Package ‘scplot’

February 25, 2026

Type Package

Title Plot Function for Single-Case Data Frames

Version 0.6.1

Date 2026-02-25

Description Add-on for the 'scan' package that creates plots from single-case data frames ('scdf'). It includes functions for styling single-case plots, adding phase-based lines to indicate various statistical parameters, and predefined themes for presentations and publications. More information and in depth examples can be found in the online book ``Analyzing Single-Case Data with R and 'scan'' Jürgen Wilbert (2026) <https://jazznbass.github.io/scan-Book/>.

Depends R (>= 4.1.0)

Imports ggplot2, scan (>= 0.61.0), stats, utils, mblm

Suggests rmarkdown, knitr

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Juergen Wilbert [aut, cre] (ORCID: <https://orcid.org/0000-0002-8392-2873>)

Maintainer Juergen Wilbert <juergen.wilbert@uni-muenster.de>

Repository CRAN

Date/Publication 2026-02-25 07:30:02 UTC

Contents

`scplot-package` *Single-Case Data Plots*

Description

A collection of procedures for visualizing single-case data. It is an add-on package for the `scan` package.

Author(s)

Juergen Wilbert [aut, cre]

add_arrow *Add arrrows to an scplot*

Description

Draws an arrow between two points in the scplot.

Usage

```
add_arrow(
  object,
  case = 1,
  x0,
  y0,
  x1,
  y1,
  color = "black",
  angle = 30,
  length = unit(5, "points"),
  type = "open",
  ends = "last",
  linewidth = 0.7
)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`case` Numerical vector with the case number or character string. `case = "all"` for all cases.

`x0` Origin x position of the line.

`y0` Origin y position of the line.

`x1` End x position of the line.

`y1` End y position of the line.

`color` A character string or a number defining the color of an element.

`angle` Angle (in [0,360])

`length` Size of the arrow angels.

`type` One of `"open"` or `"closed"` indicating whether the arrow head should be a closed triangle.

`ends` One of `"last"`, `"first"`, or `"both"`, indicating which ends of the line to draw arrow heads.

`linewidth` A number with the width of the line.

Details

The arrow is drawn from point `(x0, y0)` to point `(x1, y1)`. The arrow head can be customized with the arguments `angle`, `length`, `type`, and `ends`.

Value

An object of class `scplot` (see `scplot()`) with added element `lines`.

Examples

```
data(exampleAB, package = "scan")
p1 <- scplot(exampleAB$Anja)  |>
  add_arrow(case = 1, 2, 70, 6, 55, color = "darkred")
```

`add_grid` *Add grid to an scplot theme*

Description

Adds grid line specifications to the theme of an `scplot` object.

Usage

```
add_grid(object, ...)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`...` Line arguments (see `element_line()`).

Details

The function allows customization of grid lines by passing arguments similar to those used in `element_line()`, such as `color`, `size`, and `linetype`.

Value

An object of class `scplot` (see`scplot()`) with modified `theme` element to include the specified grid line properties.

See Also

`element_line()`

Examples

```
data(exampleAB, package = "scan")
p1 <- scplot(exampleAB$Anja)  |>
  set_theme("minimal")  |>
  add_grid(color = "grey70")
```

`add_labels` *Add value labels to an scplot*

Description

This function adds value labels to an existing `scplot` object. The labels display the values of a specified variable (default is `.dvar`, which typically represents the main variable being plotted). You can customize the position, rounding, text style, and background style of the labels.

