type	Type of the arrow below x-axis, see arrow .
arrow_label_just	The justification of the arrow label relative to arrow. Control the arrow label to align to the starting point of the arrow "start" (default) or the ending point of the arrow "end".
arrow_length	The length of the arrow head, default is 0.05. See arrow .
arrow_gp	gpar graphical parameters of arrow, see gpar .
xlab_adjust	Control the alignment of xlab to reference line (default) or center of the x-axis.
xlab_gp	gpar graphical parameters of xlab, see gpar .
...	Other parameters passed to table. See tableGrob for details.

Value

A list.

See Also

[tableGrob](#) [forest](#) [textGrob](#) [gpar](#) [arrow](#) [segmentsGrob](#) [linesGrob](#) [pointsGrob](#) [legendGrob](#)

get_wh	<i>Get widths and height the forestplot</i>
--------	---

Description

get_wh can be used to find the correct width and height of the forestplot for saving, as the width and height is difficult to fit for saving.

Usage

```
get_wh(plot, unit = c("in", "cm", "mm"))
```

Arguments

plot	A forest plot object.
unit	Unit of the plot size in units ("in", "cm", or "mm")to be saved.

Value

A named vector of width and height

Examples

```
## Not run:
dt <- read.csv(system.file("extdata", "example_data.csv", package = "forestploter"))
dt <- dt[1:6,1:6]

dt$` ` <- paste(rep(" ", 20), collapse = " ")

p <- forest(dt[,c(1:3, 7)],
            est = dt$est,
            lower = dt$low,
            upper = dt$hi,
            ci_column = 4)

# get_wh example
p_wh <- get_wh(p)
pdf('test.pdf',width = p_wh[1], height = p_wh[2])
plot(p)
dev.off()

## End(Not run)
```

insert_text	<i>Insert text to forest plot</i>
-------------	-----------------------------------

Description

This function can be used to insert text to forest plot. Remember to adjust for the row number if you have added text before, including header. This is achieved by inserted new row(s) to the plot and will affect the row number. A text vector can be inserted to multiple columns or rows.

Usage

```
insert_text(
  plot,
  text,
  row = NULL,
  col = NULL,
  part = c("body", "header"),
  just = c("center", "left", "right"),
  before = TRUE,
  gp = gpar(),
  padding = unit(1, "mm"),
  parse = FALSE
)
```

Arguments

plot	A forest plot object.
text	A character or expression vector, see textGrob .
row	Row to insert the text, this will be ignored if the part is "header".
col	A numeric value or vector indicating the columns the text will be added. The text will span over the column if a vector is given.
part	Part to insert text, body (default) or header.
just	The justification of the text, "center" (default), "left" or "right".
before	Indicating the text will be inserted before or after the row.
gp	An object of class "gpar", this is the graphical parameter settings of the text. See gpar .
padding	Padding of the text, default is <code>unit(1, "mm")</code>
parse	Logical, behaviour for parsing text as plotmath, see plotmath

Value

A [gtable](#) object.

See Also

[gpar](#) [textGrob](#) [gtable_add_grob](#)

makeci

*Create confidence interval grob***Description**

Create confidence interval grob

Usage

```
makeci(
  est,
  lower,
  upper,
  pch,
  sizes = 1,
  gp = gpar(),
  t_height = NULL,
  xlim = c(0, 1),
  nudge_y = 0,
  name = NULL
)
```

Arguments

est	Point estimation. Can be a list for multiple columns and/or multiple groups. If the length of the list is larger than then length of ci_column, then the values reused for each column and considered as different groups.
lower	Lower bound of the confidence interval, same as est.
upper	Upper bound of the confidence interval, same as est.
pch	Numeric or character vector indicating what sort of plotting symbol to use. See pointsGrob .
sizes	Size of the point estimation box, can be a unit, vector or a list. Values will be used as it is, no transformation will be applied.
gp	Graphical parameters of gpar . Please refer to forest_theme for more details.
t_height	The height confidence interval line end vertices. If value is NULL (default), no vertices will be drawn.
xlim	Limits for the x axis as a vector of length 2, i.e. c(low, high). It will take the minimum and maximum of the lower and upper value if not provided. This will apply to all CI columns if provided, and will be calculated automatically for each column if not provided. This should be a list with the same length of ci_column if different xlim for different column is desired.
nudge_y	Horizontal adjustment to nudge groups by, must be within 0 to 1.
name	name of the grob.

Value

A gTree object

make_boxplot	Create horizontal boxplot grob
--------------	--------------------------------

Description

Create horizontal boxplot grob

Usage

```
make_boxplot(
  est,
  lower,
  upper,
  lowhinge,
  uphinge,
  hinge_height = 0.2,
  pch,
  sizes = 1,
  gp = gpar(),
  gp_box = gp,
  t_height = NULL,
  xlim = c(0, 1),
  nudge_y = 0
)
```

Arguments

est	Median value.
lower	Lower whisker.
upper	Upper whisker.
lowhinge	Lower hinge, a standard whisker will be drawn if this is missing.
uphinge	Upper hinge, a standard whisker will be drawn if this is missing.
hinge_height	Height of the hinge, default is 0.2.
pch	Numeric or character vector indicating what sort of plotting symbol to use. See pointsGrob .
sizes	Size of the point estimation box, can be a unit, vector or a list. Values will be used as it is, no transformation will be applied.
gp	Graphical parameters of gpar . Please refer to forest_theme for more details.
gp_box	Graphical parameters passed to the hinge, this will be passed to rectGrob . This does not support multiple groups.

t_height	Height of the whisker end vertices. If value is NULL (default), no vertices will be drawn.
xlim	Limits for the x axis as a vector of length 2, i.e. c(low, high).
nudge_y	Horizontal adjustment to nudge groups by, must be within 0 to 1.

