Package: BubbleTree (via r-universe)

June 15, 2024

Type Package

Title BubbleTree: an intuitive visualization to elucidate tumoral aneuploidy and clonality in somatic mosaicism using next generation sequencing data

Version 2.35.0

Date 2019-10-03

Author Wei Zhu <zhuw@medimmune.com>, Michael Kuziora <kuzioram@medimmune.com>, Todd Creasy <creasyt@medimmune.com>, Brandon Higgs <higgsb@medimmune.com>

Maintainer Todd Creasy <creasyt@medimmune.com>, Wei Zhu <weizhu365@gmail.com>

Description CNV analysis in groups of tumor samples.

License LGPL (>= 3)

Imports BiocGenerics (>= 0.31.6), BiocStyle, Biobase, ggplot2, WriteXLS, gtools, RColorBrewer, limma, grid, gtable, gridExtra, biovizBase, e1071, methods, grDevices, stats, utils

Depends R (>= 3.5), IRanges, GenomicRanges, plyr, dplyr, magrittr

Suggests knitr, rmarkdown

biocViews CopyNumberVariation, Software, Sequencing, Coverage

VignetteBuilder knitr

RoxygenNote 5.0.1

Repository https://bioc.r-universe.dev

RemoteUrl https://github.com/bioc/BubbleTree

RemoteRef HEAD

RemoteSha 80c07795ca847f6e6f249f4c7e659f37e3ab824b

Contents

`all.somatic.lst` *all.somatic.lst*

Description

A dataset containing pre-calculated BAF scores for annotated SNVs.

Format

S4 object with seqnames, genomic ranges, strand, BAF score

Source

internal

`allCall.lst` *allCall.lst*

Description

A dataset containing precalculated data from CNV segment analysis.

Format

S4 object with rbd, rbd.adj, results

Source

internal

`allCNV.lst` *allCNV.lst*

Description

A dataset containing pre-calculated segment calls.

Format

S4 object with seqnames, genomic ranges, num.mark, score

Source

internal

allHetero.lst *allHetero.lst*

Description

S4 GRanges dataset containing pre-calculated heterozygosity data.

Format

S4

Source

internal

allRBD.lst *allRBD.lst*

Description

A dataset containing precalculated data from CNV segment analysis.

Format

S4 object with rbd, rbd.adj

Source

internal

annoByGenesAndCyto *annoByGenesAndCyto*

Description

get annotation for genes and cytobands

Usage

```
annoByGenesAndCyto(.Object, chr, beg, end, critical.genes, gene.uni.clean.gr,
  cyto.gr)

## S4 method for signature 'Annotate'
annoByGenesAndCyto(.Object, chr, beg, end, critical.genes,
  gene.uni.clean.gr, cyto.gr)
```

Arguments

.Object	the objet
chr	the chromosome
beg	genomic start coord
end	genomic end coord
critical.genes	set of critical genes
gene.uni.clean.gr	gr object of genes
cyto.gr	gr object of cyto positions

Value

list of annotation for genes and cytobands

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "cancer.genes.minus2.rda", package="BubbleTree"))
load(system.file("data", "vol.genes.rda", package="BubbleTree"))
load(system.file("data", "gene.uni.clean.gr.rda", package="BubbleTree"))
load(system.file("data", "cyto.gr.rda", package="BubbleTree"))

comm <- btcompare(vol.genes, cancer.genes.minus2)
btreeplotter <- new("BTreePlotter", branch.col="gray50")
annotator <-new("Annotate")
nn <- "sam2"
cc <- allCall.lst[[nn]]
z <- drawBTree(btreeplotter, cc@rbd.adj) +
    ggplot2::labs(title=sprintf("%s (%s)", nn, info(cc)))
out <- cc@result$dist  %>%
    filter(seg.size >= 0.1 ) %>%
    arrange(gtools::mixedorder(as.character(seqnames)), start)

ann <- annoByGenesAndCyto(annotator,
                  as.character(out$seqnames),
                  as.numeric(out$start),
                  as.numeric(out$end),
                  comm$comm,
                  gene.uni.clean.gr=gene.uni.clean.gr,
                  cyto.gr=cyto.gr)
```

Annotate *Annotate*

Description

Annotate

Examples

```
annotate <- new("Annotate")
```

---

bafTrack *bafTrack*

Description

get the BAF track

Usage