Usage

```
add_labels(
  object,
  nudge_y = 5,
  nudge_x = 0,
  round = NULL,
  text = list(),
  background = list(),
  variable = ".dvar",
  padding = NULL
)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`nudge_y` Offset on the y-axis.

`nudge_x` Offset on the x-axis.

`round` Number of digits of the labels. If NULL, no rounding is applied.

`text` List with text parameters (`"family"`, `"face"`, `"colour"`, `"size"`, `"hjust"`, `"vjust"`, `"angle"`, `"l`
See `element_text()`.

`background` A list with background styling arguments (fill, color, size, linetype).

`variable` Name of the dataline variable to apply the style.

`padding` Padding size around text.

Value

An object of class `scplot` (see`scplot()`) with added/changed element `labels`.

add_legend *Add a legend to an scplot*

Description

This function adds a legend to an `scplot` object. The legend can include entries for datalines, statlines, and phases. You can customize the position of the legend, as well as the text style for the title and text within the legend.

Usage

```
add_legend(
  object,
  labels = NULL,
  section_labels = c("Lines", "Phases"),
  case = 1,
  position = "right",
  datalines = TRUE,
  statlines = TRUE,
  phases = TRUE,
  title = NULL,
  text = NULL,
  background = NULL
)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`labels` A Character vector with text labels.

`section_labels` A character vector of length two. The labels for the lines section and phase section.

`case` Numerical vector with the case number or character string. `case = "all"` for all cases.

`position` The position ("none", "left", "right", "bottom", "top", or two-element numeric vector) of the legend.

`datalines` If TRUE, a legend for the datalines is generated.

`statlines` If TRUE, a legend for the statlines is generated.

`phases` If TRUE, a legend for the phases is generated. Note that you also have to set the `set_panel` argument (e.g., `set_panel(fill = c("lightblue", "grey80"))`).

`title` A list with text style parameters for the title.

`text` List with text parameters (`"family"`, `"face"`, `"colour"`, `"size"`, `"hjust"`, `"vjust"`, `"angle"`, `"li`
See `element_text()`.

`background` A list with background styling arguments (fill, color, size, linetype).

Details

Deprecated argument: The labels argument is deprecated. Please set the label argument in the add_statline and set_dataline functions instead. The legend can be customized to include different sections for datalines, statlines, and phases. You can control which sections are included in the legend using the datalines, statlines, and phases arguments. The position argument allows you to specify where the legend should be placed on the plot. You can also customize the text style for the title and text within the legend using the title and text arguments, respectively. The background argument allows you to set the background style of the legend.

Value

An object of class scplot (seescplot()) with changed element legend.

Examples

```
data(exampleAB_add, package = "scan")
scplot(exampleAB_add) |>
  set_dataline("depression") |>
  add_statline("mean") |>
  add_legend()

scplot(exampleAB_add) |>
  set_dataline(label = "Pychological Wellbeing") |>
  set_dataline("depression", color = "darkblue", label = "Depression") |>
  add_statline("mean", label = "Wellbeing mean") |>
  add_statline("mean", variable = "depression", label = "Depression mean") |>
  set_phasenames(color = NA) |>
  set_panel(fill = c("lightblue", "grey80")) |>
  add_legend(
    position = "left",
    section_labels = c("Variables", "Section"),
    title = list(color = "brown", size = 10, face = 2),
    text = list(color = "darkgreen", size = 10, face = 2),
    background = list(color = "lightgrey")
  )
```

---

add_line *Add line to an scplot object*

Description

Draws a line on the scplot. Either a horizontal line (hline), vertical line (vline), or a line defined by its start (x0, y0) and end (x1, y1) points can be drawn.

Usage

```
add_line(
  object,
  case = 1,
  x0 = NULL,
  y0 = NULL,
  x1 = NULL,
  y1 = NULL,
  hline = NULL,
  vline = NULL,
  color = "black",
  linewidth = 0.7,
  linetype = "solid"
)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.
`case`	Numerical vector with the case number or character string. `case = "all"` for all cases.
`x0`	Origin x position of the line.
`y0`	Origin y position of the line.
`x1`	End x position of the line.
`y1`	End y position of the line.
`hline`	y position of horizontal line.
`vline`	x position of vertical line.
`color`	A character string or a number defining the color of an element.
`linewidth`	A number with the width of the line.
`linetype`	A character string with the line type: `"solid", "dashed", "dotted"`

Value

An object of class `scplot` (see`scplot()`) with added element `lines`.

Examples

