Package 'karyoploteR'

April 12, 2018

Type Package

Title Plot customizable linear genomes displaying arbitrary data

Version 1.4.2

Date 2017-05-03

Author Bernat Gel <`bgel@igtp.cat`>

Maintainer Bernat Gel <`bgel@igtp.cat`>

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, GenomeInfoDb, S4Vectors, biovizBase, digest, bezier

Suggests BiocStyle, knitr, testthat, magrittr, BSgenome.Hsapiens.UCSC.hg19, pasillaBamSubset

VignetteBuilder knitr

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

NeedsCompilation no

RoxygenNote 6.0.1

R topics documented:

`darker` *darker*

Description

Given a color, return a darker one

Usage

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

Arguments

`col`	(color) The original color
`amount`	(integer, [0-255]) The fixed amount to add 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")
```

`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)
```

`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)
```

`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.

Format

An object of class `NULL` of length 0.

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: hg19, hg38, mm9, mm10, rn5, rn6, danRer10, dm6, ce6 and 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")

#do not use the included data and force the download from UCSC
cyto <- getCytobands("hg19", use.cache=FALSE)

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

#get the cytobands for Chimpanzee (not included in the package)
cyto <- getCytobands("panTro4")
```

`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)
```

`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"="#
getVariantsColors(ref, alt, col.table)
```

`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
```

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, minor.ticks=TRUE, minor.tick.dist=5000
```

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
`tick.len`	(numeric) The length of the major tick marks in plot coordinates
`minor.ticks`	(boolean) Whether to add unlabeled minor ticks between the major ticks
`minor.tick.dist`	(numeric) The distance between the minor ticks in bases
`minor.tick.len`	(numeric) The length of the minor tick marks in plot coordinates
`cex`	(numeric) The cex parameter for the major ticks label
`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)
`...`	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 witll 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, ...)
```

Arguments

`karyoplot`	a `karyoplot` object returned by a call to `plotKaryotype`
`...`	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)
```

`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, clipping=TRUE, ...)
```

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)
`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,
```

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)
```

`kpAddLabels` *kpAddLabels*

Description

Add labels to identify the data in the plot

Usage

```
kpAddLabels(karyoplot, labels, r0=NULL, r1=NULL, label.margin=0.01, data.panel=1, pos=2, offset=0
```

Arguments

`karyoplot`	a `karyoplot` object returned by a call to `plotKaryotype`
`labels`	(character) the text on the labels
`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)
`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)
`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)
`pos`	(numeric) The standard graphical parameter. See `text`. (Defaults to 2)
`offset`	(numeric) The standard graphical parameter. See `text`. (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 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
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 = 0.1, 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)
```

`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)
```

`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
```

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.po
```

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)
`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)
```

`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, 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)
`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, 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)
`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=NUL
```

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)
```

`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, da
```

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.

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="dar
kp <- kpPlotBAMDensity(kp, data = un3.bam.file, window.size = window.size, r0=0.5, r1=0, ymax=50000, col="dar
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
kp <- kpPlotBAMDensity(kp, data = un3.bam.file, window.size = window.size, normalize=TRUE, r0=0.5, r1=0, ymax
```

`kpPlotCoverage` *kpCoverage*

Description

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

Usage

```
kpPlotCoverage(karyoplot, data, data.panel=1, r0=NULL, r1=NULL, col="#0e87eb", ymax=NULL, clippin
```

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 wich the coverage will be computed or a `SimpleRleList` result of computing the coverage.
`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")
`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.

Value

Returns the original karyoplot object, unchanged.

See Also

`plotKaryotype`, `kpPlotRegions`, `kpBars`

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)
```

`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,
```

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.

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, co
```

`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, y
```

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 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.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)` 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`, `kpBars`, `kpLines`

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")
```

`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, 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)
`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.

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), la

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, da
```

`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=NULL, col=NULL, ymin=NULL, ymax=7, data.panel=1, r0=NULL, r1=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 features to be plotted.
`col`	(a color vector) 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. 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.

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` *kpPlotCoverage*

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.ov
```

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 regions to plot.
`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=
```

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)
```

`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=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)
`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`, ...)

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, 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)
  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)
```

---

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=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 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=
```

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`, ...)

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=”lightblu
  kpRect(kp, data=randomizeRegions(data.points, mask=NA), y0=0, y1=1,  r0=0.3, r1=0.5, border=NA, col=”lightg
  kpRect(kp, data=randomizeRegions(data.points, mask=NA), y0=0, y1=1,  r0=0.6, r1=0.8, border=NA, col=”purple
```

`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=NUL
```

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

`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=kpAddCh
```

Arguments

`genome`	The genome to plot. It can be either a UCSC style genome name (hg19, mm10, etc), 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 for named genomes and no filtering for custom genomes. (defaults to "auto")
`zoom`	A GRanges object specifiyng a single region to zoom in. 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 specifying the positions and types of the cytobands. If NULL, the cytobands are recovered from the package cache or downloaded from UCSC. 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 a two data panels, one above and one below them

More plot types are expected to come in the near future.

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=".")
```

`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
```

Arguments

`function.name`	(character) The name of the function calling `prepareParameters2`. Only user for error reporting.
`karyoplot`	(KaryoPlot) A karyoplot object.
`data`	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
```

`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=N
```

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),
```

Index