Package 'karyoploteR'

November 14, 2025

Type Package

Title Plot customizable linear genomes displaying arbitrary data

Version 1.37.0

Date 2025-06-20

Description karyoploteR creates karyotype plots of arbitrary genomes and offers a complete set of functions to plot arbitrary data on them. It mimicks many R base graphics functions coupling them with a coordinate change function automatically mapping the chromosome and data coordinates into the plot coordinates. In addition to the provided data plotting functions, it is easy to add new ones.

License Artistic-2.0

Depends R (>= 3.4), regioneR, GenomicRanges, methods

Imports regioneR, GenomicRanges, IRanges, Rsamtools, stats, graphics, memoise, rtracklayer, Seqinfo, GenomeInfoDb, S4Vectors, biovizBase, digest, bezier, GenomicFeatures, bamsignals, AnnotationDbi, grDevices, VariantAnnotation

Suggests BiocStyle, knitr, rmarkdown, markdown, testthat, magrittr, BSgenome.Hsapiens.UCSC.hg19, BSgenome.Hsapiens.UCSC.hg19.masked, TxDb.Hsapiens.UCSC.hg19.knownGene, TxDb.Mmusculus.UCSC.mm10.knownGene, org.Hs.eg.db, org.Mm.eg.db, pasillaBamSubset

URL https://github.com/bernatgel/karyoploteR

BugReports https://github.com/bernatgel/karyoploteR/issues

VignetteBuilder knitr

biocViews Visualization, CopyNumberVariation, Sequencing, Coverage, DNASeq, ChIPSeq, MethylSeq, DataImport, OneChannel

NeedsCompilation no

Encoding UTF-8

RoxygenNote 7.3.2

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

git_branch devel

git_last_commit fd32b1e

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-11-13

Author Bernat Gel [aut, cre] (ORCID: <https://orcid.org/0000-0001-8878-349X>)

Maintainer Bernat Gel <bgel@igtp.cat>

Contents

addGeneNames . . . 3
autotrack . . . 5
colByCategory . . . 6
colByChr . . . 7
colByRegion . . . 8
colByValue . . . 10
darker . . . 11
filterParams . . . 12
findIntersections . . . 13
getChromosomeNamesBoundingBox . . . 14
getColorSchemas . . . 15
getCytobandColors . . . 15
getCytobands . . . 16
getDataPanelBoundingBox . . . 17
getDefaultPlotParams . . . 18
getMainTitleBoundingBox . . . 19
getTextSize . . . 20
getVariantsColors . . . 21
horizonColors . . . 22
is.color . . . 23
kpAbline . . . 23
kpAddBaseNumbers . . . 25
kpAddChromosomeNames . . . 27
kpAddChromosomeSeparators . . . 28
kpAddColorRect . . . 29
kpAddCytobandLabels . . . 30
kpAddCytobands . . . 32
kpAddCytobandsAsLine . . . 33
kpAddLabels . . . 34
kpAddMainTitle . . . 36
kpArea . . . 37
kpArrows . . . 39
kpAxis . . . 41
kpBars . . . 44

kpDataBackground . . . 46
kpHeatmap . . . 47
kpLines . . . 49
kpPlotBAMCoverage . . . 51
kpPlotBAMDensity . . . 53
kpPlotBigWig . . . 55
kpPlotCoverage . . . 58
kpPlotDensity . . . 59
kpPlotGenes . . . 61
kpPlotHorizon . . . 66
kpPlotLinks . . . 68
kpPlotLoess . . . 71
kpPlotManhattan . . . 73
kpPlotMarkers . . . 76
kpPlotNames . . . 78
kpPlotRainfall . . . 80
kpPlotRegions . . . 82
kpPlotRibbon . . . 84
kpPlotTranscripts . . . 86
kpPoints . . . 90
kpPolygon . . . 93
kpRect . . . 95
kpSegments . . . 97
kpText . . . 99
lighter . . . 101
makeGenesDataFromTxDb . . . 102
mergeTranscripts . . . 103
plotDefaultPlotParams . . . 104
plotKaryotype . . . 105
plotPalettes . . . 108
prepareParameters2 . . . 109
prepareParameters4 . . . 111
processClipping . . . 112
transparent . . . 113

Index **114**

---

addGeneNames *addGeneNames*

Description

Adds the gene names (defaults to symbols) to a GenesData object to be used by kpPlotGenes

Usage

```
addGeneNames(genes.data, orgDb="auto", keys=NULL, keytype="ENTREZID", names="SYMBOL")
```

Arguments

genes.data	(GenesData object) A valid genes.dat object like the ones obtained by makeGenesDataFromTxDb
orgDb	The orgDb object to use to extract the gene symbols. If "auto" the function will try to determine automatically the correct organism. See available obects at https://bioconductor.org/packages/release/BiocViews.html#___OrgDb (defaults to "auto")
keys	(character vector) The keys to be used in the internal select statement to get the names. If NULL, the first column of mcols(GenesData$genes) will be used. (defaults to NULL)
keytype	(character) The keytype used in the internal select statement. (defaults to "ENTREZID", that is, gene_id)
names	The column to extract from orgDb to use as gene names. (deafults to "SYMBOL")

Details

This function takes a valid data object and uses an OrgDb object to find the gene names (symbols by default) and add them. Names are added as a column named names to the genes element of GenesData and they replace anything that was present there before. If no ObjDb object is given, the function will try to identify the correct organism using the data in GenesData$metadata and select the OrgDb object if available. If it cannot identify the organism or there's no valid OrgDb for that organism it will fail with an error. Internally, the function uses a call to AnnotationDbi::select on the OrgDb. It is possible to specify the keys and keytypes as well as the column we want to use as names (defaults to SYMBOL for gene symbols).

Value

The original GenesData object with one additional column named "names" in GenesData$genes$names.

See Also

kpPlotGenes, makeGenesDataFromTxDb

Examples

```
library(TxDb.Hsapiens.UCSC.hg19.knownGene)

zoom <- toGRanges("chr17:29e6-30e6")
kp <- plotKaryotype(genome="hg19", zoom=zoom)
genes.data <- makeGenesDataFromTxDb(TxDb.Hsapiens.UCSC.hg19.knownGene,
                karyoplot=kp, plot.transcripts=FALSE,
                plot.transcripts.structure=FALSE)
genes.data <- addGeneNames(genes.data)
kpPlotGenes(kp, data=genes.data, r1=0.5, plot.transcripts=FALSE,
            gene.name.position = "left")
```

autotrack *autotrack*

Description

Computes r0 and r1 given track definition

Usage

```
autotrack(current.track, total.tracks, margin=0.05, r0=0, r1=1)
```

Arguments

current.track	(numeric) The track or tracks the current plot will occupy, starting from 1. If more than one value is provided, the plot will expand from min(current.track) to max(current.track).
total.tracks	(numeric) The total number of tracks
margin	(numeric) The margin is specified as the part of a track, by default 0.05, 5 percent of the track height.
r0	(numeric) the original r0
r1	(numeric) the original r1

Details

Small utility function to help compute r0 and r1 given the total number of tracks and the track(s) the current plot will occupy. It also takes into account a margin between tracks and original r0 and r1, so we can say something like, "Out of 5 tracks between 0 and 0.5, this plot will be at track 2", and it will return r0=0.1 and r1=0.2

Value

A list of two numerics: r0 and r1

Examples

```
#first track out of 4
autotrack(1, 4)

#the same, but without margin
autotrack(1, 4, 0)

#first and second tracks out of 4
autotrack(c(1,2), 4)

#The first track out of 4, fitting the four track between 0 and 0.5
autotrack(1, 4, r0=0, r1=0.5)
```

colByCategory *colByCategory*

Description

Given a vector of categorical values, assign a color to each one bases on its category

Usage

```
colByCategory(categories, colors)
```

Arguments

categories	A vector of categories. Will be transformed into a factor if it's not one (if characters, integers, etc...)
colors	(color vector, possibly named) The colors to to use for the different categories. If "auto", a rainbow palette will be used. (defaults to "auto")

Details

The vector of values will be first transformed into a factor (if it's not already a factor) and then colors will be selected from the palette in the order defined by the factor levels. If more colors than the ones available in the palette are needed, they will be reused. If the color vector is named using the factor levels and all levels are included, the link bewteen levels and colors will be honored. If the color palette is set to "auto", it will create a palette using "rainbow".

Value

A vector of colors of the same length as value

See Also

kpAddColorRect, colByValue

Examples

```
colByCategory(c("A", "A", "C", "A", "C", "B"))
#The order of the colors is defined by the order of the factor, not the "natural order" of the elements
colByCategory(c("A", "A", "C", "A", "C", "B"), colors=c("red", "green", "blue"))
#integer categories plus reuse of colors
colByCategory(categories=c(3,3,1,2,1,4,2,3,2,1), colors=c("red", "green", "blue"))
```

colByChr *colByChr*

Description

Given a set of data elements, return a color for each one based on their chromosome

Usage

```
colByChr(data, colors="2grays", all.chrs=NULL, default.col="black")
```

Arguments

data	Either a vector of characters or a GRanges object
colors	The name of a color set ("2grays", "blackgreen", "rainbow"...) or a vector of colors. If the vector is named, names are expected to be the chromosome names. (defaults to "2grays")
all.chrs	A vector with all possible chromosomes. If NULL, the list will be extracted from data (using seqlevels if available). (defaults to NULL)
default.col	The default color to return when something is unavailable

Details

Returns a color for each data element based on its chromosome. The returned colors might com from one of the predefined color sets or passed in as a parameter.

If `colors` is the name of one of the available color sets, it the color set is used. If it's a named character vector with the chromosome as names, they will be assigned by name and any missing chromosome will be `default.col`. If it's a non-named chraracter vector, will be used in order and recycled if necessary.

Data might be either a GRanges object or a vector of chromosomes.

Value

A vector of colors

Note

Available color.sets: "2grays"=c("#888888", "#444444"), "2blues"=c("#6caeff", "#2b5d9b") "blackgreen"=c("black", "green"), "greengray"=c("#c6ffb7", "#888888"), "brewer.set1"=c("#E41A1C", "#377EB8", "#4DAF4A", "#984EA3", "#FF7F00", "#FFFF33", "#A65628", "#F781BF", "#999999") "brewer.set2"=c("#66C2A5", "#FC8D62", "#8DA0CB", "#E78AC3", "#A6D854", "#FFD92F", "#E5C494", "#B3B3B3") "brewer.set3"=c("#8DD3C7", "#FFFFB3", "#BEBADA", "#FB8072", "#80B1D3", "#FDB462", "#B3DE69", "#FCCDE5", "#D9D9D9", "#BC80BD", "#CCEBC5", "#FFED6F") "brewer.pastel1"=c("#FBB4A "#B3CDE3", "#CCEBC5", "#DECBE4", "#FED9A6", "#FFFFCC", "#E5D8BD", "#FDDAEC", "#F2F2F2"), "brewer.pastel2"=c("#B3E2CD", "#FDCDAC", "#CBD5E8", "#F4CAE4", "#E6F5C9", "#FFF2AE", "#F1E2CC", "#CCCCCC"), "rainbow"=rainbow(n=length(all.chrs))

See Also

kpPoints

Examples

```
chrs <- c("chr1", "chr2", "chr2", "chr1", "chr5")
points <- toGRanges(paste0("chr", c(1:22, "X", "Y")), rep(10e6, 24), rep(10e6, 24))

colByChr(chrs)
colByChr(points)

kp <- plotKaryotype(plot.type=4, labels.plotter=NULL, ideogram.plotter=NULL)
kpAddChromosomeNames(kp, srt=45)
kpAddChromosomeSeparators(kp)

total.tracks <- 6

kpPoints(kp, points, col=colByChr(points), y=0.5, cex=1, r0=autotrack(1,total.tracks)$r0, r1=autotrack(1,total.t
colors <- NULL
kpPoints(kp, points, y=0.5, col=colByChr(points, colors=colors), cex=1, r0=autotrack(2,total.tracks)$r0, r1=autot
colors <- c("red", "blue")
kpPoints(kp, points, y=0.5, col=colByChr(points, colors=colors), cex=1, r0=autotrack(3,total.tracks)$r0, r1=autot
colors <- c(chr1="red", chr7="blue")
kpPoints(kp, points, y=0.5, col=colByChr(points, colors=colors), cex=1, r0=autotrack(4,total.tracks)$r0, r1=autot
kpPoints(kp, points, y=0.5, col=colByChr(points, colors=colors, default.col="green"), cex=1, r0=autotrack(5,total
colors <- c("red", "yellow", 3, "orchid", "blue")
kpPoints(kp, points, y=0.5, col=colByChr(points, colors=colors), cex=1, r0=autotrack(6,total.tracks)$r0, r1=autot

#Color sets
pp <- getDefaultPlotParams(plot.type=4)
pp$leftmargin <- 0.2
kp <- plotKaryotype(plot.type=4, labels.plotter=NULL, ideogram.plotter=NULL, plot.params=pp)
kpAddChromosomeNames(kp, srt=45)
kpAddChromosomeSeparators(kp)

color.sets <- c( "2grays", "2blues", "blackgreen", "greengray", "brewer.set1",
            "brewer.set2", "brewer.set3", "brewer.pastel1", "brewer.pastel2", "rainbow" )
total.tracks <- length(color.sets)
for(i in seq_len(length(color.sets))) {
   kpPoints(kp, points, y=0.5, col=colByChr(points, colors=color.sets[i]), cex=1, r0=autotrack(i,total.tracks)$r0
   kpAddLabels(kp, labels=color.sets[i], cex=0.7, r0=autotrack(i,total.tracks)$r0, r1=autotrack(i,total.tracks)$
}
```

---

colByRegion *colByRegion*

Description

Given a set of data elements, return a color for each one based on whether they overlap a given set of regions. This might be useful, for example, to set a different color for data points overlapping a certain region of interest.

Usage

```
colByRegion(data, regions, colors=NULL, original.colors=NULL, default.col="black")
```

Arguments

`data`	Either a vector of characters or a GRanges object
`regions`	(GRanges or equivalent) A set of regions where the color will be modified. Internally it will be converted into a Genomic Ranges object by `toGRanges` (from regioneR package) and so it can be either a GRanges, a data.frame, a character or any other value type accepted by that function. If `colors` is NULL (the default) and regions has additional columns in addition to chr, start and end, if any has a name in c("color", "colors", "col", "cols") it will be used. Otherwise the first additional column will be used.
`colors`	(color) The colors to be used for each region. The content will be recycled if needed. If NULL, the colors are assumed to be available in the regions object. (defaults to NULL)
`original.colors`	(color vector) The original colors of the data points. They will be use instead of the default color for data points not overlapping the regions. If NULL, the default color will be used. (defaults to NULL)
`default.col`	The default color to return for data elements not overlapping the regions. Only used if original.colors is NULL (defaults to "black")

Details

Given a set of data elements, return a color for each one based on whether they overlap a given set of regions. The colors might be different for each region and can be specified either in the regions object itself or in a separate `colors` parameter. If specified in `colors`, the values will be recycled as needed. Data points not in the specified region can take either a default color or keep their "original.color" if given. This is useful when using colByRegion to highlight data points as in kpPlotManhattan.

Value

A vector of colors

See Also

kpPoints, colByChr, toGRanges

Examples

```
data <- toGRanges("chr1", c(1e6*1:245), c(1e6*1:245)+10)
data$y <- rnorm(n = length(data), mean = 0.5, sd = 0.15)

regions <- toGRanges(c("chr1:10e6-20e6", "chr1:100e6-150e6"))
regions$col <- c("red", "blue")

kp <- plotKaryotype(chromosomes="chr1")
kpPoints(kp, data=data, r0=0, r1=0.2)
kpPoints(kp, data=data, r0=0.2, r1=0.4, col=colByRegion(data, regions = regions) )
kpText(kp, data=data, r0=0.4, r1=0.6, col=colByRegion(data, regions = regions), label="A", cex=0.5 )
kpBars(kp, data=data, y0=0, y1=data$y, r0=0.6, r1=0.8, border=colByRegion(data, regions = regions))
#It might not work wor objects where R expects a single color such as lines. Segments should be used instead
kpLines(kp, data=data, r0=0.8, r1=1, col=colByRegion(data, regions = regions) )


kp <- plotKaryotype(chromosomes="chr1")
kpPoints(kp, data=data, r0=0, r1=0.25)
kpPoints(kp, data=data, r0=0.25, r1=0.5, col=colByRegion(data, regions = regions, colors="green") )
kpText(kp, data=data, r0=0.5, r1=0.75, col=colByRegion(data, regions = regions, color=c("gray", "gold")), label="A
kpBars(kp, data=data, y0=0, y1=data$y, r0=0.75, r1=1, border=colByRegion(data, regions = regions))
```

`colByValue` *colByValue*

Description

Given a set of values, return a color for each of them based on their numeric value.

Usage

```
colByValue(value, colors, min=NULL, max=NULL)
```

Arguments

`value`	A vector of numeric values
`colors`	(color) The colors to built the color ramp. Refer to `colorRamp` for more details.
`min`	(NULL or numeric) The min value used to normalize the values. If NULL, min(value) will be used. (defaults to NULL)
`max`	(NULL or numeric) The max value used to normalize the values. If NULL, max(value) will be used. (defaults to NULL)

Details

A color ramp (similar to a gradient) will be built using the colors in the ‘colors‘ parameter using `colorRamp`. Values will be normalized to [0,1] using ‘min‘ and ‘max‘ (if NULL, min(value) will be 0 and max(value) will be 1) and these values will be used to determine the color. It uses

Value

A vector of colors

Note

Alpha values (transparency) are also used in the color computation (see examples)

See Also

kpPoints, colByChr

Examples

```
colByValue(c(0,0.25,0.5,0.75,1), colors=c("red", "green"))
colByValue(c(0,0.25,0.5,0.75,1), colors=c("#00000000", "#00000011"))

data <- toGRanges("chr1", c(1e6*1:245), c(1e6*1:245)+10)
data$y <- rnorm(n = length(data), mean = 0.5, sd = 0.15)

kp <- plotKaryotype(chromosomes="chr1")
kpPoints(kp, data=data, r0=0, r1=0.3)
kpPoints(kp, data=data, r0=0.35, r1=0.65, col=colByValue(data$y, colors=c("black", "green")) )
kpPoints(kp, data=data, r0=0.7, r1=1, col=colByValue(data$y, colors=c("black", "green"), min=0.4, max=0.6))

kp <- plotKaryotype(chromosomes="chr1")
kpPoints(kp, data=data, r0=0, r1=0.3, col=colByValue(data$y, colors=c("#00000000", "#000000FF")))
kpPoints(kp, data=data, r0=0.35, r1=0.65, col=colByValue(data$y, colors=c("black", "orange", "green")) )
kpPoints(kp, data=data, r0=0.7, r1=1, col=colByValue(data$y, colors=c("red", "#00000022","#00000022", "green"),mi
```

---

darker *darker*

Description

Given a color, return a darker one

Usage

```
darker(col, amount=150)
```

Arguments

col	(color) The original color. Might be specified as a color name or a "#RRGGBB(AA)" hex color definition.
amount	(integer, [0-255]) The fixed amount to subtract to each RGB channel (Defaults to 150).

Details

Very simple utility function to create darker colors. Given a color, it transforms it to rgb space, adds a set amount to all chanels and transforms it back to a color.

Value

A darker color

See Also

lighter

Examples

```
darker("red")
darker("#333333")
darker(c("red", 3, "#FF00FF"))
```

---

filterParams *filterParams*

Description

Given a list, select just only the valid.elements from each member. Also works with vectors instead of lists

Usage

```
filterParams(p, valid.elements, orig.length)
```

Arguments

p	a list or a single vector
valid.elements	a boolean vector with the elements to keep
orig.length	the length of the elements on which to apply the filtering

Details

This function is used in filtering the graphical parameters when plotting only a part of the genome. For each element of the list, if it has the exact specified length, filters it using the 'valid.elements' parameter.

