Package 'ATACseqQC'

November 11, 2026

Type Package

Title ATAC-seq Quality Control

Version 1.36.1

Author Jianhong Ou, Haibo Liu, Feng Yan, Jun Yu, Michelle Kelliher, Lucio Castilla, Chao Deng, Nathan Lawson, Lihua Julie Zhu

Maintainer Jianhong Ou <jou@morgridge.org>

Description ATAC-seq, an assay for Transposase-Accessible Chromatin using sequencing, is a rapid and sensitive method for chromatin accessibility analysis. It was developed as an alternative method to MNase-seq, FAIRE-seq and DNAse-seq. Comparing to the other methods, ATAC-seq requires less amount of the biological samples and time to process. In the process of analyzing several ATAC-seq dataset produced in our labs, we learned some of the unique aspects of the quality assessment for ATAC-seq data.To help users to quickly assess whether their ATAC-seq experiment is successful, we developed ATACseqQC package partially following the guideline published in Nature Method 2013 (Greenleaf et al.), including diagnostic plot of fragment size distribution, proportion of mitochondria reads, nucleosome positioning pattern, and CTCF or other Transcript Factor footprints.

Depends R (>= 3.4), BiocGenerics, S4Vectors

Imports BSgenome, Biostrings, ChIPpeakAnno, IRanges, GenomicRanges, GenomicAlignments, GenomeInfoDb, GenomicScores, graphics, grid, limma, polynom, Rsamtools (>= 1.31.2), randomForest, rtracklayer, stats, motifStack, utils, KernSmooth, edgeR, BiocParallel

Suggests BiocStyle, knitr, BSgenome.Hsapiens.UCSC.hg19, TxDb.Hsapiens.UCSC.hg19.knownGene, phastCons100way.UCSC.hg19, MotifDb, trackViewer, testthat, rmarkdown

License GPL (>= 2)

LazyData TRUE

VignetteBuilder knitr

RoxygenNote 8.0.0

biocViews Sequencing, DNASeq, ATACSeq, GeneRegulation, QualityControl, Coverage, NucleosomePositioning, ImmunoOncology

Encoding UTF-8

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

git_branch RELEASE_3_23

git_last_commit e2bc93a

git_last_commit_date 2026-07-13

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

ATACseqQC-package . . . 3
bamQC . . . 3
DB . . . 4
distanceDyad . . . 4
enrichedFragments . . . 5
estimateLibComplexity . . . 7
factorFootprints . . . 8
footprintsScanner . . . 10
fragSizeDist . . . 12
NFRscore . . . 13
peakdet . . . 14
plotCorrelation . . . 15
plotFootprints . . . 16
pseudoPausingIndex . . . 17
PTscore . . . 19
pwmscores . . . 20
readBamFile . . . 20
readsDupFreq . . . 21
saturationPlot . . . 22
shiftGAlignments . . . 24
shiftGAlignmentsList . . . 25
shiftReads . . . 26
splitBam . . . 27
splitGAlignmentsByCut . . . 28
TSSEscore . . . 30
vPlot . . . 32
writeListOfGAlignments . . . 33

Index **35**

ATACseqQC-package *ATAC-seq Quality Control*

Description

ATAC-seq, an assay for Transposase-Accessible Chromatin using sequencing, is a rapid and sensitive method for chromatin accessibility analysis. It was developed as an alternative method to MNase-seq, FAIRE-seq and DNAse-seq. Comparing to the other methods, ATAC-seq requires less amount of the biological samples and time to process. In the process of analyzing several ATAC-seq dataset produced in our labs, we learned some of the unique aspects of the quality assessment for ATAC-seq data.To help users to quickly assess whether their ATAC-seq experiment is successful, we developed ATACseqQC package partially following the guideline published in Nature Method 2013 (Greenleaf et al.), including diagnostic plot of fragment size distribution, proportion of mitochondria reads, nucleosome positioning pattern, and CTCF or other Transcript Factor footprints.

bamQC *Mapping quality control*

Description

Check the mapping rate, PCR duplication rate, and mitochondria reads contamination.

Usage

```
bamQC(
  bamfile,
  index = bamfile,
  mitochondria = "chrM",
  outPath = sub(".bam", ".clean.bam", basename(bamfile)),
  doubleCheckDup = FALSE
)
```

Arguments

bamfile	character(1). File name of bam.
index	character(1). File name of index file.
mitochondria	character(1). Sequence name of mitochondria.
outPath	character(1). File name of cleaned bam.
doubleCheckDup	logical(1). Double check duplicates or not if there is no tags for that.

Value

A list of quality summary.

Author(s)

Jianhong Ou

Examples

```
bamfile <- system.file("extdata", "GL1.bam", package="ATACseqQC")
bamQC(bamfile, outPath=NULL)
```

DB *helper function for differential binding*

Description

helper function for differential binding

Usage

```
DB(counts, libSize, group, default.bcv = 0.3)
```

Arguments

counts	count table
libSize	library size
group	group design
default.bcv	a reasonable dispersion value

Value

topTable

---

distanceDyad *Distance of potential nucleosome dyad*

Description

Calculate the distance of potential nucleosome dyad and the linear model for V.

Usage

```
distanceDyad(vPlotOut, fragLenRanges = c(60, 180, 250), draw = TRUE, ...)
```

Arguments

vPlotOut	The output of vPlot.
fragLenRanges	A numeric vector (length=3) for fragment size of nucleosome free and mono-nucleosome. Default c(60, 180, 250).
draw	Plot the results or not. Default TRUE.
...	Prameters could be passed to plot.

Value

an invisible list with distance of nucleosome and the linear model.

