Package 'ATACseqTFEA'

October 25, 2024

Type Package

Title Transcription Factor Enrichment Analysis for ATAC-seq

Version 1.7.0

Description Assay for Transpose-Accessible Chromatin using sequencing (ATAC-seq) is a technique to assess genome-wide chromatin accessibility by probing open chromatin with hyperactive mutant Tn5 Transposase that inserts sequencing adapters into open regions of the genome. ATACseqTFEA is an improvement of the current computational method that detects differential activity of transcription factors (TFs). ATACseqTFEA not only uses the difference of open region information, but also (or emphasizes) the difference of TFs footprints (cutting sites or insertion sites). ATACseqTFEA provides an easy, rigorous way to broadly assess TF activity changes between two conditions.

BugReports https://github.com/jianhong/ATACseqTFEA/issues

URL https://github.com/jianhong/ATACseqTFEA

Depends R (>= 4.2)

Imports BiocGenerics, S4Vectors, IRanges, Matrix, GenomicRanges, GenomicAlignments, GenomeInfoDb, SummarizedExperiment, Rsamtools, motifmatchr, TFBSTools, stats, pracma, ggplot2, ggrepel, dplyr, limma, methods, rtracklayer

Suggests BSgenome.Drerio.UCSC.danRer10, knitr, testthat, ATACseqQC, rmarkdown, BiocStyle

biocViews Sequencing, DNASeq, ATACSeq, MNaseSeq, GeneRegulation

License GPL-3

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.2.3

git_url https://git.bioconductor.org/packages/ATACseqTFEA

git_branch devel

git_last_commit f88ce03

git_last_commit_date 2024-04-30

Repository Bioconductor 3.20

Date/Publication 2024-10-25

Author Jianhong Ou [aut, cre] (<https://orcid.org/0000-0002-8652-2488>)

Maintainer Jianhong Ou <jianhong.ou@duke.edu>

Contents

`ATACseqTFEA-package` *Transcription Factor Enrichment Analysis for ATAC-seq*

Description

Assay for Transpose-Accessible Chromatin using sequencing (ATAC-seq) is a technique to assess genome-wide chromatin accessibility by probing open chromatin with hyperactive mutant Tn5 Transposase that inserts sequencing adapters into open regions of the genome. ATACseqTFEA is an improvement of the current computational method that detects differential activity of transcription factors (TFs). ATACseqTFEA not only uses the difference of open region information, but also (or emphasizes) the difference of TFs footprints (cutting sites or insertion sites). ATACseqTFEA provides an easy, rigorous way to broadly assess TF activity changes between two conditions.

Author(s)

Maintainer: Jianhong Ou <jianhong.ou@duke.edu> (ORCID)

See Also

Useful links:

- https://github.com/jianhong/ATACseqTFEA
- Report bugs at https://github.com/jianhong/ATACseqTFEA/issues

calWeights *Calculate the weights for binding score*

Description

Use open score to calculate the weights for the binding score. The open score is calculated by the counts of the proximal region divided by the counts of the distal region. And the counts RangedSummarizedExperiment will be filtered by the Z-score of the open score. The weight is calculated by converting the Z score to the range of 0-1 following the normal distribution.

Usage

```
calWeights(se, openscoreZcutoff = 0, ...)
```

Arguments

se	An RangedSummarizedExperiment object. Outputs of countsNormalization.
openscoreZcutoff	Open score Z value cutoff value. Default is 0. Open score is calculated by the count ratio of proximal site and distal site.
...	Not used.

Value

A RangedSummarizedExperiment object with assays of count matrix with bindingSites, proximalRegion and distalRegion as column names and bindingSites GRanges object with the weights as rowRanges.

