Package ‘plotgardener’

April 10, 2025

Title Coordinate-Based Genomic Visualization Package for R

Version 1.13.0

Date 2024-06-28

Author Nicole Kramer [aut, cre], Eric S. Davis [aut], Craig Wenger [aut], Sarah Parker [ctb], Erika Deoudes [art], Michael Love [ctb], Douglas H. Phanstiel [aut, cre, cph]

Maintainer Nicole Kramer <`nekramer27@gmail.com`>, Douglas Phanstiel <`douglas_phanstiel@med.unc.edu`>

Description Coordinate-based genomic visualization package for R. It grants users the ability to programmatically produce complex, multi-paneled figures. Tailored for genomics, plotgardener allows users to visualize large complex genomic datasets and provides exquisite control over how plots are placed and arranged on a page.

Depends R (>= 4.1.0)

License MIT + file LICENSE

URL `https://phanstiellab.github.io/plotgardener`, `https://github.com/PhanstielLab/plotgardener`

Encoding UTF-8

RoxygenNote 7.3.1

Imports curl, data.table, dplyr, GenomeInfoDb, GenomicRanges, glue, grDevices, grid, ggplotify, IRanges, methods, plyranges, purrr, Rcpp, RColorBrewer, rhdf5, rlang, stats, strawr, tools, utils, withr

biocViews Visualization, GenomeAnnotation, FunctionalGenomics, GenomeAssembly, HiC

LinkingTo Rcpp

Suggests AnnotationDbi, AnnotationHub, BSgenome, BSgenome.Hsapiens.UCSC.hg19, ComplexHeatmap, GenomicFeatures, ggplot2, InteractionSet, knitr, org.Hs.eg.db, rtracklayer, plotgardenerData, pdftools, png, rmarkdown, scales, showtext, testthat (>= 3.0.0), TxDb.Hsapiens.UCSC.hg19.knownGene

VignetteBuilder knitr

Config/testthat/edition 3

BugReports `https://github.com/PhanstielLab/plotgardener/issues`

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

git_branch devel

git_last_commit bc6f006

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-04-09

Contents

`.checkCool` *Check for .(m)cool file and contents*

Description

Check for .(m)cool file and contents

Usage

```
.checkCool(file)
```

Arguments

`file`	Path to .(m)cool file

Author(s)

Sarah Parker

`.checkCoolErrors` *Error checking function for .(m)cool files*

Description

Error checking function for .(m)cool files

Usage

```
.checkCoolErrors(
  file,
  chrom,
  chromstart,
  chromend,
  zrange,
  altchrom,
  altchromstart,
  altchromend,
  norm,
  resolution
)
```

Arguments

`file`	Path to .(m)cool file
`chrom`	User-inputted chromosome
`chromstart`	User-inputted chromstart, can still be NULL at this point.
`chromend`	User-inputted chromend, can still be NULL at this point.
`zrange`	User-inputted zrange.
`altchrom`	User-inputted alt chromosome.
`altchromstart`	User-inputted alt chromstart.
`altchromend`	User-inputted alt chromend.
`norm`	User-inputted normalization.
`resolution`	Resolution, either user-inputted or determined by ’auto’.

`.coolAutoResolution` *Determine best resolution for size of region for .(m)cool files*

Description

Determine best resolution for size of region for .(m)cool files

Usage

```
.coolAutoResolution(file, chromstart, chromend)
```

Arguments

`file`	Path to .(m)cool file
`chromstart`	Chromstart of region
`chromend`	Chromend of region

`.coolRegion` *Add (alt)chromstart and (alt)chromend for NULL (alt)chrom region of .(m)cool files*

Description

Add (alt)chromstart and (alt)chromend for NULL (alt)chrom region of .(m)cool files

Usage

```
.coolRegion(file, chrom, resolution)
```

Arguments

`file`	Path to .(m)cool file
`chrom`	Chromosome of region; can also be altchromosome
`resolution`	Resolution to read chromsome info from

`.pullBinChunks` *Read in data for a bin chunk*

Description

Read in data for a bin chunk

Usage

```
.pullBinChunks(
  binChunk,
  file,
  bin_offsets,
  binChunkSize,
  datasetPath,
  end1bin,
  start2bin,
  end2bin
)
```

Arguments

`binChunk`	The binChunk indeces to read
`file`	Path to .(m)cool file
`bin_offsets`	Read in bin1 offsets
`binChunkSize`	Size of bin chunk, for comparison against the end of the bin chunk
`datasetPath`	Dataset path, for specifying resolution in .mcool file
`end1bin`	Bin where end1 starts
`start2bin`	Bin for chr2 starts
`end2bin`	Bin for end2 starts

`annoDomains` *Annotate domains in a Hi-C plot*

Description

Annotate domains in a Hi-C plot

Usage

```
annoDomains(
    plot,
    data,
    half = "inherit",
    linecolor = "black",
    params = NULL,
    ...
)
```

Arguments

`plot`	Hi-C plot object from `plotHicSquare` or `plotHicTriangle` on which to annotate pixels.
`data`	A string specifying the BED file path, a dataframe in BED format, or a GRanges object specifying domain ranges.
`half`	Character value specifying which half of hic plots to annotate. Triangle Hi-C plots will always default to the entirety of the triangular plot. Default value is `half = "inherit"`. Options are: • `"inherit"`: Domains will be annotated on the `half` inherited by the input Hi-C plot. • `"both"`: Domains will be annotated on both halves of the diagonal of a square Hi-C plot. • `"top"`: Domains will be annotated on the upper diagonal half of a square Hi-C plot. • `"bottom"`: Domains will be annotated ont the bottom diagonal half of a square Hi-C plot.
`linecolor`	A character value specifying the color of the domain annotations. Default value is `linecolor = "black"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `domain` object containing relevant genomic region, placement, and grob information.

Examples

```
## Define a GRanges object with TAD ranges
library(GenomicRanges)
library(IRanges)
domains <- GRanges("chr21",
    ranges = IRanges(
        start = c(28210000, 29085000, 29430000, 29700000),
        end = c(29085000, 29430000, 29700000, 30125000)
    )
)
```

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create page
pageCreate(width = 4.5, height = 4, default.units = "inches")

## Plot and place a square Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 0.5, width = 3, height = 3,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate domains on bottom half 0f Hi-C plot
annoDomains(
    plot = hicPlot, data = domains,
    half = "bottom", linecolor = "red"
)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot,
    x = 3.6, y = 0.5, width = 0.12, height = 1.2,
    just = c("left", "top"), default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = hicPlot, x = 0.5, y = 3.53, scale = "Mb",
    just = c("left", "top")
)

## Hide page guides
pageGuideHide()
```

`annoGenomeLabel` *Annotate genomic coordinates along the x or y-axis of a plot*

Description

Annotate genomic coordinates along the x or y-axis of a plot

Usage

```
annoGenomeLabel(
    plot,
    fontsize = 10,
    fontcolor = "black",
    linecolor = "black",
    margin = unit(1, "mm"),
    scale = "bp",
    commas = TRUE,
    sequence = TRUE,
    boxWidth = 0.5,
    axis = "x",
    at = NULL,
    tcl = 0.5,
    x,
    y,
    just = c("left", "top"),
    default.units = "inches",
    params = NULL,
    ...
)
```

Arguments

`plot`	Input plot to annotate genomic coordinates. Genomic coordinates and assembly will be inherited from `plot`.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 10`.
`fontcolor`	A character value indicating the color for text. Default value is `fontcolor = "black"`.
`linecolor`	A character value indicating the color of the genome label axis. Default value is `linecolor = "black"`.
`margin`	A numeric or unit vector specifying space between axis and coordinate labels. Default value is `margin = unit(1, "mm")`.
`scale`	A character value indicating the scale of the coordinates along the genome label. Default value is `scale = "bp"`. Options are: • `"bp"`: base pairs. • `"Kb"`: kilobase pairs. 1 kilobase pair is equal to 1000 base pairs. • `"Mb"`: megabase pairs. 1 megabase pair is equal to 1000000 base pairs.
`commas`	A logical value indicating whether to include commas in start and stop labels. Default value is `commas = TRUE`.
`sequence`	A logical value indicating whether to include sequence information above the label of an x-axis (only at appropriate resolutions).
`boxWidth`	A numeric value indicating the width of the boxes representing sequence information at appropriate resolutions. Default value is `boxWidth = 0.5`.

`axis`	A character value indicating along which axis to add genome label. Sequence information will not be displayed along a y-axis. Default value is `axis = "x"`. Options are: • `"x"`: Genome label will be plotted along the x-axis. • `"y"`: Genome label will be plotted along the y-axis. This is typically used for a square Hi-C plot made with `plotHicSquare`.
`at`	A numeric vector of x-value locations for tick marks.
`tcl`	A numeric specifying the length of tickmarks as a fraction of text height. Default value is `tcl = 0.5`.
`x`	A numeric or unit object specifying genome label x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying genome label y-location. The character value will place the genome label y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`just`	Justification of genome label relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x` or `y` are only given as numerics. Default value is `default.units = "inches"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters or digit specifications. See gpar and formatC.

Value

Returns a `genomeLabel` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load hg19 genomic annotation packages
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
library("org.Hs.eg.db")

## Create page
pageCreate(width = 5, height = 2, default.units = "inches")

## Plot and place gene track on page
genesPlot <- plotGenes(
    chrom = "chr8",
    chromstart = 1000000, chromend = 2000000,
    assembly = "hg19", fill = c("grey", "grey"),
    fontcolor = c("grey", "grey"),
    x = 0.5, y = 0.25, width = 4, height = 1,
    just = c("left", "top"),
    default.units = "inches"
```

```
)

## Annotate x-axis genome labels at different scales
annoGenomeLabel(
    plot = genesPlot, scale = "Mb",
    x = 0.5, y = 1.25, just = c("left", "top"),
    default.units = "inches"
)
annoGenomeLabel(
    plot = genesPlot, scale = "Kb",
    x = 0.5, y = 1.5, just = c("left", "top"),
    default.units = "inches"
)
annoGenomeLabel(
    plot = genesPlot, scale = "bp",
    x = 0.5, y = 1.75, just = c("left", "top"),
    default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`annoHeatmapLegend` *Add a color scale legend for heatmap-style plots*

Description

Add a color scale legend for heatmap-style plots

Usage

```
annoHeatmapLegend(
    plot,
    orientation = "v",
    fontsize = 8,
    fontcolor = "dark grey",
    scientific = FALSE,
    digits = 1,
    ticks = FALSE,
    breaks = NULL,
    border = FALSE,
    x,
    y,
    width,
    height,
    just = c("left", "top"),
    default.units = "inches",
    params = NULL,
    ...
)
```

Arguments

`plot`	Heatmap-style plot object to add heatmap legend for.
`orientation`	A string specifying legend orientation. Default value is `orientation = "v"`. Options are: • `"v"`: Vertical legend orientation. • `"h"`: Horizontal legend orientation.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 8`.
`fontcolor`	Character value specfying text fontcolor. Default value is `fontcolor = "dark grey"`.
`scientific`	Logical value specifying if numeric color value labels should be encoded in scientific format. Default value is `scientific = FALSE`.
`digits`	Numeric specifying how many significant digits to include of numeric color value labels. Default value is `digits = 1`.
`ticks`	Logical value specifying if tick marks on the heatmap colorbar should be visible. Default value is `ticks = FALSE`.
`breaks`	A numeric vector specifying tick breaks. Default value is `breaks = NULL`.
`border`	Logical value indicating whether to add a border around heatmap legend. Default value is `border = FALSE`.
`x`	A numeric or unit object specifying x-location of legend.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying y-location of legend. The character value will place the legend y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying width of legend.
`height`	A numeric or unit object specifying height of legend.
`just`	Justification of heatmap legend relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `heatmapLegend` object with relevant color value, placement, and grob information.

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create page
pageCreate(width = 4, height = 3.5, default.units = "inches")

## Plot and place a square Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 0.5, width = 2.5, height = 2.5,
    just = c("left", "top"),
    default.units = "inches"
)

## Add heatmap legend
annoHeatmapLegend(
    plot = hicPlot,
    x = 3.2, y = 0.5, width = 0.12, height = 1.2,
    just = c("left", "top"), default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = hicPlot, x = 0.5, y = 3.03, scale = "Mb",
    just = c("left", "top")
)

## Hide page guides
pageGuideHide()
```

`annoHighlight` *Annotates a highlight box around a specified genomic region of a plot*

Description

Annotates a highlight box around a specified genomic region of a plot

Usage

```
annoHighlight(
    plot,
    chrom,
    chromstart = NULL,
```

```
    chromend = NULL,
    fill = "grey",
    linecolor = NA,
    alpha = 0.4,
    y,
    height,
    just = c("left", "top"),
    default.units = "inches",
    params = NULL,
    ...
)
```

Arguments

`plot`	Input plot on which to annotate genomic region.
`chrom`	Chromosome of region to be highlighted, as a string.
`chromstart`	Integer start position on chromosome to be highlighted.
`chromend`	Integer end position on chromosome to be highlighted.
`fill`	A character value specifying highlight box fill color. Default value is `fill = "grey"`.
`linecolor`	A character value specifying highlight box line color. Default value is `linecolor = NA`.
`alpha`	Numeric value specifying color transparency. Default value is `alpha = 0.4`.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying square highlight box y-location. The character value will place the highlight box y relative to the bottom of the most recently plotted plot according to the units of the `plotgardener` page.
`height`	A numeric or unit object specifying highlight box height.
`just`	Justification of highlight box relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `y` or `height` are only given as numerics or numeric vectors. Default value is `default.units = "inches"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `highlight` object containing relevant genomic region, placement, and grob information.

Examples

```
## Create a page
pageCreate(width = 7.5, height = 1.5, default.units = "inches")
```

```
## Plot and place a signal plot
library(plotgardenerData)
data("IMR90_ChIP_H3K27ac_signal")
region <- pgParams(
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    range = c(0, 45)
)
signalPlot <- plotSignal(
    data = IMR90_ChIP_H3K27ac_signal, params = region,
    x = 0.5, y = 0.25, width = 6.5, height = 0.65,
    just = c("left", "top"),
    default.units = "inches"
)

## Highlight genomic region on signal plot
annoHighlight(
    plot = signalPlot,
    chrom = "chr21",
    chromstart = 29000000, chromend = 29125000,
    y = 0.25, height = 1, just = c("left", "top"),
    default.units = "inches"
)

## Plot text label
plotText(
    label = "region of interest", fontsize = 8, fontcolor = "black",
    x = 3.5, y = 0.2, just = "bottom", default.units = "inches"
)

## Plot genome label
plotGenomeLabel(
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 1.3, length = 6.5, default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`annoPixels` *Annotate pixels in a Hi-C plot*

Description

Annotate pixels in a Hi-C plot

Usage

```
annoPixels(
    plot,
    data,
    type = "box",
    half = "inherit",
    shift = 4,
    params = NULL,
    quiet = FALSE,
    ...
)
```

Arguments

`plot`	Hi-C plot object from `plotHicSquare` or `plotHicTriangle` on which to annotate pixels.
`data`	A string specifying the BEDPE file path, a dataframe in BEDPE format specifying pixel positions, or a GInteractions object specifying pixel positions.
`type`	Character value specifying type of annotation. Default value is `type = "box"`. Options are: • `"box"`: Boxes are drawn around each pixel. • `"circle"`: Circles are drawn around each pixel. • `"arrow"`: Arrows are drawn pointing to each pixel.
`half`	Character value specifying which half of hic plots to annotate. Triangle Hi-C plots will always default to the entirety of the triangular plot. Default value is `half = "inherit"`. Options are: • `"inherit"`: Pixels will be annotated on the `half` inherited by the input Hi-C plot. • `"both"`: Pixels will be annotated on both halves of the diagonal of a square Hi-C plot. • `"top"`: Pixels will be annotated on the upper diagonal half of a square Hi-C plot. • `"bottom"`: Pixels will be annotated on the bottom diagonal half of a square Hi-C plot.
`shift`	Numeric specifying the number of pixels on either end of main pixel in a box or circle. Numeric specifying number of pixels for the length of an arrow.
`params`	An optional pgParams object containing relevant function parameters.
`quiet`	A logical indicating whether or not to print messages.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `pixel` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load Hi-C data and BEDPE data
library(plotgardenerData)
data("IMR90_HiC_10kb")
data("IMR90_DNAloops_pairs")

## Create page
pageCreate(width = 4.5, height = 4, default.units = "inches")

## Plot and place a square Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 0.5, width = 3, height = 3,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate loops of both sides of Hi-C plot with squares
pixels <- annoPixels(
    plot = hicPlot, data = IMR90_DNAloops_pairs, type = "box",
    half = "both"
)

## Annotate loops on one side of Hi-C plot with arrows
## and the other side with circles
pagePlotRemove(plot = pixels)
pixels1 <- annoPixels(
    plot = hicPlot, data = IMR90_DNAloops_pairs,
    type = "arrow", half = "top", shift = 8
)
pixels2 <- annoPixels(
    plot = hicPlot, data = IMR90_DNAloops_pairs,
    type = "circle", half = "bottom"
)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot,
    x = 3.6, y = 0.5, width = 0.12, height = 1.2,
    just = c("left", "top"), default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = hicPlot, x = 0.5, y = 3.53, scale = "Mb",
    just = c("left", "top")
)
```

```
## Hide page guides
pageGuideHide()
```

`annoSegments` *Annotates a line segment within a plot*

Description

Annotates a line segment within a plot

Usage

```
annoSegments(
    x0,
    y0,
    x1,
    y1,
    plot,
    default.units = "native",
    linecolor = "black",
    lwd = 1,
    lty = 1,
    lineend = "butt",
    linejoin = "mitre",
    arrow = NULL,
    params = NULL,
    ...
)
```

Arguments

`x0`	A numeric vector or unit object indicating the starting x-values of the line segments.
`y0`	A numeric vector or unit object indicating the starting y-values of the line segments.
`x1`	A numeric vector or unit object indicating the stopping x-values of the line segments.
`y1`	A numeric vector or unit object indicating the stopping y-values of the line segments.
`plot`	Input plotgardener plot to internally plot line segments relative to.
`default.units`	A string indicating the default units to use if `x0`, `y0`, `x1`, or `y1` are only given as numeric vectors. Default value is `default.units = "native"`.
`linecolor`	A character value specifying segment line color. Default value is `linecolor = "black"`.
`lwd`	A numeric specifying segment line width. Default value is `lwd = 1`.

