Package ‘multipanelfigure’

July 23, 2025

Type Package

Title Infrastructure to Assemble Multi-Panel Figures (from Grobs)

Version 2.1.6

Date 2024-04-09

Author Johannes Graumann [cre, aut],
Richard Cotton [aut]

Maintainer Johannes Graumann `<johannes.graumann@uni-marburg.de>`

Description Tools to create a layout for figures made of multiple panels, and to fill the panels with base, 'lattice', 'ggplot2' and 'ComplexHeatmap' plots, grobs, as well as content from all image formats supported by 'ImageMagick' (accessed through 'magick').

Imports ggplot2 (>= 2.2.1), grid, gridGraphics (>= 0.3-0), gtable (>= 0.2.0), magick (>= 1.9), magrittr (>= 1.5), methods, stringi (>= 1.2.3), utils

Suggests ComplexHeatmap (>= 1.17.1), grDevices, lattice (>= 0.20-35), roxygen2 (>= 6.0.1), VennDiagram (>= 1.6.20), knitr, rmarkdown, markdown

License GPL (>= 3)

RoxygenNote 7.2.3

LazyData TRUE

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2024-04-09 15:40:02 UTC

Contents

`assert_is_multipanelfigure` *Check that the input is a multipanelfigure*

Description

Checks that the input is of class `multipanelfigure` and has the appropriate attributes.

Usage

```
assert_is_multipanelfigure(x)
```

Arguments

`x`	Object to check.

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

`billing2016_suppfig4e` *Mass spectrometry intensities by stem cell type and organelle*

Description

This data was used to create Supplementary Figure 4e of Billing 2016 (see references).

Format

A data frame with 81 rows and the following columns:

GeneName A factor with three levels naming genes that have interesting properties.

Intensity A numeric vector of positive intensities of proteins corresponding to the genes as determined by mass spectrometry.

StemCellType A factor with three levels indicating the type of stem cell experimented on. "ESC" means embryonic stem cell; "ESC-MSC" means mesenchymal stem cell derived from an embryonic stem cell; "BM-MSC" means mesenchymal stem cell derived from bone marrow.

Organelle The region of the cell experimented on. "CH" means chromatin, "Cyt" means cytosol, "Nuc" means nucleus.

Replicate An integer specifying the experimental replicate.

Experiment The interaction of StemCellType, Organelle and Replicate.

Details

A data frame of genes corresponding to protein intensities as measured by mass spectrometry proteomics experiments on embryonic and mesenchymal stem cells.

References

Billing AM, Ben Hamidane H, Dib SS, et al. Comprehensive transcriptomic and proteomic characterization of human mesenchymal stem cells reveals source specific cellular markers. Scientific Reports. 2016;6:21507. doi:10.1038/srep21507.

Article text available at: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4746666`

Supplementary figures available at: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4746666/bin/srep21507-s1.pdf`

See Also

`billing2016_suppfig4g`

Examples

```
ggplot2::ggplot(billing2016_suppfig4e, ggplot2::aes(Experiment, Intensity)) +
  ggplot2::geom_bar(stat = "identity") +
  ggplot2::geom_vline(xintercept = seq(3.5, 24.5, 3), linetype = "dotted") +
  ggplot2::facet_wrap(~ GeneName) +
  ggplot2::xlab(NULL) +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 45, hjust = 1, size = 4))
```

`billing2016_suppfig4g` *Mass spectrometry intensities by stem cell type*

Description

This data was used to create Supplementary Figure 4g of Billing 2016 (see references).

Format

A matrix with 13 rows and 9 columns. Rows represent genes, columns represent experiments and are split by:

1. The type of stem cell experimented on. "ESC" means embryonic stem cell; "ESC-MSC" means mesenchymal stem cell derived from an embryonic stem cell; "BM-MSC" means mesenchymal stem cell derived from bone marrow.
2. The experimental replicate.

Values in the matrix are intensities of proteins coresponding to the genes, as measured by mass spectrometry.

