

# Package ‘gtrellis’

October 16, 2019

**Type** Package

**Title** Genome Level Trellis Layout

**Version** 1.16.1

**Date** 2019-08-22

**Author** Zuguang Gu

**Maintainer** Zuguang Gu <z.gu@dkfz.de>

**Depends** R (>= 3.1.2), grid, IRanges, GenomicRanges

**Imports** circlize (>= 0.4.8), GetoptLong, grDevices, utils

**Suggests** testthat (>= 1.0.0), knitr, RColorBrewer, markdown,  
ComplexHeatmap (>= 1.99.0), Cairo, png, jpeg, tiff

**VignetteBuilder** knitr

**Description** Genome level Trellis graph visualizes genomic data conditioned by genomic categories (e.g. chromosomes). For each genomic category, multiple dimensional data which are represented as tracks describe different features from different aspects. This package provides high flexibility to arrange genomic categories and to add self-defined graphics in the plot.

**biocViews** Software, Visualization, Sequencing

**URL** <https://github.com/jokergoo/gtrellis>

**License** MIT + file LICENSE

**git\_url** <https://git.bioconductor.org/packages/gtrellis>

**git\_branch** RELEASE\_3\_9

**git\_last\_commit** a9003ed

**git\_last\_commit\_date** 2019-09-13

**Date/Publication** 2019-10-15

## R topics documented:

add_heatmap_track . . . . .	2
add_ideogram_track . . . . .	3
add_lines_track . . . . .	4
add_points_track . . . . .	4
add_rect_track . . . . .	5
add_segments_track . . . . .	6

add_track . . . . .	7
current_track . . . . .	9
get_cell_meta_data . . . . .	9
gtrellis_layout . . . . .	11
gtrellis_show_index . . . . .	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

add_heatmap_track	<i>add heatmap to a new track</i>
-------------------	-----------------------------------

---

## Description

add heatmap to a new track

## Usage

```
add_heatmap_track(gr, mat, fill, border = NA, track = current_track() + 1, ...)
```

## Arguments

gr	genomic regions, it can be a data frame or a <a href="#">GRanges</a> object
mat	matrix in which rows correspond to intervals in gr
fill	a color mapping function which maps values to colors. Users can consider <a href="#">colorRamp2</a> to generate a color mapping function.
border	border of the grids in heatmap
track	which track the graphics will be added to. By default it is the next track. The value should only be a scalar.
...	other arguments passed to <a href="#">add_track</a>

## Value

No value is returned.

## Author(s)

Zuguang Gu <z.gu@dkfz.de>

## See Also

[add\\_rect\\_track](#), [add\\_track](#)

## Examples

```
require(circlize)
bed = generateRandomBed(200)
col_fun = colorRamp2(c(-1, 0, 1), c("green", "black", "red"))
gtrellis_layout(nrow = 3, byrow = FALSE, track_axis = FALSE)
mat = matrix(rnorm(nrow(bed)*4), ncol = 4)
add_heatmap_track(bed, mat, fill = col_fun)
```

---

add_ideogram_track	<i>Add ideogram track</i>
--------------------	---------------------------

---

## Description

Add ideogram track

## Usage

```
add_ideogram_track(cytoband = system.file("extdata", "cytoBand.txt",  
  package = "circlize"), species = NULL, track = current_track() + 1)
```

## Arguments

cytoband	Path of the cytoband file or a data frame that already contains cytoband data. Pass to <a href="#">read.cytoband</a> .
species	Abbreviations of species. e.g. hg19 for human, mm10 for mouse. If this value is specified, the function will download cytoBand.txt.gz from UCSC ftp automatically. Pass to <a href="#">read.cytoband</a> .
track	which track the ideogram is added in. By default it is the next track in the layout.

## Details

A track which contains ideograms will be added to the plot.

The function tries to download cytoband file from UCSC ftp. If there is no cytoband file available for the species, there will be an error.

## Value

No value is returned.

