

# Package ‘pheatmap’

June 5, 2025

**Type** Package  
**Title** Pretty Heatmaps  
**Version** 1.0.13  
**Date** 2025-06-05  
**Depends** R (>= 2.0)  
**Description** Implementation of heatmaps that offers more control over dimensions and appearance.  
**Imports** grid, RColorBrewer, scales, gtable, stats, grDevices, graphics  
**License** GPL-2  
**Encoding** UTF-8  
**LazyLoad** yes  
**RoxygenNote** 7.3.2  
**NeedsCompilation** no  
**Author** Raivo Kolde [aut, cre]  
**Maintainer** Raivo Kolde <rkolde@gmail.com>  
**Repository** CRAN  
**Date/Publication** 2025-06-05 13:20:01 UTC

## Contents

pheatmap . . . . .	<a href="#">2</a>
<b>Index</b>	<b><a href="#">8</a></b>

---

pheatmap

*A function to draw clustered heatmaps.*


---

## Description

A function to draw clustered heatmaps where one has better control over some graphical parameters such as cell size, etc.

## Usage

```
pheatmap(
  mat,
  color = colorRampPalette(rev(brewer.pal(n = 7, name = "RdYlBu")))(100),
  kmeans_k = NA,
  breaks = NA,
  border_color = "grey60",
  cellwidth = NA,
  cellheight = NA,
  scale = "none",
  cluster_rows = TRUE,
  cluster_cols = TRUE,
  clustering_distance_rows = "euclidean",
  clustering_distance_cols = "euclidean",
  clustering_method = "complete",
  clustering_callback = identity2,
  cutree_rows = NA,
  cutree_cols = NA,
  treeheight_row = ifelse((inherits(cluster_rows, "hclust")) || cluster_rows, 50, 0),
  treeheight_col = ifelse((inherits(cluster_cols, "hclust")) || cluster_cols, 50, 0),
  legend = TRUE,
  legend_breaks = NA,
  legend_labels = NA,
  annotation_row = NA,
  annotation_col = NA,
  annotation = NA,
  annotation_colors = NA,
  annotation_legend = TRUE,
  annotation_names_row = TRUE,
  annotation_names_col = TRUE,
  drop_levels = TRUE,
  show_rownames = T,
  show_colnames = T,
  main = NA,
  fontsize = 10,
  fontsize_row = fontsize,
  fontsize_col = fontsize,
  angle_col = c("270", "0", "45", "90", "315"),
```

```

display_numbers = F,
number_format = "%.2f",
number_color = "grey30",
fontsize_number = 0.8 * fontsize,
gaps_row = NULL,
gaps_col = NULL,
labels_row = NULL,
labels_col = NULL,
filename = NA,
width = NA,
height = NA,
silent = FALSE,
na_col = "#DDDDDD",
...
)

```

### Arguments

<code>mat</code>	numeric matrix of the values to be plotted.
<code>color</code>	vector of colors used in heatmap.
<code>kmeans_k</code>	the number of kmeans clusters to make, if we want to aggregate the rows before drawing heatmap. If NA then the rows are not aggregated.
<code>breaks</code>	a sequence of numbers that covers the range of values in <code>mat</code> and is one element longer than <code>color</code> vector. Used for mapping values to colors. Useful, if needed to map certain values to certain colors, to certain values. If value is NA then the breaks are calculated automatically. When breaks do not cover the range of values, then any value larger than <code>max(breaks)</code> will have the largest color and any value lower than <code>min(breaks)</code> will get the lowest color.
<code>border_color</code>	color of cell borders on heatmap, use NA if no border should be drawn.
<code>cellwidth</code>	individual cell width in points. If left as NA, then the values depend on the size of plotting window.
<code>cellheight</code>	individual cell height in points. If left as NA, then the values depend on the size of plotting window.
<code>scale</code>	character indicating if the values should be centered and scaled in either the row direction or the column direction, or none. Corresponding values are "row", "column" and "none"
<code>cluster_rows</code>	boolean values determining if rows should be clustered or <code>hclust</code> object,
<code>cluster_cols</code>	boolean values determining if columns should be clustered or <code>hclust</code> object.
<code>clustering_distance_rows</code>	distance measure used in clustering rows. Possible values are "correlation" for Pearson correlation and all the distances supported by <a href="#">dist</a> , such as "euclidean", etc. If the value is none of the above it is assumed that a distance matrix is provided.
<code>clustering_distance_cols</code>	distance measure used in clustering columns. Possible values the same as for <code>clustering_distance_rows</code> .