`lty`	A numeric specifying segment line type. Default value is `lty = 1`.
`lineend`	A character value specifying line end style. Default value is `lineend = "butt"`. Options are: • `"round"`: Segment ends are rounded. • `"butt"`: Segment ends end exactly where ended. • `"square"`: Segment ends are squared.
`linejoin`	A character value specifying line join style. Default value is `linejoin = "mitre"`. Options are: • `"round"`: Line joins are rounded. • `"mitre"`: Line joins are sharp corners. • `"bevel"`: Line joins are flattened corners.
`arrow`	A list describing arrow heads to place at either end of the line segments, as produced by the arrow function.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `segments` object containing relevant placement and grob information.

See Also

grid.segments, arrow

Examples

```
library(grid)
## Create a page
pageCreate(width = 7.5, height = 2.5, default.units = "inches")

## Plot a Manhattan plot
library(plotgardenerData)
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
data("hg19_insulin_GWAS")
manhattanPlot <- plotManhattan(
    data = hg19_insulin_GWAS, assembly = "hg19",
    fill = c("grey", "#37a7db"),
    sigLine = TRUE,
    col = "grey", lty = 2, range = c(0, 14),
    x = 0.5, y = 0, width = 6.5, height = 2,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = manhattanPlot, x = 0.5, y = 2, fontsize = 8,
    just = c("left", "top"),
```

```
    default.units = "inches"
)
plotText(
    label = "Chromosome", fontsize = 8,
    x = 3.75, y = 2.20, just = "center", default.units = "inches"
)

## Annotate y-axis
annoYaxis(
    plot = manhattanPlot, at = c(0, 2, 4, 6, 8, 10, 12, 14),
    axisLine = TRUE, fontsize = 8
)

## Annotate a line segment for an additional significance line of
## the Manhattan plot
annoSegments(
    x0 = unit(0, "npc"), y0 = 10,
    x1 = unit(1, "npc"), y1 = 10,
    plot = manhattanPlot, default.units = "native",
    linecolor = "red", lty = 2
)

## Plot y-axis label
plotText(
    label = "-log10(p-value)", x = 0.15, y = 1, rot = 90,
    fontsize = 8, fontface = "bold", just = "center",
    default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`annoText` *Annotates text within a plot*

Description

Annotates text within a plot

Usage

```
annoText(
    label,
    fontcolor = "black",
    fontsize = 12,
    rot = 0,
    check.overlap = FALSE,
    plot,
    x,
```

```
    y,
    just = "center",
    default.units = "native",
    params = NULL,
    ...
)
```

Arguments

`label`	Character or expression of text to be plotted.
`fontcolor`	A character value specifying text fontcolor. Default value is `fontcolor = "black"`.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 12`.
`rot`	A numeric specifying the angle to rotate the text. Default value is `rot = 0`.
`check.overlap`	A logical value to indicate whether to check for and omit overlapping text. Default value is `check.overlap = FALSE`.
`plot`	Input plotgardener plot to internally place text relative to.
`x`	A numeric vector or unit object specifying text x-location.
`y`	A numeric vector or unit object specifying text y-location.
`just`	Justification of text relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = "center"`.
`default.units`	A string indicating the default units to use if `x` or `y` are only given as numerics or numeric vectors. Default value is `default.units = "native"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `text` object containing relevant placement and grob information.

See Also

grid.text

Examples

```
## Create a page
pageCreate(width = 4, height = 4, default.units = "inches")

## Plot text relative to a plotgardener plot
library(plotgardenerData)
data("IMR90_HiC_10kb")
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, chrom = "chr21",
    chromstart = 28000000, chromend = 29500000,
    assembly = "hg19",
```

```
    zrange = c(0, 70),
    x = 0.5, y = 0.5, width = 3, height = 3,
    just = c("left", "top"),
    default.units = "inches"
)
annoGenomeLabel(
    plot = hicPlot, x = 0.5, y = 3.55, scale = "Mb",
    just = c("left", "top"), default.units = "inches"
)

annoText(
    label = "Loop", fontsize = 8, plot = hicPlot,
    x = 29075000, y = 28150000,
    just = "center", default.units = "native"
)

## Hide page guides
pageGuideHide()
```

`annoXaxis` *Add an x-axis to a plot*

Description

Add an x-axis to a plot

Usage

```
annoXaxis(
    plot,
    at = NULL,
    label = TRUE,
    main = TRUE,
    scipen = 999,
    axisLine = FALSE,
    params = NULL,
    ...
)
```

Arguments

`plot`	Plot object to annotate with x-axis.
`at`	A numeric vector of x-value locations for tick marks.
`label`	A logical value indicating whether to draw the labels on the tick marks, or an expression or character vector which specify the labels to use. If not logical, must be the same length as the `at` argument.
`main`	A logical value indicating whether to draw the x-axis at the bottom of the plot. Default value is `main = TRUE`. Options are:

	- `TRUE`: x-axis is drawn at the bottom of the plot. - `FALSE`: x-axis is drawn at the top of the plot.
`scipen`	An integer indicating the penalty to be applied when deciding to print numeric values in fixed or exponential notation. Default value is `scipen = 999`.
`axisLine`	A logical value indicating whether to show the axis line. Default value is `axisLine = FALSE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `xaxis` object containing relevant grob information.

Examples

```
## Load transcript information
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
library("org.Hs.eg.db")

## Create page
pageCreate(width = 7.5, height = 4.5, default.units = "inches")

## Plot gene transcripts
transcriptPlot <- plotTranscripts(
    chrom = "chr1",
    chromstart = 1000000,
    chromend = 2000000,
    assembly = "hg19",
    x = 0.5, y = 0,
    width = 6.5, height = 4,
    just = c("left", "top"),
    default.units = "inches"
)

## Add standard x-axis to transcript plot
annoXaxis(
    plot = transcriptPlot,
    at = c(1000000, 1250000, 1500000, 1750000, 2000000),
    fontsize = 8
)
plotText(
    label = "Basepairs", fontsize = 10, fontface = "bold",
    x = 3.75, y = 4.3, just = "top"
)

## Hide page guides
pageGuideHide()
```

`annoYaxis` *Add a y-axis to a plot*

Description

Add a y-axis to a plot

Usage

```
annoYaxis(
    plot,
    at = NULL,
    label = TRUE,
    main = TRUE,
    scipen = 999,
    axisLine = FALSE,
    params = NULL,
    ...
)
```

Arguments

`plot`	Plot object to annotate with y-axis.
`at`	A numeric vector of y-value locations for tick marks.
`label`	A logical value indicating whether to draw the labels on the tick marks, or an expression or character vector which specify the labels to use. If not logical, must be the same length as the `at` argument.
`main`	A logical value indicating whether to draw the y-axis at the left of the plot. Default value is `main = TRUE`. Options are: • `TRUE`: y-axis is drawn at the left of the plot. • `FALSE`: y-axis is drawn at the right of the plot.
`scipen`	An integer indicating the penalty to be applied when deciding to print numeric values in fixed or exponential notation. Default value is `scipen = 999`.
`axisLine`	A logical value indicating whether to show the axis line. Default value is `axisLine = FALSE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `yaxis` object containing relevant grob information.

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create page
pageCreate(width = 4, height = 3.5, default.units = "inches")

## Plot and place a square Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 1, y = 0.5, width = 2.5, height = 2.5,
    just = c("left", "top"),
    default.units = "inches"
)

## Add standard y-axis to Hi-C plot
annoYaxis(
    plot = hicPlot, at = c(28000000, 29000000, 30300000),
    fontsize = 10
)

## Annotate genome label on x-axis
annoGenomeLabel(plot = hicPlot, x = 1, y = 3.03)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot,
    x = 3.6, y = 0.5, width = 0.12, height = 1.2
)

## Hide page guides
pageGuideHide()
```

`annoZoomLines` *Annotates zoom lines for a specified genomic region of a plot*

Description

Annotates zoom lines for a specified genomic region of a plot

Usage

```
annoZoomLines(
    plot,
```

```
    chrom,
    chromstart = NULL,
    chromend = NULL,
    y0,
    x1 = NULL,
    y1,
    extend = 0,
    default.units = "inches",
    linecolor = "grey",
    lty = 2,
    params = NULL,
    ...
)
```

Arguments

`plot`	Input plot to annotate genomic region zoom lines from.
`chrom`	Chromosome of region to draw zoom lines from, as a string.
`chromstart`	Integer start position on chromosome to draw zoom lines from.
`chromend`	Integer end position on chromosome to draw zoom lines from.
`y0`	A numeric vector or unit object indicating the starting y-values of the zoom line segments. If two values are given, the first value will correspond to the left zoom line and the second value will correspond to the right zoom line.
`x1`	A numeric vector or unit object indicating the stopping x-values of the zoom line segments. If two values are given, the first value will correspond to the left zoom line and the second value will correspond to the right zoom line. If NULL, straight lines from zoomed genomic region will be drawn.
`y1`	A numeric vector or unit object indicating the stopping y-values of the zoom line segments. If two values are given, the first value will correspond to the left zoom line and the second value will correspond to the right zoom line.
`extend`	A numeric vector or unit object indicating the length to extend straight lines from each end of the zoom line segments. If two values are given, the first value will correspond to the top extension length and the second value will correspond to the bottom extension length. Default value is `extend = 0`.
`default.units`	A string indicating the default units to use if `y0`, `x1`, `y1`, or `extend` are only given as numerics or numeric vectors. Default value is `default.units = "inches"`.
`linecolor`	A character value specifying zoom line color. Default value is `linecolor = "grey"`.
`lty`	A numeric specifying zoom line type. Default value is `lty = 2`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `zoom` object containing relevant genomic region, placement, and grob information.

Examples

```
## Create a page
pageCreate(width = 7.5, height = 4.75, default.units = "inches")

## Plot and place a Manhattan plot
library(plotgardenerData)
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
data("hg19_insulin_GWAS")
manhattanPlot <- plotManhattan(
    data = hg19_insulin_GWAS, assembly = "hg19",
    fill = c("grey", "#37a7db"),
    sigLine = FALSE,
    col = "grey", lty = 2, range = c(0, 14),
    x = 0.5, y = 0, width = 6.5, height = 2,
    just = c("left", "top"),
    default.units = "inches"
)
annoYaxis(
    plot = manhattanPlot, at = c(0, 2, 4, 6, 8, 10, 12, 14),
    axisLine = TRUE, fontsize = 8
)

## Annotate zoom lines for a region on chromsome 21
zoomRegion <- pgParams(
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19"
)
annoZoomLines(
    plot = manhattanPlot, params = zoomRegion,
    y0 = 2, x1 = c(0.5, 7), y1 = 2.5, extend = c(0, 1.1),
    default.units = "inches",
    lty = 3
)

## Annotate highlight region for zoom region
annoHighlight(
    plot = manhattanPlot, params = zoomRegion,
    y = 2, height = 2, just = c("left", "bottom"),
    default.units = "inches",
    fill = "red", alpha = 0.8
)

## Plot Manhattan plot data and signal track under zoom lines
manhattanPlotZoom <- plotManhattan(
    data = hg19_insulin_GWAS,
    fill = "grey",
    sigLine = FALSE,
    baseline = TRUE,
    params = zoomRegion, range = c(0, 14),
    x = 0.5, y = 2.6,
    width = 6.5, height = 1
```

```
)
data("IMR90_ChIP_H3K27ac_signal")
signalPlot <- plotSignal(
    data = IMR90_ChIP_H3K27ac_signal, params = zoomRegion,
    range = c(0, 45),
    x = 0.5, y = "b0.1",
    width = 6.5, height = 0.65,
    just = c("left", "top"),
    default.units = "inches"
)

## Plot genome label
plotGenomeLabel(
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 4.4, length = 6.5,
    default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`assembly` *Make a* `assembly` *object for alternate TxDb, OrgDb,and BSgenome genomic annotation packages*

Description

Make a `assembly` object for alternate TxDb, OrgDb,and BSgenome genomic annotation packages

Usage

```
assembly(
    Genome,
    TxDb,
    OrgDb,
    gene.id.column = "ENTREZID",
    display.column = "SYMBOL",
    BSgenome = NULL
)
```

Arguments

`Genome`	String indicating the name of the genome assembly.
`TxDb`	String of existing TxDb package name or a TxDb object.
`OrgDb`	String of the desired OrgDb package name.

`gene.id.column`	String of the TxDb column name that refers to the given TxDb gene IDs. Default value is `gene.id.column = "ENTREZID"`.
`display.column`	String of the OrgDb column name that refers to the type of gene symbol to be displayed in plots. Default value is `display.column = "SYMBOL"`.
`BSgenome`	String of the desired BSgenome package name.

Value

Returns a `assembly` object containing all input parameters.

See Also

TxDb, OrgDb-class, BSgenome

Examples

```
## Create a custom `assembly` object for hg38/GRCh38 packages
newAssembly <- assembly(
    Genome = "hg38_GRCh38",
    TxDb = "TxDb.Hsapiens.UCSC.hg38.knownGene",
    OrgDb = "org.Hs.eg.db",
    BSgenome = "BSgenome.Hsapiens.NCBI.GRCh38"
)
```

`c` *Combine multiple pgParams objects into a vector*

Description

Combine multiple pgParams objects into a vector

Usage

```
c(..., recursive = FALSE)
```

Arguments

`...`	pgParams objects to be concatenated.
`recursive`	logical. If `recursive = TRUE`, the function recursively descends through lists (and pairlists) combining all their elements into a vector.

Value

`NULL` or an expression or a vector of an appropriate mode. (With no arguments the value is `NULL`.)

Examples

```
## Define parameters
p1 <- pgParams(chrom = "chr1", assembly = "hg19")

## Define another set of parameters
p2 <- pgParams(fontsize = 10, assembly = "hg19")

## Combine parameters into one `pgParams` object
pTotal <- c(p1, p2)
```

`calcSignalRange` *Calculate a score range for multiple signals*

Description

Calculate a score range for multiple signals

Usage

```
calcSignalRange(
    data,
    chrom = NULL,
    chromstart = 1,
    chromend = .Machine$integer.max,
    assembly = "hg38",
    negData = FALSE)
```

Arguments

`data`	List of data to be plotted as character values specifying multiple bigwig file paths, dataframes in BED format, or GRanges objects with metadata column `score`.
`chrom`	Chromosome of data region ragne as a string, if range for a specific chromosome is desired.
`chromstart`	Integer start position on chromosome to get data range.
`chromend`	Integer end position on chromosome to get data range.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`negData`	A logical value indicating whether any of the data has both positive and negative scores and the signal range should be adjusted accordingly. Default value is `negData = FALSE`.

Value

Returns a vector of length 2 with the calculated c(min, max) range.

Examples

```
library("plotgardenerData")
data("GM12878_ChIP_CTCF_signal")
data("IMR90_ChIP_CTCF_signal")
data("GM12878_ChIP_H3K27ac_signal")
data("IMR90_ChIP_H3K27ac_signal")

calcSignalRange(data = list(GM12878_ChIP_CTCF_signal,
                            GM12878_ChIP_H3K27ac_signal,
                            IMR90_ChIP_CTCF_signal,
                            IMR90_ChIP_H3K27ac_signal),
                chrom = "chr21",
                chromstart = 28150000, chromend = 29150000,
                assembly = "hg38", negData = FALSE)
```

`colorby` *Handle plotgardener color scaling parameters*

Description

`colorby` should be used to create a set of parameters that specify color scaling for the functions `plotPairs`, `plotPairsArches`, and `plotRanges`.

Usage

