Package 'SNPchip'

April 12, 2018

Version 2.24.0

Title Visualizations for copy number alterations

Author Robert Scharpf <rscharpf@jhu.edu> and Ingo Ruczinski

Maintainer Robert Scharpf <rscharpf@jhu.edu>

Depends R (>= 2.14.0)

Imports methods, graphics, lattice, grid, foreach, utils, Biobase, S4Vectors (>= 0.9.25), IRanges, GenomeInfoDb, GenomicRanges, SummarizedExperiment, oligoClasses (>= 1.31.1)

Suggests crlmm (>= 1.17.14), RUnit

Enhances doSNOW, VanillaICE (>= 1.21.24), RColorBrewer

Description Functions for plotting SNP array data; maintained for historical reasons

License LGPL (>= 2)

LazyLoad yes

Collate AllGenerics.R coerce-methods.R xyplot-methods.R grid-functions.R idiogram-functions.R panel-functions.R annotation-functions.R zzz.R

URL http://www.biostat.jhsph.edu/~iruczins/software/snpchip.html

biocViews CopyNumberVariation, SNP, GeneticVariability, Visualization

NeedsCompilation no

R topics documented:

arrangeFigs . . . 2
arrangeSideBySide . . . 2
centromere . . . 3
dataFrame . . . 4
dataFrame-methods . . . 4
getCytoband . . . 5
latticeFigs . . . 5
plotIdiogram . . . 6
xypanel . . . 7
xypanelBaf . . . 9
xyplot . . . 10
xyplotLrrBaf . . . 12

Index **15**

`arrangeFigs` *Arranging two trellis objects on a grid.*

Description

Helper function for arranging multipanel displays of log R ratios and B allele frequencing in a single figure

Usage

```
arrangeFigs(lattice.figs, ...)
```

Arguments

`lattice.figs`	A named list ('lrr' and 'baf') of two `trellis` object.
`...`	ignored

Value

nothing

Author(s)

R. Scharpf

See Also

`latticeFigs`, `arrangeSideBySide`

`arrangeSideBySide` *Helper function to arrange two trellis objects side by side on a grid.*

Description

For visualizing copy number alterations, it is often helpful to plot estimates of copy number along with the corresponding estimate of the B allele frequencies. Creating a trellis object for the copy number estimates and a separate trellis object for the B allele frequencies, this function can be used to arrange the two trellis objects side by side on a grid.

Usage

```
arrangeSideBySide(object1, object2)
```

Arguments

`object1`	A trellis object (e.g., a trellis object of the copy number estimates).
`object2`	A trellis object (e.g., a trellis object of the B allele frequencies).

Author(s)

Rob Scharpf

See Also

xypanel, xyplot

`centromere` *Coordinates of centromere*

Description

Extracts coordinates of centromere for a particular chromosome

Usage

```
centromere(chromosome, build, verbose=FALSE)
```

Arguments

`chromosome`	Chromosome name. Several formats for specifying chromosome are allowed (see examples).
`build`	character string. Supported UCSC builds are 'hg18' and 'hg19'.
`verbose`	Logical. Displays build used to annotate the centromere coordinates when TRUE

Value

integer: start and stop coordinates of centromere in basepairs

Author(s)

R. Scharpf

Examples

```
centromere(1, "hg18")
centromere("1", "hg18")
centromere("chr1", "hg18")
centromere(1, "hg19")
centromere("X", "hg18")
```

dataFrame *Generic function for coercing gSet objects to data.frame*

Description

Generic function for coercing gSet objects to data.frame as a precursor to plotting with lattice

Usage

```
dataFrame(range, data, ...)
```

Arguments

range	A GenomicRanges object containing interval(s) for which low-level data should be plotted
data	A container for the low-level data (e.g., BafLrrSet) or a RangedSummarizedExperiment
...	Additional arguments passed to findOverlaps. E.g., argument maxgap can be used to select the size of the window surrounding the genomic intervals in range for plotting.