Value

p with some members filtered

Examples

```
a <- 1:10
b <- 3:5
c <- 2

filterParams(list(a,b,c), c(rep(TRUE,5), rep(FALSE,5)), 10)
filterParams(a, c(rep(TRUE,5), rep(FALSE,5)), 10)
```

`findIntersections` *findIntersections*

Description

Finds the intersections of a data line with a given threshold

Usage

```
findIntersections(data, thr)
```

Arguments

`data`	(GRanges with y mcol) A GRanges with the data points
`thr`	(numeric) The value at wich we want to calculate the intersections

Details

Given a GRanges with an mcol with name "y" representing the values. This function will return a GRanges with the points intersecting a specific value "thr".

Value

A GRanges representing the intersection points between the data line and the threshold. It will return an empty GRanges if the line does not intersect the threshold.

Note

Important: It will only return the intersection points where the line crosses the threshold but not if a data point lies exactly at the threshold.

Examples

```
d <- toGRanges(c("1:1-1", "1:5-5", "1:15-15"))
d$y <- c(-2, 3, 1)

findIntersections(d, 1.5)
findIntersections(d, 0)
findIntersections(d, 5)
```

getChromosomeNamesBoundingBox *getChromosomeNamesBoundingBox*

Description

Return the regions where the chromosome names should be placed

Usage

```
getChromosomeNamesBoundingBox(karyoplot)
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype

Details

Given a KaryoPlot object, return the regions where the chromosome labels should be placed. The positions will depend on the plot type used.

Value

Returns a list with four elements (x0, x1, y0 and y1), each of them a named vector of integers with one coordinatefor every chromosome in the plot.

Note

In general, this function is automatically called by karyoploteR and the user never needs to call it.

See Also

plotKaryotype, kpAddChromosomeNames

Examples

```
kp <- plotKaryotype()
bb <- getChromosomeNamesBoundingBox(kp)
```

getColorSchemas *getColorSchemas*

Description

Return a structure with the color schemas included in karyoploteR

Usage

```
getColorSchemas()
```

Value

A list with the color schemas included in karyoploteR for cytobands, variants, horizons...

Examples

```
getColorSchemas()
```

getCytobandColors *getCytobandColors*

Description

Returns a named character vector with the colors of associated with the cytoband names

Usage

```
getCytobandColors(color.table=NULL, color.schema=c("circos", "biovizbase", "only.centromeres"))
```

Arguments

color.table	(named character vector) if present, it's returned as-is. Useful to specify your own color.tables.
color.schema	(character) The name of the color schema to use: circos, biovizBase, only.centromeres (everything in gray, except for centromeres in red). (defaults to circos)

Details

The function returns a named character vector with the colors of associated with the cytoband names. Two color schemas are available: circos (which copies the colors used by Circos) and biovizbase (that gets the cytoband colors from the biovizBase Bioconductor package). If a color.table is given, it is returned untouched.

Value

a named character vector with the colors associated to each cytoband name

See Also

plotKaryotype, kpAddCytobands

Examples

```
getCytobandColors()
getCytobandColors(color.schema="biovizbase")
```

---

getCytobands *getCytobands*

Description

Get the cytobands of the specified genome.

Usage

```
getCytobands(genome="hg19", use.cache=TRUE)
```

Arguments

genome	(character or other) specifies a genome using the UCSC genome name. Defaults to "hg19". If it's not a character, genome is ignored and and empty GRanges is returned.
use.cache	(boolean) wether to use or not the cytoband information included in the packge. use.cache=FALSE will force a download from the UCSC.

Details

It returns GRanges object with the cytobands of the specified genome. The cytobands for some organisms and genome versions have been pre-downloaded from UCSC and included in the karyoploteR package. For any other genome, getCytobands will use rtracklayer to try to fetch the cytoBandIdeo table from UCSC. If for some reason it is not possible to retrieve the cytobands, it will return an empty GRanges object. Setting the parameter use.cache to FALSE, the data included in the package will be ignored and the cytobands will be downloaded from UCSC.

The genomes (and versions) with pre-downloaded cytobands are: hg18, hg19, hg38, mm9, mm10, mm39, rn5, rn6, rn7, susScr11, bosTau9, bosTau8, equCab3, equCab2, panTro6, panTro5, rheMac10, danRer10, danRer11, xenTro10, dm3, dm6, ce6, ce10, ce11, sacCer2, sacCer3

Value

It returns a GenomicRanges object with the cytobands of the specified genome. If no cytobands are available for any reason, an empty GRanges is returned.

Note

This function is memoised (cached) using the memoise package. To empty the cache, use forget(getCytobands)

See Also

plotKaryotype

Examples

```
#get the cytobands for hg19 (using the data included in the package)
cyto <- getCytobands("hg19")


#get the cytobands for Drosophila Melanogaster
cyto <- getCytobands("dm6")
```

---

getDataPanelBoundingBox *getDataPanelBoundingBox*

Description

Return the bounding box of a data panel in plot coordinates

Usage

```
getDataPanelBoundingBox(karyoplot, data.panel)
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
data.panel	a valid data panel name (i.e. 1, 2, "ideogram", "all")

Details

Given a KaryoPlot object and a data.panel name, return the region where the data panel is placed. The returned values are in plot coordinates, that is, the coord.change.function has been applied.

Value

Returns a list with four elements (x0, x1, y0 and y1), each of them an integer with the plot coordinates for the data.panel

Note

A user is not expected to need this function. It is mainly used by plotting functions, specially when clipping the plot in a zoomed region.

See Also

plotKaryotype, kpDataBackground

Examples

```
kp <- plotKaryotype(plot.type=2)
dp1 <- getDataPanelBoundingBox(kp, 1)
```

---

getDefaultPlotParams *getDefaultParameters*

Description

Returns the default parameters for the given plot.type

Usage

```
getDefaultPlotParams(plot.type)
```

Arguments

plot.type	(integer) the required plot type. can be any valid plot type (see plotKaryotype)

Details

Given a plot.type, this function returns a list suitable as a valid plot.params object. The user can then proceed to change the parameter values as needed and supply the modified list to the plotKaryotype function.#'

Value

A valid plot.params object with the default values for the plotting parameters and ready to be used in the plotKaryotype

See Also

plotKaryotype

Examples

```
pp <- getDefaultPlotParams(plot.type=2)
pp

#Change the ideogramheight param to create thicker ideograms
pp$ideogramheight <- 150

plotKaryotype(genome="hg19", plot.type=2, plot.params=pp)
```

getMainTitleBoundingBox *getMainTitleBoundingBox*

Description

Return the regions where the chromosome names should be placed

Usage

```
getMainTitleBoundingBox(karyoplot)
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype

Details

Given a KaryoPlot object, return the regions where the main plot should be placed. The position will depend on the plot type used.

Value

Returns a list with four elements (x0, x1, y0 and y1), each of them an integer with the coordinates for the main title

Note

In general, this function is automatically called by karyoploteR and the user never needs to call it.

See Also

plotKaryotype, kpAddMainTitle

Examples

```
kp <- plotKaryotype()
bb <- getMainTitleBoundingBox(kp)
```

getTextSize *getTextSize*

Description

Returns the size of character strings in bases and r's

Usage

```
getTextSize(karyoplot, labels, cex=1, data.panel="1")
```

Arguments

karyoplot	(KaryoPlot) A KaryoPlot object representing the current plot
labels	(character) The character strings to measure
cex	(numeric) The cex value used to plot the text (defaults to 1)
data.panel	(data panel identifier) The name of the data panel on which text was plotted. (defaults to "1")

Details

Small utility function to get the size of text labels in usable units for karyoploteR: bases for the width and r's for the height. The r units are the ones passed to r0 and r1 and take into account that a data panel has always total height of 1 r (from r0=0 to r1=1)

Value

Returns a list with two elements: width and height. Each of them is a numeric vector of the same length as "labels" with the width in bases of each label and the height in r units of each label.

Examples

```
pp <- getDefaultPlotParams(plot.type=2)
pp$data2height <- 50

kp <- plotKaryotype(chromosomes="chr1", plot.type=2, plot.params=pp)

label <- "Looooooong label"
kpText(kp, chr="chr1", x=70e6, y=0.5, labels=label)
text.size <- getTextSize(kp, labels=label)
kpRect(kp, chr="chr1", x0=70e6-text.size$width/2, x1=70e6+text.size$width/2,
           y0=0.5-text.size$height/2, y1=0.5+text.size$height/2)

label <- "SHORT"
text.size <- getTextSize(kp, labels=label, cex=3)
kpRect(kp, chr="chr1", x0=170e6-text.size$width/2, x1=170e6+text.size$width/2,
           y0=0.2-text.size$height/2, y1=0.2+text.size$height/2, col="gold")
kpText(kp, chr="chr1", x=170e6, y=0.2, labels=label, cex=3)
```

```
label <- c("two_labels", "in a small data.panel=2")
kpText(kp, chr="chr1", x=c(100e6, 170e6), y=c(0.4, 0.2), labels=label, cex=0.6, data.panel=2)
text.size <- getTextSize(kp, labels=label, cex=0.6, data.panel=2)
kpRect(kp, chr="chr1", x0=c(100e6, 170e6)-text.size$width/2, x1=c(100e6, 170e6)+text.size$width/2,
       y0=c(0.4, 0.2)-text.size$height/2, y1=c(0.4, 0.2)+text.size$height/2, data.panel=2)
```

`getVariantsColors` *getVariantsColors*

Description

Given the reference and alternative for a set of variants, assigns a color to each of them

Usage

```
getVariantsColors(ref, alt, color.table=NULL, color.schema=c("cell21breast"))
```

Arguments

`ref`	(character vector) The reference nucleotides of the variants. It has to have the same length as `alt`.
`alt`	(character vector) The alternative nucleotides of the variants. It has to have the same length as `ref`
`color.table`	(named character vector) if present, its used to assign colors to the nucleotide substitutions.
`color.schema`	(character) The name of the color schema to use: `cell21breast` (the color schema used in "Mutational Processes Molding the Genomes of 21 Breast Cancers" by S. Nik-Zainal, Cell, 2012). (defaults to `cell21breast`)

Details

The function creates an nucleotide substitution identifier with for each variant and uses it to query the color.table lookup table. If color.table is NULL, a color.table based in the selected color.schema is used. All unkwonwn nucleotide substitutions are assigned a gray color. Color table needs to have entries for C>A, C>G, C>T, T>A, T>C and T>G (and optionally "others"), since other changes can be reverse complemented to these.

Value

a named character vector with the colors associated to each variant

See Also

`plotKaryotype`, `kpPlotRainfall`

Examples

```
ref <- c("A", "A", "C", "T", "G", "A")
alt <- c("G", "C", "T", "A", "A", "-")
getVariantsColors(ref, alt)

col.table <- c("C>A"="#FF0000", "C>G"="#000000", "C>T"="#00FF00", "T>A"="#0000FF", "T>C"="#BB00BB", "T>G"="#00BBB
getVariantsColors(ref, alt, col.table)
```

`horizonColors` *horizonColors*

Description

Returns the color structure needed by kpPlotHorizon

Usage

```
horizonColors(col, num.parts)
```

Arguments

`col`	(array of colors) An array of colors
`num.parts`	(positive integer) The number of colors to generate for pos and neg

Details

This function transforms an array of colors into a list of colors **internally** needed by kpPlotHorizon: a list with two elements, "neg" and "pos", each an array of colors of length num.parts. If col is a character of length one, it is interpreted as the name of a color scheme.

horizonColors(col, num.parts)

Value

A list with 2 elements, pos and neg, each with num.parts colors

Examples

```
horizonColors("redblue6", 3)
horizonColors("redblue6", 6)
horizonColors("bluegold3", 2)
horizonColors(c("red", "blue"), 3)
horizonColors(c("red", "#FFFFFF00", "blue"), 3)
```

is.color *is.color*

Description

Test if something is a valid color

Usage

```
is.color(x)
```

Arguments

x	The element to test

Details

This function tests if something is a valid color. Returns TRUE or FALSE. The function is vectorised.

Value

TRUE is x is a valid color, FALSE otherwise

Examples

```
is.color("red")
is.color("#333333")
is.color(NA)
is.color(NULL)
is.color("not_a_color")
is.color(3)

is.color(c("not_a_color", "red", 3, "#FF0000"))
```

kpAbline *kpAbline*

Description

This is the KaryoploteR version of the abline function to add horizontal or vertical lines to the plot.

Usage

```
kpAbline(karyoplot, chr=NULL, h=NULL, v=NULL, ymin=NULL, ymax=NULL, data.panel=1, r0=NULL, r1=NULL, cl
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes where the lines will be plotted. If NULL, the lines will be plotted in all chromosomes. (defaults to NULL)
h	(a numeric vector) A numeric vector with the heights where the horizontal lines will be plotted. If h is NULL, no horizontal lines will be plotted. (defaults to NULL)
v	(a numeric vector) A numeric vector with the positions (in base pairs) where the vertical lines will be plotted. If v is NULL, no vertical lines will be plotted. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

As with all other base-inspired low-level plotting functions in karyoploteR, the function has been designed to accept mostly the same parameters as the base one (see the package vignette for more information). In this case, however, the interface has been reduced and it is only possible to plot vertical and horizontal lines and it's not possible to provide an intercept and slope. In addition, the function accepts graphical parameters that are valid for the base function segments.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpSegments, kpLines

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=1000, mask=NA))
mcols(data.points) <- data.frame(y=rnorm(1000, mean = 0.5, sd = 0.1))

kp <- plotKaryotype("hg19", plot.type=1, chromosomes=c("chr1", "chr2"))
kpDataBackground(kp, data.panel=1)

kpPoints(kp, data=data.points, pch=".", col="#2222FF", cex=3)

#Add horizontal lines at mean
kpAbline(kp, h=0.5, col="red")

#and at the 1 sd
kpAbline(kp, h=c(0.4, 0.6), col="orange", lwd=0.5)
#and 2 sd's
kpAbline(kp, h=c(0.3, 0.7), col="orange", lwd=0.5, lty=2)

#And add two vertical lines at specific chromosomal locations
kpAbline(kp, v=c(67000000, 190000000), chr="chr1")
```

---

kpAddBaseNumbers *kpAddBaseNumbers*

Description

Plots the base numbers along the chromosome ideograms

Usage

```
kpAddBaseNumbers(karyoplot, tick.dist=20000000, tick.len=5, units="auto", add.units=FALSE, digits=2,
```

Arguments

karyoplot	(karyoplot object) A valid karyoplot object created by a call to plotKaryotype
tick.dist	(numeric) The distance between the major numbered tick marks in bases (defaults to 20 milions, one major tick every 20Mb)
tick.len	(numeric) The length of the major tick marks in plot coordinates (defaults to 5)

units	(character) the units for the numbers to be represented. Must be one of "auto", "b", "kb" or "Mb". If auto, it will utomatically choose the most suitable unit. (Defaults to "auto")
add.units	(boolean) Add the units (Mb, Kb...) to the tick labels. (Defaults to FALSE)
digits	(integer) The maximum number of digits after the decimal point in labels. (defaults to 2)
minor.ticks	(boolean) Whether to add unlabeled minor ticks between the major ticks (defaults to TRUE)
minor.tick.dist	(numeric) The distance between the minor ticks in bases (defaults to 5 milions, a minor tick mark every 5Mb)
minor.tick.len	(numeric) The length of the minor tick marks in plot coordinates (defaults to 2)
cex	(numeric) The cex parameter for the major ticks label (defaults to 0.5)
tick.col	(color) If specified, the color to plot the major ticks. Otherwise the default color or, if given, the col parameter will be used. (Defaults to NULL)
minor.tick.col	(color) If specified, the color to plot the minor ticks. Otherwise the default color or, if given, the col parameter will be used. (Defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	Any other parameter to be passed to internal function calls. Specially useful for graphic parameters.

Details

This function can be used to add the base numbers scale to the chromosome ideograms. The base numbers and ticks will be drawn next to the ideograms and not on a separate independent x axis. It is possible to control the number and position of the tick marks and labels

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype

Examples

```
kp <- plotKaryotype()
kpAddBaseNumbers(kp)

kp <- plotKaryotype(chromosomes="chr17")
kpAddBaseNumbers(kp, tick.dist=10000000, minor.tick.dist=1000000)
```

kpAddChromosomeNames *kpAddChromosomeNames*

Description

Plots the chromosome names in the karyoplot

Usage

```
kpAddChromosomeNames(karyoplot, chr.names=NULL, xoffset=0, yoffset=0, ...)
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
chr.names	(character vector) the names to use for the chromosomes. If NULL, the chromosome names in the original genome will be used. (defaults to NULL)
xoffset	(numeric) a number of units to move the the chromosome names on the x axis with respect to their standard position (defaults to 0)
yoffset	(numeric) a number of units to move the the chromosome names on the y axis with respect to their standard position (defaults to 0)
...	any additional parameter to be passed to the text plotting. All R base graphics params are passed along.

Details

Given a KaryoPlot object, plot the names of the depicted chromosomes. This function is usually automatically called by plotKaryotype unless labels.plotter is NULL.

Value

invisibly returns the given karyoplot object

See Also

plotKaryotype, getChromosomeNamesBoundingBox

Examples

```
kp <- plotKaryotype(labels.plotter = NULL)
kpAddChromosomeNames(kp, col="red", srt=30)
```

kpAddChromosomeSeparators
kpAddChromosomeSeparators

Description

Plots between the chromosomes

Usage

```
kpAddChromosomeSeparators(karyoplot, col="gray", lty=3, data.panel="all", ...)
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
col	(color) The color of the separator lines (defaults to "gray")
lty	(integer) The line type of the separators (defaults to 3, dashed lines)
data.panel	(data panel specification) For vertical lines, the span of the separator lines. Ignored for horizontal lines. (defaults to "all")
...	any additional parameter to be passed to the text plotting. All R base graphics params are passed along.

Details

Depending on the plot type it will draw vertical lines (if all chromosomes are in a one line (3,4,5,7)) or horizontal lines (1,2,6)

By default the lines will occupy the whole chromsome extent (data.panel="all") but using the data.panel parameter it can be tuned.

Value

invisibly returns the given karyoplot object

See Also

plotKaryotype

Examples

```
kp <- plotKaryotype(plot.type=4)
kpAddChromosomeSeparators(kp)

kp <- plotKaryotype(plot.type=5, ideogram.plotter=NULL)
kpAddChromosomeSeparators(kp)

kp <- plotKaryotype(plot.type=2)
kpAddChromosomeSeparators(kp, col="red")
```

kpAddColorRect *kpAddColorRect*

Description

Add color rectangles next to the data panels. Ideal to identify the data in the plot

Usage

```
kpAddColorRect(karyoplot, col="gray", rect.width=0.02, rect.margin=0.01,  side="left", y0=0, y1=1, r0=
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
col	(color) the color of the rectangle
rect.width	(numeric) the width of the rectangle in plot coordinates (the whole plot has a width of 1). Usual value might be 0.05. Can be negative. (defaults to 0.02)
rect.margin	(numeric) the additional the margin between the rectangle and the chromosome. In plot coordinates (the whole plot has a width of 1). Usual value might be 0.05. Can be negative. (defaults to 0.01)
side	("left" or "right") The side of the plot where to plot the labels. (defaults to "left")
y0	(numeric) If the vertical space defined by r0 and r1 goes from 0 to 1, the value at which the bottom of the rectangle starts. Ex: y0=0.5, y1=1 will create a rectangle occupying the top half of the total allocated vertical space. (defaults to 0)
y1	(numeric) If the vertical space defined by r0 and r1 goes from 0 to 1, the value at which the top of the rectangle ends. Ex: y0=0, y1=0.5 will create a rectangle occupying the bottom half of the total allocated vertical space. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to position the rectangle. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to position the rectangle. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the labels are to be added. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)

border	(color) The color of the rectangle border. If NA, no border will be plotted. (defaults to NA)
...	any additional parameter to be passed to the text plotting. All R base graphics params are passed along.