```
colorby(column, palette = NULL, range = NULL, scalePerRegion = FALSE)
```

Arguments

`column`	String specifying name of data column to scale colors by.
`palette`	(optional) A function describing the color palette to use for color scaling.
`range`	(optional) A numeric vector specifying the range of values to apply a color scale to.
`scalePerRegion`	A logical value indicating whether to adjust NULL range of numerical ‘colorby‘ values to subset of data in a plotted genomic region. Default value is `scalePerRegion = FALSE`.

Value

Returns a "`colorby`" object.

Examples

```
## Load paired ranges data in BEDPE format
library(plotgardenerData)
data("IMR90_DNAloops_pairs")

## Add a length column
IMR90_DNAloops_pairs$length <-
        (IMR90_DNAloops_pairs$start2 - IMR90_DNAloops_pairs$start1) / 1000

## Plot pairs with colorby object set for `length` column
bedpePlot <- plotPairs(
    data = IMR90_DNAloops_pairs,
    chrom = "chr21",
    chromstart = 27900000, chromend = 30700000,
    assembly = "hg19",
    fill = colorby("length", palette =
                colorRampPalette(c("dodgerblue2", "firebrick2"))),
    lwd = 2, spaceHeight = .7,
)
```

`defaultPackages` *Display the default genomic annotation packages associated with a genome build*

Description

Display the default genomic annotation packages associated with a genome build

Usage

```
defaultPackages(Genome)
```

Arguments

`Genome`	String indicating the name of the genome assembly.

Value

Returns a list of the default data packages for a genome build.

Examples

```
## View default genomic annotation packages associated with "hg19"
defaultPackages(Genome = "hg19")

## View default genomic annotation packages associated with "mm9"
defaultPackages(Genome = "mm9")
```

`genomes` *Display the included available default genome assemblies*

Description

Display the included available default genome assemblies

Usage

```
genomes()
```

Value

Returns the included available default genome assemblies

Examples

```
genomes()
```

`mapColors` *Maps a numeric or character vector to a color palette and returns the vector of colors*

Description

Maps a numeric or character vector to a color palette and returns the vector of colors

Usage

```
mapColors(vector, palette, range = NULL)
```

Arguments

`vector`	Vector to map to color.
`palette`	Color palette function.
`range`	Range of values to map for a numerical value.

Details

This function allows for the manual mapping of a numerical or factor vector to a palette of colors. For a more automatic implementation of this functionality in plotgardener functions, colorby objects can be used.

Value

Returns a character vector of color values. If the input vector is numerical, this vector will have additional ‘palette‘ and ‘range‘ attributes.

See Also

colorby

Examples

```
## Load paired ranges data in BEDPE format
library(plotgardenerData)
data("IMR90_DNAloops_pairs")

## Add a length column
IMR90_DNAloops_pairs$length <- (IMR90_DNAloops_pairs$start2 -
        IMR90_DNAloops_pairs$start1) / 1000

## Map length column to a vector of colors
colors <- mapColors(vector = IMR90_DNAloops_pairs$length,
            palette = colorRampPalette(c("dodgerblue2", "firebrick2")))

## Pass color vector into bbPlotPairsArches
heights <- IMR90_DNAloops_pairs$length / max(IMR90_DNAloops_pairs$length)
pageCreate(width = 7.5, height = 2.1, default.units = "inches",
            showGuides = FALSE, xgrid = 0, ygrid = 0)
params <- pgParams(
    chrom = "chr21",
    chromstart = 27900000, chromend = 30700000,
    assembly = "hg19",
    width = 7
)

archPlot <- plotPairsArches(
    data = IMR90_DNAloops_pairs, params = params,
    fill = colors,
    linecolor = "fill",
    archHeight = heights, alpha = 1,
    x = 0.25, y = 0.25, height = 1.5,
    just = c("left", "top"),
    default.units = "inches"
)

annoGenomeLabel(plot = archPlot, x = 0.25, y = 1.78, scale = "Mb")
annoHeatmapLegend(
    plot = archPlot, fontcolor = "black",
    x = 7.0, y = 0.25,
    width = 0.10, height = 1, fontsize = 10
)
plotText(
    label = "Kb", rot = 90, x = 6.9, y = 0.75,
    just = c("center", "center"),
    fontsize = 10
)
```

`pageCreate` *Create a page for a plotgardener layout*

Description

Create a page for a plotgardener layout

Usage

```
pageCreate(
    width = 8.5,
    height = 11,
    default.units = "inches",
    bg = NA,
    xgrid = 0.5,
    ygrid = 0.5,
    showGuides = TRUE,
    params = NULL
)
```

Arguments

`width`	A numeric or unit object specifying page width. Default value is `width = 8`.
`height`	A numeric or unit object specifying page height. Default value is `height = 11`.
`default.units`	A string indicating the default units to use if `width` or `height` are only given as numerics. Default value is `default.units = "inches"`.
`bg`	Character value indicating page background color. Default value is `bg = NA`.
`xgrid`	A numeric indicating the increment by which to place vertical gridlines. Default value is `xgrid = 0.5`.
`ygrid`	A numeric indicating the increment by which to place horizontal gridlines. Default value is `ygrid = 0.5`.
`showGuides`	A logical value indicating whether to draw a black border around the entire page and guiding rulers along the top and left side of the page. Default value is `showOutline = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.

Details

`width` and `height` must be specified in the same units.

Value

None.

Examples

```
## Create a 6-inch wide, 4.5-inch high page
pageCreate(width = 6, height = 4.5, default.units = "inches")

## Create a 14-cm wide, 10-cm high page
pageCreate(width = 14, height = 10, default.units = "cm")
```

`pageGuideHide` *Remove guides from a plotgardener page*

Description

Remove guides from a plotgardener page

Usage

```
pageGuideHide()
```

Value

None.

Note

Please note that due to the implementation of ‘grid‘ removal functions, using ‘pageGuideHide‘ within a ‘pdf‘ call will result in the rendering of a separate, new page with the plot guides removed. To avoid this artifact, hide guides in the ‘pageCreate‘ function call with ‘showGuides = FALSE‘.

Examples

```
## Make a page
pageCreate(width = 7, height = 4, default.units = "inches")

## Hide page guides
pageGuideHide()
```

`pageGuideHorizontal` *Draw a horizontal guideline at a specified y-coordinate on a plotgardener page*

Description

Draw a horizontal guideline at a specified y-coordinate on a plotgardener page

Usage

```
pageGuideHorizontal(
    y,
    default.units = "inches",
    linecolor = "grey55",
    params = NULL,
    ...
)
```

Arguments

`y`	A numeric or unit object specifying y-coordinate of guide.
`default.units`	A string indicating the default units to use if `y` is only given as a numeric. Default value is `default.units = "inches"`.
`linecolor`	Character value indicating color of guideline. Default value is `linecolor = "grey55"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

None.

Examples

```
## Create a page
pageCreate(width = 6, height = 5, default.units = "inches")

## Add red horizontal guideline at y = 2.5 inches
pageGuideHorizontal(y = 2.5, linecolor = "red")
```

`pageGuideShow` *Reshow guides drawn with* `pageCreate`*,* `pageGuideHorizontal`*, and* `pageGuideVertical`

Description

Reshow guides drawn with `pageCreate`, `pageGuideHorizontal`, and `pageGuideVertical`

Usage

```
pageGuideShow()
```

Value

None.

See Also

pageCreate, pageGuideHorizontal, pageGuideVertical

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create a page
pageCreate(width = 3, height = 3, default.units = "inches")

## Add a page guide
pageGuideHorizontal(y = 0.5, default.units = "inches")

## Plot and place Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    x = 0.5, y = 0.5, width = 2, height = 2,
    just = c("left", "top"),
    default.units = "inches"
)

## Hide page guides
pageGuideHide()

## Re-show page guides
pageGuideShow()

## Annotate genome label
```

```
annoGenomeLabel(
    plot = hicPlot, scale = "Mb", axis = "x",
    x = 0.5, y = 2.53, just = c("left", "top")
)
```

`pageGuideVertical` *Draw a vertical guideline at a specified x-coordinate on a plotgardener page*

Description

Draw a vertical guideline at a specified x-coordinate on a plotgardener page

Usage

```
pageGuideVertical(
    x,
    default.units = "inches",
    linecolor = "grey55",
    params = NULL,
    ...
)
```

Arguments

`x`	A numeric or unit object specifying x-coordinate of guide.
`default.units`	A string indicating the default units to use if `x` is only given as a numeric. Default value is `default.units = "inches"`.
`linecolor`	Character value indicating color of guideline. Default value is `linecolor = "grey55"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

None.

Examples

```
## Create a page
pageCreate(width = 6, height = 5, default.units = "inches")

## Add blue vertical guideline at x = 1.7 inches
pageGuideVertical(x = 1.7, linecolor = "blue")
```

`pageLayoutCol` *Generate column positions for a number of plot elements with a specified width and space between them*

Description

Generate column positions for a number of plot elements with a specified width and space between them

Usage

```
pageLayoutCol(x, width, space, n, default.units = "inches")
```

Arguments

`x`	A numeric or unit object specifying the starting column x-position.
`width`	A numeric or unit object specifying the width of columns.
`space`	A numeric or unit object specifying the space between columns.
`n`	An integer specifying the number of elements to generate column positions for.
`default.units`	A string indicating the default units to use if `x`, `w`, or `s` are only given as numerics. Default value is `default.units = "inches"`

Value

Returns a unit vector of page positions.

Examples

```
# Starting at 0.5 units, return a vector of positions for 3 objects that
# are 2 units in width with 0.1 units of space between them

pageLayoutCol(x = 0.5, width = 2, space = 0.1, n = 3,
            default.units = "inches")
```

`pageLayoutRow` *Generate row positions for a number of plot elements with a specified height and space between them*

Description

Generate row positions for a number of plot elements with a specified height and space between them

Usage

```
pageLayoutRow(y, height, space, n, default.units = "inches")
```

Arguments

`y`	A numeric or unit object specifying the starting row y-position.
`height`	A numeric or unit object specifying the height of rows.
`space`	A numeric or unit object specifying the space between rows.
`n`	An integer specifying the number of elements to generate row positions for.
`default.units`	A string indicating the default units to use if `y`, `h`, or `s` are only given as numerics. Default value is `default.units = "inches"`.

Value

Returns a unit vector of page positions.

Examples

```
# Starting at 0.5 units, return a vector of positions for 3 objects that
# are 2 units in height with 0.1 units of space between them

pageLayoutRow(y = 0.5, height = 2, space = 0.1, n = 3,
            default.units = "inches")
```

`pagePlotPlace` *Place a plot that has been previously created but not drawn*

Description

Place a plot that has been previously created but not drawn

Usage

```
pagePlotPlace(
    plot,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL
)
```

Arguments

`plot`	Plot object to be placed, defined by the output of a plotgardener plotting function.
`x`	A numeric or unit object specifying plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying plot y-location. The character value will place the plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying plot width.
`height`	A numeric or unit object specifying plot height.
`just`	Justification of plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.

Value

Function will update dimensions of an input plot and return an updated plot object.

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create, but do not plot, square Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    draw = FALSE
)

## Create page
pageCreate(width = 3.75, height = 3.5, default.units = "inches")

## Place Hi-C plot on page
pagePlotPlace(
    plot = hicPlot,
    x = 0.25, y = 0.25, width = 3, height = 3,
    just = c("left", "top"),
    default.units = "inches", draw = TRUE
```

```
)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot,
    x = 3.4, y = 0.25, width = 0.12, height = 1.2,
    just = c("left", "top"), default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`pagePlotRemove` *Remove plotgardener plots and annotations*

Description

Remove plotgardener plots and annotations

Usage

```
pagePlotRemove(plot)
```

Arguments

`plot`	Plot object to be removed, defined by the output of a plotgardener plotting function.

Value

None.

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create page
pageCreate(width = 5.5, height = 4, default.units = "inches")

## Plot and place a square Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    x = 0.5, y = 0.5, width = 2.5, height = 2.5,
    just = c("left", "top"),
    default.units = "inches"
```

```
)

## Remove square Hi-C plot from page
pagePlotRemove(plot = hicPlot)
```

`pgParams` *pgParams: plotgardener parameters object*

Description

Creates an object of class "pgParams" that can be used by plotgardener functions. pgParams can be used to set a set of parameters to be shared across multiple functions.

Usage

```
pgParams(assembly = "hg38", gene = NULL, geneBuffer = NULL, ...)
```

Arguments

`assembly`	String defining the genome build. Default value is `assembly = "hg38"`.
`gene`	(optional) String naming a gene used to set the chrom, chromstart, and chromend arguments.
`geneBuffer`	(optional) Integer base-pairs to extend the start and end of a gene defined by argument gene. Can be one integer or a vector of length 2, where the first integer will extend the start of the gene and the second integer will extend the end of the gene.
`...`	This function will take any plotgardener function parameters and their values:

- `alpha`
- `altchrom`
- `altchromend`
- `altchromstart`
- `archHeight`
- `arrow`
- `at`
- `axis`
- `axisLine`
- `baseline`
- `baseline.color`
- `baseline.lwd`
- `bg`
- `binCap`
- `binSize`
- `border`
- `boxHeight`

- `boxWidth`
- `breaks`
- `BSgenome`
- `cex`
- `check.overlap`
- `chrom`
- `chromend`
- `chromstart`
- `clip`
- `collapse`
- `colorbyStrand`
- `colorTrans`
- `column`
- `commas`
- `curvature`
- `data`
- `default.units`
- `digits`
- `display.column`
- `draw`
- `extend`
- `file`
- `fill`
- `flip`
- `fontcolor`
- `fontsize`
- `geneBackground`
- `geneHighlights`
- `gene.id.column`
- `geneOrder`
- `Genome`
- `half`
- `height`
- `id`
- `id.lengths`
- `image`
- `interpolate`
- `just`
- `label`
- `labels`
- `leadSNP`
- `legend`

- `length`
- `limitLabel`
- `linecolor`
- `lineend`
- `linejoin`
- `lty`
- `lwd`
- `main`
- `margin`
- `matrix`
- `negData`
- `norm`
- `OrgDb`
- `orientation`
- `palette`
- `pch`
- `plot`
- `quiet`
- `r`
- `range`
- `resolution`
- `res_scale`
- `rot`
- `scale`
- `scientific`
- `scipen`
- `sequence`
- `shift`
- `showBands`
- `showGuides`
- `sigCol`
- `sigLine`
- `sigVal`
- `spaceHeight`
- `spaceWidth`
- `strand`
- `strandLabels`
- `strandSplit`
- `stroke`
- `style`
- `tcl`
- `ticks`

- `title`
- `TxDb`
- `type`
- `width`
- `x`
- `xgrid`
- `x0`
- `x1`
- `y`
- `ygrid`
- `ymax`
- `y0`
- `y1`
- `zrange`

Details

pgParams generates arguments from exported plotgardener functions at loading time of the package. Arguments defined in a pgParams object can be passed into the `params` argument of plotgardener functions. params arguments can be overridden from within plotgardener functions.

pgParams also provides an alternative region definition mechanism. Given a gene name and genome assembly, pgParams returns the appropriate "chrom", "chromstart", and "chromend" with a default buffer of `(gene length) / 2` added to the ends of the gene coordinates. The buffer amount can be set manually with the `geneBuffer` parameter. Buffer extending beyond the length of the chromosome will be trimmed.

Value

Returns an object of class `pgParams` containing plotgardener function arguments.

Examples

```
## Load hg19 genomic annotation packages
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
library("org.Hs.eg.db")

## Define parameters
p1 <- pgParams(gene = "IL1B", assembly = "hg19")

## Optionally add more parameters
p2 <- pgParams(fontsize = 10, assembly = "hg19")

## Combine parameters and pass them to a plotgardener function
plotGenes(params = c(p1, p2))
```

`plotCircle` *Plot a circle within a plotgardener layout*

Description

Plot a circle within a plotgardener layout

Usage

```
plotCircle(
    x,
    y,
    r,
    default.units = "inches",
    linecolor = "black",
    lwd = 1,
    lty = 1,
    fill = NA,
    alpha = 1,
    params = NULL,
    ...
)
```

Arguments

`x`	A numeric vector or unit object specifying circle x-locations relative to center.
`y`	A numeric vector, unit object, or a character vector of values containing a "b" combined with a numeric value specifying circle y-locations relative to center. The character vector will place circle y-locations relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`r`	A numeric vector or unit object specifying radii.
`default.units`	A string indicating the default units to use if `r`, `x`, or `y` are only given as numerics or numeric vectors. Default value is `default.units = "inches"`.
`linecolor`	A character value specifying circle line color. Default value is `linecolor = "black"`.
`lwd`	A numeric specifying circle line width. Default value is `lwd = 1`.
`lty`	A numeric specifying circle line type. Default value is `lty = 1`.
`fill`	A character value specifying circle fill color. Default value is `fill = NA`.
`alpha`	Numeric value specifying color transparency. Default value is `alpha = 1`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `circle` object containing relevant placement and grob information.

See Also

grid.circle

Examples

```
## Create a page
pageCreate(width = 2, height = 2, default.units = "inches")