```
bafTrack(.Object, result.dat, gr2, somatic.gr = NULL, min.prev = 0.15,
  cex = 1.2)

## S4 method for signature 'TrackPlotter'
bafTrack(.Object, result.dat, gr2, somatic.gr = NULL,
  min.prev = 0.15, cex = 1.2)
```

Arguments

.Object	the object
result.dat	the result dataframe
gr2	the gr2 object
somatic.gr	somatic gr object annotation
min.prev	previous min
cex	the cex

Value

the highlighted BAF track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "centromere.dat.rda", package="BubbleTree"))
load(system.file("data", "all.somatic.lst.RData", package="BubbleTree"))
load(system.file("data", "hg19.seqinfo.rda", package="BubbleTree"))

trackplotter <- new("TrackPlotter")
gr2 = centromere.dat
nn <- "sam2"
p2 <- bafTrack(trackplotter,
               result.dat=allCall.lst[[nn]]@result,
               gr2=gr2,
               somatic.gr=all.somatic.lst[[nn]])
```

btcompare *btcompare*

Description

btcompare

Usage

```
btcompare(set1, set2)
```

Arguments

set1	first set
set2	second set to compare

Value

combined, unique list of genes

Examples

```
load(system.file("data", "cancer.genes.minus2.rda", package="BubbleTree"))
load(system.file("data", "vol.genes.rda", package="BubbleTree"))

# 77 common cancer genes
comm <- btcompare(vol.genes, cancer.genes.minus2)
```

btpredict *btpredict*

Description

btpredict

Usage

```
btpredict(.Object)

## S4 method for signature 'BTreePredictor'
btpredict(.Object)
```

Arguments

.Object	the object

Value

.Object populated with the predictions

Examples

```
load(system.file("data", "allRBD.lst.RData", package="BubbleTree"))

btreepredictor <- new("BTreePredictor")
btreepredictor@config$cutree.h <- 0.15
high.ploidy <- rep(TRUE, length(allRBD.lst))
high.purity <- rep(TRUE, length(allRBD.lst))

high.ploidy[c("sam6",
              "ovary.wgs",
              "ovary.wes",
              "TCGA-06-0145-01A-01W-0224-08",
              "TCGA-13-1500-01A-01D-0472-01",
              "TCGA-AO-A0JJ-01A-11W-A071-09")] <- FALSE

high.purity[c("sam6", "ovary.wgs", "ovary.wes")] <- FALSE

rbd <- allRBD.lst[["sam6"]]
btreepredictor@config$high.ploidy <- high.ploidy["sam6"]
btreepredictor@config$high.purity <- high.purity["sam6"]
btreepredictor <- loadRBD(btreepredictor, rbd)
btreepredictor@config$min.segSize <- ifelse(max(btreepredictor@rbd$seg.size,
                                                na.rm=TRUE) < 0.4, 0.1, 0.4)
btreepredictor <- btpredict(btreepredictor)
cat(info(btreepredictor), "\n")
```

BTreePlotter *BTreePlotter*

Description

BTreePlotter

Examples

```
btreeplotter <- new("BTreePlotter")
```

BTreePredictor *BTreePredictor*

Description

BTreePredictor

Examples

```
btreepredictor <- new("BTreePredictor")
```

cancer.genes.minus2 *cancer.genes.minus2.rda*

Description

A dataset containing a list of known cancer genes.

Format

list

Source

internal

centromere.dat *centromere.dat*

Description

A dataset containing an annotated list of centromere locations.

Format

list

Source

internal

cnv.gr *cnv.gr*

Description

S4 GRanges object containing data on chromosomal locations with seqnames, genomic range, strand, name

Format

S4

Source

internal

cyto.gr *cyto.gr*

Description

S4 GRanges object containing data on chromosomal locations with seqnames, genomic range, strand, name, gieStain.

