Package ‘NADfinder’

May 30, 2024

Type  Package
Title  Call wide peaks for sequencing data
Version  1.28.0
Encoding  UTF-8
Author  Jianhong Ou, Haibo Liu, Jun Yu, Hervé Pagès, Paul Kaufman, Lihua Julie Zhu
Maintainer  Jianhong Ou <jianhong.ou@duke.edu>,
            Lihua Julie Zhu <julie.zhu@umassmed.edu>
Description  Nucleolus is an important structure inside the nucleus in eukaryotic cells. It is the site for transcribing rDNA into rRNA and for assembling ribosomes, aka ribosome biogenesis. In addition, nucleoli are dynamic hubs through which numerous proteins shuttle and contact specific non-rDNA genomic loci. Deep sequencing analyses of DNA associated with isolated nucleoli (NAD-seq) have shown that specific loci, termed nucleolus-associated domains (NADs) form frequent three-dimensional associations with nucleoli. NAD-seq has been used to study the biological functions of NAD and the dynamics of NAD distribution during embryonic stem cell (ESC) differentiation. Here, we developed a Bioconductor package NADfinder for bioinformatic analysis of the NAD-seq data, including baseline correction, smoothing, normalization, peak calling, and annotation.
License  GPL (>= 2)
Depends  R (>= 3.4), BiocGenerics, IRanges, GenomicRanges, S4Vectors, SummarizedExperiment
Imports  graphics, methods, baseline, signal, GenomicAlignments, GenomeInfoDb, rtracklayer, limma, trackViewer, stats, utils, Rsamtools, metap, EmpiricalBrownsMethod, ATACseqQC, corrplot, csaw
Suggests  RUnit, BiocStyle, knitr, BSgenome.Mmusculus.UCSC.mm10, testthat, BiocManager, rmarkdown
NADfinder-package

Identify nucleolus-associated domains (NADs) from NAD-seq

Description

Sliding-window based peak calling algorithm using whole genome sequences as control
backgroundCorrection

Correct ratios for background

Description

Correct ratios of read counts per sliding window for background.

Usage

backgroundCorrection(ratios, degree = 3, ...)

Arguments

ratios  A vector of numeric. It is log2-transformed ratios, CPMRatios or OddRatios of
        counts for each window.
degree  Degree of polynomial. default 3.
...     parameters could be passed to baseline.modpolyfit.

Details

This function implements the background correction methods of algorithm for polynomial fitting.
See details via baseline.modpolyfit. This function expects the tendency of decreasing of the
ratios from 5' end to 3' end.

Value

A vector of numeric. It is the background corrected log2-transformed ratios, CPMRatios or Odd-
Ratios.

Examples

x <- runif(200)
background <- rep(c(20:1)/100, each=10)
backgroundCorrection(x)

butterFilter

Low pass filter on ratios by butterworth filter

Description

The Butterworth filter is a type of signal processing filter designed to have as flat a frequency
response as possible in the passband.
Usage

butterFilter(ratios, N = ceiling(length(ratios)/200))

Arguments

ratios  A vector of numeric. It is log2-transformed ratios, CPMRatios or OddRatios in each window.

N      numeric(1) or integer(1). Critical frequencies of the low pass filter will be set as 1/N. 1/N is a cutoff at 1/N-th of the Nyquist frequency. By default, it is suppose there are about 200 peaks in the inputs.

Value

A vector of numeric with same length of input ratios. The vector indicates smoothed ratios.

Examples

ratios <- runif(20000)
butterFilter(ratios)

callPeaks

Call peaks using transformed, background corrected, and smoothed ratios with biological replicates

Description

Use limma to calculate p-values for NADs

Usage

callPeaks(
  se,
  backgroundCorrectedAssay = "bcRatio",
  normalization.method = "quantile",
  N = 100,
  cutoffAdjPvalue = 1e-04,
  countFilter = 1000,
  combineP.method = "minimump",
  smooth.method = "loess",
  lfc = log2(1.5),
  ...
)
callPeaks

Arguments

se     An object of `RangedSummarizedExperiment` with assays of raw counts, transformed ratios, background corrected ratios, smoothed ratios and z-scores. It should be an element of output of `smoothRatiosByChromosome`

backgroundCorrectedAssay

character(1). Assays names for background corrected log2-transformed ratios, CPMRatios or OddRatios.

normalization.method

character(1) specifying the normalization method to be used. Choices are "none", "scale", "quantile" or "cyclicloess". See `normalizeBetweenArrays` for details.