## Plot two circles, one at a time
plotCircle(
    x = 0.6, y = 0.5, r = 0.1, fill = "black",
    default.units = "inches"
)
plotCircle(
    x = 1.4, y = 0.5, r = 0.1, fill = "black",
    default.units = "inches"
)

## Plot a vector of circles
xVals <- 1 + (0.5 * cos(seq(0, pi, pi / 8)))
yVals <- 1 + (0.5 * sin(seq(0, pi, pi / 8)))
plotCircle(x = xVals, y = yVals, r = 0.05, default.units = "inches")

## Hide page guides
pageGuideHide()
```

`plotgardener` *plotgardener: Coordinate-based Genomic Visualization Package for R*

Description

plotgardener is a coordinate-based genomic visualization package for R. It grants users the ability to programmatically produce complex, multi-paneled figures. Tailored for genomics, plotgardener allows users to visualize large complex genomic datasets and provides exquisite control over how plots are placed and arranged on a page.

Author(s)

Maintainer: Nicole Kramer <`nekramer@live.unc.edu`> (ORCID)

Authors:

- Eric S. Davis <`esdavis@live.unc.edu`> (ORCID)
- Craig Wenger <`craig.wenger@gmail.com`> (ORCID)
- Douglas H. Phanstiel <`douglas_phanstiel@med.unc.edu`> [copyright holder]

Other contributors:

- Sarah Parker <`sarmae@live.unc.edu`> [contributor]

- Erika Deoudes <`ed@erikadudes.com`> [artist]
- Michael Love <`milove@email.unc.edu`> [contributor]

See Also

Useful links:

- `https://phanstiellab.github.io/plotgardener`
- `https://github.com/PhanstielLab/plotgardener`

`plotGenes` *Plot a gene track for a specified genomic region*

Description

Plot a gene track for a specified genomic region

Usage

```
plotGenes(
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    fontsize = 8,
    fontcolor = c("#669fd9", "#abcc8e"),
    fill = c("#669fd9", "#abcc8e"),
    geneOrder = NULL,
    geneHighlights = NULL,
    geneBackground = "grey",
    strandLabels = TRUE,
    stroke = 0.1,
    bg = NA,
    x = NULL,
    y = NULL,
    width = NULL,
    height = unit(0.6, "inches"),
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL
)
```

Arguments

`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 8`.
`fontcolor`	A character value or vector of length 2 indicating the fontcolors for the plus strand and minus strand gene labels. The first value will color the plus strand gene labels and the second value will color the minus strand gene labels. Default value is `fontcolor = c("#669fd9", "#abcc8e")`.
`fill`	A character value or vector of length 2 indicating the strand fill colors for the plus strand and minus strand plot elements. The first value will color the plus strand plot elements and the second label will color the minus strand plot elements. Default value is `fill = c("#669fd9", "#abcc8e")`.
`geneOrder`	An ordered character vector of gene names to prioritize when labeling genes.
`geneHighlights`	A two-column dataframe with a column named "gene" containing gene names as strings to highlight and a named column "color" containing corresponding highlight colors.
`geneBackground`	If `geneHighlights` is given, a character value indicating the color for genes that are not highlighted.
`strandLabels`	A logical value indicating whether to include + and - strand labels to the left of the gene track.
`stroke`	A numeric value indicating the stroke width for gene body outlines. Default value is `stroke = 0.1`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`x`	A numeric or unit object specifying genes plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying genes plot y-location. The character value will place the genes plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying genes plot width.
`height`	A numeric or unit object specifying genes plot height.
`just`	Justification of genes plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.

Details

A gene track can be placed on a page by providing plot placement parameters:

```
plotGenes(chrom, chromstart = NULL, chromend = NULL,
            x, y, width, height, just = c("left", "top"),
            default.units = "inches")
```

This function can be used to quickly plot an unannotated gene track by ignoring plot placement parameters:

```
plotGenes(chrom, chromstart = NULL, chromend = NULL)
```

Genomic annotation information is acquired through TxDb and OrgDb-class packages, as determined through the `assembly` parameter. To avoid overcrowding of gene name labels, plotted gene labels are by default prioritized according to citation counts.

Value

Returns a `genes` object containing relevant genomic region, placement, and grob information.

See Also

assembly, genomes, defaultPackages

Examples

```
## Load hg19 genomic annotation packages
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
library("org.Hs.eg.db")

## Set genomic coordinates
paramssmall <- pgParams(
    chrom = "chr8",
    chromstart = 1, chromend = 3000000,
    assembly = "hg19", width = 7
)
paramsbig <- pgParams(
    chrom = "chr8",
    chromstart = 1, chromend = 146364022,
    assembly = "hg19", width = 7
)
## Set colors
cols <- c("#41B6C4", "#225EA8")

## Create page
pageCreate(width = 7.5, height = 3.5, default.units = "inches")

## Plot genes big
genesPlot <- plotGenes(
    params = paramsbig, fill = cols,
    fontcolor = cols,
```

```
    x = 0.25, y = 0.25, height = 0.75,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = genesPlot, x = 0.25, y = 1.0,
    scale = "Mb", just = c("left", "top")
)

## Plot genes small
genesPlot <- plotGenes(
    params = paramssmall,
    geneHighlights = data.frame(
        "gene" = c("DLGAP2"),
        "color" = c("#225EA8")
    ),
    geneBackground = "grey",
    x = 0.25, y = 2.25, height = 0.75,
    just = c("left", "top"), default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = genesPlot, x = 0.25, y = 3.0, scale = "Mb",
    just = c("left", "top")
)

## Hide page guides
pageGuideHide()
```

`plotGenomeLabel` *Plot genomic coordinates along the x or y-axis of a plotgardener plot*

Description

Plot genomic coordinates along the x or y-axis of a plotgardener plot

Usage

```
plotGenomeLabel(
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    fontsize = 10,
    fontcolor = "black",
    linecolor = "black",
```

```
    margin = unit(1, "mm"),
    scale = "bp",
    commas = TRUE,
    sequence = TRUE,
    boxWidth = 0.5,
    axis = "x",
    at = NULL,
    tcl = 0.5,
    x,
    y,
    length,
    just = c("left", "top"),
    default.units = "inches",
    params = NULL,
    ...
)
```

Arguments

`chrom`	Chromosome of genome label, as a string, or a character vector of chromosomes for a whole genome Manhattan plot.
`chromstart`	Integer start of genome label.
`chromend`	Integer end of genome label.
`assembly`	Default genome assembly as a string or a assembly object.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 10`.
`fontcolor`	A character value indicating the color for text. Default value is `fontcolor = "black"`.
`linecolor`	A character value indicating the color of the genome label axis. Default value is `linecolor = "black"`.
`margin`	A numeric or unit vector specifying space between axis and coordinate labels. Default value is `margin = unit(1, "mm")`,
`scale`	A character value indicating the scale of the coordinates along the genome label. Default value is `scale = "bp"`. Options are: • `"bp"`: base pairs. • `"Kb"`: kilobase pairs. 1 kilobase pair is equal to 1000 base pairs. • `"Mb"`: megabase pairs. 1 megabase pair is equal to 1000000 base pairs.
`commas`	A logical value indicating whether to include commas in start and stop labels. Default value is `commas = TRUE`.
`sequence`	A logical value indicating whether to include sequence information above the label of an x-axis (only at appropriate resolutions).
`boxWidth`	A numeric value indicating the width of the boxes representing sequence information at appropriate resolutions. Default value is `boxWidth = 0.5`.
`axis`	A character value indicating along which axis to add genome label. Sequence information will not be displayed along a y-axis. Default value is `axis = "x"`. Options are:

	- `"x"`: Genome label will be plotted along the x-axis. - `"y"`: Genome label will be plotted along the y-axis. This is typically used for a square Hi-C plot made with `plotHicSquare`.
`at`	A numeric vector of x-value locations for tick marks.
`tcl`	A numeric specifying the length of tickmarks as a fraction of text height. Default value is `tcl = 0.5`.
`x`	A numeric or unit object specifying genome label x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying genome label y-location. The character value will place the genome label y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`length`	A numeric or unit object specifying length of genome label axis.
`just`	Justification of genome label relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, or `length` are only given as numerics. Default value is `default.units = "inches"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters or digit specifications. See gpar and formatC.

Value

Returns a `genomeLabel` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load hg19 genomic annotation packages
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
library("org.Hs.eg.db")
library("BSgenome.Hsapiens.UCSC.hg19")

## Create page
pageCreate(width = 5, height = 3, default.units = "inches")

## Plot and place gene track on page
genesPlot <- plotGenes(
    chrom = "chr8",
    chromstart = 1000000, chromend = 2000000,
    assembly = "hg19", fill = c("grey", "grey"),
    fontcolor = c("grey", "grey"),
    x = 0.5, y = 0.25, width = 4, height = 1,
    just = c("left", "top"),
    default.units = "inches"
)
```

```
## Plot x-axis genome labels at different scales
plotGenomeLabel(
    chrom = "chr8",
    chromstart = 1000000, chromend = 2000000,
    assembly = "hg19",
    scale = "Mb",
    x = 0.5, y = 1.25, length = 4, just = c("left", "top"),
    default.units = "inches"
)
plotGenomeLabel(
    chrom = "chr8",
    chromstart = 1000000, chromend = 2000000,
    assembly = "hg19",
    scale = "Kb",
    x = 0.5, y = 1.5, length = 4, just = c("left", "top"),
    default.units = "inches"
)
plotGenomeLabel(
    chrom = "chr8",
    chromstart = 1000000, chromend = 2000000,
    assembly = "hg19",
    scale = "bp",
    x = 0.5, y = 1.75, length = 4, just = c("left", "top"),
    default.units = "inches"
)

## Plot a different genomic label region, zooming in enough
## to see base pairs
plotGenomeLabel(
    chrom = "chr8",
    chromstart = 1000000, chromend = 1000050,
    assembly = "hg19",
    x = 0.25, y = 2.2, length = 4.5
)
plotGenomeLabel(
    chrom = "chr8",
    chromstart = 1000000, chromend = 1000020,
    assembly = "hg19",
    x = 0, y = 2.6, length = 5
)

## Hide page guides
pageGuideHide()
```

`plotGG` *Plot a ggplot2 plot, gtable, or grob object in a plotgardener layout*

Description

Plot a ggplot2 plot, gtable, or grob object in a plotgardener layout

Usage

```
plotGG(
    plot,
    x,
    y,
    width,
    height,
    just = c("left", "top"),
    default.units = "inches",
    params = NULL
)
```

Arguments

`plot`	ggplot, gtable, or grob object.
`x`	A numeric or unit object specifying ggplot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying ggplot y-location. The character value will place the ggplot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying ggplot width.
`height`	A numeric or unit object specifying ggplot height.
`just`	Justification of ggplot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`params`	An optional pgParams object containing relevant function parameters.

Value

Returns a pg_gg object containing relevant placement and grob information.

See Also

ggplot

Examples

```
## Create a plot using ggplot2
library(ggplot2)
p <- ggplot(mtcars) +
    geom_point(aes(mpg, disp))

## Create a page
pageCreate(width = 4, height = 4, default.units = "inches")
```

```
## Place ggplot in page
plotGG(
    plot = p, x = 0.5, y = 0.5, width = 3, height = 3,
    just = c("left", "top"), default.units = "inches"
)

## Add title
plotText(
    label = "mtcars", fontsize = 14, fontface = "bold",
    x = 1, y = 0.35
)

## Hide page guides
pageGuideHide()
```

`plotHicRectangle` *Plot a triangular Hi-C interaction matrix in a rectangular format*

Description

Plot a triangular Hi-C interaction matrix in a rectangular format

Usage

```
plotHicRectangle(
    data,
    resolution = "auto",
    zrange = NULL,
    norm = "KR",
    matrix = "observed",
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    palette = colorRampPalette(brewer.pal(n = 9, "YlGnBu")),
    colorTrans = "linear",
    flip = FALSE,
    bg = NA,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    quiet = FALSE
)
```

Arguments

`data`	Path to .hic or .(m)cool file as a string or a 3-column dataframe of interaction counts in sparse upper triangular format.
`resolution`	A numeric specifying the width in basepairs of each pixel. For files, "auto" will attempt to choose a resolution based on the size of the region. For dataframes, "auto" will attempt to detect the resolution the dataframe contains.
`zrange`	A numeric vector of length 2 specifying the range of interaction scores to plot, where extreme values will be set to the max or min.
`norm`	Character value specifying hic data normalization method, if giving .hic or .(m)cool file. This value must be found in the .hic or .(m)cool file. Default value is `norm = "KR"`.
`matrix`	Character value indicating the type of matrix to output for .hic files. Default value is `matrix = "observed"`. Options are: • `"observed"`: Observed counts. • `"oe"`: Observed/expected counts. • `"log2oe"`: Log2 transformed observed/expected counts.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`palette`	A function describing the color palette to use for representing scale of interaction scores. Default value is `palette = colorRampPalette(brewer.pal(n = 9, "YlGnBu"))`.
`colorTrans`	A string specifying how to scale Hi-C colors. Options are "linear", "log", "log2", or "log10". Default value is `colorTrans = "linear"`.
`flip`	A logical indicating whether to flip the orientation of the Hi-C matrix over the x-axis. Default value is `flip = FALSE`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`x`	A numeric or unit object specifying rectangle Hi-C plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying rectangle Hi-C plot y-location. The character value will place the rectangle Hi-C plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying the width of the Hi-C plot rectangle.
`height`	A numeric or unit object specifying the height of the Hi-C plot rectangle.
`just`	Justification of rectangle Hi-C plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.

`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`quiet`	A logical indicating whether or not to print messages.

Details

This function is similar is plotHicTriangle but will fill in additional pixels around the the triangular portion of the plot to make a rectangle. The x-axis represents the genomic coordinates and the y-axis corresponds to distance in Hi-C bins.

A rectangle Hi-C plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotHicRectangle(data, chrom,
                    chromstart = NULL, chromend = NULL,
                    x, y, width, height, just = c("left", "top"),
                    default.units = "inches")
```

This function can also be used to quickly plot an unannotated rectangle Hi-C plot by ignoring plot placement parameters:

```
plotHicRectangle(data, chrom,
                    chromstart = NULL, chromend = NULL)
```

Value

Returns a `hicRectangle` object containing relevant genomic region, Hi-C data, placement, and grob information.

See Also

readHic, readCool, plotHicTriangle

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create a page
pageCreate(width = 6, height = 3.5, default.units = "inches")

## Plot and place rectangle Hi-C plot
hicPlot <- plotHicRectangle(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
```

```
    chromstart = 28950000, chromend = 29800000,
    assembly = "hg19",
    x = 0.5, y = 0.5, width = 5, height = 2.5,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate x-axis genome label
annoGenomeLabel(
    plot = hicPlot, scale = "Kb", x = 0.5, y = 3.03,
    just = c("left", "top")
)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot, x = 5.6, y = 0.5,
    width = 0.13, height = 1.5,
    just = c("left", "top")
)

## Hide page guides
pageGuideHide()
```

`plotHicSquare` *Plot a Hi-C interaction matrix in a square format*

Description

Plot a Hi-C interaction matrix in a square format

Usage

```
plotHicSquare(
    data,
    resolution = "auto",
    zrange = NULL,
    norm = "KR",
    matrix = "observed",
    chrom,
    chromstart = NULL,
    chromend = NULL,
    altchrom = NULL,
    altchromstart = NULL,
    altchromend = NULL,
    assembly = "hg38",
    palette = colorRampPalette(brewer.pal(n = 9, "YlGnBu")),
    colorTrans = "linear",
    half = "both",
    yaxisDir = "increase",
```

```
    bg = NA,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    quiet = FALSE
)
```

Arguments

`data`	Path to .hic or .(m)cool file as a string or a 3-column dataframe of interaction counts in sparse upper triangular format.
`resolution`	A numeric specifying the width in basepairs of each pixel. For files, "auto" will attempt to choose a resolution based on the size of the region. For dataframes, "auto" will attempt to detect the resolution the dataframe contains.
`zrange`	A numeric vector of length 2 specifying the range of interaction scores to plot, where extreme values will be set to the max or min.
`norm`	Character value specifying hic data normalization method, if giving .hic or .(m)cool file. This value must be found in the .hic or .(m)cool file. Default value is `norm = "KR"`.
`matrix`	Character value indicating the type of matrix to output for .hic files. Default value is `matrix = "observed"`. Options are: • `"observed"`: Observed counts. • `"oe"`: Observed/expected counts. • `"log2oe"`: Log2 transformed observed/expected counts.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`altchrom`	Alternate chromosome for off-diagonal plotting or interchromosomal plotting, as a string.
`altchromstart`	Alternate chromosome integer start position for off-diagonal plotting or interchromosomal plotting.
`altchromend`	Alternate chromosome integer end position for off-diagonal plotting or interchromosomal plotting.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`palette`	A function describing the color palette to use for representing scale of interaction scores. Default value is `palette = colorRampPalette(brewer.pal(n = 9, "YlGnBu"))`.
`colorTrans`	A string specifying how to scale Hi-C colors. Options are "linear", "log", "log2", or "log10". Default value is `colorTrans = "linear"`.