Details

Given a KaryoPlot object, plot colored rectangles on the side of the data panels to help identify the different samples or types of data plotted

Value

invisibly returns the given karyoplot object

See Also

plotKaryotype, autotrack

Examples

```
 num.samples <- 10
samples.metadata <- data.frame(stage=c("I", "I", "I", "III", "IV", "III", "II", "IV", "IV", "IV"),
                               metastasis=c(0,0,0,0,1,0,0,0,1,0),
                            subtype=c("A", "A", "B", "B", "A", "B", "B", "B", "A", "B"))

kp <- plotKaryotype(plot.type=4)
for(i in 1:num.samples) {
  #metastasis
 kpAddColorRect(kp, col = colByCategory(samples.metadata$metastasis, color=c("white", "black"))[i],
                rect.margin = 0.01, rect.width = 0.01, r0=autotrack(i, num.samples))
  #stage
  kpAddColorRect(kp,
   col = colByCategory(samples.metadata$stage, colors=c("I"="plum1", "II"="hotpink", "III"="orange", "IV"="red2")
    rect.margin = 0.021, rect.width = 0.01, r0=autotrack(i, num.samples))
  #subtype
  kpAddColorRect(kp,
           col = colByCategory(samples.metadata$subtype, colors=c("dodgerblue", "gold"))[i],
                rect.margin = 0.032, rect.width = 0.01, r0=autotrack(i, num.samples))
}
```

---

kpAddCytobandLabels *kpAddCytobandLabels*

Description

Plots the base numbers along the chromosome ideograms

Usage

```
kpAddCytobandLabels(karyoplot, cex=0.5, force.all=FALSE, clipping=TRUE, ...)
```

Arguments

karyoplot	(karyoplot object) A valid karyoplot object created by a call to plotKaryotype
cex	(numeric) The cex parameter for the cytoband labels
force.all	(boolean) If true, all cytoband labels are plotted, even if they do not fit into the cytobands (Defaults to FALSE)
clipping	(boolean) Only used if zooming is active. If TRUE, the name will be not drawn out of the drawing are (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the labels may overflow into the margins of the plot. (defaults to TRUE)
...	Any other parameter to be passed to internal function calls. Specially useful for graphic parameters.

Details

This function can be used to add labels idenfifying the cytobands. It gets the labels from the cytobands information stored in the karyoplot object and it will only plot the labels that fit inside the available space. This means than in some cases (such as when plotting a complete genome with default parameters) it is possible that no labels at all are added.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype

Examples

```
kp <- plotKaryotype()
kpAddBaseNumbers(kp)
kpAddCytobandLabels(kp)

kp <- plotKaryotype(chromosomes="chr17")
kpAddBaseNumbers(kp, tick.dist=10000000, minor.tick.dist=1000000)
kpAddCytobandLabels(kp)
```

kpAddCytobands *kpAddCytobands*

Description

Plots the chromosome cytobands in a karyoplot

Usage

```
kpAddCytobands(karyoplot, color.table=NULL, color.schema=c("circos", "biovizbase", "only.centromeres'
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
color.table	(named character vector) a table specifying the colors to plot the cytobands. If NULL, it gets the colors calling getCytobandColors. (defaults to NULL)
color.schema	(character) The name of the color schema to use: circos, biovizBase, only.centromeres (everything in gray, except for centromeres in red). (defaults to circos)
clipping	(boolean) Only used if zooming is active. If TRUE, cytoband representation will be not drawn out of the drawing are (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the cytobands representation may overflow into the margins of the plot. (defaults to TRUE)
...	any additional parameter to be passed to the functions called from kpAddCytobands.

Details

Plots the cytobands representing the chromosome structure in a karyoplot. It extracts the cytobands from the karyoplot object it recieves as a parameter. It is possible to specify the colors used to plot the cytobands.

Value

invisibly returns the given karyoplot object

Note

In general, this function is automatically called by plotKaryotype and the user never nees to call it.

See Also

plotKaryotype, getCytobandColors, kpAddBaseNumbers, kpAddCytobandLabels

Examples

```
kp <- plotKaryotype(ideogram.plotter = NULL)
kpAddCytobands(kp)
```

---

kpAddCytobandsAsLine *kpAddCytobandsAsLine*

Description

Plots the chromosome cytobands in a karyoplot as a line

Usage

```
kpAddCytobandsAsLine(karyoplot, color.table=NULL, color.schema='only.centromeres', lwd=3, lend=1, cli
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
color.table	(named character vector) a table specifying the colors to plot the cytobands. If NULL, it gets the colors calling getCytobandColors. (defaults to NULL)
color.schema	(character: 'only.centromeres', 'circos', 'biovizbase') The name of the color schema to use. It is directly passed along to getCytobandColors. color.table takes precendence over color.schema. (defaults to 'only.centromeres')
lwd	(integer) The width of the line used to represent the ideogram (defaults to 3)
lend	(0, 1 or 2) The type of line end. (defaults to 1, "butt")
clipping	(boolean) Only used if zooming is active. If TRUE, cytoband representation will be not drawn out of the drawing are (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the cytobands representation may overflow into the margins of the plot. (defaults to TRUE)
...	any additional parameter to be passed to the functions called from kpAddCytobands.

Details

Plots the cytobands representing the chromosome structure in a karyoplot. It extracts the cytobands from the karyoplot object it recieves as a parameter. It is possible to specify the colors used to plot the cytobands. In contrast to kpAddCytobands it represents the chromosomes as a thin line

Value

invisibly returns the given karyoplot object

Note

In general, this function is automatically called by plotKaryotype and the user never needs to call it.

See Also

plotKaryotype, getCytobandColors, kpAddBaseNumbers, kpAddCytobandLabels

Examples

```
kp <- plotKaryotype(ideogram.plotter = NULL)
kpAddCytobandsAsLine(kp)

kp <- plotKaryotype(ideogram.plotter = NULL, plot.type=2)
kpAddCytobandsAsLine(kp)
```

---

kpAddLabels *kpAddLabels*

Description

Add labels to identify the data in the plot

Usage

```
kpAddLabels(karyoplot, labels, label.margin=0.01, side="left", pos=NULL, offset=0, r0=NULL, r1=NULL,
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
labels	(character) the text on the labels
label.margin	(numeric) the additional the margin between the labels the first base of the chromosome. In plot coordinates. Usual value might be 0.05. Can be negative. (defaults to 0.01)
side	("left" or "right") The side of the plot where to plot the labels. (defaults to "left")
pos	(numeric) The standard graphical parameter. See text. If NULL, pos will be selected automatically based on "side" (Defaults to NULL)
offset	(numeric) The standard graphical parameter. See text. (Defaults to 0)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to position the label. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to position the label. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the labels are to be added. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
...	any additional parameter to be passed to the text plotting. All R base graphics params are passed along.

Details

Given a KaryoPlot object, plot labels on the side of the data panels to help identify the different types of data plotted

Value

invisibly returns the given karyoplot object

See Also

plotKaryotype

Examples

```
plot.params <- getDefaultPlotParams(plot.type=2)
plot.params$leftmargin <- 0.2
plot.params$rightmargin <- 0.2

#In standard whole karyotypes, labels are drawn for all chromosomes

kp <- plotKaryotype("hg19", chromosomes=c("chr1", "chr2"), plot.type=2, plot.params = plot.params)
#data panel 1
kpDataBackground(kp, r0=0, r1=0.5, col="#FFDDDD")
kpDataBackground(kp, r0=0.5, r1=1, col="#DDFFDD")
kpAddLabels(kp, "Everything", label.margin = 0.12, srt=90, pos=3, cex=0.8)
kpAddLabels(kp, "Red", r0=0, r1=0.5, cex=0.6)
kpAddLabels(kp, "Green", r0=0.5, r1=1, cex=0.6)
#data panel 2
kpDataBackground(kp, col="#DDDDFF", data.panel = 2)
kpAddLabels(kp, "BLUE", data.panel=2)

#Plot on the right
#data panel 1
kpAddLabels(kp, "Everything", label.margin = 0.12, srt=90, pos=1, cex=0.8, side="right")
kpAddLabels(kp, "Red", r0=0, r1=0.5, cex=0.6, side="right")
kpAddLabels(kp, "Green", r0=0.5, r1=1, cex=0.6, side="right")
```

```
#In karyotypes with all chromosomes in a single line,
#labels are added on the first (side="left") or last (side="right") chromosome

kp <- plotKaryotype("hg19", chromosomes=c("chr1", "chr2", "chr3"), plot.type=3, plot.params = plot.params)
#data panel 1
kpDataBackground(kp, r0=0, r1=0.5, col="#FFDDDD")
kpDataBackground(kp, r0=0.5, r1=1, col="#DDFFDD")
kpAddLabels(kp, "Everything", label.margin = 0.12, srt=90, pos=3, cex=0.8)
kpAddLabels(kp, "Red", r0=0, r1=0.5, cex=0.6)
kpAddLabels(kp, "Green", r0=0.5, r1=1, cex=0.6)
#data panel 2
kpDataBackground(kp, col="#DDDDFF", data.panel = 2)
kpAddLabels(kp, "BLUE", data.panel=2)

#Plot on the right
#data panel 1
kpAddLabels(kp, "Everything", label.margin = 0.12, srt=90, pos=1, cex=0.8, side="right")
kpAddLabels(kp, "Red", r0=0, r1=0.5, cex=0.6, side="right")
kpAddLabels(kp, "Green", r0=0.5, r1=1, cex=0.6, side="right")



#In Zoomed regions, they are placed at the correct position too
kp <- plotKaryotype("hg19", zoom="chr1:20000000-40000000", plot.type=2, plot.params = plot.params)
kpAddBaseNumbers(kp, tick.dist=5000000, add.units=TRUE)
#data panel 1
kpDataBackground(kp, r0=0, r1=0.5, col="#FFDDDD")
kpDataBackground(kp, r0=0.5, r1=1, col="#DDFFDD")
kpAddLabels(kp, "Everything", label.margin = 0.12, srt=90, pos=3, cex=0.8)
kpAddLabels(kp, "Red", r0=0, r1=0.5, cex=0.6)
kpAddLabels(kp, "Green", r0=0.5, r1=1, cex=0.6)
#data panel 2
kpDataBackground(kp, col="#DDDDFF", data.panel = 2)
kpAddLabels(kp, "BLUE", data.panel=2)

#Plot on the right
#data panel 1
kpAddLabels(kp, "Everything", label.margin = 0.12, srt=90, pos=1, cex=0.8, side="right")
kpAddLabels(kp, "Red", r0=0, r1=0.5, cex=0.6, side="right")
kpAddLabels(kp, "Green", r0=0.5, r1=1, cex=0.6, side="right")
```

---

kpAddMainTitle *kpAddMainTitle*

Description

Plots the chromosome names in the karyoplot

Usage

```
kpAddMainTitle(karyoplot, main=NULL, ...)
```

Arguments

karyoplot	a karyoplot object returned by a call to plotKaryotype
main	(character) the main title of the plot
...	any additional parameter to be passed to the text plotting. All R base graphics params are passed along.

Details

Given a KaryoPlot object and a character string, plot the character strings as the main title of the plot. This function is usually automatically called by plotKaryotype unless.

Value

invisibly returns the given karyoplot object

See Also

plotKaryotype, getMainTitleBoundingBox

Examples

```
kp <- plotKaryotype(labels.plotter = NULL)
kpAddMainTitle(kp, col="red", srt=30)
```

---

kpArea *kpArea*

Description

Plots a line joining the data points along the genome and fills the area below the line.

Usage

```
kpArea(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, base.y=0, ymin=NULL, ymax=NULL, data.panel=1, r
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
base.y	(numeric) The y value at which the polygon will be closed. (defaults to 0)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(color) The fill color of the area. A single color. If NULL the color will be assigned automatically, either a lighter version of the color used for the outer line or gray if the line color is not defined. If NA no area will be drawn. (defaults to NULL)
border	(color) The color of the line enclosing the area. A single color. If NULL the color will be assigned automatically, either a darker version of the color used for the area or black if col=NA. If NA no border will be drawn. (Defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

... The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is a karyoploteR low-level plotting functions. Given a set of positions on the genome (chromosome and base) and a value (y) for each of them, it plots a line joining them and shades the area below them. Data can be provided via a `GRanges` object (`data`), independent parameters for chr, x and y or a combination of both. A number of parameters can be used to define exactly where and how the line and area are drawn. In addition, via the ellipsis operator (`...`), `kpArea` accepts any parameter valid for `lines` and `polygon` (e.g. `lwd`, `lty`, `col`, `density`...) The lines are drawn in a per chromosome basis, so it is not possible to draw lines encompassing more than one chromosome.

Value

Returns the original karyoplot object, unchanged.

See Also

`plotKaryotype`, `kpLines`, `kpText`, `kpPlotRibbon`

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, mask=NA))
mcols(data.points) <- data.frame(y=runif(500, min=0, max=1))

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpArea(kp, data=data.points)
  kpArea(kp, data=data.points, col="lightgray", border="red", lty=2, r0=0, r1=0.5)
  kpArea(kp, data=data.points, border="red", data.panel=2, r0=0, r1=0.5)
  kpArea(kp, data=data.points, border="blue", data.panel=2, r0=0, r1=0.5, base.y=1)

  kpArea(kp, data=data.points, border="gold", data.panel=2, r0=0.5, r1=1, base.y=0.5)
```

`kpArrows` *kpArrows*

Description

Plots segments at the specified genomic positions.

Usage

```
kpArrows(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=NULL, y0=NULL, y1=NULL, ymin=NULL, ymax=NULL, dat
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data), x0 to start(data) and x1 to end(data). If no parameter y0 is specified and data has a column named y0, this column will be used. The same for y1. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(a numeric vector) A numeric vector of x left positions (in base pairs). If data is not NULL, x0. (defaults to NULL)
x1	(a numeric vector) A numeric vector of x right positions (in base pairs). If data is not NULL, x1. (defaults to NULL)
y0	(a numeric vector) A numeric vector of y bottom positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
y1	(a numeric vector) A numeric vector of y top positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome, x0 and x1) and values (y0 and y1) for each of them, it plots arrows going from (x0, y0) to (x1, y1). Data can be provided via a `GRanges` object (`data`), independent parameters for chr, x0, x1, y0 and y1, or a combination of both. A number of parameters can be used to define exactly where and how the arrows are drawn. In addition, via the ellipsis operator (`...`), `kpSegments` accepts any parameter valid for `segments` (e.g. `code`, `lwd`, `lty`, `col`, ...)

Value

Returns the original karyoplot object, unchanged.

See Also

`plotKaryotype`, `kpRect`, `kpPoints`,

`kpPlotRegions`

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, length.mean=2000000, mask=NA))
y <- runif(500, min=0, max=0.8)
mcols(data.points) <- data.frame(y0=y, y1=y+0.2)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpArrows(kp, data=data.points, col="black", lwd=2, length=0.04)

 kpArrows(kp, data=data.points, y0=0, y1=1,  r0=0.2, r1=0.8, col="lightblue", data.panel=2)
```

`kpAxis` *kpAxis*

Description

Plot axis at the sides of the data panels

Usage

```
kpAxis(karyoplot, ymin=NULL, ymax=NULL, r0=NULL, r1=NULL, side=1, numticks=3, labels=NULL, tick.pos=NU
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
side	(numeric) In which side of the data panel should the axis be plotted. 1 - plot it on the right of the data panel. 2 - Plot it on the left. (defualts to 1)
numticks	(numeric) the number of ticks (and labels) of the axis. If tick.pos is present, it takes precedence over num.ticks and num.ticks is ignored. (defaults to 3)
labels	(character) the labels to be placed next to the ticks. If the number of labels is lower than the number of tickes, the labels will be reused. If NULL, the numeric values of the ticks will be used. (defaults to NULL)
tick.pos	(numeric) the places in the axis where a tick should be drawn. If present, num.ticks is ignored. If NULL, ticks are placed equidistant. (defaults to NULL)
tick.len	(numeric) the length of the ticks to be drawn measured in base pairs. If NULL, tick length is 0.01 times the length in bases of the longest chromosome. (defaults to NULL)
label.margin	(numeric) the additional the margin between the labels and ticks. Can be negative. If NULL, the default margin is used. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
text.col	(color) The color of the text elements (defaults to "black")
col	(color) The color of the axis lines (defaults to "black")
chromosomes	(character) To which chromosomes should we add the axis: "first", "last", "auto", "all" or a vector of chromosome names. With auto, the chromosomes will depend on the plot type and side of axis plotting. (defaults to "auto")
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

kpAxis plots axis at the sides of the data panels. It is possible to control the number of ticks and their labels, the placement of the plots and whether they span the whole data panel or just part of it. To do that they use the same placement parameters used by other karyoploteR functions (r0 and r1). This function does not have a chr option: axis are always plotted for all chromosomes.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpDataBackground, kpAbline

Examples

```
kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))

#Prepare data panel 1
kpDataBackground(kp, data.panel=1)
kpAxis(kp, data.panel = 1)
kpAxis(kp, data.panel = 1, ymin = 0, ymax=10, numticks = 11, side = 2, cex = 0.4, col="red")

#Prepare data panel 2
#Data panel 2 is conceptually split into two parts and the second part is "inverted"
kpDataBackground(kp, data.panel=2, r0 = 0, r1 = 0.45, color = "#EEEEFF")
kpAxis(kp, data.panel = 2, r0=0, r1=0.45, ymin = 0, ymax = 1, cex=0.5,
           tick.pos = c(0.3, 0.5, 0.7), labels = c("-1 sd", "mean", "+1 sd"))
kpAxis(kp, data.panel = 2, r0=0, r1=0.45, ymin = 0, ymax = 1, cex=0.5, side=2)

kpDataBackground(kp, data.panel=2, r0 = 0.55, r1 = 1, color = "#EEFFEE")
kpAxis(kp, data.panel = 2, r0=1, r1=0.55, ymin = 0, ymax = 1, side=1, cex=0.5)
kpAxis(kp, data.panel = 2, r0=1, r1=0.55, ymin = 0, ymax = 1, side=2, cex=0.5)


#Customizing the axis appearance
pp <- getDefaultPlotParams(plot.type=4)
pp$leftmargin <- 0.2
pp$rightmargin <- 0.2
kp <- plotKaryotype("hg19", plot.type=4, chromosomes=c("chr1", "chr2"), plot.params=pp)

#Prepare data panel 1
kpDataBackground(kp, data.panel=1)
kpAxis(kp, data.panel = 1, text.col="red", cex=1.6, srt=45)
kpAxis(kp, data.panel = 1, ymin = 0, ymax=10, numticks = 11, side = 2,
       cex = 0.7, col="red", text.col="gold", tick.len=30e6, )
```

kpBars *kpBars*

Description

Plot bars along the genome

Usage

```
kpBars(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=x0, y1=NULL, y0=NULL, ymin=NULL, ymax=NULL, data.pa
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data), x0 to start(data) and x1 to end(data). If no parameter y0 is specified and data has a column named y0, this column will be used. The same for y1. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(a numeric vector) A numeric vector of x left positions (in base pairs). If data is not NULL, x0. (defaults to NULL)
x1	(a numeric vector) A numeric vector of x right positions (in base pairs). If data is not NULL, x1. (defaults to NULL)
y1	(a numeric vector) A numeric vector of y top positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
y0	(a numeric vector) A numeric vector of y bottom positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

kpBars plots bars (rectangles) along the genome. It is very similar to kpRect except that if y0 is missing, it's automatically set to ymin so all bars start from the base of the plotting region.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpRect, kpLines

Examples

```
set.seed(1000)

data <- toGRanges(data.frame(chr="chr1", start=10000000*(0:23), end=10000000*(1:24)))
y1 <- ((sin(start(data)) + rnorm(n=24, mean=0, sd=0.1))/5)+0.5
y0 <- y1 - rnorm(n=24, mean = 0, sd = 0.15)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))

#We can specify all data values separately. If missing y0, it defaults to ymin
kpBars(kp, chr=as.character(seqnames(data)), x0=start(data), x1=end(data), y1=y1,
       col="#FFBBBB", border="#EEAAAA")
kpLines(kp, data=data, y=y1, col="red")

#or we can provide all data into a single GRanges object
mcols(data) <- data.frame(y0=y0, y1=y1)
kpBars(kp, data[data$y0>data$y1], col="orange", border="orange", data.panel=2)
kpBars(kp, data[data$y0<=data$y1], col="purple", border="purple", data.panel=2)

kpLines(kp, data, y=data$y1, data.panel=2, col="red")
kpLines(kp, data, y=data$y0, data.panel=2, col="blue")

kpAxis(kp, data.panel = 1, cex=0.8, numticks = 5, col="#777777")
```

```
kpAxis(kp, data.panel = 2, cex=0.8, numticks = 5, col="#777777")
```

---

kpDataBackground *kpDataBackground*

Description

Draws a solid rectangle delimiting the plotting area

Usage

