Package: mosaics (via r-universe)

November 1, 2026

Type Package

Title MOSAiCS (MOdel-based one and two Sample Analysis and Inference for ChIP-Seq)

Version 2.51.0

Depends R (>= 3.0.0), methods, graphics, Rcpp

Imports MASS, splines, lattice, IRanges, GenomicRanges, GenomicAlignments, Rsamtools, Seqinfo, S4Vectors

Suggests mosaicsExample

Enhances parallel

LinkingTo Rcpp

SystemRequirements Perl

Date 2025-07-24

Author Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

Maintainer Dongjun Chung <dongjun.chung@gmail.com>

Description This package provides functions for fitting MOSAiCS and MOSAiCS-HMM, a statistical framework to analyze one-sample or two-sample ChIP-seq data of transcription factor binding and histone modification.

License GPL (>= 2)

URL http://groups.google.com/group/mosaics_user_group

LazyLoad yes

biocViews ChIPseq, Sequencing, Transcription, Genetics, Bioinformatics

Config/pak/sysreqs libbz2-dev liblzma-dev perl xz-utils zlib1g-dev

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

Date/Publication 2026-04-28 12:35:06 UTC

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

RemoteRef HEAD

RemoteSha f0756a232b5b88b4bdf20cfdfa4706905a7dad1c

Contents

mosaics-package . . . 2
adjustBoundary . . . 5
BinData-class . . . 8
constructBins . . . 10
estimates . . . 12
export . . . 14
extractReads . . . 16
filterPeak . . . 19
findSummit . . . 21
generateWig . . . 22
mosaicsFit . . . 25
MosaicsFit-class . . . 27
MosaicsFitEst-class . . . 28
mosaicsFitHMM . . . 30
MosaicsFitParam-class . . . 33
MosaicsHMM-class . . . 34
mosaicsPeak . . . 36
MosaicsPeak-class . . . 38
mosaicsPeakHMM . . . 42
MosaicsPeakParam-class . . . 45
mosaicsRunAll . . . 46
readBins . . . 50
TagData-class . . . 52

Index **53**

`mosaics-package` *MOSAiCS (MOdel-based one and two Sample Analysis and Inference for ChIP-Seq)*

Description

This package provides functions for fitting MOSAiCS, a statistical framework to analyze one-sample or two-sample ChIP-seq data.

Details

Package:	mosaics
Type:	Package
Version:	2.9.9
Date:	2016-03-15
License:	GPL (>= 2)
LazyLoad:	yes

This package contains four main classes, BinData, MosaicsFit, MosaicsHMM, and MosaicsPeak, which represent bin-level ChIP-seq data, MOSAiCS model fit, MOSAiCS-HMM model fit, and MOSAiCS peak calling results, respectively. This package contains ten main methods, constructBins, readBins, mosaicsFit, mosaicsPeak, mosaicsFitHMM, mosaicsPeakHMM, extractReads, findSummit, adjustBoundary, filterPeak. constructBins method constructs bin-level files from the aligned read file. readBins method imports bin-level data and construct BinData class object. mosaicsFit method fits a MOSAiCS model using BinData class object and constructs MosaicsFit class object. mosaicsPeak method calls peaks using MosaicsFit class object and construct MosaicsPeak class object. mosaicsFitHMM and mosaicsPeakHMM are designed to identify broad peaks and their functions correspond to mosaicsFit and mosaicsPeak, respectively. mosaicsFitHMM method fits a MOSAiCS-HMM model using MosaicsFit class object and constructs MosaicsHMM class object. mosaicsPeakHMM method calls MOSAiCS-HMM peaks using MosaicsHMM class object and construct MosaicsPeak class object. extractReads, findSummit, adjustBoundary, filterPeak methods postprocess MOSAiCS and MOSAiCS-HMM peaks using MosaicsPeak class object (incorporate read-level data, identify peak summits, adjust peak boundaries, and filter potential false positive peaks, respectively) and construct MosaicsPeak class object. MosaicsPeak class object can be exported as text files or transformed into data frame, which can be used for the downstream analysis. This package also provides methods for simple exploratory analysis.

The mosaics package companion website, http://www.stat.wisc.edu/~keles/Software/mosaics/, provides preprocessing scripts, preprocessed files for diverse reference genomes, and easy-to-follow instructions. We encourage questions or requests regarding mosaics package to be posted on our Google group, http://groups.google.com/group/mosaics_user_group. Please check the vignette for further details on the mosaics package and these websites.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

Maintainer: Dongjun Chung <dongjun.chung@gmail.com>

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

constructBins, readBins, mosaicsFit, mosaicsPeak, mosaicsFitHMM, mosaicsPeakHMM, extractReads, findSummit, adjustBoundary, filterPeak, BinData, MosaicsFit, MosaicsHMM, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

# example analysis workflow for ChIP-seq data of transcription factor binding
```

```
# (STAT1 factor binding in GM12878 cell line, from ENCODE database)

generateWig( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam")
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, span=200, capping=0, normConst=1 )

constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binTFBS <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
    "./wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binTFBS
head(print(binTFBS))
plot(binTFBS)
plot( binTFBS, plotType="input" )

fitTFBS <- mosaicsFit( binTFBS, analysisType="IO", bgEst="rMOM" )
fitTFBS
plot(fitTFBS)

peakTFBS <- mosaicsPeak( fitTFBS, signalModel="2S", FDR=0.05,
maxgap=200, minsize=50, thres=10 )
peakTFBS
head(print(peakTFBS))

peakTFBS <- extractReads( peakTFBS,
 chipFile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam"), package=
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam"), packa
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakTFBS

peakTFBS <- findSummit( peakTFBS, parallel=FALSE, nCore=8 )

export( peakTFBS, type = "txt", filename = "./peakTFBS.txt" )
export( peakTFBS, type = "bed", filename = "./peakTFBS.bed" )
export( peakTFBS, type = "gff", filename = "./peakTFBS.gff" )
export( peakTFBS, type = "narrowPeak", filename = "./peakTFBS.narrowPeak" )

# example analysis workflow for ChIP-seq data of histone modification
# (H3K4me3 modification in GM12878 cell line, from ENCODE database)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
```

```
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

fileName <- file.path(
   c("wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
    "wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt"))
binHM <- readBins( type=c("chip","input"), fileName=fileName )
binHM
plot(binHM)
plot( binHM, plotType="input" )

fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
fitHM
plot(fitHM)

hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
hmmHM
plot(hmmHM)

peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM

peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
head(print(peakHM))
plot( peakHM, filename="./peakplot_HM.pdf" )

peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

export( peakHM, type = "txt", filename = "./peakHM.txt" )
export( peakHM, type = "bed", filename = "./peakHM.bed" )
export( peakHM, type = "gff", filename = "./peakHM.gff" )
export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

---

adjustBoundary *Adjust boundaries of peak regions*

Description

Adjust boundaries of peak regions in the MosaicsPeak class object, which is a peak calling result.

Usage

```
adjustBoundary( object, ... )
## S4 method for signature 'MosaicsPeak'
adjustBoundary( object, minRead=10, extendFromSummit=100,
    trimMinRead1=1.5, trimFC1=5, extendMinRead1=2, extendFC1=50,
    trimMinRead2=1.5, trimFC2=50, extendMinRead2=1.5, extendFC2=50,
    normC=NA, parallel=FALSE, nCore=8 )
```

Arguments

object	Object of class MosaicsPeak, a peak list object obtained using either functions mosaicsPeak or mosaicsPeakHMM.
minRead	Parameter to determine baseline for trimming and extension of peak boundaries.
extendFromSummit	If the updated peak regions do not include the peak summit, either peak start or end is extended by extendFromSummit.
trimMinRead1	Parameter to determine to trim peak boundaries.
trimFC1	Parameter to determine to trim peak boundaries.
extendMinRead1	Parameter to determine to extend peak boundaries.
extendFC1	Parameter to determine to extend peak boundaries.
trimMinRead2	Parameter used to trim peak boundaries.
trimFC2	Parameter used to trim peak boundaries.
extendMinRead2	Parameter used to extend peak boundaries.
extendFC2	Parameter used to extend peak boundaries.
normC	Normalizing constant. If not provided, normC is estimated as ratio of sequencing depth of ChIP over matched control samples.
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).
nCore	Number of CPUs when parallel computing is utilized.
...	Other parameters to be passed through to generic mosaicsHMM.

Details

adjustBoundary adjusts peak boundaries. While adjustBoundary can be applied to a peak list object obtained using either functions mosaicsPeak or mosaicsPeakHMM, adjustBoundary is developed and tested mainly for peak lists from MOSAiCS-HMM model (i.e., from function mosaicsPeakHMM). Note that extractReads should be run first because adjustBoundary is used.

Parallel computing can be utilized for faster computing if parallel=TRUE and parallel package is loaded. nCore determines number of CPUs used for parallel computing.

