Package 'methyAnalysis'

October 9, 2015

Type Package

Title DNA methylation data analysis and visualization

Version 1.10.0

Date 2014-10-30

Depends R (>= 2.10), grid, BiocGenerics, S4Vectors, IRanges, GenomeInfoDb, GenomicRanges, Biobase (>= 2.5.5), org.Hs.eg.db

Imports lumi, methylumi, Gviz, genoset, GenomicRanges, VariantAnnotation, IRanges, rtracklayer, GenomicFeatures, annotate, Biobase (>= 2.5.5), AnnotationDbi, genefilter, biomaRt, methods, parallel

Suggests FDb.InfiniumMethylation.hg19, TxDb.Hsapiens.UCSC.hg19.knownGene

Author Pan Du, Richard Bourgon

Maintainer Pan Du <dupan.mail@gmail.com>

Description The methyAnalysis package aims for the DNA methylation data analysis and visualization. A MethyGenoSet class is defined to keep the chromosome location information together with the data. The package also includes functions of estimating the methylation levels from Methy-Seq data.

License Artistic-2.0

LazyLoad yes

biocViews Microarray, DNAMethylation, Visualization

Collate 'MethyGenoSet-class.R' 'methyAnalysis.R' 'heatmapByChromosome.R' 'methyl-seq.R'

NeedsCompilation no

R topics documented:

`annotateDMRInfo` *Annotate the DMR (Differentially Methylated Region)*

Description

Annotate the DMR (Differentially Methylated Region) information

Usage

```
annotateDMRInfo(DMRInfo, annotationDatabase, CpGInfo = NULL, flankRange = 500, promoterRange = 2000, E
```

Arguments

`DMRInfo`	A GRanges object or a list of GRanges objects (sigDMRInfo and sigDataInfo), which is the return of `identifySigDMR`
`annotationDatabase`	Annotation database: a TxDb package, TxDb object or GRanges object.
`CpGInfo`	A Bed file or GRanges object, which keeps the CpG-island information
`flankRange`	The flank range to be added to the input GRanges object
`promoterRange`	Define the size of promoter range at the upstream of TSS. User can also directly provide the GRanges object
`EntrezDB`	The Entrez database for mapping from Entrez ID to gene symbols
`as.GRanges`	Whether return a GRanges object or a data.frame

Details

This function is to annotate the DMRs to the gene promoters or bodies. The annotation information is attached as additional columns of the GRanges object values.

Value

Return a GRanges object or list of GRanges when the as.GRanges is TRUE. Or else it returns a data.frame or a list of data.frame objects (sigDMRInfo and sigDataInfo). The annotation information is attached as additional columns of the GRanges object values or the data.frame.

Author(s)

Pan Du

See Also

See Also annotateGRanges

Examples

```
data(exampleMethyGenoSet)
## get sample type information
sampleType <- pData(exampleMethyGenoSet)$SampleType

## Do differential test
allResult <- detectDMR.slideWin(exampleMethyGenoSet, sampleType=sampleType, testMethod='ttest')

## Identify the DMR (Differentially Methylated Region) by setting proper parameters.
## Here we just use default ones
allDMRInfo <- identifySigDMR(allResult)

## Annotate significant DMR info
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene)) {
DMRInfo.ann <- annotateDMRInfo(allDMRInfo, 'TxDb.Hsapiens.UCSC.hg19.knownGene')
}
```

---

annotateGRanges *Annotate a GRanges object*

Description

Annotate a GRanges object based on a transcription database

Usage

```
annotateGRanges(grange, annotationDatabase, CpGInfo = NULL, exons = FALSE, flankRange = 0, promoterRa
```

Arguments

grange	A GRanges object
annotationDatabase	Annotation database: a TxDb package, TxDb object or GRanges object.
CpGInfo	A Bed file or GRanges object, which keeps the CpG-island information
exons	Whether to annotate at the exon level. exons can be either TRUE/FALSE or a GRanges object represent the exon annotation.
flankRange	The flank range to be added to the input GRanges object
promoterRange	Define the size of promoter range at the upstream of TSS. Users can also directly provide the GRanges object
checkGeneBody	Determine whether to check the overlapping with gene body or just check the promoter region
EntrezDB	The Entrez database for mapping from Entrez ID to gene symbols

Details

This function is to annotate a GRanges object to the gene promoters or bodies. The annotation information is attached as additional columns of the GRanges object values.