Author(s)

Jianhong Ou

Examples

```
bam <- system.file("extdata",
                   "KD.shift.rep1.bam",
                   package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
## get the count regions
bsEx <- expandBindingSites(bindingSites)
## count reads by 5'ends
res <- count5ends(bam, positive=0L, negative=0L,
                  bindingSites=bindingSites,
                  bindingSitesWithGap=bsEx$bindingSitesWithGap,
                  bindingSitesWithProximal=bsEx$bindingSitesWithProximal,
                  bindingSitesWithProximalAndGap=
                      bsEx$bindingSitesWithProximalAndGap,
                  bindingSitesWithDistal=bsEx$bindingSitesWithDistal)
## filter 0 counts in proximal
se <- eventsFilter(res, proximalRegion>0)
## normalize counts by width of count region
```

```
se <- countsNormalization(se, proximal=40, distal=40)
## calculate the weights
calWeights(se)
```

`count5ends` *Prepare counts matrix for enrichment analysis*

Description

Prepare the counts matrix by 5'end of reads.

Usage

```
count5ends(
  bam,
  index = bam,
  yieldSize = 1e+05,
  positive = 4L,
  negative = 5L,
  bindingSites,
  bindingSitesWithGap,
  bindingSitesWithProximal,
  bindingSitesWithProximalAndGap,
  bindingSitesWithDistal
)
```

Arguments

`bam`	A character vector indicates the file names of the bams or an object of BamFile.
`index`	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
`yieldSize`	Number of records to yield each time the file is read. See BamFile for details.
`positive, negative`	integer(1). the size to be shift for positive/negative strand. If the bam file is 5'end shifed files, please set the parameter to 0.
`bindingSites`	A object of GenomicRanges indicates candidate binding sites. The prepareBindingSites function is a helper function to generate the binding sites. Users can also use other software for example fimo to generate the list.
`bindingSitesWithGap`	bindingSites with gaps and in both ends,
`bindingSitesWithProximal`	bindingSites with gaps and proximal region in both ends,
`bindingSitesWithProximalAndGap`	bindingSites with gaps, and then proximal and gaps in both ends,
`bindingSitesWithDistal`	bindingSites with gap, proximal, gap and distal regions.

Value

A RangedSummarizedExperiment object with assays of count matrix with bindingSites, proximalRegion and distalRegion as column names and bindingSites GRanges object as rowRanges.

Author(s)

Jianhong Ou

Examples

```
bam <- system.file("extdata",
                   "KD.shift.rep1.bam",
                   package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
## get the count regions
bsEx <- expandBindingSites(bindingSites)
res <- count5ends(bam, positive=0L, negative=0L,
                  bindingSites=bindingSites,
                  bindingSitesWithGap=bsEx$bindingSitesWithGap,
                  bindingSitesWithProximal=bsEx$bindingSitesWithProximal,
                  bindingSitesWithProximalAndGap=
                      bsEx$bindingSitesWithProximalAndGap,
                  bindingSitesWithDistal=bsEx$bindingSitesWithDistal)
head(res)
```

`countsNormalization` *Normalize counts by width of count region*

Description

Do normalization by width for counts in binding sites, proximal and distal regions.

Usage

```
countsNormalization(se, proximal, distal)
```

Arguments

`se`	An RangedSummarizedExperiment object. Outputs of count5ends or eventsFilter.
`proximal, distal`	numeric(1) or integer(1). bases for open region from binding sites (proximal) and extended region for background (distal) of the binding region for aggregate ATAC-seq footprint.

Value

A RangedSummarizedExperiment object with assays of count matrix with bindingSites, proximalRegion and distalRegion as column names and bindingSites GRanges object as rowRanges.

Author(s)

Jianhong Ou

Examples

```
bam <- system.file("extdata",
                   "KD.shift.rep1.bam",
                   package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
## get the count regions
bsEx <- expandBindingSites(bindingSites)
## count reads by 5'ends
res <- count5ends(bam, positive=0L, negative=0L,
                  bindingSites=bindingSites,
                  bindingSitesWithGap=bsEx$bindingSitesWithGap,
                  bindingSitesWithProximal=bsEx$bindingSitesWithProximal,
                  bindingSitesWithProximalAndGap=
                      bsEx$bindingSitesWithProximalAndGap,
                  bindingSitesWithDistal=bsEx$bindingSitesWithDistal)
## filter 0 counts in proximal
se <- eventsFilter(res, proximalRegion>0)
## normalize counts by width of count region
countsNormalization(se, proximal=40, distal=40)
```

DBscore *Differential binding analysis*

Description

Use limma to do differential binding analysis for binding scores.