Format

S4

Source

internal

drawBTree *drawBTree*

Description

draw the BTree track

Usage

```
drawBTree(.Object, rbd, size = 1)

## S4 method for signature 'BTreePlotter'
drawBTree(.Object, rbd, size = 1)
```

Arguments

.Object	the object
rbd	the rbd object
size	the size

Value

draw the BTree track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "cancer.genes.minus2.rda", package="BubbleTree"))
load(system.file("data", "vol.genes.rda", package="BubbleTree"))
load(system.file("data", "gene.uni.clean.gr.rda", package="BubbleTree"))
load(system.file("data", "cyto.gr.rda", package="BubbleTree"))

# 77 common cancer genes
comm <- btcompare(vol.genes, cancer.genes.minus2)

btreeplotter <- new("BTreePlotter", branch.col="gray50")
annotator <-new("Annotate")
cc <- allCall.lst[["sam2"]]
z <- drawBTree(btreeplotter, cc@rbd.adj) +
    ggplot2::labs(title=sprintf("%s (%s)", "sam2", info(cc)))
```

---

drawBubbles *drawBubbles*

Description

draw the Bubbles

Usage

```
drawBubbles(.Object, rbd, col = NULL)

## S4 method for signature 'BTreePlotter'
drawBubbles(.Object, rbd, col = "gray80")
```

Arguments

.Object	the object
rbd	the rbd object
col	the col value

Value

draw the bubbles on the track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))

btreeplotter <- new("BTreePlotter", max.ploidy=5, max.size=10)
nn <- "sam2"
rbd1 <- allCall.lst[[nn]]@rbd
rbd2 <- allCall.lst[[nn]]@rbd.adj
arrows <- trackBTree(btreeplotter, rbd1, rbd2, min.srcSize=0.01,
                     min.trtSize=0.01)
btree <- drawBTree(btreeplotter, rbd1) +
    drawBubbles(btreeplotter, rbd2, "gray80") + arrows
```

---

drawFeatures *drawFeatures*

Description

draw the features

Usage

```
drawFeatures(.Object, rbd, col = NULL)

## S4 method for signature 'BTreePlotter'
drawFeatures(.Object, rbd, col = "black")
```

Arguments

.Object	the object
rbd	the rbd object
col	the col value

Value

draw the annotation on the track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "cancer.genes.minus2.rda", package="BubbleTree"))
load(system.file("data", "vol.genes.rda", package="BubbleTree"))
load(system.file("data", "gene.uni.clean.gr.rda", package="BubbleTree"))
load(system.file("data", "cyto.gr.rda", package="BubbleTree"))
```

```
# 77 common cancer genes merged from 2 sets
comm <- btcompare(vol.genes, cancer.genes.minus2)

btreeplotter <- new("BTreePlotter", branch.col="gray50")
annotator <- new("Annotate")

nn <- "sam12"
cc <- allCall.lst[[nn]]
z <- drawBTree(btreeplotter, cc@rbd.adj) +
    ggplot2::labs(title=sprintf("%s (%s)", nn, info(cc)))
out <- cc@result$dist  %>% filter(seg.size >= 0.1 ) %>%
    arrange(gtools::mixedorder(as.character(seqnames)), start)

ann <- with(out, {
    annoByGenesAndCyto(annotator,
                       as.character(out$seqnames),
                       as.numeric(out$start),
                       as.numeric(out$end),
                       comm$comm,
                       gene.uni.clean.gr=gene.uni.clean.gr,
                       cyto.gr=cyto.gr)
})

out$cyto <- ann$cyto
out$genes <- ann$ann
v <- z + drawFeatures(btreeplotter, out)
print(v)
```

---

gene.uni.clean.gr *gene.uni.clean.gr*

Description

S4 GRanges object containing human gene annotation with seqnames, genomic coordinates, stand, gene.sybmol.

Format

S4

Source

internal

getTracks *getTracks*

Description

get all tracks

Usage

```
getTracks(p1, p2, title = "")
```

Arguments

p1	set 1
p2	set 2
title	the title

Value

all of the requested tracks

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "centromere.dat.rda", package="BubbleTree"))
load(system.file("data", "all.somatic.lst.RData", package="BubbleTree"))
load(system.file("data", "hg19.seqinfo.rda", package="BubbleTree"))

trackplotter <- new("TrackPlotter")
gr2 = centromere.dat
nn <- "sam2"
ymax <- ifelse(nn %in% c("lung.wgs", "lung.wes"), 9, 4.3)
p1 <- xyTrack(trackplotter,
              result.dat=allCall.lst[[nn]]@result,
              gr2=gr2,
              ymax=ymax) + ggplot2::labs(title=nn)

p2 <- bafTrack(trackplotter,
               result.dat=allCall.lst[[nn]]@result,
               gr2=gr2,
               somatic.gr=all.somatic.lst[[nn]])

t1 <- getTracks(p1, p2)
```

heteroLociTrack *heteroLociTrack*

Description

get the heteroLoci track

Usage

```
heteroLociTrack(.Object, result.dat, gr2, hetero.gr = NULL, min.prev = 0.15,
  ymax = 4.3, cex = 0.5)

## S4 method for signature 'TrackPlotter'
heteroLociTrack(.Object, result.dat, gr2,
  hetero.gr = NULL, min.prev = 0.15, ymax = 4.3, cex = 0.5)
```

