Package ‘iheatmapr’

January 26, 2024

Type Package

Title Interactive, Complex Heatmaps

Version 0.7.1

Description Make complex, interactive heatmaps. 'iheatmapr' includes a modular system for iteratively building up complex heatmaps, as well as the iheatmap() function for making relatively standard heatmaps.

License MIT + file LICENSE

Depends R (>= 3.5.0)

Imports methods, utils, magrittr, stats, ggdendro, fastcluster, RColorBrewer, htmlwidgets, jsonlite, scales, knitr, tools

Suggests htmltools, datasets, shiny, testthat, rmarkdown, reshape2, roxygen2, covr, webshot

RoxygenNote 7.3.1

VignetteBuilder knitr

URL `https://docs.ropensci.org/iheatmapr/` (website)
`https://github.com/ropensci/iheatmapr`

BugReports `https://github.com/ropensci/iheatmapr/issues`

Encoding UTF-8

Collate 'AllClasses.R' 'AllGenerics.R' 'annotations.R' 'axes.R' 'axis_labels.R' 'axis_titles.R' 'barplot.R' 'clustering.R' 'clusters.R' 'colorbars.R' 'colors.R' 'components.R' 'dendogram.R' 'generic_subplots.R' 'groups.R' 'main_heatmap.R' 'iheatmap.R' 'iheatmapr.R' 'layout.R' 'list_utils.R' 'shiny.R' 'shiny_test.R' 'signal.R' 'subplot.R' 'summary.R' 'to_widget.R' 'utils.R'

X-schema.org-applicationCategory Visualization

X-schema.org-keywords heatmap, plotly, r, interactive-visualizations, data-visualization, htmlwidgets

X-schema.org-isPartOf https://ropensci.org

Language en-US

NeedsCompilation no

Author Alicia Schep [aut] (<https://orcid.org/0000-0002-3915-0618>),
Sarah Kummerfeld [aut] (<https://orcid.org/0000-0002-0089-2358>),
Genentech [cph],
Andee Kaplan [rev] (Reviewer for rOpensci review process
https://github.com/ropensci/onboarding/issues/107),
Carl Ganz [rev] (Reviewer for rOpensci review process
https://github.com/ropensci/onboarding/issues/107),
Alex Johnson [ctb] (plotly.js library),
Étienne Tétreault-Pinard [ctb] (plotly.js library),
Mikola Lysenko [ctb] (plotly.js library),
Ricky Reusser [ctb] (plotly.js library),
Robert Monfera [ctb] (plotly.js library),
Nicolas Riesco [ctb] (plotly.js library),
Miklós Tusz [ctb] (plotly.js library),
Chelsea Douglas [ctb] (plotly.js library),
Ben Postlethwaite [ctb] (plotly.js library),
Chris Parmer [ctb] (plotly.js library),
Alex Vados [ctb] (plotly.js library),
Plotly [cph] (plotly.js library),
Frederick Boehm [ctb] (colorbars & main_heatmap),
Alan O'Callaghan [cre]

Maintainer Alan O'Callaghan <alan.ocallaghan@outlook.com>

Repository CRAN

Date/Publication 2024-01-25 23:50:02 UTC

R topics documented:

`add_col_annotation` *add_col_annotation*

Description

Adds annotation heatmaps for one or more qualitative or quantitative annotations for each column of a main heatmap.

Usage

```
## S4 method for signature 'Iheatmap'
add_col_annotation(
  p,
  annotation,
  colors = NULL,
  side = c("top", "bottom"),
  size = 0.05,
  buffer = 0.015,
  inner_buffer = buffer/2,
  layout = list(),
  show_colorbar = TRUE
)
```

Arguments

`p`	`link{Iheatmap-class}` object
`annotation`	data.frame or object that can be converted to data frame
`colors`	list of color palettes, with one color per annotation column name
`side`	side of plot on which to add column annotation
`size`	relative size of each row annotation
`buffer`	relative size of buffer between previous subplot and column annotation
`inner_buffer`	relative size of buffer between each annotation
`layout`	layout properties for new y axis
`show_colorbar`	logical indicator to show or hide colorbar

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_row_annotation`, `add_col_signal`, `add_col_groups`

Examples

```
mat <- matrix(rnorm(24), ncol = 6)
annotation <- data.frame(gender = c(rep("M", 3),rep("F",3)),
                         age = c(20,34,27,19,23,30))
hm <- iheatmap(mat) %>% add_col_annotation(annotation)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_barplot` *add_col_barplot*

Description

Add bar plot with one bar per column above or below a main heatmap

Usage

```
## S4 method for signature 'Iheatmap'
add_col_barplot(
  p,
  y,
  ...,
  color = NULL,
  tracename = NA_character_,
  showlegend = !is.na(tracename),
  side = c("top", "bottom"),
  layout = list(),
  size = 0.2,
  buffer = 0.02,
  xname = current_xaxis(p),
  yname = NULL,
  pname = if (!is.na(tracename)) tracename else "col_barplot"
)
```

Arguments

`p`	iheatmap object
`y`	y axis values
`...`	additional arguments to add to plotly scatter trace, see `https://plotly.com/javascript/reference/#scatter`
`color`	color of bars
`tracename`	name of trace (for legend and hover)
`showlegend`	show in legend?
`side`	side of plot on which to add subplot
`layout`	yaxis layout list
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_col_signal`, `iheatmap`, `add_col_plot`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_col_barplot(y = 1:5, tracename = "Strength")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_clustering` *add_col_clustering*

Description

add_col_clustering

Usage

```
## S4 method for signature 'Iheatmap'
add_col_clustering(
  p,
  method = c("hclust", "kmeans", "groups"),
  name = "Col<br>Clusters",
  k = NULL,
  groups = NULL,
  clust_dist = stats::dist,
  colors = NULL,
  show_colorbar = TRUE,
  side = c("top", "bottom"),
  yname = NULL,
  xname = current_xaxis(p)
)
```

Arguments

`p`	iheatmap object
`method`	"hclust" or "kmeans" for hierarchical or k-means clustering, respectively
`name`	name of colorbar indicating cluster membership
`k`	number of clusters for rows, needed if order is kmeans or optional if hclust
`groups`	vector of group assignments
`clust_dist`	distance function to use for clustering if hierarchical clustering
`colors`	colors to use for annotation of grouping, can be RColorBrewer palette name or vector of colors
`show_colorbar`	show the colorbar for the heatmap indicating cluster membership
`side`	side of plot on which to add subplot
`yname`	name of yaxis
`xname`	name of xaxis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_clustering`, `iheatmap`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_col_clustering(method = "hclust", k = 2)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_clusters` *add_col_clusters*

Description

Add column groups and order columns based on groups

Usage