```
kpDataBackground(karyoplot, r0=NULL, r1=NULL, data.panel=1, color="gray90", clipping=TRUE, ...)
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
color	(color) a valid color specification
clipping	(boolean) Only used if zooming is active. If TRUE, the data background will be not drawn out of the drawing area (i.e. in margins, etc) even if it overflows the visible drawing area. If FALSE, the data background representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This function is used to add a background color to delimit the plotting area. It can either delimit the whole plotting area or part of it so different data plotting regions can be seen.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpAxis

Examples

```
kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))

#Prepare data panel 1
kpDataBackground(kp, data.panel=1)
kpAxis(kp, data.panel = 1)
kpAxis(kp, data.panel = 1, ymin = 0, ymax=10, numticks = 11, side = 2, cex = 0.4, col="red")

#Prepare data panel 2
#Data panel 2 is conceptually split into two parts and the second part is "inverted"
kpDataBackground(kp, data.panel=2, r0 = 0, r1 = 0.45, color = "#EEEEFF")
kpAxis(kp, data.panel = 2, r0=0, r1=0.45, ymin = 0, ymax = 1, cex=0.5,
       tick.pos = c(0.3, 0.5, 0.7), labels = c("-1 sd", "mean", "+1 sd"))
kpAxis(kp, data.panel = 2, r0=0, r1=0.45, ymin = 0, ymax = 1, cex=0.5, side=2)

kpDataBackground(kp, data.panel=2, r0 = 0.55, r1 = 1, color = "#EEFFEE")
kpAxis(kp, data.panel = 2, r0=1, r1=0.55, ymin = 0, ymax = 1, side=1, cex=0.5)
kpAxis(kp, data.panel = 2, r0=1, r1=0.55, ymin = 0, ymax = 1, side=2, cex=0.5)
```

---

kpHeatmap *kpHeatmap*

Description

Plots the given data as a heatmap along the genome

Usage

```
kpHeatmap(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=x0, y=NULL, ymax=NULL, ymin=NULL, r0=NULL, r1=NU
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)

chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(numeric) the position (in base pairs) where the data region starts
x1	(numeric) the position (in base pairs) where the data region ends
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
colors	(colors) A set of color used to determine the color associated with each value. Internally, it uses colorRamp. (defaults to c("blue", "white", "yellow"))
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

Given regions of the genome with a start, end and a value, draws a heatmap-like representation, with the color of the region determined by its value. It is important to note that kpHeatmap will not extend the regions in any way, so if regions are not contiguous, they will appear as a series of rectangles and not as a continuous plot.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpRect, kpLines

Examples

```
dd <- toGRanges(data.frame(chr="chr1", start=4980000*(0:49), end=4980000*(1:50)))
y <- sin(x=c(1:length(dd))/2)

kp <- plotKaryotype("hg19", plot.type=1, chromosomes=c("chr1", "chr2"))

kpLines(kp, dd, y=y, r0=0.4, r1=0.6, ymin=-1, ymax=1)
kpAxis(kp, r0=0.4, r1=0.6, ymin=-1, ymax=1, cex=0.5)

kpHeatmap(kp, dd, y=y, colors = c("red", "black", "green"), r0=0, r1=0.2)
kpHeatmap(kp, dd, y=y, colors = c("green", "black", "red"), r0=0.2, r1=0.4)

#or we can provide all data into a single GRanges object
mcols(dd) <- data.frame(y=y)

kpHeatmap(kp, dd, r0=0.6, r1=0.8)
#non-contiguous regions appear as solitary rectangles
kpHeatmap(kp, sample(x = dd, 10), r0=0.8, r1=1, color=c("orange", "black", "purple", "green"))
```

---

kpLines *kpLines*

Description

Plots a line joining the data points along the genome.

Usage

```
kpLines(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, ymin=NULL, ymax=NULL, data.panel=1, r0=NULL, r
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)

x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome and base) and a value (y) for each of them, it plots a line joining them. Data can be provided via a GRanges object (data), independent parameters for chr, x and y or a combination of both. A number of parameters can be used to define exactly where and how the lines are drawn. In addition, via the ellipsis operator (...), kpLines accepts any parameter valid for lines (e.g. lwd, lty, col, ...) The lines are drawn in a per chromosome basis, so it is not possible to draw lines encompassing more than one chromosome.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpLines, kpText, kpPlotRegions

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, mask=NA))
mcols(data.points) <- data.frame(y=runif(500, min=0, max=1))

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpLines(kp, data=data.points, col="red")

  #Three ways of specifying the exact same data.points
  kpPoints(kp, data=data.points)
  kpPoints(kp, data=data.points, y=data.points$y, pch=16, col="#CCCCFF", cex=0.6)
  kpPoints(kp, chr=as.character(seqnames(data.points)),
           x=(start(data.points)+end(data.points))/2, y=data.points$y, pch=".",
           col="black", cex=1)

  #plotting in the data.panel=2 and using r0 and r1, ymin and ymax
  kpLines(kp, data=data.points, col="red", r0=0, r1=0.3, data.panel=2)
  kpPoints(kp, data=data.points, r0=0, r1=0.3, data.panel=2, pch=".", cex=3)

  kpLines(kp, data=data.points, col="blue", r0=0.4, r1=0.7, data.panel=2)
  kpLines(kp, data=data.points, col="blue", y=-1*(data.points$y),
          ymin=-1, ymax=0, r0=0.7, r1=1, data.panel=2)
  #It is also possible to "flip" the data by giving an r0 > r1
  kpPoints(kp, data=data.points, col="red", y=(data.points$y),
           r0=1, r1=0.7, data.panel=2, pch=".", cex=2)
```

---

kpPlotBAMCoverage *kpPlotBAMCoverage*

Description

Plots the coverage of a BAM file along the genome

Usage

```
kpPlotBAMCoverage(karyoplot, data=NULL, max.valid.region.size=1e6, ymin=NULL, ymax=NULL, data.panel=1
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a character) The path to a bam file (must be indexed).

`max.valid.region.size`
: (numeric) If the length of plotted region exceeds this number, nothing will be plotted. It's a safety mechanism to from excessive memory usage. (Defaults to 1e6, 1 milion bases)

`ymin`
: (numeric) The minimum value to be plotted on the data panel. If NULL, it is set to 0. (deafults to NULL)

`ymax`
: (numeric) The maximum value to be plotted on the data.panel. If NULL the maximum density is used. (defaults to NULL)

`data.panel`
: (numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to `plotKaryotype`. (defaults to 1)

`r0`
: (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

`r1`
: (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

`col`
: (color) The fill color to plot. If NULL the color will be assigned automatically, either a lighter version of the color used for the outer line or gray if the line color is not defined. If NA no area will be drawn. (defaults to NULL)

`border`
: (color) The color to use to plot the borders of the bars. If NULL, it will be a darker version of 'col'. If NA, no border will be plotted. (Defaults to NA)

`clipping`
: (boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

`...`
: The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions. In particular `col` and `border` can be used to set the colors used.

Details

`kpPlotBAMCoverage` plots the read coverage of a BAM file, that is, the number of reads overlapping each position. It uses the `bamsignals` package to efficiently access the BAM file. The BAM file must be indexed. This function is only recommended when plotting small parts of the genome. For larger plots consider using `kpPlotBAMDensity`.

There's more information at the `https://bernatgel.github.io/karyoploter_tutorial/`karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (max.coverage) stored at `karyoplot$latest.plot`

Note

Since the plotting the exact coverage for large regions of the genome may be unfeasable, it includes a safety mechanism causing it to raise a warning and do nothing if the region is larger than a threshold specified by `max.valid.region.size`.

See Also

kpPlotBAMDensity, kpPlotCoverage

Examples

```
library(pasillaBamSubset) #A package with 2 example bam files
un1.bam.file <- untreated1_chr4() # get the name of the first bam
un3.bam.file <- untreated3_chr4() #and the name of the second

kp <- plotKaryotype(genome="dm6", chromosomes="chr4") #The pasilla data comes from drosophila
kp <- kpAddBaseNumbers(kp, tick.dist = 1e5)
kp <- kpPlotBAMCoverage(kp, data = un1.bam.file) #Warning and does not plot. region too large.
kp <- kpPlotBAMCoverage(kp, data = un1.bam.file, max.valid.region.size=2000000)

#Use zoom to plot a smaller region to see the coverage with more detail
kp <- plotKaryotype(genome="dm6", zoom=toGRanges("chr4", 340000, 350000))
kp <- kpAddBaseNumbers(kp, tick.dist = 1e3)
kp <- kpPlotBAMCoverage(kp, data = un1.bam.file)


#Change the colors and borders and compare  two bams
kp <- plotKaryotype(genome="dm6", zoom=toGRanges("chr4", 340000, 350000))
kp <- kpAddBaseNumbers(kp, tick.dist = 1e3)
kp <- kpPlotBAMCoverage(kp, data = un1.bam.file, r0=0.5, r1=1, border="orange")
kp <- kpPlotBAMCoverage(kp, data = un3.bam.file, r0=0.5, r1=0, border="darkgreen") #r1 < r0 will flip the plot
kpAbline(kp, h=0.5, col="darkgray")
```

---

kpPlotBAMDensity *kpPlotBAMDensity*

Description

Plots the density of features along the genome

Usage

```
kpPlotBAMDensity(karyoplot, data=NULL, window.size=1e6, normalize=FALSE, ymin=NULL, ymax=NULL, data.p
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a BamFile or character) The path to a bam file (must be indexed) or a BamFile object.
window.size	(numeric) The size of the windows for wich the density is computed. (Defaults to 1e6, one megabase windows)
normalize	(boolean) Specifies if the density values should be normalized by the total number of mapped reads in the bam file. (Defaults to FALSE)
ymin	(numeric) The minimum value to be plotted on the data panel. If NULL, it is set to 0. (deafults to NULL)
ymax	(numeric) The maximum value to be plotted on the data.panel. If NULL the maximum density is used. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(color) The background color to plot. If NULL, it will be a lighter version of 'border' or 'black' if border is null. (Defaults to "gray80")
border	(color) The color to use to plot the borders of the bars. If NULL, it will be a darker version of 'col'. If NA, no border will be plotted. (Defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions. In particular col and border can be used to set the colors used.

Details

kpPlotBAMDensity plots the read density of a BAM file. It does not plot the coverage but the read density as the number of reads overlapping a every window. It uses Rsamtools to efficiently access the BAM file. The BAM file must be indexed.

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (windows and density) stored at `karyoplot$latest.plot`

See Also

plotKaryotype, kpPlotRibbon, kpPlotCoverage

Examples

```
library(pasillaBamSubset) #A package with 2 example bam files
un1.bam.file <- untreated1_chr4() # get the name of the first bam
un3.bam.file <- untreated3_chr4() #and the name of the second

window.size <- 1e4 #compute the density with 10kb windows

kp <- plotKaryotype(genome="dm6", chromosomes="chr4") #The pasilla data comes from drosophila
kp <- kpAddBaseNumbers(kp, tick.dist = 1e5)
kp <- kpPlotBAMDensity(kp, data = un1.bam.file, window.size = window.size, r0=0.5, r1=1, ymax=50000, col="darkoran
kp <- kpPlotBAMDensity(kp, data = un3.bam.file, window.size = window.size, r0=0.5, r1=0, ymax=50000, col="darkorch
kpAxis(kp, ymin=0, ymax=50000, r0=0.5, r1=1, labels = c("0", "25K", "50K"))
kpAxis(kp, ymin=0, ymax=50000, r0=0.5, r1=0, labels = c("0", "25K", "50K"))

kpText(kp, chr = "chr4", x=7e5, y=0.85, labels = paste0("Untreated 1 (reads per ", window.size, " bases)"))
kpText(kp, chr = "chr4", x=7e5, y=0.15, labels = paste0("Untreated 3 (reads per ", window.size, " bases)"))



#Or normalizing by the number of mapped reads
kp <- plotKaryotype(genome="dm6", chromosomes="chr4") #The pasilla data comes from drosophila
kp <- kpAddBaseNumbers(kp, tick.dist = 1e5)
kp <- kpPlotBAMDensity(kp, data = un1.bam.file, window.size = window.size, normalize=TRUE, r0=0.5, r1=1, ymax=0.2,
kp <- kpPlotBAMDensity(kp, data = un3.bam.file, window.size = window.size, normalize=TRUE, r0=0.5, r1=0, ymax=0.2,
```

---

kpPlotBigWig *kpPlotBigWig*

Description

Plots the wiggle values in a BigWig file. This function does not work on windows.

Usage

```
kpPlotBigWig(karyoplot, data, ymin=NULL, ymax="global", data.panel=1, r0=NULL, r1=NULL, col=NULL, bord
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a BigWigFile or character) The path to a bigwig file (either local or a URL to a remote file) or a BigWigFile object.
ymin	(numeric) The minimum value to be plotted on the data panel. If NULL, the minimum between 0 and the minimum value in the WHOLE GENOME will be used. (deafults to NULL)
ymax	(numeric or c("global", "per.chr", "visible.region")) The maximum value to be plotted on the data.panel. It can be either a numeric value or one of c("global", "per.chr", "per.region"). "Global" will set ymax to the maximum value in the whole data file. "per.chr" will set ymax to the maximum value in each chromosome. "visible.region" will set ymax to the maximum value in the visible region. (defaults to "global")
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(color) The fill color of the area. A single color. If NULL the color will be assigned automatically, either a lighter version of the color used for the outer line or gray if the line color is not defined. If NA no area will be drawn. (defaults to NULL)
border	(color) The color of the line enclosing the area. A single color. If NULL the color will be assigned automatically, either a darker version of the color used for the area or black if col=NA. If NA no border will be drawn. (Defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

kpPlotBigWig plots the data contained in a binary file format called BigWig. BigWig are used to efficiently store numeric values computed for windows covering the whole genome, ususally the coverage from an NGS experiment such as ChIP-seq. Only data required for the plotted region is loaded, and when more than one chromosome is visible, it will load the data for one crhomosome at a time. The function accepts either a BigWigFile oject or a character with the path to a valid big wig file. The character can also be a URL to a remote server. In this case data will be loaded transparently using the import function from rtracklayer. The data is plotted using kpArea and therefore it is possible to plot as a single line, a line with shaded area below or as a shaded area only adjusting the col and border parameters.

Value

Returns the original karyoplot object with the data computed (ymax and ymin values used) stored at karyoplot$latest.plot

Note

Since this functions uses rtracklayer BigWig infrastructure and it does not work on windows, this function won't work on windows either.

See Also

plotKaryotype, kpArea, kpPlotBAMDensity

Examples

```
if (.Platform$OS.type != "windows") {
  bigwig.file <- system.file("extdata", "BRCA.genes.hg19.bw", package = "karyoploteR")
 brca.genes.file <- system.file("extdata", "BRCA.genes.hg19.txt", package = "karyoploteR")
  brca.genes <- toGRanges(brca.genes.file)
  library(GenomeInfoDb)  # for seqlevelsStyle()
  seqlevelsStyle(brca.genes) <- "UCSC"

  kp <- plotKaryotype(zoom = brca.genes[1])
  kp <- kpPlotBigWig(kp, data=bigwig.file, r0=0, r1=0.2)
  kp <- kpPlotBigWig(kp, data=bigwig.file, r0=0.25, r1=0.45, border="red", lwd=2)
 kp <- kpPlotBigWig(kp, data=bigwig.file, r0=0.5, r1=0.7, ymin=0, ymax=1000, border="gold", col=NA)
  kpAxis(kp, r0=0.5, r1=0.7, ymin=0, ymax=1000)
 kp <- kpPlotBigWig(kp, data=bigwig.file, r0=0.75, r1=0.95, ymin=0, ymax="visible.region", border="orchid", col=N
  kpAxis(kp, r0=0.75, r1=0.95, ymin=0, ymax=kp$latest.plot$computed.values$ymax)
}
```

kpPlotCoverage *kpPlotCoverage*

Description

Given a GRanges object, plot the coverage along the genome.

Usage

```
kpPlotCoverage(karyoplot, data, show.0.cov=TRUE, data.panel=1, r0=NULL, r1=NULL, col="#0e87eb", border
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges or a coverage object) A GRanges object from wich the coverage will be computed or a SimpleRleList result of computing the coverage.
show.0.cov	(boolean) Wether to plot a thin line representing the regions with no coverage at all. (defaults to TRUE, plot the line)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(color) The background color of the regions. (defaults to "#0e87eb")
border	(color) The color of the border used to plot the coverage. If NULL, NA (no border) is used. (defaults to NULL)
ymax	(numeric) The maximum value to be plotted on the data.panel. If NULL the maximum coverage is used. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the high-level, or specialized, plotting functions of karyoploteR. It takes a GRanges object and plots it's coverage, that is, the number of regions overlapping each genomic position. The input can also be a SimpleRleList resulting from computing the coverage with coverage(data). In contrast with the low-level functions such as kpRect, it is not possible to specify the data using independent numeric vectors and the function only takes in the expected object types.

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (max.coverage, ymax) stored in latest.plot.

See Also

plotKaryotype, kpPlotRegions, kpBars, kpPlotBAMCoverage, kpPlotDensity

Examples

```
set.seed(1000)

#Example 2: Do the same with a single bigger set of possibly overlapping regions

kp <- plotKaryotype("hg19", plot.type=1, chromosomes=c("chr1", "chr2"))

regs <- createRandomRegions(nregions = 1000, length.mean = 10000000, length.sd = 1000000,
                            non.overlapping = FALSE, genome = "hg19", mask=NA)
kpPlotRegions(kp, regs, r0 = 0, r1 = 0.8, col="#AAAAAA")

kpPlotCoverage(kp, regs, ymax = 20, r0=0.8,  r1=1, col="#CCCCFF")
kpAxis(kp, ymin = 0, ymax= 20, numticks = 2, r0 = 0.8, r1=1)
```

---

kpPlotDensity *kpPlotDensity*

Description

Plots the density of features along the genome

Usage

```
kpPlotDensity(karyoplot, data=NULL, window.size=1e6, ymin=NULL, ymax=NULL, data.panel=1, r0=NULL, r1=N
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object from which the density will be computed.
window.size	(numeric) The size of the windows for wich the density is computed. (Defaults to 1e6, one megabase windows)
ymin	(numeric) The minimum value to be plotted on the data panel. If NULL, it is set to 0. (deafults to NULL)
ymax	(numeric) The maximum value to be plotted on the data.panel. If NULL the maximum density is used. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions. In particular col and border can be used to set the colors used.

Details

kpPlotDensity plots the density of a set of features represented by a GRanges object along the genome. It creates a non-overlapping tiling of the genome and computes the number of features per window. It's possible to specify the window size.

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (windows and density) stored at karyoplot$latest.plot

See Also

plotKaryotype, kpPlotRibbon, kpPlotCoverage

Examples

```
set.seed(1000)

data <- createRandomRegions(nregions=20000)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes="chr1")

kp <- kpPlotDensity(kp, data)
kpAxis(kp, ymin = 0, ymax=kp$latest.plot$computed.values$max.density)

kp <- kpPlotDensity(kp, data, data.panel=2, col="#CCCCFF",  ymax=20, lwd=2)
kpAxis(kp, ymin = 0, ymax=20, data.panel=2)

kp <- kpLines(kp, data=kp$latest.plot$computed.values$windows, y=kp$latest.plot$computed.values$density, col="bl
```

---

kpPlotGenes *kpPlotGenes*

Description

Plot genes and transcripts in the genome. Can get the genes and trancripts information from TxDb or from custom objects.

