Package 'heatmaps'

July 16, 2025

Title Flexible Heatmaps for Functional Genomics and Sequence Features

Version 1.32.0

Date 2021-11-21

Author Malcolm Perry <mgperry32@gmail.com>

Maintainer Malcolm Perry <mgperry32@gmail.com>

Depends R (>= 3.4)

Imports methods, grDevices, graphics, stats, Biostrings, GenomicRanges, IRanges, KernSmooth, plotrix, Matrix, EBImage, RColorBrewer, BiocGenerics, GenomeInfoDb

Suggests BSgenome.Drerio.UCSC.danRer7, knitr, rmarkdown, testthat

biocViews Visualization, SequenceMatching, FunctionalGenomics

License Artistic-2.0

Description This package provides functions for plotting heatmaps of genome-wide data across genomic intervals, such as ChIP-seq signals at peaks or across promoters. Many functions are also provided for investigating sequence features.

VignetteBuilder knitr

Collate Heatmap-class.R PlotHeatmap.R PlotHeatmapList.R PlotPatternDensityMap.R PWMScanHeatmap.R PatternHeatmap.R CoverageHeatmap.R SmoothHeatmap.R PlotHeatmapMeta.R Data.R

NeedsCompilation no

RoxygenNote 6.0.1

git_url https://git.bioconductor.org/packages/heatmaps

git_branch RELEASE_3_21

git_last_commit 26ce830

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-07-16

Contents

`coords` *Return or set the coords in a Heatmap*

Description

Return or set the coords in a Heatmap

Usage

```
coords(x)

## S4 method for signature 'Heatmap'
coords(x)

coords(x) <- value
```

```
## S4 replacement method for signature 'Heatmap'
coords(x) <- value
```

Arguments

x	A heatmap
value	Replacement value

Value

integer, length 2, value of x@coords

Examples

```
data(HeatmapExamples)
coords(hm) = c(-100, 100)
```

CoverageHeatmap *Generate a Heatmap of coverage*

Description

Generate a Heatmap of coverage

Usage

```
CoverageHeatmap(windows, track, ...)

## S4 method for signature 'GenomicRanges,GenomicRanges'
CoverageHeatmap(windows, track,
  coords = NULL, weight = 1, label = NULL, nbin = 0)

## S4 method for signature 'GenomicRanges,RleList'
CoverageHeatmap(windows, track,
  coords = NULL, label = NULL, nbin = 0)
```

Arguments

windows	A set of GRanges of equal length
track	A GRanges or RleList object specifying coverage
...	additional arguments used by methods

This function generates a Heatmap object from a set of windows and an object containing genome-wide information about coverage. Either a GRanges or an RleList can be used. In the former case, the "weight" paramter is passed directly to the ‘coverage‘ function. If nbin is set, binned coverage is calculated which will save memory and time when plotting and average out varible data.

	If the coverage track contains negative values, then the scale will be centered on zero, ie. c(-max(abs(image(hm))), max(abs(image(hm)))). This makes more sense for most color schemes which are centered on zero, and avoids misleading plots where either positive or negative values are over-emphasised. See ?getScale for details. The scale can be manually reset if desired using the "scale" method.
`coords`	Co-ordinates for the heatmap, defaults to c(0, width(windows))
`weight`	Passed to coverage(track) constructor if class(track) == "GRanges"
`label`	Label for the heatmap
`nbin`	If set, number of bins to use across each window

Value

A Heatmap object

Methods (by class)

- `windows = GenomicRanges, track = GenomicRanges`: Heatmap of Coverage from 2 GRanges
- `windows = GenomicRanges, track = RleList`: Heatmap of Coverage from GRanges + RleList

Examples

```
data(HeatmapExamples)
CoverageHeatmap(windows, rle_list, coords=c(-100, 100), label="Example")
```

`default_color` *Predifined color palettes from RColorBrewer + Rainbow*

Description

Predifined color palettes from RColorBrewer + Rainbow

Usage

```
default_color(col)
```

Arguments

`col`	Character, RColorBrewer colorscheme or "Rainbow" This function provides a convenient function to all color palettes from RColorBrewer, and a better version of R's rainbow function (specifically rev(rainbow(9, start=0, end=4/6)), so it starts blue and ends with red).