Author(s)

Jianhong Ou

See Also

vPlot

Examples

```
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC")
library(MotifDb)
CTCF <- query(MotifDb, c("CTCF"))
CTCF <- as.list(CTCF)
library(BSgenome.Hsapiens.UCSC.hg19)
vp <- vPlot(bamfile, pfm=CTCF[[1]],
     genome=Hsapiens,
     min.score="95%", seqlev="chr1",
     draw=FALSE)
distanceDyad(vp)
```

---

enrichedFragments *enrichment for nucleosome-free fragments and nucleosome signals*

Description

Get the enrichment signals for nucleosome-free fagments and nucleosomes.

Usage

```
enrichedFragments(
  bamfiles,
  index = bamfiles,
  gal,
  TSS,
  librarySize,
  upstream = 1010L,
  downstream = 1010L,
  n.tile = 101L,
  normal.method = "quantile",
  adjustFragmentLength = 80L,
  TSS.filter = 0.5,
  seqlev = paste0("chr", c(1:22, "X", "Y"))
)
```

Arguments

bamfiles	A vector of characters indicates the file names of bams.
index	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
gal	A GAlignmentsList object or a list of GAlignmentPairs. If bamfiles is missing, gal is required.
TSS	an object of GRanges indicates the transcript start sites. All the width of TSS should equal to 1. Otherwise, TSS will be reset to the center of input TSS.
librarySize	A vector of numeric indicates the library size. Output of estLibSize
upstream, downstream	numeric(1) or integer(1). Upstream and downstream size from each TSS.
n.tile	numeric(1) or integer(1). The number of tiles to generate for each element of TSS.
normal.method	character(1). Normalization methods, could be "none" or "quantile". See normalizeBetweenArrays.
adjustFragmentLength	numeric(1) or integer(1). The size of fragment to be adjusted to. Default is set to half of the nucleosome size (80)
TSS.filter	numeric(1). The filter for signal strength of each TSS. Default 0.5 indicates the average signal strength for the TSS from upstream to downstream bins should be greater than 0.5.
seqlev	A vector of character indicates the sequence names to be considered.

Value

A list of matrixes. In each matrix, each row record the signals for corresponding feature.

Author(s)

Jianhong Ou

Examples

```
bamfiles <- system.file("extdata", "splited",
                        c("NucleosomeFree.bam",
                          "mononucleosome.bam",
                          "dinucleosome.bam",
                          "trinucleosome.bam"), package="ATACseqQC")
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
TSS <- promoters(txs, upstream=0, downstream=1)
library(ChIPpeakAnno)
librarySize <- estLibSize(bamfiles)
sigs <- enrichedFragments(bamfiles, TSS=TSS, librarySize=librarySize,
                          seqlev="chr1", TSS.filter=0)
sigs.log2 <- lapply(sigs, function(.ele) log2(.ele+1))
featureAlignedHeatmap(sigs.log2, reCenterPeaks(TSS, width=2020),
                      zeroAt=.5, n.tile=101, upper.extreme=2)
featureAlignedDistribution(sigs, reCenterPeaks(TSS, width=2020),
                           zeroAt=.5, n.tile=101, type="l")
```

---

estimateLibComplexity *Library complexity estimation*

Description

Estimating the library complexity.

Usage

```
estimateLibComplexity(
  histFile,
  times = 100,
  interpolate.sample.sizes = seq(0.1, 1, by = 0.1),
  extrapolate.sample.sizes = seq(5, 20, by = 5)
)
```

Arguments

histFile	A two-column matrix of integers. The 1st column is the frequency j = 1,2,3,.... The 2nd column is the number of genomic regions with the same fequency (j) of duplication. This file should be sorted by the first column in ascending order. For example, one row of a histogram file: 10 20 means there are 10 genomic regions, each of which is covered by 20 identical fragments at a given sequencing depth of a sequencing library.
times	An positive integer representing the minimum required number of successful estimation. Default is 100.
interpolate.sample.sizes	A numeric vector with values between (0, 1].

`extrapolate.sample.sizes`
: A numeric vector with values greater than 1.

Value

invisible estimates, a data frame of 3 columns: relative sequence depth, number of distinct fragments, number of putative sequenced reads.

Author(s)

Haibo Liu, Feng Yan, Chao Deng

See Also

readsDupFreq

Examples

```
FisherButterfly <- readRDS(system.file('extdata', 'FisherButterfly.rds',
                           package='ATACseqQC'))
estimateLibComplexity(histFile=FisherButterfly, times=100)
```

---

factorFootprints *plot ATAC-seq footprints infer factor occupancy genome wide*

Description

Aggregate ATAC-seq footprint for a given motif generated over binding sites within the genome.

Usage

```
factorFootprints(
  bamfiles,
  index = bamfiles,
  pfm,
  genome,
  min.score = "95%",
  bindingSites,
  seqlev = paste0("chr", c(1:22, "X", "Y")),
  upstream = 100,
  downstream = 100,
  maxSiteNum = 1e+06,
  anchor = "cut site",
  group = "strand",
  ...
)
```

Arguments

bamfiles	A vector of characters indicates the file names of bams. All the bamfiles will be pulled together.
index	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
pfm	A Position frequency Matrix represented as a numeric matrix with row names A, C, G and T.
genome	An object of BSgenome.
min.score	The minimum score for counting a match. Can be given as a character string containing a percentage (e.g. "95 score or as a single number. See matchPWM.
bindingSites	A object of GRanges indicates candidate binding sites (eg. the output of fimo). The GRanges object must have score column in the metadata column.
seqlev	A vector of characters indicates the sequence levels.
upstream, downstream	numeric(1) or integer(1). Upstream and downstream of the binding region for aggregate ATAC-seq footprint.
maxSiteNum	numeric(1). Maximal number of predicted binding sites. if predicted binding sites is more than this number, top maxSiteNum binding sites will be used.
anchor	"cut site" or "fragment center". If "fragment center" is used, the input bamfiles must be paired-end.
group	Group information for the bamfiles. Default by strand. Otherwise, a factor or vector of characters with same length of bamfiles. The group is limited to 2 groups.
...	xlab, legTitle, xlim or ylim could be used by plotFootprints

Value

an invisible list of matrixes with the signals for plot. It includes: - signal mean values of coverage for positive strand and negative strand in feature regions - spearman.correlation spearman correlations of cleavage counts in the highest 10-nucleotide-window and binding prediction score. - bindingSites predicted binding sites.