Usage

```
kpPlotGenes(karyoplot, data, gene.margin=0.3, gene.col=NULL, gene.border.col=NULL,
                add.gene.names=TRUE, gene.names=NULL, gene.name.position="top", gene.name.cex=1, gene
                plot.transcripts=TRUE, transcript.margin=0.5, transcript.col=NULL, transcript.border.
                add.transcript.names=FALSE, transcript.names=NULL, transcript.name.position="left", t
                        plot.transcripts.structure=TRUE,
                        non.coding.exons.height=0.5,
                add.strand.marks=TRUE, mark.height=0.20, mark.width=1, mark.distance=4,
                        coding.exons.col=NULL, coding.exons.border.col=NULL,
                    non.coding.exons.col=NULL, non.coding.exons.border.col=NULL,
                        introns.col=NULL, marks.col=NULL,
                        data.panel=1, r0=NULL, r1=NULL, col="black",
                        border=NULL, avoid.overlapping=TRUE, clipping=TRUE, ...)
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.

`data` (a `TxDb` object or a list with the required elements) A `TxDb` object with information on genes, transcripts and exons and their position on the genome.

`gene.margin` (numeric) If whole genes (as oposed to transcripts) are plotted (`plot.transcripts=FALSE`), the vertical margin between overlapping genes. The value is with respect to the gene height. A value of 0.5 will create a space above the genes with half the height of the genes themselves. (defaults to 0.3)

`gene.col` (color) If whole genes (as oposed to transcripts) are plotted (`plot.transcripts=FALSE`), the color used to fill the rectangles representing the genes. If NULL, the value of `col` will be used. (Defaults to NULL)

`gene.border.col` (color) If whole genes (as oposed to transcripts) are plotted (`plot.transcripts=FALSE`), the color used in the border of the rectangles representing the genes. If NULL, the value of `col` will be used. If NA, no border is drawn. (Defaults to NULL)

`add.gene.names` (boolean) Whether to add the names of the genes to the plot.

`gene.names` (named character vector) A named character vector with the labels of the genes. If not NULL, it will be used as a dictionary, so gene ids should be names and desired labels the values. If NULL, if genes.data$genes$name exists, it will use it, otherwise it will use the mcols(genes.data$genes)[,1] as labels. (defaults to NULL)

`gene.name.position` (character) The position of the gene name text relative to the rectangle. Can be "left", "right", "top", "bottom" or "center". (Defaults to "top")

`gene.name.cex` (numeric) The cex value to plot the gene names (defaults to 1)

`gene.name.col` (color) The color of the gene labels. If NULL, it will use col. (defaults to NULL)

`plot.transcripts` (boolean) Whether to plot the individual transcripts (TRUE) or a single rectangle to represent the whole gene (FALSE). (Defaults to TRUE)

`transcript.margin` (numeric) If transcripts are plotted (`plot.transcripts=TRUE`), the vertical margin between overlapping transcripts. The value is with respect to the transcript height. A value of 0.5 will create a space above the transcripts with half the height of the transcripts themselves. (defaults to 0.5)

`transcript.col` (color) If transcripts are plotted (`plot.transcripts=TRUE`), the color used to fill the rectangles representing the transcripts. If NULL, the value of `col` will be used. (Defaults to NULL)

`transcript.border.col` (color) If transcripts are plotted (`plot.transcripts=TRUE`), the color used in the border the rectangles representing the transcripts. If NULL, the value of `col` will be used. If NA, no border will be drawn. (Defaults to NULL)

`add.transcript.names` (boolean) Whether to add the names of the tramscripts to the plot. (defaults to FALSE)

`transcript.names` (named character) A named character vector with the labels of the transcripts. If not null, it will be used as a dictionary, so transcript ids should be names and

desired labels the values. If NULL, the transcript ids will be used as labels. (defaults to null)

`transcript.name.position`
(character) The position of the transcript name text relative to the rectangle. Can be "left", "right", "top", "bottom" or "center". (Defaults to "left")

`transcript.name.cex`
(numeric) The cex value to plot the transcript names (defaults to 0.6)

`transcript.name.col`
(color) The color of the transcript labels. If NULL, it will use col. (defaults to NULL)

`plot.transcripts.structure`
(boolean) Whether to draw the transcripts as single rectangles (FALSE) or to show the complete transcript structure (introns and exons) (TRUE). (Defaults to TRUE)

`non.coding.exons.height`
(numeric) The height of the non.coding exons relative to the transcript height. For example, if 0.5, non-coding exons will have a height half the size of the coding ones. (default 0.5)

`add.strand.marks`
(boolean) Whether strand marks should be plotted or not. Strand marks are small arrows along the introns (or whole transcripts if plot.transcript.structure=FALSE). (defaults to TRUE)

`mark.height` (numeric) The height of the strand marks in "coding exons heights", that is, if mark.height is 0.5, the mark will have a height of half the height of an exon. (defaults to 0.2)

`mark.width` (numeric) The width of the strand marks, in mark heights. mark.width=1 will produce arrow heads with a slope pf 45 degrees. A value higher than 1 will produce smaller angles and a value below 1 larger angles with more vertical lines. (defaults to 1, 45 degrees)

`mark.distance` (numeric) The distance between marks, in mark widths. A distance of 2, will add a space of 2*mark.width between consecutive marks. (defaults to 4)

`coding.exons.col`
(color) The fill color of the rectangles representing the coding exons. If NULL, it will use col. (defaults to NULL)

`coding.exons.border.col`
(color) The color of the border of the coding exons. If NULL, it will use border. (defaults to NULL)

`non.coding.exons.col`
(color) The fill color of the rectangles representing the non-coding exons. If NULL, it will use col. (defaults to NULL)

`non.coding.exons.border.col`
(color) The color of the border of the non-coding exons. If NULL, it will use border. (defaults to NULL)

`introns.col` (color) The color of the lines representing the introns. If NULL, it will use col. (defaults to NULL)

`marks.col` (color) The color of the arrows representing the strand. If NULL, it will use col. (defaults to NULL)

`data.panel` (numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to `plotKaryotype`. (defaults to 1)

`r0` (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

`r1` (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

`col` (color) The color of the genes, transcripts and labels. It is possible to specify different colors for each element class (transcript names, exons, strand marks...). All elements with no explicit color will be plotted using col. (Defaults to "black")

`border` (color) The color of the border of rectangles representing genes, transcripts and exons. Every element class may have its own specific color using the appropiate parameters. The ones with no explicit color will use border. At the same time, if border is NULL, it will default to col. (Defaults to NULL)

`avoid.overlapping` (boolean) If two or more regions (genes, transcripts) overlap in the genome (even partially), they can be drawn in two different layers (TRUE) or in the same layer, with superposing representations (FALSE). (Defaults to TRUE, draw non-overlapping elements)

`clipping` (boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

`...` The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of karyoploteR's higher level functions. It takes a transcript database (`TxDb`) object or a custom object with a specific structure and plots the genes along the genome. It's possible to plot genes as a whole using rectangle for each gene or to plot each transcript independently. If transcripts are drawn, it's possible to plot them as single boxes or to plot the detailed structure, differentiating coding exons, non-coding exons and introns. Transcripts may have, in addition, little arrows to mark the trascript strand (plus or minus). These strand marks are plotted in introns if the transcript structure is shown or on the whole transcript length if transcripts are plotted as boxes. Finally, it's possible to add labels to genes and transcripts. By default genes and transcripts identifiers in the input data structure will be used as labels, but it's possible to provide named character vectors to be used as dictionaries to change id's to better names.

The genes and transcripts representations are customizable. It's possible to change the colors of the different elements individually (i.e. to have red coding exons, blue non-coding exons and green introns); it's possible to change the relative height of the non-coding exons and to change the slate and density of the strand marks.

The data stating the positions of genes, transcripts and exons in the genome and their relations (which transcripts belong to which genes) can be given as a standard transcript database (`TxDb`) object or as a custom list with the following elements: `genes`, `transcripts`, `coding.exons` and `non.coding.exons`.

Value

Returns the original karyoplot object, unchanged.

Note

Plotting transcripts, specially plotting their structure might get quite slow in comparison to the usual speed of plotting in karyoploteR. It is not advised to plot genes and transcripts on the whole genome or in large regions of it. These functions have been designed to work with zoomed in karyoplots.

See Also

`plotKaryotype`, `kpRect`, `kpSegments`, `kpPlotTranscripts`

Examples

```
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene

zoom <- toGRanges("chr2", 47986268, 48147403)
gene.names <- c("2956"="MSH6", "80204"="FBXO11")

kp <- plotKaryotype(genome="hg19", zoom=zoom)
kpPlotGenes(kp, data=txdb, add.transcript.names = FALSE, gene.names=gene.names, r1=0.6)

kp <- plotKaryotype(genome="hg19", zoom=zoom)
kpPlotGenes(kp, data=txdb, plot.transcripts=FALSE, gene.names=gene.names, r1=0.6)

kp <- plotKaryotype(genome="hg19", zoom=zoom)
kpPlotGenes(kp, data=txdb, plot.transcripts.structure=FALSE, add.transcript.names=FALSE, gene.names=gene.names,

library(TxDb.Mmusculus.UCSC.mm10.knownGene)
kp <- plotKaryotype(genome="mm10", zoom="chr1:10.5e6-12.5e6")
genes.data <- makeGenesDataFromTxDb(txdb=TxDb.Mmusculus.UCSC.mm10.knownGene, karyoplot=kp)
genes.data <- addGeneNames(genes.data)
genes.data <- mergeTranscripts(genes.data)
kpPlotGenes(kp, genes.data, r1=0.25, mark.height = 0.5, gene.name.position = "left")
```

kpPlotHorizon *kpPlotHorizon*

Description

Plot a horizon plot, an area-like plot where different value levels are plotted in different colors.

Usage

```
kpPlotHorizon(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL,
                      num.parts=3, breaks=NULL, ymin=NULL, ymax=NULL,
                      data.panel=1, r0=0, r1=1, col="redblue6",
                      border=NA, clipping=TRUE, ...)
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
num.parts	(numeric) The number of parts into which the positive and the negative y spaces will be cut. Only used if breaks is NULL. (defaults to 3)
breaks	(numeric vector or list) A numeric vector of a list with two numeric vectors named "pos" and "neg". The exact break points where the y space will be cut. If NULL, the breaks will be automatically computed using num.parts. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)

r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(character, color vector or list) The palette name to be used, a color vector with the color to be used to define the color gradients or a list with two color vectors named "pos" and "neg". Available palettes are: redblue6, bluepurple10 and bluegold3 (defaults to "redblue6")
border	(color) The color of the line delimiting the filled areas. If NULL the color will be assigned automatically to a darker version of the color used for the area. If NA no border will be drawn. (Defaults to NA)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

kpPlotHorizon will create a horizon plot, a plot usually used in time series, that will represent a wiggle value (coverage, methylation, expression...) that we'd usually plot with a line or area in a fraction of the vertical space used by these function (usually in 1/6, but that's configurable). To do that, it will cut the y space into N parts and assign a different color to each one, flip the negative values into the positive space (with negative value colors) and plot all parts in the same vertical space.

A more detailed explanation of horizon plots can be found at https://flowingdata.com/2015/07/02/changing-price-of-food-items-and-horizon-graphs/ and a more detailed explanation of the horizon plots implemented in karyoploteR together with an explicative animation can be found at https://bernatgel.github.io/karyoploter_tutori

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpLines, kpText, kpPlotRibbon

Examples

```
data.points <- toGRanges(data.frame(chr=c("chr1", "chr1"), start=c(1, 100), end=c(1, 100), y=c(-2, 2)))
kp <- plotKaryotype(zoom=toGRanges("chr1:1-100"))
kpLines(kp, data.points, r0=0, r1=0.5, ymin=-2, ymax=2)
kpAbline(kp, h=c(-2,-1,0,1,2), col="#AAAAAA", r0=0, r1=0.5, ymin=-2, ymax=2)
kpAxis(kp, r0=0, r1=0.5, ymin=-2, ymax=2)
kpPlotHorizon(kp, data.points, r0=0.55, r1=1)


data.points <- toGRanges(data.frame(chr="chr1", start=1:100, end=1:100))
data.points$y <- sin(start(data.points)/2)
kp <- plotKaryotype(zoom=toGRanges("chr1:1-100"))
kpLines(kp, data.points, r0=0, r1=0.5, ymin=-1, ymax=1)
kpAbline(kp, h=c(-1,0,1), col="#AAAAAA", r0=0, r1=0.5, ymin=-1, ymax=1)
kpAxis(kp, r0=0, r1=0.5, ymin=-1, ymax=1)
kpPlotHorizon(kp, data.points, r0=0.55, r1=1)


set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, mask=NA))
mcols(data.points) <- data.frame(y=runif(500, min=-3, max=3))

kp <- plotKaryotype(chromosomes=c("chr1", "chr2"))
kpPlotHorizon(kp, data=data.points)

kp <- plotKaryotype(chromosomes=c("chr1", "chr2"))
kpPlotHorizon(kp, data=data.points, num.parts=1, r0=autotrack(1, 6)$r0, r1=autotrack(1, 6)$r1)
kpPlotHorizon(kp, data=data.points, num.parts=2, r0=autotrack(2, 6)$r0, r1=autotrack(2, 6)$r1)
kpPlotHorizon(kp, data=data.points, num.parts=3, r0=autotrack(3, 6)$r0, r1=autotrack(3, 6)$r1)
kpPlotHorizon(kp, data=data.points, num.parts=5, r0=autotrack(4, 6)$r0, r1=autotrack(4, 6)$r1)
kpPlotHorizon(kp, data=data.points, num.parts=9, r0=autotrack(5, 6)$r0, r1=autotrack(5, 6)$r1)
kpPlotHorizon(kp, data=data.points, num.parts=15, r0=autotrack(6, 6)$r0, r1=autotrack(6, 6)$r1)
kpLines(kp, data=data.points, ymin=-3, ymax=3)
kpAbline(kp, h=0, ymin=-3, ymax=3)

kp <- plotKaryotype(chromosomes=c("chr1", "chr2"))
kpPlotHorizon(kp, data=data.points, num.parts=4, r0=autotrack(1, 6)$r0, r1=autotrack(1, 6)$r1)
kpPlotHorizon(kp, data=data.points, col="redblue6", num.parts=4, r0=autotrack(2, 6)$r0, r1=autotrack(2, 6)$r1)
kpPlotHorizon(kp, data=data.points, col="bluepurple10", num.parts=4, r0=autotrack(3, 6)$r0, r1=autotrack(3, 6)$r1
kpPlotHorizon(kp, data=data.points, col="bluegold3", num.parts=4, r0=autotrack(4, 6)$r0, r1=autotrack(4, 6)$r1)
kpPlotHorizon(kp, data=data.points, col=c("red", "black", "green"), num.parts=4, r0=autotrack(5, 6)$r0, r1=autotr
kpPlotHorizon(kp, data=data.points, col=c("red", "yellow"), num.parts=4, r0=autotrack(6, 6)$r0, r1=autotrack(6, 6
```

---

kpPlotLinks *kpPlotLinks*

Description

Given 2 GRanges objects, plot lines or ribbons between region pairs

Usage

```
kpPlotLinks(karyoplot, data, data2=NULL, y=0, arch.height=NULL, data.panel=1, r0=NULL, r1=NULL, ymin=N
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with link start regions. If data2 is NULL, mcols(data) should be a bed-like structure with "link.chr", "link.start", "link.end" and optionally a "link.strand" columns. The first thee columns can have any name and the strand information will be extracted from the first column with "strand" in its name.
data2	(a GRanges) A GRanges object with the link end regions. If null, the end of the regions will be extracted from mcols(data). (Defaults to NULL)
y	(numeric) The y value where the origin and end of the links should be plotted (Defaults to 0)
arch.height	(numeric) The approximate arch height in links in the same chromosome in "y" scale. If NULL, it defaults to the whole span of the data panel.Also affects the curvature of links between chromosomes (Defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
ymin	(numeric) The minimum value to be plotted on the data panel. If NULL, it is set to 0. (deafults to NULL)
ymax	(numeric) The maximum value to be plotted on the data.panel. If NULL the maximum density is used. (defaults to NULL)
col	(color) The background color of the links. If NULL and border is specified, it defaults to a lighter version of border.
border	(color) The border color of the links. If NULL and col is specified, it defaults to a darker version of col.

clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the high-level, or specialized, plotting functions of karyoploteR. It takes two GRanges objects (or a single specially crafted one) and plots links (either lines or ribbons) between region pairs. Links are plotted bewteen the first region of both objects, between the second one, etc... and therefore both objects need to have the same length. Specifying a region as negative strand, will "flip" it, so the the start of a region can be linked to the end of its pair.

Value

Returns the original karyoplot object, unchanged.

Note

For a link to be plotted BOTH ends must be visible in the karyoplot. In particular, if a chromosome is not included in the plot (due to not being specified in chromosomes, for example) any link with an end on it will NOT be plotted. The same is true for zoomed in plots, where only intrachromosomal links will be visible. No warning or message will be generated.

See Also

plotKaryotype, kpPlotRibbon, kpSegments

Examples

```
set.seed(222)

starts <- sort(createRandomRegions(nregions = 15))
ends <- sort(createRandomRegions(nregions = 15))

kp <- plotKaryotype()
kpPlotLinks(kp, data=starts, data2=ends)

#Create larger regions, so they look like ribbons
starts <- sort(createRandomRegions(nregions = 15, length.mean = 8e6, length.sd = 5e6))
ends <- sort(createRandomRegions(nregions = 15, length.mean = 8e6, length.sd = 5e6))

kp <- plotKaryotype()
kpPlotLinks(kp, data=starts, data2=ends)

#flip some of them to represent inversions
strand(ends) <- sample(c("+", "-"), length(ends), replace = TRUE)
```

```
kp <- plotKaryotype()
kpPlotLinks(kp, data=starts, data2=ends)
```

---

kpPlotLoess *kpPlotLoess*

Description

Plot a LOESS smoothed line with confidence intervals given a list of points.

Usage

```
kpPlotLoess(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, conf.interval=0.95, span=0.5, data.panel=1
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
conf.interval	The confidence interval to plot. If a number in (0,1), the confidence interval is plotted. Else, no confidence interval is plotted. (defaults to 0.95)
span	A parameter to control the smoothing level. It is passed to underlying function loess. (deafults to 0.5)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
col	The color of the fitting line. (defaults to "black")
lty	The line type (dashed, dotted...) of the fitting line (defaults to 1, solid)
lwd	The line width of the fitting line (defaults to 1)
ci.col	The color of the area representing the confidence interval. If NA no CI is plotted. (defaults to #888888AA, transparent gray)
ci.border	The color of line marking the border of the confidence interval. If NA, it's not plotted. (defaults to NA)
ci.border.lty	The line type of line marking the border of the confidence interval. (defaults to 1, solid)
ci.border.lwd	The line width of line marking the border of the confidence interval. (defaults to 1)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

Given a set of data points (specified in any way accepted by kpPoints), plot a LOESS smoothed line with optional confidence intervals. LOESS is computed independently per each chromosome and data points are sorted before fitting. It is possible to adjust the confidence interval with conf.interval and setting it to NULL or NA will plot no CI. It is also possible to control the smoothing level with span. In addition to the standard plotting parameters, it is possible to control independently the color of the fitting line and CI area and CI borders. It is also possible to adjust the line type and line width of the fitting line and CI border.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpPoints, kpLines, kpPlotRibbon

Examples

```
set.seed(1000)

dd <- data.frame(chr="chr1", x=1:48*5e6, y=rnorm(n=48, 0.5, 0.1 ))

kp <- plotKaryotype(chromosomes="chr1")
kpPoints(kp, chr=dd$chr, x=dd$x, y=dd$y)
kpPlotLoess(kp, chr=dd$chr, x=dd$x, y=dd$y, col="red", conf.interval = 0.99, ci.col = "#FAAAAAAA")
```

---

kpPlotManhattan *kpPlotManhattan*

Description

Creates a manhattan plot, the ones usually seen in GWAS studies.