## Author(s)

Zuguang Gu <z.gu@dkfz.de>

## Examples

```
gtrellis_layout(n_track = 2, ncol = 3,  
  track_height = unit.c(unit(1, "null"), unit(5, "mm")))  
add_ideogram_track(track = 2)
```

---

add\_lines\_track      *add lines to a new or exsited track*

---

**Description**

add lines to a new or exsited track

**Usage**

```
add_lines_track(gr, value, area = FALSE, baseline = "bottom", gp = gpar(), ...)
```

**Arguments**

gr	genomic regions, it can be a data frame or a <a href="#">GRanges</a> object
value	numeric values associated with gr
area	whether draw polygon for the area under the line
baseline	baseline for drawing polygon
gp	graphic settings, should be specified by <a href="#">gpar</a> .
...	other arguments passed to <a href="#">add_track</a>

**Value**

No value is returned.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**Examples**

```
require(circlize)
bed = generateRandomBed(200)
gtrellis_layout(n_track = 2, track_ylim = rep(range(bed[[4]]), 2), nrow = 3, byrow = FALSE)
add_lines_track(bed, bed[[4]])
add_lines_track(bed, bed[[4]], area = TRUE, gp = gpar(fill = "grey", col = NA))
```

---

add\_points\_track      *add points to a new or exsited track*

---

**Description**

add points to a new or exsited track

**Usage**

```
add_points_track(gr, value, pch = 16, size = unit(1, "mm"), gp = gpar(), ...)
```

**Arguments**

gr	genomic regions, it can be a data frame or a <a href="#">GRanges</a> object
value	numeric values associated with gr
pch	shape of points
size	size of points, should be a <a href="#">unit</a> object
gp	graphic settings, should be specified by <a href="#">gpar</a> .
...	other arguments passed to <a href="#">add_track</a>

**Value**

No value is returned.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**Examples**

```
require(circlize)
bed = generateRandomBed()
gtrellis_layout(track_ylim = range(bed[[4]]), nrow = 3, byrow = FALSE)
add_points_track(bed, bed[[4]], gp = gpar(col = ifelse(bed[[4]] > 0, "red", "green")))
```

---

add_rect_track	<i>add rectangles to a new or exsited track</i>
----------------	---

---

**Description**

add rectangles to a new or exsited track

**Usage**

```
add_rect_track(gr, h1, h2, gp = gpar(), ...)
```

**Arguments**

gr	genomic regions, it can be a data frame or a <a href="#">GRanges</a> object
h1	top/bottom positions for rectangles
h2	top/bottom positions for rectangles
gp	graphic settings, should be specified by <a href="#">gpar</a> .
...	other arguments passed to <a href="#">add_track</a>

**Value**

No value is returned.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**See Also**

[add\\_heatmap\\_track](#), [add\\_track](#)

**Examples**

```
require(circlize)
bed = generateRandomBed(200)
col_fun = colorRamp2(c(-1, 0, 1), c("green", "black", "red"))
gtrellis_layout(track_ylim = range(bed[[4]]), nrow = 3, byrow = FALSE)
add_rect_track(bed, h1 = bed[[4]], h2 = 0,
  gp = gpar(col = NA, fill = col_fun(bed[[4]])))
```

---

add_segments_track	<i>add segments to a new or exsited track</i>
--------------------	---

---

**Description**

add segments to a new or exsited track

**Usage**

```
add_segments_track(gr, value, gp = gpar(), ...)
```

**Arguments**

gr	genomic regions, it can be a data frame or a <a href="#">GRanges</a> object
value	numeric values associated with gr
gp	graphic settings, should be specified by <a href="#">gpar</a> .
...	other arguments passed to <a href="#">add_track</a>

**Value**

No value is returned.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**Examples**

```
require(circlize)
bed = generateRandomBed(nr = 100)
gtrellis_layout(track_ylim = range(bed[[4]]), nrow = 3, byrow = FALSE)
add_segments_track(bed, bed[[4]], gp = gpar(col = ifelse(bed[[4]] > 0, "red", "green"), lwd = 4))
```

---

add\_track

*Add self-defined graphics track by track*


---

## Description

Add self-defined graphics track by track

## Usage

```
add_track(gr = NULL, category = NULL, track = current_track() + 1,
          clip = TRUE, panel_fun = function(gr) NULL, panel.fun = NULL,
          use_raster = FALSE,
          raster_device = c("png", "jpeg", "tiff", "CairoPNG", "CairoJPEG", "CairoTIFF"),
          raster_quality = 1,
          raster_device_param = list())
```