```
data(exampleAB, package = "scan")
p1 <- scplot(exampleAB$Anja)  |>
  add_line(hline = 70, color = "darkred") |>
  add_line(vline = 3, color = "blue") |>
  add_line(x0 = 1, y0 = 70, x1 = 4, y1 = 80, color = "green")
```

add_marks *Add marks to an scplot object*

Description

Marks specific points in an scplot object.

Usage

```
add_marks(
  object,
  case = 1,
  positions,
  color = "red",
  size = 1,
  shape = 1,
  variable = ".dvar"
)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`case` Numerical vector with the case number or character string. `case = "all"` for all cases.

`positions` Either a vector indicating the points to be highlighted or a character string with a logical expression (e.g. `values < mean(values)`) indicating the points to be highlighted. Alternatively, an object of class `sc_outlier` returned from the `scan::outlier()` function can be used to mark the detected outliers. A list of such vectors or expressions can also be provided to add marks for multiple cases at once.

`color` A character string or a number defining the color of an element.

`size` Text size relative to the base text size.

`shape` Number. See pch graphical parameter on `par` help page `par()`.

`variable` Name of the dataline variable to apply the style.

Details

Marks are visualized as points on top of the existing plot. Multiple marks can be added by calling `add_marks()` multiple times.

If `positions` is an object returned from an outlier analysis via `outlier()`, the corresponding outliers are marked. If `positions` is a list, marks are added for each case in the list. The `variable` argument specifies the variable on which the marks are applied. By default, the variable `.dvar` is used. If multiple cases are plotted, the `case` argument specifies for which case(s) the marks are added.

Value

An object of class scplot (see scplot()) with changed element marks.

Author(s)

Juergen Wilbert

Examples

```
library(scan)
p1 <- scplot(exampleA1B1A2B2$Moritz) |> add_marks(positions = c(1,5,10,14))
p1 <- scplot(Huber2014) |> add_marks(positions = outlier(Huber2014))
```

add_ridge *Add a ridge to an scplot object*

Description

Adds a ridge element to an scplot object. Ridges are density plots displayed vertically along the y-axis, often used to show distributions of data points across different categories or groups.

Usage

```
add_ridge(object, color = "grey98", variable = ".dvar")
```

Arguments

object	An scplot object (class scplot) returned from the scplot() function.
color	A character string or a number defining the color of an element.
variable	Name of the dataline variable to apply the style.

Value

An object of class scplot (seescplot()) with changed element ridges.

add_statline *Add a statline to an scplot object*

Description

This function adds a statistical line or curve to an existing scplot object. Various statistical functions are available such as mean, median, min, max, quantile, standard deviation, moving average, and trend lines.