Details

A matrix of log base 10 protein intensities as measured by mass spectrometry proteomics experiments on embryonic and mesenchymal stem cells.

References

Billing AM, Ben Hamidane H, Dib SS, et al. Comprehensive transcriptomic and proteomic characterization of human mesenchymal stem cells reveals source specific cellular markers. Scientific Reports. 2016;6:21507. doi:10.1038/srep21507.

Article text available at: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4746666`

Supplementary figures available at: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4746666/bin/srep21507-s1.pdf`

See Also

`billing2016_suppfig4e`

Examples

```
color_scale <- grDevices::cm.colors(25)
heatmap(
  billing2016_suppfig4g,
  margins = c(12, 5), col = color_scale,
  cexRow = 0.5, cexCol = 0.4)
```

`capture_base_plot` *Capture a base plot*

Description

Capture a plot drawn using base graphics as a grid `grob`.

Usage

```
capture_base_plot(expr)
```

Arguments

`expr`	A expression that draws a plot using base graphics.

Value

An object of class `gTree`.

Note

A side effect of this function is that plots get drawn twice: once as a base plot, and secondly as a grid plot.

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

See Also

`grid.echo`, `grid.grab`

Examples

```
p <- capture_base_plot(hist(rnorm(1000), seq(-4, 4, 0.2)))
grid::grid.draw(p)
# If the plot takes multiple lines to draw, then wrap the code in braces.
p2 <- capture_base_plot({
  par(las = 1)
  plot(1:5)
  title("One to five")
})
grid::grid.draw(p2)
```

`fill_panel` *fill_panel*

Description

A convenience function adding graphical objects to a `gtable` constructed by `multi_panel_figure`.

Usage

```
fill_panel(
  figure,
  panel,
  row = "auto",
  column = "auto",
  label = NULL,
  label_just = c("right", "bottom"),
  panel_clip = c("on", "off", "inherit"),
  scaling = c("none", "stretch", "fit", "shrink"),
  allow_panel_overwriting = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`figure`	Object of classes `multipanelfigure`/ `gtable` as produced by `multi_panel_figure` and representing the figure the panel is to be placed in.
`panel`	Single `character` object representing URL or path to a bitmap image accessible by `ImageMagick` as used through `magick`, a `Heatmap` or `HeatmapList` object, a `ggplot` object , a `trellis.object`, a `gList` object or a `grob` object to be placed in a multipanel figure. See 'Details'.
`row`	`numeric` object of length 1 or a range, indicating the row indeces the panel that is to be placed in the figure, or "auto" to automatically pick the row (see details). May be used to define panel spanning (if `length(row)` > `1`; see examples).
`column`	`numeric` object of length 1 or a range, indicating the column indeces of the panel that is to be placed in the figure, or "auto" to automatically pick the column (see details). May be used to define panel spanning (if `length(column)` > `1`; see examples).
`label`	Single `character` object defining the panel label used for automated annotation.
`label_just`	Justification for the label within the interpanel spacing grob to the top-left of the panel content grob. Passed to `textGrob`.
`panel_clip`	Should the display of panel contents be clipped at the panel borders? See `viewport`.

`scaling`	Only used when importing image files. Either "none" to preserve the dimensions of an image, "stretch" to make it fit the panels, "fit" to shrink or enlarge it so that it fills one dimension of the panels while preserving the height to width ratio, or "shrink which does the same but won't enlarge it.
`allow_panel_overwriting`	A logical value. If `TRUE`, overwriting panels is allowed, with a warning. Otherwise (the default) it will cause an error.
`verbose`	A logical value. Reduces verbosity if `FALSE`.
`...`	Additional arguments. Used to deal with deprecated arguments `top_panel`, `bottom_panel`, `left_panel` and `right_panel`.

Details

Currently supported as panel-representing objects (`panel`) are

1. ComplexHeatmap `Heatmap` or `HeatmapList` objects.
2. ggplot2 `ggplot` objects.
3. grid `grob`, `gList`, and `gTree` objects.
4. lattice `trellis.objects`.
5. Single `character` objects representing URLs or paths to image formats accessible by `ImageMagick` as used through `magick`, which will be read and placed into panels as requested.

Native resolution is determined from attributes in the file. If the attributes are not present, then the DPI is determined by the the `multipanelfigure.defaultdpi` global option, or 300 if this has not been set.

lattice-generated `trellis.objects` are converted to `grobs` using `grid.grabExpr(print(x))`, as are `Heatmap` and `HeatmapLists` from **ComplexHeatmap** - the side effects of which with respect to plot formatting are not well studied.

If the `row` argument is "auto", then the first row with a free panel is used. If the `column` argument is "auto", then the first column in the row with a free panel is used.

Value

Returns the `gtable` object fed to it (`figure`) with the addition of the `panel`.

Author(s)

Johannes Graumann, Richard Cotton

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

See Also

`gtable`, `multi_panel_figure`

Examples