Value

Construct MosaicsPeak class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, mosaicsPeakHMM, extractReads, findSummit, filterPeak, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
```

```
## End(Not run)
```

BinData-class *Class "BinData"*

Description

This class represents bin-level ChIP-seq data.

Objects from the Class

Objects can be created by calls of the form `new("BinData", ...)`.

Slots

`chrID`: Object of class `"character"`, a vector of chromosome IDs.

`coord`: Object of class `"numeric"`, a vector of genomic coordinates.

`tagCount`: Object of class `"numeric"`, a vector of tag counts of ChIP sample.

`mappability`: Object of class `"numeric"`, a vector of mappability score.

`gcContent`: Object of class `"numeric"`, a vector of GC content score.

`input`: Object of class `"numeric"`, a vector of tag counts of matched control sample.

`dataType`: Object of class `"character"`, indicating how reads were processed. Possible values are "unique" (only uniquely aligned reads were retained) and "multi" (reads aligned to multiple locations were also retained).

`seqDepth`: Object of class `"numeric"`, a vector of sequencing depth of length 2, where the first and second elements correpond to sequencing depths of ChIP and control samples, respectively. If there is not control sample, the second element is set to NA.

Methods

mosaicsFit `signature(object = "BinData")`: fit a MOSAiCS model using a bin-level ChIP-seq data.

plot `signature(x = "BinData", y = "missing", plotType = NULL )`: provide exploratory plots of mean ChIP tag counts. This method plots mean ChIP tag counts versus mappability score, GC content score, and Control tag counts, with 95% confidence intervals, for `plotType="M"`, `plotType="GC"`, and `plotType="input"`, respectively. `plotType="M|input"` and `plotType="GC|input"` provide plots of mean ChIP tag counts versus mappability and GC content score, respectively, conditional on Control tag counts. If `plotType` is not specified, this method plots histogram of ChIP tag counts.

print `signature(x = "BinData")`: return bin-level data in data frame format.

show `signature(object = "BinData")`: provide brief summary of the object.

chrID `signature(object = "BinData")`: provide a vector of chromosome ID.

coord `signature(object = "BinData")`: provide a vector of genomic coordinates.

tagCount `signature(object = "BinData")`: provide a vector of tag count of ChIP sample.

input `signature(object = "BinData")`: provide a vector of tag count of input sample.

mappability `signature(object = "BinData")`: provide a vector of mappability score.

gcContent `signature(object = "BinData")`: provide a vector of GC content score.

seqDepth `signature(object = "BinData")`: provide a vector of sequencing depth.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`readBins`, `mosaicsFit`.

Examples

```
showClass("BinData")
## Not run:
library(mosaicsExample)
data(exampleBinData)

exampleBinData
print(exampleBinData)[1:10,]
plot(exampleBinData)
plot( exampleBinData, plotType="M" )
plot( exampleBinData, plotType="GC" )
plot( exampleBinData, plotType="input" )
plot( exampleBinData, plotType="M|input" )
plot( exampleBinData, plotType="GC|input" )

exampleFit <- mosaicsFit( exampleBinData, analysisType="IO" )

## End(Not run)
```

constructBins *Construct bin-level ChIP-sep data from an aligned read file*

Description

Preprocess and construct bin-level ChIP-sep data from an aligned read file.

Usage

```
constructBins( infile=NULL, fileFormat=NULL, outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0, perl = "perl" )
```

Arguments

infile	Name of the aligned read file to be processed.
fileFormat	Format of the aligned read file to be processed. Currently, constructBins permits the following aligned read file formats for SET data (PET = FALSE): "eland_result" (Eland result), "eland_extended" (Eland extended), "eland_export" (Eland export), "bowtie" (default Bowtie), "sam" (SAM), "bam" (BAM), "bed" (BED), and "csem" (CSEM). For PET data (PET = TRUE), the following aligned read file formats are allowed: "eland_result" (Eland result), "sam" (SAM), and "bam" (BAM).
outfileLoc	Directory of processed bin-level files. By default, processed bin-level files are exported to the current directory.
byChr	Construct separate bin-level file for each chromosome? Possible values are TRUE or FALSE. If byChr=FALSE, bin-level data for all chromosomes are exported to one file. If byChr=TRUE, bin-level data for each chromosome is exported to a separate file. Default is FALSE.
useChrfile	Is the file for chromosome info provided? Possible values are TRUE or FALSE. If useChrfile=FALSE, it is assumed that the file for chromosome info is not provided. If useChrfile=TRUE, it is assumed that the file for chromosome info is provided. Default is FALSE.
chrfile	Name of the file for chromosome info. In this file, the first and second columns are ID and size of each chromosome, respectively.
excludeChr	Vector of chromosomes that will be excluded from the analysis. This argument is ignored if useChrfile=TRUE.
PET	Is the file paired-end tag (PET) data? If PET=FALSE, it is assumed that the file is SET data. If PET=TRUE, it is assumed that the file is PET data. Default is FALSE (SET data).
fragLen	Average fragment length. Default is 200. This argument is ignored if PET=TRUE.
binSize	Size of bins. Default is 200.

capping	Maximum number of reads allowed to start at each nucleotide position. To avoid potential PCR amplification artifacts, the maximum number of reads that can start at a nucleotide position is capped at capping. Capping is not applied if non-positive value is used for capping. Default is 0 (no capping).
perl	Name of the perl executable to be called. Default is "perl".

Details

Bin-level files are constructed from the aligned read file and exported to the directory specified in outfileLoc argument. If byChr=FALSE, bin-level files are named as [infileName]_fragL[fragLen]_bin[binSize].txt for SET data (PET = FALSE) and [infileName]_bin[binSize].txt for PET data (PET = TRUE). If byChr=TRUE, bin-level files are named as [infileName]_fragL[fragLen]_bin[binSize]_[chrID].txtfor SET data (PET = FALSE) and [infileName]_bin[binSize]_[chrID].txt for PET data (PET = TRUE), where chrID is chromosome IDs that reads align to. These chromosome IDs are extracted from the aligned read file.

If the file for chromosome information is provided (useChrfile=TRUE and chrfile is not NULL), only the chromosomes specified in the file will be considered. Chromosomes that are specified in excludeChr will not be included in the processed bin-level files. excludeChr argument is ignored if useChrfile=TRUE. Constructed bin-level files can be loaded into the R environment using the method readBins.

constructBins currently supports the following aligned read file formats for SET data (PET = FALSE): Eland result ("eland_result"), Eland extended ("eland_extended"), Eland export ("eland_export"), default Bowtie ("bowtie"), SAM ("sam"), "bam" (BAM), BED ("bed"), and CSEM ("csem"). For PET data (PET = TRUE), the following aligned read file formats are allowed: "eland_result" (Eland result), "sam" (SAM), and "bam" (BAM).

If input file format is neither BED nor CSEM BED, this method retains only reads mapping uniquely to the reference genome.

Value

Processed bin-level files are exported to the directory specified in outfileLoc.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

readBins, BinData.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binHM

## End(Not run)
```

---

estimates *Extract estimates of the fitted MOSAiCS model*

Description

Extract estimates from `MosaicsFit` class object, which is a fitted MOSAiCS model.

Usage

```
estimates( object, ... )
## S4 method for signature 'MosaicsFit'
estimates( object )
```

Arguments

`object`	Object of class `MosaicsFit`, which represents a fitted MOSAiCS model obtained using method `mosaicsFit`.
`...`	Other parameters to be passed through to generic `estimates`.

Value

Returns a list with components:

`pi0`	Mixing proportion of background component.
`a`	Parameter for background component.
`betaEst`	Parameter for background component (coefficient estimates).
`muEst`	Parameter for background component.

b	Parameter for one-signal-component model.
c	Parameter for one-signal-component model.
p1	Parameter for two-signal-component model (mixing proportion of signal components).
b1	Parameter for two-signal-component model (the first signal component).
c1	Parameter for two-signal-component model (the first signal component).
b2	Parameter for two-signal-component model (the second signal component).
c2	Parameter for two-signal-component model (the second signal component).
analysisType	Analysis type. Possible values are "OS" (one-sample analysis), "TS" (two-sample analysis using mappability and GC content), and "IO" (two-sample analysis without using mappability and GC content).

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsFit, MosaicsFit.

Examples

```
## Not run:
library(mosaicsExample)
data(exampleBinData)
exampleFit <- mosaicsFit( exampleBinData, analysisType="IO" )
estimates(exampleFit)

## End(Not run)
```

export *Export peak calling results to text files*

Description

Export peak calling results to text files in TXT, BED, or GFF file formats.

Usage

```
export(object, ...)
## S4 method for signature 'MosaicsPeak'
export( object, type=NA, filename=NA )
```

Arguments

object	Object of class MosaicsPeak, peak calling results obtained using method mosaicsPeak.
type	Format of the exported file. Possible values are "txt", "bed", "gff", "narrowPeak", and "broadPeak". See Details.
filename	Name of the exported file.
...	Other parameters to be passed through to generic export.

Details

TXT file format (type="txt") exports peak calling results in the most informative way. Columns include chromosome ID, peak start position, peak end position, peak width, -log10 transformed average posterior probability, -log10 transformed minimum posterior probability, average of -log10 transformed posterior probability, average ChIP tag count, maximum ChIP tag count (always), average input tag count, average input tag count scaled by sequencing depth, average log base 2 ratio of ChIP over input tag counts (if matched control sample is also provided), average mappability score, and average GC content score (when mappability and GC content scores are used in the analysis) in each peak. type="bed" and type="gff" export peak calling results in standard BED and GFF file formats, respectively, where score is the average ChIP tag counts in each peak. type="narrowPeak" and type="broadPeak" export peak calling results in ENCODE narrowPeak and broadPeak file formats, respectively, where score, signalValue, pValue, and qValue are average log base 2 ratio of ChIP over input tag counts (or average ChIP tag count, if matched control sample is not provided), ChIP signal at the summit, -log10 transformation of minimum posterior probability (logMinP), and average of -log10 transformed posterior probability (aveLogP), respectively, in each peak. If no peak is detected, export method will not generate any file.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

export( peakHM, type = "txt", filename = "./peakHM.txt" )
export( peakHM, type = "bed", filename = "./peakHM.bed" )
export( peakHM, type = "gff", filename = "./peakHM.gff" )

# read-level data is needed to loaded and peak summits need to be identified
# to export in narrowPeak and broadPeak file formats

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM

peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
```

```
peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

`extractReads` *Load read-level data and extract reads corresponding to each peak region*

Description

Load read-level data and extract reads corresponding to each peak region in the `MosaicsPeak` class object, which is a peak calling result.