`half`	A character value indicating which diagonal regions to plot. For intrachromosomal plotting, options are `"both"`, `"top"`, or `"bottom"`. For off-diagonal or interchromosomal plotting, options are `"top"` or `"bottom"`. Default value is `half = "both"`. • `"both"`: Both diagonal halves. • `"top"`: Half above the diagonal. • `"bottom"`: Half below the diagonal.
`yaxisDir`	A string specifying the genomic direction of the y-axis. Options are `"increase"`, where the bottom left corner of the Hi-C map indicates the origin, and `"decrease"`, where the top left corner of the Hi-C map indicates the origin. This will change the direction of the diagonal for intrachromosomal maps. Default value is `yaxisDir = "increase"`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`x`	A numeric or unit object specifying square Hi-C plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying square Hi-C plot y-location. The character value will place the square Hi-C plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying square Hi-C plot width.
`height`	A numeric or unit object specifying square Hi-C plot height.
`just`	Justification of square Hi-C plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`quiet`	A logical indicating whether or not to print messages.

Details

A square Hi-C plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotHicSquare(data, chrom,
                chromstart = NULL, chromend = NULL,
                x, y, width, height, just = c("left", "top"),
                default.units = "inches")
```

This function can be used to quickly plot an unannotated square Hi-C plot by ignoring plot placement parameters:

```
plotHicSquare(data, chrom,
                chromstart = NULL, chromend = NULL)
```

Value

Returns a `hicSquare` object containing relevant genomic region, Hi-C data, placement, and grob information.

See Also

readHic, readCool

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create a page
pageCreate(width = 3, height = 3, default.units = "inches")

## Plot and place Hi-C plot
hicPlot <- plotHicSquare(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 0.5, width = 2, height = 2,
    just = c("left", "top"),
    default.units = "inches", bg = "black"
)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot, x = 2.6, y = 0.5,
    width = 0.12, height = 1.2,
    just = c("left", "top"), default.units = "inches"
)

## Annotate x-axis and y-axis genome labels
annoGenomeLabel(
    plot = hicPlot, scale = "Mb", axis = "x",
    x = 0.5, y = 2.53, just = c("left", "top")
)
annoGenomeLabel(
    plot = hicPlot, scale = "Mb", axis = "y",
    x = 0.47, y = 0.5, just = c("right", "top")
)

## Hide page guides
pageGuideHide()
```

`plotHicTriangle` *Plot a Hi-C interaction matrix in a triangular format*

Description

Plot a Hi-C interaction matrix in a triangular format

Usage

```
plotHicTriangle(
    data,
    resolution = "auto",
    zrange = NULL,
    norm = "KR",
    matrix = "observed",
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    palette = colorRampPalette(brewer.pal(n = 9, "YlGnBu")),
    colorTrans = "linear",
    flip = FALSE,
    bg = NA,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    quiet = FALSE
)
```

Arguments

`data`	Path to .hic or .(m)cool file as a string or a 3-column dataframe of interaction counts in sparse upper triangular format.
`resolution`	A numeric specifying the width in basepairs of each pixel. For files, "auto" will attempt to choose a resolution based on the size of the region. For dataframes, "auto" will attempt to detect the resolution the dataframe contains.
`zrange`	A numeric vector of length 2 specifying the range of interaction scores to plot, where extreme values will be set to the max or min.
`norm`	Character value specifying hic data normalization method, if giving .hic or .(m)cool file. This value must be found in the .hic or .(m)cool file. Default value is `norm = "KR"`.

`matrix`	Character value indicating the type of matrix to output for .hic files. Default value is `matrix = "observed"`. Options are: • `"observed"`: Observed counts. • `"oe"`: Observed/expected counts. • `"log2oe"`: Log2 transformed observed/expected counts.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`palette`	A function describing the color palette to use for representing scale of interaction scores. Default value is `palette = colorRampPalette(brewer.pal(n = 9, "YlGnBu"))`.
`colorTrans`	A string specifying how to scale Hi-C colors. Options are "linear", "log", "log2", or "log10". Default value is `colorTrans = "linear"`.
`flip`	A logical indicating whether to flip the orientation of the Hi-C matrix over the x-axis. Default value is `flip = FALSE`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`x`	A numeric or unit object specifying triangle Hi-C plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying triangle Hi-C plot y-location. The character value will place the triangle Hi-C plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying the bottom width of the Hi-C plot triangle.
`height`	A numeric or unit object specifying the height of the Hi-C plot triangle.
`just`	Justification of triangle Hi-C plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`quiet`	A logical indicating whether or not to print messages.

Details

In this orientation, the x-axis represents the genomic coordinates and the y-axis corresponds to distance in Hi-C bins.

A triangle Hi-C plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotHicTriangle(data, chrom,
                chromstart = NULL, chromend = NULL,
                x, y, width, height, just = c("left", "top"),
                default.units = "inches")
```

This function can also be used to quickly plot an unannotated triangle Hi-C plot by ignoring plot placement parameters:

```
plotHicTriangle(data, chrom,
                chromstart = NULL, chromend = NULL)
```

If `height` is $< 0.5 * width$, the top of the triangle will be cropped to the given `height`.

Value

Returns a `hicTriangle` object containing relevant genomic region, Hi-C data, placement, and grob information.

See Also

readHic, readCool

Examples

```
## Load Hi-C data
library(plotgardenerData)
data("IMR90_HiC_10kb")

## Create a page
pageCreate(width = 4, height = 2.5, default.units = "inches")

## Plot and place triangle Hi-C plot
hicPlot <- plotHicTriangle(
    data = IMR90_HiC_10kb, resolution = 10000,
    zrange = c(0, 70),
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 2, y = 0.5, width = 3, height = 1.5,
    just = "top", default.units = "inches"
)

## Annotate x-axis genome label
annoGenomeLabel(
    plot = hicPlot, scale = "Mb", x = 0.5, y = 2.03,
    just = c("left", "top")
)

## Annotate heatmap legend
annoHeatmapLegend(
    plot = hicPlot, x = 3.5, y = 0.5,
    width = 0.13, height = 1.2,
```

```
    just = c("right", "top")
)

## Hide page guides
pageGuideHide()
```

---

plotIdeogram *Plot a chromosome ideogram with or without cytobands*

Description

Plot a chromosome ideogram with or without cytobands

Usage

```
plotIdeogram(
    chrom,
    assembly = "hg38",
    data = NULL,
    orientation = "h",
    flip = FALSE,
    showBands = TRUE,
    fill = NULL,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL
)
```

Arguments

`chrom`	Chromosome to be plotted, as a string.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`data`	Custom cytoband data, as a dataframe with the following columns: "seqnames", "start", "end", "width", "strand", "name", "gieStain".
`orientation`	Character value indicating the orientation of the ideogram. Default value is `orientation = "h"`. Options are: • `"v"`: Vertical ideogram orientation. • `"h"`: Horizontal ideogram orientation.

`flip`	Logical value indicating whether to reverse the start and end positions of the ideogram. If `flip = FALSE` (default), the ideogram starts at the top/left and ends at the bottom/right. If `flip = TRUE`, the start and end positions are reversed, meaning the ideogram starts at the bottom/right and ends at the top/left.
`showBands`	Logical value indicating whether to draw colored cytobands within ideogram. Default value is `showBands = TRUE`.
`fill`	(optional) A vector specifying alternate colors for cytoband stains. To change specific gieStain values (i.e. gneg, gpos, etc.) to specific colors, this vector can be named. This vector must have the same number of colors as there are gieStain values for each genome assembly.
`x`	A numeric or unit object specifying ideogram x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying ideogram y-location. The character value will place the ideogram y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying ideogram width.
`height`	A numeric or unit object specifying ideogram height.
`just`	Justification of ideogram relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.

Details

An ideogram can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotIdeogram(chrom,
                x, y, width, height, just = c("left", "top"),
                default.units = "inches")
```

This function can also be used to quickly plot an unannotated ideogram by ignoring plot placement parameters:

```
plotIdeogram(chrom)
```

If no data is provided, Giemsa stain band data will first try to fetch UCSC with AnnotationHub. The results are cached for faster access, but these cached items can be deleted. If no internet connection is available and AnnotationHub has not previously cached the data, custom Giemsa stain band data can be loaded with the ‘data‘ parameter.

Value

Returns a `ideogram` object containing relevant genomic region, placement, and grob information.

See Also

AnnotationHub

Examples

```
## Load Giemsa stain band information and genomic
## annotation data for hg19 genome assembly
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(AnnotationHub)

## Create page
pageCreate(width = 4.5, height = 1, default.units = "inches")

## Plot and place ideogram
ideogramPlot <- plotIdeogram(
    chrom = "chr2", assembly = "hg19",
    x = 0.25, y = 0.25, width = 4, height = 0.3,
    just = c("left", "top"),
    default.units = "inches"
)

## Plot text
plotText(
    label = "Chromosome 2", fontcolor = "dark grey",
    x = 4.25, y = 0.65, just = "right"
)

## Hide page guides
pageGuideHide()
```

`plotLegend` *Plot a legend*

Description

Plot a legend

Usage

```
plotLegend(
    legend,
    fill = NULL,
    pch = NULL,
    lty = NULL,
    orientation = "v",
```

```
    title = NULL,
    fontsize = 10,
    border = TRUE,
    bg = NA,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    ...
)
```

Arguments

`legend`	A character or expression vector to appear in the legend.
`fill`	If specified, this argument will produce boxes filled with the specified colors to appear beside the legend text.
`pch`	The plotting symbols appearing in the legend, as a numeric vector.
`lty`	The line types for lines appearing in the legend.
`orientation`	A string specifying legend orientation. Default value is `orientation = "v"`. Options are: • `"v"`: Vertical legend orientation. • `"h"`: Horizontal legend orientation.
`title`	A character value giving a title to be placed at the top of the legend.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 10`.
`border`	Logical value indicating whether to add a border around heatmap legend. Default value is `border = TRUE`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`x`	A numeric or unit object specifying legend x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying legend y-location. The character value will place the legend y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying legend width.
`height`	A numeric or unit object specifying legend height.
`just`	Justification of legend relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.

`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `legend` object containing relevant placement and grob information.

Examples

```
## Load BED data
library(plotgardenerData)
data("IMR90_ChIP_CTCF_reads")

## Create page
pageCreate(width = 7.5, height = 4, default.units = "inches")

## Plot a pileup plot, coloring elements by strand
pileupPlot <- plotRanges(
    data = IMR90_ChIP_CTCF_reads, chrom = "chr21",
    chromstart = 29072500, chromend = 29075000,
    assembly = "hg19",
    fill = colorby("strand", palette =
                colorRampPalette(c("steel blue", "light salmon"))),
    x = 0.5, y = 3.5, width = 6.5, height = 3.5,
    just = c("left", "bottom"),
    default.units = "inches"
)

## Add a legend depicting strand colors
legendPlot <- plotLegend(
    legend = c("- strand", "+ strand"),
    fill = c("steel blue", "light salmon"),
    border = FALSE,
    x = 5, y = 0.5, width = 1.5, height = 0.7,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = pileupPlot, x = 0.5, y = 3.5,
    just = c("left", "top")
)

## Hide page guides
pageGuideHide()
```

`plotManhattan` *Plot a Manhattan plot*

Description

Plot a Manhattan plot

Usage

```
plotManhattan(
    data,
    sigVal = 5e-08,
    chrom = NULL,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    fill = "black",
    pch = 19,
    cex = 0.25,
    leadSNP = NULL,
    sigLine = FALSE,
    sigCol = NULL,
    trans = "-log10",
    range = NULL,
    yscale_reverse = FALSE,
    space = 0.01,
    bg = NA,
    baseline = FALSE,
    baseline.color = "grey",
    baseline.lwd = 1,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    flip = FALSE,
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    ...
)
```

Arguments

`data`	Data to be plotted, as a character value specifying a file path of GWAS data, a dataframe, or a GRanges object. Each of these data types must have the following columns:

- `"chrom"`: Chromosome names. This column must be a character.
- `"pos"`: Chromosomal position. This column must be an integer or numeric.
- `"p"`: p-value or similar measure. This column must be numeric. Values will be converted according to the `trans` parameter.
- `"snp"`(optional): SNP name or rsid. This column should be a character.

`sigVal`	A numeric specifying the significance level of p-values. Along with data p-values, this value will be converted according to the `trans` parameter. Default value is `sigVal = 5e-08`.
`chrom`	Chromosome of region to be plotted, as a string. If left `NULL`, all chromosomes found in data will be plotted.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`fill`	A single character value, a vector, or a colorby object specifying fill colors of data points. For a Manhattan plot with multiple chromosomes, a vector of colors will be used to color points of different chromosomes. Default value is `fill = "black"`.
`pch`	A numeric value or numeric vector specifying point symbols. If colorby object is supplied for `fill`, point symbols will be mapped to `colorby` values. Default value is `pch = 19`.
`cex`	A numeric indicating the amount by which points should be scaled relative to the default. Default value is `cex = 0.25`.
`leadSNP`	A list specifying the lead SNP in the desired region and any associated aesthetic features of the lead SNP data point and text label. The lead SNP should be specified as a character with the name slot `"snp"` in the list. Accepted lead SNP aesthetic features in the list include `fill`, `pch`, `cex`, `fontcolor`, and `fontsize`.
`sigLine`	Logical value indicating whether to draw a line at the significance level indicated with `sigVal`. Default value is `sigLine = FALSE`.
`sigCol`	Single character value specifying the color of significant data points.
`trans`	Character value specifying the transformation to apply to the "p" column plotted along the y-axis. For no transformation, set value to the empty character "". Default value is `trans = "-log10"`.
`range`	A numeric vector of length 2 specifying the y-range of p-values to plot (c(min, max)).
`yscale_reverse`	Logical value indicating whether to reverse the y-scale and order points from max to min.
`space`	A numeric value indicating the space between each chromosome as a fraction of the width of the plot, if plotting multiple chromosomes. Default value is `space = 0.01`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`baseline`	Logical value indicating whether to include a baseline along the x-axis. Default value is `baseline = FALSE`.

`baseline.color`	Baseline color. Default value is `baseline.color = "grey"`.
`baseline.lwd`	Baseline line width. Default value is `baseline.lwd = 1`.
`x`	A numeric or unit object specifying Manhattan plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying Manhattan plot y-location. The character value will place the Manhattan plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying Manhattan plot width.
`height`	A numeric or unit object specifying Manhattan plot height.
`just`	Justification of Manhattan plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`flip`	Logical value indicating whether to reflect Manhattan plot over the x-axis. Default value is `flip = FALSE`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Details

A Manhattan plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotManhattan(data,
                chrom = NULL,
                chromstart = NULL, chromend = NULL,
                x, y, width, height, just = c("left", "top"),
                default.units = "inches")
```

This function can also be used to quickly plot an unannotated Manhattan plot by ignoring plot placement parameters:

```
plotManhattan(data,
                chrom = NULL,
                chromstart = NULL, chromend = NULL)
```

Value

Returns a `manhattan` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load genomic assembly information
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
## Load GWAS data
library(plotgardenerData)
data("hg19_insulin_GWAS")

## Create a page
pageCreate(width = 7.5, height = 4.5, default.units = "inches")

## Plot all GWAS data
manhattanPlot <- plotManhattan(
    data = hg19_insulin_GWAS, assembly = "hg19",
    fill = c("grey", "#37a7db"),
    sigLine = TRUE,
    trans = "-log10",
    col = "grey", lty = 2, range = c(0, 14),
    x = 0.5, y = 0, width = 6.5, height = 2,
    just = c("left", "top"),
    default.units = "inches"
)
## Annotate genome label
annoGenomeLabel(
    plot = manhattanPlot, x = 0.5, y = 2, fontsize = 8,
    just = c("left", "top"),
    default.units = "inches"
)
plotText(
    label = "Chromosome", fontsize = 8,
    x = 3.75, y = 2.20, just = "center", default.units = "inches"
)

## Annotate y-axis
annoYaxis(
    plot = manhattanPlot, at = c(0, 2, 4, 6, 8, 10, 12, 14),
    axisLine = TRUE, fontsize = 8
)

## Plot y-axis label
plotText(
    label = "-log10(p-value)", x = 0.15, y = 1, rot = 90,
    fontsize = 8, fontface = "bold", just = "center",
    default.units = "inches"
)


## Plot GWAS data zooming in on chromosome 11
## highlighting a lead SNP, and coloring by LD score
hg19_insulin_GWAS$LD <- as.numeric(hg19_insulin_GWAS$LD)
## Group LD column into LD ranges
hg19_insulin_GWAS <- as.data.frame(dplyr::group_by(hg19_insulin_GWAS,
                                        LDgrp = cut(
```

```
                                        hg19_insulin_GWAS$LD,
                                        c(0, 0.2, 0.4, 0.6, 0.8, 1))))
hg19_insulin_GWAS$LDgrp <- addNA(hg19_insulin_GWAS$LDgrp)
leadSNP_p <- min(hg19_insulin_GWAS[
    which(hg19_insulin_GWAS$chrom == "chr11"), ]$p)
leadSNP <- hg19_insulin_GWAS[which(hg19_insulin_GWAS$p == leadSNP_p), ]$snp
chr11_manhattanPlot <- plotManhattan(
    data = hg19_insulin_GWAS, chrom = "chr11",
    chromstart = 60000000,
    chromend = 130000000,
    assembly = "hg19",
    fill = colorby("LDgrp",
    palette = colorRampPalette(c(
        "#1f4297",
        "#37a7db", "green",
        "orange", "red", "grey"
    ))),
    trans = "-log10",
    sigLine = TRUE, col = "grey",
    lty = 2, range = c(0, 16),
    leadSNP = list(
        snp = leadSNP,
        pch = 18,
        cex = 0.75,
        fill = "#7ecdbb",
        fontsize = 8
    ),
    x = 0.5, y = 2.5, width = 6.5,
    height = 1.5,
    just = c("left", "top"),
    default.units = "inches"
)

## Plot legend for LD scores
plotLegend(
    legend = c(
        "LD Ref Var",
        paste("0.4", ">", "r^2",
        "", ">=", "0.2"),
        paste("0.2", ">", "r^2",
        "", ">=", "0"),
        "no LD data"
    ),
    fill = c("#7ecdbb", "#37a7db", "#1f4297", "grey"), cex = 0.75,
    pch = c(18, 19, 19, 19), border = FALSE, x = 7, y = 2.5,
    width = 1.5, height = 0.6, just = c("right", "top"),
    default.units = "inches"
)


## Annotate genome label
annoGenomeLabel(
    plot = chr11_manhattanPlot, x = 0.5, y = 4.01,
```

```
        fontsize = 8, scale = "Mb",
        just = c("left", "top"), default.units = "inches"
)

## Annotate y-axis
annoYaxis(
    plot = chr11_manhattanPlot,
    at = c(0, 2, 4, 6, 8, 10, 12, 14, 16),
    axisLine = TRUE, fontsize = 8
)

## Plot y-axis label
plotText(
    label = "-log10(p-value)", x = 0.15, y = 3.25, rot = 90,
    fontsize = 8, fontface = "bold", just = "center",
    default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`plotMultiSignal` *Plot multiple signal tracks in line with each other*

Description

Plot multiple signal tracks in line with each other

Usage

```
plotMultiSignal(
   data,
   binSize = NA,
   binCap = TRUE,
   negData = FALSE,
   chrom,
   chromstart = NULL,
   chromend = NULL,
   assembly = "hg38",
   linecolor= "#37a7db",
   fill = NA,
   ymax = 1,
   range = NULL,
   scale = FALSE,
   label = NULL,
   bg = NA,
   baseline = TRUE,
   baseline.color = "grey",
   baseline.lwd = 1,
```

```
    orientation = "h",
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    gapdistance = .2,
    default.units = "inches",
    draw = TRUE,
    params = NULL, ...
)
```

Arguments

`data`	List of data to be plotted as character values specifying multiple bigwig file paths, dataframes in BED format, or GRanges objects with metadata column `score`.
`binSize`	A numeric specifying the length of each data bin in basepairs. Default value is `binSize = NA`.
`binCap`	A logical value indicating whether the function will limit the number of data bins to 8,000. Default value is `binCap = TRUE`.
`negData`	A logical value indicating whether any of the data has both positive and negative scores and the y-axis of each signal track should be split. Default value is `negData = FALSE`.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`linecolor`	A character value or vector of character values specifying the line color(s) outlining the signal tracks. Default value is `linecolor = "#37a7db"`.
`fill`	A character value or vector specifying the fill color(s) of the signal tracks. Default value is `fill = NA`.
`ymax`	A numeric specifying the fraction of the max y-value to set as the height of each plot. Default value is `ymax = 1`.
`range`	A numeric vector of length 2 specifying the y-range of data to plot (c(min, max)) in each signal track. If `range = NULL`, an optimal range for all signal tracks will be calculated.
`scale`	A logical value indicating whether to include a data scale label in the top left corner of each plot. Default value is `scale = FALSE`.
`label`	An optional character vector to conveniently add text labels to signal tracks. If `scale = TRUE`, the labels will be drawn in the top right of the signal tracks. Otherwise, the label will be drawn in the top left of the plot. For more customizable labels, use plotText.
`bg`	Character value indicating background color. Default value is `bg = NA`.

`baseline`	Logical value indicating whether to include a baseline along the x-axis. Default value is `baseline = TRUE`.
`baseline.color`	Baseline color. Default value is `baseline.color = "grey"`.
`baseline.lwd`	Baseline line width. Default value is `baseline.lwd = 1`.
`orientation`	A string specifying signal track orientations. Default value is `orientation = "h"`. Options are: • `"v"`: Vertical signal track orientations, where signal tracks will be stacked from left to right. • `"h"`: Horizontal signal track orientations, where signal tracks will be stacked from top to bottom.
`x`	A numeric vector or unit object specifying the overall multisignal x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying overall multisignal plot y-location. The character value will place the multisignal plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying overall multisignal plot width.
`height`	A numeric or unit object specifying overall multisignal plot height.
`just`	Justification of overall multisignal plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`gapdistance`	A numeric or unit object specifying space between plots. Default value is `gapdistance = 0.2`.
`default.units`	A string indicating the default units to use if `x` or `y` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a list of `signal` objects containing relevant genomic region, placement, and grob information for each signal track.

Examples

```
library("plotgardenerData")
data("GM12878_ChIP_CTCF_signal")
data("IMR90_ChIP_CTCF_signal")
data("GM12878_ChIP_H3K27ac_signal")
data("IMR90_ChIP_H3K27ac_signal")
library("RColorBrewer")
```

```
## List of multiple signal datasets
signalList <- list(GM12878_ChIP_CTCF_signal, GM12878_ChIP_H3K27ac_signal,
    IMR90_ChIP_CTCF_signal, IMR90_ChIP_H3K27ac_signal)

## Create page
pageCreate(width = 6.9, height = 3.5, default.units = "inches")

## Plot multiple signals
multisignal <- plotMultiSignal(signalList, chrom = "chr21",
                               chromstart = 28150000, chromend = 29150000,
                               linecolor = c(brewer.pal(n = 9,"YlGnBu")[4],
                                             brewer.pal(n = 9,"YlGnBu")[5],
                                             brewer.pal(n = 9,"YlGnBu")[6],
                                             brewer.pal(n = 9,"YlGnBu")[7]),
                               label = c("GM12878 CTCF", "GM12878 H3K27ac",
                                         "IMR90 CTCF", "IMR90 H3K27ac"),
                               assembly = "hg19",
                               x = 0.2, y = 0.2,
                               width = 6.5, height = 3,
                               default.units = "inches",
                               gapdistance = 0.1)

## Plot genome label
plotGenomeLabel(
    chrom = "chr21",
    chromstart = 28150000, chromend = 29150000,
    assembly = "hg19",
    scale = "Kb",
    x = 0.2, y = 3.25, length = 6.5,
    default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`plotPairs` *Plot paired-end genomic range elements*

Description

Plot paired-end genomic range elements

Usage

```
plotPairs(
    data,
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
```

```
    fill = "#1f4297",
    linecolor = NA,
    bg = NA,
    boxHeight = unit(2, "mm"),
    spaceWidth = 0.02,
    spaceHeight = 0.3,
    limitLabel = TRUE,
    baseline = FALSE,
    baseline.color = "grey",
    baseline.lwd = 1,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    ...
)
```

Arguments

`data`	A string specifying the BEDPE file path, a dataframe in BEDPE format specifying data to be plotted, or a GInteractions object.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`fill`	A single character value, a vector, or a colorby object specifying fill colors of paired range elements. Default value is `fill = "#1f4297"`.
`linecolor`	A single character value, a vector, or a colorby object specifying the color of the lines outlining paired range elements. Default value is `linecolor = NA`. Special options include: • `NA`: No line color. • `"fill"`: Same color as `fill`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`boxHeight`	A numeric or unit object specifying height of boxes at either end of paired range elements. Default value is `boxHeight = unit(2, "mm")`.
`spaceWidth`	A numeric specifying the width of spacing between paired range elements, as a fraction of the plot's genomic range. Default value is `spaceWidth = 0.02`.
`spaceHeight`	A numeric specifying the height of space between boxes of paired range elements on different rows. Default value is `spaceHeight = 0.3`.

`limitLabel`	A logical value indicating whether to draw a "+" when not all elements can be plotted in the plotting space. Default value is `limitLabel = TRUE`.
`baseline`	Logical value indicating whether to include a baseline along the x-axis. Default value is `baseline = FALSE`.
`baseline.color`	Baseline color. Default value is `baseline.color = "grey"`.
`baseline.lwd`	Baseline line width. Default value is `baseline.lwd = 1`.
`x`	A numeric or unit object specifying paired range plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying paired range plot y-location. The character value will place the paired range plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying paired range plot width.
`height`	A numeric or unit object specifying paired range plot height.
`just`	Justification of paired range plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Details

#' A paired ranges plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotPairs(data, chrom,
            chromstart = NULL, chromend = NULL,
            x, y, width, height, just = c("left", "top"),
            default.units = "inches")
```

This function can also be used to quickly plot an unannotated paired ranges plot by ignoring plot placement parameters:

```
plotPairs(data, chrom,
            chromstart = NULL, chromend = NULL)
```

Value

Returns a `pairs` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load paired ranges data in BEDPE format
library(plotgardenerData)
data("IMR90_DNAloops_pairs")

## Set the coordinates
params <- pgParams(
    chrom = "chr21",
    chromstart = 27900000, chromend = 30700000,
    assembly = "hg19",
    width = 7
)

## Create a page
pageCreate(width = 7.5, height = 2.1, default.units = "inches")

## Add a length column
IMR90_DNAloops_pairs$length <-
        (IMR90_DNAloops_pairs$start2 - IMR90_DNAloops_pairs$start1) / 1000

## Plot the data
bedpePlot <- plotPairs(
    data = IMR90_DNAloops_pairs, params = params,
    fill = colorby("length", palette =
                colorRampPalette(c("dodgerblue2", "firebrick2"))),
    lwd = 2, spaceHeight = .7,
    x = 0.25, y = 0.25, height = 1.5,
    just = c("left", "top"), default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(plot = bedpePlot, x = 0.25, y = 1.78, scale = "Mb")

## Add heatmap legend
annoHeatmapLegend(
    plot = bedpePlot, fontcolor = "black",
    x = 7.0, y = 0.25,
    width = 0.10, height = 1, fontsize = 10
)

## Add heatmap legend label
plotText(
    label = "Kb", rot = 90, x = 6.9, y = 0.75,
    just = c("center", "center"), fontsize = 10
)

## Hide page guides
pageGuideHide()
```

`plotPairsArches` *Plot paired-end genomic range data in an arch style*

Description

Plot paired-end genomic range data in an arch style

Usage

```
plotPairsArches(
    data,
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    style = "2D",
    flip = FALSE,
    curvature = 5,
    archHeight = NULL,
    fill = "#1f4297",
    linecolor = NA,
    alpha = 0.4,
    bg = NA,
    clip = FALSE,
    clip.noAnchor = TRUE,
    range = NULL,
    baseline = FALSE,
    baseline.color = "grey",
    baseline.lwd = 1,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    ...
)
```

Arguments

`data`	A string specifying the BEDPE file path, a dataframe in BEDPE format specifying data to be plotted, or a GInteractions object.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`style`	Character value describing the style of arches. Default value is `style = "2D"`. Options are:

	- `"2D"`: Arches will be drawn in a 2-dimensional style. - `"3D"`: Arches will be drawn in a 3-dimensional style.
`flip`	Logical value indicating whether to reflect arches over the x-axis. Default value is `flip = FALSE`.
`curvature`	Numeric indicating the number of points along the arch curvature. Default value is `curvature = 5`.
`archHeight`	Single numeric value, numeric vector, or column name in data specifying the arch heights. When NULL, all arches will be the same height, filling up the given plot area.
`fill`	A single character value, a vector, or a colorby object specifying fill colors of arches. Default value is `fill = #1f4297"`.
`linecolor`	A single character value, a vector, or a colorby object specifying the color of the lines outlining arches. Default value is `linecolor = NA`. Special options include: - `NA`: No line color. - `"fill"`: Same color as `fill`.
`alpha`	Numeric value specifying transparency. Default value is `alpha = 0.4`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`clip`	A logical value indicating whether to clip any arches that get cutoff in the given genomic region. Default value is `clip = FALSE`.
`clip.noAnchor`	A logical value indicating whether to clip any arches that overlap the given genomic region but do not have an anchor in that region. Default value is `clip.noAnchor = TRUE`.
`range`	A numeric vector of length 2 specifying the y-range of `archHeight` to plot (c(min, max)).
`baseline`	Logical value indicating whether to include a baseline along the x-axis. Default value is `baseline = FALSE`.
`baseline.color`	Baseline color. Default value is `baseline.color = "grey"`.
`baseline.lwd`	Baseline line width. Default value is `baseline.lwd = 1`.
`x`	A numeric or unit object specifying pair arches plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying BEDPE arches plot y-location. The character value will place the pair arches plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying pair arches plot width.
`height`	A numeric or unit object specifying pair arches plot height.
`just`	Justification of pair arches plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.

`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Details

A pair arches plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotPairsArches(data chrom,
                chromstart = NULL, chromend = NULL,
                x, y, width, height, just = c("left", "top"),
                default.units = "inches")
```

This function can also be used to quickly plot an unannotated pair arches plot by ignoring plot placement parameters:

```
plotPairsArches(data, chrom,
                chromstart = NULL, chromend = NULL)
```

Value

Returns a `arches` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load paired ranges data in BEDPE format
library(plotgardenerData)
data("IMR90_DNAloops_pairs")

## Set the coordinates
params <- pgParams(
    chrom = "chr21",
    chromstart = 27900000, chromend = 30700000,
    assembly = "hg19",
    width = 7
)

## Create a page
pageCreate(width = 7.5, height = 2.1, default.units = "inches")

## Add a length column to color by
IMR90_DNAloops_pairs$length <-
        (IMR90_DNAloops_pairs$start2 - IMR90_DNAloops_pairs$start1) / 1000

## Translate lengths into heights
IMR90_DNAloops_pairs$h <-
        IMR90_DNAloops_pairs$length / max(IMR90_DNAloops_pairs$length)

## Plot the data
```

```
archPlot <- plotPairsArches(
    data = IMR90_DNAloops_pairs, params = params,
    fill = colorby("length", palette =
                colorRampPalette(c("dodgerblue2", "firebrick2"))),
    linecolor = "fill",
    archHeight = "h", alpha = 1,
    x = 0.25, y = 0.25, height = 1.5,
    just = c("left", "top"),
    default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(plot = archPlot, x = 0.25, y = 1.78, scale = "Mb")


## Annotate heatmap legend
annoHeatmapLegend(
    plot = archPlot, fontcolor = "black",
    x = 7.0, y = 0.25,
    width = 0.10, height = 1, fontsize = 10
)

## Add the heatmap legend title
plotText(
    label = "Kb", rot = 90, x = 6.9, y = 0.75,
    just = c("center", "center"),
    fontsize = 10
)


## Hide page guides
pageGuideHide()
```

`plotPolygon` *Plot a polygon within a plotgardener layout*

Description

Plot a polygon within a plotgardener layout

Usage

```
plotPolygon(
    x,
    y,
    default.units = "inches",
    linecolor = "black",
    lwd = 1,
    lty = 1,
    fill = NA,
```

```
    alpha = 1,
    id = NULL,
    id.lengths = NULL,
    params = NULL,
    ...
)
```

Arguments

`x`	A numeric vector or unit object specifying polygon vertex x-locations.
`y`	A numeric vector, unit object, or a character vector of values containing a "b" combined with a numeric value specifying polygon vertex y-locations. The character vector will place polygon vertex y-locations relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`default.units`	A string indicating the default units to use if `x` or `y` are only given as numeric vectors. Default value is `default.units = "inches"`.
`linecolor`	A character value specifying polygon line color. Default value is `linecolor = "black"`.
`lwd`	A numeric specifying polygon line width. Default value is `lwd = 1`.
`lty`	A numeric specifying polygon line type. Default value is `lty = 1`.
`fill`	A character value specifying polygon fill color. Default value is `fill = NA`.
`alpha`	Numeric value specifying color transparency. Default value is `alpha = 1`.
`id`	A numeric vector used to separate locations in `x` and `y` into multiple polygons. All locations with the same `id` belong to the same polygon.
`id.lengths`	A numeric vector used to separate locations in `x` and `y` into multiple polygons. Specifies consecutive blocks of locations which make up separate polygons.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `polygon` object containing relevant placement and grob information.