clustering_method	clustering method used. Accepts the same values as <a href="#">hclust</a> .
clustering_callback	callback function to modify the clustering. Is called with two parameters: original hclust object and the matrix used for clustering. Must return a hclust object.
cutree_rows	number of clusters the rows are divided into, based on the hierarchical clustering (using cutree), if rows are not clustered, the argument is ignored
cutree_cols	similar to cutree_rows, but for columns
treeheight_row	the height of a tree for rows, if these are clustered. Default value 50 points.
treeheight_col	the height of a tree for columns, if these are clustered. Default value 50 points.
legend	logical to determine if legend should be drawn or not.
legend_breaks	vector of breakpoints for the legend.
legend_labels	vector of labels for the legend_breaks.
annotation_row	data frame that specifies the annotations shown on left side of the heatmap. Each row defines the features for a specific row. The rows in the data and in the annotation are matched using corresponding row names. Note that color schemes takes into account if variable is continuous or discrete.
annotation_col	similar to annotation_row, but for columns.
annotation	deprecated parameter that currently sets the annotation_col if it is missing
annotation_colors	list for specifying annotation_row and annotation_col track colors manually. It is possible to define the colors for only some of the features. Check examples for details.
annotation_legend	boolean value showing if the legend for annotation tracks should be drawn.
annotation_names_row	boolean value showing if the names for row annotation tracks should be drawn.
annotation_names_col	boolean value showing if the names for column annotation tracks should be drawn.
drop_levels	logical to determine if unused levels are also shown in the legend
show_rownames	boolean specifying if column names are be shown.
show_colnames	boolean specifying if column names are be shown.
main	the title of the plot
fontsize	base fontsize for the plot
fontsize_row	fontsize for rownames (Default: fontsize)
fontsize_col	fontsize for colnames (Default: fontsize)
angle_col	angle of the column labels, right now one can choose only from few predefined options (0, 45, 90, 270 and 315)
display_numbers	logical determining if the numeric values are also printed to the cells. If this is a matrix (with same dimensions as original matrix), the contents of the matrix are shown instead of original values.

number_format	format strings (C printf style) of the numbers shown in cells. For example "%.2f" shows 2 decimal places and "%.1e" shows exponential notation (see more in <a href="#">sprintf</a> ).
number_color	color of the text
fontsize_number	fontsize of the numbers displayed in cells
gaps_row	vector of row indices that show where to put gaps into heatmap. Used only if the rows are not clustered. See <a href="#">cutree_row</a> to see how to introduce gaps to clustered rows.
gaps_col	similar to <a href="#">gaps_row</a> , but for columns.
labels_row	custom labels for rows that are used instead of rownames.
labels_col	similar to <a href="#">labels_row</a> , but for columns.
filename	file path where to save the picture. Filetype is decided by the extension in the path. Currently following formats are supported: png, pdf, tiff, bmp, jpeg. Even if the plot does not fit into the plotting window, the file size is calculated so that the plot would fit there, unless specified otherwise.
width	manual option for determining the output file width in inches.
height	manual option for determining the output file height in inches.
silent	do not draw the plot (useful when using the <a href="#">gtable</a> output)
na_col	specify the color of the NA cell in the matrix.
...	graphical parameters for the text used in plot. Parameters passed to <a href="#">grid.text</a> , see <a href="#">gpar</a> .

## Details

The function also allows to aggregate the rows using kmeans clustering. This is advisable if number of rows is so big that R cannot handle their hierarchical clustering anymore, roughly more than 1000. Instead of showing all the rows separately one can cluster the rows in advance and show only the cluster centers. The number of clusters can be tuned with parameter `kmeans_k`.