Value

character, a length-9 color palette

Examples

```
default_color("Blues")
default_color("Rainbow")
```

`getScale` *Make an appropriate scale for a heatmap*

Description

Make an appropriate scale for a heatmap

Usage

```
getScale(x, y)
```

Arguments

`x`	Min/max values for the heatmap
`y`	Min/max values for the heatmap
	This function takes min/max values for a heatmap and generates a scale either starting, ending or centered on zero.

Value

numeric, length 2, a new scale

Examples

```
getScale(0.5, 5) # c(0, 5)
getScale(-6, -2) # c(-6, 6)
getScale(-6, 2) # c(-6, 6)
```

`Heatmap` *Function to create a heatmap object*

Description

Function to create a heatmap object

Usage

```
Heatmap(image, coords = NULL, label = "", nseq = NULL, scale = NULL,
  metadata = list())
```

Arguments

image	A numeric Matrix
coords	A length-2 integer vector
label	A character vector
nseq	An integer
scale	A length-2 vector
metadata	A list containing arbitrary metadata
	Using this function avoids calling 'new' directly or manually setting coords and nseq to integers. Other constructors exist for creating heatmaps from data, rather than a raw matrix.

Value

A Heatmap object

See Also

PatternHeatmap CoverageHeatmap PWMScanHeatmap

Examples

```
data(HeatmapExamples)
hm = Heatmap(mat, coords=c(-100, 100), label="Test")
```

Heatmap-class *An S4 class to represent a heatmap*

Description

An S4 class to represent a heatmap

Slots

image A numeric Matrix

scale A length-2 vector

coords A length-2 integer vector

nseq An integer

label A character vector

metadata A list containing arbitrary metadata

A class used to represent a heatmap in a simple, self-contained way

Slots can be accessed and set using getters and setters with the same name.

See Also

CoverageHeatmap PatternHeatmap plotHeatmap plotHeatmapMeta

Examples

```
data(HeatmapExamples)

hm = new("Heatmap",
         image=mat,
         scale=c(0,max(mat)),
         coords=c(-100L, 100L),
         nseq=1000L,
         label="Test",
         metadata=list())

# or use the constructor:
hm = Heatmap(mat, coords=c(-100, 100), label="Test")
```

`HeatmapExamples` *Data for man page examples*

Description

Generated Data for examples

An example heatmap

A second example heatmap

An example matrix

An example RleList

An example DNAStringSet

An example PWM

An example GRanges

Usage

```
hm

hm2

mat

rle_list

string_set

tata_pwm

windows
```

Format

An object of class `Heatmap` of length 500.

Value

invisible("HeatmapExamples")

`heatmapOptions` *Generate default options for a Heatmap*

Description

Generate default options for a Heatmap

Usage

```
heatmapOptions(...)
```

Arguments

`...`	options to set manually Guide to Heatmap options This is an reference to all the possible options for plotting heatmaps. Some options are handled by heatmaps functions (either plotHeatmap or plotHeatmapList), others are passed directly to plotting functions. Further explanation is available in the vignette. Arguments are numeric if not otherwise stated. color: A vector of colors or a default color, see ?default_color. plotHeatmap will interpolate between these colors to form a scale. box.width: width of box around the heatmap, passed to box() x.ticks: Logical, plot x axis ticks x.tick.labels: Character, labels to use for x ticks, (default blank) tcl: Length of x axis ticks padj: Vertical adjustment of x axis labels cex.axis: cex for axis labels scale: Logical, Plot scale or not scale.label: Character, label for scale scale.lwd: Width for line around scale cex.scale: Cex for Scale label: Logical, plot label or not label.xpos: x position for label, from left label.ypos: y position for label, from top cex.label: cex for axis labels label.col: Color for label, white is often useful for dark plots legend: Logical, plot legend (scale indicating values for colors)

legend: Color for label, white is often useful for dark plots

legend.pos: Character, position of legend relative to heatmap: 'l' for left, 'r' for right

legend.ticks: Number of ticks to use on legend.