See Also

grid.polygon

Examples

```
## Create a page
pageCreate(width = 7.5, height = 6, default.units = "inches")

## Plot complex polygons one at a time
plotPolygon(
    x = c(2.6, 4.65, 4.75, 6.05, 1.4, 1.3),
    y = c(2.5, 3.1, 3.5, 4, 3.15, 2.8),
    fill = "#4a168e", linecolor = NA
```

```
)

plotPolygon(
    x = c(4.65, 4.75, 6.05, 5.05, 4.4),
    y = c(3.1, 3.5, 4, 1.45, 1.2),
    fill = "#9d28b0", linecolor = NA
)

## Plot multiple triangles with different id's and colors
plotPolygon(
    x = c(
        0.45, 6.05, 3, 3, 6.05, 5.25, 4.4, 5.05, 4.95,
        1.3, 2.6, 1, 4.4, 4.95, 5, 4.95, 5, 6.25
    ),
    y = c(
        2.85, 4, 5.55, 5.55, 4, 5.55, 1.2, 1.45, 1.1,
        2.8, 2.5, 2.1, 1.2, 1.1, 0.45, 1.1, 0.45, 1.1
    ),
    id = c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6),
    fill = c(
        "#ce93d9", "#bb6ac9", "#4a168e",
        "#7b1fa0", "#bb6ac9", "#ce93d9"
    ),
    linecolor = NA
)

## Hide page guides
pageGuideHide()
```

`plotRanges` *Plot genomic range elements in a pileup or collapsed format*

Description

Plot genomic range elements in a pileup or collapsed format

Usage

```
plotRanges(
    data,
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    fill = "#7ecdbb",
    linecolor = NA,
    order = "width",
    collapse = FALSE,
    boxHeight = unit(2, "mm"),
```

```
    spaceWidth = 0.02,
    spaceHeight = 0.3,
    limitLabel = TRUE,
    strandSplit = FALSE,
    bg = NA,
    baseline = FALSE,
    baseline.color = "grey",
    baseline.lwd = 1,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
    ...
)
```

Arguments

`data`	Data to be plotted; as a character value specifying a BED file path, a data frame in BED format, a character value specifying a .bam file path where a bam index file (.bam.bai) is in the same directory, or a GRanges object.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`fill`	A single character value, a vector, or a colorby object specifying fill colors of range elements. Default value is `fill = "#7ecdbb"`.
`linecolor`	A single character value, a vector, or a colorby object specifying the color of the lines outlining range elements. Default value is `linecolor = NA`. Special options include: • `NA`: No line color. • `"fill"`: Same color as `fill`. .
`order`	A character value specifying how to order pileup data before assigning rows. Default value is `order = "width"`. Options include: • `"width"`: Ordered by decreasing width of elements. • `"random"`: Ordered randomly in each function call. .
`collapse`	A logical value indicating whether to collapse range elements into a single row, or into two rows if `strandSplit = TRUE`. If `collapse = TRUE`, `boxHeight` will be ignored and elements will be the height of the entire plot if `strandSplit =`

	`FALSE` or be the height of half of the entire plot if `strandSplit = TRUE`. Default value is `collapse = FALSE`.
`boxHeight`	A numeric or unit object specifying height of range element boxes. Default value is `boxHeight = unit(2, "mm")`.
`spaceWidth`	A numeric value specifying the width of minimum spacing between range element boxes, as a fraction of the plot's genomic range. Default value is `spaceWidth = 0.02`.
`spaceHeight`	A numeric value specifying the height of spacing between range element boxes on different rows, as a fraction of boxHeight. Default value is `spaceHeight = 0.3`.
`limitLabel`	A logical value indicating whether to draw a "+" when not all elements can be plotted in the plotting space. Default value is `limitLabel = TRUE`.
`strandSplit`	A logical value indicating whether plus and minus-stranded elements should be separated. Elements can only be split by strand if a `strand` column is found in `data`. Default value is `strandSplit = FALSE`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`baseline`	Logical value indicating whether to include a baseline along the x-axis. Default value is `baseline = FALSE`.
`baseline.color`	Baseline color. Default value is `baseline.color = "grey"`.
`baseline.lwd`	Baseline line width. Default value is `baseline.lwd = 1`.
`x`	A numeric or unit object specifying ranges plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying ranges plot y-location. The character value will place the ranges plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying ranges plot width.
`height`	A numeric or unit object specifying ranges plot height.
`just`	Justification of ranges plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Details

A ranges plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotRanges(data, chrom,
            chromstart = NULL, chromend = NULL,
            x, y, width, height, just = c("left", "top"),
            default.units = "inches")
```

This function can also be used to quickly plot an unannotated BED plot by ignoring plot placement parameters:

```
plotRanges(data, chrom,
            chromstart = NULL, chromend = NULL)
```

Value

Returns a `ranges` object containing relevant genomic region, coloring data, placement, and grob information.

Examples

```
## Load ranges data in BED format
library(plotgardenerData)
data("IMR90_ChIP_CTCF_reads")

## Create page
pageCreate(width = 7.5, height = 5, default.units = "inches")

## Plot and place a pileup ranges plot
pileupPlot <- plotRanges(
    data = IMR90_ChIP_CTCF_reads, chrom = "chr21",
    chromstart = 29073000, chromend = 29074000,
    assembly = "hg19",
    order = "random",
    fill = colorby("strand", palette =
                colorRampPalette(c("#7ecdbb", "#37a7db"))),
    strandSplit = TRUE,
    x = 0.5, y = 0.25, width = 6.5, height = 4.25,
    just = c("left", "top"), default.units = "inches"
)

## Annotate genome label
annoGenomeLabel(
    plot = pileupPlot, x = 0.5, y = 4.5,
    just = c("left", "top")
)

## Add text labels
plotText(
    label = "+ strand", fontcolor = "#37a7db", fontsize = 12,
    x = 0.5, y = 1.25, just = "left"
)
plotText(
    label = "- strand", fontcolor = "#7ecdbb", fontsize = 12,
    x = 0.5, y = 3.5, just = "left"
```

```
)

## Hide page guides
pageGuideHide()
```

`plotRaster` *Plot a raster object within a plotgardener layout*

Description

Plot a raster object within a plotgardener layout

Usage

```
plotRaster(
    image,
    x,
    y,
    width,
    height,
    just = "center",
    default.units = "inches",
    interpolate = TRUE,
    params = NULL,
    ...
)
```

Arguments

`image`	Any R object that can be coerced to a raster object.
`x`	A numeric vector or unit object specifying raster x-locations.
`y`	A numeric vector, unit object, or a character vector of values containing a "b" combined with a numeric value specifying raster y-locations. The character vector will place raster y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric vector or unit object specifying raster widths.
`height`	A numeric vector or unit object specifying raster heights.
`just`	Justification of text relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = "center"`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics or numeric vectors. Default value is `default.units = "inches"`.
`interpolate`	A logical value indicating whether to linearly interpolate the image. Default value is `interpolate = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `raster` object containing relevant placement and grob information.

See Also

grid.raster

Examples

```
library(png)

## Load images

pg_type <- readPNG(system.file("images",
    "pg-wordmark.png",
    package = "plotgardener"
))

gene_gnome <- readPNG(system.file("images",
    "pg-gnome-hole-shadow.png",
    package = "plotgardener"
))

rlogo <- readPNG(system.file("images", "Rlogo.png",
                package = "plotgardener"))

## Create page
pageCreate(width = 5, height = 6)

## Plot images
plotRaster(
    image = pg_type,
    x = 2.5, y = 0.25, width = 4, height = 1.5,
    just = "top"
)

plotRaster(
    image = gene_gnome,
    x = 2.5, y = 2.25, width = 3.5, height = 3.5,
    just = "top"
)

plotRaster(
    image = rlogo,
    x = 1, y = 1.5, width = 0.5, height = 0.45,
    just = c("left", "top")
)

## Hide page guies
pageGuideHide()
```

`plotRect` *Plot a rectangle within a plotgardener layout*

Description

Plot a rectangle within a plotgardener layout

Usage

```
plotRect(
    x,
    y,
    width,
    height,
    just = "center",
    default.units = "inches",
    linecolor = "black",
    lwd = 1,
    lty = 1,
    fill = NA,
    alpha = 1,
    params = NULL,
    ...
)
```

Arguments

`x`	A numeric vector or unit object specifying rectangle x-locations.
`y`	A numeric vector, unit object, or a character vector of values containing a "b" combined with a numeric value specifying rectangle y-locations. The character vector will place rectangle y-locations relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric vector or unit object specifying rectangle widths.
`height`	A numeric vector or unit object specifying rectangle heights.
`just`	Justification of rectangle relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = "center"`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, and `height` are only given as numerics or numeric vectors. Default value is `default.units = "inches"`.
`linecolor`	A character value specifying rectangle line color. Default value is `linecolor = "black"`.
`lwd`	A numeric specifying rectangle line width. Default value is `lwd = 1`.
`lty`	A numeric specifying rectangle line type. Default value is `lty = 1`.

`fill`	A character value specifying rectangle fill color. Default value is `fill = NA`.
`alpha`	Numeric value specifying color transparency. Default value is `alpha = 1`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `rect` object containing relevant placement and grob information.

See Also

grid.rect

Examples

```
## Create a page
pageCreate(width = 7.5, height = 6, default.units = "inches")

## Plot one rectangle with no fill
plotRect(
    x = 0.5, y = 0.5, width = 3, height = 3,
    just = c("left", "top"), default.units = "inches",
    lwd = 2, fill = NA
)


## Plot two rectangles with same width and height at different locations
plotRect(
    x = 4, y = c(0.5, 2.25), width = 3, height = 1.25,
    just = c("left", "top"), default.units = "inches",
    fill = "#7ecdbb"
)

## Plot two rectangles with different widths, heights,
## locations, and colors
plotRect(
    x = 3.75, y = c(4, 5.25), width = c(6.5, 4.5),
    height = c(1, 0.25),
    just = "top", default.units = "inches",
    fill = c("#7ecdbb", "#37a7db"), linecolor = NA, alpha = 0.4
)

## Hide page guides
pageGuideHide()
```

`plotSegments` *Draw a line segment within a plotgardener layout*

Description

Draw a line segment within a plotgardener layout

Usage

```
plotSegments(
    x0,
    y0,
    x1,
    y1,
    default.units = "inches",
    linecolor = "black",
    lwd = 1,
    lty = 1,
    lineend = "butt",
    linejoin = "mitre",
    arrow = NULL,
    params = NULL,
    ...
)
```

Arguments

`x0`	A numeric vector or unit object indicating the starting x-values of the line segments.
`y0`	A numeric vector, unit object, or a character vector of values containing a "b" combined with a numeric value specifying starting y-values of the line segments. The character vector will place starting y-values relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`x1`	A numeric vector or unit object indicating the stopping x-values of the line segments.
`y1`	A numeric vector, unit object, or a character vector of v alues containing a "b" combined with a numeric value specifying stopping y-values of the line segments. The character vector will place stopping y-values relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`default.units`	A string indicating the default units to use if `x0`, `y0`, `x1`, or `y1` are only given as numeric vectors. Default value is `default.units = "inches"`.
`linecolor`	A character value specifying segment line color. Default value is `linecolor = "black"`.
`lwd`	A numeric specifying segment line width. Default value is `lwd = 1`.
`lty`	A numeric specifying segment line type. Default value is `lty = 1`.

`lineend`	A character value specifying line end style. Default value is `lineend = "butt"`. Options are: • `"round"`: Segment ends are rounded. • `"butt"`: Segment ends end exactly where ended. • `"square"`: Segment ends are squared.
`linejoin`	A character value specifying line join style. Default value is `linejoin = "mitre"`. Options are: • `"round"`: Line joins are rounded. • `"mitre"`: Line joins are sharp corners. • `"bevel"`: Line joins are flattened corners.
`arrow`	A list describing arrow heads to place at either end of the line segments, as produced by the arrow function.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `segments` object containing relevant placement and grob information.

See Also

grid.segments, arrow

Examples

```
library(grid)
## Create a page
pageCreate(width = 7.5, height = 6, default.units = "inches")

## Plot one line segment
plotSegments(
    x0 = 3.75, y0 = 0.25, x1 = 3.75, y1 = 5.75,
    default.units = "inches",
    lwd = 3, lty = 2
)

## Plot multiple line segments at different locations in different colors
plotSegments(
    x0 = 0.5, y0 = c(1, 3, 5), x1 = 3.25, y1 = c(1, 3, 5),
    default.units = "inches",
    lwd = 2, linecolor = c("#7ecdbb", "#37a7db", "grey")
)

## Plot a line segment with an arrowhead
plotSegments(
    x0 = 4.5, y0 = 0.5, x1 = 7, y1 = 3,
    default.units = "inches",
    arrow = arrow(type = "closed"), fill = "black"
)
```

```
## Plot lines with round lineends
plotSegments(
    x0 = c(4, 7), y0 = 3.5, x1 = 5.5, y1 = 4.5,
    default.units = "inches",
    lwd = 5, lineend = "round"
)

## Hide page guides
pageGuideHide()
```

`plotSignal` *Plot any kind of signal track data for a single chromosome*

Description

Plot any kind of signal track data for a single chromosome

Usage

```
plotSignal(
    data,
    binSize = NA,
    binCap = TRUE,
    negData = FALSE,
    chrom,
    chromstart = NULL,
    chromend = NULL,
    assembly = "hg38",
    linecolor = "#37a7db",
    fill = NA,
    ymax = 1,
    range = NULL,
    scale = FALSE,
    label = NULL,
    bg = NA,
    baseline = TRUE,
    baseline.color = "grey",
    baseline.lwd = 1,
    orientation = "h",
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL,
```

```
    ...
)
```

Arguments

`data`	Data to be plotted as a character value specifying a bigwig file path, a dataframe in BED format, or a GRanges object with metadata column `score`. Either one `data` argument or a list of two can be provided, where the second `data` will be plotted below the x-axis if positive. The second `data` can also be negative data.
`binSize`	A numeric specifying the length of each data bin in basepairs. Default value is `binSize = NA`.
`binCap`	A logical value indicating whether the function will limit the number of data bins to 8,000. Default value is `binCap = TRUE`.
`negData`	A logical value indicating whether the provided data has negative scores. Default value is `negData = FALSE`.
`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`linecolor`	A character value or vector of length 2 specifying the line color(s) outlining the signal track(s). Default value is `linecolor = "#37a7db"`.
`fill`	A character value or vector of length 2 specifying the fill color(s) of the signal track(s). Default value is `fill = NA`.
`ymax`	A numeric specifying the fraction of the max y-value to set as the height of the plot. Default value is `ymax = 1`.
`range`	A numeric vector of length 2 specifying the y-range of data to plot (c(min, max)).
`scale`	A logical value indicating whether to include a data scale label in the top left corner of the plot. Default value is `scale = FALSE`.
`label`	An optional character value to conveniently add a text label to the plot. If `scale = TRUE`, the label will be draw in the top right of the plot. Otherwise, the label will be drawn in the top left of the plot. For more customizable labels, use plotText. Default value is `label = NULL`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`baseline`	Logical value indicating whether to include a baseline along the x-axis. Default value is `baseline = TRUE`.
`baseline.color`	Baseline color. Default value is `baseline.color = "grey"`.
`baseline.lwd`	Baseline line width. Default value is `baseline.lwd = 1`.
`orientation`	A string specifying signal track orientation. Default value is `orientation = "h"`. Options are: • `"v"`: Vertical signal track orientation. • `"h"`: Horizontal signal track orientation.

`x`	A numeric or unit object specifying signal plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying signal plot y-location. The character value will place the signal plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying signal plot width.
`height`	A numeric or unit object specifying signal plot height.
`just`	Justification of signal plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Details

#A signal track can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotSignal(data, chrom,
           chromstart = NULL, chromend = NULL,
           x, y, width, height, just = c("left", "top"),
           default.units = "inches")
```

This function can also be used to quickly plot an unannotated signal track by ignoring plot placement parameters:

```
plotSignal(data, chrom,
           chromstart = NULL, chromend = NULL)
```

Value

Returns a `signal` object containing relevant genomic region, placement, and grob information.