Usage

```
DBscore(se, design, coef, ...)
```

Arguments

se	An RangedSummarizedExperiment object. Outputs of getWeightedBindingScore.
design	Design table for lmFit.
coef	column number or column name specifying which coefficient or contrast of the linear model is of interest. See topTable.
...	Parameters can be used by lmFit.

Value

A RangedSummarizedExperiment object with the dataframe returned by topTable as appendence of the origin rowData.

Author(s)

Jianhong Ou

Examples

```
library(SummarizedExperiment)
set.seed(1)
sigma2 <- 0.05 / rchisq(100, df=10) * 10
y <- matrix(rnorm(100*6,sd=sqrt(sigma2)),100,6)
design <- cbind(Intercept=1,Group=c(0,0,0,1,1,1))
y[1,4:6] <- y[1,4:6] + 1
se <- SummarizedExperiment(assays=list(counts=y))
DBscore(se, design, coef=1)
```

---

doTFEA *Transcription factor enrichment analysis*

Description

Transcription factor enrichment analysis for the filtered output of DBscore

Usage

```
doTFEA(se, ...)
```

Arguments

se	An RangedSummarizedExperiment object. Filtered outputs of DBscore.
...	Not used.

Value

A TFEAresults object.

Author(s)

Jianhong Ou

Examples

```
bamExp <- system.file("extdata",
                      c("KD.shift.rep1.bam",
                        "KD.shift.rep2.bam"),
                      package="ATACseqTFEA")
bamCtl <- system.file("extdata",
                      c("WT.shift.rep1.bam",
                        "WT.shift.rep2.bam"),
                      package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
## get the count regions
bsEx <- expandBindingSites(bindingSites)
## count reads by 5'ends
res <- count5ends(c(bamExp, bamCtl),
                  positive=0L, negative=0L,
                  bindingSites=bindingSites,
```

```
                  bindingSitesWithGap=bsEx$bindingSitesWithGap,
                  bindingSitesWithProximal=bsEx$bindingSitesWithProximal,
                  bindingSitesWithProximalAndGap=
                      bsEx$bindingSitesWithProximalAndGap,
                  bindingSitesWithDistal=bsEx$bindingSitesWithDistal)
## filter 0 counts in proximal
se <- eventsFilter(res, proximalRegion>0)
## normalize counts by width of count region
se <- countsNormalization(se, proximal=40, distal=40)
## get the weighted binding scores
se <- getWeightedBindingScore(se)
design <- cbind(CTL=1, EXPvsCTL=c(1, 1, 0, 0))
rownames(design) <- colnames(se)
counts <- DBscore(se, design=design, coef="EXPvsCTL")
doTFEA(counts)
```

ESvolcanoplot *Plot enrichment score for one transcription factor*

Description

Plot GSEA style enrichment score curve.

Usage

```
ESvolcanoplot(
  TFEAresults,
  xlab = "Enrichment Score",
  ylab = "-log10(p value)",
  TFnameToShow = 20,
  significantCutoff = 0.05,
  col = c("red", "blue", "gray"),
  ...
)
```

Arguments

TFEAresults	A TFEAresults object. Output of TFEA.
xlab, ylab	character string giving label for x-axis/y-axis.
TFnameToShow	Transcription factor names to be drawn.
significantCutoff	Cutoff value for significant.
col	Color sets for the points.
...	parameter passed to pdf.

Value

ggplot object.

Examples

```
res <- system.file("extdata", "res.rds", package="ATACseqTFEA")
res <- readRDS(res)
ESvolcanoplot(TFEAresults=res)
```

`eventsFilter` *Filter the RangedSummarizedExperiment objects*

Description

A helper function to subset the counts object outputed by count5ends.

