Package 'derfinderPlot'

April 9, 2015

Type Package

Title Plotting functions for derfinder

Version 1.0.3

Date 2014-11-04

Depends R(>= 3.1.1)

Imports derfinder (>= 1.0.0), GenomeInfoDb (>= 1.2.2), GenomicFeatures, GenomicRanges (>= 1.17.40), ggbio (>= 1.13.13), ggplot2, IRanges (>= 1.99.28), plyr, RColorBrewer, reshape2, scales

Suggests biovizBase, bumphunter, derfinderData (>= 0.99.0), devtools (>= 1.6), knitcitations (>= 1.0.1), knitr (>= 1.6), knitrBootstrap (>= 0.9.0), rmarkdown (>= 0.3.3), testthat, TxDb.Hsapiens.UCSC.hg19.knownGene

VignetteBuilder knitr

Description Plotting functions for derfinder

License Artistic-2.0

LazyData false

URL https://github.com/lcolladotor/derfinderPlot

BugReports https://github.com/lcolladotor/derfinderPlot/issues

biocViews DifferentialExpression, Sequencing, RNASeq, Software, Visualization

Author Leonardo Collado-Torres [aut, cre], Andrew E. Jaffe [aut], Jeffrey T. Leek [aut, ths]

Maintainer Leonardo Collado-Torres <lcollado@jhu.edu>

R topics documented:

`derfinderPlot-package` *Plotting functions for derfinder*

Description

Plotting functions for derfinder results

Author(s)

Leonardo Collado-Torres <lcollado@jhu.edu>

`plotCluster` *Plot the coverage information surrounding a region cluster*

Description

For a given region found in calculatePvalues, plot the coverage for the cluster this region belongs to as well as some padding. The mean by group is shown to facilitate comparisons between groups. If annotation exists, you can plot the trancripts and exons (if any) overlapping in the vicinity of the region of interest.

Usage

```
plotCluster(idx, regions, annotation, coverageInfo, groupInfo,
  titleUse = "qval", txdb = NULL, p.ideogram = NULL, ...)
```

Arguments

`idx`	A integer specifying the index number of the region of interest. This region is graphically highlighted by a red bar.
`regions`	The `$regions` output from calculatePvalues.
`annotation`	The output from running annotateNearest on the output from calculatePvalues.
`coverageInfo`	A DataFrame resulting from loadCoverage using `cutoff=NULL`.
`groupInfo`	A factor specifying the group membership of each sample. It will be used to color the samples by group.
`titleUse`	Whether to show the p-value (`pval`), the q-value (`qval`) or the FWER adjusted p-value (`fwer`) in the title. If `titleUse=none` then no p-value or q-value information is used; useful if no permutations were performed and thus p-value and q-value information is absent.
`txdb`	A transcript data base such as `TxDb.Hsapiens.UCSC.hg19.knownGene`. If `NULL` then no annotation information is used.
`p.ideogram`	If `NULL`, the ideogram for hg19 is built for the corresponding chromosome. Otherwise an ideogram resuling from plotIdeogram.
`...`	Arguments passed to other methods and/or advanced arguments.

Details

See the parameter `significantCut` in calculatePvalues for how the significance cutoffs are determined.

Value

A ggplot2 plot that is ready to be printed out. Tecnically it is a ggbio object. The region with the red bar is the one whose information is shown in the title.

Author(s)

Leonardo Collado-Torres

See Also

loadCoverage, calculatePvalues, annotateNearest, plotIdeogram

Examples

```
## Load data
library(derfinder)

## Annotate the results
suppressMessages(library(bumphunter))
annotation <- annotateNearest(genomeRegions$regions, hg19)

## Make the plot
suppressMessages(library(TxDb.Hsapiens.UCSC.hg19.knownGene))
plotCluster(idx=1, regions=genomeRegions$regions, annotation=annotation,
    coverageInfo=genomeDataRaw$coverage, groupInfo=genomeInfo$pop,
    txdb=TxDb.Hsapiens.UCSC.hg19.knownGene)
## Resize the plot window and the labels will look good.

## Not run:
## For a custom plot, check the ggbio and ggplot2 packages.
## Also feel free to look at the code for this function:
plotCluster


## End(Not run)
```

---

plotOverview *Plot a karyotype overview of the genome with the identified regions*

Description

Plots an overview of the genomic locations of the identified regions (see calculatePvalues) in a karyotype view. The coloring can be done either by significant regions according to their p-values, significant by adjusted p-values, or by annotated region if using annotateNearest.

Usage

```
plotOverview(regions, annotation = NULL, type = "pval",
  significantCut = c(0.05, 0.1), ...)
```

Arguments

`regions`	The `$regions` output from calculatePvalues.
`annotation`	The output from running annotateNearest on the output from calculatePvalues. It is only required if `type=annotation`.
`type`	Must be either `pval`, `qval`, `fwer` or `annotation`. It determines whether the plot coloring should be done according to significant p-values (<0.05), significant q-values (<0.10), significant FWER adjusted p-values (<0.05) or annotation regions.
`significantCut`	A vector of length two specifiying the cutoffs used to determine significance. The first element is used to determine significance for the p-values and the second element is used for the q-values.
`...`	Arguments passed to other methods and/or advanced arguments.