cex.legend: cex to use for legend marks

refline: Logical, Draw dashed line at coords = 0 label: Logical, plot label or not

label.xpos: x position for label, from left

label.ypos: y position for label, from top

cex.label: cex for axis labels

label.col: Color for label, white is often useful for dark plots

legend: Logical, plot legend (scale indicating values for colors)

legend: Color for label, white is often useful for dark plots

legend.pos: Character, position of legend relative to heatmap: 'l' for left, 'r' for right

legend.ticks: Number of ticks to use on legend.

cex.legend: cex to use for legend marks

refline: Logical, Draw dashed line at coords = 0

refline.width: Width of reference line

transform: Function to transform values before plotting

plot.mai: Length-4 numeric, margins around plot

legend.mai: Length-4 numeric, margins around legend

partition: Numeric, relative sizes of clusters

partition.lines: Logical, plot lines delineating clusters

partition.legend: Logical, plot cluster legend in HeatmapList

partition.col: Character, colours to use for plotting clusters. Defaults to RColorBrewer's Set1

hook: Function called after plotting is complete.

Value

a list containing the specified options

See Also

plotHeatmap plotHeatmapList

Examples

```
myOptions = heatmapOptions()
myOptions$color = "Reds"
# plotHeatmap(hm, options=myOptions)
```

image *Return or set the image in a Heatmap*

Description

Return or set the image in a Heatmap

Usage

```
## S4 method for signature 'Heatmap'
image(x)

image(x) <- value

## S4 replacement method for signature 'Heatmap'
image(x) <- value
```

Arguments

x	A heatmap
value	Replacement value

Value

matrix, from hm@image

Examples

```
data(HeatmapExamples)
image(hm) = log(image(hm))
scale(hm) = c(0, max(image(hm)))
```

label *Return or set the label in a Heatmap*

Description

Return or set the label in a Heatmap

Usage

```
label(x)

## S4 method for signature 'Heatmap'
label(x)

label(x) <- value

## S4 replacement method for signature 'Heatmap'
label(x) <- value
```

Arguments

x	A heatmap
value	Replacement value

Value

character, value of hm@label

Examples

```
data(HeatmapExamples)
label(hm) = "NewLabel"
label(hm) # "NewLabel"
```

---

length,Heatmap-method *Return the number of sequences in a heatmap*

Description

Return the number of sequences in a heatmap

Usage

```
## S4 method for signature 'Heatmap'
length(x)
```

Arguments

x	A heatmap

Value

integer, value of x@nseq

metadata *Return or set the metadata in a Heatmap*

Description

Store arbitrary metadata in a list, if desired.

Usage

```
metadata(x)

## S4 method for signature 'Heatmap'
metadata(x)

metadata(x) <- value

## S4 replacement method for signature 'Heatmap'
metadata(x) <- value
```

Arguments

x	A heatmap
value	Replacement value

Value

list, value of hm@metadata

Examples

```
data(HeatmapExamples)
metadata(hm) = list(replicate=1, cell_line="ESC")
metadata(hm)$replicate == 1
```

nseq *Return or set nseq in a Heatmap*

Description

Return or set nseq in a Heatmap

Usage

```
nseq(x)

## S4 method for signature 'Heatmap'
nseq(x)

nseq(x) <- value

## S4 replacement method for signature 'Heatmap'
nseq(x) <- value
```

Arguments

x	A heatmap
value	Replacement value

Value

integer, value of hm@nseq

Examples

```
data(HeatmapExamples)
nseq(hm) = 1000
```

PatternHeatmap *Generate a Heatmap of patterns in DNA sequence*

Description

Generate a Heatmap of patterns in DNA sequence

Usage

```
PatternHeatmap(seq, pattern, ...)

## S4 method for signature 'DNAStringSet,character'
PatternHeatmap(seq, pattern, coords = NULL,
  min.score = NULL, label = NULL)

## S4 method for signature 'DNAStringSet,matrix'
PatternHeatmap(seq, pattern, coords = NULL,
  min.score = "80%", label = NULL)
```