## Arguments

gr	genomic regions. It should be a data frame in BED format or a GRanges object.
category	subset of categories (e.g. chromosomes) that users want to add graphics. The value can be a vector which contains more than one category. By default it is all available categories.
track	which track the graphics will be added to. By default it is the next track. The value should only be a scalar.
clip	whether graphics are restricted inside the cell.
panel_fun	self-defined panel function to add graphics in each 'cell'. The argument gr in panel_fun only contains data for the current category which is a subset of the main gr. The function can also contains no argument if nothing needs to be passed in.
panel.fun	deprecated
use_raster	whether render the each panel as a raster image. It helps to reduce file size when the file size is huge.
raster_device	graphic device which is used to generate the raster image
raster_quality	a value set to larger than 1 will improve the quality of the raster image. A temporary image with raster_quality*raster_quality times the original size of panel is generated first and then fit into the panel by <a href="#">grid.raster</a> .
raster_device_param	a list of further parameters for the selected graphic device

## Details

Initialization of the Trellis layout and adding graphics are two independent steps. Once the layout initialization finished, each cell will be an independent plotting region. As same as panel\_fun in [circlize-package](#), the self-defined function panel\_fun will be applied on every cell in the specified track (by default it is the 'current' track).

When adding graphics in each cell, [get\\_cell\\_meta\\_data](#) can return several meta data for the current cell.

Since this package is implemented by the grid graphic system, grid-family functions (such as [grid.points](#), [grid.rect](#), ...) should be used to add graphics. The usage of grid functions is quite similar as the traditional graphic functions. Followings are several examples:

```
grid.points(x, y)
grid.lines(x, y)
grid.rect(x, y, width, height)
```

Graphical parameters are usually passed by [gpar](#):

```
grid.points(x, y, gp = gpar(col = "red")
grid.rect(x, y, width, height, gp = gpar(fill = "black", col = "red"))
```

grid system also support a large number of coordinate measurement systems by defining proper [unit](#) object which provides high flexibility to place graphics on the plotting regions.

```
grid.points(x, y, default.units = "npc")
grid.rect(x, y, width = unit(1, "cm"))
```

You can refer to the documentations and vignettes of [grid-package](#) to get a overview.

### Value

No value is returned.

### Author(s)

Zuguang Gu <z.gu@dkfz.de>

### See Also

There are several functions which draw specific graphics and are implemented by [add\\_track](#):

- [add\\_points\\_track](#)
- [add\\_segments\\_track](#)
- [add\\_lines\\_track](#)
- [add\\_rect\\_track](#)
- [add\\_heatmap\\_track](#)

### Examples

```
require(circlize)
bed = circlize::generateRandomBed()
gtrellis_layout(track_ylim = range(bed[[4]]))
add_track(bed, panel.fun = function(bed) {
  x = (bed[[2]] + bed[[3]]) / 2
  y = bed[[4]]
  grid.points(x, y, pch = 16, size = unit(0.5, "mm"))
})

# you can add graphics in any cell by specifying `category` and `track`
all_chr = paste0("chr", 1:22)
letter = strsplit("MERRY CHRISTMAS!", "")[[1]]
```



```

gtrellis_layout(nrow = 5)
for(i in seq_along(letter)) {
  add_track(category = all_chr[i], track = 1, panel.fun = function(gr) {
    grid.text(letter[i], gp = gpar(fontsize = 30))
  })
}

```

---

current_track	<i>The index of current track</i>
---------------	-----------------------------------

---

### Description

The index of current track

### Usage

```
current_track()
```

### Value

No value is returned.

### Author(s)

Zuguang Gu <z.gu@dkfz.de>

### Examples

```
# There is no example
NULL
```

---

get_cell_meta_data	<i>Get meta data in a cell</i>
--------------------	--------------------------------

---

### Description

Get meta data in a cell

### Usage

```
get_cell_meta_data(name, category, track)
```

### Arguments

name	name of the supported meta data, see 'details' section.
category	which category. By default it is the current category.
track	which track. By default it is the current track.

**Details**

Following meta data can be retrieved:

`name` name of the category.

`xlim` xlim without including padding. Cells in the same column share the same xlim.

`ylim` ylim without including padding.