Usage

```
add_statline(
  object,
 stat = c("mean", "median", "min", "max", "quantile", "sd", "mad", "trend", "trendA",
    "trendA theil-sen", "trendA bisplit", "trendA trisplit", "moving mean",
    "moving median", "loreg", "lowess", "loess"),
  phase = NULL,
  color = NULL,
  linewidth = NULL,
  linetype = NULL,
  variable = NULL,
  label = NULL,
  ...
)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`stat` A character string for defining a statistical line or curve to be plotted. Possible values are `"mean"`, `"median"`, `"min"`, `"max"`, `"quantile"`, `"sd"`, `"mad"`, `"trend"`, `"trendA"`, `"trendA theil-sen"`, `"trendA bisplit"`, `"trendA trisplit"`, `"moving mean"`, `"moving median"`, `"loreg"` (local regression using `loess`), and `"lowess"` (local regression using `lowess`).

`phase` Either a numeric or a character vector specifying the reference phase (see details).

`color` A character string or a number defining the color of an element.

`linewidth` A number with the width of the line.

`linetype` A character string with the line type: `"solid", "dashed", "dotted"`

`variable` Name of the dataline variable to apply the style.

`label` A character string which is used to set the label in a legend.

`...` additional parameters passed to the statistical function.

Details

The phase argument defines the reference phase for some statistical functions ("median", "mean", "min", "max", "quant
The default is NULL which calculates and plots statistics for each phase separately. The arguments takes a numeric vector (phase number(s)) or a character vector (phase name(s)). When more than one phase is defines, statistics are based on the combined values of these phases. Various methods for an extrapolated *trendA* line exist: "trendA" is based on an OLS regression, "trendA theil-sen" on a nonparametric regression, and "trendA bisplit" / "trendA trisplit" are two median based approaches. Some of the functions defined in stats have additional arguments. The mean() function has a trim argument (e.g. trim = 0.1). quantile() has a proportion argument (e.g. prob = 0.75 for calculating the 75% quantile). moving mean and moving median have a lag argument (e.g. lag = 2). The local-regression curve function "lowess" (or "loreg") has a proportion argument (e.g. f = 0.5; see lowess()) and the local-regression curve function "loess" has a span argument (e.g. span = 0.75; see loess()).

Value

An object of class scplot (seescplot()) with changed element statlines.

add_text *Add test to an scplot object*

Description

Adds a text element to an scplot object at specified coordinates.

Usage

```
add_text(
  object,
  label,
  case = 1,
  x,
  y,
  color = "black",
  size = 1,
  angle = 0,
  hjust = 0.5,
  vjust = 0.5,
  face = 1
)
```

Arguments

object	An scplot object (class scplot) returned from the scplot() function.
label	A Character vector with text labels.
case	Numerical vector with the case number or character string. case = "all" for all cases.

`x`	x position.
`y`	y position.
`color`	A character string or a number defining the color of an element.
`size`	Text size relative to the base text size.
`angle`	Angle (in [0,360])
`hjust`	Horizontal justification (in [0,1])
`vjust`	Vertical justification (in [0,1])
`face`	Font face (`"plain", "italic", "bold", "bold.italic"`)

Value

An object of class `scplot` (see`scplot()`) with a changed `texts` element.

`add_title` *Add title and caption to an scplot object*

Description

Adds title and caption elements to an `scplot` object.

Usage

```
add_title(object, label, ...)

add_caption(object, label, header = "Note:\n", ...)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.
`label`	A Character vector with text labels.
`...`	List with text parameters (`"family", "face", "colour", "size", "hjust", "vjust", "angle", "l` See `element_text()`.
`header`	String with header above footnote/ caption

Details

The functions allow customization of title and caption text properties by passing arguments similar to those used in `element_text()`, such as `family`, `face`, `colour`, `size`, `hjust`, `vjust`, `angle`, `lineheight`, and `margin`.

Value

An object of class `scplot` (see`scplot()`) with changed `title` and `caption` elements.

`as_ggplot` *Creates a ggplot2 object from an* `scplot()` *object*

Description

`as_ggplot()` takes an `scplot` object as input and returns a `ggplot2` object.

Usage

```
as_ggplot(scplot)
```

Arguments

`scplot`	An `scplot` object

Details

`as_ggplot()` is used directly when you want to return a ggplot2 object for further use with external ggplot functions.

Value

A `ggplot2` plot object.

`new_theme` *Create a new scplot theme object*

Description

`new_theme()` creates a new theme object for use with scplot.

`extract_theme()` extracts the theme from an existing scplot object, allowing users to reuse or modify the theme for other scplot visualizations.