Usage

```
eventsFilter(se, filter)
```

Arguments

`se`	An RangedSummarizedExperiment object. Outputs of count5ends.
`filter`	An expression which, when evaluated in the context of assays(se), is a logical vector indicating elements or rows to keep. The expression results for each assay will be combined and use 'or' operator to filter the counts assays.

Value

A RangedSummarizedExperiment object with assays of count matrix with bindingSites, proximalRegion and distalRegion as column names and bindingSites GRanges object as rowRanges.

Author(s)

Jianhong Ou

Examples

```
bam <- system.file("extdata",
                   "KD.shift.rep1.bam",
                   package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
## get the count regions
bsEx <- expandBindingSites(bindingSites)
## count reads by 5'ends
res <- count5ends(bam, bindingSites=bindingSites,
                  bindingSitesWithGap=bsEx$bindingSitesWithGap,
                  bindingSitesWithProximal=bsEx$bindingSitesWithProximal,
                  bindingSitesWithProximalAndGap=
                      bsEx$bindingSitesWithProximalAndGap,
                  bindingSitesWithDistal=bsEx$bindingSitesWithDistal)
eventsFilter(res, proximalRegion>0)
eventsFilter(res, seqnames(res)=="chr1")
eventsFilter(res, sample(c(TRUE, FALSE), length(res), replace=TRUE))
eventsFilter(res, "proximalRegion>0")
```

```
filter <- "proximalRegion>0"
eventsFilter(res, filter)
filter <- sample(c(TRUE, FALSE), length(res), replace=TRUE)
eventsFilter(res, filter)
```

`expandBindingSites` *Prepare the genomic ranges for proximal and distal regions for counting*

Description

Create multiple GRanges objects for downstream counting. The GRanges objects including bindingSitesWithGap: bindingSites with gaps and in both ends, bindingSitesWithProximal: bindingSites with gaps and proximal region in both ends, bindingSitesWithProximalAndGap: bindingSites with gaps, and then proximal and gaps in both ends, and bindingSitesWithDistal: bindingSites with gaps, proximal, gaps and distal regions.

Usage

```
expandBindingSites(bindingSites, proximal = 40L, distal = proximal, gap = 10L)
```

Arguments

`bindingSites`	A object of GenomicRanges indicates candidate binding sites. The prepareBindingSites function is a helper function to generate the binding sites. Users can also use other software for example fimo to generate the list.
`proximal, distal`	numeric(1) or integer(1). bases for open region from binding sites (proximal) and extended region for background (distal) of the binding region for aggregate ATAC-seq footprint.
`gap`	numeric(1) or integer(1). bases for gaps among binding sites, proximal, and distal. default is 10L.

Value

an GRangesList object with elements bindingSitesWithGap, bindingSitesWithProximal, bindingSitesWithProximalAndGap, and bindingSitesWithDistal for count5ends

Author(s)

Jianhong Ou

Examples

```
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
bs <- expandBindingSites(bindingSites)
length(bs)
names(bs)
lengths(bs)
```

extdata *Data in extdata*

Description

The list of data saved in extdata folder.

Details

The 'PWMatrixList' is a collection of jasper2018, jolma2013 and cisbp_1.02 from package motifDB (v 1.28.0) and merged by distance smaller than 1e-9 calculated by MotIV::motifDistances function (v 1.42.0). The merged motifs were exported by motifStack (v 1.30.0).

The 'cluster_PWMs' is a list of non-redundant TF motifs downloaded from [DeepSTARR](https://github.com/bernardo-de-almeida/motif-clustering). There are 6502 motifs in the data set.

The 'best_curated_Human' is a list of TF motifs downloaded from [TFEA github](https://github.com/Dowell-Lab/TFEA). There are 1279 human motifs in the data set.