Value

A gTree object

See Also

[pointsGrob](#) [gpar](#) [rectGrob](#) [linesGrob](#) [segmentsGrob](#)

Examples

```
library(grid)

# Function to calculate Box plot values
box_func <- function(x){
  iqr <- IQR(x)
  q3 <- quantile(x, probs = c(0.25, 0.5, 0.75), names = FALSE)
  c("min" = q3[1] - 1.5*iqr, "q1" = q3[1], "med" = q3[2],
    "q3" = q3[3], "max" = q3[3] + 1.5*iqr)
}

# Prepare data
val <- split(ToothGrowth$len, list(ToothGrowth$supp, ToothGrowth$dose))
val <- lapply(val, box_func)

dat <- do.call(rbind, val)
dat <- data.frame(Dose = row.names(dat),
                  dat, row.names = NULL)

dat$Box <- paste(rep(" ", 20), collapse = " ")

# Draw single group box plot
tm <- forest_theme(ci_Theight = 0.2)

p <- forest(dat[,c(1, 7)],
            est = dat$med,
            lower = dat$min,
            upper = dat$max,
            # sizes = sizes,
            fn_ci = make_boxplot,
            ci_column = 2,
            lowhinge = dat$q1,
            uphinge = dat$q3,
            hinge_height = 0.2,
            index_args = c("lowhinge", "uphinge"),
            gp_box = gpar(fill = "black", alpha = 0.4),
            theme = tm
          )
p
```

```

# Multiple group
# Prepare data
dat_oj <- dat[c(1, 3, 5),]
dat_vc <- dat[c(2, 4, 6), ]

dat <- data.frame(Dose = c(0.5, 1, 2))
dat$Box <- paste(rep(" ", 20), collapse = " ")

# Draw plot
tm <- forest_theme(ci_Theight = 0.2,
                   ci_pch = 3)

p <- forest(dat,
            est = list(dat_oj$med, dat_vc$med),
            lower = list(dat_oj$min, dat_vc$min),
            upper = list(dat_oj$max, dat_vc$max),
            fn_ci = make_boxplot,
            ci_column = 2,
            lowhinge = list(dat_oj$q1, dat_vc$q1),
            uphinge = list(dat_oj$q3, dat_vc$q3),
            hinge_height = 0.2,
            index_args = c("lowhinge", "uphinge"),
            theme = tm
          )
p

```

make_summary

Create pooled summary diamond shape

Description

Create pooled summary diamond shape

Usage

```
make_summary(est, lower, upper, sizes = 1, gp, xlim, nudge_y = NULL)
```

Arguments

est	Point estimation. Can be a list for multiple columns and/or multiple groups. If the length of the list is larger than then length of ci_column, then the values reused for each column and considered as different groups.
lower	Lower bound of the confidence interval, same as est.
upper	Upper bound of the confidence interval, same as est.

sizes	Size of the point estimation box, can be a unit, vector or a list. Values will be used as it is, no transformation will be applied.
gp	Graphical parameters of gpar . Please refer to forest_theme for more details.
xlim	Limits for the x axis as a vector of length 2, i.e. c(low, high). It will take the minimum and maximum of the lower and upper value if not provided. This will apply to all CI columns if provided, and will be calculated automatically for each column if not provided. This should be a list with the same length of ci_column if different xlim for different column is desired.
nudge_y	Horizontal adjustment to nudge groups by, must be within 0 to 1.

Value

A gTree object

<code>print.forestplot</code>	<i>Draw plot</i>
-------------------------------	------------------

Description

Print or draw forestplot.

Usage

```
## S3 method for class 'forestplot'
print(x, autofit = FALSE, ...)

## S3 method for class 'forestplot'
plot(x, autofit = FALSE, ...)
```

Arguments

x	forestplot to display
autofit	If true, the plot will be autofit.
...	other arguments not used by this method

Value

Invisibly returns the original forestplot.

Index

`add_border`, [2](#), [2](#)
`add_grob`, [3](#)
`add_text`, [4](#)
`add_underline`, [2](#)
`add_underline (add_border)`, [2](#)
`arrow`, [11](#), [12](#)

`edit_plot`, [5](#)
`editGrob`, [5](#)

`forest`, [6](#), [12](#)
`forest_theme`, [5](#), [6](#), [8](#), [9](#), [14](#), [15](#), [18](#)

`get_wh`, [12](#)
`gpar`, [3–5](#), [9](#), [11–16](#), [18](#)
`gtable`, [3–5](#), [8](#), [13](#)
`gtable_add_grob`, [3](#), [4](#), [13](#)

`insert_text`, [13](#)

`legendGrob`, [11](#), [12](#)
`linesGrob`, [12](#), [16](#)

`make_boxplot`, [8](#), [15](#)
`make_summary`, [8](#), [17](#)
`makeci`, [7](#), [8](#), [14](#)

`plot.forestplot (print.forestplot)`, [18](#)
`plotmath`, [4](#), [13](#)
`pointsGrob`, [12](#), [14–16](#)
`print.forestplot`, [18](#)

`rectGrob`, [5](#), [15](#), [16](#)

`segmentsGrob`, [2](#), [3](#), [12](#), [16](#)

`tableGrob`, [8](#), [11](#), [12](#)
`textGrob`, [4](#), [5](#), [12](#), [13](#)