Arguments

.Object	the object
result.dat	the results
gr2	the gr2 object
hetero.gr	hetero annotation
min.prev	previous min
ymax	max y
cex	the cex

Value

the highlightted heterozygosity track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "centromere.dat.rda", package="BubbleTree"))
load(system.file("data", "allHetero.lst.RData", package="BubbleTree"))
load(system.file("data", "hg19.seqinfo.rda", package="BubbleTree"))


trackplotter <- new("TrackPlotter")
gr2 = centromere.dat
nn <- "sam2"
z1 <- heteroLociTrack(trackplotter, allCall.lst[[nn]]@result,
                      gr2, allHetero.lst[[nn]])
```

hg19.seqinfo *hg19.seqinfo.Rd*

Description

Seqinfo object containing names and lengths of each chromosome of the human genome.

Format

Seqinfo

Source

internal

info *info*

Description

info

Usage

```
info(.Object)

## S4 method for signature 'BTreePredictor'
info(.Object)
```

Arguments

.Object	the object

Value

print out info of prediction data

Examples

```
load(system.file("data", "allRBD.lst.RData", package="BubbleTree"))

btreepredictor <- new("BTreePredictor")
btreepredictor@config$cutree.h <- 0.15


high.ploidy <- rep(TRUE, length(allRBD.lst))
high.purity <- rep(TRUE, length(allRBD.lst))
```

```
high.ploidy[c("sam6",
              "ovary.wgs",
              "ovary.wes",
              "TCGA-06-0145-01A-01W-0224-08",
              "TCGA-13-1500-01A-01D-0472-01",
              "TCGA-AO-A0JJ-01A-11W-A071-09")] <- FALSE

high.purity[c("sam6", "ovary.wgs", "ovary.wes")] <- FALSE

nn <- "sam6"

rbd <- allRBD.lst[[nn]]
btreepredictor@config$high.ploidy <- high.ploidy[nn]
btreepredictor@config$high.purity <- high.purity[nn]
btreepredictor <- loadRBD(btreepredictor, rbd)
btreepredictor@config$min.segSize <- ifelse(max(btreepredictor@rbd$seg.size,
                                                na.rm=TRUE) < 0.4, 0.1, 0.4)
btreepredictor <- btpredict(btreepredictor)
cat(info(btreepredictor), "\n")
```

---

loadRBD *loadRBD*

Description

load the RBD data

Usage

```
loadRBD(.Object, rbd, total.mark = NA)

## S4 method for signature 'BTreePredictor'
loadRBD(.Object, rbd, total.mark = NA)
```

Arguments

.Object	the object
rbd	rbd object
total.mark	total mark

Value

.Object populated with the RBD list with updated segment size

Examples

```
load(system.file("data", "allRBD.lst.RData", package="BubbleTree"))

btreepredictor <- new("BTreePredictor")
btreepredictor@config$cutree.h <- 0.15

high.ploidy <- rep(TRUE, length(allRBD.lst))
high.purity <- rep(TRUE, length(allRBD.lst))

high.ploidy[c("sam6",
              "ovary.wgs",
              "ovary.wes",
              "TCGA-06-0145-01A-01W-0224-08",
              "TCGA-13-1500-01A-01D-0472-01",
              "TCGA-AO-A0JJ-01A-11W-A071-09")] <- FALSE

high.purity[c("sam6", "ovary.wgs", "ovary.wes")] <- FALSE

nn <- "sam6"

rbd <- allRBD.lst[[nn]]
btreepredictor@config$high.ploidy <- high.ploidy[nn]
btreepredictor@config$high.purity <- high.purity[nn]
btreepredictor <- loadRBD(btreepredictor, rbd)
```

---

makeRBD *makeRBD*

Description

make the RBD object

Usage

```
makeRBD(.Object, ...)