N     numeric(1) or integer(1). The number of neighboring windows used for loess smoothing or the inverse of the critical frequencies of the low pass filter for butterworth filter. 1/N is a cutoff at 1/N-th of the Nyquist frequency. Default 100.

cutoffAdjPvalue

numeric(1). Cutoff adjust p-value.

countFilter

numeric(1). Cutoff value for mean of raw reads count in each window.

combineP.method

A method used to combine P-values. Default `min`.

smooth.method

A method used to smooth the ratios. Choices are "loess", "none" and "butterworthfilter".

lfc

the minimum log2-fold-change that is considered scientifically meaningful

...  Parameter not used.

Details

By default, use the mean smoothed ratio for each peak region to calculate p-values

Value

An object of GRanges of peak list with metadata "AveSig", "P.Value", and "adj.P.Val", where "AveSig" means average signal such as average log2OddsRatio, log2CPMRatio or log2Ratio.

Author(s)

Jianhong Ou, Haibo Liu and Julie Zhu

Examples

data(triplicate.count)
se <- triplicate.count
se <- log2se(se, transformation = "log2CPMRatio",
            nucleolusCols = c("N18.subsampled.srt-2.bam",
                               "N18.subsampled.srt-3.bam",
                               "N18.subsampled.srt.bam"),
            genomeCols = c("G18.subsampled.srt-2.bam",
                           "G18.subsampled.srt-3.bam",
                           "G18.subsampled.srt.bam",
                           "G18.subsampled.srt.bam",
                           "G18.subsampled.srt.bam",
                           "G18.subsampled.srt.bam",
                           "G18.subsampled.srt.bam",
                           "G18.subsampled.srt.bam",
                           "G18.subsampled.srt.bam",...`
se<- smoothRatiosByChromosome(se, chr="chr18")
#add some variability to the data since the triplicate.count data was created using one sample only
assays(se[[1]])$bcRatio[,2] <- assays(se[[1]])$bcRatio[,2] + 0.3
assays(se[[1]])$bcRatio[,3] <- assays(se[[1]])$bcRatio[,3] - 0.3
peaks <- callPeaks(se[[1]],
                   cutoffAdjPvalue=0.001, countFilter=10)

---

**computeLibSizeChrom**

Perform overlap queries between reads and genome by sliding windows Count reads over sliding windows.

**Description**

Perform overlap queries between reads and genome by sliding windows Count reads over sliding windows.

**Usage**

computeLibSizeChrom(aln_list)

**Arguments**

- **aln_list** a list.

**Value**

A RangedSummarizedExperiment object with chromosome-level depth The assays slot holds the counts, rowRanges holds the annotation from the sliding widows of genome. metadata contains lib.size.chrom for holding chromosome-level sequence depth

**Author(s)**

Jun Yu, Hervé Pagès and Julie Zhu

---

**cumulativePercentage**

Plot the cumulative percentage of tag allocation

**Description**

Plot the difference between the cumulative percentage of tag allocation in paired samples.
cumulativePercentage

Usage

cumulativePercentage(
  se,
  binWidth = 1e+05,
  backgroundCorrectedAssay = "bcRatio",
  ...
)

Arguments

se An object of RangedSummarizedExperiment with assays of raw counts, transformed ratios, background correct ratios, smoothed ratios and z-scores. It should be an element of the output of smoothRatiosByChromosome.

binWidth numeric(1) or integer(1). The width of each bin.

backgroundCorrectedAssay character(1). Assays names for background correction ratios.

... Parameter not used.

Value

A list of data.frame with the cumulative percentages.

References


Examples

library(SummarizedExperiment)
data(triplicate.count)
se <- triplicate.count
se <- log2se(se, transformation = "log2CPMRatio",
  nucleolusCols = c("N18.subsampled.srt-2.bam",
  "N18.subsampled.srt-3.bam",
  "N18.subsampled.srt.bam"),
  genomeCols = c("G18.subsampled.srt-2.bam",
  "G18.subsampled.srt-3.bam",
  "G18.subsampled.srt.bam"))
se <- smoothRatiosByChromosome(se, chr="chr18")
cumulativePercentage(se[["chr18"]])
exportSignals  Output signals for visualization

Description
Output signals to bedgraph, bed, wig, etc, for track viewer

Usage
exportSignals(dat, assayName, colName, con, format = "bedGraph", ...)