```
## S4 method for signature 'Iheatmap'
add_col_clusters(
  p,
  clusters,
  name = "Col<br>Clusters",
  reorder = TRUE,
  side = c("top", "bottom"),
  xname = current_xaxis(p),
  ...
)
```

Arguments

`p`	iheatmap object
`clusters`	cluster assignments, should be vector of integers, characters, or factors
`name`	name of colorbar indicating cluster membership
`reorder`	reorder rows based on clusters? default is TRUE
`side`	side of plot on which to add subplot

`xname`	name of xaxis
`...`	additional arguments to pass to `add_col_groups` for creation of annotation heatmap indicating cluster membership

Details

This function is very similar to `add_col_groups`; the main difference is that with this function column will get reordered based on the groups.

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_clusters`, `add_col_clustering`, `iheatmap`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
clusters <- c("A","B","A","B","A")

hm <- iheatmap(mat) %>% add_col_clusters(clusters)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_dendro` *add_col_dendro*

Description

Adds column dendrogram to iheatmap object

Usage

```
## S4 method for signature 'Iheatmap,hclust'
add_col_dendro(
  p,
  dendro,
  reorder = TRUE,
  side = c("top", "bottom"),
  size = 0.15,
```

```
  buffer = 0.005,
  xname = current_xaxis(p),
  yname = NULL,
  sname = "col_dendro"
)
```

Arguments

`p`	iheatmap object
`dendro`	hclust object
`reorder`	reorder rows based on dendrogram order?
`side`	side of plot on which to add dendro
`size`	relative size of dendrogram (relative to the main heatmap)
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`sname`	internal name of shape

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_col_clustering`, `iheatmap`, `add_row_dendro`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
dend <- hclust(dist(t(mat)))
hm <- iheatmap(mat) %>% add_col_dendro(dend)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_groups` *add_col_groups*

Description

Adds annotation to heatmap indicating what group every column of main heatmap belongs to

Usage

```
## S4 method for signature 'Iheatmap'
add_col_groups(
  p,
  groups,
  name = "Column<br>Groups",
  title = "Groups",
  colors = pick_discrete_colors(groups, p),
  colorbar_position = get_colorbar_position(p),
  show_colorbar = TRUE,
  show_title = TRUE,
  side = c("top", "bottom"),
  layout = list(),
  size = 0.05,
  buffer = 0.015,
  tooltip = setup_tooltip_options(),
  xname = current_xaxis(p),
  yname = NULL,
  pname = name
)
```

Arguments

`p`	`Iheatmap-class` object
`groups`	vector of group names
`name`	name of colorbar
`title`	name of x axis label
`colors`	palette name or vector of colors
`colorbar_position`	colorbar placement
`show_colorbar`	show the colorbar?
`show_title`	show title as axis label
`side`	side of plot on which to groups annotation
`layout`	list of layout parameters for x axis
`size`	relative size of dendrogram (relative to the main heatmap)
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap

`tooltip`	tooltip options, see `setup_tooltip_options`
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_row_groups`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
col_groups <- c("A","A","B","D","B")
hm <- iheatmap(mat) %>% add_col_groups(col_groups, name = "My Groups")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_labels` *add_col_labels*

Description

Add x axis labels to plot

Usage

```
## S4 method for signature 'Iheatmap'
add_col_labels(
  p,
  tickvals = NULL,
  ticktext = NULL,
  textangle = -90,
  font = get_layout(p)$font,
  side = c("bottom", "top"),
  size = 0.1,
  buffer = 0.005,
  xname = current_xaxis(p),
  yname = NULL
)
```

Arguments

`p`	`link{Iheatmap-class}` object
`tickvals`	column indices at which to place axis tick labels
`ticktext`	text for axis tick labels
`textangle`	angle for ticktext
`font`	list of plotly font attributes, see `https://plotly.com/javascript/reference/#layout-font`
`side`	side of plot on which to add subplot
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	name for xaxis
`yname`	name for yaxis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_title`, `iheatmap`, `add_col_labels`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm1 <- iheatmap(mat) %>% add_col_labels()
hm2 <- iheatmap(mat) %>% add_col_labels(ticktext = letters[22:26])

# Print heatmap if interactive session
if (interactive()) hm1
if (interactive()) hm2
```

`add_col_plot` *add_col_plot*

Description

Add a scatter or line plot with one point per column of the main heatmap

Usage

```
## S4 method for signature 'Iheatmap'
add_col_plot(
  p,
  y,
  ...,
  mode = c("lines+markers", "lines", "markers"),
  color = NULL,
  tracename = NA_character_,
  showlegend = !is.na(tracename),
  side = c("top", "bottom"),
  layout = list(),
  size = 0.2,
  buffer = 0.02,
  xname = current_xaxis(p),
  yname = NULL,
  pname = if (!is.na(tracename)) tracename else "col_plot"
)
```

Arguments

`p`	iheatmap object
`y`	y axis values
`...`	additional arguments to add to plotly scatter trace, see `https://plotly.com/javascript/reference/#scatter`
`mode`	mode of plot – one of "lines+markers","lines", or "markers"
`color`	color of bars
`tracename`	name of trace (for legend and hover)
`showlegend`	show in legend?
`side`	side of plot on which to add subplot
`layout`	yaxis layout list
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

add_col_signal, iheatmap, add_col_barplot

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_col_plot(y = 1:5, tracename = "Strength")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_signal` *add_col_signal*

Description

Adds column signal to iheatmap object

Usage

```
## S4 method for signature 'Iheatmap'
add_col_signal(
  p,
  signal,
  name,
  title = name,
  yname = NULL,
  xname = current_xaxis(p),
  pname = name,
  colorbar_position = get_colorbar_position(p),
  colors = pick_continuous_colors(zmid, zmin, zmax, p = p),
  zmin = min(signal, na.rm = TRUE),
  zmax = max(signal, na.rm = TRUE),
  zmid = 0,
  side = c("top", "bottom"),
  size = 0.05,
  buffer = 0.015,
  text = signif(signal, digits = 3),
  tooltip = setup_tooltip_options(),
  show_colorbar = TRUE,
  show_title = TRUE,
  layout = list()
)
```

Arguments

`p`	iheatmap object
`signal`	vector of signal
`name`	name of colorbar
`title`	label for y axis
`yname`	internal name of yaxis
`xname`	internal name of xaxis
`pname`	internal name of plot
`colorbar_position`	colorbar placement
`colors`	palette or vector of colors to use
`zmin`	minimum for colorscale
`zmax`	maximum for colorscale
`zmid`	midpoint for colorscale
`side`	side of plot on which to add groups
`size`	relative size of dendrogram (relative to the main heatmap)
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`text`	text of value to display for data
`tooltip`	tooltip options, see `setup_tooltip_options`
`show_colorbar`	show the colorbar?
`show_title`	show title as axis label
`layout`	y axis layout parameters to use

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_row_groups`

`add_row_signal`, `iheatmap`, `add_col_annotation`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_col_signal(signal = 1:5, name = "Strength")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_col_summary` *add_col_summary*

Description

Adds a line plot summarizing the values across columns

Usage

```
## S4 method for signature 'Iheatmap'
add_col_summary(
  p,
  groups = NULL,
  heatmap_name = NULL,
  colors = NULL,
  tracename = "Col Summary",
  showlegend = FALSE,
  side = c("top", "bottom"),
  layout = list(),
  size = 0.3,
  buffer = 0.02,
  xname = current_xaxis(p),
  yname = NULL,
  type = c("scatter", "bar"),
 summary_function = c("mean", "median", "sd", "var", "mad", "max", "min", "sum"),
  ...
)
```

Arguments

`p`	`Iheatmap-class` object
`groups`	vector of group labels, name of groups colorbar, or TRUE – see Details
`heatmap_name`	name of a heatmap within the plot
`colors`	vector of colors or RColorBrewer palette name
`tracename`	name of trace
`showlegend`	show legend?
`side`	side of plot on which to add subplot
`layout`	xaxis layout list
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`type`	scatter or bar?