Author(s)

Jianhong Ou, Julie Zhu

References

Chen, K., Xi, Y., Pan, X., Li, Z., Kaestner, K., Tyler, J., Dent, S., He, X. and Li, W., 2013. DANPOS: dynamic analysis of nucleosome position and occupancy by sequencing. Genome research, 23(2), pp.341-351.

Examples

```
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC")
library(MotifDb)
CTCF <- query(MotifDb, c("CTCF"))
CTCF <- as.list(CTCF)
library(BSgenome.Hsapiens.UCSC.hg19)
factorFootprints(bamfile, pfm=CTCF[[1]],
                 genome=Hsapiens,
                 min.score="95%", seqlev="chr1",
                 upstream=100, downstream=100)
##### Using user defined binding sites #####
bds <- readRDS(system.file("extdata", "jolma2013.motifs.bindingList.95.rds",
                           package="ATACseqQC"))
bindingSites <- bds[["Hsapiens-jolma2013-CTCF"]]
seqlev <- "chr1" #seqlevels(bindingSites)
ff <- factorFootprints(bamfile, pfm=CTCF[[1]],
                 bindingSites=bindingSites,
                 seqlev=seqlev,
                 upstream=100, downstream=100)
##### normalize the plot by distal signals #####
library(motifStack)
Profile <- lapply(ff$signal, function(.ele) colMeans(.ele, na.rm = TRUE))
pfm <- new('pfm', mat=as.matrix(CTCF[[1]]), name='CTCF')
plotFootprints(Profile=c(Profile[["+"]], Profile[["-"]]),
               Mlen=ff$Mlen,
               motif=pfm,
               segmentation=ff$Profile.segmentation,
               reNormalizeByDistalSig=TRUE)
```

`footprintsScanner` *scan ATAC-seq footprints infer factor occupancy genome wide*

Description

Aggregate ATAC-seq footprint for a bunch of motifs generated over binding sites within the genome.

Usage

```
footprintsScanner(
  bamExp,
  bamCtl,
  indexExp = bamExp,
  indexCtl = bamCtl,
  bindingSitesList,
  seqlev = paste0("chr", c(1:25, "X", "Y")),
  proximal = 40L,
  distal = proximal,
```

```
  gap = 10L,
  maximalBindingWidth = NA,
  cutoffLogFC = log2(1.5),
  cutoffPValue = 0.05,
  correlatedFactorCutoff = 3/4
)

prepareBindingSitesList(
  pfms,
  genome,
  seqlev = paste0("chr", c(1:22, "X", "Y")),
  expSiteNum = 5000
)
```

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.
bindingSitesList	A object of GRangesList indicates candidate binding sites (eg. the output of fimo).
seqlev	A vector of characters indicates the sequence levels.
proximal, distal	numeric(1) or integer(1). basepair for open region from binding sites (proximal) and extented region for background (distal) of the binding region for aggregate ATAC-seq footprint.
gap	numeric(1) or integer(1). basepair for gaps among binding sites, proximal, and distal. default is 5L.
maximalBindingWidth	numeric(1) or integer(1). Maximal binding sites width for all the motifs. If setted, all motif binding sites will be re-sized to this value.
cutoffLogFC, cutoffPValue	numeric(1). Cutoff value for differential bindings.
correlatedFactorCutoff	numeric(1). Cutoff value for correlated factors. If the overlapping binding site within 100bp is more than cutoff, the TFs will be treated as correlated factors.
pfms	A list of Position frequency Matrix represented as a numeric matrix with row names A, C, G and T.
genome	An object of BSgenome.
expSiteNum	numeric(1). Expect number of predicted binding sites. if predicted binding sites is more than this number, top expSiteNum binding sites will be used.

Value

a list. It includes: - bindingSites GRanges of binding site with hits of reads - data a list with test result for each binding site - results a data.frame with open score and enrichment score of motifs

Author(s)

Jianhong Ou

Examples

```
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC")
bsl <- system.file("extdata", "jolma2013.motifs.bindingList.95.rds",
                   package="ATACseqQC")
bindingSitesList <- readRDS(bsl)
footprintsScanner(bamfile, seqlev="chr1", bindingSitesList=bindingSitesList)

library(MotifDb)
motifs <- query(MotifDb, c("Hsapiens"))
motifs <- as.list(motifs)
library(BSgenome.Hsapiens.UCSC.hg19)
#bindingSitesList <- prepareBindingSitesList(motifs, genome=Hsapiens)
```

---

fragSizeDist *fragment size distribution*

Description

estimate the fragment size of bams

Usage

```
fragSizeDist(
  bamFiles,
  bamFiles.labels,
  index = bamFiles,
  ylim = NULL,
  logYlim = NULL
)
```

Arguments

bamFiles	A vector of characters indicates the file names of bams.
bamFiles.labels	labels of the bam files, used for pdf file naming.
index	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
ylim	numeric(2). ylim of the histogram.
logYlim	numeric(2). ylim of log-transformed histogram for the insert.

Value

Invisible fragment length distribution list.