Arguments
  dat  An object of GRanges, or RangedSummarizedExperiment with assays of raw counts, ratios, background correct ratios, smoothed ratios and z-scores. It should be an element of output of smoothRatiosByChromosome
  assayName  character(1). Assay name for RangedSummarizedExperiment
  colName  character(1). Column name of metadata of dat or assay of dat for coverage weight, see coverage, RangedSummarizedExperiment.
  con  The connection to which data is saved. If this is a character vector, it is assumed to be a filename and a corresponding file connection is created and then closed after exporting the object. If missing, a SimpleRleList will be returned.
  format  The format of the output. see export.
  ...  Parameters to be passed to export

Value
If con is missing, a SimpleRleList will be returned. Otherwise, nothing is returned.

Examples
gr <- GRanges("chr1", IRanges(seq_len(100), 201:300), reads=rep(1, 100))
myTrackLine <- new("TrackLine", name="my track",
                   description="description of my track",
                   color=col2rgb("red")[, 1],
                   visibility="full")
exportSignals(gr, colName="reads",
              con="test.bedGraph", trackLine=myTrackLine)
data(triplicate.count)
exportSignals(triplicate.count, "counts",
              "G18.subsampled.srt.bam", "test.bw", format="bigWig")
getCorrelations

Get correlation coefficients and p-values between biological replicates

Description

Get correlations and p-values between biological replicates based on coverage signal for peak regions. The signals will be filtered by the background cutoff value before calculated correlations. This function also output a correlation plots using the corplot.

Usage

```r
getCorrelations(
  se,
  chr = paste0("chr", seq_len(19)),
  ratioAssay = "ratio",
  window = 10000L,
  cutoff = 1,
  method = c("spearman", "pearson", "kendall"),
  file_name = "Correlation plots.pdf",
  ...
)
```

Arguments

- `se` A RangedSummarizedExperiment object. The output from log2se.
- `chr` A vector of character. Filter for seqnames. It should be the chromosome names to be kept.
- `ratioAssay` character(1). Column name of ratio for correlation calculation.
- `window` numeric(1) or integer(1). The window size for summary of the ratios.
- `cutoff` numeric(1). All the coverage signals lower than cutoff value in a given window will be filtered out.
- `method` character(1) indicating which correlation coefficient is to be computed. See cor.
- `file_name` A file name for output correlation plots
- `...` Parameters not used.

Value

A list of matrixes of correlation coefficients and p-values.

Author(s)

Jianhong Ou, Haibo Liu
Examples

data(triplicate.count)
se <- triplicate.count
se <- log2se(se, transformation = "log2CPMRatio",
            nucleolusCols = c("N18.subsampled.srt-2.bam",
              "N18.subsampled.srt-3.bam",
              "N18.subsampled.srt.bam"),
            genomeCols = c("G18.subsampled.srt-2.bam",
                "G18.subsampled.srt-3.bam",
                "G18.subsampled.srt.bam"))
getCorrelations(se, chr="chr18")

---

groupZscores

**Calculate z-scores for each peak**

Description

Detect peaks and calculate z-scores for each peak

Usage

groupZscores(zscore)

Arguments

zscore

A vector of numeric. It is the z-scores of ratios for each window.

Value

A data.frame with column names as "zscore", "group", "grp.zscore", and "pvalue".

Examples

x <- seq_len(500)
a <- 2 * 2*pi/length(x)
y <- 20 * sin(x*a)
noise1 <- 20 * 1/10 * sin(x*a*10)
zscore <- y+noise1
groupZscores(zscore)
**IntersectionNotStrict**  
*Count reads overlapping genomic ranges*

**Description**

Count reads overlapping a set of genome features represented as genomic ranges. This function does not work for parallel.