`summary_function`	summary function to use, default is mean, options are mean, median, sd, var, mad, max, min, and sum
`...`	additional arguments to `add_col_plot` or `add_col_barplot`

Details

If adding the column summary to a vertically oriented heatmap, the summary will be based on the topmost heatmap if side is "top" and based on the bottom heatmap if side is "bottom" unless a "heatmap_name" is specified. The heatmap_name should match the "pname" argument given to a previously added heatmap.

The column summary is based on specific rows if a "groups" argument is given. The groups argument can either be a vector of group assignments for each row, the "pname" for an existing set of groups incorporated into the plot using `add_row_groups`, `add_row_annotation`, `add_row_clusters`, or `add_row_clustering`. If groups is set to TRUE, then the function will use an existing set of row groups added to the plot.

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_summary`, `iheatmap`, `add_col_plot`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm1 <- iheatmap(mat) %>% add_col_summary()
hm2 <- iheatmap(mat) %>% add_col_summary(groups = c("A","A","B","B"))

# Print heatmap if interactive session
if (interactive()) hm1
if (interactive()) hm2
```

`add_col_title` *add_col_title*

Description

Add x axis title to plot

Usage

```
## S4 method for signature 'Iheatmap'
add_col_title(
  p,
  title,
  textangle = 0,
  font = get_layout(p)$font,
  side = c("bottom", "top"),
  size = 0.1,
  buffer = 0.01,
  xname = current_xaxis(p),
  yname = NULL
)
```

Arguments

`p`	iheatmap object
`title`	title of axis
`textangle`	angle of text
`font`	list of plotly font attributes, see `https://plotly.com/javascript/reference/#layout-font`
`side`	side of plot on which to add subplot
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	name for xaxis
`yname`	name for yaxis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_col_labels`, `iheatmap`, `add_row_title`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_col_title("My x-axis")

# Print heatmap if interactive session
if (interactive()) hm
```

---

add_iheatmap *add_iheatmap*

Description

add_iheatmap

Usage

```
## S4 method for signature 'IheatmapHorizontal,matrix'
add_iheatmap(
  p,
  data,
  x = default_x(data),
  cluster_cols = c("none", "hclust", "kmeans", "groups"),
  col_clusters = NULL,
  col_k = NULL,
  col_clust_dist = stats::dist,
  name = "Signal",
  scale = c("none", "rows", "cols"),
  scale_method = c("standardize", "center", "normalize"),
  colors = NULL,
  col_clusters_colors = NULL,
  col_clusters_name = "Col<br>Clusters",
  show_col_clusters_colorbar = TRUE,
  row_annotation = NULL,
  col_annotation = NULL,
  row_annotation_colors = NULL,
  col_annotation_colors = NULL,
  row_labels = NULL,
  col_labels = NULL,
  row_title = NULL,
  col_title = NULL,
  buffer = 0.2,
  ...
)

## S4 method for signature 'IheatmapVertical,matrix'
add_iheatmap(
  p,
  data,
  y = default_y(data),
  cluster_rows = c("none", "hclust", "kmeans", "groups"),
  row_clusters = NULL,
  row_k = NULL,
  row_clust_dist = stats::dist,
  name = "Signal",
```

```
  scale = c("none", "rows", "cols"),
  scale_method = c("standardize", "center", "normalize"),
  colors = NULL,
  row_clusters_colors = NULL,
  row_clusters_name = "Col<br>Clusters",
  show_row_clusters_colorbar = TRUE,
  row_annotation = NULL,
  col_annotation = NULL,
  row_annotation_colors = NULL,
  col_annotation_colors = NULL,
  row_labels = NULL,
  col_labels = NULL,
  row_title = NULL,
  col_title = NULL,
  buffer = 0.2,
  ...
)
```

Arguments

`p`	iheatmap object
`data`	matrix of values to be plotted as heatmap
`x`	x xaxis labels, by default colnames of data
`cluster_cols`	"none","hclust", or "k-means" for no clustering, hierarchical clustering, and k-means clustering of columnsrespectively
`col_clusters`	vector of pre-determined column cluster assignment
`col_k`	number of clusters for columns, needed if cluster_rows is kmeans or optional if hclust
`col_clust_dist`	distance function to use for column clustering if hierarchical clustering
`name`	Name for colorbar
`scale`	scale matrix by rows, cols or none
`scale_method`	what method to use for scaling, either standardize, center, normalize
`colors`	name of RColorBrewer palette or vector of colors for main heatmap
`col_clusters_colors`	colors for col clusters annotation heatmap
`col_clusters_name`	name for col clusters colorbar
`show_col_clusters_colorbar`	show the colorbar for column clusters?
`row_annotation`	row annotation data.frame
`col_annotation`	column annotation data.frame
`row_annotation_colors`	list of colors for row annotations heatmap
`col_annotation_colors`	list of colors for col annotations heatmap

`row_labels`	axis labels for y axis
`col_labels`	axis labels for x axis
`row_title`	x axis title
`col_title`	y axis title
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`...`	additional argument to add_iheatmap
`y`	y axis labels, by default rownames of data
`cluster_rows`	"none","hclust", or "k-means" for no clustering, hierarchical clustering, and k-means clustering of rows respectively
`row_clusters`	vector of pre-determined row cluster assignment
`row_k`	number of clusters for rows, needed if cluster_rows is kmeans or optional if hclust
`row_clust_dist`	distance function to use for row clustering if hierarchical clustering
`row_clusters_colors`	colors for row clusters annotation heatmap
`row_clusters_name`	name for row clusters colorbar
`show_row_clusters_colorbar`	show the colorbar for row clusters?

Details

By default, no scaling is done of rows or columns. This can be changed by specifying the 'scale' argument. There are three options for scaling methods. "standardize" subtracts the mean and divides by standard deviation, "center" just subtracts the mean, and "normalize" divides by the sum of the values. "normalize" should only be used for data that is all positive! If alternative scaling is desired, the scaling should be done prior to calling the iheatmap function.