## S4 method for signature 'RBD'
makeRBD(.Object, snp.gr, cnv.gr, unimodal.kurtosis = -0.1)
```

Arguments

.Object	the object
...	other input (not needed)
snp.gr	SNP GenomicRanges object
cnv.gr	CNV GenomicRanges object
unimodal.kurtosis	kurtosis

Value

RBD object

Examples

```
# load sample files
load(system.file("data", "cnv.gr.rda", package="BubbleTree"))
load(system.file("data", "snp.gr.rda", package="BubbleTree"))

# load annotations
load(system.file("data", "centromere.dat.rda", package="BubbleTree"))
load(system.file("data", "cyto.gr.rda", package="BubbleTree"))
load(system.file("data", "cancer.genes.minus2.rda", package="BubbleTree"))
load(system.file("data", "vol.genes.rda", package="BubbleTree"))
load(system.file("data", "gene.uni.clean.gr.rda", package="BubbleTree"))


# initialize RBD object
r <- new("RBD", unimodal.kurtosis=-0.1)

# create new RBD object with GenomicRanges objects for SNPs and CNVs
rbd <- makeRBD(r, snp.gr, cnv.gr)
head(rbd)

# create a new prediction
btreepredictor <- new("BTreePredictor", rbd=rbd, max.ploidy=6, prev.grid=seq(0.2,1, by=0.01))
pred <- btpredict(btreepredictor)

# create rbd plot
btreeplotter <- new("BTreePlotter", max.ploidy=5, max.size=10)
btree <- drawBTree(btreeplotter, pred@rbd)
print(btree)

# create rbd.adj plot
btreeplotter <- new("BTreePlotter", branch.col="gray50")
btree <- drawBTree(btreeplotter, pred@rbd.adj)
print(btree)

# create a combined plot with rbd and rbd.adj that shows the arrows indicating change
# THIS IS VERY MESSY WITH CURRENT DATA from Dong
btreeplotter <- new("BTreePlotter", max.ploidy=5, max.size=10)
arrows <- trackBTree(btreeplotter,
                     pred@rbd,
                     pred@rbd.adj,
                     min.srcSize=0.01,
                     min.trtSize=0.01)

btree <- drawBTree(btreeplotter, pred@rbd) + arrows
print(btree)


# create a plot with overlays of significant genes
```

```
btreeplotter <- new("BTreePlotter", branch.col="gray50")
annotator <- new("Annotate")

comm <- btcompare(vol.genes, cancer.genes.minus2)

sample.name <- "22_cnv_snv"

btree <- drawBTree(btreeplotter, pred@rbd.adj) +
    ggplot2::labs(title=sprintf("%s (%s)", sample.name, info(pred)))

out <- pred@result$dist  %>%
    filter(seg.size >= 0.1 ) %>%
    arrange(gtools::mixedorder(as.character(seqnames)), start)

ann <- with(out, {
    annoByGenesAndCyto(annotator,
                       as.character(out$seqnames),
                       as.numeric(out$start),
                       as.numeric(out$end),
                       comm$comm,
                       gene.uni.clean.gr=gene.uni.clean.gr,
                       cyto.gr=cyto.gr)
})

out$cyto <- ann$cyto
out$genes <- ann$ann

btree <- btree + drawFeatures(btreeplotter, out)
print(btree)


# print out purity and ploidy values
info <- info(pred)
cat("\nPurity/Ploidy: ", info, "\n")
```

---

mergeSnpCnv *mergeSnpCnv*

Description

merge snp and cnv data

Usage

```
mergeSnpCnv(.Object, snp.gr, cnv.gr)

## S4 method for signature 'RBD'
mergeSnpCnv(.Object, snp.gr, cnv.gr)
```

Arguments

.Object	the object
snp.gr	SNP GenomicRanges object
cnv.gr	CNV GenomicRanges object

Value

combined, unique list of genes

RBD *RBD*

Description

RBD

Examples

```
rbd <- new("RBD")
```

RscoreTrack *RscoreTrack*

Description

get the RScore track

Usage

```
RscoreTrack(.Object, result.dat, gr2, cnv.gr = NULL, min.prev = 0.15,
  ymax = 3, cex = 1.5)