Usage

```
new_theme()

extract_theme(object)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.

Details

Themes control the overall appearance of scplot visualizations, including colors, fonts, line styles, and layout options. By creating a custom theme, users can ensure consistent styling across multiple plots and tailor the visualizations to their specific needs or branding requirements.

Value

An object of class `scplot-theme` which can be used with the `set_theme()` function.

Examples

```
data(exampleABC, package = "scan")
my_theme <- new_theme() |>
  set_panel(color = "red")  |>
  set_base_text(size = 12, color = "blue")  |>
  set_dataline(color = "darkred", linewidth = 2)
p1 <- scplot(exampleABC)  |> set_theme(my_theme)
```

---

scplot.scdf *Plot single-case data from a single-case data-frame*

Description

This function provides a plot of a single-case or multiple single-cases. It takes a single-case data-frame (scdf) as input and returns an object of class `scplot`, which can be further customized using various functions provided in the package.

Usage

```
## S3 method for class 'scdf'
scplot(object, ...)
```

Arguments

`object`	A single-case data-frame object (scdf).
`...`	further arguments.

Details

The function automatically extracts relevant information from the single-case data-frame, such as dependent variable, phase variable, and measurement time variable. It also sets default values for various plot elements, including data lines, statistical lines, ridges, marks, texts, and lines.

Value

An object of class `scplot` containing the single-case data (element `scdf`), and information about the plot style (element `theme`). This object can be further customized using various functions provided in the package.

Author(s)

Juergen Wilbert

See Also

`set_dataline()`, add_statline, `set_theme()`

Examples

```
data(exampleAB, package = "scan")
p1 <- scplot(exampleAB)
p2 <- scplot(exampleAB$Anja)
p3 <- scplot(exampleAB[c("Anja", "Berta")])
print(p1)
```

`scplot.sc_hplm` *Forest Plot for Random Effects of Mixed HPLM Models*

Description

This function generates a forest plot for the random effects of a mixed hplm model.

Usage

```
## S3 method for class 'sc_hplm'
scplot(object, effect = "intercept", mark = "fixed", ci = 0.95, ...)
```

Arguments

`object`	The return from the `hplm()` function.
`effect`	The specific effect to be plotted (default is the intercept).
`mark`	Set a reference line.
`ci`	Value between 0 and 1 for calculating the confidence interval.
`...`	Further arguments.

Value

A forest plot displaying Tau-U effects.

Examples

```
model <- scan::hplm(scan::Leidig2018, random.slopes = TRUE)
scplot(model, effect = "level")
```

scplot.sc_rand *Plot Randomization Effects*

Description

This function generates plots of randomization test results.

Usage

```
## S3 method for class 'sc_rand'
scplot(object, type = "hist", add_density_curve = TRUE, ...)
```

Arguments

object	The return from the tau_u() function.
type	Either "hist" or "xy".
add_density_curve	If TRUE, adds a density curve to the histogram.
...	Further arguments.

Value

A plot displaying the results of the randomization test.

Examples

```
## Not run:
res <- scan::rand_test(scan::exampleAB$Anja, limit = 1)
scplot(res, type = "hist")

scplot(res, type = "xy")

## End(Not run)
```

scplot.sc_tauu *Plot Tau-U Effects Forest Plot*

Description

This function generates a forest plot of Tau-U effects.

Usage

```
## S3 method for class 'sc_tauu'
scplot(object, effect = 1, ...)
```

Arguments

`object`	The return from the `tau_u()` function.
`effect`	The specific effect to be plotted (default is "A vs. B - Trend A").
`...`	Further arguments.

Value

A forest plot displaying Tau-U effects.

Examples

```
res <- scan::tau_u(scan::Leidig2018)
scplot(res, effect = 3)
```

`scplot_rand` *Random start position plot Plot of statistics for random phase B start positions*

Description

Random start position plot Plot of statistics for random phase B start positions

Usage

```
scplot_rand(
  scdf,
  statistic = "Mean B-A",
  x_label = "Start phase B",
  color_label = "Compared to\nobserved",
 colors = c(Above = "coral3", Below = "aquamarine4", Observed = "#56B4E9", Equal =
    "black"),
  ...
)
```

Arguments

`scdf`	A single-case data frame object.
`statistic`	A string with a the name of a statistic. Defaults to `Mean B-A`. See rand_test() function for all options.
`x_label`	Character string with the x label.
`color_label`	Character string with the color label.
`colors`	Named vector with color codes.
`...`	further arguments passted to the scan `rand_test()` function.

Examples