Author(s)

Jianhong Ou

Examples

```
bamFiles <- dir(system.file("extdata", package="ATACseqQC"), "GL.*.bam$", full.names=TRUE)
bamFiles.labels <- sub(".bam", "", basename(bamFiles))
fragSizeDist(bamFiles, bamFiles.labels)
```

NFRscore *Nucleosome Free Regions (NFR) score*

Description

NFR score is a raio between cut signal adjacent to TSS and that flanking the corresponding TSS. Each TSS window of 400 bp is first seperated into 3 sub-regions: the most upstream 150 bp (n1), the most downstream of 150 bp (n2), and the middle 100 bp (nf). Then the number of fragments with 5' ends overlapping each region are calculated for each TSS. The NFR score for each TSS is calculated as NFR-score = log2(nf) - log2((n1+n2)/2). A plot can be generate with the NFR scores as Y-axis and the average signals of 400 bp window as X-axis, very like a MA plot for gene expression data.

Usage

```
NFRscore(
  obj,
  txs,
  seqlev = intersect(seqlevels(obj), seqlevels(txs)),
  nucleosomeSize = 150,
  nucleosomeFreeSize = 100
)
```

Arguments

obj	an object of GAlignments
txs	GRanges of transcripts
seqlev	A vector of characters indicates the sequence levels.
nucleosomeSize	numeric(1) or integer(1). Default is 150
nucleosomeFreeSize	numeric(1) or integer(1). Default is 100

Value

A object of GRanges with NFR scores

Author(s)

Jianhong Ou

Examples

```
library(GenomicRanges)
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC", mustWork=TRUE)
gal1 <- readBamFile(bamFile=bamfile, tag=character(0),
                    which=GRanges("chr1", IRanges(1, 1e6)),
                    asMates=FALSE)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
nfr <- NFRscore(gal1, txs)
```

---

peakdet *Detect peak positions*

Description

Detect the peaks positions and valley positions. The algorithm is modified from github::dgromer/peakdet

Usage

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

Arguments

y	A vector of numeric where to search peaks
delta	A numeric of length 1, defining the local threshold for peak detection. 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 vectors of numeric which indicate the positions of peak or valley.

plotCorrelation *plot Correlations of multiple samples*

Description

plot PCA or heatmap for multiple bamfiles. The correlation is calculated by the counts in promoter regions.

Usage

```
plotCorrelation(
  objs,
  txs,
  seqlev = intersect(seqlevels(objs[[1]]), seqlevels(txs)),
  upstream = 2000,
  downstream = 500,
  type = c("heatmap", "PCA"),
  ...
)
```

Arguments

objs	an object of GAlignmentsList
txs	GRanges of transcripts
seqlev	A vector of characters indicates the sequence levels.
upstream	numeric(1) or integer(1). Start position of promoter. Default is 2000
downstream	numeric(1) or integer(1). End position of promoter. Default is 500
type	Figure type, heatmap or PCA plot.
...	parameters could be passed to downstream functions such as plot for pca or heatmap for heatmap.

Details

The correlation will be calculated by the correlation of insertion sites within promoter regions. Even the sequencing is paired-end, please treat it as single ends.

Value

A invisible object of GRanges with counts

Author(s)

Jianhong Ou

Examples

```
library(GenomicRanges)
library(GenomicAlignments)
path <- system.file("extdata", package="ATACseqQC", mustWork=TRUE)
bamfiles <- dir(path, "*.bam$", full.name=TRUE)
gals <- lapply(bamfiles, function(bamfile){
               readBamFile(bamFile=bamfile, tag=character(0),
                           which=GRanges("chr1", IRanges(1, 1e6)),
                           asMates=FALSE)
       })
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
plotCorrelation(GAlignmentsList(gals), txs, seqlev="chr1")
```

---

plotFootprints *Plots a footprint estimated by Centipede*

Description

Visualizing the footprint profile

Usage

```
plotFootprints(
  Profile,
  Mlen = 0,
  xlab = "Dist. to motif (bp)",
  ylab = "Cut-site probability",
  legLabels = c("For. strand", "Rev. strand"),
  legTitle,
  xlim,
  ylim,
  newpage = TRUE,
  motif,
  segmentation,
  reNormalizeByDistalSig = FALSE,
  ...
)
```

Arguments

Profile	A vector with the profile estimated by CENTIPEDE
Mlen	Length of the motif for drawing vertical lines delimiting it
xlab	Label of the x axis
ylab	Label for the y axis
legLabels	Labels for legend.

legTitle	Title for one of the plot corners
xlim	xlim
ylim	ylim
newpage	Plot the figure in a new page?
motif	a pfm object.
segmentation	the segmentation position and abundance
reNormalizeByDistalSig	Re-normalized the curver by distal signals.
...	Not used.

Value

Null.

Author(s)

Jianhong Ou

Examples

```
library(MotifDb)
CTCF <- query(MotifDb, c("CTCF"))
CTCF <- as.list(CTCF)
motif <- new("pfm", mat=CTCF[[1]], name="CTCF")
ATACseqQC:::plotFootprints(Profile=sample.int(500),
                        Mlen=ncol(CTCF[[1]]), motif=motif)
```

---

pseudoPausingIndex *Simulation pausing index*

Description

The polII pausing index is the ratio of the reads density at the 5' end of the gene to that in the gene body. This function will simulate the pausing index by open chromatin coverage instead of PolII signaling. The pausing index is a raio between aggregate distribution of reads in TSS and that elongating gene bodys. The default PI = [average coverage of TSS (+1 to +200bp) - average coverage of avoidance region (+21 to +60bp)] / the average coverage of in transcripts (+401bp to 600bp).