Value

A ggplot2 plot that is ready to be printed out. Tecnically it is a ggbio object.

Author(s)

Leonardo Collado-Torres

See Also

calculatePvalues, annotateNearest

Examples

```
## Construct toy data
chrs <- paste0(chr, c(1:22, X, Y))
chrs <- factor(chrs, levels=chrs)
library(GenomicRanges)
regs <- GRanges(rep(chrs, 10), ranges=IRanges(runif(240, 1, 4e7),
    width=1e3), significant=sample(c(TRUE, FALSE), 240, TRUE, p=c(0.05,
    0.95)), significantQval=sample(c(TRUE, FALSE), 240, TRUE, p=c(0.1,
    0.9)), area=rnorm(240))
annotation <- data.frame(region=sample(c(upstream, promoter,
    "overlaps 5", inside, "overlaps 3", "close to 3", downstream),
    240, TRUE))

## Type pval
plotOverview(regs)

## Not run:
## Type qval
```

```
plotOverview(regs, type=qval)

## Annotation
plotOverview(regs, annotation, type=annotation)

## Resize the plots if needed.

## You might prefer to leave the legend at ggplot2s default option: right
plotOverview(regs, legend.position=right)

## Although the legend looks better on the bottom
plotOverview(regs, legend.position=bottom)

## Example knitr chunk for higher res plot using the CairoPNG device
{r overview, message=FALSE, fig.width=7, fig.height=9, dev=CairoPNG, dpi=300}
plotOverview(regs, base_size=30, areaRel=10, legend.position=c(0.95, 0.12))


## For more custom plots, take a look at the ggplot2 and ggbio packages
## and feel free to look at the code of this function:
plotOverview

## End(Not run)
```

`plotRegionCoverage` *Makes plots for every region while summarizing the annotation*

Description

This function takes the regions found in calculatePvalues and assigns them genomic states contructed with makeGenomicState. The main workhorse functions are countOverlaps and findOverlaps. For an alternative plot check plotCluster which is much slower and we recommend it's use only after quickly checking the results with this function.

Usage

```
plotRegionCoverage(regions, regionCoverage, groupInfo, nearestAnnotation,
  annotatedRegions, txdb = NULL, whichRegions = seq_len(min(100,
  length(regions))), colors = NULL, scalefac = 32, ask = interactive(),
  ylab = "Coverage", verbose = TRUE)
```

Arguments

`regions`	The `$regions` output from calculatePvalues.
`regionCoverage`	The output from getRegionCoverage used on `regions`.
`groupInfo`	A factor specifying the group membership of each sample. It will be used to color the samples by group.
`nearestAnnotation`	The output from annotateNearest used on `regions`.

annotatedRegions	The output from annotateRegions used on regions.
txdb	A TxDb object. If specified, transcript annotation will be extracted from this object and used to plot the transcripts.
whichRegions	An integer vector with the index of the regions to plot.
colors	If NULL then brewer.pal with the Dark2 color scheme is used.
scalefac	The parameter used in preprocessCoverage.
ask	If TRUE then the user is prompted before each plot is made.
ylab	The name of the of the Y axis.
verbose	If TRUE basic status updates will be printed along the way.

Value

A plot for every region showing the coverage of each sample at each base of the region as well as the summarized annotation information.

Author(s)

Andrew Jaffe, Leonardo Collado-Torres

See Also

calculatePvalues, getRegionCoverage, annotateNearest, annotateRegions, plotCluster

Examples

```
## Load data
library(derfinder)

## Annotate regions, first two regions only
regions <- genomeRegions$regions[1:2]
annotatedRegions <- annotateRegions(regions = regions,
    genomicState = genomicState$fullGenome, minoverlap = 1)

## Find nearest annotation
library(bumphunter)
nearestAnnotation <- annotateNearest(regions, hg19)

## Obtain fullCov object
fullCov <- list(21=genomeDataRaw$coverage)

## Assign chr lengths using hg19 information
library(GenomicRanges)
data(hg19Ideogram, package = biovizBase, envir = environment())
seqlengths(regions) <- seqlengths(hg19Ideogram)[names(seqlengths(regions))]

## Get the region coverage
regionCov <- getRegionCoverage(fullCov=fullCov, regions=regions)
## Make plots for the regions
```

```
plotRegionCoverage(regions=regions, regionCoverage=regionCov,
    groupInfo=genomeInfo$pop, nearestAnnotation=nearestAnnotation,
    annotatedRegions=annotatedRegions, whichRegions=1:2)

## Re-make plots with transcript information
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
plotRegionCoverage(regions=regions, regionCoverage=regionCov,
    groupInfo=genomeInfo$pop, nearestAnnotation=nearestAnnotation,
    annotatedRegions=annotatedRegions, whichRegions=1:2, txdb = txdb)

## Not run:
## If you prefer, you can save the plots to a pdf file
pdf(ders.pdf, h = 6, w = 9)
plotRegionCoverage(regions=regions, regionCoverage=regionCov,
    groupInfo=genomeInfo$pop, nearestAnnotation=nearestAnnotation,
    annotatedRegions=annotatedRegions, whichRegions=1:2, txdb = txdb,
    ask=FALSE)
dev.off()

## End(Not run)
```

Index