```
scplot_rand(scan::byHeart2011[1:5])
```

---

set_background *Set plot and panel background of an scplot object*

Description

The set_background() function allows you to customize the plot background of an scplot object by specifying parameters such as fill color, border color, line width, and line type. Similarly, the set_panel() function enables you to set the panel background with the same customizable parameters. These functions enhance the visual appearance of your plots by allowing you to tailor the background styles to your preferences.

Usage

```
set_background(object, ...)

set_panel(object, ...)
```

Arguments

object	An scplot object (class scplot) returned from the scplot() function.
...	List with rectangle parameters ("fill", "colour", "linewidth", "linetype"). See element_rect().

Value

An object of class scplot (seescplot()).

Examples

```
data(exampleAB, package = "scan")
p1 <- scplot(exampleAB)  |>
 set_background(fill = "lightblue", colour = "darkblue", linewidth = 1.5) |>
 set_panel(fill = "deepskyblue", color = "darkblue", linewidth = 0.3)
```

set_base_text *Set base text parameters of an scplot object*

Description

Sets the base text parameters for all text elements in an scplot object.

Usage

```
set_base_text(object, ...)
```

Arguments

`object` An scplot object (class scplot) returned from the scplot() function.

`...` List with text parameters ("family", "face", "colour", "size", "hjust", "vjust", "angle", "l
See element_text().

Details

This function modifies the text element in the theme list of the scplot object, allowing you to define default text properties such as font family, face, color, size, alignment, angle, line height, and margin using parameters similar to those in element_text().

Value

An object of class scplot (seescplot()).

set_casenames *Set casenames of an scplot object*

Description

Sets the case names (labels) and their position in an scplot object.

Usage

```
set_casenames(object, labels = NULL, position = NULL, background = list(), ...)
```

Arguments

`object` An scplot object (class scplot) returned from the scplot() function.

`labels` A Character vector with text labels.

`position` Either "topleft", "bottomleft", "topright", "bottomright", "strip-right", "strip-top", or a numerical vector of length 2 with the x and y position (e.g. c(19, 20)).

`background` A list with background styling arguments (fill, color, size, linetype).

`...` List with text parameters ("family", "face", "colour", "size", "hjust", "vjust", "angle", "l
See element_text().

Details

If labels is NULL, the current labels are kept. If position is NULL, the current position is kept.

Value

An object of class scplot (seescplot()) with a changed casenames element.

set_dataline *Set data lines of an scplot object*

Description

Either set aesthetics of the default data line or add another data line. The function allows customization of data lines by passing arguments similar to those used in element_line() and element_point(), such as color, size, and linetype.

Usage

```
set_dataline(
  object,
  variable = NULL,
  line,
  point,
  type = "continuous",
  label = NULL,
  ...
)

add_dataline(...)
```

Arguments

object An scplot object (class scplot) returned from the scplot() function.

variable String. The name of a new variable for adding a new line. If left empty, the aesthetics of the default data line are changed.

line List with line parameters ("colour", "linewidth", "linetype", "lineend", "arrow"). See element_line().

point A list with point parameters ("colour", "size", "shape"). See element_point().

type Either "continuous" or "discrete". Specifies how the data line should be treated.

label A character string which is used to set the label in a legend.

... As a shortcut, arguments passed hear are bundled as line arguments (see element_line()).

Details

If variable is left empty or set to ".dvar", the aesthetics of the default data line are changed. Otherwise, a new data line is added for the specified variable.

Value

An object of class `scplot` (see`scplot()`) with a changed `datalines` element.

See Also

`element_line()`, `element_point()`

Examples

```
data(exampleAB_add, package = "scan")
scplot(exampleAB_add)  |>
  set_dataline("depression", color = "darkblue")
```

`set_phasenames` *Set phasenames of an scplot object*

Description

Set phasenames of an scplot object

Usage

```
set_phasenames(object, labels = NULL, position = NULL, ...)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`labels` A Character vector with text labels.