Usage

```
pseudoPausingIndex(
  obj,
  txs,
  seqlev = intersect(seqlevels(obj), seqlevels(txs)),
  nascentRegion = c(-200, -1),
```

```
  pausedRegion = c(1, 200),
  avoidanceRegion = c(21, 60),
  elongationRegion = c(401, 600),
  pseudocount = 1L
)
```

Arguments

obj	an object of GAlignments
txs	GRanges of transcripts
seqlev	A vector of characters indicates the sequence levels.
nascentRegion, pausedRegion, avoidanceRegion, elongationRegion	numeric(2) or integer(2). The start and end position of the pre-initiation complex, paused complex, paused complex avoidance region and productive elongation.
pseudocount	numeric(1) or integer(1). Pseudocount. Default is 1.

Value

A object of GRanges with pseudo pausing index.

Author(s)

Jianhong Ou

References

https://doi.org/10.1098 https://www.nature.com/articles/nmeth.2688/figures/3

Examples

```
library(GenomicRanges)
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC", mustWork=TRUE)
gal1 <- readBamFile(bamFile=bamfile, tag=character(0),
                    which=GRanges("chr1", IRanges(1, 1e6)),
                    asMates=FALSE)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
ppi <- pseudoPausingIndex(gal1, txs)
```

PTscore *Promoter/Transcript body (PT) score*

Description

PT score is calculated for coverage of promoter divided by the coverage of transcripts body. PT score will show if the signal is enriched in promoters.

Usage

```
PTscore(
  obj,
  txs,
  seqlev = intersect(seqlevels(obj), seqlevels(txs)),
  upstream = 2000,
  downstream = 500
)
```

Arguments

obj	an object of GAlignments
txs	GRanges of transcripts
seqlev	A vector of characters indicates the sequence levels.
upstream	numeric(1) or integer(1). Start position of promoter. Default is 2000
downstream	numeric(1) or integer(1). End position of promoter. Default is 500

Value

A object of GRanges with PT scores

Author(s)

Jianhong Ou

Examples

```
library(GenomicRanges)
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC", mustWork=TRUE)
gal1 <- readBamFile(bamFile=bamfile, tag=character(0),
                    which=GRanges("chr1", IRanges(1, 1e6)),
                    asMates=FALSE)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
pt <- PTscore(gal1, txs)
```

pwmscores *max PWM scores for sequences*

Description

calculate the maximal PWM scores for each given sequences

Usage

```
pwmscores(pwm, subject)
```

Arguments

pwm	A Position Weight Matrix represented as a numeric matrix with row names A, C, G and T.
subject	Typically a DNAString object. A Views object on a DNAString subject, a MaskedDNAString object, or a single character string, are also supported. IUPAC ambiguity letters in subject are ignored (i.e. assigned weight 0) with a warning.

Value

a numeric vector

Author(s)

Jianhong

readBamFile *read in bam files*

Description

wraper for readGAlignments/readGAlignmentsList to read in bam files.

Usage

```
readBamFile(
  bamFile,
  which,
  tag = character(0),
  what = c("qname", "flag", "mapq", "isize", "seq", "qual", "mrnm"),
  flag = scanBamFlag(isSecondaryAlignment = FALSE, isUnmappedQuery = FALSE,
    isNotPassingQualityControls = FALSE, isSupplementaryAlignment = FALSE),
  asMates = FALSE,
  bigFile = FALSE,
  ...
)
```

Arguments

bamFile	character(1). Bam file name.
which	A GRanges, IntegerRangesList, or any object that can be coerced to a RangesList, or missing object, from which a IRangesList instance will be constructed. See ScanBamParam.
tag	A vector of characters indicates the tag names to be read. See ScanBamParam.
what	A character vector naming the fields to return. Fields are described on the scanBam help page.
flag	An integer(2) vector used to filter reads based on their 'flag' entry.
asMates	logical(1). Paired ends or not
bigFile	If the file take too much memory, set it to true to avoid read the reads into memory. scanBamFlag helper function.
...	parameters used by readGAlignmentsList or readGAlignments

Value

A GAlignmentsList object when asMates=TRUE, otherwise A GAlignments object. If bigFile is set to TRUE, no reads will be read into memory at this step and empty GAlignments/GAlignmentsList will be returned.

Author(s)

Jianhong Ou

Examples

```
library(BSgenome.Hsapiens.UCSC.hg19)
which <- as(seqinfo(Hsapiens)["chr1"], "GRanges")
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC", mustWork=TRUE)
readBamFile(bamfile, which=which, asMates=TRUE)
```

---

readsDupFreq *Calculating duplication frequency*

Description

Calculating the frequency of read duplication based on alignment status determined by rname, strand, pos, cigar, mrnm, mpos and isize.

Usage

```
readsDupFreq(bamFile, index = bamFile)
```

Arguments

bamFile	A character vector of length 1L containing the name of a BAM file. Only a BAM file with duplication reads are meaningful for estimating the library complexity. For example, a raw BAM file output by aligners, or a BAM file with mitochondrial reads removed.
index	A character vector of length 1L containing the name of a BAM index file.

Value

A two-column matrix of integers. The 1st column is the frequency j = 1,2,3,.... The 2nd column is the number of genomic regions with the same fequency (j) of duplication. The frequency column is in ascending order.

Author(s)

Haibo Liu

Examples

```
bamFile <- system.file("extdata", "GL1.bam", package = "ATACseqQC")
freq <- readsDupFreq(bamFile)
```

---

saturationPlot *Plotting Saturation curves*

Description

Plotting the saturation curves.