Value

Return an annotated GRanges object with the annotation information attached as additional columns.

Author(s)

Pan Du

See Also

See Also annotateDMRInfo

Examples

```
data(exampleMethyGenoSet)
## get sample type information
sampleType <- pData(exampleMethyGenoSet)$SampleType

## Do differential test
allResult <- detectDMR.slideWin(exampleMethyGenoSet, sampleType=sampleType, testMethod='ttest')

## Identify the DMR (Differentially Methylated Region) by setting proper parameters.
## Here we just use default ones
allDMRInfo <- identifySigDMR(allResult)
sigDMRInfo <- allDMRInfo$sigDMRInfo
class(sigDMRInfo)

## Annotate significant DMR info
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene)) {
```

```
sigDMRInfo.ann <- annotateDMRInfo(sigDMRInfo, 'TxDb.Hsapiens.UCSC.hg19.knownGene')
}
```

---

asBigMatrix-methods *convert the data matrix in the assayData of a GenoSet as BigMatrix*

Description

convert the data matrix in the assayData of a GenoSet as BigMatrix

Usage

```
## S4 method for signature 'GenoSet'
asBigMatrix(object, rowInd=NULL, colInd=NULL, nCol=NULL, dimNames=NULL, saveDir='.', savePrefix=NULL
```

Arguments

object	an object of GenoSet or its inherited class
rowInd	the subset of row index
colInd	the subset of column index
nCol	the number of columns of the data, which can be larger than the real data dimension. It is designed for adding future data.
dimNames	the dimension names, which is a list of two character vectors (rownames and colnames)
saveDir	the parent directory to save the BigMatrix data files
savePrefix	the folder name prefix of the directory to save the BigMatrix data files. The fold name will be like this: paste(savePrefix, '_bigmat', sep='')
...	optional arguments to BigMatrix

Details

This function does not work in Windows because the dependent package bigmemoryExtras does not support it. In order to make lumi package still compilation under Windows, I deliberately remove the dependency of bigmemoryExtras package. As a result, users need to manually load the bigmemoryExtras function before using this function.

The BigMatrix data files will be save in the directory file.path(saveDir, paste(savePrefix, '_bigmat', sep='))

See Also

BigMatrix

`buildAnnotationTracks` *Build annotation tracks for visualizing using Gviz package*

Description

Build annotation tracks for visualizing using Gviz package

Usage

```
buildAnnotationTracks(gene, extendRange = c(2000, 2000), includeGeneBody = TRUE, cytobandInfo = NULL,
  lib = "org.Hs.eg.db", genome = "hg19", genomicFeature = "TxDb.Hsapiens.UCSC.hg19.knownGene", selectT
```

Arguments

`gene`	An Entrez gene id or a GRanges object with length equals one
`extendRange`	extended range on each side of the gene
`includeGeneBody`	whether to include genebody of the provided gene
`cytobandInfo`	cytoband information. Set NA to suppress it.
`CpGInfo`	CpG-island information, GRanges or bed file are supported
`genomeAxis`	whether to add genome axis or not
`lib`	gene annotation library
`genome`	genome version
`genomicFeature`	genomic features: "TxDb" library or object, "Mart" object
`selectTranscripts`	selected transcripts to show in the annotation track. If it is NULL, all transcripts will be shown.
`...`	other parameters used by createTranscriptTrack function

Details

This function aims to build annotation tracks to be visualized using Gviz package. If the cytobandInfo and CpGInfo are NULL and internet connection is available, it will download information directly from UCSC website. Set them as NAs if you want suppress this default behavior.