Value

A data.frame with column labels that depend on the class of data.

Author(s)

R. Scharpf

dataFrame-methods *Construct a data.frame from genomic data for plotting*

Description

Construct a data.frame of genomic data (log R ratios and BAFs) from a RangedSummarizedExperiment with markers in the interval given by the GRanges object.

Methods

signature(range = "GRanges", data = "gSet") The argument range is often intervals from a hidden Markov model fit to the genomic data in the data object. gSet-derived classes contain assay data on copy number and allele frequencies.

signature(range = "GRanges", data = "RangedSummarizedExperiment") The argument range is often intervals from a hidden Markov model fit to the genomic data in the data object. The RangedSummarizedExperiment is assumed to contain log R ratio (lrr) and B allele frequency (baf) assays.

getCytoband *getCytoband*

Description

This function generates a data.frame with the respective cytoband names, chromosomes, Giemsa stain, and the start and end positions. These tables can then be used to plot chromosome idiograms. Currently, cytoband annotation for UCSC genome builds hg18 and hg19 are supported.

Usage

```
getCytoband(build)
```

Arguments

build	A character string indicating UCSC build ("hg18" or "hg19").

Value

data.frame

Author(s)

Michael Considine

See Also

plotIdiogram

Examples

```
cytoband <- getCytoband("hg19")
cytoband <- cytoband[cytoband$chr == "chr1", ]
plotIdiogram(1, "hg18", cytoband=cytoband, cex.axis=0.6)
```

---

latticeFigs *Generate trellis objects of log R ratios and B allele frequencies*

Description

Generate trellis objects of log R ratios and B allele frequencies

Usage

```
latticeFigs(gr, data, colors, ...)
```

Arguments

gr	A GRanges object
data	A RangedSummarizedExperiment with assays "lrr" and "baf"
colors	Colors for copy number states
...	Additional arguments to panel.xyplot

Value

A list (length 2) of trellis objects with names 'lrr' and 'baf'.

Author(s)

R. Scharpf

`plotIdiogram` *Plots idiogram for one chromosome*

Description

Draw an idiogram for the specified chromosome.

Usage

```
plotIdiogram(chromosome, build, cytoband, cytoband.ycoords, xlim, ylim=c(0, 2),
new=TRUE, label.cytoband=TRUE, label.y=NULL, srt, cex.axis=1,
outer=FALSE, taper=0.15, verbose=FALSE, unit=c("bp", "Mb"),
is.lattice=FALSE,...)
plotCytoband2(chromosome, build, cytoband, xlim, xaxs="r", new=TRUE,
label.cytoband=TRUE, cex.axis=1, outer=FALSE, verbose=TRUE, ...)
```

Arguments

`chromosome`	character string or integer: which chromosome to draw the cytoband
`build`	UCSC genome build. Supported builds are "hg18" and "hg19".
`cytoband`	data.frame containing cytoband information
`cytoband.ycoords`	numeric: y coordinates
`xlim`	x-axis limits
`xaxs`	numeric. See `par`
`ylim`	y-axis limits
`new`	logical: new plotting device
`label.cytoband`	logical: if TRUE, labels the cytobands
`label.y`	numeric: height (y-coordinate) for cytoband label
`srt`	string rotation for cytoband labels. See `par`
`cex.axis`	size of cytoband labels. See par
`outer`	logical: whether to draw the labels in the outer margins. See par
`taper`	tapering for the ends of the cytoband
`verbose`	Logical. If TRUE, displays human genome build used to annotated the cytoband coordinates.
`unit`	Character string indicating the unit for physical position on the x-axis. Available options are basepairs (bp) or Mb.
`is.lattice`	logical indicating whether your drawing the cytoband on a lattice graphic.
`...`	additional arguments to plot

Author(s)