Usage

```
extractReads( object, ... )
## S4 method for signature 'MosaicsPeak'
extractReads( object, chipFile=NULL, chipFileFormat=NULL,
    chipPET=FALSE, chipFragLen=200,
    controlFile=NULL, controlFileFormat=NULL,
    controlPET=FALSE, controlFragLen=200, keepReads=FALSE,
    parallel=FALSE, nCore=8, tempDir=NULL, perl="perl" )
```

Arguments

`object`	Object of class `MosaicsPeak`, a peak list object obtained using either functions `mosaicsPeak` or `mosaicsPeakHMM`.
`chipFile`	Name of the aligned read file for ChIP sample.
`chipFileFormat`	Format of the aligned read file for ChIP sample. Currently, `constructBins` permits the following aligned read file formats for SET data (`chipPET = FALSE`): `"eland_result"` (Eland result), `"eland_extended"` (Eland extended), `"eland_export"` (Eland export), `"bowtie"` (default Bowtie), `"sam"` (SAM), `"bam"` (BAM), `"bed"` (BED), and `"csem"` (CSEM). For PET data (`chipPET = TRUE`), the following aligned read file formats are allowed: `"eland_result"` (Eland result), `"sam"` (SAM), and `"bam"` (BAM).
`chipPET`	Is the file of ChIP sample paired-end tag (PET) data? If `chipPET=FALSE`, it is assumed that the file is SET data. If `chipPET=TRUE`, it is assumed that the file is PET data. Default is `FALSE` (SET data).
`chipFragLen`	Average fragment length of ChIP sample. Default is 200. This argument is ignored if `chipPET=TRUE`.
`controlFile`	Name of the aligned read file for matched control sample.

controlFileFormat
: Format of the aligned read file for matched control sample. Currently, constructBins permits the following aligned read file formats for SET data (controlPET = FALSE): "eland_result" (Eland result), "eland_extended" (Eland extended), "eland_export" (Eland export), "bowtie" (default Bowtie), "sam" (SAM), "bam" (BAM), "bed" (BED), and "csem" (CSEM). For PET data (controlPET = TRUE), the following aligned read file formats are allowed: "eland_result" (Eland result), "sam" (SAM), and "bam" (BAM).

controlPET
: Is the file of matched control sample paired-end tag (PET) data? If controlPET=FALSE, it is assumed that the file is SET data. If controlPET=TRUE, it is assumed that the file is PET data. Default is FALSE (SET data).

controlFragLen
: Average fragment length of matched control sample. Default is 200. This argument is ignored if controlPET=TRUE.

keepReads
: Keep read-level data? Default is FALSE (do not keep read-level data).

parallel
: Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).

nCore
: Number of CPUs when parallel computing is utilized.

tempDir
: Directory to store temporary files. If tempDir=NULL, extractReads() will use the temporary directory used by R.

perl
: Name of the perl executable to be called. Default is "perl".

...
: Other parameters to be passed through to generic extractReads.

Details

Read-level data is first loaded from aligned read files, and then the reads corresponding to each peak region are extracted and incorporated into the MosaicsPeak class object. extractReads currently supports the following aligned read file formats for SET data (chipPET = FALSE and controlPET = FALSE): Eland result ("eland_result"), Eland extended ("eland_extended"), Eland export ("eland_export"), default Bowtie ("bowtie"), SAM ("sam"), BAM ("bam"), BED ("bed"), and CSEM ("csem"). For PET data (chipPET = FALSE and controlPET = FALSE), the following aligned read file formats are allowed: "eland_result" (Eland result), "sam" (SAM), and "bam" (BAM).

If input file format is neither BED nor CSEM BED, this method retains only reads mapping uniquely to the reference genome.

Value

Construct MosaicsPeak class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, mosaicsPeakHMM, export, findSummit, adjustBoundary, filterPeak, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )

## End(Not run)
```

filterPeak *Filter out potentially false positive peaks*

Description

Filter out potentially false positive peaks from MosaicsPeak class object, which is a peak calling result.

Usage

```
filterPeak( object, ... )
## S4 method for signature 'MosaicsPeak'
filterPeak( object,
 minSummitQuantile=0.01, minRead=10, FC=50, minSignal = 20, minLen = 200,
 normC=NA, parallel=FALSE, nCore=8 )
```

Arguments

object	Object of class MosaicsPeak, a peak list object obtained using either functions mosaicsPeak or mosaicsPeakHMM.
minSummitQuantile	Parameter for the filtering step #1. Peaks are filtered out if signal strengths at summits < minSummitQuantile quantile of signal strengths at summits across all the peak regions.
minRead	Parameter for the filtering step #1. Peaks are filtered out if the number of reads in the peak region < minRead.
FC	Parameter for the filtering step #1. Peaks are filtered out if the improvement of ChIP over matched control < FC.
minSignal	Parameter for the filtering step #2. Peaks are filtered out if signal strengths at summits <= minSignal.
minLen	Parameter for the filtering step #2. Peaks are filtered out if their lengths <= minLen.
normC	Normalizing constant. If not provided, normC is estimated as ratio of sequencing depth of ChIP over matched control samples.
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).
nCore	Number of CPUs when parallel computing is utilized.
...	Other parameters to be passed through to generic filterPeak.

Details

filterPeak filters out potentially false positive peaks, based on signal strengths and peak lengths. While filterPeak can be applied to a peak list object obtained using either functions mosaicsPeak or mosaicsPeakHMM, filterPeak is developed and tested mainly for peak lists from MOSAiCS-HMM model (i.e., from function mosaicsPeakHMM). Note that extractReads should be run first because filterPeak is used.

Value

Construct MosaicsPeak class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, mosaicsPeakHMM, extractReads, findSummit, adjustBoundary, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
```

```
## End(Not run)
```

---

findSummit *Find a summit for each peak region*

Description

Find a summit for each peak region in the MosaicsPeak class object, which is a peak calling result.

Usage

```
findSummit( object, ... )
## S4 method for signature 'MosaicsPeak'
findSummit( object, parallel=FALSE, nCore=8 )
```

Arguments

object	Object of class MosaicsPeak, a peak list object obtained using either functions mosaicsPeak or mosaicsPeakHMM.
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).
nCore	Number of CPUs when parallel computing is utilized.
...	Other parameters to be passed through to generic findSummit.

Details

Note that extractReads should be run first because findSummit is used.

Value

Construct MosaicsPeak class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, mosaicsPeakHMM, export, extractReads, adjustBoundary, filterPeak, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )

export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

---

generateWig *Construct wiggle files from an aligned ChIP-sep read file*

Description

Construct wiggle files from an aligned ChIP-sep read file.

Usage

```
generateWig( infile=NULL, fileFormat=NULL, outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, span=200, capping=0, normConst=1, perl = "perl" )
```

Arguments

infile	Name of the aligned read file to be processed.
fileFormat	Format of the aligned read file to be processed. Currently, generateWig permits the following aligned read file formats for SET data (PET = FALSE): "eland_result" (Eland result), "eland_extended" (Eland extended), "eland_export" (Eland export), "bowtie" (default Bowtie), "sam" (SAM), "bam" (BAM), "bed" (BED), and "csem" (CSEM). For PET data (PET = TRUE), the following aligned read file formats are allowed: "eland_result" (Eland result), "sam" (SAM), and "bam" (BAM).
outfileLoc	Directory of processed wiggle files. By default, processed wiggle files are exported to the current directory.
byChr	Construct separate wiggle file for each chromosome? Possible values are TRUE or FALSE. If byChr=FALSE, all chromosomes are exported to one file. If byChr=TRUE, each chromosome is exported to a separate file. Default is FALSE.
useChrfile	Is the file for chromosome info provided? Possible values are TRUE or FALSE. If useChrfile=FALSE, it is assumed that the file for chromosome info is not provided. If useChrfile=TRUE, it is assumed that the file for chromosome info is provided. Default is FALSE.
chrfile	Name of the file for chromosome info. In this file, the first and second columns are ID and size of each chromosome, respectively.
excludeChr	Vector of chromosomes that will be excluded from the analysis. This argument is ignored if useChrfile=TRUE.
PET	Is the file paired-end tag (PET) data? If PET=FALSE, it is assumed that the file is SET data. If PET=TRUE, it is assumed that the file is PET data. Default is FALSE (SET data).
fragLen	Average fragment length. Default is 200. This argument is ignored if PET=TRUE.
span	Span used in wiggle files. Default is 200.
capping	Maximum number of reads allowed to start at each nucleotide position. To avoid potential PCR amplification artifacts, the maximum number of reads that can start at a nucleotide position is capped at capping. Capping is not applied if non-positive value is used for capping. Default is 0 (no capping).
normConst	Normalizing constant to scale values in each position.
perl	Name of the perl executable to be called. Default is "perl".

Details

Wiggle files are constructed from the aligned read file and exported to the directory specified in outfileLoc argument. If byChr=FALSE, wiggle files are named as [infileName]_fragL[fragLen]_span[span].wig for SET data (PET = FALSE) and [infileName]_span[span].wig for PET data (PET = TRUE). If

byChr=TRUE, wiggle files are named as [infileName]_fragL[fragLen]_span[span]_[chrID].wig for SET data (PET = FALSE) and [infileName]_span[span]_[chrID].wig for PET data (PET = TRUE), where chrID is chromosome IDs that reads align to. These chromosome IDs are extracted from the aligned read file.

If the file for chromosome information is provided (useChrfile=TRUE and chrfile is not NULL), only the chromosomes specified in the file will be considered. Chromosomes that are specified in excludeChr will not be included in the processed wiggle files. excludeChr argument is ignored if useChrfile=TRUE.

generateWig currently supports the following aligned read file formats for SET data (PET = FALSE): Eland result ("eland_result"), Eland extended ("eland_extended"), Eland export ("eland_export"), default Bowtie ("bowtie"), SAM ("sam"), , "bam" (BAM), BED ("bed"), and CSEM ("csem"). For PET data (PET = TRUE), the following aligned read file formats are allowed: "eland_result" (Eland result), "sam" (SAM), and "bam" (BAM).

If input file format is neither BED nor CSEM BED, this method retains only reads mapping uniquely to the reference genome.