```
 # Not testing - slow grid graphics makes CRAN timing excessive
# Create the figure layout
(figure <- multi_panel_figure(
  width = c(30,40,60),
  height = c(40,60,60,60),
  panel_label_type = "upper-roman"))

# Fill the top-left panel using a grob object directly
a_grob <- grid::linesGrob(arrow = grid::arrow())
figure %<>% fill_panel(a_grob)

# Add a ggplot object directly to the top row, second column.
# The panels are chosen automatically, but you can achieve the same effect
# using column = 2
a_ggplot <- ggplot2::ggplot(mtcars, ggplot2::aes(disp, mpg)) +
  ggplot2::geom_point()
figure %<>% fill_panel(a_ggplot)

# Bitmap images are added by passing the path to their file.
image_files <- system.file("extdata", package = "multipanelfigure") %>%
  dir(full.names = TRUE) %>%
  magrittr::set_names(basename(.))

# Add the JPEG to the top row, third column
figure %<>% fill_panel(image_files["rhino.jpg"], column = 3)

# Add the PNG to the second and third row, first and second column
figure %<>% fill_panel(
  image_files["Rlogo.png"],
  row = 2:3, column = 1:2)

# Add the TIFF to the second row, third column
figure %<>% fill_panel(
  image_files["unicorn.svg"],
  row = 2, column = 3)

# lattice/trellis plot objects are also added directly
Depth <- lattice::equal.count(quakes$depth, number=4, overlap=0.1)
a_lattice_plot <- lattice::xyplot(lat ~ long | Depth, data = quakes)
# Add the lattice plot to the third row, third column
figure %<>% fill_panel(
  a_lattice_plot,
  row = 3, column = 3)

# Incorporate a gList object (such as produced by VennDigram)
if(requireNamespace("VennDiagram"))
{
  a_venn_plot <- VennDiagram::draw.pairwise.venn(50, 30, 20, ind = FALSE)
  # Add the Venn diagram to the fourth row, firstd column
  figure %<>% fill_panel(
    a_venn_plot,
```

```
    row = 4, column = 1)
}

# Incorporate a base plot figure
a_base_plot <- capture_base_plot(
 heatmap(
   cor(USJudgeRatings), Rowv = FALSE, symm = TRUE, col = topo.colors(16),
   distfun = function(c) as.dist(1 - c), keep.dendro = TRUE,
   cexRow = 0.5, cexCol = 0.5))
# Add the heatmap to the fourth row, second column
figure %<>% fill_panel(
  a_base_plot,
  row = 4, column = 2)

# Incorporate a ComplexHeatmap figure
require(ComplexHeatmap)
mat = matrix(rnorm(80, 2), 8, 10)
mat = rbind(mat, matrix(rnorm(40, -2), 4, 10))
rownames(mat) = letters[1:12]
colnames(mat) = letters[1:10]
ht = Heatmap(mat)
a_complex_heatmap <- ht + ht + ht
# Add the ComplexHeatmap to the fourth row, third column
(figure %<>% fill_panel(
  a_complex_heatmap,
  row = 4, column = 3))
```

`grob_dimensions` *Convenient Access to* `grob` *Dimensions*

Description

Convenience functions extracting dimensions from `grob` objects.

Usage