**Usage**

```r
IntersectionNotStrict(
  features,
  reads,
  ignore.strand = TRUE,
  inter.feature = FALSE
)
```

**Arguments**

- **features**: A object of GRanges representing the feature regions to be counted.
- **reads**: An object that represents the data to be counted. See `summarizeOverlaps`. If reads are more than 1 bam files, it should be a vector of character with full path, otherwise current working directory is the default directory. For paired end reads,
- **ignore.strand**: logical(1). ignore strand?
- **inter.feature**: not used. This parameter is required by `summarizeOverlaps`.

**Value**

return a summarized experiment object with chromosome-level depth information for each input sample as metadata.

---

**log2se**  
*calculate the log2 transformed ratios for SummarizedExperiment class*

**Description**

Calculate the log2 transformed ratios for nucleolus vs genome. pseudo-count will be used to avoid x/0 or log(0).
Usage

log2se(
  se, 
  nucleolusCols, 
  genomeCols, 
  pseudocount = 1L, 
  transformation = c("log2OddsRatio", "log2CPMRatio", "log2Ratio"), 
  chrom.level.lib = TRUE 
)

Arguments

se            A RangedSummarizedExperiment object. The output of tileCount.
nucleolusCols, genomeCols
  column Names of counts for nucleolus and genome. They should be the column names in the assays of se. Ratios will be calculated as log2(transformed nucleolusCols/transformed genomeCols).
pseudocount   default to 1, pseudo-count used to avoid x/0 or log(0).
transformation transformation type
  indicating whether calculating CPM or odds using sequence depth of the whole genome or the corresponding chromosome

Value

A RangedSummarizedExperiment object with log2 transformed ratios. Assays will be named as nucleolus, genome and ratio.

Author(s)

Jianhong Ou and Julie Zhu

Examples

library(SummarizedExperiment)
se <- SummarizedExperiment(assays=list(counts=DataFrame(A=seq_len(3), B=rep(1, 3), C=rep(4, 3), D=rep(2, 3))), rowRanges=GRanges(c("chr1","chr1", "chr2"), IRanges(c(1, 10, 20), width=9))),
metadata(se)$lib.size.chrom <- data.frame( c(1000, 1000), c(2000, 2000), c(200,200), c(300,300))
rownames(metadata(se)$lib.size.chrom) <- c("A", "B", "C", "D")
log2se(se, nucleolusCols = c("A", "C"), genomeCols = c("B", "D"), transformation = "log2Ratio")
log2se(se, nucleolusCols = c("A", "C"), genomeCols = c("B", "D"), transformation = "log2CPMRatio")
log2se(se, nucleolusCols = c("A", "C"), genomeCols = c("B", "D"), transformation = "log2OddsRatio")
peakdet

Detect peak positions

Description

Detect the peak positions and valley positions leveraging github::dgromer/peakdet

Usage

peakdet(y, delta = 0, silence = TRUE)

Arguments

- **y**: A numeric vector for searching peaks
- **delta**: A numeric vector of length 1, defining the minimum absolute changes required for local maximum or minimum detection when slope sign changes. If it is set to 0, the delta will be set to 1/10 of the range of y.
- **silence**: logical(1). If false, echo the delta value when delta is set as 0.

Value

A list with peakpos and valleypos. Both peakpos and valleypos are numeric vectors storing the positions of peaks or valleys.

Examples

```r
y <- runif(200)
peakdet(y)
y <- sin(seq(0,20))
peakdet(y)
```

plotSig

Plot signals with ideograms

Description

Plot signals with ideograms for GRangesList.

Usage

plotSig(ideo, grList, mcolName, ...)

plotSig(ideo, grList, mcolName, ...)
smoothRatiosByChromosome

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ideo</td>
<td>Output of loadIdeogram.</td>
</tr>
<tr>
<td>grList</td>
<td>A GRangesList of data to plot.</td>
</tr>
<tr>
<td>mcolName</td>
<td>Column name of metadata of GRangesList for plotting.</td>
</tr>
<tr>
<td>...</td>
<td>Parameters to pass to ideogramPlot.</td>
</tr>
</tbody>
</table>

Value

Invisible argument list for ideogramPlot.