Arguments

seq	A DNAString of equal length
pattern	A nucleotide pattern or PWM
...	additional arguments used by methods This function creates a Heatmap from a set of DNA sequences. The resulting heatmap will be binary, with 1 representing a match and 0 otherwise. Patterns can be specified as a character vectore, eg. "CTCCC", or as a PWM. These arguments are passed to Biostrings functions, ‘vmatchPattern‘ and ‘matchPWM‘. Character arguments can contain standard ambiguity codes. PWMs must be 4 by n matricies with columns names ACGT. "min.score" is specified either as an absolute value, or more commonly as a percentage e.g. "80 for details. PatternHeatmaps often look much better after smoothing.
coords	Co-ordinates for the heatmap, defaults to c(0, width(windows))
min.score	Minimum score for PWM match
label	Label for the heatmap

Value

A heatmap

Methods (by class)

- seq = DNAStringSet,pattern = character: Heatmap of sequence patterns from sequence and character
- seq = DNAStringSet,pattern = matrix: Heatmap of sequence patterns from sequence and matrix

See Also

smoothHeatmap

Examples

```
data(HeatmapExamples)
PatternHeatmap(string_set, "TA", coords=c(-100, 100), label="TA")
PatternHeatmap(string_set, tata_pwm, coords=c(-100, 100), min.score="80%", label="TATA PWM")
```

---

plotHeatmap *Plot a Heatmap object to the device*

Description

Plot a Heatmap object to the device

Usage

```
plotHeatmap(heatmap, options = NULL, ...)

## S4 method for signature 'Heatmap'
plotHeatmap(heatmap, options = NULL, ...)
```

Arguments

heatmap	A heatmap object
options	A list containing plotting options
...	Used for passing individual options

This function will take a heatmap and plot it to the device with the specified options. Options can be passed together in a list or individually as additional arguments. If passing options as a list, it's best to first create a list containing the default settings using heatmapOptions() andmethod then setting options individually.

plotHeatmap() does not control device settings at all, these can be set using plotHeatmapList() and the relevant options in heatmapOptions()

See ?heatmapOptions for a full list of options.

Value

invisible(0)

Methods (by class)

- Heatmap: Plot a Heatmap object to the device

See Also

heatmapOptions plotHeatmapList

Examples

```
data(HeatmapExamples)
plotHeatmap(hm, color="Blues")
```

---

plotHeatmapList *Plot a list of heatmaps*

Description

Plot a list of heatmaps

Usage

```
plotHeatmapList(heatmap_list, groups = 1:length(heatmap_list),
  options = heatmapOptions(), ...)
```

Arguments

heatmap_list	A list of Heatmaps
groups	Optionally group heatmaps together
options	Heatmap options
...	Additional options

This function takes a list of one or more heatmaps and plots them to a single image tiled horizontally.

The "groups" argument specifies heatmaps to be grouped together and plotted using the same display parameters and a unified scale. plotHeatmapList will try to guess the best scale, either starting or finishing at zero, or symetrical aronud zero - if this is not the desired behaviour, make sure the scales are identical before the heatmaps are passed to the function.

Options are specified as for plotHeatmap, but can be specified per group by passing a list of options instead of a single vector. Note the difference between a length-2 character vector, c("Reds", "Blues"), and a list contatining two length-1 character vectors: list("Reds", "Blues").

These are generally large, complex plots, so it can better to plot straight to a file. PNG is preferred since pdf files generated can be if the images are not downsized. The default settings are designed for plots of about 10cm x 20cm per heatmap, but all of the relevant settigns can be tweaked using the options. For display-quality images, it helps to increase the resolution at to at least 150ppi, double the default of 72ppi on most systems.