Examples

```
motifs <- readRDS(system.file("extdata", "PWMatrixList.rds",
                  package="ATACseqTFEA"))
motifs2 <- readRDS(system.file("extdata", "cluster_PWMs.rds",
                   package="ATACseqTFEA"))
motifs3 <- readRDS(system.file("extdata", "best_curated_Human.rds",
                   package="ATACseqTFEA"))
```

getEnrichmentScore *The methods for TFEAresults-class*

Description

The assessment and replacement methods for TFEAresults-class

Usage

```
getEnrichmentScore(x)

getBindingSites(x, TF)

getMotifID(x)

## S4 method for signature 'TFEAresults'
show(object)

## S4 method for signature 'TFEAresults'
x$name

## S4 replacement method for signature 'TFEAresults'
x$name <- value
```

```
## S4 method for signature 'TFEAresults,ANY,ANY'
x[[i, j, ..., exact = TRUE]]

## S4 replacement method for signature 'TFEAresults,ANY,ANY'
x[[i, j, ...]] <- value

## S4 method for signature 'TFEAresults'
getEnrichmentScore(x)

## S4 method for signature 'TFEAresults'
getBindingSites(x, TF)

## S4 method for signature 'TFEAresults'
getMotifID(x)
```

Arguments

x	TFEAresults object.
TF	Transcription factor
object	an object of TFEAresults
name	A literal character string or a name (possibly backtick quoted).
value	value to replace.
i, j	indices specifying elements to extract or replace.
...	Named or unnamed arguments to form a signature.
exact	see Extract

Value

The ‘getEnrichmentScore‘ method will return the enrichment score matrix.

The ‘getBindingSites‘ method will return a GRanges object indicates binding sites.

The method ‘getMotifID‘ will return A list of positions of the binding sites for the motifs.

Examples

```
res <- readRDS(system.file("extdata", "res.rds", package="ATACseqTFEA"))
as(res, "data.frame")
res
head(res$resultsTable)
head(res[["resultsTable"]])
head(getEnrichmentScore(res))
```

`getWeightedBindingScore` *Calculate the weighted binding score*

Description

Use user predefined weight to get the weighted binding score or use open score to weight the binding score. The open score is calculated by the counts of proximal region divided by the counts of distal region. The binding score is calculated by the counts of proximal region divided by the counts of binding region. This value is the measure of avoidance of reads in the binding sites.

Usage

```
getWeightedBindingScore(se, weight = NA, ...)
```

Arguments

se	An RangedSummarizedExperiment object. Outputs of countsNormalization.
weight	If NA, the weight will be calculated by the open score. See calWeights. User can define the weight by a matrix or numeric vector.
...	The parameters will be passed to calWeights.

Value

A RangedSummarizedExperiment object with assays of count matrix with bindingSites, proximalRegion and distalRegion as column names and bindingSites GRanges object as rowRanges.

Author(s)

Jianhong Ou

Examples

```
bam <- system.file("extdata",
                   "KD.shift.rep1.bam",
                   package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
## get the count regions
bsEx <- expandBindingSites(bindingSites)
## count reads by 5'ends
res <- count5ends(bam, positive=0L, negative=0L,
                  bindingSites=bindingSites,
                  bindingSitesWithGap=bsEx$bindingSitesWithGap,
                  bindingSitesWithProximal=bsEx$bindingSitesWithProximal,
                  bindingSitesWithProximalAndGap=
                      bsEx$bindingSitesWithProximalAndGap,
                  bindingSitesWithDistal=bsEx$bindingSitesWithDistal)
## filter 0 counts in proximal
se <- eventsFilter(res, proximalRegion>0)
## normalize counts by width of count region
se <- countsNormalization(se, proximal=40, distal=40)
## get the weighted binding scores
getWeightedBindingScore(se)
```

`importFimoBindingSites` *Prepare binding site by fimo results*

Description

Prepare binding sites by given fimo gff files

Usage

```
importFimoBindingSites(
  fimoGFFfiles,
  maximalBindingWidth = 40L,
  mergeBindingSitesByPercentage = 0.8,
  ignore.strand = TRUE,
  ...
)
```

Arguments

`fimoGFFfiles`	Filenames of gff files of fimo output.
`maximalBindingWidth`	A numeric vector(length=1). Maximal binding site width. Default is 40.
`mergeBindingSitesByPercentage`	A numeric vector (length=1). The percentage of overlapping region of binding sites to merge as one binding site.
`ignore.strand`	When set to TRUE, the strand information is ignored in the calculations.
`...`	Parameter to be passed to import.gff

Value

A GenomicRanges with all the positions of matches.