Value

Processed wig files are exported to the directory specified in outfileLoc.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

Examples

```
## Not run:
library(mosaicsExample)

generateWig( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam")
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, span=200, capping=0, normConst=1 )

## End(Not run)
```

mosaicsFit *Fit MOSAiCS model*

Description

Fit one-sample or two-sample MOSAiCS models with one signal component and two signal components.

Usage

```
mosaicsFit( object, ... )
## S4 method for signature 'BinData'
mosaicsFit( object, analysisType="automatic", bgEst="rMOM",
   k=3, meanThres=NA, s=2, d=0.25, trans="power", truncProb=0.999, parallel=FALSE, nCore=8 )
```

Arguments

object	Object of class BinData, bin-level ChIP-seq data imported using method readBins.
analysisType	Analysis type. Possible values are "OS" (one-sample analysis), "TS" (two-sample analysis using mappability and GC content), and "IO" (two-sample analysis without using mappability and GC content). If analysisType="automatic", this method tries to make the best guess for analysisType, based on the data provided.
bgEst	Parameter to determine background estimation approach. Possible values are "matchLow" (estimation using bins with low tag counts) and "rMOM" (estimation using robust method of moment (MOM)). If bgEst="automatic", this method tries to make the best guess for bgEst, based on the data provided. Default is bgEst="rMOM".
k	Parameter for estimating background distribution. It is not recommended for users to change this value.
meanThres	Parameter for estimating background distribution. Default is 1 for analysisType="TS" and 0 for analysisType="OS". Not relevant when analysisType="IO".
s	Parameter for estimating background distribution. Relevant only when analysisType="TS". Default is 2.
d	Parameter for estimating background distribution. Relevant only when analysisType="TS" or analysisType="IO". Default is 0.25.
trans	Transformation of matching control tag count. Possible values are "log" (logarithm transformation) and "power" (power transformation). Relevant only when analysisType="IO". Default is trans="power".
truncProb	Parameter for estimating background distribution. Relevant only when analysisType="IO".
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).
nCore	Number of CPUs when parallel computing is utilized.
...	Other parameters to be passed through to generic mosaicsFit.

Details

The imported data type constraints the analysis that can be implemented. If only data for ChIP sample and matched control sample (i.e., either `type=c("chip", "input")` or `type=c("chip", "input", "N")` was used in method `readBins`), only two-sample analysis without using mappability and GC content (`analysisType="IO"`) is allowed. If matched control data is available with mappability score, GC content score, and sequence ambiguity score, (i.e., `type=c("chip", "input", "M", "GC", "N")` was used in method `readBins`), user can do all of three analysis types (`analysisType="OS"`, `analysisType="TS"`, or `analysisType="IO"`). If there is no data for matched control sample (i.e., `type=c("chip", "M", "GC", "N")` was used in method `readBins`), only one-sample analysis (`analysisType="OS"`) is permitted.

Parallel computing can be utilized for faster computing if `parallel=TRUE` and `parallel` package is loaded. `nCore` determines number of CPUs used for parallel computing. `meanThres`, `s`, `d`, `trans`, and `truncProb` are the tuning parameters for estimating background distribution. The vignette and Kuan et al. (2011) provide further details about these tuning parameters. Please do not try different value for `k` argument.

Value

Construct `MosaicsFit` class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`readBins`, `mosaicsFitHMM`, `MosaicsFit`.

Examples

```
## Not run:
library(mosaicsExample)
data(exampleBinData)
exampleFit <- mosaicsFit( exampleBinData, analysisType="IO" )

## End(Not run)
```

MosaicsFit-class *Class "MosaicsFit"*

Description

This class represents MOSAiCS model fit.

Objects from the Class

Objects can be created by calls of the form `new("MosaicsFit", ...)`.

Slots

`mosaicsEst`: Object of class `"MosaicsFitEst"`, representing estimates of MOSAiCS model fit.

`mosaicsParam`: Object of class `"MosaicsFitParam"`, representing tuning parameters for fitting MOSAiCS model.

`chrID`: Object of class `"character"`, a vector of chromosome IDs.

`coord`: Object of class `"numeric"`, a vector of genomic coordinates.

`tagCount`: Object of class `"numeric"`, a vector of tag counts of ChIP sample.

`mappability`: Object of class `"numeric"`, a vector of mappability score.

`gcContent`: Object of class `"numeric"`, a vector of GC content score.

`input`: Object of class `"numeric"`, a vector of tag counts of matched control sample.

`bic1S`: Object of class `"numeric"`, Bayesian Information Criterion (BIC) value of one-signal-component model.

`bic2S`: Object of class `"numeric"`, Bayesian Information Criterion (BIC) value of two-signal-component model.

`seqDepth`: Object of class `"numeric"`, a vector of sequencing depth of length 2, where the first and second elements correpond to sequencing depths of ChIP and control samples, respectively. If there is not control sample, the second element is set to NA.

Methods

estimates `signature(object = "MosaicsFit")`: extract estimates from MOSAiCS model fit.

mosaicsPeak `signature(object = "MosaicsFit")`: call peaks using MOSAiCS model fit.

plot `signature(x = "MosaicsFit", y = "missing")`: draw Goodness of Fit (GOF) plot.

print `signature(x = "MosaicsFit")`: (not supported yet)

show `signature(object = "MosaicsFit")`: provide brief summary of the object.

seqDepth `signature(object = "MosaicsFit")`: provide a vector of sequencing depth.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`mosaicsFit`, `mosaicsPeak`, `mosaicsFitHMM`, `estimates`.

Examples

```
showClass("MosaicsFit")
## Not run:
library(mosaicsExample)

data(exampleBinData)
exampleFit <- mosaicsFit( exampleBinData, analysisType="IO" )

exampleFit
plot(exampleFit)
estimates(exampleFit)

examplePeak <- mosaicsPeak( exampleFit, signalModel = "2S", FDR = 0.05 )

## End(Not run)
```

MosaicsFitEst-class *Class "MosaicsFitEst"*

Description

This class represents estimates of mosaicsFit function.

Objects from the Class

Objects can be created by calls of the form `new("MosaicsFitEst", ...)`.

Slots

`pi0`: Object of class `"numeric"`, representing mixing proportion of background component.

`a`: Object of class `"numeric"`, representing parameter for background component.

`betaEst`: Object of class `"numeric"`, representing parameter for background component (coefficient estimates).

`muEst`: Object of class `"numeric"`, representing parameter for background component.

`pNfit`: Object of class `"list"`, representing background model fit.

`b`: Object of class `"numeric"`, representing parameter for one-signal-component model.

`c`: Object of class `"numeric"`, representing parameter for one-signal-component model.

`p1`: Object of class `"numeric"`, representing parameter for two-signal-component model (mixing proportion of signal components).

`b1`: Object of class `"numeric"`, representing parameter for two-signal-component model (the first signal component).

`c1`: Object of class `"numeric"`, representing parameter for two-signal-component model (the first signal component).

`b2`: Object of class `"numeric"`, representing parameter for two-signal-component model (the second signal component).

`c2`: Object of class `"numeric"`, representing parameter for two-signal-component model (the second signal component).

`inputTrunc`: Object of class `"numeric"`, representing parameter for input-only analysis (threshold for maximum tag count for matched control sample.

`analysisType`: Object of class `"character"`, representing type of data analysis.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`mosaicsFit`.

Examples

```
showClass("MosaicsFitEst")
```

`mosaicsFitHMM` *Fit MOSAiCS-HMM model*

Description

Fit MOSAiCS-HMM model.

Usage

```
mosaicsFitHMM( object, ... )
## S4 method for signature 'MosaicsFit'
mosaicsFitHMM( object, signalModel="2S", binsize=NA,
 init="mosaics", init.FDR=0.05,
 init.maxgap=200, init.minsize=50, init.thres=10, init.piMat=as.matrix(NA),
 max.iter=100, eps=1e-20, parallel=FALSE, nCore=8 )
```

Arguments

`object`	Object of class `MosaicsFit`, a fitted MOSAiCS model obtained using function `mosaicsFit`.
`signalModel`	Signal model. Possible values are "1S" (one-signal-component model) and "2S" (two-signal-component model). Default is "2S".
`binsize`	Size of each bin. Value should be positive integer. If `binsize=NA`, `mosaicsFitHMM` function calcuates the value from data. Default is `NA`.
`init`	Approach to initialize MOSAiCS-HMM. Possible values are `"mosaics"` (use MOSAiCS peak calling results for initialization) or `"specify"` (explicitly specify transition matrix). Default is `"mosaics"`.
`init.FDR`	Parameter for the MOSAiCS-HMM initialization. False discovery rate. Default is 0.05. Related only if `init="mosaics"`.
`init.maxgap`	Parameter for the MOSAiCS-HMM initialization. Initial nearby peaks are merged if the distance (in bp) between them is less than `init.maxgap`. Default is 200. Related only if `init="mosaics"`.
`init.minsize`	Parameter for the MOSAiCS-HMM initialization. An initial peak is removed if its width is narrower than `init.minsize`. Default is 50. Related only if `init="mosaics"`.
`init.thres`	Parameter for the MOSAiCS-HMM initialization. A bin within initial peak is removed if its ChIP tag counts are less than `init.thres`. Default is 10. Related only if `init="mosaics"`.
`init.piMat`	Initial value for transition matrix. The first rows/columns correspond to the non-binding state while the second rows/columns correspond to the binding state. Related only if `init="specify"`. If `init="specify"` but `init.piMat` is not specified, `mosaicsFitHMM()` uses its default for the MOSAiCS-HMM initialization.
`max.iter`	Number of iterations for fitting MOSAiCS-HMM. Default is 100.

eps	Criterion to stop iterations for fitting MOSAiCS-HMM. Default is 1e-20.
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).
nCore	Number of CPUs when parallel computing is utilized.
...	Other parameters to be passed through to generic mosaicsFitHMM.