Value

invisible(0)

See Also

plotHeatmap heatmapOptions plot_legend

Examples

```
data(HeatmapExamples)
plotHeatmapList(list(hm, hm2), groups=c(1,2), color=list("Reds", "Blues"))
```

---

plotHeatmapMeta *Plot a Meta-region plot from heatmaps*

Description

Plot a Meta-region plot from heatmaps

Usage

```
plotHeatmapMeta(hm_list, binsize = 1, colors = gg_col(length(hm_list)),
  addReferenceLine = FALSE)
```

Arguments

hm_list	A list of heatmaps
binsize	Integer, size of bins to use in plot
colors	Color to use for each heatmap
addReferenceLine	Logical, add reference line at zero or not

This function creates a meta-region plot from 1 or more heatmaps with the same coordinates. A meta-region plot graphs the sum of the signal at each position in each heatmap rather than visualising the signal in two dimensions. Often binning is required to smooth noisy signal.

Value

invisible(0)

Examples

```
data(HeatmapExamples)
plotHeatmapMeta(hm, color="steelblue")
```

---

plotPatternDensityMap *Plot heatmaps for several patterns in DNA sequence*

Description

Plot heatmaps for several patterns in DNA sequence

Usage

```
plotPatternDensityMap(seq, patterns, ...)

## S4 method for signature 'DNAStringSet'
plotPatternDensityMap(seq, patterns, coords = NULL,
  min.score = "80%", sigma = c(3, 3), output.size = NULL,
  options = NULL, ...)
```

Arguments

seq	DNAStringSet of equal width
patterns	A vector or list of patterns
...	Additional Heatmap plotting options

This function is a convenient wrapper for plotting many different patterns for the same set of sequences. PatternHeatmap() is applied to the sequence for each pattern in the list, they are passed to smoothHeatmap() with the supplied parameters and finally PlotHeatmapList().

If fine-grained control is desired, or you want to mix other plot types, then more information is available in the vignette.

coords	Heatmap coords
min.score	Minimum score for PWM match
sigma	Bandwith for smoothing kernel
output.size	Output size of final image
options	Heatmap plotting options

Value

invisible(0)

Methods (by class)

- DNAStringSet: Plot heatmaps for several patterns in DNA sequence

See Also

PatternHeatmap plotHeatmapList smoothHeatmap

Examples

```
data(HeatmapExamples)
plotPatternDensityMap(string_set, c("AT", "CG"), coords=c(-200, 200))
```

---

plot_clusters *Plot partition in a separate panel*

Description

Plot partition in a separate panel

Usage

```
plot_clusters(options)
```

Arguments

options	heatmapOptions passed as a list Two heatmapOptions values are relevant: * partition Numeric vector containing relative sizes of the clusters * colors Colors to use for clusters, additional colors are discarded This function plots a vertical color scale (or legend). With the default parameters, it looks good at about 1/5 the width of a heatmap, about 1cm x 10cm. This function only plots the legend, it does not set margin parameters.

Value

invisible(0)

See Also

plotHeatmapList

Examples

```
data(HeatmapExamples)
opts = heatmapOptions()
opts$partition = c(1,2,3,4)
par(mai=opts$legend.mai)
plot_clusters(opts)
```

`plot_legend` *Plot a color legend for a heatmap*

Description

Plot a color legend for a heatmap

Usage

```
plot_legend(scale, options)
```

Arguments

`scale`	Numeric vector contain min and max for the scale
`options`	heatmapOptions passed as a list
	This function plots a vertical color scale (or legend). With the default parameters, it looks good at about 1/5 the width of a heatmap, about 1cm x 10cm. This function only plots the legend, it does not set margin parameters.

Value

invisible(0)

See Also

plotHeatmapList

Examples

```
data(HeatmapExamples)
opts = heatmapOptions()
opts$color = "Rainbow"
par(mai=opts$legend.mai)
plot_legend(c(0,1), opts)
```

PWMScanHeatmap *Generate a Heatmap of PWM Scores in DNA sequnce*

Description

Generate a Heatmap of PWM Scores in DNA sequnce

Usage

```
PWMScanHeatmap(seq, pwm, ...)

## S4 method for signature 'DNAStringSet,matrix'
PWMScanHeatmap(seq, pwm, coords = NULL,
  label = "")
```

Arguments

seq	A DNAString of equal length
pwm	A PWM
...	additional arguments used by methods
	This function creates a heatmap where each point is the score of a PWM match starting from that position, which can visualise regions of enrichment or exclusion of certain motifs
coords	Co-ordinates for the heatmap, defaults to c(0, width(windows))
label	Label for the heatmap