Value

A list of different annotation Tracks

Author(s)

Pan Du

See Also

`help`

Examples

```
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene) && require(Gviz)) {
annotationTracks <- buildAnnotationTracks('1826', includeGeneBody = FALSE, genomicFeature = "TxDb.Hsapiens.UCSC.
}
```

---

checkChrName *check chromosome names*

Description

Check chromosome names and make sure chromosome names start with "chr" (or not if addChr is FALSE)

Usage

```
checkChrName(grange, addChr = TRUE)
```

Arguments

grange	a GRanges, RangedData object, character or named vector
addChr	Whether to add "chr" in front of chromosome names

Details

Because some annotation database names the chromosomes without "chr" prefix, while many others do, it causes problems when both types of data exist in the analysis. This function aims to resolve such issues by checking chromosome names and make sure chromosome names start with "chr" (or not if addChr is FALSE).

Value

return the same type of object with chromosome names checked.

Author(s)

Pan Du

Examples

```
data(exampleMethyGenoSet)
seqlevels(locData(exampleMethyGenoSet))

tt <- checkChrName(exampleMethyGenoSet, addChr = TRUE)
seqlevels(locData(tt))
```

createTranscriptTrack *Create a transcript annotation track*

Description

Create a transcript track, which is a GeneRegionTrack object

Usage

```
createTranscriptTrack(gene, genomicFeature = "TxDb.Hsapiens.UCSC.hg19.knownGene", lib = "org.Hs.eg.db
genome = "hg19", extendRange = c(2000, 2000), includeOtherGene=FALSE, includeGeneBody = TRUE,
thinBox_utrOnly = FALSE, background.title = "gray", fill = "#8282d2", ...)
```

Arguments

gene	An Entrez gene ID or a GRanges object with length equals one
genomicFeature	a TxDb library, TxDb object, or Mart object
lib	Entrez annotation library
genome	The version of genome
extendRange	extended range on each side of the gene
includeOtherGene	whether to include other genes in the same chromosome ranges, only useful when "gene" is a gene ID.
includeGeneBody	whether to include the whole gene body or not
thinBox_utrOnly	whether to only show UTRs as thin boxs in the plot
background.title	the background color of the title
fill	fill color for transcript track
...	other parameters

Details

This function is to create a GeneRegionTrack object for visualization using Gviz package.

Value

a GeneRegionTrack object

Author(s)

Pan Du

See Also

plotTracks, plotTracksWithDataTrackInfo, heatmapByChromosome, plotMethylationHeatmapByGene

Examples

```
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene) && require(Gviz)) {
rangeTrack <- createTranscriptTrack('7157', genomicFeature = "TxDb.Hsapiens.UCSC.hg19.knownGene")
# plotTracks(rangeTrack)
}
```

---

detectDMR.slideWin *Detect DMR (Differentially Methylated Region) using slide window smoothing*

Description

Detect DMR (Differentially Methylated Region) using slide window smoothing

Usage

```
detectDMR.slideWin(methyGenoSet, sampleType, winSize = 250, testMethod = c("ttest", "wilcox"),
  p.adjust.method = "fdr", p.value.detection.th = 0.05, ...)
```

Arguments

methyGenoSet	A GenoSet object includes the methylation data information
sampleType	A vector shows the sample type information
winSize	Slide window size (half window size, bp at each side of the probe)
testMethod	test methods
p.adjust.method	p.value FDR adjust method
p.value.detection.th	the threshold of detection p.value used to determine the failed probes, which will be set as NAs.
...	other paramters

Details

The function will check whether the data was previously smoothed. If not, slide-window smoothing will be performed first, and then followed by differential methylation tests

Value

A GRanges object with additional test information (difference, p.value, p.adjust, and etc.)