Usage

```
kpPlotManhattan(karyoplot, data=NULL, pval=NULL, points.col="2grays",
points.pch=16, points.cex=1, suggestiveline = -log10(1e-05),
suggestive.col="black", suggestive.lwd=1, suggestive.lty=2,
genomewideline = -log10(5e-08), genomewide.col="black", genomewide.lwd=1,
genomewide.lty=1, logp=TRUE, highlight=NULL, highlight.col="greenyellow",
ymin=NULL, ymax=NULL, data.panel=1, r0=NULL, r1=NULL, clipping=TRUE, ...)
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object (or any other format accepted by toGRanges) with the data points to plot. If the pval parameter is NULL and data has a column named "p" or "pval", it will be used as the pvalues to plot.
pval	(numeric) The pvalues to plot. It must have the same length as data. If NULL, data$p or data$pval will be used, if present. (defaults to NULL)
points.col	(colors or character) The colors used to plot the points. It can be either a vector of colors of the same length of data or a valid color specification for colByChr. (defaults to "2grays")
points.pch	(numeric between 1 and 25) The symbol used to plot every point. (Defaults to 16, a filled circle)
points.cex	(numeric) The size of the point symbols. (defaults to 1)
suggestiveline	(numeric) The suggestive significance threshold. The suggestive line will be plotted at this vertical position. If NULL, no line will be plotted. (defaults to -log10(1e-05))

suggestive.col	(color) The color of the suggestive line. (defaults to "black")
suggestive.lwd	(numeric) The width of the suggestive line (defaults to 1)
suggestive.lty	(numeric) The line type of the suggestive line (defaults to 2, dashed line)
genomewideline	(numeric) The genomewide significance threshold. The genomewide line will be plotted at this vertical position. If NULL, no line will be plotted (defaults to -log10(5e-08))
genomewide.col	(color) The color of the genomewide line. (defaults to "black")
genomewide.lwd	(numeric) The width of the genomewide line. (defaults to 1)
genomewide.lty	(numeric) The line type of the genomewide line. (defaults to 1, solid line)
logp	(logical) If TRUE, pval will be transformed using -log10(pval). (defaults to TRUE)
highlight	(GRanges, character vector, logical vector or numeric vector) The points to highlight in a different color. If a GRanges (or anythng accepted by toGRanges) the points overlapping these regions will be highlighted. Otherwise the points will be selected with data[highlight]. If NULL no point will be highlighted. (defaults to NULL)
highlight.col	The color of the highlighted points (defaults to "greenyellow")
ymin	(numeric) The minimum value to be plotted on the data panel. If NULL, it is set to 0. (deafults to NULL)
ymax	(numeric) The maximum value to be plotted on the data.panel. If NULL, it is set to 1. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

Creates a manhattan plot, the ones usually seen in GWAS studies. By default, it will compute the -log10 of the pvalues given and plot them as points. In addition, it can plot to horizontal lines, one for the "suggestive" threshold and another one for the "genomewide" significance threshold. In addition, it can highlight some of the data points in a different color. Highlighted data points can be specified per name, per position in the data structure or by their position on the genome (see examples).

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (ymin, ymax, suggestiveline, genomewideline and data (with two additional columns pval and color)) stored at `karyoplot$latest.plot`

See Also

plotKaryotype, kpPoints, colByChr, colByRegion

Examples

```
set.seed(1000)

#First simulate a GWAS result with a single significant peak
data <- createRandomRegions(nregions=20000, length.mean=1, length.sd=0, genome=filterChromosomes(getGenome("hg19
names(data) <- paste0("rs", 1:20000)
data$pval <- rnorm(n = 20000, mean = 0.5, sd = 0.5)
data$pval[data$pval<0] <- -1*data$pval[data$pval<0]
snps.in.peak <- which(overlapsAny(data, toGRanges("chr3:70e6-80e6")))
data$pval[snps.in.peak] <- runif(n = length(snps.in.peak), min=0.1, max=8)
data$pval <- 10^(-1*data$pval)

kp <- plotKaryotype("hg19", plot.type=4)
kp <- kpPlotManhattan(kp, data, ymax=8)
kpAxis(kp, ymax=8)

#Highlighting
kp <- plotKaryotype("hg19", plot.type=4)
kp <- kpPlotManhattan(kp, data, ymax=8, highlight="chr3:70e6-80e6", r0=autotrack(1,4))
kp <- kpPlotManhattan(kp, data, ymax=8, highlight=snps.in.peak, highlight.col="orchid", r0=autotrack(2,4))
kp <- kpPlotManhattan(kp, data, ymax=8, highlight=names(data)[snps.in.peak], highlight.col="orange", r0=autotrack
kp <- kpPlotManhattan(kp, data, ymax=8, highlight=overlapsAny(data, toGRanges("chr3:70e6-80e6")), r0=autotrack(4,

#Look and feel
kp <- plotKaryotype("hg19", plot.type=4)
kp <- kpPlotManhattan(kp, data, ymax=8, points.col="2blues", highlight="chr3:70e6-80e6", points.pch=2, points.cex
```

kpPlotMarkers *kpPlotMarkers*

Description

Plots markers on the genome as a line with a label on top.

Usage

```
kpPlotMarkers(karyoplot, data=NULL, chr=NULL, x=NULL, y=0.75, labels=NULL,
                    adjust.label.position=TRUE,  ignore.chromosome.ends=FALSE,
                    label.margin=0.001, max.iter=150, label.dist=0.001,
                  marker.parts = c(0.8,0.1, 0.1), text.orientation ="vertical",
                    ymin=NULL, ymax=NULL, data.panel=1, r0=NULL, r1=NULL,
                    line.color="black", label.color="black",
                    pos=NULL, srt=NULL, offset=NULL, clipping=TRUE, ...)
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to 0.75)
labels	(a character vector) The labels to be plotted. (defaults to NULL)
adjust.label.position	(logical) whether to adjust the label positions to avoid label overlapping (defaults to TRUE)
ignore.chromosome.ends	(logical) If TRUE, when adjusting label position marker labels can move beyond the chromosome ends. (defaults to FALSE, do not move out of origin chromosome)
label.margin	(numeric) The vertical margin to leave between the ens of the marker line and the marker label. In plot coordinates. (defaults to 0.001)
max.iter	(numeric) The maximum number of iterations in the iterative algorithm to adjust the label positioning. (defaults to 150)

label.dist (numeric) The minimum distance between labels to consider them as non-overlapping (defaults to 0.001)

marker.parts (numeruc vector of three elements) The portion of the distance between 0 and y to be filled with a: vertical, diagonal of vertical part of the marker line. (defaults to c(0.8,0.1,0.1), long vertical stem, small diagonal and small vertical on top)

text.orientation ("vertical" or "horizontal) How should the text be plotted. Forced values of srt and pos take precedence. (defaults to "vertical")

ymin (numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)

ymax (numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)

data.panel (numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)

r0 (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

r1 (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

line.color (color) The color of marker line. (defaults to "black")

label.color (color) The color of the label (defaults to "black")

pos (1,2,3,4) The standard pos graphical parameter. If NULL, it's automatically set depending on "text.orientation". (defaults to NULL)

srt (numeric) The standard srt graphical parameter. If NULL, it's automatically set depending on "text.orientation". (defaults to NULL)

offset (numeric) The standard offset graphical parameter. If NULL, it's automatically set depending on "text.orientation". (defaults to NULL)

clipping (boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

... The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This function plots markers on the genome. It implements an interative algorithm to avoid ovelapping between the labels of different markers. Since labels might be plotted in a different position

than the original points, a line with three parts (a vertical, a diagonal and another vertical) is plotted to link the label with the original position. It is possible to plot labels in horizontal or vertical text and to specify different colors for the marker line and label.

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (adjusted label positioning) stored at `karyoplot$latest.plot`

Note

The iterative algorithm is not guaranteed to suceed and might end up with overlapping labels if labels are too dense or if too few iterations allowed. With many markers, the algorithm might be slow.

See Also

plotKaryotype, kpLines, kpText

Examples

```
data <- toGRanges(data.frame(c("chr1", "chr1", "chr1"), c(20e6, 21e6, 22e6), c(20.01e6, 21.01e6, 22.01e6), labels=

kp <- plotKaryotype("hg19", plot.type=1, chromosomes = "chr1", main="Default markers")
kpPlotMarkers(kp, data)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes = "chr1", main="Markers Horizontal")
kpPlotMarkers(kp, data, text.orientation = "horizontal")
kpPlotMarkers(kp, data, text.orientation = "horizontal", label.dist = 0.02, data.panel=2)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes = "chr1", main="Different Marker parts")
kpPlotMarkers(kp, data, text.orientation = "horizontal", marker.parts=c(0, 1, 0), line.color="red")
kpPlotMarkers(kp, data, text.orientation = "horizontal", marker.parts=c(0.1, 0.2, 0.4), label.dist = 0.02, data.pa
```

---

kpPlotNames *kpPlotNames*

Description

Plots text labels with positioning relative to rectangles along the genome.

Usage

```
kpPlotNames(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=x0, y0=NULL, y1=NULL, labels=NULL, position="l
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data), x0 to start(data) and x1 to end(data). If no parameter y0 is specified and data has a column named y0, this column will be used. The same for y1. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(a numeric vector) A numeric vector of x left positions (in base pairs). If data is not NULL, x0. (defaults to NULL)
x1	(a numeric vector) A numeric vector of x right positions (in base pairs). If data is not NULL, x1. (defaults to NULL)
y0	(a numeric vector) A numeric vector of y bottom positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
y1	(a numeric vector) A numeric vector of y top positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
labels	(character) The labels to use in the plot. They will be associated to the rectangles by its order and recycled as needed.
position	(character) The position of the text relative to the rectangle. Can be "left", "right", "top", "bottom" or "center". Defaults to "left".
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

... The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is a simple wrapper around kpText that positions the text relative to the rectangles defined by its arguments. They may be used to name or label different graphical elements in the plot. The rectangles may be specified as in kpRect the relative positions accepted are: "left", "right", "top", "bottom", "center". It is possible to specify and empty label (labels="") to leave an element without name.

Value

Returns the original karyoplot object, unchanged.

See Also

kpText, kpRect

Examples

```
regs <- toGRanges(data.frame(chr=c("chr1", "chr1", "chr1"),
                 start=c(20e6, 100e6, 200e6),
                 end=c(40e6, 170e6, 210e6),
                 y0=c(0.1, 0.5, 0.7),
                 y1=c(0.5, 0.6, 0.95)))

kp <- plotKaryotype(genome="hg19", chromosomes="chr1")
kpRect(kp, data=regs)

kpPlotNames(kp, data=regs, labels=c("R1", "R2", "R3"))
kpPlotNames(kp, data=regs, labels=c("R1", "R2", "R3"), position="top", cex=2)
kpPlotNames(kp, data=regs, labels=c("R1", "", "R3"), position="right", col="red")
kpPlotNames(kp, data=regs, labels="bottom", position="bottom", col=rainbow(3))
kpPlotNames(kp, data=regs, labels="o", position="center", col=rainbow(3), cex=1)
```

---

kpPlotRainfall *kpPlotRainfall*

Description

Creates a rainfall plot showing the distances between features in the genome. Usually used to plot the distance bewteen somatic mutations to idenify kataegis.

Usage

```
kpPlotRainfall(karyoplot, data, ref=NULL, alt=NULL, col="cell21breast", ymin=NULL, ymax=7, data.panel=
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges, a VRanges or a path to a VCF file) A GRanges or VRanges object with the variants to plot or the path to a VCF file.
ref	(character vector or NULL) A character vector of with the reference base of the variants in data. Used to determine the color. If NULL and data is a VRanges or a VCF file, the information in data will be used (defaults to NULL)
alt	(character vector or NULL) A character vector of with the alternative base of the variants in data. Used to determine the color. If NULL and data is a VRanges or a VCF file, the information in data will be used (defaults to NULL)
col	(a color vector, a color table or a color schema) The colors to use to draw the points. If the length of the vector is lower than the length of data, it will be recycled. Color table and color schema must be compatible with getVariantColors. If NULL, points will be plotted in black. (defaults to NULL)
ymin	(numeric) The minimum value to be plotted on the data panel. If NULL, it is set to 0. (deafults to NULL)
ymax	(numeric) The maximum value to be plotted on the data.panel. (defaults to 7, (equivalent to 10Mb between consecutive features))
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions. In particular col and border can be used to set the colors used.

Details

kpPlotRainfall plots the distances between a feature and the next one in a log scale along the genome. It is usually used to plot the distance between somatic mutations in order to identify regions with an accumulation of close mutations.

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object with the data computed (distances) stored at `karyoplot$latest.plot`

See Also

plotKaryotype, kpPlotDensity, kpPlotCoverage

Examples

```
set.seed(1000)

data <- createRandomRegions(nregions=2000)

kp <- plotKaryotype("hg19", plot.type=4)
kp <- kpPlotRainfall(kp, data)
kpAxis(kp, ymax=7, tick.pos=c(0:7))
```

---

kpPlotRegions *kpPlotRegions*

Description

Plots rectangles along the genome representing the regions (or intervals) specified by a GRanges object

Usage

```
kpPlotRegions(karyoplot, data, data.panel=1, r0=NULL, r1=NULL, col="black", border=NULL, avoid.overlap
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges or equivalent) It can be any of the formats accepted by the toGRanges function from the package regioneR.
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(color) The background color of the regions. (defaults to black)
border	(color) The color used to draw the border of the regions. If NULL, no border is drawn. (defaults to NULL)
avoid.overlapping	(boolean) Whether overlapping regions should be drawn as stacks (TRUE) on drawing one occluding the other in a single layer (FALSE). (defaults to TRUE)
num.layers	(numeric) The number of layers the plotting space should be divided into to allow for plotting overlapping regions. The lotting region will be divided into this many pieces regardless if any overlapping regions actually exist. If NULL, the maximum number of regions overlapping a single point in the genome. (defaults to NULL)
layer.margin	(numeric) The blank space left between layers of regions. (defaults to 0.05)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the high-level, or specialized, plotting functions of karyoploteR. It takes a GRanges object and plots its content. Overlapping regions can be stacked and the number of layers for overlapping regions can be set. In contrast with the low-level functions such as kpRect, it is not possible to specify the data using independent numeric vectors and the function only takes in GRanges.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpRect, kpSegments

Examples

```
set.seed(1000)

#Example 1: create 20 sets of non-overlapping random regions and plot them all. Add a coverage plot on top.
kp <- plotKaryotype("hg19", plot.type=1, chromosomes=c("chr1", "chr2"))
```

```
all.regs <- GRanges()

nreps <- 20
for(i in 1:nreps) {
 regs <- createRandomRegions(nregions = 100, length.mean = 10000000, length.sd = 1000000,
                             non.overlapping = TRUE, genome = "hg19", mask=NA)
  all.regs <- c(all.regs, regs)
  kpPlotRegions(kp, regs, r0 = (i-1)*(0.8/nreps), r1 = (i)*(0.8/nreps), col="#AAAAAA")
}

kpPlotCoverage(kp, all.regs, ymax = 20, r0=0.8,  r1=1, col="#CCCCFF")
kpAxis(kp, ymin = 0, ymax= 20, numticks = 2, r0 = 0.8, r1=1)


#Example 2: Do the same with a single bigger set of possibly overlapping regions

kp <- plotKaryotype("hg19", plot.type=1, chromosomes=c("chr1", "chr2"))

regs <- createRandomRegions(nregions = 1000, length.mean = 10000000, length.sd = 1000000,
                            non.overlapping = FALSE, genome = "hg19", mask=NA)

kpPlotRegions(kp, regs, r0 = 0, r1 = 0.8, col="#AAAAAA")

kpPlotCoverage(kp, regs, ymax = 20, r0=0.8,  r1=1, col="#CCCCFF")
kpAxis(kp, ymin = 0, ymax= 20, numticks = 2, r0 = 0.8, r1=1)
```

---

kpPlotRibbon *kpPlotRibbon*

Description

A variable width ribbon

Usage

```
kpPlotRibbon(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=NULL, y0=NULL, y1=NULL, ymin=NULL, ymax=NULL,
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data), x0 to start(data) and x1 to end(data). If no parameter y0 is specified and data has a column named y0, this column will be used. The same for y1. (defaults to NULL)

chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(a numeric vector) A numeric vector of x left positions (in base pairs). If data is not NULL, x0. (defaults to NULL)
x1	(a numeric vector) A numeric vector of x right positions (in base pairs). If data is not NULL, x1. (defaults to NULL)
y0	(a numeric vector) A numeric vector of y bottom positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
y1	(a numeric vector) A numeric vector of y top positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
col	(color) The background color to plot. If NULL, it will be a lighter version of 'border' or 'black' if border is null. (Defaults to "gray80")
border	(color) The color to use to plot the borders of the bars. If NULL, it will be a darker version of 'col'. If NA, no border will be plotted. (Defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

kpPlotRibbon plots a variable witdh ribbon along the genome. It can be used, for example, to plot the sd region around a line representing a mean. It can also be used as a replacement for kpBars creating a smoother plot without the the actual individual bars. kpPlotRibbon has three additional parameters controlling the smoothing of the lines and their colors.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpBars, kpLines

Examples

```
set.seed(1000)

data <- toGRanges(data.frame(chr="chr1", start=1e6*(0:239), end=1e6*(1:240)))
y <- ((sin(start(data))/5 + rnorm(n=24, mean=0, sd=0.1))/5)+0.5

kp <- plotKaryotype("hg19", plot.type=2, chromosomes="chr1")

kpPlotRibbon(kp, data, y0=y-0.3, y1=y+0.3, border="red", col=lighter("red"))
kpPlotRibbon(kp, data, y0=y-0.1, y1=y+0.1, border="blue", col=lighter("blue"))
kpLines(kp, data, y=y, col="green")
kpPlotRibbon(kp, data, y0=0.5+(y-min(y)), y1=0.5-(y-min(y)), data.panel=2)
```

---

kpPlotTranscripts *kpPlotTranscripts*

Description

Plot gene transcripts on the genome, with options to add strand markers and to differentiate between coding and non-coding exons.