```
figure_width(grob, unit_to = "mm")
figure_height(grob, unit_to = "mm")
```

Arguments

`grob`	A `grob` object for which dimensions are to be retrieved.
`unit_to`	A single `character` string representing a valid **grid**-`unit`.

Value

Single `numeric` objects are returned.

Author(s)

Johannes Graumann

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

See Also

`multi_panel_figure`, `save_multi_panel_figure`

Examples

```
# Get dimensions of a grid grob
a_circle <- grid::circleGrob(x = 15, y = 30, r = 15, default.unit = "mm")
figure_height(a_circle)
figure_width(a_circle)

# Use the unit_to arg to convert units
figure_height(a_circle, unit_to = "in")
figure_width(a_circle, unit_to = "cm")

# Get dimensions of a multi-panel figure
figure <- multi_panel_figure(width = 55, height = 55, rows = 2, columns = 2)
figure_height(figure)
figure_width(figure)

# ggsave defaults to measuring dimensions in inches
width <- figure_width(figure, unit_to = "in")
height <- figure_height(figure, unit_to = "in")
tmp_file <- tempfile(fileext = ".png")
ggplot2::ggsave(
  tmp_file, gtable::gtable_show_layout(figure),
  width = width, height = height
)
 # Not testing due to use of external software
utils::browseURL(tmp_file)
```

`images` *Images*

Description

A TIFF photograph of Farouq the cat in a washing machine. CC-BY-SA 4.0 Richard Cotton, 2014.

Format

An image file.

Details

A JPEG photograph of a greater one-horned rhinoceros (Rhinoceros unicornis) taken in Kaziranga National Park, Assam, India. CC-BY-SA 4.0 Janette Cotton, 2016.

An SVG picture of a fat, pink winged unicorn. CC-BY-SA 4.0 Sara Lendal, 2016.

A PNG of the R logo. CC-BY-SA 4.0 The R Foundation, 2016.

Examples

```
## Not run:
  figure <- multi_panel_figure(
    width = c(60, 40, 40), height = c(40, 40, 40)
  )
  image_files <- system.file("extdata", package = "multipanelfigure") %>%
    dir(full.names = TRUE) %>%
    setNames(basename(.))
  figure %>%
    fill_panel(image_files["farouq.tiff"]) %>%
    fill_panel(image_files["unicorn.svg"], column = 2:3) %>%
    fill_panel(image_files["rhino.jpg"], row = 2:3) %>%
    fill_panel(image_files["Rlogo.png"], column = 2:3, row = 2:3)

## End(Not run)
```

`multipanelfigure-deprecated`

Superceded objects in multipanelfigure

Description

Functions that are no longer used or have been superceded by functions with underscore-separated names.

Usage

```
addPanel( figure, ... )

capturebaseplot( ... )

multipanelfigure( ... )

simplegrobheight( ... )

simplegrobwidth( ... )
```

Arguments

`figure`	Object of classes `multipanelfigure`/`gtable` as produced by `multi_panel_figure` and representing the figure the panel is to be placed in.
`...`	Arguments to functions you shouldn't use.

`multi_panel_figure` *multi_panel_figure*

Description

A convenience function building `gtable`-based infrastructure for the assembly of multipanel figures.

Usage