`position` Character string either `'left'`, `'center'`, or `'none'`.

`...` List with text parameters (`"family"`, `"face"`, `"colour"`, `"size"`, `"hjust"`, `"vjust"`, `"angle"`, `"li`
See `element_text()`.

Details

This function allows to set or modify the phase names displayed above the phases in an `scplot` object. You can customize the labels and their position (left, center, or none). Additionally, you can adjust various text parameters such as font family, face, color, size, and more using the `...` argument.

Value

An object of class `scplot` (see `scplot()`) with a changed phasenames element.

`set_separator` *Set separator line in an scplot object*

Description

This function allows you to customize the appearance of separator lines in an `scplot` object by specifying parameters such as color, line width, and line type. You can pass these parameters as arguments, similar to those used in `element_line()`, to modify the visual style of the separator lines in your plot theme.

Usage

```
set_separator(object, ...)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.
`...`	List with line parameters (`"colour""`, `"linewidth"`, `"linetype"`). See `element_line()`.

Value

An object of class `scplot` (see`scplot()`) with modified `theme` element.

`set_theme` *Add a theme of to an scplot object*

Description

Adds a predefined theme or custom theme to an `scplot` object.

Usage

```
set_theme(object, theme, ...)

add_theme(...)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.
`theme`	A character string with a predefined graphical theme or a theme object created with `new_theme()`.
`...`	Further character strings or `scplot-theme` objects that are "added" on top.

Details

Possible themes are: `'basic'`, `'grid'`, `'default'`, `'small'`, `'tiny'`, `'big'`, `'minimal'`, `'dark'`, `'sienna'`, `'phas`
See `new_theme()` for details on the themes.

Value

An object of class `scplot` (see`scplot()`) with a changed `theme` element.

`set_theme_element` *Set a theme element of an scplot object*

Description

Low-level function. Sets specific theme elements of an `scplot` object.

Usage

```
set_theme_element(object, ...)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.
`...`	various style parameters for specific theme elements (see Details).

Details

Allows customization of individual theme elements by passing them as named arguments.

Usually, you don't need this function. Possible theme elements are: "text", "plot.background", "panel.background", "panel.spacing.y", "dataline", "datapoint", "statline", "axis.expand.x", "axis.expand.y", "axis.line.x", "axis.line.y", "axis.ticks.length", "axis.ticks", "axis.title.y", "axis.title.x", "axis.text.x", "axis.text.y", "plot.title", "plot.caption", "plot.margin", "casenames", "casenames.strip", "casenames.background", "casenames.position", "phasenames", "phasenames.position.x", "separators", "separators.extent", "label.text", "label.background", "label.padding", "grid", "legend.position", "legend.background", "legend.text", "legend.title", "legend.margin".

The elements are of the following classes:

- text = c("element_text", "element"),
- plot.background = c("element_rect", "element"),
- panel.spacing.y = c("simpleUnit", "unit", "unit_v2"),
- dataline = "list",
- datapoint = "list",
- statline = c("element_line", "element"),
- axis.expand.x = "numeric",
- axis.expand.y = "numeric",

- axis.line.x = c("element_line", "element"),
- axis.line.y = c("element_line", "element"),
- axis.ticks.length = c("simpleUnit", "unit", "unit_v2"),
- axis.ticks = c("element_line", "element"),
- axis.title.y = c("element_text", "element"),
- axis.title.x = c("element_text", "element"),
- axis.text.x = c("element_text", "element"),
- axis.text.y = c("element_text", "element"),
- plot.title = c("element_text", "element"),
- plot.caption = c("element_text", "element"),
- plot.margin = c("margin", "simpleUnit", "unit","unit_v2"),
- casenames = c("element_text", "element"),
- casenames.strip = c("element_rect", "element"),
- casenames.background = c("element_rect", "element"),
- casenames.position = "character",
- phasenames = c("element_text", "element"),
- phasenames.position.x = "character",
- separators = c("element_line", "element"),
- separators.extent = "character",
- label.text = c("element_text", "element"),
- label.background = c("element_rect", "element"),
- label.padding = "numeric", grid = c("element_line", "element"),
- legend.position = "character",
- legend.background = c("element_rect", "element"),
- legend.text = c("element_text", "element"),
- legend.title = c("element_text", "element"),
- legend.margin = c("margin", "simpleUnit", "unit", "unit_v2")

Value

An object of class `scplot` (see`scplot()`) with a changed `theme` element. If no arguments are provided, the possible theme elements are printed.

Examples