## S4 method for signature 'TrackPlotter'
RscoreTrack(.Object, result.dat, gr2, cnv.gr = NULL,
  min.prev = 0.15, ymax = 3, cex = 1.5)
```

Arguments

.Object	the object
result.dat	the results
gr2	the gr2 object
cnv.gr	cnv annotation
min.prev	previous min
ymax	max y
cex	the cex

Value

the highlighted RScore track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "centromere.dat.rda", package="BubbleTree"))
load(system.file("data", "allCNV.lst.RData", package="BubbleTree"))
load(system.file("data", "hg19.seqinfo.rda", package="BubbleTree"))

gr2 = centromere.dat
trackplotter <- new("TrackPlotter")
nn <- "sam2"
z <- RscoreTrack(trackplotter, allCall.lst[[nn]]@result, gr2, allCNV.lst[[nn]])
```

---

saveXLS *saveXLS*

Description

saveXLS

Usage

```
saveXLS(dat.lst, xls.fn, row.names = FALSE, ...)
```

Arguments

dat.lst	dataframe
xls.fn	filename
row.names	row names
...	misc

Value

new Excel file

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))

all.summary <- plyr::ldply(allCall.lst, function(.Object) {
    purity <- .Object@result$prev[1]
    adj <- .Object@result$ploidy.adj["adj"]
    # when purity is low the calculation result is not reliable
    ploidy <- (2*adj -2)/purity + 2
```

```
    with(.Object@result,
         return(c(Purity=round(purity,3),
                  Prevalences=paste(round(prev,3), collapse=", "),
                  "Tumor ploidy"=round(ploidy,1))))
}) %>% plyr::rename(c(".id"="Sample"))

xls.filename <- paste("all_summary", "xlsx", sep=".")
saveXLS(list(Summary=all.summary), xls.filename)
```

---

snp.gr *snp.gr*

Description

S4 GRanges object containing data on chromosomal locations with seqnames, genomic position, strand, name

Format

S4

Source

internal

---

trackBTree *trackBTree*

Description

get the geom_segment location of the BTree track

Usage

```
trackBTree(.Object, rbd1, rbd2, is.matched = FALSE, min.srcSize = 0.5,
  min.trtSize = 0.1, min.overlap = 1e+05)

## S4 method for signature 'BTreePlotter'
trackBTree(.Object, rbd1, rbd2, is.matched = FALSE,
  min.srcSize = 0.5, min.trtSize = 0.1, min.overlap = 1e+05)
```

Arguments

.Object	the object
rbd1	rbd one
rbd2	rbd two
is.matched	is it matched
min.srcSize	min src size
min.trtSize	min trt size
min.overlap	min overlap

Value

geom_segment location of BTree track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))

btreeplotter <- new("BTreePlotter", max.ploidy=5, max.size=10)
nn <- "sam2"
rbd1 <- allCall.lst[[nn]]@rbd
rbd2 <- allCall.lst[[nn]]@rbd.adj
arrows <- trackBTree(btreeplotter, rbd1, rbd2, min.srcSize=0.01,
                     min.trtSize=0.01)
btree <- drawBTree(btreeplotter, rbd1) +
    drawBubbles(btreeplotter, rbd2, "gray80") + arrows
```

TrackPlotter *TrackPlotter*

Description

TrackPlotter

Examples

```
trackplotter <- new("TrackPlotter")
```

vol.genes *vol.genes*

Description

A dataset containing a list of known cancer genes.

Format

list

Source

internal

xyTrack *xyTrack*

Description

get the xy track

Usage

```
xyTrack(.Object, result.dat, gr2, min.prev = 0.15, ymax = 4.3)

## S4 method for signature 'TrackPlotter'
xyTrack(.Object, result.dat, gr2, min.prev = 0.15,
  ymax = 4.3)
```

Arguments

.Object	the object
result.dat	result dataframe
gr2	gr2 object
min.prev	previous min
ymax	the max y

Value

the highlighted xy track

Examples

```
load(system.file("data", "allCall.lst.RData", package="BubbleTree"))
load(system.file("data", "centromere.dat.rda", package="BubbleTree"))
load(system.file("data", "hg19.seqinfo.rda", package="BubbleTree"))

trackplotter <- new("TrackPlotter")
gr2 = centromere.dat
nn <- "sam2"
ymax <- ifelse(nn %in% c("lung.wgs", "lung.wes"), 9, 4.3)
p1 <- xyTrack(trackplotter,
              result.dat=allCall.lst[[nn]]@result,
              gr2=gr2,
              ymax=ymax) + ggplot2::labs(title=nn)
```

Index