Usage

```
kpPlotTranscripts(karyoplot, data, y0=NULL, y1=NULL, non.coding.exons.height=0.5,
detail.level=2,
add.strand.marks=TRUE, mark.height=0.20, mark.width=1, mark.distance=4,
add.transcript.names=TRUE, transcript.names=NULL, transcript.name.position="left", transcript.name.c
col="black", border=NULL, coding.exons.col=NULL, coding.exons.border.col=NULL,
non.coding.exons.col=NULL, non.coding.exons.border.col=NULL,
introns.col=NULL, marks.col=NULL, transcript.name.col=NULL,
ymax=NULL, ymin=NULL, r0=NULL, r1=NULL,
data.panel=1, clipping=TRUE, ...)
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.

`data` (a `list`) data must be a list with an element called 'transcripts' with a `GRanges` with all transcripts to be plotted. Additionally, to plot the transcript structure it needs two other elements, 'coding.exons' and 'non.coding.exons', each a list with an element for every transcript with `GRanges` objects representing the transcript exons. An example of the data structure needed can be seen in the function examples.

`y0` (numeric) The bottom of the transcripts in the y axis. It can have a different value for each transcript and values will be recycled if needed. If null, it will be set to the minimum y value in the data.panel, usually 0. (Defaults to NULL)

`y1` (numeric) The top of the transcripts in the y axis. It can have a different value for each transcript and values will be recycled if needed. If null, it will be set to the maximum y value in the data.panel, usually 1. (Defaults to NULL)

`non.coding.exons.height`
(numeric) The height of the non.coding exons relative to the transcript height. For example, if 0.5, non-coding exons will have a height half the size of the coding ones. (default 0.5)

`detail.level` (numeric: 1 or 2) The detail level of the transcript representation: 1 will plot only boxes representing the transcripts, 2 will plot detailed structure of the trasncripts (coding and non-coding exons and introns). (Defaults to 2)

`add.strand.marks`
(boolean) Whether strand marks should be plotted or not. Strand marks are small arrows along the introns (or whole transcripts if detail level=1). (defaults to TRUE)

`mark.height` (numeric) The height of the strand marks in "coding exons heights", that is, if mark.height is 0.5, the mark will have a height of half the height of an exon. (defaults to 0.2)

`mark.width` (numeric) The width of the strand marks, in mark heights. mark.width=1 will produce arrow heads with a slope pf 45 degrees. A value higher than 1 will produce smaller angles and a value below 1 larger angles with more vertical lines. (defaults to 1, 45 degrees)

`mark.distance` (numeric) The distance between marks, in mark widths. A distance of 2, will add a space of 2*mark.width between consecutive marks. (defaults to 4)

`add.transcript.names`
(boolean) Whether to add transcript names to the plot (defailts to TRUE)

`transcript.names`
(named character) A named character vector with the labels of the transcripts. If not null, it will be used as a dictionary, so transcript ids should be names and desired labels the values. If NULL, the transcript ids will be used as labels. (defaults to null)

`transcript.name.position`
(character) The position of the text relative to the rectangle. Can be "left", "right", "top", "bottom" or "center". (Defaults to "left")

`transcript.name.cex`
(numeric) The cex value to plot the transcript labels. (defaults to 1)

`col` (color) The color of the transcripts and labels. It is possible to specify different colors for each element class (transcript names, exons, strand marks...). All elements with no explicit color will be plotted using col. (defaults to "black)

`border` (color) The color of the border of rectangles representing genes, transcripts and exons. Every element class may have its own specific color using the appropiate parameters. The ones with no explicit color will use border. At the same time, if border is NULL, it will default to col. (fesults to NULL)

`coding.exons.col` (color) The fill color of the rectangles representing the coding exons. If NULL, it will use col. (defaults to NULL)

`coding.exons.border.col` (color) The color of the border of the coding exons. If NULL, it will use border. (defaults to NULL)

`non.coding.exons.col` (color) The fill color of the rectangles representing the non-coding exons. If NULL, it will use col. (defaults to NULL)

`non.coding.exons.border.col` (color) The color of the border of the non-coding exons. If NULL, it will use border. (defaults to NULL)

`introns.col` (color) The color of the lines representing the introns. If NULL, it will use col. (defaults to NULL)

`marks.col` (color) The color of the arrows representing the strand. If NULL, it will use col. (defaults to NULL)

`transcript.name.col` (color) The color of the transcript labels. If NULL, it will use col. (defaults to NULL)

`ymax` (numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)

`ymin` (numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)

`r0` (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

`r1` (numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)

`data.panel` (numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to `plotKaryotype`. (defaults to 1)

`clipping` (boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data

overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

... The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the high-level, or specialized, plotting functions of karyoploteR. It takes a list with the transcripts, coding and non-coding exons and creates a traditional boxes and line representation of the transcripts. With y0 and y1, it is possible to specify a different vertical position and different height for each transcript. It can add little arrows, strand marks, along the introns to show the transcript strand. The marks appearance can be customized using 4 different parameters specifying the height (relative to the height of the transcript), width of each mark (relative to the height), distance between marks (relative to the width) and color of the marks. Marks are centered on the space they have available and if the available space to too tight for a single mark, no mark will be plotted. The direction of the marks is based on the transcript strand as specified in `data$transcripts` object. Two detail levels are available: `detail.level=1` will represent transcripts as solid boxes (optionally with strand marks along the whole transcript); `detail.level=2` will represent the internal structure of the transcripts -coding and non coding exons, introns, and optionally the strand marks only in the introns.

Value

Returns the original karyoplot object, unchanged.

Note

IMPORTANT: The direction of the strand marks is taken from the strand information in the `data$transcripts` object. If transcripts have no strand information there (they have `strand="*"`), no marks will be drawn.

See Also

kpPlotGenes

Examples

```
#Build the data objet expected by transcripts

transcripts <- c(toGRanges("chr1", 100, 1000),
                 toGRanges("chr1", 1500, 3000))
names(transcripts) <- c("T1", "T2")
strand(transcripts) <- c("+", "-")

coding.exons <- list("T1"=c(toGRanges("chr1", 200, 300),
                            toGRanges("chr1", 500, 800),
                            toGRanges("chr1", 900, 950)),
                     "T2"=c(toGRanges("chr1", 2200, 2300),
                            toGRanges("chr1", 2500, 2510),
```

```
                              toGRanges("chr1", 2700, 2800)))


non.coding.exons <- list("T1"=c(toGRanges("chr1", 100, 199),
                                toGRanges("chr1", 951, 1000)),
                         "T2"=c(toGRanges("chr1", 1500, 1700),
                                toGRanges("chr1", 1900, 1950),
                                toGRanges("chr1", 2100, 2199),
                                toGRanges("chr1", 2801, 3000)))

data <- list(transcripts=transcripts, coding.exons=coding.exons, non.coding.exons=non.coding.exons)

#Create a simple example plot
karyoplot <- plotKaryotype(zoom=toGRanges("chr1", 0, 3200))
kpAddBaseNumbers(karyoplot, tick.dist = 400)
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0, r1=0.1)


#Create a plot with different variants of the transcripts
karyoplot <- plotKaryotype(zoom=toGRanges("chr1", 0, 3200))
kpAddBaseNumbers(karyoplot, tick.dist = 400)
#Standard
kpPlotTranscripts(karyoplot, data=data, r0=0, r1=0.1)
#Customize colors
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0.11, r1=0.21, transcript.name.position = "right", non.cod
#Change vertical position and transcript height
kpPlotTranscripts(karyoplot, data=data, y0=c(0, 0.4), y1=c(0.2, 1), r0=0.25, r1=0.8, add.transcript.names = TRUE,
#Change detail level, colors and transcript names
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0.9, r1=1, detail.level = 1, add.transcript.names = TRUE, t

#Create a plot with different variants of the strand marks
karyoplot <- plotKaryotype(zoom=toGRanges("chr1", 0, 3200))
kpAddBaseNumbers(karyoplot, tick.dist = 400)
#Standard
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0, r1=0.1)
#No marks
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0.15, r1=0.25, add.strand.marks=FALSE)
#Change the strand marks height
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0.3, r1=0.4, mark.height=1)
#Change the mark width
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0.45, r1=0.55, mark.width=2)
#Change the mark distance
kpPlotTranscripts(karyoplot, data=data, y0=0, y1=1, r0=0.6, r1=0.7, mark.distance=1.5)
```

---

kpPoints *kpPoints*

Description

Plots data points along the genome.

Usage

```
kpPoints(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, ymin=NULL, ymax=NULL, data.panel=1, r0=NULL,
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)

pch	(numeric) the glyph to represent the points as specified in par. (defaults to 16, a solid circle)
cex	(numeric) the relative size of the glyphs as defined at par. (defaults to 0.5)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome and base) and a value (y) for each of them, it plots the set of points representing them. Data can be provided via a GRanges object (data), independent parameters for chr, x and y or a combination of both. A number of parameters can be used to define exactly where and how the points are drawn. In addition, via the ellipsis operator (...), kpPoints accepts any parameter valid for points (e.g. pch, cex, col, ...)

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object, unchanged.

Note

The parameter r0 can be used to specify r0 and r1 together. If r1 is NULL and r0 is either a list with two elements called r0 and r1 or a numeric vector of length 2, this values will be used for r0 and r1. This might be useful when working with autotrack to compute r0 and r1.

See Also

plotKaryotype, kpLines, kpText

kpPlotRegions

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, mask=NA))
mcols(data.points) <- data.frame(y=runif(500, min=0, max=1))

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpLines(kp, data=data.points, col="red")

  #Three ways of specifying the exact same data.points
  kpPoints(kp, data=data.points, cex=0.5)
  kpPoints(kp, data=data.points, y=data.points$y, pch=16, col="#CCCCFF", cex=0.6)
```

```
kpPoints(kp, chr=as.character(seqnames(data.points)),
         x=(start(data.points)+end(data.points))/2,
         y=data.points$y, pch=".", col="black", cex=1)

#plotting in the data.panel=2 and using r0 and r1, ymin and ymax
kpLines(kp, data=data.points, col="red", r0=0, r1=0.3, data.panel=2)
#and we can specify r0 and r1 in r0
kpPoints(kp, data=data.points, r0=list(r0=0, r1=0.3), data.panel=2, pch=".", cex=3)
kpLines(kp, data=data.points, col="blue", r0=0.4, r1=0.7, data.panel=2)
kpLines(kp, data=data.points, col="blue", y=-1*(data.points$y),
        ymin=-1, ymax=0, r0=0.7, r1=1, data.panel=2)
#It is also possible to "flip" the data by giving an r0 > r1
kpPoints(kp, data=data.points, col="red", y=(data.points$y),
         r0=1, r1=0.7, data.panel=2, pch=".", cex=2)
```

---

kpPolygon *kpPolygon*

Description

Plots the the given polygons along the genome

Usage

```
kpPolygon(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, ymin=NULL, ymax=NULL, data.panel=1, r0=NULL,
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data) and x to the midpoints of the rages in data. If no parameter y is specified and data has a column named y or value this column will be used to define the y value of each data point. (defaults to NULL)
chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)

ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome, base and y), it plots the polygons defined by taking these position as vertices. Data can be provided via a GRanges object (data), independent parameters for chr, x and y or a combination of both. A number of parameters can be used to define exactly where and how the polygon is drawn. In addition, via the ellipsis operator (...), kpPolygon accepts any parameter valid for polygon (e.g. border, density, fillOddEven, ...)

Value

Returns the original karyoplot object, unchanged.

Note

IMPORTANT: kpPolygon allows the creation of polygons encompassing multilple chromosomes. In some cases, when plotting only some of the chromosomes or when zooming, the default data filtering automatically discards some points before plotting, altering the polygon shape. See example below.

See Also

plotKaryotype, kpLines, kpPoints

kpPlotRegions

Examples

```
set.seed(1000)
x <- c(1,2,5,9,13,20,15,11,7,3)*10000000
y <- c(0,1,0.8,0.2,0.5,0.2,1,0.3,0.1,0.2)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpPolygon(kp, chr="chr1", x=x, y=y, col="red")
  kpPolygon(kp, chr="chr1", x=x, y=y, col="orange", r0=0.2, r1=0.8, density=30)
  #use kpPolygon to draw triangles at the specified positions
  chr2.x <- c(1,3,7,26,48,79,120, 124, 128)*1000000
  for(x in chr2.x) {
   kpPolygon(kp, chr="chr2", x=c(x-2000000, x+2000000, x), y=c(1,1,0), r0=0, r1=0.3, col="lightblue")
  }


#Effect of data filtering

dp <- toGRanges(data.frame(rep(paste0("chr", (1:2)), 3), 10e6*1:6, 10e6*1:6+5e5, y=c(0,0,1,1,0,0)))
kp <- plotKaryotype(chromosomes=c("chr1", "chr2"))
kpPolygon(kp, dp)

kp <- plotKaryotype(chromosomes=c("chr2"))
kpPolygon(kp, dp)
```

---

kpRect *kpRect*

Description

Plots rectangles at the specified genomic positions.

Usage

```
kpRect(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=x0, y0=NULL, y1=NULL, ymax=NULL, ymin=NULL, r0=NULL
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data), x0 to start(data) and x1 to end(data). If no parameter y0 is specified and data has a column named y0, this column will be used. The same for y1. (defaults to NULL)

chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(a numeric vector) A numeric vector of x left positions (in base pairs). If data is not NULL, x0. (defaults to NULL)
x1	(a numeric vector) A numeric vector of x right positions (in base pairs). If data is not NULL, x1. (defaults to NULL)
y0	(a numeric vector) A numeric vector of y bottom positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
y1	(a numeric vector) A numeric vector of y top positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome, x0 and x1) and values (y0 and y1) for each of them, it plots rectangles going from (x0, y0) to (x1, y1). Data can be provided via a GRanges object (data), independent parameters for chr, x0, x1, y0 and y1, or a combination of both. A number of parameters can be used to define exactly where and how the rectangles are drawn. In addition, via the ellipsis operator (...), kpRect accepts any parameter valid for rect (e.g. border, col, ...)

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpLines, kpPoints

kpPlotRegions

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, length.mean=2000000, mask=NA))
y <- runif(500, min=0, max=0.8)
mcols(data.points) <- data.frame(y0=y, y1=y+0.2)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpRect(kp, data=data.points, col="black")
 kpRect(kp, data=randomizeRegions(data.points, mask=NA), y0=0, y1=1,  r0=0, r1=0.2, border=NA, col="lightblue", d
 kpRect(kp, data=randomizeRegions(data.points, mask=NA), y0=0, y1=1,  r0=0.3, r1=0.5, border=NA, col="lightgreen"
 kpRect(kp, data=randomizeRegions(data.points, mask=NA), y0=0, y1=1,  r0=0.6, r1=0.8, border=NA, col="purple", da
```

---

kpSegments *kpSegments*

Description

Plots segments at the specified genomic positions.

Usage

```
kpSegments(karyoplot, data=NULL, chr=NULL, x0=NULL, x1=NULL, y0=NULL, y1=NULL, ymin=NULL, ymax=NULL, d
```

Arguments

karyoplot	(a KaryoPlot object) This is the first argument to all data plotting functions of karyoploteR. A KaryoPlot object referring to the currently active plot.
data	(a GRanges) A GRanges object with the data. If data is present, chr will be set to seqnames(data), x0 to start(data) and x1 to end(data). If no parameter y0 is specified and data has a column named y0, this column will be used. The same for y1. (defaults to NULL)

chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x0	(a numeric vector) A numeric vector of x left positions (in base pairs). If data is not NULL, x0. (defaults to NULL)
x1	(a numeric vector) A numeric vector of x right positions (in base pairs). If data is not NULL, x1. (defaults to NULL)
y0	(a numeric vector) A numeric vector of y bottom positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
y1	(a numeric vector) A numeric vector of y top positions. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome, x0 and x1) and values (y0 and y1) for each of them, it plots segments going from (x0, y0) to (x1, y1). Data can be provided via a GRanges object (data), independent parameters for chr, x0, x1, y0 and y1, or a combination of both. A number of parameters can be used to define exactly where and how the segments are drawn. In addition, via the ellipsis operator (...), kpSegments accepts any parameter valid for segments (e.g. lwd, lty, col, ...)

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpRect, kpPoints

kpPlotRegions

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, length.mean=2000000, mask=NA))
y <- runif(500, min=0, max=0.8)
mcols(data.points) <- data.frame(y0=y, y1=y+0.2)

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpRect(kp, data=data.points, col="black")
  kpSegments(kp, data=data.points, col="white")

 kpSegments(kp, data=data.points, y0=0, y1=1,  r0=0.2, r1=0.8, col="lightblue", data.panel=2)
 kpSegments(kp, data=data.points, y0=0, y1=1,  r0=0.8, r1=0.2, col="lightgreen", data.panel=2)
```

---

kpText *kpText*

Description

Plots the text given in `labels` at the positions defined by chr, x and y along the genome.

Usage

```
kpText(karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, labels=NULL, ymin=NULL, ymax=NULL, data.panel=1
```

Arguments

`karyoplot`	(a `KaryoPlot` object) This is the first argument to all data plotting functions of `karyoploteR`. A KaryoPlot object referring to the currently active plot.
`data`	(a `GRanges`) A GRanges object with the data. If `data` is present, `chr` will be set to `seqnames(data)` and `x` to the midpoints of the rages in data. If no parameter `y` is specified and `data` has a column named `y` or `value` this column will be used to define the `y` value of each data point. (defaults to NULL)

chr	(a charecter vector) A vector of chromosome names specifying the chromosomes of the data points. If data is not NULL, chr is ignored. (defaults to NULL)
x	(a numeric vector) A numeric vector with the positions (in base pairs) of the data points in the chromosomes. If data is not NULL, x is ignored. (defaults to NULL)
y	(a numeric vector) A numeric vector with the values of the data points. If y is not NULL, it is used instead of any data column in data. (defaults to NULL)
labels	(a character vector) The labels to be plotted. (defaults to NULL)
ymin	(numeric) The minimum value of y to be plotted. If NULL, it is set to the min value of the selected data panel. (defaults to NULL)
ymax	(numeric) The maximum value of y to be plotted. If NULL, it is set to the max value of the selected data panel. (defaults to NULL)
data.panel	(numeric) The identifier of the data panel where the data is to be plotted. The available data panels depend on the plot type selected in the call to plotKaryotype. (defaults to 1)
r0	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
r1	(numeric) r0 and r1 define the vertical range of the data panel to be used to draw this plot. They can be used to split the data panel in different vertical ranges (similar to tracks in a genome browser) to plot differents data. If NULL, they are set to the min and max of the data panel, it is, to use all the available space. (defaults to NULL)
clipping	(boolean) Only used if zooming is active. If TRUE, the data representation will be not drawn out of the drawing area (i.e. in margins, etc) even if the data overflows the drawing area. If FALSE, the data representation may overflow into the margins of the plot. (defaults to TRUE)
...	The ellipsis operator can be used to specify any additional graphical parameters. Any additional parameter will be passed to the internal calls to the R base plotting functions.

Details

This is one of the functions from karyoploteR implementing the adaptation to the genome context of basic plot functions from R base graphics. Given a set of positions on the genome (chromosome and base), a value (y) for each of them and a label, it plots the label at the position specified by the data point. Data can be provided via a GRanges object (data), independent parameters for chr, x and y or a combination of both. A number of parameters can be used to define exactly where and how the text is drawn. In addition, via the ellipsis operator (...), kpText accepts any parameter valid for text (e.g. cex, col, ...)

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype, kpLines, kpPoints

kpPlotRegions

Examples

```
set.seed(1000)
data.points <- sort(createRandomRegions(nregions=500, mask=NA))
mcols(data.points) <- data.frame(y=runif(500, min=0, max=1))

kp <- plotKaryotype("hg19", plot.type=2, chromosomes=c("chr1", "chr2"))
  kpDataBackground(kp, data.panel=1)
  kpDataBackground(kp, data.panel=2)

  kpLines(kp, data=data.points, col="red")

  #Three ways of specifying the exact same data.points
  kpPoints(kp, data=data.points)
  kpPoints(kp, data=data.points, y=data.points$y, pch=16, col="#CCCCFF", cex=0.6)
  kpPoints(kp, chr=as.character(seqnames(data.points)),
           x=(start(data.points)+end(data.points))/2,
           y=data.points$y, pch=".", col="black", cex=1)

  #plotting in the data.panel=2 and using r0 and r1, ymin and ymax
  kpLines(kp, data=data.points, col="red", r0=0, r1=0.3, data.panel=2)
 kpText(kp, data=data.points, labels=as.character(1:500), r0=0, r1=0.3, data.panel=2, pch=".", cex=3)

  kpLines(kp, data=data.points, col="blue", r0=0.4, r1=0.7, data.panel=2)
 kpLines(kp, data=data.points, col="blue", y=-1*(data.points$y), ymin=-1, ymax=0, r0=0.7, r1=1, data.panel=2)
  #It is also possible to "flip" the data by giving an r0 > r1
 kpPoints(kp, data=data.points, col="red", y=(data.points$y), r0=1, r1=0.7, data.panel=2, pch=".", cex=2)
```

`lighter` *lighter*

Description

Given a color, return a lighter one

Usage

```
lighter(col, amount=150)
```

Arguments

col	(color) The original color. Might be specified as a color name or a "#RRGGBB(AA)" hex color definition.
amount	(integer, [0-255]) The fixed amount to add to each RGB channel (Defaults to 150).

Details

Very simple utility function to create lighter colors. Given a color, it transforms it to rgb space, adds a set amount to all chanels and transforms it back to a color.

Value

A lighter color

See Also

darker

Examples

```
lighter("red")
lighter("#333333")
lighter(c("red", 3, "#FF00FF"))
```

---

makeGenesDataFromTxDb *makeGenesDataFromTxDb*

Description

This is a utility function that transforms a TxDb object into a custom object valid as input for kpPlotGenes.

Usage

```
makeGenesDataFromTxDb(txdb, karyoplot=NULL, plot.transcripts=TRUE, plot.transcripts.structure=TRUE)
```

Arguments

txdb	(a TxDb object) The transcript database object from which the data will be extracted.
karyoplot	(karyoplot object) A valid karyoplot object created by a call to plotKaryotype. If present, genes data will be restricted to the visible region in the plot. If NULL, all genes will be included. (defaults to NULL)
plot.transcripts	(boolean) TRUE if transcripts are needed in addition to the genes. (defaults to TRUE)

plot.transcripts.structure
: (boolean) TRUE if the coding and non-coding exons are needed in addition to the genes and transcripts. (defaults to TRUE)

Details

This function creates a valid data object for kpPlotGenes starting from a TxDb object. The resulting object will contain only the genes and transcripts ovelapping the plot region of the given Karyoplot object.

Value

Returns a list with at least one element called genes, a GRanges with all genes overlapping karyoplot. If plot.transcripts is TRUE, the returned list will have a transcript element, a list of GRanges objects, one per gene (named with the gene ids), with the transcripts of that gene. If plot.transcripts.structure is TRUE, two more elements are present: coding.exons and non.coding.exons, each a list with one element per trascript (named with the transcript id), and each element the coding or non-coding exons of that transcript.

Note

kpPlotGenes accepts TxDb objects directly. This function is only expected to be used when the user want to manipulate the results somehow (i.e. removing some of the genes).

See Also

kpPlotGenes

Examples