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `main_heatmap`

Examples

```
mat <- matrix(rnorm(24), nrow = 6)
mat2 <- matrix(rnorm(24), nrow = 6)
annotation = data.frame(gender = c(rep("M", 3),rep("F",3)),
```

```
                         age = c(20,34,27,19,23,30))
hm <- iheatmap(mat,
 cluster_rows = "hclust",
 cluster_cols = "hclust",
 col_k = 3) %>%
add_iheatmap(mat2,
 cluster_cols = "hclust",
 col_k = 3,
 row_annotation = annotation)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_main_heatmap` *add_main_heatmap*

Description

Adds an additional main heatmap to an iheatmap object

Usage

```
## S4 method for signature 'IheatmapHorizontal,matrix'
add_main_heatmap(
  p,
  data,
  name = "Signal",
  x = default_x(data),
  colors = pick_continuous_colors(zmid, zmin, zmax, p),
  colorbar_position = get_colorbar_position(p),
  show_colorbar = TRUE,
  zmin = min(data, na.rm = TRUE),
  zmax = max(data, na.rm = TRUE),
  zmid = 0,
  col_order = NULL,
  x_categorical = NULL,
  side = c("right", "left"),
  size = 1,
  buffer = 0.04,
  text = signif(data, digits = 3),
  tooltip = setup_tooltip_options(),
  xname = NULL,
  pname = name,
  ...
)

## S4 method for signature 'IheatmapVertical,matrix'
add_main_heatmap(
```

```
  p,
  data,
  name = "Signal",
  y = default_y(data),
  colors = pick_continuous_colors(zmid, zmin, zmax, p),
  colorbar_position = get_colorbar_position(p),
  show_colorbar = TRUE,
  zmin = min(data, na.rm = TRUE),
  zmax = max(data, na.rm = TRUE),
  zmid = 0,
  row_order = NULL,
  y_categorical = NULL,
  side = c("bottom", "top"),
  size = 1,
  buffer = 0.04,
  text = signif(data, digits = 3),
  tooltip = setup_tooltip_options(),
  yname = NULL,
  pname = name,
  ...
)
```

Arguments

`p`	`Iheatmap-class` object
`data`	matrix
`name`	name of colorbar, will determine if colorbar is shared with existing plot
`x`	x axis labels (by default rownames of data); only used if orientation is horizontal
`colors`	color palette name or vector of colors
`colorbar_position`	colorbar placement
`show_colorbar`	display the colorbar?
`zmin`	minimum for colorscale
`zmax`	maximum for colorscale
`zmid`	midpoint for scale
`col_order`	column ordering for this heatmap; only used if orientation is horizontal
`x_categorical`	is x categorical? will guess if not provided
`side`	which side of the current plot to add this heatmap?
`size`	relative size of plot. size relative to first heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`text`	text of value to display for data
`tooltip`	tooltip options, see `setup_tooltip_options`
`xname`	internal name for x axis

`pname`	internal name for plot
`...`	additional arguments (ignored)
`y`	y axis labels (by default colnames of data); only used if orientation is vertical
`row_order`	row ordering for this heatmap; only used if orientation is vertical
`y_categorical`	is y categorical? will guess if not provided
`yname`	internal name for y axis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `main_heatmap`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
mat2 <-  matrix(rnorm(24), ncol = 6, nrow = 4)
hm <- iheatmap(mat) %>% add_main_heatmap(mat2)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_annotation` *add_row_annotation*

Description

Adds annotation heatmaps for one or more qualitative or quantitative annotations for each row of a main heatmap.

Usage

```
## S4 method for signature 'Iheatmap'
add_row_annotation(
  p,
  annotation,
  colors = NULL,
  side = c("right", "left"),
  size = 0.05,
  buffer = 0.015,
```

```
  inner_buffer = buffer/2,
  layout = list(),
  show_colorbar = TRUE
)
```

Arguments

`p`	`link{Iheatmap-class}` object
`annotation`	data.frame or object that can be converted to data frame
`colors`	list of color palettes, with one color per annotation column name
`side`	side of plot on which to add row annotation
`size`	relative size of each row annotation
`buffer`	relative size of buffer between previous subplot and row annotation
`inner_buffer`	relative size of buffer between each annotation
`layout`	layout properties for new x axis
`show_colorbar`	logical indicator to show or hide colorbar

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_row_annotation`, `add_col_signal`, `add_col_groups`

Examples

```
mat <- matrix(rnorm(24), nrow = 6)
annotation <- data.frame(gender = c(rep("M", 3),rep("F",3)),
                        age = c(20,34,27,19,23,30))
hm <- iheatmap(mat) %>% add_row_annotation(annotation)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_barplot` *add_row_barplot*

Description

add_row_barplot

Usage

```
## S4 method for signature 'Iheatmap'
add_row_barplot(
  p,
  x,
  ...,
  color = NULL,
  tracename = NA_character_,
  showlegend = !is.na(tracename),
  side = c("right", "left"),
  layout = list(),
  size = 0.2,
  buffer = 0.02,
  xname = NULL,
  yname = current_yaxis(p),
  pname = if (!is.na(tracename)) tracename else "row_barplot"
)
```

Arguments

`p`	iheatmap object
`x`	x axis values
`...`	additional arguments to add to plotly scatter trace, see `https://plotly.com/javascript/reference/#scatter`
`color`	color of bars
`tracename`	name of trace (for legend and hover)
`showlegend`	show in legend?
`side`	side of plot on which to add subplot
`layout`	yaxis layout list
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_signal`, `iheatmap`, `add_row_plot`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_row_barplot(x = 1:4, tracename = "Strength")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_clustering` *add_row_clustering*

Description

add_row_clustering

Usage

```
## S4 method for signature 'Iheatmap'
add_row_clustering(
  p,
  method = c("hclust", "kmeans", "groups"),
  name = "Row<br>Clusters",
  k = NULL,
  groups = NULL,
  clust_dist = stats::dist,
  colors = NULL,
  show_colorbar = TRUE,
  side = c("left", "right"),
  xname = NULL,
  yname = current_yaxis(p)
)
```

Arguments

`p`	iheatmap object
`method`	"hclust" or "kmeans" for hierarchical or k-means clustering, respectively
`name`	name of colorbar indicating cluster membership
`k`	number of clusters for rows, needed if order is kmeans or optional if hclust
`groups`	vector of group assignments
`clust_dist`	distance function to use for clustering if hierarchical clustering
`colors`	colors to use for annotation of grouping, can be RColorBrewer palette name or vector of colors
`show_colorbar`	show the colorbar for the heatmap indicating cluster membership
`side`	side of plot on which to add subplot
`xname`	name of xaxis
`yname`	name of yaxis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_col_clustering`, `iheatmap`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_row_clustering(method = "hclust", k = 2)

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_clusters` *add_row_clusters*

Description

Add row groups and order rows based on groups

Usage

```
## S4 method for signature 'Iheatmap'
add_row_clusters(
  p,
  clusters,
  name = "Row<br>Clusters",
  reorder = TRUE,
  side = c("left", "right"),
  yname = current_yaxis(p),
  ...
)
```

Arguments

`p`	iheatmap object
`clusters`	cluster assignments, should be vector of integers, characters, or factors
`name`	name of colorbar indicating cluster membership
`reorder`	reorder rows based on clusters? default is TRUE
`side`	side of plot on which to add subplot
`yname`	name of yaxis
`...`	additional arguments to pass to `add_row_groups` for creation of annotation heatmap indicating cluster membership

Details

This function is very similar to `add_row_groups`; the main difference is that with this function rows will get reordered based on the groups.