Author(s)

Jianhong Ou

Examples

```
extdata <- system.file('extdata', package='ATACseqTFEA')
fimoGFFfiles <- dir(extdata, 'fimo.*.gff', full.names=TRUE)
mts <- importFimoBindingSites(fimoGFFfiles)
```

plotES *Plot enrichment score for one transcription factor*

Description

Plot GSEA style enrichment score curve.

Usage

```
plotES(
  TFEAresults,
  TF,
  outfolder = ".",
  xlab = "rank",
  ylab = "Enrichment",
  resolution = 500L,
  device = "pdf",
  ...
)
```

Arguments

TFEAresults	A TFEAresults object. Output of TFEA.
TF	A character vector. The transcription factor names.
outfolder	character(1). Output file path.
xlab, ylab	character string giving label for x-axis/y-axis.
resolution	integer(1). The number of bars plotted in the bottom of figure to show the density of occurrence of events.
device	Device to use. Can be one of "eps", "ps", "tex" (pictex), "pdf", "jpeg", "tiff", "png", "bmp", "svg" or "wmf" (windows only).
...	parameter passed to ggsave.

Value

NULL if outfolder is set or ggplot object.

Examples

```
res <- system.file("extdata", "res.rds", package="ATACseqTFEA")
res <- readRDS(res)
g <- plotES(res, TF="KLF9", outfolder=NA)
g
```

`prepareBindingSites` *Prepare binding site for TFEA*

Description

Prepare binding sites by given position weight matrix and genome.

Usage

```
prepareBindingSites(
  pwms,
  genome,
  seqlev = seqlevels(genome),
  p.cutoff = 1e-05,
  w = 7,
  grange,
  maximalBindingWidth = 40L,
  mergeBindingSitesByPercentage = 0.8,
  ignore.strand = TRUE
)
```

Arguments

`pwms`	either `PFMatrix`, `PFMatrixList`, `PWMatrix`, `PWMatrixList`
`genome`	`BSgenome` object.
`seqlev`	A character vector. Sequence levels to be searched.
`p.cutoff`	p-value cutoff for returning motifs; default is 1e-05
`w`	parameter controlling size of window for filtration; default is 7
`grange`	GRanges for motif search. If it is set, function will only search the binding site within the grange. Usually a peak list should be supplied.
`maximalBindingWidth`	A numeric vector(length=1). Maximal binding site width. Default is 40.
`mergeBindingSitesByPercentage`	A numeric vector (length=1). The percentage of overlapping region of binding sites to merge as one binding site.
`ignore.strand`	When set to TRUE, the strand information is ignored in the calculations.

Value

A `GenomicRanges` with all the positions of matches.

Author(s)

Jianhong Ou

Examples

```
library(TFBSTools)
motifs <- readRDS(system.file("extdata", "PWMatrixList.rds",
                              package="ATACseqTFEA"))
library(BSgenome.Drerio.UCSC.danRer10)
seqlev <- "chr1" #paste0("chr", 1:25)
mts <- prepareBindingSites(motifs, Drerio, seqlev,
                           grange=GRanges("chr1",
                                          IRanges(5000, 100000)))
```

`reduceByPercentage` *Reduce by percentage of overlaps of GRanges object*

Description

Merge the ranges by percentage of overlaps to avoid broad ranges of continues ranges overlapped with limit bases.

Usage

```
reduceByPercentage(
  query,
  percentage,
  ignore.strand = TRUE,
  colnToKeep = c("score", "motif")
)
```

Arguments

`query`	An object of GRanges
`percentage`	A numeric vector (length=1). The percentage of overlapping region of binding sites to merge as one range.
`ignore.strand`	When set to TRUE, the strand information is ignored in the calculations.
`colnToKeep`	The metadata colnums should be kept for reduced GRanges

Value

An object of GRanges.