```
library(TxDb.Hsapiens.UCSC.hg19.knownGene)

zoom <- toGRanges("chr17", 32.6e6, 33.2e6)
kp <- plotKaryotype(genome="hg19", zoom=zoom)

genes.data <- makeGenesDataFromTxDb(TxDb.Hsapiens.UCSC.hg19.knownGene,
                karyoplot=kp, plot.transcripts=TRUE,
                plot.transcripts.structure=TRUE)
```

---

mergeTranscripts *mergeTranscripts*

Description

Merges the transcripts of each gene and creates one transcript per gene with all exons and UTR regions combined

Usage

```
mergeTranscripts(genes.data)
```

Arguments

genes.data	(GenesData object) A valid genes.dat object like the ones obtained by makeGenesDataFromTxDb

Details

This function takes a valid data object and merges all transcripts from each gene into a single transcript. This is useful to reduce the plot complexity while keeping partial information on transcript structure.#' In this transcript, any region that is a coding exon in any transcript, will be an exon, any region that is a non-coding exon in any transcript and is not an exon in any transcript, will be a non-coding exon. Anything between coding and non-coding exons will be introns.

Value

The original GenesData object with a single transcript per gene `GenesData$genes$names`.

See Also

kpPlotGenes, makeGenesDataFromTxDb

Examples

```
library(TxDb.Hsapiens.UCSC.hg19.knownGene)

zoom <- toGRanges("chr17:29.4e6-29.8e6")
kp <- plotKaryotype(genome="hg19", zoom=zoom)
genes.data <- makeGenesDataFromTxDb(TxDb.Hsapiens.UCSC.hg19.knownGene, karyoplot=kp)
genes.data <- addGeneNames(genes.data)
kpPlotGenes(kp, data=genes.data, r1=0.5, plot.transcripts=TRUE, gene.name.position = "left")
genes.data.merged <- mergeTranscripts(genes.data)
kpPlotGenes(kp, data=genes.data.merged, r0=0.6, r1=0.8, plot.transcripts=TRUE, gene.name.position = "left")
```

---

plotDefaultPlotParams *plotDefaultPlotParameters*

Description

Creates a karyoplot with the default parameters drawn.

Usage

```
plotDefaultPlotParams(plot.type=2, plot.params=NULL, ...)
```

Arguments

plot.type	(numeric) plot the params of this plot type. Currently, only plot types 2 and 3 are accepted. (defaults to 2)
plot.params	(a plot params object) a plot params object such the one returned by getDefaultPlotParams. If specified, it will be used to create the plots.
...	The ellipsis operator can be used to pass any additional graphics parameter

Details

Given a plot.type, this function creates a new karyoplot with lines and arrows showing the meaning and values of the plot.params

Value

Returns the original karyoplot object, unchanged.

See Also

plotKaryotype

Examples

```
kp <- plotDefaultPlotParams(plot.type=2)
```

---

plotKaryotype *plotKaryotype*

Description

Create a new empty plot with a karyotype (the chromosome ideograms and chromosome names).

Usage

```
plotKaryotype(genome="hg19", plot.type=1, ideogram.plotter=kpAddCytobands, labels.plotter=kpAddChrom
```

Arguments

genome	The genome to plot. It can be either a UCSC style genome name (hg19, mm10, etc), a BSgenome, a Seqinfo object, a GRanges object with the chromosomes as ranges or in general any genome specification accepted by getGenomeAndMask. (defaults to "hg19")
plot.type	The orientation of the ideogram and placing of the data panels. Values explained above.. (defaults to 1)

`ideogram.plotter`
: The function to be used to plot the ideograms. Only one function is included with the package, `kpAddCytobands`, but it is possible to create custom ones. If NULL, no ideograms are plotted. (defaults to `kpAddCytobands`)

`labels.plotter`
: The function to be used to plot the labels identifying the chromosomes. Only one function is included with the package, `kpAddChromosomeNames`, but it is possible to create custom ones. If NULL, no labels are plotted. (defaults to `kpAddChromosomeNames`)

`chromosomes`
: The chromosomes to plot. Can be either a vector of chromosome names or a chromosome group name ("canonical", "autosomal", "all"). Setting it yo "auto" will select canonical. If no predefined filtering data is available, a heuristic filtering will be used. To deactivate filtering set chromosomes="all". (defaults to "auto")

`zoom`
: A GRanges object specifiyng a single region to zoom in or any format accepted by `regioneR::toGRanges`. If not NULL, it takes precedence over `chromosome` and only the zoomed in region is represented. If more than one region is present in the GRanges, only the first one is used. (defaults to NULL, do not zoom in and show the whole plot as specified by `genome` and `chromosomes`)

`cytobands`
: A GRanges object (or anything accepted by `toGRanges` function: bed file, data.frame...) specifying the positions and types of the cytobands. The type of the cytobands MUST be in a column named "gieStain" (as used by UCSC) with values such as 'gneg', 'gpos50', 'stalk', 'acen'... If NULL, the cytobands are recovered from the package cache or downloaded from UCSC if possible (it's not possible for custom genomes). If empty, no cytobands will be plotted. (defaults to NULL)

`plot.params`
: An object obtained from `getDefaultPlotParams` and possibly modified, containing the basic plotting parameters. If NULL, the defaults parameters will be used. (defaults to NULL)

`use.cache`
: `karyoploteR` has a small cache with the chromosome names and lengths and the cytobands for a handful of organisms so it's not needed to retrieve them from databses or `BSGenomes` objects. Set this parameter to FALSE to ignore the cache. (defaults to TRUE, use the cache)

`main`
: The text to be used as the title of the plot. NULL produces no title. (defaults to NULL)

`...`
: The ellipsis can be used to pass in any additional parameter accepted by the internal functions used.

Details

This is the main function of `karyoploteR`. It creates the basic empty plot with the chromosome ideograms and returns the karyoplot object needed for all other plotting functions. Both the basic plotting parameters (margins, sizes, etc.) and the specific plotting functions for the ideograms and chromosome labels are customizable. In particular, passing in a `plot.params` object specifies the basic plotting parameters to use and the `ideogram.plotter` and `labels.plotter` parameters can be used to specify custom plotting functions for the ideogram and the chromosome labels. It is also possible to specify the genome and a list with the chromosomes to be plotted.

The `plot.type` parameter specifies the type of karyoplot to create: the number and positions of the data panels respect to the ideograms:

- plot.type=1 Horizontal ideograms with a single data panel above them
- plot.type=2 Horizontal ideograms with two data panels, one above and one below them
- plot.type=3 Horizontal ideograms with all chromosomes in a single line with two data panels, one above and one below them
- plot.type=4 Horizontal ideograms with all chromosomes in a single line with one data panel above
- plot.type=5 Horizontal ideograms with all chromosomes in a single line with one data panel below them
- plot.type=6 Horizontal ideograms with NO data panels. Only plotting in the ideograms is possible.
- plot.type=7 Horizontal ideograms with all chromosomes in a single line with NO data panels. Only plotting in the ideograms is possible.

There's more information at the https://bernatgel.github.io/karyoploter_tutorial/karyoploteR tutorial.

Value

The KaryoPlot object needed by the plotting functions.

See Also

getDefaultPlotParams, kpPoints

Examples

```
 set.seed(1000)

rand.data <- createRandomRegions(genome="hg19", nregions=10000, length.mean=1,
                                 length.sd=0, mask=NA, non.overlapping=TRUE)
mcols(rand.data) <- data.frame(y=rnorm(n=10000, mean = 0.5, sd=0.1))

#The simplest way, with all default parameters
kp <- plotKaryotype()
kpPoints(kp, rand.data, pch=".")

#Or we can plot only a few chromosomes, with 2 data panels
kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 2)
kpDataBackground(kp, data.panel = 1, color = "lightgreen")
kpDataBackground(kp, data.panel = 2, color = "lightblue")
kpPoints(kp, rand.data, pch=".", data.panel = 1)
kpPoints(kp, rand.data, pch=".", data.panel = 2)


#Or we can use a different organism,
kp <- plotKaryotype(genome = "mm10")
kp <- plotKaryotype(genome = "dm6")

# Or we can change the plotting parameters. In this case, to create a smaller ideogram
# and smaller data panel below it
```

```
plot.params <- getDefaultPlotParams(plot.type=2)
plot.params$ideogramheight <- 5
plot.params$data2height <- 50

kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 2, plot.params = plot.params)
kpDataBackground(kp, data.panel = 1, color = "lightgreen")
kpDataBackground(kp, data.panel = 2, color = "lightblue")
kpPoints(kp, rand.data, pch=".", data.panel = 1)
kpPoints(kp, rand.data, pch=".", data.panel = 2)

#Or we can remove the cytobands, passing an empty GRanges object
kp <- plotKaryotype(cytobands = GRanges())

#Or remove the chromosome labels
kp <- plotKaryotype(labels.plotter = NULL)
kpPoints(kp, rand.data, pch=".")

#In addition, it's possible to use maggrittr piping to chain the plotting calls
library(magrittr)
kp <- plotKaryotype() %>%
   kpDataBackground(color = "lightgreen") %>%
   kpPoints(rand.data, pch=".")
```

`plotPalettes` *plotPalette*

Description

Create a plot of the palette.

Usage

```
plotPalettes(cols, add.color.name=TRUE, border=NA, palette.names.col="black", palette.names.cex=1, pa
```

Arguments

`cols`	(color vector or list of color vectors) The colors to plot
`add.color.name`	(logical) Wether to add or not the names of the colors, their definition.
`border`	(color) The color of the border of the palette rectangles. If NA, no border. (defaults to NA)
`palette.names.col`	(color) The color of the palette names (defaults to "black")
`palette.names.cex`	(numeric) The cex value (size) for the palette names (defaults to 1)
`palette.names.srt`	(numeric) The srt value (rotation) for the palette names (defaults to 0)

color.names.col
: (color) The color of the color names. If auto, it will be blak for light colors and white for the dark ones (defaults to "auto")

color.names.cex
: (numeric) The cex value (size) for the color names (defaults to 1)

color.names.srt
: (numeric) The srt value (rotation) for the color names (defaults to 0)

...
: Any additional plotting parameters

Details

Creates a simple plot with a rectangle for every color of every palette. cols must be either a vector of colors (in any format accepted by karyoploteR::"is.color") or a list of such vectors. The names of the list elements will be treated as the palette names, if the list has no names, palettes will be called "Pallete1", "Palette2", ...

Value

nothing

Examples

```
plotPalettes(c("red", "blue", "yellow", "green"))
palettes <- list("P1"=c("red", "#000000", lighter("gold")),
                 "P2"=c("orchid", "yellow"))
plotPalettes(palettes, color.names.col=c("blue", "green", "red"), border="black", color.names.srt=45)
plotPalettes(palettes, color.names.col="auto", border="black", color.names.srt=45)
```

---

prepareParameters2 *prepareParameters2*

Description

Prepare and normalize the parameters for functions with x and y parameters

Usage

```
prepareParameters2(function.name, karyoplot, data=NULL, chr=NULL, x=NULL, y=NULL, ymax=NULL, ymin=NULL
```

Arguments

function.name
: (character) The name of the function calling prepareParameters2. Only user for error reporting.

karyoplot
: (KaryoPlot) A karyoplot object.

data
: A GRanges. It can be NULL or a GRanges.

chr
: A character representing the chromosome names.

x	The position in the chromosome in number of bases.
y	The value to be plotted.
ymax	The maximum value of y
ymin	The minimum value of y
r0	The start of the range to use for plotting
r1	The end of the range to use for plotting
data.panel	The data panel to use
filter.data	A boolean indicating if data should be filtered so only data in visible chromosomes is kept. (defaults to TRUE, filter data)
...	Any additional parameter

Details

This function prepares and normalizes the parameters for plotting functions with x and y parameters (as opposed to x0, x1, y0 and y1) so functions can offer a richer interface while internally dealing only with standard and simple code. It extracts the positions from `data` if available and applies the `r0` and `r1` scaling. It returns the ready to plot values in a list with only `chr`, `x` and `y`. Individual parameters (`chr`, `x` and `y`) take precedence over `data`. All parameters are interpreted and used as explained in `kpPoints`. It also filters out any data points corresponding to chromosomes not present in the current karyoplot.

Value

A list with three values: `chr`, `x` and `y`. Each of them a vector of the same length with the normalized values to plot.

Note

This function is only useful when creating custom plotting functions. It is not intended to the general user.

For detailed documentation on the parameters, see `kpPoints`

See Also

`kpPoints`

Examples

```
kp <- plotKaryotype()
prepareParameters2("TestFunc", kp, data=NULL, chr="chr1", x=c(10, 20, 30), y=c(0, 1, 2), r0=0, r1=0.5, ymin=0, yma
```

prepareParameters4 *prepareParameters4*

Description

Prepare and normalize the parameters for functions with x0, x1 and y0, y1 parameters

Usage

```
prepareParameters4(function.name, karyoplot, data=NULL, chr=NULL, x0=NULL, x1=NULL, y0=NULL, y1=NULL,
```

Arguments

function.name	(character) The name of the function calling prepareParameters4. Only user for error reporting.
karyoplot	(KaryoPlot) A karyoplot object.
data	A GRanges
chr	A character representing the chromosome names.
x0	The position in the chromosome in number of bases.
x1	The position in the chromosome in number of bases.
y0	The value to be plotted.
y1	The value to be plotted.
ymax	The maximum value of y
ymin	The minimum value of y
r0	The start of the range to use for plotting
r1	The end of the range to use for plotting
data.panel	The data panel to use
filter.data	A boolean indicating if data should be filtered so only data in visible chromosomes is kept. (defaults to TRUE, filter data)
...	Any additional parameter

Details

This function prepares and normalizes the parameters for plotting functions with x0, x1, y0 and y1 parameters (as opposed to x and y) so functions can offer a richer interface while internally dealing only with standard and simple code. It extracts the positions from data if available and applies the r0 and r1 scaling. It returns the ready to plot values in a list with only chr, x0, x1, y0 and y1. Individual parameters (chr, x0, x1, y0 and All parameters are interpreted and used as explained in kpRect. It also filters out any data points corresponding to chromosomes not present in the current karyoplot.

Value

A list with five values: `chr`, `x0`, `x1`, `y0` and `y1`. Each of them a vector of the same length with the normalized values to plot.

Note

This function is only useful when creating custom plotting functions. It is not intended to the general user.

For detailed documentation on the parameters, see `kpRect`

See Also

`kpRect`

Examples

```
kp <- plotKaryotype()
prepareParameters4("TestFunc", kp, data=NULL, chr="chr1", x0=c(10, 20, 30), x1=c(20, 30, 40), y0=c(0, 1, 2), y1=c(0
```

---

processClipping *processClipping*

Description

Sets image clipping if needed

Usage

```
processClipping(karyoplot, clipping, data.panel)
```

Arguments

`karyoplot`	(KaryoPlot) A KaryoPlot object representing the current plot
`clipping`	(logical) Wheter clipping should be activated or not
`data.panel`	(data panel identifier) The name of the data panel on which the plot should be allowed. Anything plotted outside it will be hidden (if clipping==TRUE and the plot is a zoom plot)

Details

Small utility function to help manage clipping. If the current plot is a zoomed plot and clipping is TRUE, activate the clip to the current data.panel. This will hide any plotting ocurring out of the data.panel region.

Value

Returns the original karyoplot object, unchanged.

Note

Users wont usually use this function. It is used by the plotting functions to set the clipping if needed

Examples

```
kp <- plotKaryotype()
processClipping(kp, TRUE, 1)
```

---

transparent *transparent*

Description

Given a color, return a transparent one

Usage

```
transparent(col, amount=0.5)
```

Arguments

col	(color) The original color. Might be specified as a color name or a "#RRGGBB(AA)" hex color definition.
amount	(number, [0-1]) The amount of transparency. 0 for completely visible, 1 for completely transparent. (Defaults to 0.5).

Details

Very simple utility function to create transparent colors. Given a color, it transforms it to rgb space, adds a set amount to all chanels and transforms it back to a color.

Value

A transparent color

See Also

lighter

Examples

```
transparent("red")
transparent("#333333")
```

Index

abline, *23*
addGeneNames, 3
autotrack, 5, *30*, *92*

bamsignals, *52*
BigWigFile, *57*

colByCategory, 6
colByChr, 7, *9*, *11*, *73*, *75*
colByRegion, 8, *75*
colByValue, *6*, 10
colorRamp, *10*, *48*

darker, 11, *102*

filterParams, 12
findIntersections, 13
forget, *17*

GenomicRanges, *16*
getChromosomeNamesBoundingBox, 14, *27*
getColorSchemas, 15
getCytobandColors, 15, *32–34*
getCytobands, 16
getDataPanelBoundingBox, 17
getDefaultPlotParams, 18, *105–107*
getGenomeAndMask, *105*
getMainTitleBoundingBox, 19, *37*
getTextSize, 20
getVariantsColors, 21

horizonColors, 22

is.color, 23

kpAbline, 23, *43*
kpAddBaseNumbers, 25, *32*, *34*
kpAddChromosomeNames, *14*, 27
kpAddChromosomeSeparators, 28
kpAddColorRect, *6*, 29
kpAddCytobandLabels, 30, *32*, *34*
kpAddCytobands, *16*, 32, *33*
kpAddCytobandsAsLine, 33
kpAddLabels, 34
kpAddMainTitle, *19*, 36
kpArea, 37, *57*
kpArrows, 39
kpAxis, 41, *47*
kpBars, 44, *59*, *85, 86*
kpDataBackground, *18*, *43*, 46
kpHeatmap, 47
kpLines, *25*, *39*, *45*, *49*, 49, *50*, *67*, *72*, *78*, *86*, *92*, *94*, *97*, *101*
kpPlotBAMCoverage, 51, *59*
kpPlotBAMDensity, *52, 53*, 53, *57*
kpPlotBigWig, 55
kpPlotCoverage, *53*, *55*, 58, *61*, *82*
kpPlotDensity, *59*, 59, *82*
kpPlotGenes, *4*, 61, *89*, *102–104*
kpPlotHorizon, 66
kpPlotLinks, 68
kpPlotLoess, 71
kpPlotManhattan, 73
kpPlotMarkers, 76
kpPlotNames, 78
kpPlotRainfall, *21*, 80
kpPlotRegions, *41*, *50*, *59*, 82, *92*, *94*, *97*, *99*, *101*
kpPlotRibbon, *39*, *55*, *61*, *67*, *70*, *72*, 84
kpPlotTranscripts, *65*, 86
kpPoints, *8, 9*, *11*, *41*, *72*, *75*, 90, *94*, *97*, *99*, *101*, *107*, *110*
kpPolygon, 93
kpRect, *41*, *45*, *49*, *59*, *65*, *80*, *83*, 95, *99*, *111, 112*
kpSegments, *25*, *65*, *70*, *83*, 97
kpText, *39*, *50*, *67*, *78*, *80*, *92*, 99

lighter, *12*, 101, *113*
lines, *39*, *50*
loess, *71*

makeGenesDataFromTxDb, *4*, 102, *104*
memoise, *17*
mergeTranscripts, 103

par, *92*
plotDefaultPlotParams, 104
plotKaryotype, *14*, *16–19*, *21*, *24–32*, *34, 35*, *37–50*, *52*, *54–61*, *64–67*, *69–72*, *74, 75*, *77–79*, *81–83*, *85, 86*, *88*, *91, 92*, *94*, *96–102*, *105*, 105
plotPalettes, 108
polygon, *39*
prepareParameters2, 109
prepareParameters4, 111
processClipping, 112

Rsamtools, *54*

segments, *24*

text, *34*
toGRanges, *9*, *73, 74*, *82*, *106*
transparent, 113