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_clustering`, `add_col_clusters`, `iheatmap`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
clusters <- c("A","B","A","B")

hm <- iheatmap(mat) %>% add_row_clusters(clusters)
```

```
# Print heatmap if interactive session
if (interactive()) hm
```

---

add_row_dendro *add_row_dendro*

Description

Adds row dendrogram to iheatmap object

Usage

```
## S4 method for signature 'Iheatmap,hclust'
add_row_dendro(
  p,
  dendro,
  reorder = TRUE,
  side = c("left", "right"),
  size = 0.15,
  buffer = 0.005,
  xname = NULL,
  yname = current_yaxis(p),
  sname = "row_dendro"
)
```

Arguments

`p`	iheatmap object
`dendro`	hclust object
`reorder`	reorder rows based on dendrogram order?
`side`	side of plot on which to add dendrogram
`size`	relative size of dendrogram (relative to the main heatmap)
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`sname`	internal name of shapes

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

add_row_clustering, iheatmap, add_col_dendro

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
dend <- hclust(dist(mat))
hm <- iheatmap(mat) %>% add_row_dendro(dend)

# Print heatmap if interactive session
if (interactive()) hm
```

---

add_row_groups *add_row_groups*

Description

Adds annotation to heatmap indicating what group every row of main heatmap belongs to

Usage

```
## S4 method for signature 'Iheatmap'
add_row_groups(
  p,
  groups,
  name = "Row<br>Groups",
  title = "Groups",
  colors = pick_discrete_colors(groups, p),
  colorbar_position = get_colorbar_position(p),
  show_colorbar = TRUE,
  show_title = TRUE,
  side = c("right", "left"),
  layout = list(),
  size = 0.05,
  buffer = 0.005,
  tooltip = setup_tooltip_options(),
  xname = NULL,
  yname = current_yaxis(p),
  pname = name
)
```

Arguments

`p`	Iheatmap-class object
`groups`	vector of group names
`name`	name of colorbar

`title`	name of x axis label
`colors`	palette name or vector of colors
`colorbar_position`	colorbar placement
`show_colorbar`	show the colorbar?
`show_title`	show title as axis label
`side`	side of plot on which to groups annotation
`layout`	list of layout parameters for x axis
`size`	relative size of dendrogram (relative to the main heatmap)
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`tooltip`	tooltip options, see `setup_tooltip_options`
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_col_groups`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
row_groups <- c("A","A","B","D")
hm <- iheatmap(mat) %>% add_row_groups(row_groups, name = "My Groups")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_labels` *add_row_labels*

Description

Add y axis labels to plot

Usage

```
## S4 method for signature 'Iheatmap'
add_row_labels(
  p,
  tickvals = NULL,
  ticktext = NULL,
  textangle = 0,
  font = get_layout(p)$font,
  side = c("left", "right"),
  size = 0.1,
  buffer = 0.005,
  xname = NULL,
  yname = current_yaxis(p)
)
```

Arguments

`p`	`Iheatmap-class` object
`tickvals`	row indices at which to place axis tick labels
`ticktext`	text for axis tick labels
`textangle`	angle for ticktext
`font`	list of plotly font attributes, see `https://plotly.com/javascript/reference/#layout-font`
`side`	side of plot on which to add subplot
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name for xaxis
`yname`	internal name for yaxis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_title`, `iheatmap`, `add_col_labels`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm1 <- iheatmap(mat) %>% add_row_labels()
hm2 <- iheatmap(mat) %>% add_row_labels(ticktext = letters[23:26])


# Print heatmaps if interactive session
if (interactive()) hm1
if (interactive()) hm2
```

`add_row_plot` *add_row_plot*

Description

Add a scatter or line plot with one point per row of the main heatmap

Usage

```
## S4 method for signature 'Iheatmap'
add_row_plot(
  p,
  x,
  ...,
  mode = c("lines+markers", "lines", "markers"),
  color = NULL,
  tracename = NA_character_,
  showlegend = !is.na(tracename),
  side = c("right", "left"),
  layout = list(),
  size = 0.2,
  buffer = 0.02,
  xname = NULL,
  yname = current_yaxis(p),
  pname = if (!is.na(tracename)) tracename else "row_plot"
)
```

Arguments

`p`	iheatmap object
`x`	x axis values

`...`	additional arguments to add to plotly scatter trace, see https://plotly.com/javascript/reference/#scatter
`mode`	mode of plot – one of "lines+markers","lines", or "markers"
`color`	color of bars
`tracename`	name of trace (for legend and hover)
`showlegend`	show in legend?
`side`	side of plot on which to add subplot
`layout`	yaxis layout list
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_signal`, `iheatmap`, `add_row_barplot`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_row_plot(x = 1:4, tracename = "Strength")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_signal` *add_row_signal*

Description

Adds single column heatmap to iheatmap object

Usage

```
## S4 method for signature 'Iheatmap'
add_row_signal(
  p,
  signal,
  name,
  title = name,
  xname = NULL,
  yname = current_yaxis(p),
  pname = name,
  colorbar_position = get_colorbar_position(p),
  colors = pick_continuous_colors(zmid, zmin, zmax, p = p),
  zmin = min(signal, na.rm = TRUE),
  zmax = max(signal, na.rm = TRUE),
  zmid = 0,
  side = c("right", "left"),
  size = 0.05,
  buffer = 0.015,
  text = signif(signal, digits = 3),
  tooltip = setup_tooltip_options(),
  show_colorbar = TRUE,
  show_title = TRUE,
  layout = list()
)
```

Arguments

`p`	iheatmap object
`signal`	vector of signal
`name`	name of colorbar
`title`	label for x axis
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot
`colorbar_position`	colorbar placement
`colors`	color palette or vector of colors
`zmin`	minimum for colorscale
`zmax`	maximum for colorscale
`zmid`	midpoint for colorscale
`side`	side of plot on which to add dendro
`size`	relative size of dendrogram (relative to the main heatmap)
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`text`	text of value to display for data

`tooltip`	tooltip options, see `setup_tooltip_options`
`show_colorbar`	show the colorbar?
`show_title`	show title as axis label
`layout`	list of x axis layout parameters

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_col_groups`

`add_col_signal`, `iheatmap`, `add_row_annotation`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_row_signal(signal = 1:4, name = "Strength")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_row_summary` *add_row_summary*

Description

Adds a line plot summarizing the values across rows

Usage