Examples

library(trackViewer)
ideo <- loadIdeogram("mm10")
ideo <- readRDS(system.file("extdata", "ideo.mm10.rds",
                            package = "NADfinder"))
gr1 <- gr2 <- ideo
mcols(gr1) <- DataFrame(score=runif(length(gr1)))
mcols(gr2) <- DataFrame(score=runif(length(gr2)))
grList <- GRangesList(gr1, gr2)
plotSig(ideo, grList, mcolName="score", layout=list("chr1"))

---

single.count

Counts data for chromosome 18 for an experiment of a single pair of samples

Description

Counts data for chromosome 18 for an experiment of a single pair of samples

---

smoothRatiosByChromosome

Background correction and signal smoothing per chromosome

Description

Split the ratios by chromosome and do background correction and signal smoothing.
smoothRatiosByChromosome

Usage

smoothRatiosByChromosome(
  se,
  chr = paste0("chr", c(seq_len(21), "X", "Y")),
  ratioAssay = "ratio",
  backgroundCorrectedAssay = "bcRatio",
  smoothedRatioAssay = "smoothedRatio",
  zscoreAssay = "zscore",
  backgroundPercentage = 0.25,
  chrom.level.background = TRUE,
  ...
)

Arguments

se  An object of RangedSummarizedExperiment with log2-transformed ratios, CPM-Ratios or OddRatios. Output of log2se
chr A character vector, used to filter out seqnames. It should be the chromosome names to be kept.
ratioAssay The name of assay in se, which store the values (log2-transformed ratios, CPM-Ratios or OddRatios) to be smoothed.
backgroundPercentage numeric(1). Percentage of values for background, see zscoreOverBck. The percentage of values lower than this threshold will be treated as background, with 25 percentile as default.
chrom.level.background logical(1): TRUE or FALSE, default to TRUE, use chromosome-level background to calculate z-score
...
Parameters could be passed to butterFilter.

Value

A SimpleList of RangedSummarizedExperiment with smoothed ratios.

Author(s)

Jianhong Ou, Haibo Liu and Julie Zhu

Examples

data(single.count)
se <- single.count
dat <- log2se(se, nucleolusCols="N18.subsampled.srt.bam", genomeCols="G18.subsampled.srt.bam", transformation="log2CPMRatio")
dat1 <- smoothRatiosByChromosome(dat, N=100, chr = c("chr18", "chr19"))
dat2 <- smoothRatiosByChromosome(dat, N=100, chr = c("chr18", "chr19"),
                                       chrom.level.background = FALSE)

---

## tileCount

### Description

tileCount extends `summarizeOverlaps` by finding coverage for each fixed window in the whole genome.

### Usage

```r
tileCount(
  reads,
  genome,
  excludeChrs = c("chrM", "M", "Mt", "MT"),
  windowSize = 50000,
  step = 10000,
  mode = IntersectionNotStrict,
  dataOverSamples = FALSE,
  ...
)
```

### Arguments

- **reads**: A `GRanges`, `GRangesList` (should be one read per list element), `GAlignments`, `GAlignmentsList`, `GAlignmentPairs` or `BamFileList` object that represents the data to be counted by `summarizeOverlaps`. If reads are more than 1 bam files, it should be a vector of character with full path, otherwise current working directory is the default directory.
- **genome**: A `BSgenome` object from/on which to get/set the sequence and metadata information.
- **excludeChrs**: A vector of string: chromosomes/scaffolds of no interest for NAD analysis. see `summarizeOverlaps`. default is countByOverlaps, alia of countOverlaps(features, reads, ignore.strand=ignore.strand)
- **windowSize**: numeric(1) or integer(1). Size of the windows.
- **step**: numeric(1) or integer(1). Step of generating sliding windows.
- **mode**: One of the pre-defined count methods.
- **dataOverSamples**: logical(1). Data over several samples when use `GRangesList` as input.
- **...**: Additional arguments passed to `summarizeOverlaps`. 