Examples

```
library(GenomicRanges)
gr <- GRanges("chr1", IRanges(c(1, 5, 10), width=c(10, 5, 2)))
reduceByPercentage(gr, 0.5, colnToKeep=NULL)
```

TFEA *Transcription factor enrichment analysis*

Description

Transcription factor enrichment analysis for ATAC-seq (Assay for Transposase-Accessible Chromatin using sequencing). We treat all the binding sites for one TF as a TF set and all the open regions as features for random walking.

Usage

```
TFEA(
  bamExp,
  bamCtl,
  indexExp = bamExp,
  indexCtl = bamCtl,
  positive = 4L,
  negative = 5L,
  bindingSites,
  proximal = 40L,
  distal = proximal,
  gap = 10L,
  filter = "proximalRegion>0",
  openscoreZcutoff = 0,
  bindingScoreLog2FCcutoff = 0,
  bindingScorePvalCutoff = 1
)
```

Arguments

bamExp	A vector of characters indicates the file names of experiment bams. The bam file must be the one with shifted reads.
bamCtl	A vector of characters indicates the file names of control bams. The bam file must be the one with shifted reads.
indexExp, indexCtl	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
positive, negative	integer(1). the size to be shift for positive/negative strand. If the bam file is 5'end shifed files, please set the parameter to 0.
bindingSites	A object of GenomicRanges indicates candidate binding sites. The prepareBindingSites function is a helper function to generate the binding sites. Users can also use other software for example fimo to generate the list.
proximal, distal	numeric(1) or integer(1). bases for open region from binding sites (proximal) and extended region for background (distal) of the binding region for aggregate ATAC-seq footprint.
gap	numeric(1) or integer(1). bases for gaps among binding sites, proximal, and distal. default is 10L.

`filter` An expression which, when evaluated in the context of assays(se), is a logical vector indicating elements or rows to keep. The expression results for each assay will be combined and use 'or' operator to filter the counts assays.

`openscoreZcutoff`
Open score Z value cutoff value. Default is 0. Open score is calculated by the count ratio of proximal site and distal site.

`bindingScorePvalCutoff`, `bindingScoreLog2FCcutoff`
Binding score cutoff values. Default is 1 and 0. Binding score is calculated by the count ratio of proximal site and binding site. The cutoff values are used to decrease the total number of binding site for ranking. Increasing the 'log2FCcutoff' value and decreasing the P-value cutoff value can greatly decrease the memory cost and computing time by decreasing the total binding sites.

Value

A TFEAresults object.

Author(s)

Jianhong Ou

Examples

```
bamExp <- system.file("extdata",
                      c("KD.shift.rep1.bam",
                        "KD.shift.rep2.bam"),
                      package="ATACseqTFEA")
bamCtl <- system.file("extdata",
                      c("WT.shift.rep1.bam",
                        "WT.shift.rep2.bam"),
                      package="ATACseqTFEA")
bsl <- system.file("extdata", "bindingSites.rds",
                   package="ATACseqTFEA")
bindingSites <- readRDS(bsl)
res <- TFEA(bamExp, bamCtl, bindingSites=bindingSites,
            positive=0, negative=0)
res
```

TFEAresults-class *Class* "TFEAresults"

Description

An object of class "TFEAresults" represents the results of TFEA.

Usage

```
TFEAresults(...)
```

Arguments

`...` Each argument in ... becomes an slot in the new "TFEAresults"-class.

Value

A TFEAresults object.

Slots

`enrichmentScore` `"numeric Matrix"`, specify the enrichment score for each transcription factor (TF). Every row represents a TF. The columns represents the accumulated enrichment score for that rank.

`bindingSites` `GenomicRanges` object. It is keep same length and order as the columns in enrichmentScore.

`motifID` `"list"`. The ranks of binding sites for each TF.

`resultsTable` `"data.frame"`. The data frame contains the summarized enrichment score, the p-value, and adjuct p-value for each TF.

Examples

```
res <- readRDS(system.file("extdata", "res.rds", package="ATACseqTFEA"))
res
```

Index