Usage

```
saturationPlot(
  subsamplingPeakFiles,
  subsamplingSizes,
  sep = "\t",
  header = FALSE,
  fdr = 0.05,
  fdrCol = 9,
  startCol = 2,
  endCol = 3,
  skipLines = 1,
  peakCaller = "MACS2",
  outPrefix,
  span = 2,
  degree = 2
)
```

Arguments

subsamplingPeakFiles	A character vector containing peak files from peak calling tools, such as MACS2. Currently only MACS2 output is supported.
subsamplingSizes	A named vector of integers, which are the sizes of subsamples for peak calling. The names of subsamplingPeakFiles should be identical to the basenames of subsamplingPeakFiles.
sep	A character vector of length 1L, which is the column separator used in peak files.
header	A boolean (TRUE or FASLE) vector of length 1L, showing whether there are column headers in the peak files.
fdr	A decimal between 0 and 1, a cutoff of statistical significance of peak detection.
fdrCol	An integer, column index for fdr.
startCol	An integer, column index for start positions of peak regions.
endCol	An integer, column index for end positions of peak regions.
skipLines	An integer, the number of lines (comments or instruction) to skip when peak files are read into R.
peakCaller	A character vector of length 1L contatining the name of the peak caller used to generate the peak files, such as "MACS2". Currently only MACS2 output (XXX.narrowPeak or XXX.broadPeak) is support.
outPrefix	A character vector of length 1L, the file prefix for outputting saturation plots.
span	An integer, the span parameter for loess smoothing to fit a smoothed saturation curve.
degree	An integer, the degree of local polynomial used for loess.

Value

A data frame of three columns: subsamplingSizes, the number of subsampled fragments; numPeaks, the number of peaks with fdr less than a given threshold when a given number of fragmetns are subsampled; breadth, the total breadth of peaks with fdr less than a given threshold for give subsampling when a given number of fragmetns are subsampled.

Author(s)

Haibo Liu

Examples

```
if(interactive()){
}
```

`shiftGAlignments` *shift 5' ends for single end reads*

Description

shift the GAlignmentsLists by 5' ends. All reads aligning to the positive strand will be offset by +4bp, and all reads aligning to the negative strand will be offset -5bp by default.

Usage

```
shiftGAlignments(gal, positive = 4L, negative = 5L, outbam)
```

Arguments

`gal`	An object of GAlignments.
`positive`	integer(1). the size to be shift for positive strand
`negative`	integer(1). the size to be shift for negative strand
`outbam`	file path to save shift reads. If missing, no file will be write.

Value

An object of GAlignments with 5' end shifted reads.

Author(s)

Jianhong Ou

Examples

```
bamfile <- system.file("extdata", "GL1.bam", package="ATACseqQC")
tags <- c("AS", "XN", "XM", "XO", "XG", "NM", "MD", "YS", "YT")
library(BSgenome.Hsapiens.UCSC.hg19)
which <- as(seqinfo(Hsapiens)["chr1"], "GRanges")
gal <- readBamFile(bamfile, tag=tags,
                   what=c("qname", "flag", "mapq", "seq", "qual"),
                   which=which, asMates=FALSE, bigFile=TRUE)
objs <- shiftGAlignments(gal)
export(objs, "shift.bam")
```

shiftGAlignmentsList *shift 5' ends*

Description

shift the GAlignmentsLists by 5' ends. All reads aligning to the positive strand will be offset by +4bp, and all reads aligning to the negative strand will be offset -5bp by default.

Usage

```
shiftGAlignmentsList(
  gal,
  positive = 4L,
  negative = 5L,
  outbam,
  BPPARAM = NULL,
  slidingWindowSize = 5e+07
)
```

Arguments

gal	An object of GAlignmentsList.
positive	integer(1). the size to be shift for positive strand
negative	integer(1). the size to be shift for negative strand
outbam	file path to save shift reads. If missing, no file will be write.
BPPARAM	The parallel parameters used by BiocParrallel.
slidingWindowSize	The width of each tile when the input is big file. By default 50e6, the memory cost will be about 6GB for each thread for a 5GB bam file. Increase the value will increase the memory cost but may speed up the process.

Value

An object of GAlignments with 5' end shifted reads. The PCR duplicated will be removed unless there is metadata keepDuplicates set to TRUE.

Author(s)

Jianhong Ou

Examples

```
bamfile <- system.file("extdata", "GL1.bam", package="ATACseqQC")
tags <- c("AS", "XN", "XM", "XO", "XG", "NM", "MD", "YS", "YT")
library(BSgenome.Hsapiens.UCSC.hg19)
which <- as(seqinfo(Hsapiens)["chr1"], "GRanges")
```

```
gal <- readBamFile(bamfile, tag=tags, which=which, asMates=TRUE)
objs <- shiftGAlignmentsList(gal)
export(objs, "shift.bam")
## Not run:
  bamfile <- 'a.big.file.bam'
  tags <- c("NM", "MD")
  which <- GRanges(c('chr1:1-249250621:*', 'chr2:1-243199373:*'))
  gal <- readBamFile(bamfile, tag=tags, which=which,
  asMates=TRUE, bigFile=TRUE)
  library(BiocParallel)
  BPPARAM <- MulticoreParam(workers = 2, progress=TRUE)
  objs <- shiftGAlignmentsList(gal, BPPARAM = BPPARAM,
   outbam="shift.bam")

## End(Not run)
```

`shiftReads` *shift read for 5'end*

Description

shift reads for 5'ends

Usage

```
shiftReads(x, positive = 4L, negative = 5L)
```

Arguments

`x`	an object of GAlignments
`positive`	integer(1). the size to be shift for positive strand
`negative`	integer(1). the size to be shift for negative strand

Value

an object of GAlignments

Author(s)

Jianhong Ou

splitBam *prepare bam files for downstream analysis*

Description

shift the bam files by 5'ends and split the bam files.