Details

mosaicsFitHMM and mosaicsPeakHMM are developed to identify broad peaks such as histone modifications, using Hidden Markov Model (HMM) approach, as proposed in Chung et al. (2014). If you are interested in identifying narrow peaks such as transcription factor binding sites, please use mosaicsPeak instead of mosaicsFitHMM and mosaicsPeakHMM.

When peaks are called, proper signal model needs to be specified. The optimal choice for the number of signal components depends on the characteristics of ChIP-seq data. In order to support users in the choice of optimal signal model, Bayesian Information Criterion (BIC) values and Goodness of Fit (GOF) plot are provided for the fitted MOSAiCS model. BIC values and GOF plot can be obtained by applying show and plot methods, respectively, to the MosaicsFit class object, which is a fitted MOSAiCS model.

init.FDR, init.maxgap, init.minsize, and init.thres are the parameters for MOSAiCS-HMM initialization when MOSAiCS peak calling results are used for initialization (init="mosaics"). If user specifies transition matrix (init="specify"), only init.piMat is used for initialization. If you use a bin size shorter than the average fragment length of the experiment, we recommend to set init.maxgap to the average fragment length and init.minsize to the bin size. If you set the bin size to the average fragment length or if bin size is larger than the average fragment length, set init.maxgap to the average fragment length and init.minsize to a value smaller than the average fragment length. See the vignette for further details.

Parallel computing can be utilized for faster computing if parallel=TRUE and parallel package is loaded. nCore determines number of CPUs used for parallel computing.

Value

Construct MosaicsHMM class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsFit, mosaicsPeakHMM, MosaicsFit, MosaicsHMM.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binHM
plot(binHM)
plot( binHM, plotType="input" )

fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
fitHM
plot(fitHM)

hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
hmmHM
plot(hmmHM)

peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM

peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
head(print(peakHM))
plot( peakHM, filename="./peakplot_HM.pdf" )

peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
```

```
peakHM
head(print(peakHM))

export( peakHM, type = "txt", filename = "./peakHM.txt" )
export( peakHM, type = "bed", filename = "./peakHM.bed" )
export( peakHM, type = "gff", filename = "./peakHM.gff" )
export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

`MosaicsFitParam-class` *Class "MosaicsFitParam"*

Description

This class represents parameters for mosaicsFit function.

Objects from the Class

Objects can be created by calls of the form `new("MosaicsFitParam", ...)`.

Slots

`k`: Object of class `"numeric"`, representing a parameter for mosaicsFit function.

`meanThres`: Object of class `"numeric"`, representing a parameter for mosaicsFit function.

`s`: Object of class `"numeric"`, representing a parameter for mosaicsFit function.

`d`: Object of class `"numeric"`, representing a parameter for mosaicsFit function.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`mosaicsFit`.

Examples

```
showClass("MosaicsFitParam")
```

`MosaicsHMM-class` *Class "MosaicsHMM"*

Description

This class represents MOSAiCS-HMM model fit.

Objects from the Class

Objects can be created by calls of the form `new("MosaicsHMM", ...)`.

Slots

`HMMfit`: Object of class `"list"`, representing the fitted MOSAiCS-HMM model.

`chrID`: Object of class `"character"`, a vector of chromosome IDs.

`coord`: Object of class `"numeric"`, a vector of genomic coordinates.

`tagCount`: Object of class `"numeric"`, a vector of tag counts of ChIP sample.

`mappability`: Object of class `"numeric"`, a vector of mappability score.

`gcContent`: Object of class `"numeric"`, a vector of GC content score.

`input`: Object of class `"numeric"`, a vector of tag counts of matched control sample.

`inputdata`: Object of class `"list"`, representing the bin-level data.

`mosaicsEst`: Object of class `"MosaicsFitEst"`, representing estimates of MOSAiCS model fit.

`init`: Object of class `"character"`, representing the approach to initialize MOSAiCS-HMM.

`initPiMat`: Object of class `"numeric"`, representing initial transition matrix.

`peakParam`: Object of class `"MosaicsPeakParam"`, representing parameters for peak calling.

`binsize`: Object of class `"numeric"`, representing size of a bin.

`nRatio`: Object of class `"numeric"`, representing ratio of sequencing depth of ChIP vs. control.

`bicMosaics`: Object of class `"numeric"`, representing the BIC value of MOSAiCS fit.

`bicMosaicsHMM`: Object of class `"numeric"`, representing the BIC value of MOSAiCS-HMM fit.

`seqDepth`: Object of class `"numeric"`, a vector of sequencing depth of length 2, where the first and second elements correpond to sequencing depths of ChIP and control samples, respectively. If there is not control sample, the second element is set to NA.

Methods

estimates `signature(object = "MosaicsHMM")`: extract estimates from MOSAiCS-HMM model fit. =

plot `signature(x = "MosaicsHMM", y = "missing", seed=12345, parallel=FALSE, nCore=8 )`: draw Goodness of Fit (GOF) plot. You can specify random seed in `seed`. If `parallel=TRUE`, parallel computing is utilized to simulate data, where `nCore` indicates CPUs used for parallel computing.

print `signature(x = "MosaicsHMM")`: (not supported yet)

show `signature(object = "MosaicsHMM")`: provide brief summary of the object.

seqDepth `signature(object = "MosaicsHMM")`: provide a vector of sequencing depth.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsFitHMM, mosaicsPeakHMM.

Examples

```
showClass("MosaicsHMM")
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binHM
plot(binHM)
plot( binHM, plotType="input" )

fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
fitHM
plot(fitHM)

hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
hmmHM
plot(hmmHM)

peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
```

```
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM

peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
head(print(peakHM))
plot( peakHM, filename="./peakplot_HM.pdf" )

peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

export( peakHM, type = "txt", filename = "./peakHM.txt" )
export( peakHM, type = "bed", filename = "./peakHM.bed" )
export( peakHM, type = "gff", filename = "./peakHM.gff" )
export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

`mosaicsPeak` *Call peaks using fitted MOSAiCS model*

Description

Call peaks using `MosaicsFit` class object, which is a fitted MOSAiCS model.

Usage

```
mosaicsPeak( object, ... )
## S4 method for signature 'MosaicsFit'
mosaicsPeak( object, signalModel="2S", FDR=0.05,
    binsize=NA, maxgap=200, minsize=50, thres=10 )
```

Arguments

`object`	Object of class `MosaicsFit`, a fitted MOSAiCS model obtained using function `mosaicsFit`.
`signalModel`	Signal model. Possible values are "1S" (one-signal-component model) and "2S" (two-signal-component model). Default is "2S".
`FDR`	False discovery rate. Default is 0.05.
`binsize`	Size of each bin. Value should be positive integer. If `binsize=NA`, `mosaicsPeak` function calcuates the value from data. Default is `NA`.

maxgap	Initial nearby peaks are merged if the distance (in bp) between them is less than maxgap. Default is 200.
minsize	An initial peak is removed if its width is narrower than minsize. Default is 50.
thres	A bin within initial peak is removed if its ChIP tag counts are less than thres. Default is 10.
...	Other parameters to be passed through to generic mosaicsPeak.

Details

mosaicsPeak is developed to identify narrow peaks such as transcription factor binding sites. If you are interested in identifying broad peaks such as histone modifications, please use mosaicsFitHMM and mosaicsPeakHMM instead of mosaicsPeak.

When peaks are called, proper signal model needs to be specified. The optimal choice for the number of signal components depends on the characteristics of ChIP-seq data. In order to support users in the choice of optimal signal model, Bayesian Information Criterion (BIC) values and Goodness of Fit (GOF) plot are provided for the fitted MOSAiCS model. BIC values and GOF plot can be obtained by applying show and plot methods, respectively, to the MosaicsFit class object, which is a fitted MOSAiCS model.

maxgap, minsize, and thres are for refining initial peaks called using specified signalModel and FDR. If you use a bin size shorter than the average fragment length of the experiment, we recommend to set maxgap to the average fragment length and minsize to the bin size. If you set the bin size to the average fragment length or if bin size is larger than the average fragment length, set maxgap to the average fragment length and minsize to a value smaller than the average fragment length. See the vignette for further details.

Value

Construct MosaicsPeak class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsFit, MosaicsPeak, MosaicsFit.

Examples

```
## Not run:
library(mosaicsExample)
data(exampleBinData)
exampleFit <- mosaicsFit( exampleBinData, analysisType="IO" )
examplePeak <- mosaicsPeak( exampleFit, signalModel = "2S", FDR = 0.05 )