```
## S4 method for signature 'Iheatmap'
add_row_summary(
  p,
  groups = NULL,
  heatmap_name = NULL,
  colors = NULL,
  tracename = "Row Summary",
  showlegend = FALSE,
  side = c("right", "left"),
  layout = list(),
  size = 0.3,
```

```
  buffer = 0.02,
  xname = NULL,
  yname = current_yaxis(p),
  type = c("scatter", "bar"),
 summary_function = c("mean", "median", "sd", "var", "mad", "max", "min", "sum"),
  ...
)
```

Arguments

`p`	`Iheatmap-class` object
`groups`	vector of group labels, name of groups colorbar, or TRUE – see Details
`heatmap_name`	name of a heatmap within the `Iheatmap-class` object
`colors`	vector of colors or RColorBrewer palette name
`tracename`	name of trace
`showlegend`	show legend?
`side`	side of plot on which to add subplot
`layout`	xaxis layout list
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`type`	scatter or bar?
`summary_function`	summary function to use, default is mean, options are mean, median, sd, var, mad, max, min, and sum
`...`	additional arguments to `add_row_plot` or `add_row_barplot`

Details

If adding the row summary to a horizontally oriented heatmap, the summary will be based on the right-most heatmap if side is "right" and based on the left heatmap if side is "left" unless a "heatmap_name" is specified. The heatmap_name should match the "pname" argument given to a previously added heatmap.

The row summary is based on specific columns if a "groups" argument is given. The groups argument can either be a vector of group assignments for each row, the "pname" for an existing set of groups incorporated into the plot using `add_col_groups`, `add_col_annotation`, `add_col_clusters`, or `add_col_clustering`. If groups is set to TRUE, then the function will use an existing set of column groups added to the plot.

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_col_summary`, `iheatmap`, `add_row_plot`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm1 <- iheatmap(mat) %>% add_row_summary()
hm2 <- iheatmap(mat) %>% add_row_summary(groups = c("A","A","B","B","B"))

# Print heatmap if interactive session
if (interactive()) hm1
if (interactive()) hm2
```

`add_row_title` *add_row_title*

Description

Add y axis title to plot

Usage

```
## S4 method for signature 'Iheatmap'
add_row_title(
  p,
  title,
  textangle = ifelse(side == "left", -90, 90),
  font = get_layout(p)$font,
  side = c("left", "right"),
  size = 0.1,
  buffer = 0.01,
  xname = NULL,
  yname = current_yaxis(p)
)
```

Arguments

`p`	iheatmap object
`title`	title of axis
`textangle`	angle of text
`font`	list of plotly font attributes, see `https://plotly.com/javascript/reference/#layout-font`

`side`	side of plot on which to add subplot
`size`	relative size of subplot relative to main heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name for xaxis
`yname`	internal name for yaxis

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_col_title`, `iheatmap`, `add_row_labels`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- iheatmap(mat) %>% add_row_title("Samples")

# Print heatmap if interactive session
if (interactive()) hm
```

`add_subplot` *add_subplot*

Description

Adds an arbitrary subplot to iheatmap

Usage

```
## S4 method for signature 'Iheatmap'
add_subplot(
  p,
  ...,
  side = c("top", "bottom", "right", "left"),
  layout = list(),
  size = 1,
  buffer = 0.1,
  xname = if (side %in% c("top", "bottom")) current_xaxis(p) else NULL,
  yname = if (side %in% c("left", "right")) current_yaxis(p) else NULL,
  pname = "subplot"
)
```

Arguments

`p`	iheatmap object
`...`	arguments to pass to plotly trace, see plotly.js documentation at `https://plotly.com/javascript/reference/`
`side`	which side of the current plot to add this heatmap? "right", "left","top", or "bottom"
`layout`	axis layout parameters (list)
`size`	relative size of plot. size relative to first heatmap
`buffer`	amount of space to leave empty before this plot, relative to size of first heatmap
`xname`	internal name of xaxis
`yname`	internal name of yaxis
`pname`	internal name of plot

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`

Examples

```
mat <- matrix(rnorm(24), ncol = 6)
hm <- iheatmap(mat) %>% add_subplot(x = 1:5, y=1:5, side = "top")

# Print heatmap if interactive session
if (interactive()) hm
```

`iheatmap` *iheatmap*

Description

Make a farily standard interactive heatmap with optional clustering and row and column annotations. For more flexibility and options, see the `main_heatmap` function and other modular functions as described in vignette.

Usage

```
## S4 method for signature 'matrix'
iheatmap(
  data,
  x = default_x(data),
  y = default_y(data),
  cluster_rows = c("none", "hclust", "kmeans"),
  cluster_cols = c("none", "hclust", "kmeans"),
  row_clusters = NULL,
  col_clusters = NULL,
  row_k = NULL,
  col_k = NULL,
  row_clust_dist = stats::dist,
  col_clust_dist = stats::dist,
  name = "Signal",
  scale = c("none", "rows", "cols"),
  scale_method = c("standardize", "center", "normalize"),
  colors = NULL,
  col_clusters_colors = NULL,
  col_clusters_name = "Col<br>Clusters",
  row_clusters_colors = NULL,
  row_clusters_name = "Row<br>Clusters",
  show_row_clusters_colorbar = TRUE,
  show_col_clusters_colorbar = TRUE,
  row_annotation = NULL,
  col_annotation = NULL,
  row_annotation_colors = NULL,
  col_annotation_colors = NULL,
  row_labels = NULL,
  col_labels = NULL,
  row_title = NULL,
  col_title = NULL,
  colorbar_grid = setup_colorbar_grid(),
  layout = list(),
  source = "iheatmapr",
  ...
)
```

Arguments

`data`	matrix of values to be plotted as heatmap
`x`	x xaxis labels, by default colnames of data
`y`	y axis labels, by default rownames of data
`cluster_rows`	"none","hclust", or "k-means" for no clustering, hierarchical clustering, and k-means clustering of rows respectively
`cluster_cols`	"none","hclust", or "k-means" for no clustering, hierarchical clustering, and k-means clustering of columnsrespectively

`row_clusters`	vector of pre-determined row cluster assignment
`col_clusters`	vector of pre-determined column cluster assignment
`row_k`	number of clusters for rows, needed if cluster_rows is kmeans or optional if hclust
`col_k`	number of clusters for columns, needed if cluster_rows is kmeans or optional if hclust
`row_clust_dist`	distance function to use for row clustering if hierarchical clustering
`col_clust_dist`	distance function to use for column clustering if hierarchical clustering
`name`	Name for colorbar
`scale`	scale matrix by rows, cols or none
`scale_method`	what method to use for scaling, either none, standardize, center, normalize
`colors`	name of RColorBrewer palette or vector of colors for main heatmap
`col_clusters_colors`	colors for col clusters annotation heatmap
`col_clusters_name`	name for col clusters colorbar
`row_clusters_colors`	colors for row clusters annotation heatmap
`row_clusters_name`	name for row clusters colorbar
`show_row_clusters_colorbar`	show the colorbar for row clusters?
`show_col_clusters_colorbar`	show the colorbar for column clusters?
`row_annotation`	row annotation data.frame
`col_annotation`	column annotation data.frame
`row_annotation_colors`	list of colors for row annotations heatmap
`col_annotation_colors`	list of colors for col annotations heatmap
`row_labels`	axis labels for y axis
`col_labels`	axis labels for x axis
`row_title`	x axis title
`col_title`	y axis title
`colorbar_grid`	colorbar grid parameters, should be result from `setup_colorbar_grid`
`layout`	list of layout attributes to pass to plotly, eg. list(font = list(size = 15))
`source`	source name for use with shiny
`...`	additional argument to iheatmap

Details

By default, no scaling is done of rows or columns. This can be changed by specifying the ’scale’ argument. There are three options for scaling methods. "standardize" subtracts the mean and divides by standard deviation, "center" just subtracts the mean, and "normalize" divides by the sum of the values. "normalize" should only be used for data that is all positive! If alternative scaling is desired, the scaling should be done prior to calling the iheatmap function.

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`iheatmap`, `add_iheatmap`, `to_widget`

Examples