Examples

```
## Load signal data
library(plotgardenerData)
data("IMR90_ChIP_H3K27ac_signal")
data("GM12878_ChIP_H3K27ac_signal")

## Create a page
pageCreate(width = 7.5, height = 2.1, default.units = "inches")
```

```
## Define region
region <- pgParams(
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    range = c(0, 45)
)

## Plot and place signal plots
signal1 <- plotSignal(
    data = IMR90_ChIP_H3K27ac_signal, params = region,
    x = 0.5, y = 0.25, width = 6.5, height = 0.65,
    just = c("left", "top"), default.units = "inches"
)

signal2 <- plotSignal(
    data = GM12878_ChIP_H3K27ac_signal, params = region,
    linecolor = "#7ecdbb",
    x = 0.5, y = 1, width = 6.5, height = 0.65,
    just = c("left", "top"), default.units = "inches"
)

## Plot genome label
plotGenomeLabel(
    chrom = "chr21",
    chromstart = 28000000, chromend = 30300000,
    assembly = "hg19",
    x = 0.5, y = 1.68, length = 6.5,
    default.units = "inches"
)

## Add text labels
plotText(
    label = "IMR90", fonsize = 10, fontcolor = "#37a7db",
    x = 0.5, y = 0.25, just = c("left", "top"),
    default.units = "inches"
)
plotText(
    label = "GM12878", fonsize = 10, fontcolor = "#7ecdbb",
    x = 0.5, y = 1, just = c("left", "top"),
    default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`plotText` *Plot text within a plotgardener layout*

Description

Plot text within a plotgardener layout

Usage

```
plotText(
    label,
    fontcolor = "black",
    fontsize = 12,
    rot = 0,
    check.overlap = FALSE,
    x,
    y,
    just = "center",
    default.units = "inches",
    params = NULL,
    ...
)
```

Arguments

`label`	Character or expression of text to be plotted.
`fontcolor`	A character value specifying text fontcolor. Default value is `fontcolor = "black"`.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 12`.
`rot`	A numeric specifying the angle to rotate the text. Default value is `rot = 0`.
`check.overlap`	A logical value to indicate whether to check for and omit overlapping text. Default value is `check.overlap = FALSE`.
`x`	A numeric vector or unit object specifying text x-location.
`y`	A numeric vector, unit object, or a character vector of values containing a "b" combined with a numeric value specifying text y-locations. The character vector will place text y-locations relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`just`	Justification of text relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = "center"`.
`default.units`	A string indicating the default units to use if `x` or `y` are only given as numerics or numeric vectors. Default value is `default.units = "inches"`.
`params`	An optional pgParams object containing relevant function parameters.
`...`	Additional grid graphical parameters. See gpar.

Value

Returns a `text` object containing relevant placement and grob information.

See Also

grid.text

Examples

```
## Create a page
pageCreate(width = 4, height = 2, default.units = "inches")

## Plot text, adjusting fontsize and fontface
plotText(
    label = "plotgardener", fontsize = 14, fontface = "bold",
    x = 1, y = 1, just = "center", default.units = "inches"
)

## Plot text, adjusting color, rotation, and fontfamily
plotText(
    label = "coordinate-based", fontcolor = "#225EA8", rot = 90,
    fontfamily = "HersheyScript", x = 2, y = 1, just = "center",
    default.units = "inches"
)

## Plot a text label in multiple places at once
plotText(
    label = "R", x = c(0.5, 1, 1.5), y = 1.5, just = "center",
    default.units = "inches"
)

## Plot a vector of text labels
plotText(
    label = c("pg", "plot", "gardener"), x = 3, y = c(0.5, 1, 1.75),
    just = "center", default.units = "inches"
)

## Hide page guides
pageGuideHide()
```

`plotTranscripts` *Plot gene transcripts in a pileup style for a single chromosome*

Description

Plot gene transcripts in a pileup style for a single chromosome

Usage

```
plotTranscripts(
    chrom,
    chromstart = NULL,
    chromend = NULL,
```

```
    assembly = "hg38",
    fill = c("#669fd9", "#abcc8e"),
    colorbyStrand = TRUE,
    strandSplit = FALSE,
    boxHeight = unit(2, "mm"),
    spaceWidth = 0.02,
    spaceHeight = 0.3,
    limitLabel = TRUE,
    transcriptHighlights = NULL,
    fontsize = 8,
    labels = "transcript",
    stroke = 0.1,
    bg = NA,
    x = NULL,
    y = NULL,
    width = NULL,
    height = NULL,
    just = c("left", "top"),
    default.units = "inches",
    draw = TRUE,
    params = NULL
)
```

Arguments

`chrom`	Chromosome of region to be plotted, as a string.
`chromstart`	Integer start position on chromosome to be plotted.
`chromend`	Integer end position on chromosome to be plotted.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`fill`	Character value(s) as a single value or vector specifying fill colors of transcripts. Default value is `fill = c("#669fd9", "#abcc8e")`.
`colorbyStrand`	A logical value indicating whether to color plus and minus strands by the first two colors in a `fill` vector, where plus strand transcripts will be colored by the first `fill` color and minus strand transcripts will be colored by the second `fill` color. Default value is `colorbyStrand = TRUE`.
`strandSplit`	A logical value indicating whether plus and minus-stranded transcripts should be separated, with plus strand transcripts plotted above the x-axis and minus strand transcripts plotted below the x-axis. Default value is `strandSplit = FALSE`.
`boxHeight`	A numeric or unit object specifying height of transcripts. Default value is `boxHeight = unit(2, "mm")`.
`spaceWidth`	A numeric value specifying the width of minimum spacing between transcripts, as a fraction of the plot's genomic range. Default value is `spaceWidth = 0.02`.
`spaceHeight`	A numeric value specifying the height of spacing between transcripts on different rows, as a fraction of `boxHeight`. Default value is `spaceHeight = 0.3`.

`limitLabel`	A logical value indicating whether to draw a "+" when not all elements can be plotted in the plotting space. Default value is `limitLabel = TRUE`.
`transcriptHighlights`	A two-column dataframe with a column named "transcript" or "gene" containing transcript names or their associated gene names as strings to highlight and a column named "color" containing corresponding highlight colors.
`fontsize`	A numeric specifying text fontsize in points. Default value is `fontsize = 8`.
`labels`	A character value describing the format of transcript text labels. Default value is `labels = "trancript"`. Options are: • `NULL`: No labels. • `"transcript"`: Transcript name labels. • `"gene"`: Gene name labels. • `"both"`: Combined transcript and gene name labels with the format "gene name:transcript name".
`stroke`	A numeric value indicating the stroke width for transcript body outlines. Default value is `stroke = 0.1`.
`bg`	Character value indicating background color. Default value is `bg = NA`.
`x`	A numeric or unit object specifying transcript plot x-location.
`y`	A numeric, unit object, or character containing a "b" combined with a numeric value specifying transcript plot y-location. The character value will place the transcript plot y relative to the bottom of the most recently plotted plot according to the units of the plotgardener page.
`width`	A numeric or unit object specifying transcript plot width.
`height`	A numeric or unit object specifying transcript plot height.
`just`	Justification of transcript plot relative to its (x, y) location. If there are two values, the first value specifies horizontal justification and the second value specifies vertical justification. Possible string values are: `"left"`, `"right"`, `"centre"`, `"center"`, `"bottom"`, and `"top"`. Default value is `just = c("left", "top")`.
`default.units`	A string indicating the default units to use if `x`, `y`, `width`, or `height` are only given as numerics. Default value is `default.units = "inches"`.
`draw`	A logical value indicating whether graphics output should be produced. Default value is `draw = TRUE`.
`params`	An optional pgParams object containing relevant function parameters.

Details

A transcripts plot can be placed on a plotgardener coordinate page by providing plot placement parameters:

```
plotTranscripts(chrom, chromstart = NULL, chromend = NULL,
                x, y, width, height, just = c("left", "top"),
                default.units = "inches")
```

This function can also be used to quickly plot an unannotated transcripts plot by ignoring plot placement parameters:

```
plotTranscripts(chrom, chromstart = NULL, chromend = NULL)
```

Genomic annotation information is acquired through TxDb and OrgDb-class packages, as determined through the `assembly` parameter.

Value

Returns a `transcripts` object containing relevant genomic region, placement, and grob information.

See Also

assembly, genomes, defaultPackages

Examples

```
## Load hg19 genomic annotation packages
library("TxDb.Hsapiens.UCSC.hg19.knownGene")
library("org.Hs.eg.db")

## Create page
pageCreate(width = 7.5, height = 3.5, default.units = "inches")

## Plot and place transcripts
plotTranscripts(
    chrom = "chr8", chromstart = 1000000, chromend = 2000000,
    assembly = "hg19", labels = "gene",
    x = 0.5, y = 0.5, width = 6.5, height = 2.5,
    just = c("left", "top"), default.units = "inches"
)

## Plot genome label
plotGenomeLabel(
    chrom = "chr8", chromstart = 1000000, chromend = 2000000,
    assembly = "hg19",
    x = 0.5, y = 3.03, length = 6.5, default.units = "inches"
)

## Plot a legend
plotLegend(
    legend = c("+ strand", "- strand"),
    fill = c("#669fd9", "#abcc8e"), border = FALSE,
    x = 0.5, y = 1, width = 1, height = 0.5,
    just = c("left", "top")
)

## Hide page guides
pageGuideHide()
```

`readBigwig` *Read a bigWig file and return it as a data frame*

Description

Read a bigWig file and return it as a data frame

Usage

```
readBigwig(
    file,
    chrom = NULL,
    chromstart = 1,
    chromend = .Machine$integer.max,
    strand = "*",
    params = NULL
)
```

Arguments

`file`	A character value specifying the path to the bigwig file.
`chrom`	Chromosome of data as a string, if data for a specific chromosome is desired.
`chromstart`	Integer start position on chromosome.
`chromend`	Integer end position on chromosome.
`strand`	A character value specifying strand. Default value is `strand = "*"`. Options are: • `"+"`: Plus strand. • `"-"`: Minus strand. • `"*"`: Plus and minus strands.
`params`	An optional pgParams object containing relevant function parameters.

Details

This function does not work on Windows.

Value

Returns a 6-column dataframe of bigwig information.

See Also

import.bw

Examples

```
if (.Platform$OS.type != "windows"){
    bwFile <- system.file("extdata/test.bw", package="plotgardenerData")

    ## Read in entire file
    bwData <- readBigwig(file = bwFile)

    ## Read in specified region
    bwRegion <- readBigwig(file = bwFile,
                            chrom = "chr2",
                            chromstart = 1,
                            chromend = 1500)
}
```

`readCool` *Read a .(m)cool file and return Hi-C data as a dataframe*

Description

Read a .(m)cool file and return Hi-C data as a dataframe

Usage

```
readCool(
    file,
    chrom,
    chromstart = NULL,
    chromend = NULL,
    altchrom = NULL,
    altchromstart = NULL,
    altchromend = NULL,
    resolution = "auto",
    zrange = NULL,
    norm = "NONE",
    binChunkSize = 5e6,
    params = NULL,
    quiet = FALSE
)
```

Arguments

`file`	A character value specifying the path to the .(m)cool file.
`chrom`	Chromosome of data, as a string.
`chromstart`	Integer start position on chromosome.
`chromend`	Integer end position on chromosome.
`altchrom`	Alternate chromosome for interchromosomal data, as a string.

`altchromstart`	Alternate chromosome integer start position for interchromosomal data.
`altchromend`	Alternate chromosome integer end position for interchromosomal data.
`resolution`	A numeric specifying the width of each pixel. "auto" will attempt to choose a resolution in basepairs based on the size of the region.
`zrange`	A numeric vector of length 2 specifying the range of interaction scores, where extreme values will be set to the max or min.
`norm`	Character value specifying hic data normalization method. This value must be found in the .(m)cool file. Default value is `norm = "NONE"`.
`binChunkSize`	A numeric specifying the number of bin indices to read from a file for a given region at a given resolution. If the total amount of data is larger than the `binChunkSize`, data will be read in multiple chunks. Default value is `binChunkSize = 5e6`.
`params`	An optional pgParams object containing relevant function parameters.
`quiet`	A logical indicating whether or not to print messages.

Value

Returns a 3-column dataframe in sparse upper triangular format with the following columns: `chrom`, `altchrom`, `counts`.

Author(s)

Sarah Parker, Nicole Kramer

See Also

readHic

Examples

```
## .cool file
coolFile <- file.path(tempdir(), "Rao2014-IMR90-MboI-allreps-filtered.1000kb.cool”)
download.file(url = "https://usgs2.osn.mghpcc.org/cooler01/examples/hg19/Rao2014-IMR90-MboI-allreps-filtered.10
    destfile = coolFile)

## Read in region `chr2:10000000-22000000` at 1000Kb cool file resolution
coolData <- readCool(file = coolFile, chrom = "chr2", chromstart = 10000000,
                     chromend = 22000000,
                     resolution = 1000000)

## .mcool file
mcoolFile <- file.path(tempdir(), "LEUK_HEK_PJA27_inter_30.mcool”)
download.file(url = "https://zenodo.org/records/10906240/files/LEUK_HEK_PJA27_inter_30.mcool?download=1”,
    destfile = mcoolFile)

## Read in region `chr2:1000000-5000000` at 100Kb resolution
mcoolData_100Kb <- readCool(file = mcoolFile, chrom = "2”,
                            chromstart = 1000000, chromend = 5000000,
                            resolution = 100000)
```

```
## Read in data for chr2 at 2500Kb resolution
mcoolData_2500Kb <- readCool(file = mcoolFile, chrom = "2",
                             resolution = 2500000)
```

`readCoolBpResolutions` *Read basepair resolutions from an .(m)cool file*

Description

Read basepair resolutions from an .(m)cool file

Usage

```
readCoolBpResolutions(file)
```

Arguments

`file`	A character value specifying the path to the .(m)cool file

Value

Vector of basepair resolutions

Author(s)

Sarah Parker

`readCoolChroms` *Read chromosomes included in .(m)cool files*

Description

Read chromosomes included in .(m)cool files

Usage

```
readCoolChroms(file, resolution = NULL)
```

Arguments

`file`	A character value specifying the path to the .(m)cool file
`resolution`	optional, specify which resolution(s) to read chromosomes from. Default is all resolutions in ‘file‘.

Value

Data frame or list of data frames of chromosome names and lengths

Author(s)

Sarah Parker, Nicole Kramer

`readCoolNorms` *Read normalizations included in .(m)cool files*

Description

Read normalizations included in .(m)cool files

Usage

```
readCoolNorms(file, resolution = NULL)
```

Arguments

`file`	A character value specifying the path to the .(m)cool file
`resolution`	optional, specify which resolution(s) to read normalization types from. Default is all resolutions in ‘file‘.

Details

The "BALANCE" normalization refers to applying the pre-calculated matrix balancing weights in the ‘weight‘ dataset of ‘file‘, typically present in files created using cooler. VC is vanilla coverage, VC_SQRT is square root of vanilla coverage, and KR is Knight-Ruiz normalization.

Please note that if using a file from HiC-Pro, ICE normalizations will come from files stored in a separate folder, and thus will not contain any normalizations explicitly called "ICE" or "BALANCE" since values are included already normalized. This normalization can be specified as "NONE".

Value

A vector or list of vectors of available normalizations

Author(s)

Sarah Parker, Nicole Kramer

`readHic` *Read a .hic file and return Hi-C data as a dataframe*

Description

Read a .hic file and return Hi-C data as a dataframe

Usage

```
readHic(
    file,
    chrom,
    chromstart = NULL,
    chromend = NULL,
    altchrom = NULL,
    altchromstart = NULL,
    altchromend = NULL,
    assembly = "hg38",
    resolution = "auto",
    res_scale = "BP",
    zrange = NULL,
    norm = "KR",
    matrix = "observed",
    params = NULL,
    quiet = FALSE
)
```

Arguments

`file`	A character value specifying the path to the .hic file.
`chrom`	Chromosome of data, as a string.
`chromstart`	Integer start position on chromosome.
`chromend`	Integer end position on chromosome.
`altchrom`	Alternate chromosome for interchromosomal data, as a string.
`altchromstart`	Alternate chromosome integer start position for interchromosomal data.
`altchromend`	Alternate chromosome integer end position for interchromosomal data.
`assembly`	Default genome assembly as a string or a assembly object. Default value is `assembly = "hg38"`.
`resolution`	A numeric specifying the width of each pixel. "auto" will attempt to choose a resolution in basepairs based on the size of the region.
`res_scale`	A character value specifying the resolution scale. Default value is `res_scale = "BP"`. Options are: • `"BP"`: Base pairs. • `"FRAG"`: Fragments.

`zrange`	A numeric vector of length 2 specifying the range of interaction scores, where extreme values will be set to the max or min.
`norm`	Character value specifying hic data normalization method. This value must be found in the .hic file. Default value is `norm = "KR"`.
`matrix`	Character value indicating the type of matrix to output. Default value is `matrix = "observed"`. Options are: • `"observed"`: Observed counts. • `"oe"`: Observed/expected counts. • `"log2oe"`: Log2 transformed observed/expected counts.
`params`	An optional pgParams object containing relevant function parameters.
`quiet`	A logical indicating whether or not to print messages.

Value

Returns a 3-column dataframe in sparse upper triangular format with the following columns: `chrom`, `altchrom`, `counts`.

See Also

straw

Examples

```
hicFile <- system.file("extdata/test_chr22.hic", package="plotgardenerData")

## Read in data for all chr22 file at 2.5Mb bp resolution
hicData <- readHic(file = hicFile, chrom = "22",
                    assembly = "hg19",
                    resolution = 2500000)

## Read in region `chr22:20000000-47500000` at 100 Kb resolution
hicData10Kb <- readHic(file = hicFile, chrom = "22",
                        chromstart = 20000000, chromend = 47500000,
                        assembly = "hg19",
                        resolution = 100000)
```

Index