## End(Not run)
```

MosaicsPeak-class *Class "MosaicsPeak"*

Description

This class represents peak calling results.

Objects from the Class

Objects can be created by calls of the form `new("MosaicsPeak", ...)`.

Slots

peakList: Object of class `"data.frame"`, representing peak list.

chrID: Object of class `"character"`, a vector of chromosome IDs.

coord: Object of class `"numeric"`, a vector of genomic coordinates.

tagCount: Object of class `"numeric"`, a vector of tag counts of ChIP sample.

mappability: Object of class `"numeric"`, a vector of mappability score.

gcContent: Object of class `"numeric"`, a vector of GC content score.

input: Object of class `"numeric"`, a vector of tag counts of matched control sample.

peakParam: Object of class `"MosaicsPeakParam"`, representing parameters for peak calling.

bdBin: Object of class `"numeric"`, representing a vector of bounded bins.

empFDR: Object of class `"numeric"`, representing empirical FDR.

postProb: Object of class `"numeric"`, representing posterior probability that a bin belongs to background.

tagLoaded: Object of class `"logical"`, representing whether read-level data is loaded.

tagData: Object of class `"TagData"`, representing read-level data.

seqDepth: Object of class `"numeric"`, a vector of sequencing depth of length 2, where the first and second elements correpond to sequencing depths of ChIP and control samples, respectively. If there is not control sample, the second element is set to NA.

Methods

export `signature(object = "MosaicsPeak")`: export peak list into text files.

print `signature(x = "MosaicsPeak")`: return peak list in data frame format.

plot `signature( x = "MosaicsPeak", y = "missing", filename=NA, peakNum=NA )`: plot ChIP profile in each peak region. If file name is specified in `filename`, plots are exported to a PDF file named `filename`. Oterwise, ChIP profiles are plotted to standard output. If users are interested in specific peak regions, users can specify them as a vector in `peakNum`, where elements indicate which rows peak regions of interest are located. If `peakNum` is specified, only ChIP profiles in specified peak regions are plotted. Otherwise, ChIP profiles for all the peak regions are plotted.

show `signature(object = "MosaicsPeak")`: provide brief summary of the object.

empFDR `signature(object = "MosaicsPeak")`: return estimated empirical false discovery rate.

bdBin `signature(object = "MosaicsPeak")`: return a vector of bin-level binary indicator of peak calling, where each element is 1 if the corresponding bin belongs to a peak and zero otherwise.

readCoverage `signature(object = "MosaicsPeak")`: export a list of coverage matrices of ChIP and matched control samples, where each matrix consists of two columns, genomic coordinates and read count. Each element of this list is a list of length 2, which are matrices for each of ChIP and matched control samples. Elements of this list are sorted to match the rows of peak list. Users can use `readCoverage` method only after `extractReads` method is applied to the `MosaicsPeak` object.

read `signature(object = "MosaicsPeak")`: return a list of read-level data of ChIP and matched control samples, where each element is `GenomicRanges` class. Each element of this list is a list of length 2, which are `GenomicRanges` objects for each of ChIP and matched control samples. Elements of this list are sorted to match the rows of peak list. Users can use `read` method only after `extractReads` method is applied to the `MosaicsPeak` object.

seqDepth `signature(object = "MosaicsPeak")`: return a vector of sequencing depth, where the first and second elements are for ChIP and matched control sample, respectively. If matched control sample is not provided, the second element is NA. Users can use `seqDepth` method only after `extractReads` method is applied to the `MosaicsPeak` object.

postProb `signature( object = "MosaicsPeak", peakRegion=NULL, summaryStat="aveLogP", parallel=FALSE, nCore=8 )`: return a data frame of peak-level posterior probabilities of being background (PP) for arbitrary peak regions, where the columns are chromosome ID, peak start position, and peak end position, and summary of PP. Peak regions of interest can be specified in `peakRegion` argument and a data frame of three columns (chromosome ID, peak start position, and peak end position) is expected. If `peakRegion` is not provided, a data frame of bin-level PP is provided, where columns are chromosome ID, genomic coordinate, and PP. Summary statistics can be chosen among `"aveLogP"` (average of -log10 transformed PP; default), `"medianLogP"` (median of -log10 transformed PP), `"sumLogP"` (sum of -log10 transformed PP), `"logMinP"` (-log10 transformation of minimum PP), `"logAveP"` (-log10 transformation of average PP), and `"logMedianP"` (-log10 transformation of median PP). Multiple CPUs can be used for parallel computing using `"parallel"` package if `parallel=TRUE`, where the number of cores to be used can be specified using `nCore`.

seqDepth `signature(object = "MosaicsPeak")`: provide a vector of sequencing depth.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, mosaicsPeakHMM, export, extractReads, findSummit, adjustBoundary, filterPeak.

Examples

```
showClass("MosaicsPeak")
## Not run:
library(mosaicsExample)

# example analysis workflow for ChIP-seq data of transcription factor binding
# (STAT1 factor binding in GM12878 cell line, from ENCODE database)

generateWig( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam")
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, span=200, capping=0, normConst=1 )

constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binTFBS <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
    "./wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binTFBS
head(print(binTFBS))
plot(binTFBS)
plot( binTFBS, plotType="input" )

fitTFBS <- mosaicsFit( binTFBS, analysisType="IO", bgEst="rMOM" )
fitTFBS
plot(fitTFBS)

peakTFBS <- mosaicsPeak( fitTFBS, signalModel="2S", FDR=0.05,
maxgap=200, minsize=50, thres=10 )
peakTFBS
```

```
head(print(peakTFBS))

peakTFBS <- extractReads( peakTFBS,
 chipFile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam"), package=
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam"), packa
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakTFBS

peakTFBS <- findSummit( peakTFBS, parallel=FALSE, nCore=8 )

export( peakTFBS, type = "txt", filename = "./peakTFBS.txt" )
export( peakTFBS, type = "bed", filename = "./peakTFBS.bed" )
export( peakTFBS, type = "gff", filename = "./peakTFBS.gff" )
export( peakTFBS, type = "narrowPeak", filename = "./peakTFBS.narrowPeak" )

# example analysis workflow for ChIP-seq data of histone modification
# (H3K4me3 modification in GM12878 cell line, from ENCODE database)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binHM
plot(binHM)
plot( binHM, plotType="input" )

fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
fitHM
plot(fitHM)

hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
hmmHM
plot(hmmHM)

peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM
```

```
peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
head(print(peakHM))
plot( peakHM, filename="./peakplot_HM.pdf" )

peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

export( peakHM, type = "txt", filename = "./peakHM.txt" )
export( peakHM, type = "bed", filename = "./peakHM.bed" )
export( peakHM, type = "gff", filename = "./peakHM.gff" )
export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

`mosaicsPeakHMM` *Call broad peaks using fitted MOSAiCS-HMM model*

Description

Call broad peaks using `MosaicsHMM` class object, which is a fitted MOSAiCS-HMM model.

Usage

```
mosaicsPeakHMM( object, ... )
## S4 method for signature 'MosaicsHMM'
mosaicsPeakHMM( object, FDR=0.05, decoding="posterior",
    binsize=NA, maxgap=0, minsize=0, thres=0,
    parallel=FALSE, nCore=8 )
```

Arguments

`object`	Object of class `MosaicsHMM`, a fitted MOSAiCS model obtained using function `mosaicsFitHMM`.
`FDR`	False discovery rate. Default is 0.05. Not relevant when `decoding="viterbi"`.
`decoding`	Approach to determine the undelying state. Possible values are "viterbi" (Viterbi algorithm) and "posterior" (posterior decoding). Default is "posterior".
`binsize`	Size of each bin. Value should be positive integer. If `binsize=NA`, `mosaicsPeakHMM` function calcuates the value from data. Default is `NA`.
`maxgap`	Initial nearby peaks are merged if the distance (in bp) between them is less than `maxgap`. Default is 0.

minsize	An initial peak is removed if its width is narrower than minsize. Default is 0.
thres	A bin within initial peak is removed if its ChIP tag counts are less than thres. Default is 0.
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (utilize multiple CPUs) or FALSE (do not utilize multiple CPUs). Default is FALSE (do not utilize multiple CPUs).
nCore	Number of CPUs when parallel computing is utilized.
...	Other parameters to be passed through to generic mosaicsHMM.

Details

mosaicsFitHMM and mosaicsPeakHMM are developed to identify broad peaks such as histone modifications, using Hidden Markov Model (HMM) approach, as proposed in Chung et al. (2014). If you are interested in identifying narrow peaks such as transcription factor binding sites, please use mosaicsPeak instead of mosaicsFitHMM and mosaicsPeakHMM.

maxgap, minsize, and thres are for refining initial peaks called using specified decoding (and FDR if decoding="posterior"). If you use a bin size shorter than the average fragment length of the experiment, we recommend to set maxgap to the average fragment length and minsize to the bin size. If you set the bin size to the average fragment length or if bin size is larger than the average fragment length, set maxgap to the average fragment length and minsize to a value smaller than the average fragment length. See the vignette for further details.

Parallel computing can be utilized for faster computing if parallel=TRUE and parallel package is loaded. nCore determines number of CPUs used for parallel computing.

Value

Construct MosaicsPeak class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsFitHMM, extractReads, findSummit, adjustBoundary, filterPeak, MosaicsHMM, MosaicsPeak.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sort
    fileFormat="bam", outfileLoc="./",
    byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binHM <- readBins( type=c("chip","input"),
   fileName=c( "./wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
   "./wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )
binHM
plot(binHM)
plot( binHM, plotType="input" )

fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
fitHM
plot(fitHM)

hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
  init="mosaics", init.FDR = 0.05, parallel=TRUE, nCore=8 )
hmmHM
plot(hmmHM)

peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
  thres=10, parallel=TRUE, nCore=8 )

peakHM <- extractReads( peakHM,
 chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam"), pa
  chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
 controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam")
  controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=TRUE, nCore=8 )
peakHM

peakHM <- findSummit( peakHM, parallel=TRUE, nCore=8 )
head(print(peakHM))
plot( peakHM, filename="./peakplot_HM.pdf" )

peakHM <- adjustBoundary( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

peakHM <- filterPeak( peakHM, parallel=TRUE, nCore=8 )
peakHM
head(print(peakHM))

export( peakHM, type = "txt", filename = "./peakHM.txt" )
```

```
export( peakHM, type = "bed", filename = "./peakHM.bed" )
export( peakHM, type = "gff", filename = "./peakHM.gff" )
export( peakHM, type = "narrowPeak", filename = "./peakHM.narrowPeak" )
export( peakHM, type = "broadPeak", filename = "./peakHM.broadPeak" )

## End(Not run)
```

MosaicsPeakParam-class *Class "MosaicsPeakParam"*

Description

This class represents parameters for mosaicsPeak function.

Objects from the Class

Objects can be created by calls of the form `new("MosaicsPeakParam", ...)`.

Slots

`analysisType`: Object of class `"character"`, representing type of data analysis.

`signalModel`: Object of class `"character"`, representing signal model for peak calling.

`FDR`: Object of class `"numeric"`, representing specified false discovery rate.

`maxgap`: Object of class `"numeric"`, representing maximum gap size to combine two neighboring initial peaks.

`minsize`: Object of class `"numeric"`, representing minimum peak size to be remained in the final peak list.

`thres`: Object of class `"numeric"`, representing threshold of ChIP tag count to filter initial peaks.

`decoding`: Object of class `"character"`, representing deconding approach for MOSAiCS-HMM peak calling.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

mosaicsPeak, mosaicsPeakHMM, MosaicsPeak.

Examples