Robert Scharpf and Jason Ting

Examples

```
plotIdiogram("1", "hg18")
plotIdiogram("1", "hg19")
plotIdiogram("1", build="hg19", cex=0.8, label.cytoband=FALSE)
## user-defined coordinates
plotIdiogram("1", build="hg19", cex=0.8, label.cytoband=FALSE,
ylim=c(0,1), cytoband.ycoords=c(0.1, 0.3))

library(oligoClasses)
sl <- getSequenceLengths("hg19")[c(paste("chr", 1:22, sep=""), "chrX", "chrY")]
ybottom <- seq(0, 1, length.out=length(sl)) - 0.01
ytop <- seq(0, 1, length.out=length(sl)) + 0.01
for(i in seq_along(sl)){
chr <- names(sl)[i]
if(i == 1){
plotIdiogram("1", build="hg19", cex=0.8, label.cytoband=FALSE, ylim=c(-0.05,1.05), cytoband.ycoords=c(ybotto
     xlim=c(0, max(sl)))
}
if(i > 1){
plotIdiogram(names(sl)[i], build="hg19", cex=0.8, label.cytoband=FALSE, cytoband.ycoords=c(ybottom[i], ytop[
}
}
axis(1, at=pretty(c(0, max(sl)), n=10), labels=pretty(c(0, max(sl)), n=10)/1e6, cex.axis=0.8)
mtext("position (Mb)", 1, line=2)
par(las=1)
axis(2, at=ybottom+0.01, names(sl), cex.axis=0.6)
```

---

xypanel *A panel function for plotting copy number versus physical position*

Description

A panel function for `xyplot` for plotting copy number versus physical position.

Usage

```
xypanel(x, y, gt, is.snp, range, col.hom = "grey20", fill.hom =
"lightblue", col.het = "grey20", fill.het = "salmon", col.np = "grey20",
fill.np = "grey60", show.state=TRUE, state.cex=1,  col.state="blue", ..., subscripts)
```

Arguments

`x`	Physical position in megabases.
`y`	Copy number estimates.
`gt`	Genotype calls.
`is.snp`	Logical. Whether the marker is polymorphic.
`range`	A `RangedData` or `IRanges` object. Note that we expect the units returned by `start` and `end` to be basepairs.

col.hom	A specification for the color of plotting symbols for homozygous genotypes.
fill.hom	A specification for the fill color of plotting symbols for homozygous genotypes.
col.het	A specification for the color of plotting symbols for heterozygous genotypes.
fill.het	A specification for the fill color of plotting symbols for heterozygous genotypes.
col.np	A specification for the color of plotting symbols for nonpolymorphic markers.
fill.np	A specification for the fill color of plotting symbols for nonpolymorphic genotypes.
show.state	Logical. Whether to display the predicted state in each panel.
state.cex	Numeric. cex for state label. Ignored if show.state is FALSE.
col.state	Character. color for state label. Ignored if show.state is FALSE.
...	Additional arguments passed to lattice functions xyplot, lpoints, and lrect.
subscripts	See the panel functions in lattice for more information.

Details

The order of plotting is (1) nonpolymorphic markers, (2), homozygous SNPs, and (3) heterozygous SNPs. Stretches of homozygosity should appear as blue using the default color scheme.

Note

To make the drawing of the range object border invisible, one can use border="white".

Author(s)

