Package: utile.visuals (via r-universe)

September 10, 2024

Title Create Visuals for Publication

Version 0.3.3

Description A set of functions to aid in the production of visuals in ggplot2.

License LGPL (>= 2)

URL https://efinite.github.io/utile.visuals/

BugReports https://github.com/efinite/utile.visuals/issues

Encoding UTF-8

Depends R (>= 3.4.0)

Imports ggplot2 (>= 0.3.4), gridExtra, purrr, vctrs

Suggests survival, grid

RoxygenNote 7.2.3

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

RemoteUrl https://github.com/efinite/utile.visuals

RemoteRef HEAD

RemoteSha d9cd17d286f3e1659cbfdc3055c6465a878f1718

Contents

append_table *Append a ggplot2 table to the bottom of a ggplot2 plot*

Description

Aligns axes and combines a ggplot2 plot and table into a single plot. Can handle legends.

Usage

```
append_table(
  plot = NULL,
  table = NULL,
  plot.height = 1,
  table.height = 0.1,
  plot.width = 1,
  legend.width = 0.2,
  legend.offset = -15
)
```

Arguments

plot	A ggplot2::ggplot() object. If a legend is present, it will be extracted.
table	A ggplot2::ggplot object. If a legend is present, it will be removed and ignored.
plot.height	A numeric. Height of plot relative to table. Defaults to 1.
table.height	A numeric. Height of table relative to plot. Defaults to 0.1.
plot.width	A numeric. Width of plot relative to legend. Ignored if no legend present in plot. Defaults to 1.
legend.width	A numeric. Width of legend relative to plot. Ignored if no legend present in plot. Defaults 0.2.
legend.offset	A numeric. Vertical offset of legend box. Used to raise or lower. Ignored if no legend present in plot. Defaults to -15.

Value

A ggplot2 tableGrob object. Use grid::grid.draw() to open in RStudio viewer. Works with ggplot2::ggsave() out of the box.

Note

To ensure proper alignment, double check that both plots use the same scale and breaks!

Examples

```
library(survival)
library(ggplot2)
library(grid) # grid.draw() finished plot

# Data with group names specified
data_diabetic <- diabetic
data_diabetic$trt <- as.factor(data_diabetic$trt)
levels(data_diabetic$trt) <- c('None', 'Laser')

# Survival Model
fit <- survfit(Surv(time, status) ~ trt, data = data_diabetic)
fit <- survfit0(fit)

# Kaplan Meier (KM) Plot
plot_km <- ggplot(
 data = data.frame(
  time = fit$time,
  surv = fit$surv,
  conf.low = fit$lower,
  conf.high = fit$upper,
  strata = rep(names(fit$strata), fit$strata)
 ),
 mapping = aes(x = time, y = surv)
) +
  geom_step(aes(color = strata)) +
  geom_stepconfint(aes(ymin = conf.low, ymax = conf.high, fill = strata), alpha = 0.3) +
  coord_cartesian(c(0, 50)) + # Note scale set here!
  scale_x_continuous(expand = c(0.02,0)) +
  labs(x = 'Time', y = 'Freedom From Event') +
  scale_color_manual(
    values = c('#d83641', '#1A45A7'),
    name = 'Treatment',
    labels = c('Laser', 'None'),
    aesthetics = c('colour', 'fill')) +
  theme_basic()

# Risk Table
tbl_risk <- ggrisktable(fit, c(0, 10, 20, 30, 40, 50)) +
  coord_cartesian(c(0, 50)) +
  scale_x_continuous(expand = c(0.02,0)) +
  theme_risk()

# Combine KM plot and risk table
plot_cmbd <- append_table(
  plot = plot_km,
  table = tbl_risk
)

# Draw in RStudio viewer
grid.newpage()
grid.draw(plot_cmbd)
```

`geom_stepconfint` *Step function confidence intervals for ggplot2*

Description

Produces a step function confidence interval for survival curves.

Usage

```
geom_stepconfint(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  ...
)
```

Arguments

`mapping`	Aesthetic mappings with aes() function. Like geom_ribbon(), you must provide columns for x, ymin (lower limit), ymax (upper limit).
`data`	The data to be displayed in this layer. Can inherit from ggplot parent.
`stat`	The statistical transformation to use on the data for this layer, as a string. Defaults to 'identity'.
`position`	Position adjustment, either as a string, or the result of a call to a position adjustment function.
`na.rm`	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
`...`	Optional. Any other ggplot geom_ribbon() arguments.