```
multi_panel_figure(
  width = "auto",
  columns = NULL,
  height = "auto",
  rows = NULL,
  row_spacing = NaN,
  column_spacing = NaN,
  unit = "mm",
  figure_name = "FigureX",
  panel_label_type = c("upper-alpha", "lower-alpha", "decimal", "upper-roman",
    "lower-roman", "upper-greek", "lower-greek", "none"),
  ...
)
```

Arguments

`width`	`numeric` or `link[grid]{unit}` defining the width(s) of the resulting `gtable` if `length(width)` == 1 or individual column widths. Units depends on `unit` if not provided as `unit` object. The default '`auto`' sets the parameter to the width of the currently used device. See 'Details' for dependent and interfering parameters.
`columns`	Single `numeric` defining the number of columns in the resulting `gtable`. See 'Details' for dependent and interfering parameters.
`height`	`numeric` or `link[grid]{unit}` defining the height of the resulting `gtable` if `length(height)` == 1 or individual row heights.nits depends on `unit` if not provided as `unit` object. The default '`auto`' sets the parameter to the height of the currently used device. See 'Details' for dependent and interfering parameters.
`rows`	Single `numeric` defining the number of rows in the resulting `gtable`. See 'Details' for dependent and interfering parameters.

`row_spacing`	`numeric` or #' `unit` defining the amount of white space automatically inserted between row panels. Defaults to 5 mm unless explicitly given, in which case the value may depend on the `unit` parameter. Recycled to the number of rows.
`column_spacing`	`numeric` or `unit` defining the amount of white space automatically inserted between column panels. Defaults to 5 mm unless explicitly given, in which case the value may depends on the `unit` parameter. Recycled to the number of columns.
`unit`	Single `character` object defining the unit of all dimensions defined. Must satisfy `grid:::valid.units`.
`figure_name`	Single `character` object defining the name of the resulting `gtable`.
`panel_label_type`	A string specifying the marker style for the panel labels used for automated annotation. Defaults to uppercase Latin letters.
`...`	Argument to accomodate deprecated arguments `widths` and `heights`.

Details

The `gtable` may be constructed in two ways:

1. Based on explicit width/height definitions for individual panels.
2. Based on total figure/`gtable` dimensions given by `width` and `height` together with the number of `columns` and `rows` requested.

The function automatically inserts whitespace of width `column_spacing` before column panels (and of height `row_spacing` before row panels), which has to be considered for the total dimensions of the resulting `gtable`. Width of the `gtable` in the former case, for example may be calculated

$$W[total] = sum(width) + length(width) * column_spacing$$

while width of resulting panels in the latter table construction approach may be calculated

$$W[panel] = (width - columns * column_spacing)/columns$$

`width`, `height`, `column_spacing` and `row_spacing` may be defined numerically or as `unit` objects.

Earlier implementations used parameters `widhts` and `heights` as synonyms for `width` and `height` with length greater than one. These parameters have been deprecated. They continue to work, but produce a warning.

The two approaches to `gtable` construction require interdepending parameter sets:

Individual definition of panel dimensions: Requires `width` and `height` of lengths corresponding to the number of columns/rows requested. Excludes the use of `columns` and `rows`.

Definition of global `gtable`/figure dimensions: Requires `width`, `columns`, `height` and `rows` of length 1.

Value

Returns an object of class `multipanelfigure` as well as `gtable` object with the following additional attributes:

`multipanelfigure.panelsFree`**:** A `logical` `matrix` with the dimensions of the `gtable` indicating occupancy of the panels in the table.

`multipanelfigure.panellabelsfree`**:** A `character` `vector` indicative of the `panel_labels` still available.

`multipanelfigure.unit`**:** A single `character` object storing the corresponding value given during object creation.

Author(s)

Johannes Graumann

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

See Also

`fill_panel` for more examples of filling panels `figure_width` for inspecting figure dimensions `capture_base_plot` for including plots created using base graphics `gtable` for the underlying structure of a figure

Examples

```
## Not run:
# Figure construction based on the dimensions of the current device
figure1 <- multi_panel_figure(
   columns = 2,
   rows = 2,
   figure_name = "figure1")

# With no panels, printing shows the layout
figure1

# Figure construction based on overall dimensions
figure2 <- multi_panel_figure(
   width = 100,
   columns = 4,
   height = 90,
   rows = 6,
   figure_name = "figure2")

# Still no panels ...
figure2

# Figure construction based on individual panel dimensions
(figure3 <- multi_panel_figure(
   width = c(40,30),
   height = c(40,60),
   row_spacing = c(5, 1),
   column_spacing = c(0, 10),
```