## Value

Invisibly a pheatmap object that is a list with components

- `tree_row` the clustering of rows as [hclust](#) object
- `tree_col` the clustering of columns as [hclust](#) object
- `kmeans` the kmeans clustering of rows if parameter `kmeans_k` was specified
- `gtable` a [gtable](#) object containing the heatmap, can be used for combining the heatmap with other plots

## Author(s)

Raivo Kolde <[rkolde@gmail.com](mailto:rkolde@gmail.com)>

## Examples

```
# Create test matrix
test = matrix(rnorm(200), 20, 10)
test[1:10, seq(1, 10, 2)] = test[1:10, seq(1, 10, 2)] + 3
test[11:20, seq(2, 10, 2)] = test[11:20, seq(2, 10, 2)] + 2
test[15:20, seq(2, 10, 2)] = test[15:20, seq(2, 10, 2)] + 4
colnames(test) = paste("Test", 1:10, sep = "")
rownames(test) = paste("Gene", 1:20, sep = "")

# Draw heatmaps
pheatmap(test)
pheatmap(test, kmeans_k = 2)
pheatmap(test, scale = "row", clustering_distance_rows = "correlation")
pheatmap(test, color = colorRampPalette(c("navy", "white", "firebrick3"))(50))
pheatmap(test, cluster_row = FALSE)
pheatmap(test, legend = FALSE)

# Show text within cells
pheatmap(test, display_numbers = TRUE)
pheatmap(test, display_numbers = TRUE, number_format = "%.1e")
pheatmap(test, display_numbers = matrix(ifelse(test > 5, "x", ""), nrow(test)))
pheatmap(test, cluster_row = FALSE, legend_breaks = -1:4, legend_labels = c("0",
"1e-4", "1e-3", "1e-2", "1e-1", "1"))

# Fix cell sizes and save to file with correct size
pheatmap(test, cellwidth = 15, cellheight = 12, main = "Example heatmap")
# pheatmap(test, cellwidth = 15, cellheight = 12, fontsize = 8, filename = "test.pdf")

# Generate annotations for rows and columns
annotation_col = data.frame(
  CellType = factor(rep(c("CT1", "CT2"), 5)),
  Time = 1:5
)
rownames(annotation_col) = paste("Test", 1:10, sep = "")

annotation_row = data.frame(
  GeneClass = factor(rep(c("Path1", "Path2", "Path3"), c(10, 4, 6)))
)
rownames(annotation_row) = paste("Gene", 1:20, sep = "")

# Display row and color annotations
pheatmap(test, annotation_col = annotation_col)
pheatmap(test, annotation_col = annotation_col, annotation_legend = FALSE)
pheatmap(test, annotation_col = annotation_col, annotation_row = annotation_row)

# Change angle of text in the columns
pheatmap(test, annotation_col = annotation_col, annotation_row = annotation_row, angle_col = "45")
pheatmap(test, annotation_col = annotation_col, angle_col = "0")

# Specify colors
ann_colors = list(
  Time = c("white", "firebrick"),
```

```

    CellType = c(CT1 = "#1B9E77", CT2 = "#D95F02"),
    GeneClass = c(Path1 = "#7570B3", Path2 = "#E7298A", Path3 = "#66A61E")
  )

  pheatmap(test, annotation_col = annotation_col, annotation_colors = ann_colors, main = "Title")
  pheatmap(test, annotation_col = annotation_col, annotation_row = annotation_row,
            annotation_colors = ann_colors)
  pheatmap(test, annotation_col = annotation_col, annotation_colors = ann_colors[2])

  # Gaps in heatmaps
  pheatmap(test, annotation_col = annotation_col, cluster_rows = FALSE, gaps_row = c(10, 14))
  pheatmap(test, annotation_col = annotation_col, cluster_rows = FALSE, gaps_row = c(10, 14),
            cutree_col = 2)

  # Show custom strings as row/col names
  labels_row = c("", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",
                 "", "", "I110", "I115", "I11b")

  pheatmap(test, annotation_col = annotation_col, labels_row = labels_row)

  # Specifying clustering from distance matrix
  drows = dist(test, method = "minkowski")
  dcols = dist(t(test), method = "minkowski")
  pheatmap(test, clustering_distance_rows = drows, clustering_distance_cols = dcols)

  # Modify ordering of the clusters using clustering callback option
  callback = function(hc, mat){
    sv = svd(t(mat))$v[,1]
    dend = reorder(as.dendrogram(hc), wts = sv)
    as.hclust(dend)
  }

  pheatmap(test, clustering_callback = callback)

  ## Not run:
  # Same using dendsort package
  library(dendsort)

  callback = function(hc, ...){dendsort(hc)}
  pheatmap(test, clustering_callback = callback)

  ## End(Not run)

```

# Index

`dist`, [3](#)

`gpar`, [5](#)

`grid.text`, [5](#)

`gtable`, [5](#)

`hclust`, [4](#), [5](#)

`pheatmap`, [2](#)

`sprintf`, [5](#)