Author(s)

Pan Du

See Also

identifySigDMR

Examples

```
data(exampleMethyGenoSet)

sampleType <- pData(exampleMethyGenoSet)$SampleType

## Do differential test
allResult <- detectDMR.slideWin(exampleMethyGenoSet, sampleType=sampleType, testMethod='ttest')
head(allResult)
```

---

estimateCMR.methylation

Estimate the averaged methylation levels within a chromosome region or transcript promoter

---

Description

Estimate the averaged methylation levels within a chromosome region defined as a GRanges object or transcript promoter

Usage

```
estimateCMR.methylation(cmr, methyGenoSet, tx2probe.corList = NULL, estimateFun = mean, probeAnnotati
```

Arguments

cmr	A GRanges object or transcript ID
methyGenoSet	A MethyGenoSet object keeps the DNA methylation data
estimateFun	The function used to estimate the methylation levels within the chromosome region
probeAnnotation	Pre-calculated probe annotation (a GRanges object)
selectGeneElement	Gene elements used to calculate the transcript promoter methylation levels if cmr GRanges object is not provided.
mc.cores	Number of cores used to calculate in parallel

Value

A numeric matrix (row: cmr, column: samples)

Author(s)

Pan DU

`estimateMethySeq` *Estimate the methylation level (Beta-value) of Methyl-Seq data*

Description

Estimate the methylation level (Beta-value) of Methyl-Seq data, which is a VRanges object output by NGS pipeline

Usage

```
estimateMethySeq(seqVariant, coverage, CpGInfo = NULL, mergeStrand = TRUE, cleanVariant = TRUE, minCo
```

Arguments

`seqVariant`	a VRanges object output by NGS pipeline implemented in HTSeqGenie package
`coverage`	the genome coverage (a RleList object) output by NGS pipeline
`CpGInfo`	the precalculated CpG-site information (by identifyCpG function)
`mergeStrand`	whether to merge the AT and GA conversion on opposite strands
`cleanVariant`	whether to filter those non-CpG with full CT and GA conversion, or non CT and GA variations
`minCoverage`	minimum coverage for the variants

Value

a GRanges object with the Beta column shows the methylation levels

Author(s)

Pan Du

exampleMethyGenoSet *Example MethyGenoSet dataset*

Description

Example MethyGenoSet dataset, which includes eight randomly picked cancer cell line samples from two tissue types. To save space, only 21 chromosome data was included.

Usage

```
data(exampleMethyGenoSet)
```

Details

Example MethyGenoSet dataset, which includes eight randomly picked cancer cell line samples from two tissue types. To save space, only 21 chromosome data was included.

Examples

```
data(exampleMethyGenoSet)
class(exampleMethyGenoSet)
colnames(exampleMethyGenoSet)
exampleMethyGenoSet
```

export.DMRInfo *Output the DMR (Differentially Methylated Region) data information*

Description

Output the DMR (Differentially Methylated Region) data information

Usage

```
export.DMRInfo(DMRInfo.ann, methyData = NULL, savePrefix = "")
```

Arguments

DMRInfo.ann	The annotated DMR information outputted by annotateDMRInfo.
methyData	Methylation data information in MethyGenoSet or MethyLumiM class
savePrefix	The prefix added to the output file names.

Details

This function basically save the annotated DMR information as text .csv files.

Value

results files.

Author(s)

Pan Du

See Also

annotateDMRInfo

Examples

```
data(exampleMethyGenoSet)

sampleType <- pData(exampleMethyGenoSet)$SampleType

## Do differential test
allResult <- detectDMR.slideWin(exampleMethyGenoSet, sampleType=sampleType, testMethod='ttest')

## Identify the DMR (Differentially Methylated Region) by setting proper parameters.
## Here we just use default ones
allDMRInfo <- identifySigDMR(allResult)

## Annotate significant DMR info
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene)) {
DMRInfo.ann <- annotateDMRInfo(allDMRInfo, 'TxDb.Hsapiens.UCSC.hg19.knownGene')
export.DMRInfo(DMRInfo.ann, savePrefix='testExample')
}
```

---

export.methyGenoSet *Export a MethyGenoSet object to be visualized using external genome browser tools*

Description

Export a MethyGenoSet object to be visualized in IGV, IGB or other tools. Current version supports "gct" or "bw" formats.