Note

Adapted from the survminer package <https://github.com/kassambara/survminer>.

Examples

```
library(survival)
library(ggplot2)

fit <- survfit(Surv(time, status) ~ trt, data = diabetic)
fit <- survfit0(fit) # connect origin

ggplot(
  data = data.frame(
    time = fit$time,
    surv = fit$surv,
```

```
    conf.low = fit$lower,
    conf.high = fit$upper,
    strata = rep(names(fit$strata), fit$strata)
  ),
  mapping = aes(x = time, y = surv)
) +
  geom_step(aes(color = strata)) +
  geom_stepconfint(aes(ymin = conf.low, ymax = conf.high, fill = strata), alpha = 0.3) +
  coord_cartesian(c(0, 50)) +
  scale_x_continuous(expand = c(0.02,0)) +
  labs(x = 'Time', y = 'Freedom From Event') +
  scale_color_manual(
    values = c('#d83641', '#1A45A7'),
    name = 'Treatment',
    labels = c('None', 'Laser'),
    aesthetics = c('colour', 'fill')) +
  theme_basic()
```

`ggrisktable` *Create a ggplot2 table showing the number at risk*

Description

A simple wrapper function which calculates the numbers at risk for a survival model and a given set of time points then creates a ggplot2 table with them.

Usage

```
ggrisktable(
  fit = NULL,
  times = NULL,
  text.color = "black",
  strata.order = NULL
)
```

Arguments

`fit`	Required. survival::survfit() object.
`times`	Required. Numeric. One or more time points to calculate the number at risk for.
`text.color`	Optional. Character. Color of text within table. Defaults to 'black'.
`strata.order`	Optional. Character. Ordered names of strata factor levels.

Value

An unformatted ggplot2 table showing the number at risk.

Examples

```
library(survival)

fit <- survfit(Surv(time, status) ~ trt, data = diabetic)

ggrisktable(
   fit = fit,
   times = c(0, 10, 20, 30, 40, 50),
   strata.order = c('0', '1')
) + theme_risk()
```

---

panel_border *Add a panel border to a ggplot2 plot*

Description

A simple ggplot2 theme which replaces the axis lines with a bordered panel.

Usage

```
panel_border(base_size = 12, base_color = NULL)
```

Arguments

base_size	A numeric. Base size. Used to calculate line size and spacing.
base_color	A character. Base color for lines.

Note

This should be placed after the primary theme for the plot.

Examples

```
library(ggplot2)

ggplot(datasets::mtcars, aes(x = wt, y = hp, color = as.factor(cyl))) +
  geom_point() +
  facet_wrap(~as.logical(am)) +
  theme_basic() +
  panel_border()
```

theme_basic *Minimalist theme for ggplot2*

Description

A minimalist ggplot2 theme which removes most background elements and lines.

Usage

```
theme_basic(
  base_size = 12,
  base_family = NULL,
  base_color = "black",
  base_line_size = base_size/12,
  base_rect_size = base_size/12
)
```

Arguments

base_size	A numeric. Base font size.
base_family	A numeric. Base font family.
base_color	A character. Base color for lines and text.
base_line_size	A numeric. Base line element size.
base_rect_size	A numeric. Base rectangle element size.

Note

Recommend exporting as PNG or TIFF to preserve background transparency.

Examples

```
library(ggplot2)

ggplot(datasets::mtcars, aes(x = wt, y = hp, color = as.factor(cyl))) +
  geom_point() +
  theme_basic()
```

theme_risk *Minimalist risk table theme for ggplot2*

Description

A minimalist ggplot2 theme which removes most background elements and lines.

Usage

```
theme_risk(
  base_size = 12,
  base_family = NULL,
  base_color = "black",
  base_line_size = base_size/12,
  base_rect_size = base_size/12
)
```

Arguments

base_size	A numeric. Base font size.
base_family	A numeric. Base font family.
base_color	A character. Base color for lines and text.
base_line_size	A numeric. Base line element size.
base_rect_size	A numeric. Base rectangle element size.

Note

Recommend exporting as PNG or TIFF to preserve background transparency.

See Also

ggrisktable

Index