```
showClass("MosaicsPeakParam")
```

---

mosaicsRunAll *Analyze ChIP-seq data using the MOSAiCS framework*

Description

Construct bin-level ChIP-sep data from aligned read files of ChIP and matched control samples, fit a MOSAiCS model, call peaks, export peak calling results, and generate reports for diagnostics.

Usage

```
mosaicsRunAll(
    chipFile=NULL, chipFileFormat=NULL,
    controlFile=NULL, controlFileFormat=NULL,
    binfileDir=NULL,
    peakFile=NULL, peakFileFormat=NULL,
    reportSummary=FALSE, summaryFile=NULL,
    reportExploratory=FALSE, exploratoryFile=NULL,
    reportGOF=FALSE, gofFile=NULL,
    PET=FALSE, byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
    FDR=0.05, fragLen=200, binSize=200, capping=0, bgEst="rMOM", d=0.25,
    signalModel="BIC", maxgap=200, minsize=50,
    thres=10, parallel=FALSE, nCore=8 )
```

Arguments

chipFile	Name of the aligned read file of ChIP sample to be processed.
chipFileFormat	Format of the aligned read file of ChIP sample to be processed. Currently, mosaicsRunAll permits the following aligned read file formats: "eland_result" (Eland result), "eland_extended" (Eland extended), "eland_export" (Eland export), "bowtie" (default Bowtie), "sam" (SAM), "bed" (BED), and "csem" (CSEM BED). Note that "csem" does not mean CSEM output file format, but CSEM BED file format.
controlFile	Name of the aligned read file of matched control sample to be processed.
controlFileFormat	Format of the aligned read file of matched control sample to be processed. Currently, mosaicsRunAll permits the following aligned read file formats: "eland_result" (Eland result), "eland_extended" (Eland extended), "eland_export" (Eland export), "bowtie" (default Bowtie), "sam" (SAM), "bed" (BED), and "csem" (CSEM BED). Note that "csem" does not mean CSEM output file format, but CSEM BED file format.

`binfileDir` Directory to store processed bin-level files.

`peakFile` Name of the peak list generated from the analysis.

`peakFileFormat` Format of the peak list generated from the analysis. Possible values are `"txt"`, `"bed"`, and `"gff"`.

`reportSummary` Report the summary of model fitting and peak calling? Possible values are `TRUE` (YES) and `FALSE` (NO). Default is `FALSE` (NO).

`summaryFile` File name of the summary report of model fitting and peak calling. The summary report is a text file.

`reportExploratory` Report the exploratory analysis plots? Possible values are `TRUE` (YES) and `FALSE` (NO). Default is `FALSE` (NO).

`exploratoryFile` Name of the file for exploratory analysis plots. The exploratory analysis results are exported as a PDF file.

`reportGOF` Report the goodness of fit (GOF) plots? Possible values are `TRUE` (YES) and `FALSE` (NO). Default is `FALSE` (NO).

`gofFile` Name of the file for goodness of fit (GOF) plots. The GOF plots are exported as a PDF file.

`PET` Is the file paired-end tag (PET) data? If `PET=FALSE`, it is assumed that the file is SET data. If `PET=TRUE`, it is assumed that the file is PET data. Default is `FALSE` (SET data).

`byChr` Analyze ChIP-seq data for each chromosome separately or analyze it genome-wide? Possible values are `TRUE` (chromosome-wise) and `FALSE` (genome-wide). Default is `FALSE` (genome-wide analysis).

`useChrfile` Is the file for chromosome info provided? Possible values are `TRUE` or `FALSE`. If `useChrfile=FALSE`, it is assumed that the file for chromosome info is not provided. If `useChrfile=TRUE`, it is assumed that the file for chromosome info is provided. Default is `FALSE`.

`chrfile` Name of the file for chromosome info. In this file, the first and second columns are ID and size of each chromosome, respectively.

`excludeChr` Vector of chromosomes that will be excluded from the analysis.

`FDR` False discovery rate. Default is 0.05.

`fragLen` Average fragment length. Default is 200.

`binSize` Size of bins. Default is 200.

`capping` Maximum number of reads allowed to start at each nucleotide position. To avoid potential PCR amplification artifacts, the maximum number of reads that can start at a nucleotide position is capped at `capping`. Capping is not applied if non-positive `capping` is used. Default is 0 (no capping).

`bgEst` Parameter to determine background estimation approach. Possible values are "matchLow" (estimation using bins with low tag counts) and "rMOM" (estimation using robust method of moment (MOM)). If `bgEst="automatic"`, this method tries to make the best guess for `bgEst`, based on the data provided. Default is `bgEst="rMOM"`.

d	Parameter for estimating background distribution. Default is 0.25.
signalModel	Signal model. Possible values are "BIC" (automatic model selection using BIC), "1S" (one-signal-component model), and "2S" (two-signal-component model). Default is "BIC".
maxgap	Initial nearby peaks are merged if the distance (in bp) between them is less than maxgap. Default is 200.
minsize	An initial peak is removed if its width is narrower than minsize. Default is 50.
thres	A bin within initial peak is removed if its ChIP tag counts are less than thres. Default is 10.
parallel	Utilize multiple CPUs for parallel computing using "parallel" package? Possible values are TRUE (use multiple CPUs) or FALSE (do not use multiple CPUs). Default is FALSE (do not use multiple CPUs).
nCore	Number of maximum number of CPUs used for the analysis. Default is 8.

Details

This method implements the work flow for the two-sample analysis of ChIP-seq data using the MOSAiCS framework (without using mappability and GC content scores). It imports aligned read files of ChIP and matched control samples, processes them into bin-level files, fits MOSAiCS model, calls peaks, exports the peak lists to text files, and generates reports for diagnostics. This method is a wrapper function of constructBins, readBins, mosaicsFit, mosaicsPeak, export functions, and methods of BinData, MosaicsFit, and MosaicsPeak classes.

See the vignette of the package for the illustration of the work flow and the description of employed methods and their options. Exploratory analysis plots and goodness of fit (GOF) plots are generated using the methods plot of the classes BinData and MosaicsFit, respectively. See the help of constructBins for details of the options PET, chipFileFormat, controlFileFormat, byChr, useChrfile, chrfile, excludeChr, fragLen, binSize, and capping. See the help of mosaicsFit for details of the options bgEst and d. See the help of mosaicsPeak for details of the options FDR, signalModel, maxgap, minsize, and thres. See the help of export for details of the option peakFileFormat.

When the data contains multiple chromosomes, parallel computing can be utilized for faster preprocessing and model fitting if parallel=TRUE and parallel package is loaded. nCore determines number of CPUs used for parallel computing.

Value

Processed bin-level files are exported to the directory specified in binfileDir argument. If byChr=FALSE (genome-wide analysis), one bin-level file is generated for each of ChIP and matched control samples, where file names are [chipFile]_fragL[fragLen]_bin[binSize].txt and [controlFile]_fragL[fragLen]_bin[respectively, for SET data (PET = FALSE). For PET data (PET = TRUE), file names for each of ChIP and matched control samples are [chipFile]_bin[binSize].txt and [controlFile]_bin[binSize].txt, respectively. If byChr=TRUE (chromosome-wise analysis), bin-level files are generated for each chromosome of each of ChIP and matched control samples, where file names are [chipFile]_fragL[fragLen]_bin[binSiz and [controlFile]_fragL[fragLen]_bin[binSize]_[chrID].txt, respectively, for SET data (PET = FALSE) ([chrID] is chromosome IDs that reads align to). For PET data (PET = TRUE), file names for each of ChIP and matched control samples are [chipFile]_bin[binSize]_[chrID].txt and [controlFile]_bin[binSize]_[chrID].txt, respectively.

The peak list generated from the analysis are exported to the file with the name specified in `peakFile`. If `reportSummary=TRUE`, the summary of model fitting and peak calling is exported to the file with the name specified in `summaryFile` (text file). If `reportExploratory=TRUE`, the exploratory analysis plots are exported to the file with the name specified in `exploratoryFile` (PDF file). If `reportGOF=TRUE`, the goodness of fit (GOF) plots are exported to the file with the name specified in `gofFile` (PDF file).

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`constructBins`, `readBins`, `mosaicsFit`, `mosaicsPeak`, `export`, `BinData`, `MosaicsFit`, `MosaicsPeak`.

Examples

```
## Not run:
# minimal input (without any reports for diagnostics)

mosaicsRunAll(
    chipFile = "/scratch/eland/STAT1_eland_results.txt",
    chipFileFormat = "eland_result",
    controlFile = "/scratch/eland/input_eland_results.txt",
    controlFileFormat = "eland_result",
    binfileDir = "/scratch/bin/",
    peakFile = "/scratch/peak/STAT1_peak_list.bed",
    peakFileFormat = "bed" )

# generate all reports for diagnostics

library(parallel)
mosaicsRunAll(
    chipFile = "/scratch/eland/STAT1_eland_results.txt",
    chipFileFormat = "eland_result",
    controlFile = "/scratch/eland/input_eland_results.txt",
    controlFileFormat = "eland_result",
    binfileDir = "/scratch/bin/",
    peakFile = "/scratch/peak/STAT1_peak_list.bed",
    peakFileFormat = "bed",
    reportSummary = TRUE,
    summaryFile = "/scratch/reports/mosaics_summary.txt",
```

```
    reportExploratory = TRUE,
    exploratoryFile = "/scratch/reports/mosaics_exploratory.pdf",
    reportGOF = TRUE,
    gofFile = "/scratch/reports/mosaics_GOF.pdf",
    PET = FALSE, byChr = FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr = "chrM",
    FDR = 0.05,  fragLen = 200, capping = 0, bgEst="automatic", thres=10,
    parallel = TRUE, nCore = 8 )