Usage

```
export.methyGenoSet(methyGenoSet, file.format = c("gct", "bw"), exportValue = c("beta", "M", "intensi
```

Arguments

methyGenoSet	A MethyGenoSet object.
file.format	Export file format
exportValue	Export methylation values
hgVersion.default	The default human genome version
savePrefix	The prefix used in the output filename. Only valid when outputFile is NULL.
outputFile	The output file name provided by the user. If file.format is "bw", outputFile should be a character vector with the same length as the sample number, or else it will be ignored.

Details

An easy way to visualize DNA methylation data is to export the DNA methylation data in certain formats, and visualize these files using some external genome browser tools, like IGV (http://www.broadinstitute.org/igv/) and IGB (http://bioviz.org/igb/index.html). The current implementation of this function supports two output formats: ".gct" and ".bw" files. ".gct" includes all samples in a single file. It is only supported by IGV genome browser. The BigWig format (".bw") is a more general format supported by many visualization tools. Each BigWig file represents one single sample. So it is more flexible for the users only interested in a subset of samples.

Value

Output "gct" (for IGV) or "bw" (BigWig) files

Author(s)

Pan Du

References

IGV: http://www.broadinstitute.org/igv/ IGB: http://bioviz.org/igb/index.html

Examples

```
data(exampleMethyGenoSet)
export.methyGenoSet(exampleMethyGenoSet, file.format='gct', savePrefix='test')
# export.methyGenoSet(exampleMethyGenoSet, file.format='bw', savePrefix='test')
```

`filterBisulfiteVariant` *filtering the variant calls of Bisulfite converted sequencing data*

Description

a VRanges object output by NGS pipeline implemented in HTSeqGenie package

Usage

```
filterBisulfiteVariant(seqVariant, coverage, CpGInfo, cleanVariant = TRUE, minCoverage = 1, convertTh
```

Arguments

`seqVariant`	a VRanges object output by NGS pipeline implemented in HTSeqGenie package
`coverage`	the genome coverage (a RleList object) output by NGS pipeline
`CpGInfo`	the pre-calculated CpG-site information (by identifyCpG function)
`cleanVariant`	whether to filter those variants on non-CpG sites with full CT and GA conversion, or non CT and GA variations
`minCoverage`	minimum coverage for the variants
`convertTh.nonCpG`	convert rate threshold used by filtering variants on non-CpG sites (cleanVariant is TRUE)

Details

Filtering the variant calls based on following criteria: Only keeps CG and GA conversion variants. The variants should have minimum coverage. When cleanVariant parameter is TRUE, those fully converted non-CpG sites (convert rate higher than convertTh.nonCpG) will be removed.

Value

a filtered VRanges object with two attributes: 'variantStats' and 'filterSettings'

Author(s)

Pan Du

See Also

`estimateMethySeq`

`getContinuousRegion` *Get continuous chromosome region by merging nearby or overlapping regions*

Description

Get continuous chromosome region by merging nearby or overlapping regions

Usage

```
getContinuousRegion(detectResult, scoreColumns = NULL, scoreFuns = c(mean=mean), maxGap = 2000, minGa
```

Arguments

`detectResult`	A GRanges object (with "status" column) or a data.frame with "CHROMOSOME", "POSITION" and "status" columns
`scoreColumns`	The numeric score columns to be summarized in DMR
`scoreFuns`	A named vector of summarizing functions. The vector names will be used in the output columns
`maxGap`	The maximum gap allowed between two nearby probes to be considered within a same DMR
`minGap`	If two nearby DMRs have a gap less than or equal to the minGap, they will be merged as a single DMR

Details

The "status" column in the "detectRsult" parameter is required, which is a logical vector indicating the interested probes.