### Example

```r
dat2 <- smoothRatiosByChromosome(dat, N=100, chr = c("chr18", "chr19"),
                                       chrom.level.background = FALSE)
```
Value

A `RangedSummarizedExperiment` object. The assays slot holds the counts, rowRanges holds the annotation from the sliding widows of genome. metadata contains lib.size.chrom for holding chromosome-level sequence depth

Author(s)

Jianhong Ou, Haibo Liu, Herve Pages and Julie Zhu

Examples

```r
if (interactive()) {
  fls <- list.files(system.file("extdata", package="NADfinder"), recursive=FALSE, pattern="*bam$", full=TRUE)
  names(fls) <- basename(fls)
  if (!require(BSgenome.Mmusculus.UCSC.mm10)) {
    if (!requireNamespace("BiocManager", quietly=TRUE))
      install.packages("BiocManager")
    BiocManager::install("BSgenome.Mmusculus.UCSC.mm10")
    library(BSgenome.Mmusculus.UCSC.mm10)
  }
  se <- tileCount2(reads = fls, windowSize=50000, step=10000)
}
```

---

**tileCount2**

*Perform overlap queries between reads and genome by sliding windows Count reads over sliding windows.*

Description

Perform overlap queries between reads and genome by sliding windows Count reads over sliding windows.

```r
if (interactive())
fls <- list.files(system.file("extdata", package="NADfinder"), recursive=FALSE, pattern="*bam$", full=TRUE)
names(fls) <- basename(fls)
se <- tileCount2(reads = fls, windowSize=50000, step=10000)
```
Usage

```r
tileCount2(
  reads,
  fragment.length = 100,
  windowSize = 50000,
  restrict = paste0("chr", c(1:19, "X", "Y")),
  step = 1000,
  filter = 0,
  pe = "both"
  )
```

tileCount2(
  reads,
  fragment.length = 100,
  windowSize = 50000,
  restrict = paste0("chr", c(1:19, "X", "Y")),
  step = 1000,
  filter = 0,
  pe = "both"
  )

Arguments

- **reads**
  - An object that represents the names and path of the bam files to be counted. If reads are more than 1 bam files, it should be a vector of character with full path. This function now works for paired end reads

- **fragment.length**
  - integer(1). An integer scalar or a list of two integer scalars/vectors, containing the average length(s) of the sequenced fragments in each library.

- **windowSize**
  - numeric(1) or integer(1). Size of the windows.

- **restrict**
  - restrict to a set of chromosomes, default to mouse chromosomes.

- **step**
  - numeric(1) or integer(1). Step of generating sliding windows.

- **filter**
  - default to 0 without filtering. An integer scalar for the minimum count sum across libraries for each window.

- **pe**
  - a character string indicating whether paired-end data is present; set to "none", "both", "first" or "second"

Value

A `RangedSummarizedExperiment` object with chromosome-level depth. The assays slot holds the counts, rowRanges holds the annotation from the sliding widows of genome. metadata contains lib.size.chrom for holding chromosome-level sequence depth

Author(s)

Jun Yu, Hervé Pagès and Julie Zhu
Examples

```r
if (interactive())
{
  fls <- list.files(system.file("extdata", package="NADfinder"),
                    recursive=FALSE, pattern="*bam$", full=TRUE)
  names(fls) <- basename(fls)

  se <- tileCount2(reads = fls,
                   windowSize=50000, step=10000)
}
```

transformData  transform counts to log2 cpm ratios, log2 ratios or log2 odds ratios

Description

calculate the log2 ratios, log2 cpm (count per million) ratios, or log2 odds ratios for nucleolus vs
genome. pseudo-count will be used to avoid x/0 or log(0).

Usage

```r
transformData(
  A,
  B,
  seqnames.A,
  seqnames.B,
  pseudo.count = 1L,
  transformation = c("log2OddsRatio", "log2CPMRatio", "log2Ratio"),
  chrom.level.lib = TRUE,
  lib.size.A,
  lib.size.B
)
```

Arguments

- **A, B**  
  window-level counts for nucleolus and genome, extracted from the assays of the
  output of the tileCounts function
- **seqnames.A, seqnames.B**  
  seqnames, extracted from the rowRanges of the output of the tileCounts function
- **pseudo.count**  
  pseudo-count will be used to avoid x/0 or log(0), default to 1.
- **transformation**  
  transformation type
- **chrom.level.lib**  
  indicating whether calculating CPM or odds using sequence depth of the whole
genome or the corresponding chromosome
library size for A and B, these two dataframes contain chromosome-level sequence depth for the chromosomes, which can be extracted from the metadata of the output of the tileCounts function.

Value

A numeric vector of log2 ratios, log2 CPM ratios or log2 odds ratios.