## End(Not run)
```

`readBins` *Import bin-level ChIP-sep data*

Description

Import and preprocess all or subset of bin-level ChIP-sep data, including ChIP data, matched control data, mappability score, GC content score, and sequence ambiguity score.

Usage

```
readBins( type = c("chip", "input"), fileName = NULL,
    dataType = "unique", rounding = 100, parallel=FALSE, nCore=8 )
```

Arguments

`type`	Character vector indicating data types to be imported. This vector can contain `"chip"` (ChIP data), `"input"` (matched control data), `"M"` (mappability score), `"GC"` (GC content score), and `"N"` (sequence ambiguity score). Currently, `readBins` permits only the following combinations: `c("chip", "input")`, `c("chip", "input", "N")`, `c("chip", "input", "M", "GC", "N")`, and `c("chip", "M", "GC", "N")`. Default is `c("chip", "input")`.
`fileName`	Character vector of file names, each of which matches each element of `type`. `type` and `fileName` should have the same length and corresponding elements in two vectors should appear in the same order.
`dataType`	How reads were processed? Possible values are either `"unique"` (only uniquely aligned reads were retained) or `"multi"` (reads aligned to multiple locations were also retained).
`rounding`	How are mappability score and GC content score rounded? Default is 100 and this indicates rounding of mappability score and GC content score to the nearest hundredth.
`parallel`	Utilize multiple CPUs for parallel computing using `"paralle"` package? Possible values are `TRUE` (use multiple CPUs) or `FALSE` (do not use multiple CPUs). Default is `FALSE` (do not use multiple CPUs).
`nCore`	Number of CPUs when parallel computing is utilized.

Details

Bin-level ChIP and matched control data can be generated from the aligned read files for your samples using the method `constructBins`. In `mosaics` package companion website, http://www.stat.wisc.edu/~keles/Software/mosaics/, we provide preprocessed mappability score, GC content score, and sequence ambiguity score files for diverse reference genomes. Please check the website and the vignette for further details.

The imported data type constraints the analysis that can be implemented. If `type=c("chip", "input")` or `c("chip", "input", "N")`, only two-sample analysis without using mappability and GC content is allowed. For `type=c("chip", "input", "M", "GC", "N")`, user can do the one- or two-sample analysis. If `type=c("chip", "M", "GC", "N")`, only one-sample analysis is permitted. See help page of `mosaicsFit`.

When the data contains multiple chromosomes, parallel computing can be utilized for faster preprocessing if `parallel=TRUE` and `parallel` package is loaded. `nCore` determines number of CPUs used for parallel computing.

Value

Construct `BinData` class object.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`constructBins`, `mosaicsFit`, `BinData`.

Examples

```
## Not run:
library(mosaicsExample)

constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, binSize=200, capping=0 )
constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam
    fileFormat="bam", outfileLoc="./",
    PET=FALSE, fragLen=200, binSize=200, capping=0 )

binTFBS <- readBins( type=c("chip","input"),
```

```
    fileName=c( "wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
      "wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt" ) )

## End(Not run)
```

`TagData-class` *Class "TagData"*

Description

This class represents read-level ChIP-seq data.

Objects from the Class

Objects can be created by calls of the form `new("TagData", ...)`.

Slots

`coverage`: Object of class `"list"`, a list of read coverage, where each element is a list including ChIP and matched control samples.

`numReads`: Object of class `"numeric"`, a matrix of number of reads in peak regions, where rows correspond to peak regions and columns correspond to ChIP and matched control samples.

`read`: Object of class `"list"`, a list of read-level data, where each element is a list including ChIP and matched control samples.

`keepReads`: Object of class `"logical"`, representing whether to keep read-level data.

Author(s)

Dongjun Chung, Pei Fen Kuan, Rene Welch, Sunduz Keles

References

Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keles (2011), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, Vol. 106, pp. 891-903.

Chung, D, Zhang Q, and Keles S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

See Also

`mosaicsPeak`, `mosaicsPeakHMM`, `extractReads`, `findSummit`, `filterPeak`, `MosaicsPeak`.

Examples

```
showClass("TagData")
```

Index

∗ **classes**
BinData-class, 8
MosaicsFit-class, 27
MosaicsFitEst-class, 28
MosaicsFitParam-class, 33
MosaicsHMM-class, 34
MosaicsPeak-class, 38
MosaicsPeakParam-class, 45
TagData-class, 52
∗ **methods**
adjustBoundary, 6
constructBins, 10
estimates, 12
export, 14
extractReads, 16
filterPeak, 19
findSummit, 21
generateWig, 22
mosaicsFit, 25
mosaicsFitHMM, 30
mosaicsPeak, 36
mosaicsPeakHMM, 42
mosaicsRunAll, 46
readBins, 50
∗ **models**
adjustBoundary, 6
constructBins, 10
estimates, 12
export, 14
extractReads, 16
filterPeak, 19
findSummit, 21
generateWig, 22
mosaicsFit, 25
mosaicsFitHMM, 30
mosaicsPeak, 36
mosaicsPeakHMM, 42
mosaicsRunAll, 46
readBins, 50
∗ **package**
mosaics-package, 2

adjustBoundary, *3*, 5, *18*, *20*, *22*, *40*, *43*
adjustBoundary,MosaicsPeak-method *(*adjustBoundary*)*, 6

bdBin *(*MosaicsPeak-class*)*, 38
bdBin,MosaicsPeak-method *(*MosaicsPeak-class*)*, 38
BinData, *3*, *11*, *49*, *51*
BinData-class, 8

chrID *(*BinData-class*)*, 8
chrID,BinData-method *(*BinData-class*)*, 8
constructBins, *3*, 10, *49*, *51*
coord *(*BinData-class*)*, 8
coord,BinData-method *(*BinData-class*)*, 8

empFDR *(*MosaicsPeak-class*)*, 38
empFDR,MosaicsPeak-method *(*MosaicsPeak-class*)*, 38
estimates, 12, *28*
estimates,MosaicsFit-method *(*estimates*)*, 12
estimates,MosaicsHMM-method *(*MosaicsHMM-class*)*, 34
export, 14, *18*, *22*, *40*, *49*
export,MosaicsPeak-method *(*export*)*, 14
extractReads, *3*, *7*, 16, *20*, *22*, *40*, *43*, *52*
extractReads,MosaicsPeak-method *(*extractReads*)*, 16

filterPeak, *3*, *7*, *18*, 19, *22*, *40*, *43*, *52*
filterPeak,MosaicsPeak-method *(*filterPeak*)*, 19
findSummit, *3*, *7*, *18*, *20*, 21, *40*, *43*, *52*
findSummit,MosaicsPeak-method *(*findSummit*)*, 21

gcContent *(*BinData-class*)*, 8

gcContent,BinData-method
(BinData-class), 8
generateWig, 22

input (BinData-class), 8
input,BinData-method (BinData-class), 8

mappability (BinData-class), 8
mappability,BinData-method
(BinData-class), 8
mosaics (mosaics-package), 2
mosaics-package, 2
MosaicsFit, *3*, *13*, *26*, *32*, *37*, *49*
mosaicsFit, *3*, *9*, *13*, 25, *28*, *29*, *32*, *33*, *37*, *49*, *51*
mosaicsFit,BinData-method (mosaicsFit), 25
MosaicsFit-class, 27
MosaicsFitEst-class, 28
mosaicsFitHMM, *3*, *26*, *28*, 30, *35*, *43*
mosaicsFitHMM,MosaicsFit-method
(mosaicsFitHMM), 30
MosaicsFitParam-class, 33
MosaicsHMM, *3*, *32*, *43*
MosaicsHMM-class, 34
MosaicsPeak, *3*, *7*, *15*, *18*, *20*, *22*, *37*, *43*, *46*, *49*, *52*
mosaicsPeak, *3*, *7*, *15*, *18*, *20*, *22*, *28*, 36, *40*, *46*, *49*, *52*
mosaicsPeak,MosaicsFit-method
(mosaicsPeak), 36
MosaicsPeak-class, 38
mosaicsPeakHMM, *3*, *7*, *18*, *20*, *22*, *32*, *35*, *40*, 42, *46*, *52*
mosaicsPeakHMM,MosaicsHMM-method
(mosaicsPeakHMM), 42
MosaicsPeakParam-class, 45
mosaicsRunAll, 46

plot,BinData,missing-method
(BinData-class), 8
plot,MosaicsFit,ANY-method
(MosaicsFit-class), 27
plot,MosaicsHMM,missing-method
(MosaicsHMM-class), 34
plot,MosaicsPeak,missing-method
(MosaicsPeak-class), 38
postProb (MosaicsPeak-class), 38
postProb,MosaicsPeak-method
(MosaicsPeak-class), 38
print,BinData-method (BinData-class), 8
print,MosaicsFit-method
(MosaicsFit-class), 27
print,MosaicsHMM-method
(MosaicsHMM-class), 34
print,MosaicsPeak-method
(MosaicsPeak-class), 38

read (MosaicsPeak-class), 38
read,MosaicsPeak-method
(MosaicsPeak-class), 38
readBins, *3*, *9*, *11*, *26*, *49*, 50
readCoverage (MosaicsPeak-class), 38
readCoverage,MosaicsPeak-method
(MosaicsPeak-class), 38

seqDepth (BinData-class), 8
seqDepth,BinData-method
(BinData-class), 8
seqDepth,MosaicsFit-method
(MosaicsFit-class), 27
seqDepth,MosaicsHMM-method
(MosaicsHMM-class), 34
seqDepth,MosaicsPeak-method
(MosaicsPeak-class), 38
show,BinData-method (BinData-class), 8
show,MosaicsFit-method
(MosaicsFit-class), 27
show,MosaicsHMM-method
(MosaicsHMM-class), 34
show,MosaicsPeak-method
(MosaicsPeak-class), 38

tagCount (BinData-class), 8
tagCount,BinData-method
(BinData-class), 8
TagData-class, 52