R. Scharpf

See Also

xyplot

Examples

```
## Not run:
if(require("crlmm") && require("VanillaICE") && require("IRanges")){
    library(lattice)
library(oligoClasses)
    data(oligoSetExample, package="oligoClasses")
    oligoSet <- oligoSet[chromosome(oligoSet) == 1, ]
    cn <- copyNumber(oligoSet)/100
    cn <- log2((2^cn)/2)
    gt <- calls(oligoSet)[,]
    ## simulate BAFs
    bf <- rep(NA, length(gt))
    u <- runif(length(gt))
    bf[gt==1 & u > 0.5] <- runif(sum(gt==1 & u > 0.5), 0, 0.05)
    bf[gt==1 & u <= 0.5] <- runif(sum(gt==1 & u <= 0.5), 0.95, 1)
    bf[gt==2] <- runif(sum(gt==2), 0.45, 0.55)
    bf[900:1200] <- runif(length(900:1200), 0, 0.03)
    gr <- GRanges(paste0("chr", chromosome(featureData(oligoSet))),
                  IRanges(position(oligoSet), width=1))
    cn <- as.matrix(cn)
    bf <- as.matrix(bf)
```

```
    dimnames(cn) <- dimnames(bf) <- list(featureNames(oligoSet), sampleNames(oligoSet))
    se <- SnpArrayExperiment(cn=cn,
                             baf=bf,
                             rowRanges=gr,
                             isSnp=rep(TRUE, length(gr)))
    fit <- hmm2(se)
    g <- as(segs(fit), "GRanges")
xyplot(cn ~ x | range, data=oligoSet, range=g[2], ##range=fit2[1:10, ],
       frame=2e6,
       panel=xypanel, cex=0.3, pch=21, border="blue",
       scales=list(x="free"),
       col.hom="lightblue", col.het="salmon", col.np="grey60",
       fill.np="grey60",
       xlab="Mb")
## if xyplot method is masked by lattice, do
##xyplot <- VanillaICE:::xyplot
}

## End(Not run)
```

`xypanelBaf` *Panel function for plotting copy number and B allele frequencies for a genomic interval.*

Description

Panel function for plotting copy number and B allele frequencies for a genomic interval.

Usage

```
xypanelBaf(x, y, gt, baf, is.snp, range, col.hom = "grey20", fill.hom = "lightblue", col.het = "gre
```

Arguments

`x`	physical position in basepairs
`y`	total copy number (relative or absolute)
`gt`	Genotypes coded as integers (1=AA, 2=AB, 3=BB). This is optional. If provided one can color code the plotting symbols by the genotype.
`baf`	B allele frequencies.
`is.snp`	Logical. Indicator of whether the marker hybridized to a known SNP or a non-polymorphic region of the genome.
`range`	A `RangedDataCNV`-derived object indicating the genomic interval to plot.
`col.hom`	Color to use for homozygous genotypes.
`fill.hom`	Fill color to use for homozygous genotypes.
`col.het`	Color to use for heterozygous genotypes.
`fill.het`	Fill color to use for heterozygous genotypes.
`col.np`	Color to use for nonpolymorphic markers
`fill.np`	Fill color for nonpolymorphic markers.

show.state	Logical indicating whether to display the copy number state for a RangedDataHMM object.
state.cex	Size of the font for displaying the HMM state. Ignored if show.state is FALSE.
col.state	Color for displaying the state.
...	Additional arguments passed to panel.xyplot.
subscripts	See panel.xyplot

Details

Function for plotting B allele frequencing and copy number on a trellis display. Intended to be passed to the panel argument of the function xyplotLrrBaf and should not be called directly by the user.

Author(s)

R.Scharpf

See Also

xyplotLrrBaf

---

xyplot *Plot copy number and physical position for a set of genomic intervals.*

Description

Plot copy number and physical position given by a CNSet object for a set of genomic intervals stored in a RangedDataCVN object.

Usage

```
xyplot2(x, data, range, frame=50e3L, ...)
```

Arguments

x	A formula. Currently, the formula must be one of cn~x, cn ~ x | id or cn ~ x | range when data is a CNSet. If data is a BeadStudioSet, the formula has the form lrr ~ x| range or baf ~ x | range.
data	A CNSet, BeadStudioSet, or SnpSet object.
...	A RangedDataCNV object must be passed by the name 'range'. Arguments for xyplot are passed to xyplot2. Additional arguments are passed to xypanel and panel.xyplot.
range	A RangedDataCNV object.
frame	The genomic distance (basepairs) to the left and right of the start and stop coordinates in the range object.