Author(s)

Julie Zhu

Examples

```r
transformData(seq_len(10), 10:1, seqnames.A = Rle(c("chr1", "chr2")), c(5,5)),
Rle(c("chr1", "chr2"), c(5,5)), transformation = "log2OddsRatio",
chrom.level.lib = FALSE, lib.size.A = cbind(c("chr1", "chr2"), c(10000, 12000)),
lib.size.B = cbind(c("chr1", "chr2"), c(10000, 12000)))
```

Description

Filter the peaks by pvalue and trim the range of peaks for an NAD or ChIP-seq experiment without biological replicates.

Usage

```r
trimPeaks(
  se,
  cutoffAdjPvalue = 0.05,
  padjust.method = "BH",
  backgroundPercentage = 0.25,
  countFilter = 1000,
)```
triplicate.count

Counts data for chromosome 18 for an expriment with triplicates

Description

Counts data for chromosome 18 for an expriment with triplicates

```r
ratioAssay = "ratio",
backgroundCorrectedAssay = "bcRatio",
smoothedRatioAssay = "smoothedRatio",
zscoreAssay = "zscore"
)

Arguments

se
An object of RangedSummarizedExperiment with assays of raw counts, ratios, background corrected ratios, smoothed ratios and z-scores. It should be an element of the output of smoothRatiosByChromosome
cutoffAdjPvalue
numeric(1). Cutoff of adjusted p-value.
padjust.method
character(1). The method to use for adjusting p-values, which is passed to p.adjust function
backgroundPercentage
numeric(1). Cutoff value for the peaks height.
countFilter
numeric(1) or integer(1). Cutoff value for mean of raw reads count of the Nucleolar/ChIP samples in each window.
ratioAssay
character(1). The name of assay in se, which store the values to be smoothed.
backgroundCorrectedAssay, smoothedRatioAssay, zscoreAssay
Assays names for background-corrected ratios, smoothed ratios and z-scores based on background corrected ratios.

Value

An object of GRanges.

Examples

data(single.count)
se <- single.count
dat <- log2se(se, nucleolusCols="N18.subsampled.srt.bam", genomeCols="G18.subsampled.srt.bam",
transformation="log2CPMRatio")
## Smooth the ratios for each chromosome.
dat <- smoothRatiosByChromosome(dat, N=100, chr=c("chr18","chr19"))
peaks <- trimPeaks(dat["chr18"],
  backgroundPercentage=.25,
  cutoffAdjPvalue=0.05, countFilter=1000)
```
zscoreOverBck  

Z-scores over the background

Description

Calculate the z-scores over the background distribution.

Usage

zscoreOverBck(ratios, backgroundPercentage = 0.25)

Arguments

ratios  
A numeric vector containing the transformed, background corrected and smoothed ratios in each window.

backgroundPercentage  
numeric(1). Low percentile for background distribution.

Value

A vector of numeric. Z-scores.

Author(s)

Jianhong Ou and Julie Zhu

Examples

r <- runif(200)
zscoreOverBck(r)
Index

* data
  single.count, 14
  triplicate.count, 21
backgroundCorrection, 3
BamFileList, 16
baseline.modpolyfit, 3
butterFilter, 3, 15

callPeaks, 4
computeLibSizeChrom, 6
cor, 9
corrplot, 9
coverage, 8
cumulativePercentage, 6

extport, 8
exportSignals, 8

GAlignmentPairs, 16
GAlignments, 16
GAlignmentsList, 16
getCorrelations, 9
GRanges, 8, 11, 16, 21
GRangesList, 13, 14, 16
groupZscores, 10

ideogramPlot, 14
IntersectionNotStrict, 11

loadIdeogram, 14
log2se, 9, 11, 15

NADfinder (NADfinder-package), 2
NADfinder-package, 2
normalizeBetweenArrays, 5

peakdet, 13
plotSig, 13

RangedSummarizedExperiment, 5–9, 12, 15, 17, 18, 21

SimpleList, 15
SimpleRleList, 8
single.count, 14
smoothRatiosByChromosome, 5, 7, 8, 14, 21
summarizeOverlaps, 11, 16
tileCount, 12, 16
tileCount2, 17
transformData, 19
trimPeaks, 20
triplicate.count, 21
zscoreOverBck, 15, 22