`extended_xlim` xlim with padding.

`extended_ylim` ylim with padding.

`original_xlim` xlim in original data.

`original_ylim` ylim in original data.

`column` which column in the layout.

`row` which row in the layout.

`track` which track in the layout.

The vignette has a graphical explanation of all these meta data.

**Value**

Corresponding meta data that user queried.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**Examples**

```
gtrellis_layout(ncol = 4, n_track = 3)
add_track(panel.fun = function(gr) {
  print(get_cell_meta_data("xlim"))
  print(get_cell_meta_data("ylim"))
  print(get_cell_meta_data("extended_xlim"))
  print(get_cell_meta_data("extended_ylim"))
  print(get_cell_meta_data("original_xlim"))
  print(get_cell_meta_data("original_ylim"))
  print(get_cell_meta_data("name"))
  print(get_cell_meta_data("column"))
  print(get_cell_meta_data("row"))
  print(get_cell_meta_data("track"))
  cat("\n\n")
})

for(chr in paste0("chr", 1:22)) {
  print(get_cell_meta_data("xlim", category = chr, track = 1))
  print(get_cell_meta_data("ylim", category = chr, track = 1))
  print(get_cell_meta_data("extended_xlim", category = chr, track = 1))
  print(get_cell_meta_data("extended_ylim", category = chr, track = 1))
  print(get_cell_meta_data("original_xlim", category = chr, track = 1))
  print(get_cell_meta_data("original_ylim", category = chr, track = 1))
  print(get_cell_meta_data("name", category = chr, track = 1))
  print(get_cell_meta_data("column", category = chr, track = 1))
  print(get_cell_meta_data("row", category = chr, track = 1))
  print(get_cell_meta_data("track", category = chr, track = 1))
}
```

```

    cat("\n\n")
}

```

---

gtrellis_layout	<i>Initialize genome-level Trellis layout</i>
-----------------	---

---

## Description

Initialize genome-level Trellis layout

## Usage

```

gtrellis_layout(data = NULL, category = NULL,
  species = NULL, nrow = NULL, ncol = NULL,
  n_track = 1, track_height = 1, track_ylim = c(0, 1),
  track_axis = TRUE, track_ylab = "", ylab_rot = 90, title = NULL,
  xlab = "Genomic positions", xaxis = TRUE, xaxis_bin = NULL,
  equal_width = FALSE, compact = FALSE, border = TRUE, asist_ticks = TRUE,
  xpadding = c(0, 0), ypadding = c(0, 0), gap = unit(1, "mm"),
  byrow = TRUE, newpage = TRUE, add_name_track = FALSE,
  name_fontsize = 10, name_track_fill = "#EEEEEE",
  add_ideogram_track = FALSE, ideogram_track_height = unit(2, "mm"),
  axis_label_fontsize = 6, lab_fontsize = 10, title_fontsize = 16,
  legend = list(), legend_side = c("right", "bottom"),
  padding = unit(c(2, 2, 2, 2), "mm"), remove_chr_prefix = FALSE)

```

## Arguments

data	a data frame with at least three columns. The first three columns should be genomic categories (e.g. chromosomes), start positions and end positions. This data frame is used to extract ranges for each genomic category (minimal and maximal positions are taken as the range in the corresponding category).
category	subset of categories. It is also used for ordering.
species	Abbreviations of species. e.g. hg19 for human, mm10 for mouse. If this value is specified, the function will download chromInfo.txt.gz from UCSC ftp automatically. Short scaffolds will be removed if they have obvious different length as others. Non-normal chromosomes will also be detected and removed. Sometimes this detection is not always correct and if you find chromosomes shown on the plot is not what you expect, set category manually. The argument is passed to <a href="#">read.chromInfo</a> .
nrow	Number of rows in the layout.
ncol	Number of columns in the layout.
n_track	Number of tracks in each genomic category.
track_height	height of tracks. It should be numeric which means the value is relative and will be scaled into percent, or a <a href="#">unit</a> object.
track_ylim	ranges on y axes of tracks. The value can be a vector of length two which means all tracks share same y ranges, or a matrix with two columns, or a vector of length 2*n_track which will be coerced into the two-column matrix by rows.