Details

These functions plot copy number estimates versus physical position. The function is particularly useful for multi-panel displays in which the copy number estimates for a single range of a GRanges object appears in one panel. The size of the multi-panel display depends on the number of ranges (rows) in the GRanges object.

Value

An object of class trellis.

Author(s)

R. Scharpf

See Also

xyplot, xypanel

To modify the plot appearance from the default, additional arguments can be passed to panel.xyplot, lpoints, and lrect.

Examples

```
## simulated data
library(oligoClasses)
library(IRanges)
library(VanillaICE)
data(oligoSetExample, package="oligoClasses")
## The oligoSnpSet class will likely be deprecated and made defunct
## in a future release.  Instead, we favor
## RangedSummarizedExperiment-derived classes defined in VanillaICE
oligoSet <- oligoSet[chromosome(oligoSet) == 1, ]
cn <- copyNumber(oligoSet)/100
cn <- log2((2^cn)/2)
gt <- calls(oligoSet)[,]
## simulate BAFs
bf <- rep(NA, length(gt))
u <- runif(length(gt))
bf[gt==1 & u > 0.5] <- runif(sum(gt==1 & u > 0.5), 0, 0.05)
bf[gt==1 & u <= 0.5] <- runif(sum(gt==1 & u <= 0.5), 0.95, 1)
bf[gt==2] <- runif(sum(gt==2), 0.45, 0.55)
bf[900:1200] <- runif(length(900:1200), 0, 0.03)
gr <- GRanges(paste0("chr", chromosome(featureData(oligoSet))),
              IRanges(position(oligoSet), width=1))
cn <- as.matrix(cn)
bf <- as.matrix(bf)
dimnames(cn) <- dimnames(bf) <- list(featureNames(oligoSet), sampleNames(oligoSet))
se <- SnpArrayExperiment(cn=cn,
                         baf=bf,
                         rowRanges=gr,
                         isSnp=rep(TRUE, length(gr)))
fit <- hmm2(se)
g <- as(segs(fit), "GRanges")
## To visualize each range in it's own panel surrounded by a
## frame of 2,000,000 bases:
## (here the frames are overlapping, but the method could be
```

```
## applied more generally to a collection of ranges from
## different chromsomes and samples)
xyplot2(cn~x | range, data=oligoSet,
     range=g,
     frame=2e6, panel=xypanel,
     cex=2,
     pch=".",
     col.het="salmon",
     fill.het="salmon",
     col.hom="royalblue",
     fill.hom="royalblue",
     state.cex=0.5,
     border="orange", scales=list(x="free"),
     par.strip.text=list(cex=0.5),
     xlab="Mb", ylab=expression(log[2]("copy number")))
```

`xyplotLrrBaf` *xyplot lattice function for RangedData and oligoSnpSet objects*

Description

For each genomic interval in the ranged data, a plot of the log R ratios and B allele frequencies stored in the oligoSnpSet are plotted.

Usage

```
xyplotLrrBaf(rd, object, frame, ...)
```

Arguments

`rd`	An instance of `RangedDataCNV` or `GRanges`.
`object`	A `oligoSnpSet` or `BeadStudioSet` object with assayData elements for log R ratios and B allele frequencies.
`frame`	The genomic distance in basepairs to plot on either side of the genomic interval in the `rd` object.
`...`	Additional arguments passed to the panel function. See details.

Details

The `xypanelBaf` function is a panel function that does the actual plotting of the genomic data.

Value

A `trellis` object.