```
mat <- matrix(rnorm(24), nrow = 6)
annotation = data.frame(gender = c(rep("M", 3),rep("F",3)),
 age = c(20,34,27,19,23,30))
hm <- iheatmap(mat,
 cluster_rows = "hclust",
 cluster_cols = "kmeans",
 col_k = 3,
 row_annotation = annotation)

# Print heatmap if interactive session
if (interactive()) hm
```

`Iheatmap-class` *Iheatmap-class*

Description

Class to store complex interactive heatmap objects from iheatmapr package

Details

This is a virtual class with two children classes, IheatmapHorizontal and IheatmapVertical. For IheatmapHorizontal additional main heatmaps are added horizontally, and for IheatmapVertical additional main heatmaps are added vertically. For details on accessing certain slots of this class, see `access_component` documentation.

Slots

`plots`	list of plot element in `IheatmapPlots` format
`shapes`	list of shape element in `IheatmapShapes` format
`annotations`	list of annotation elements in `IheatmapAnnotations` format
`xaxes`	list of x axes in `IheatmapAxes` format
`yaxes`	list of y axes in `IheatmapAxes` format
`colorbars`	list of colorbars in `IheatmapColorbars` format
`colorbar_grid`	colorbar grid parameters in `IheatmapColorbarGrid` format
`current_xaxis`	name of current x axis
`current_yaxis`	name of current y axis
`layout`	list of plotly layout parameters
`source`	source name, for use with shiny

Author(s)

Alicia Schep

See Also

`iheatmap`, `main_heatmap`, `access_component`

`iheatmapr` *iheatmapr*

Description

Interactive complex heatmaps in R

Details

iheatmapr is a package for building complex, interactive heatmaps in R that can be explored in interactive R sessions or incorporated into rmarkdown documents, shiny applications, or standalone html files.

The package includes a modular system for building up complex heatmaps, where subplots get iteratively added to the top/left/right/bottom of the main heatmap(s). The `iheatmap` function provides a wrapper around many of the common modular subcomponents to build fairly standard, moderately complex heatmap.

See the vignette for detailed instructions for how to use the package.

iheatmapr uses the plotly javascript library (`https://plotly.com/`) for making the interactive figures and htmlwidgets (http://www.htmlwidgets.org/) for rendering them in R.

Author(s)

Alicia Schep

See Also

`main_heatmap`, `iheatmap`, `Iheatmap-class`

`iheatmapr-shiny` *Shiny bindings for iheatmapr*

Description

Output and render functions for using iheatmapr within Shiny

Usage

```
iheatmaprOutput(outputId, width = "100%", height = "400px")

renderIheatmap(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

`outputId`	output variable to read from
`width, height`	Must be a valid CSS unit (like `"100%"`, `"400px"`, `"auto"`) or a number, which will be coerced to a string and have `"px"` appended.
`expr`	An expression that generates an Iheatmap object
`env`	The environment in which to evaluate `expr`.
`quoted`	Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable.

`iheatmapr_event` *Access iheatmapr user input event data in shiny*

Description

This function must be called within a reactive shiny context.

Usage

```
iheatmapr_event(
  object,
  event = c("hover", "click", "relayout"),
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

`object`	`Iheatmap-class` object
`event`	The type of plotly event. Currently 'plotly_hover', 'plotly_click', 'plotly_selected', and 'plotly_relayout' are supported.
`session`	a shiny session object (the default should almost always be used).

Examples

```
## Not run:
shiny::runApp(system.file("examples", "shiny_example", package = "iheatmapr"))

## End(Not run)
```

`main_heatmap` *main_heatmap*

Description

Plots initial heatmap, creates Iheatmap object

Usage

```
## S4 method for signature 'matrix'
main_heatmap(
  data,
  name = "Signal",
  x = default_x(data),
  y = default_y(data),
  colors = pick_continuous_colors(zmid, zmin, zmax),
  colorbar_grid = setup_colorbar_grid(),
  colorbar_position = 1,
  zmid = 0,
  zmin = min(data, na.rm = TRUE),
  zmax = max(data, na.rm = TRUE),
  orientation = c("horizontal", "vertical"),
  x_categorical = NULL,
  y_categorical = NULL,
  row_order = seq_len(nrow(data)),
  col_order = seq_len(ncol(data)),
  text = signif(data, digits = 3),
  tooltip = setup_tooltip_options(),
  xname = "x",
  yname = "y",
  pname = name,
  source = "iheatmapr",
  show_colorbar = TRUE,
  layout = list()
)
```

Arguments

`data`	matrix
`name`	name of colorbar
`x`	x axis labels (by default rownames of data)
`y`	y axis labels (by default colnames of data)
`colors`	color palette or vector of colors
`colorbar_grid`	colorbar grid parameters, should be result from `setup_colorbar_grid`
`colorbar_position`	colorbar placement, should be positive integer
`zmid`	midpoint for colorscale
`zmin`	minimum for colorscale
`zmax`	maximum for colorscale
`orientation`	should new main plots be added horizontally or vertically?
`x_categorical`	is x categorical? will guess if not provided
`y_categorical`	is y categorical? will guess if not provided
`row_order`	row ordering for this heatmap– will be used for all subsequent elements sharing y axis
`col_order`	column ordering for this heatmap– will be used for all subsequent elements sharing x axis
`text`	text of value to display for data
`tooltip`	tooltip options, see `setup_tooltip_options`
`xname`	internal name for xaxis
`yname`	internal name for yaxis
`pname`	internal plot name
`source`	source name for use with shiny
`show_colorbar`	logical to indicate whether to show colorbar
`layout`	list of layout attributes to pass to plotly, eg. list(font = list(size = 15))

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_iheatmap`, `to_widget`, `iheatmap`, `Iheatmap-class`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- main_heatmap(mat)

# Print heatmap if interactive session
if (interactive()) hm
```