Value

A heatmap

Methods (by class)

- seq = DNAStringSet,pwm = matrix: Heatmap of PWM Scores

See Also

PatternHeatmap

Examples

```
data(HeatmapExamples)
PatternHeatmap(string_set, tata_pwm, coords=c(-100, 100), label="TATA Scan")
```

rev,Heatmap-method *Reflect a heatmap in the x axis*

Description

Reflect a heatmap in the x axis

Usage

```
## S4 method for signature 'Heatmap'
rev(x)
```

Arguments

x	A heatmap

Value

A heatmap

scale *Return or set the scale in a Heatmap*

Description

Return or set the scale in a Heatmap

Usage

```
scale(x)

## S4 method for signature 'Heatmap'
scale(x)

scale(x) <- value

## S4 replacement method for signature 'Heatmap'
scale(x) <- value
```

Arguments

x	A heatmap
value	Replacement value

Value

numeric, length 2, the value of hm@scale

Examples

```
data(HeatmapExamples)
scale(hm) = c(-1000, 1000)
```

`smoothHeatmap` *Smooth a heatmap*

Description

Smooth a heatmap

Usage

```
smoothHeatmap(heatmap, ...)

## S4 method for signature 'Heatmap'
smoothHeatmap(heatmap, sigma = c(3, 3),
  output.size = dim(image(heatmap)), algorithm = NULL)
```

Arguments

`heatmap`	A heatmap object
`...`	additional arguments to S4 methods This function smooths a heatmap using either binned kernel density (more efficient for binary heatmaps) or gaussian blur. Sigma controls the SD of the kernel in both cases, defined in terms of pixels. This means that if you have very diffirent x and y dimensions (eg. a 200bp heatmap around 10000 promoters) you will need to compensate by setting sigma[2] higher to get the same visual effect in both dimensions "output.size" specifies the dimensions of the output matrix. This can be useful to reduce plotting time significantly. Smoothing can use either a kernel density estimate or a blurring function. The methods implemented are KernSmooth:bkde2D and EBImage::filter2 with a gaussian filter. The kernel based method assumes we are smoothing individual points so the value of these points are ignored. This is most useful for smoothing PatternHeatmaps where each cell in the matrix is either 1 or 0. For non-binary heatmaps, blur is most appropriate. Not setting this parameter will choose the method automatically. Scaling the output heatmap is handled as in CoverageHeatmap.
`sigma`	Numeric, lengt2, (recycled if length 1)
`output.size`	Numeric, length 2
`algorithm`	"kernel" or "blur"

Value

A heatmap

Methods (by class)

- Heatmap: Smooth a heatmap

Examples

```
data(HeatmapExamples)
hm_smoothed = smoothHeatmap(hm, sigma=c(5,5), algorithm="blur")
```

---

width,Heatmap-method *Return the width of sequence represented in a heatmap*

Description

Return the width of sequence represented in a heatmap

Usage

```
## S4 method for signature 'Heatmap'
width(x)
```

Arguments

x	A heatmap

Value

integer

---

xm *Generate co-ordinates for each row of the image matrix of a Heatmap*

Description

Generate co-ordinates for each row of the image matrix of a Heatmap

Usage

```
xm(x)

## S4 method for signature 'Heatmap'
xm(x)
```

Arguments

x	A Heatmap

Value

numeric, a list of co-ordinates for plotting values in hm@image

Methods (by class)

- Heatmap: Generate co-ordinates for each frow of the image matrix of a Heatmap

Examples

```
data(HeatmapExamples)
xm(hm)
```

---

ym *Generate co-ordinates for each column of the image matrix of a Heatmap*

Description

Generate co-ordinates for each column of the image matrix of a Heatmap

Usage

```
ym(x)

## S4 method for signature 'Heatmap'
ym(x)
```

Arguments

x	A Heatmap

Value

numeric, a list of co-ordinates for plotting values in hm@image

Methods (by class)

- Heatmap: Generate co-ordinates for each column of the matrix

Examples

```
data(HeatmapExamples)
ym(hm)
```

Index