track_axis	whether show y axes for tracks. The value is logical that can be either length one or number of tracks.
track_ylab	labels for tracks on y axes. The value can be either length one or number of tracks.
ylab_rot	value can only be 0 or 90.
title	title of the plot.
xlab	labels on x axes.
xaxis	whether show x axes.
xaxis_bin	bin size for x axes.
equal_width	whether all columns in the layout have the same width. If TRUE, short categories will be extended according to the longest category.
compact	For the categories which are put in a same row, will they be put compactly without being aligned by columns.
border	whether show borders.
asist_ticks	if axes ticks are added on one side in rows or columns, whether add ticks on the other sides.
xpadding	padding on x axes in each cell. Numeric value means relative ratio corresponding to the cell width. Use <code>I</code> to set it as absolute value which is measured in the data viewport (the coordinate system corresponding to the real data). Currently you cannot set it as a <code>unit</code> object.
ypadding	padding on y axes in each cell. Only numeric value is allowed currently.
gap	0 or a <code>unit</code> object. If it is length two, the first element corresponds to the gaps between rows and the second corresponds to the gaps between columns.
byrow	arrange categories (e.g. chromosomes) by rows or by columns in the layout.
newpage	whether call <code>grid.newpage</code> to create a new page.
add_name_track	whether add a pre-defined name track (insert before the first track). The name track is simply a track which only contains text. The default style of the name track is simple, but users can self define their own by <code>add_track</code> .
name_fontsize	font size for text in the name track. Note the font size also affects the height of name track.
name_track_fill	filled color for name track.
add_ideogram_track	whether to add a pre-defined ideogram track (insert after the last track). If the cytoband data for specified species is not available, this argument is ignored. The ideogram track simply contains rectangles with different colors, implemented by <code>add_track</code> .
ideogram_track_height	Height of ideogram track. The value should be a <code>unit</code> object.
axis_label_fontsize	font size for axis labels.
lab_fontsize	font size for x-labels and y-labels.
title_fontsize	font size for title.
legend	a <code>grob</code> or <code>Legends-class</code> object, or a list of them.
legend_side	side of the legend
padding	padding of the plot. Elements correspond to bottom, left, top, right paddings.
remove_chr_prefix	if chromosome names start with 'chr', whether to remove it.

**Details**

Genome-level Trellis graph visualizes genomic data conditioned by genomic categories (e.g. chromosomes). For each genomic category, multiple dimensional data which are represented as tracks describe different features from different aspects. The `gtrellis_layout` function arranges genomic categories on the plot in a quite flexible way. Then users apply `add_track` to add self-defined graphics to the plot track by track.

For more detailed demonstration of the function, please refer to the vignette.

**Value**

No value is returned.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**See Also**

[add\\_track](#), [add\\_ideogram\\_track](#)

**Examples**

```
gtrellis_layout()
gtrellis_layout(ncol = 5)
gtrellis_layout(n_track = 3, ncol = 4)

# for more examples, please go to the vignette
```

---

`gtrellis_show_index`    *Show index on each cell*

---

**Description**

Show index on each cell

**Usage**

```
gtrellis_show_index()
```

**Details**

The function adds name and index of track for each cell. It is only for demonstration purpose.

**Value**

No value is returned.

**Author(s)**

Zuguang Gu <z.gu@dkfz.de>

**Examples**

```
gtrellis_layout()  
gtrellis_show_index()
```

# Index

`add_heatmap_track`, 2, 6, 8  
`add_ideogram_track`, 3, 13  
`add_lines_track`, 4, 8  
`add_points_track`, 4, 8  
`add_rect_track`, 2, 5, 8  
`add_segments_track`, 6, 8  
`add_track`, 2, 4–6, 7, 8, 12, 13

`colorRamp2`, 2  
`current_track`, 9

`get_cell_meta_data`, 7, 9  
`gpar`, 4–6, 8  
`GRanges`, 2, 4–6  
`grid.newpage`, 12  
`grid.points`, 8  
`grid.raster`, 7  
`grid.rect`, 8  
`grob`, 12  
`gtrellis_layout`, 11, 13  
`gtrellis_show_index`, 13

I, 12

`read.chromInfo`, 11  
`read.cytoband`, 3

`unit`, 5, 8, 11, 12