`measles` *measles*

Description

Data on measles cases for different states from 1930 to 2001

Examples

```
data(measles)
```

`modify_layout` *modify_layout*

Description

modify_layout

Usage

```
## S4 method for signature 'Iheatmap'
modify_layout(x, new_layout)
```

Arguments

`x`	Iheatmap
`new_layout`	list of new layout parameter

Value

modified Iheatmap object

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm <- main_heatmap(mat) %>% modify_layout(list(margin = list(b = 120)))

# Print heatmap if interactive session
if (interactive()) hm
```

`reorder_cols` *reorder_cols*

Description

Reorder the columns of an `Iheatmap-class` object

Usage

```
## S4 method for signature 'IheatmapHorizontal,integer'
reorder_cols(p, col_order, xname = current_xaxis(p))

## S4 method for signature 'IheatmapVertical,integer'
reorder_cols(p, col_order)
```

Arguments

`p`	`Iheatmap-class` object
`col_order`	integer vector
`xname`	name of xaxis to reorder, only applicable if object is oriented horizontally

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_clustering`, `reorder_cols`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
dend <- hclust(dist(t(mat)))
hm <- iheatmap(mat) %>% reorder_cols(dend$order)

# Print heatmap if interactive session
if (interactive()) hm
```

`reorder_rows` *reorder_rows*

Description

Reorder the rows of an `Iheatmap-class` object

Usage

```
## S4 method for signature 'IheatmapHorizontal,integer'
reorder_rows(p, row_order)

## S4 method for signature 'IheatmapVertical,integer'
reorder_rows(p, row_order, yname = current_yaxis(p))
```

Arguments

`p`	`Iheatmap-class` object
`row_order`	integer vector
`yname`	name of yaxis to reorder, only applicable if object is oriented vertically

Value

`Iheatmap-class` object, which can be printed to generate an interactive graphic

Author(s)

Alicia Schep

See Also

`add_row_clustering`, `reorder_cols`

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
dend <- hclust(dist(mat))
hm <- iheatmap(mat) %>% reorder_rows(dend$order)

# Print heatmap if interactive session
if (interactive()) hm
```

`save_iheatmap` *save_iheatmap*

Description

save an `link{Iheatmap-class}` object, either as standalone HTML or as static pdf/png/jpeg

Usage

```
## S4 method for signature 'Iheatmap,character'
save_iheatmap(p, filename, ...)
```

Arguments

`p`	`link{Iheatmap-class}` object
`filename`	name of file
`...`	additional arguments to `saveWidget` for saving as html or `webshot` for saving as pdf/png/jpeg

Details

Note that this function requires the webshot package. If deploying a shiny app that calls this function in shinyapps.io, loading the webshot library and calling `webshot::install_phantomjs()` is needed for the the save functionality to work.

Author(s)

Alicia Schep

Examples

```
mat <- matrix(rnorm(24), nrow = 6)
hm <- iheatmap(mat)
## Not run:
save_iheatmap(hm, "example_iheatmap.png")

## End(Not run)
```

`setup_colorbar_grid` *setup_colorbar_grid*

Description

function to set parameters controlling colorbar placement in Iheatmap object

Usage

```
setup_colorbar_grid(
  nrows = 3,
  y_length = y_spacing * 0.9,
  x_spacing = 0.16,
  y_spacing = y_start/nrows,
  x_start = 1.05,
  y_start = 0.9
)
```

Arguments

`nrows`	number of rows in colorbar grid
`y_length`	length of colorbar
`x_spacing`	spacing along horizontal axis between colorbars
`y_spacing`	spacing along vertical axis between colorbars
`x_start`	left most position of colorbar grid
`y_start`	top most position of colorbar grid

Value

`IheatmapColorbarGrid-class` object

Examples

```
cb_grid <- setup_colorbar_grid(nrows = 2, x_spacing = 0.2)
mat <- matrix(rnorm(24), nrow = 6)
hm <- iheatmap(mat, colorbar_grid = cb_grid, cluster_rows = "kmeans",
         cluster_cols = "kmeans", row_k = 3, col_k = 2)

# Print heatmap if interactive session
if (interactive()) hm
```

`setup_tooltip_options` *Tooltip Options*

Description

This function setups tooltip options for heatmap components of iheatmapr complex heatmaps.

Usage

```
setup_tooltip_options(
  row = TRUE,
  col = TRUE,
  value = TRUE,
  prepend_row = "Row: ",
  prepend_col = "Col: ",
  prepend_value = "Value: "
)
```

Arguments

`row`	logical, include row name in tooltip?
`col`	logical, include column name in tooltip?
`value`	logical, include value in tooltip?
`prepend_row`	text to prepend to row name
`prepend_col`	text to prepend to column name
`prepend_value`	text to prepend to value

Value

a HeatmapTooltipOptions object which stores these options and can be passed to 'tooltip' argument to main_heatmap and other functions.

Examples

```
mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
hm1 <- main_heatmap(mat,
   tooltip = setup_tooltip_options(row = FALSE, col = FALSE,
                                   prepend_value = "Value is "))

# Print heatmap if interactive session
if (interactive()) hm1
```

`test_iheatmapr_event` *test_iheatmapr_event*

Description

test_iheatmapr_event

Usage

```
test_iheatmapr_event(ihm, event = c("click", "hover", "relayout"))
```

Arguments

`ihm`	Iheatmap object
`event`	name of event, either "click","hover", or "relayout"

Value

shiny app

Examples

```

## Not run:
  mat <- matrix(rnorm(20), ncol = 5, nrow = 4)
  hm <- main_heatmap(mat)
  test_iheatmapr_event(hm, "click")

## End(Not run)
```

`to_plotly` *Convert Iheatmap to plotly spec*

Description

Function to convert `link{Iheatmap-class}` object to a plotly spec either as a list or json

Usage

```
to_plotly_list(p)

to_plotly_json(p)
```

Arguments

`p`	`Iheatmap-class` object to convert

Value

Returns a JSON for a plotly spec for to_plotly_spec and as a list of same plotly object for to_plotly_list.

Examples

```
mat <- matrix(rnorm(24), nrow = 6)
hm_json <- iheatmap(mat) %>% to_plotly_json()
hm_list <- iheatmap(mat) %>% to_plotly_list()
```

`to_widget` *to_widget*

Description

Function to convert `link{Iheatmap-class}` object to widget object

Usage

```
## S4 method for signature 'Iheatmap'
to_widget(p)
```

Arguments

`p`	`Iheatmap-class` object to convert

Value

htmlwidgets object

Author(s)

Alicia Schep

See Also

`iheatmap`, `main_heatmap`

Examples

```
mat <- matrix(rnorm(24), nrow = 6)
hm <- iheatmap(mat) %>% to_widget()
class(hm)

# Print heatmap if interactive session
if (interactive()) hm
```

Index