Usage

```
splitBam(
  bamfile,
  tags,
  index = bamfile,
  outPath = NULL,
  txs,
  genome,
  conservation,
  positive = 4L,
  negative = 5L,
  breaks = c(0, 100, 180, 247, 315, 473, 558, 615, Inf),
 labels = c("NucleosomeFree", "inter1", "mononucleosome", "inter2", "dinucleosome",
    "inter3", "trinucleosome", "others"),
  seqlev = paste0("chr", c(1:22, "X", "Y")),
  cutoff = 0.8,
  flag = scanBamFlag(isSecondaryAlignment = FALSE, isUnmappedQuery = FALSE,
    isNotPassingQualityControls = FALSE, isSupplementaryAlignment = FALSE)
)
```

Arguments

bamfile	character(1). File name of bam.
tags	A vector of characters indicates the tags in bam file.
index	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
outPath	Output file path.
txs	GRanges of transcripts.
genome	An object of BSgenome
conservation	An object of GScores.
positive	integer(1). the size to be shift for positive strand
negative	integer(1). the size to be shift for negative strand
breaks	A numeric vector for fragment size of nucleosome free, mononucleosome, dinucleosome and trinucleosome
labels	A vector of characters indicates the labels for the levels of the resulting category. The length of labels = length of breaks - 1

seqlev	A vector of characters indicates the sequence levels.
cutoff	numeric(1). Cutoff value for prediction by randomForest.
flag	An integer(2) vector used to filter reads based on their 'flag' entry.

Value

an invisible list of GAlignments

Author(s)

Jianhong Ou

See Also

shiftGAlignmentsList, splitGAlignmentsByCut, and writeListOfGAlignments

Examples

```
if(Sys.getenv("USER")=="jianhongou"){
bamfile <- system.file("extdata", "GL1.bam", package="ATACseqQC")
tags <- c("AS", "XN", "XM", "XO", "XG", "NM", "MD", "YS", "YT")
library(BSgenome.Hsapiens.UCSC.hg19)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(phastCons100way.UCSC.hg19)
objs <- splitBam(bamfile, tags,
                 txs=txs, genome=Hsapiens,
                 conservation=phastCons100way.UCSC.hg19,
                 seqlev="chr1")
}
```

---

splitGAlignmentsByCut *split bams into nucleosome free, mononucleosome, dinucleosome and trinucleosome*

Description

use random forest to split the reads into nucleosome free, mononucleosome, dinucleosome and trinucleosome. The features used in random forest including fragment length, GC content, and UCSC phastCons conservation scores.

Usage

```
splitGAlignmentsByCut(
  obj,
  txs,
  genome,
  conservation,
```

```
  outPath,
  breaks = c(0, 100, 180, 247, 315, 473, 558, 615, Inf),
 labels = c("NucleosomeFree", "inter1", "mononucleosome", "inter2", "dinucleosome",
    "inter3", "trinucleosome", "others"),
  labelsOfNucleosomeFree = "NucleosomeFree",
  labelsOfMononucleosome = "mononucleosome",
  trainningSetPercentage = 0.15,
  cutoff = 0.8,
  halfSizeOfNucleosome = 80L,
  summaryFun = mean
)
```

Arguments

`obj`	an object of GAlignments
`txs`	GRanges of transcripts
`genome`	an object of BSgenome
`conservation`	an object of GScores.
`outPath`	folder to save the splitted alignments. If outPath is setting, the return of the function will not contain seq and qual fields.
`breaks`	a numeric vector for fragment size of nucleosome free, mononucleosome, dinucleosome and trinucleosome. The breaks pre-defined here is following the description of Greenleaf's paper (see reference).
`labels`	a character vector for labels of the levels of the resulting category.
`labelsOfNucleosomeFree`, `labelsOfMononucleosome`	character(1). The label for nucleosome free and mononucleosome.
`trainningSetPercentage`	numeric(1) between 0 and 1. Percentage of trainning set from top coverage.
`cutoff`	numeric(1) between 0 and 1. cutoff value for prediction.
`halfSizeOfNucleosome`	numeric(1) or integer(1). Thre read length will be adjusted to half of the nucleosome size to enhance the signal-to-noise ratio.
`summaryFun`	Function to summarize genomic scores when more than one position is retrieved. This will greatly affect the CPU time.

Value

a list of GAlignments

Author(s)

Jianhong Ou

References

Buenrostro, J.D., Giresi, P.G., Zaba, L.C., Chang, H.Y. and Greenleaf, W.J., 2013. Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. Nature methods, 10(12), pp.1213-1218.

Chen, K., Xi, Y., Pan, X., Li, Z., Kaestner, K., Tyler, J., Dent, S., He, X. and Li, W., 2013. DANPOS: dynamic analysis of nucleosome position and occupancy by sequencing. Genome research, 23(2), pp.341-351.

Examples

```
library(GenomicRanges)
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC", mustWork=TRUE)
tags <- c("AS", "XN", "XM", "XO", "XG", "NM", "MD", "YS", "YT")
gal1 <- readBamFile(bamFile=bamfile, tag=tags,
                    which=GRanges("chr1", IRanges(1, 1e6)),
                    asMates=FALSE)
names(gal1) <- mcols(gal1)$qname
library(BSgenome.Hsapiens.UCSC.hg19)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(phastCons100way.UCSC.hg19)
splitGAlignmentsByCut(gal1, txs=txs, genome=Hsapiens,
                      conservation=phastCons100way.UCSC.hg19)
```

TSSEscore *Transcription Start Site (TSS) Enrichment Score*

Description

TSS score is a raio between aggregate distribution of reads centered on TSSs and that flanking the corresponding TSSs. TSS score = the depth of TSS (each step within 1000 bp each side) / the depth of end flanks (100bp each end). TSSE score = max(mean(TSS score in each window)).