Author(s)

R. Scharpf

See Also

`xypanelBaf`

Examples

```
## Not run:
  library(crlmm)
  library(SummarizedExperiment)
  library(VanillaICE)
  data(cnSetExample, package="crlmm")
  oligoSetList <- BafLrrSetList(cnSetExample[, 1])
  oligoSet <- oligoSetList[[1]]
  cn <- copyNumber(oligoSet)/100
  cn <- log2((2^cn)/2)
  gt <- calls(oligoSet)[,]
  ## simulate BAFs
  bf <- baf(oligoSet)[, ]/1000
  gr <- GRanges(paste0("chr", chromosome(featureData(oligoSet))),
                IRanges(position(oligoSet), width=1))
  cn <- as.matrix(cn)
  bf <- as.matrix(bf)
  dimnames(cn) <- dimnames(bf) <- list(featureNames(oligoSet), sampleNames(oligoSet))
  se <- SnpArrayExperiment(cn=cn,
                           baf=bf,
                           rowRanges=gr,
                           isSnp=rep(TRUE, length(gr)))
  fit <- hmm2(se)

  ##rd <- fit[sampleNames(fit)=="NA19007", ]
  rd <- as(segs(fit), "GRanges")
  ## We're interested in this range
  range <- GRanges("chr8", IRanges(3.7e6, 5.9e6), sample="NA19007")
  index <- subjectHits(findOverlaps(range, rd))
  xyplotLrrBaf(rd[index, ], oligoSetList[[1]], frame=1e6,
         panel=xypanelBaf, cex=0.2,
         scales=list(x=list(relation="free"),
         y=list(alternating=1,
         at=c(-1, 0, log2(3/2), log2(4/2)),
         labels=expression(-1, 0, log[2](3/2), log[2](4/2)))),
         par.strip.text=list(cex=0.7),
         ylim=c(-3,1),
         col.hom="grey50",
         col.het="grey50",
         col.np="grey20",
         xlab="physical position (Mb)",
         ylab=expression(log[2]("R ratios")),
       key=list(text=list(c(expression(log[2]("R ratios")), expression("B allele freqencies")),
            col=c("grey", "blue")), columns=2))

  ## Or, plot each range of the GRanges instance in a separate panel
  xyplotLrrBaf(rd, oligoSetList[[1]], frame=1e6,
         panel=xypanelBaf, cex=0.2,
         scales=list(x=list(relation="free"),
         y=list(alternating=1,
         at=c(-1, 0, log2(3/2), log2(4/2)),
         labels=expression(-1, 0, log[2](3/2), log[2](4/2)))),
         par.strip.text=list(cex=0.7),
         ylim=c(-3,1),
         col.hom="grey50",
         col.het="grey50",
```

```
            col.np="grey20",
            xlab="physical position (Mb)",
            ylab=expression(log[2]("R ratios")),
          key=list(text=list(c(expression(log[2]("R ratios")), expression("B allele freqencies")),
              col=c("grey", "blue")), columns=2))


## End(Not run)
```

Index

∗Topic **aplot**
plotIdiogram, 6
xypanelBaf, 9
∗Topic **color**
xypanel, 7
∗Topic **dplot**
latticeFigs, 5
xypanel, 7
xyplot, 10
∗Topic **hplot**
arrangeFigs, 2
arrangeSideBySide, 2
xyplotLrrBaf, 12
∗Topic **manip**
centromere, 3
dataFrame, 4
∗Topic **methods**
dataFrame, 4
dataFrame-methods, 4
xyplot, 10
∗Topic **misc**
getCytoband, 5

arrangeFigs, 2
arrangeSideBySide, *2*, 2

centromere, 3

dataFrame, 4
dataFrame,GRanges,gSet-method *(*dataFrame-methods*)*, 4
dataFrame,GRanges,RangedSummarizedExperiment-method *(*dataFrame-methods*)*, 4
dataFrame-methods, 4

getCytoband, 5

latticeFigs, 5
lpoints, *11*
lrect, *11*

panel.xyplot, *11*
plotCytoband2 *(*plotIdiogram*)*, 6
plotIdiogram, *5*, 6

xypanel, *3*, 7, *11*
xypanelBaf, 9, *12*
xyplot, *3*, *8*, 10, *11*
xyplot2 *(*xyplot*)*, 10
xyplot2,formula,gSet-method *(*xyplot*)*, 10
xyplot2,formula,SnpSet-method *(*xyplot*)*, 10
xyplotLrrBaf, *10*, 12