```
  figure_name = "figure3"))

# A more involved example including filling and printing to device ...
# Make a simple ggplot object to fill panels
ggp <- ggplot2::ggplot(mtcars, ggplot2::aes(wt, mpg)) +
  ggplot2::geom_point()
# Fill panels
# ggplots and lattice plot objects are added directly
# The default position is the top-left panel
figure3 <- fill_panel(figure3, ggp)
# Raster images are added by passing the path to their file
jpg <- system.file("extdata/rhino.jpg", package = "multipanelfigure")
figure3 <- fill_panel(figure3, jpg, column = 2)
# Plots can take up multiple panels
figure3 <- fill_panel(figure3, ggp, row = 2, column = 1:2)
# Plot to appropriately sized png device
tmpFile <- tempfile(fileext = ".png")
ggplot2::ggsave(
  tmpFile, figure3,
  width = figure_width(figure3, "in"),
  height = figure_height(figure3, "in"))
message(
  paste0("Now have a look at '",tmpFile,"' - nicely sized PNG output."))
\donttest{ # Not testing due to use of external software
utils::browseURL(tmpFile)
}

## End(Not run)
```

`print.multipanelfigure` *Print a multi-panel figure*

Description

Prints and object of class `multipanelfigure`.

Usage

```
## S3 method for class 'multipanelfigure'
print(x, newpage = TRUE, ...)
```

Arguments

x	An object of class `multipanelfigure`.
newpage	Logical. If `TRUE`, a new device page is opened before drawing.
...	Passed from other print methods.

Value

The input `x` is invisibly returned, but the method is mostly invoked for the side effect of printing the plot to the current device.

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

Examples

```
p <- lattice::xyplot(dist ~ speed, cars)
figure <- multi_panel_figure(
  width = 100, height = 100,
  rows = 1, columns = 1
)
# With no panels, printing shows the layout
print(figure)
figure <- fill_panel(figure, p)
# After a panel is added, printing shows the plot.
print(figure) # shows plot
```

`save_multi_panel_figure` *save_multi_panel_figure*

Description

A convenience function wrapping `ggsave` from **ggplot2** for easy saving of `gtable` objects constructed by `multi_panel_figure` taking into account the table's dimensions.

Usage

```
save_multi_panel_figure(figure, filename, dpi = 300, ...)
```

Arguments

`figure`	Object of classes `multipanelfigure`/ `gtable` as produced by `multi_panel_figure`.
`filename`	Single `character` object representing file name/path to create on disk.
`dpi`	Single `numeric` indicating the plot resolution. Applies only to raster output types.
`...`	Other arguments passed to `ggsave`.

Details

Plot dimensions are determined using `figure_height` and `figure_width`.

The Device type to use is guessed from the `filename` extension. Currently supported are "eps", "ps", "tex" (pictex), "pdf", "jpeg", "tiff", "png", "bmp", "svg" or "wmf" (windows only).

Author(s)

Johannes Graumann

References

Graumann, J., and Cotton, R.J. (2018). multipanelfigure: Simple Assembly of Multiple Plots and Images into a Compound Figure. Journal of Statistical Software 84. doi: 10.18637/jss.v084.c03

See Also

`ggsave`, `figure_width`, `figure_height`

Examples

```
# Create the figure layout
(figure <- multi_panel_figure(
  width = c(30,40,60),
  height = c(40,60,60,60),
  panel_label_type = "upper-roman"))

# Fill the top-left panel using a grob object directly
a_grob <- grid::linesGrob(arrow = grid::arrow())
figure %<>% fill_panel(a_grob)

## Not run:
# Save the figure
figure %>%
  save_multi_panel_figure(
    filename = paste0(
    tempfile(),
    ".png"))

## End(Not run)
```

`%>%` *magrittr forward-pipe operator*

Description

See `%>%`.

`%<>%` *magrittr compound assignment pipe operator*

Description

See `%<>%`.

Index