Usage

```
TSSEscore(
  obj,
  txs,
  seqlev = intersect(seqlevels(obj), seqlevels(txs)),
  upstream = 1000,
  downstream = 1000,
  endSize = 100,
  width = 100,
  step = width,
  pseudocount = 0,
  ...
)
```

Arguments

obj	an object of GAlignments
txs	GRanges of transcripts
seqlev	A vector of characters indicates the sequence levels.
upstream, downstream	numeric(1) or integer(1). upstream and downstream of TSS. Default is 1000
endSize	numeric(1) or integer(1). the size of the end flanks. Default is 100
width	numeric(1) or integer(1). the window size for TSS score. Default is 100.
step	numeric(1) or integer(1). The distance between the start position of the sliding windows.
pseudocount	numeric(1) or integer(1). Pseudocount. Default is 0. If pseudocount is no greater than 0, the features with ZERO or less than ZERO counts in flank region will be removed in calculation.
...	parameter can be passed to loess.smooth other than 'x', 'y', 'family' and 'evaluation'.

Value

A object of list with TSS scores

Author(s)

Jianhong Ou

References

https://www.encodeproject.org/data-standards/terms/#enrichment

Examples

```
library(GenomicRanges)
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC", mustWork=TRUE)
gal1 <- readBamFile(bamFile=bamfile, tag=character(0),
                    which=GRanges("chr1", IRanges(1, 1e6)),
                    asMates=FALSE)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txs <- transcripts(TxDb.Hsapiens.UCSC.hg19.knownGene)
tsse <- TSSEscore(gal1, txs)
```

vPlot *V-plot*

Description

Aggregate ATAC-seq Fragment Midpoint vs. Length for a given motif generated over binding sites within the genome.

Usage

```
vPlot(
  bamfiles,
  index = bamfiles,
  pfm,
  genome,
  min.score = "95%",
  bindingSites,
  seqlev = paste0("chr", c(1:22, "X", "Y")),
  upstream = 200,
  downstream = 200,
  maxSiteNum = 1e+06,
  draw = TRUE,
  ...
)
```

Arguments

bamfiles	A vector of characters indicates the file names of bams. All the bamfiles will be pulled together.
index	The names of the index file of the 'BAM' file being processed; This is given without the '.bai' extension.
pfm	A Position frequency Matrix represented as a numeric matrix with row names A, C, G and T.
genome	An object of BSgenome.
min.score	The minimum score for counting a match. Can be given as a character string containing a percentage (e.g. "95 score or as a single number. See matchPWM.
bindingSites	A object of GRanges indicates candidate binding sites (eg. the output of fimo).
seqlev	A vector of characters indicates the sequence levels.
upstream, downstream	numeric(1) or integer(1). Upstream and downstream of the binding region for aggregate ATAC-seq footprint.
maxSiteNum	numeric(1). Maximal number of predicted binding sites. if predicted binding sites is more than this number, top maxSiteNum binding sites will be used.
draw	Plot or not. Default TRUE.
...	parameters could be used by smoothScatter

Value

an invisible data.frame for plot.

Author(s)

Jianhong Ou

References

Jorja G. Henikoff, Jason A. Belsky, Kristina Krassovsky, David M. MacAlpine, and Steven Henikoff. Epigenome characterization at single base-pair resolution. PNAS 2011 108 (45) 18318-18323

Examples

```
bamfile <- system.file("extdata", "GL1.bam",
                       package="ATACseqQC")
library(MotifDb)
CTCF <- query(MotifDb, c("CTCF"))
CTCF <- as.list(CTCF)
library(BSgenome.Hsapiens.UCSC.hg19)
vPlot(bamfile, pfm=CTCF[[1]],
     genome=Hsapiens,
     min.score="95%", seqlev="chr1",
     ylim=c(30, 250))
```

---

writeListOfGAlignments

export list of GAlignments into bam files

Description

wraper for export to export list of GAlignment into bam files.

Usage

```
writeListOfGAlignments(objs, outPath = ".")
```

Arguments

objs	A list of GAlignments.
outPath	character(1). Output file path.

Value

status of export.

Author(s)

Jianhong Ou

Examples

```
library(GenomicAlignments)
gal1 <- GAlignments(seqnames=Rle("chr1"), pos=1L, cigar="10M",
                    strand=Rle(strand(c("+"))), names="a", score=1)
galist <- GAlignmentsList(a=gal1)
writeListOfGAlignments(galist)
```

Index

ATACseqQC *(*ATACseqQC-package*)*, 3
ATACseqQC-package, 3

bamQC, 3
BSgenome, *9*, *11*, *27*, *32*

DB, 4
distanceDyad, 4
DNAString, *20*

enrichedFragments, 5
estimateLibComplexity, 7
estLibSize, *6*
export, *33*

factorFootprints, 8
footprintsScanner, 10
fragSizeDist, 12

GAlignments, *13*, *18, 19*, *24, 25*, *28, 29*, *31*, *33*
GAlignmentsList, *15*, *25*
GRanges, *6*, *9*, *14, 15*, *18, 19*, *21*, *27*, *32*
GRangesList, *11*
GScores, *27*, *29*

IntegerRangesList, *21*

MaskedDNAString, *20*
matchPWM, *9*, *32*

NFRscore, 13
normalizeBetweenArrays, *6*

peakdet, 14
plotCorrelation, 15
plotFootprints, *9*, 16
prepareBindingSitesList
        *(*footprintsScanner*)*, 10
pseudoPausingIndex, 17
PTscore, 19
pwmscores, 20

randomForest, *28*
readBamFile, 20
readGAlignments, *21*
readGAlignmentsList, *21*
readsDupFreq, *8*, 21

saturationPlot, 22
scanBam, *21*
scanBamFlag, *21*
ScanBamParam, *21*
shiftGAlignments, 24
shiftGAlignmentsList, 25, *28*
shiftReads, 26
smoothScatter, *32*
splitBam, 27
splitGAlignmentsByCut, *28*, 28

TSSEscore, 30

Views, *20*
vPlot, *5*, 32

writeListOfGAlignments, *28*, 33