```
data(exampleABC, package = "scan")
p1 <- scplot(exampleABC)  |>
  set_theme_element(
    axis.ticks.length = unit(0, "points"),
    axis.line.y = element_line(color = "darkred", linewidth = 2),
    panel.background = element_rect(color = "darkblue", linewidth = 1),
    panel.spacing.y = unit(0, "points"),
    phasenames = element_text(color = "#00000000")
  )
```

set_xaxis *Set axis parameters of an scplot*

Description

Sets limits, increments, line and text parameters of x- or y-axis of an scplot object.

Usage

```
set_xaxis(
  object,
  limits = NULL,
  increment = NULL,
  increment_from = NULL,
  line = NULL,
  expand = NULL,
  ...
)

set_yaxis(
  object,
  limits = NULL,
  increment = NULL,
  increment_from = NULL,
  line = NULL,
  expand = NULL,
  ...
)
```

Arguments

`object` An scplot object (class `scplot`) returned from the `scplot()` function.

`limits` Lower and upper limits of the axis (e.g., `limits = c(0, 20)` sets the axis to a scale from 0 to 20). With multiple single-cases you can use `limits = c(0, NA)` to scale the axis from 0 to the maximum of each case. `limits` is not set by default, which makes `scplot` set a proper scale based on the given data.

`increment` An integer. Increment of the x-axis. 1 :each mt value will be printed, 2 : every other value, 3 : every third values etc. If NULL, `scplot` chooses an appropriate increment based on the data.

`increment_from` Number from which increment starts to count. Usually set to 0 if you want marks like 1,5,10,15,... on the axis. If NULL, `scplot` chooses an appropriate value based on the data.

`line` List with line parameters (`"colour", "linewidth", "linetype", "lineend", "arrow"`). See `element_line()`.

`expand`	Vector with two values. Expansion of the axis. First value expands the lower end of the axis, second value the upper end. Default is `c(0,0)`, meaning no expansion. You can also use relative values, e.g. `c(0.05, 0.05)` to expand both ends by 5 percent of the total axis length. See `ggplot2::scale_x_continuous()` for details.
`...`	Further styling arguments: color, size, face, family, hjust, vjust, lineheight, angle, linetype, lineend, arrow, fill, margin.

Value

An object of class `scplot` (see`scplot()`) with changed `xaxis` and `yaxis` elements. Also, the `theme` element is changed to reflect line and text parameters.

`set_xlabel` *Set label for axis*

Description

Set label for axis

Usage

```
set_xlabel(object, label = NULL, ...)

set_ylabel(object, label = NULL, ...)
```

Arguments

`object`	An scplot object (class `scplot`) returned from the `scplot()` function.
`label`	A Character vector with text labels.
`...`	Further styling arguments: color, size, face, family, hjust, vjust, lineheight, angle, linetype, lineend, arrow, fill, margin.

Value

An object of class `scplot` (see`scplot()`) with a changed `xlabel` or `ylabel` element.

Index