Value

A GRanges object of DMR

Author(s)

Pan Du

See Also

`identifySigDMR`

Examples

```
data(exampleMethyGenoSet)
## get sample type information
sampleType <- pData(exampleMethyGenoSet)$SampleType

## Do differential test
allResult <- detectDMR.slideWin(exampleMethyGenoSet, sampleType=sampleType, testMethod='ttest')

## Identify the DMR (Differentially Methylated Region) by setting proper parameters.
## Here we simply using fdr.adjusted p.value cutoff 0.05 to define DMR
## "status" column is required for getContinuousRegion function.
values(allResult)$status <- values(allResult)$p.adjust < 0.05
dmrInfo <- getContinuousRegion(allResult)
```

`getCoverage` *get the coverage based on a given GRanges object*

Description

calculate the average coverage on each element of a given GRanges object

Usage

```
getCoverage(grange, coverage, startOnly = FALSE, as.GRanges = FALSE)
```

Arguments

`grange`	GRanges object specify the genome locations to get coverage
`coverage`	the genome coverage (a RleList object) output by NGS pipeline
`startOnly`	whether to calculate the coverage based on the start location or the average of the entire GRanges element
`as.GRanges`	whether return a GRanges object or a vector of coverage

Value

a vector of coverage or a GRanges object (as.GRanges is TRUE)

Author(s)

Pan Du

heatmapByChromosome *heatmap with chromosome location as x axis*

Description

heatmap with chromosome location as x axis and plot together with other gene annotation information

Usage

```
heatmapByChromosome(genoSet, gene, annotationTracks = NULL, otherTrackList = NULL, phenoData = NULL,
phonoColorMap = NULL, extendRange = c(2000, 2000), includeGeneBody = TRUE, showFullModel = FALSE, sort
cytobandInfo = NULL, CpGInfo = NULL, genomeAxis = TRUE, dataTrackName = "Methylation Profile", lib = "
genome = "hg19", genomicFeature = "TxDb.Hsapiens.UCSC.hg19.knownGene", gradient = c("blue", "white",
ncolor = 16, ylim = NULL, th = 0.99, main = "", selSample = NULL, ...)
```

Arguments

genoSet	a GenoSet object keeping the methylation data as the "exprs" numeric matrix in the AssayData
gene	a Entrez Gene ID, or a GRanges object to define the chromosome range of the plot.
annotationTracks	A annotation tracks list returned by buildAnnotationTracks
otherTrackList	A list of other tracks supported by plotTracks function
phenoData	a data matrix with the same number of rows or columns as the columns of genoSet.
phonoColorMap	the colormap for expression heatmap
extendRange	extended range on each side of the gene
includeGeneBody	whether to include genebody of the provided gene
showFullModel	whether to show full gene model track when includeGeneBody = FALSE
sortSample	whether to sort samples or not
cytobandInfo	cytoband information
CpGInfo	CpG-island information, GRanges or bed file are supported
genomeAxis	whether to add genome axis or not
dataTrackName	the title of the data track
lib	the Entrez annotation library
genome	genome name
genomicFeature	genomic features: "TxDb" library or object, "Mart" object
gradient	the gradient color used by data track heatmap
ncolor	the number of color levels

ylim	the range for plotting the data.
th	the quantile threshold used to remove outlier, which affects the plot color ranges.
main	the title of the plot
selSample	subset of samples for plotting. It is designed for BigMatrix, which have to extract the data at the last moment.
...	other parameters used by plotTracksWithDataTrackInfo

Details

This function plots heatmap with chromosome location as x axis and together with other gene annotation information. It is adapted based on the plotTracks function in Gviz package. Users can also provide a GRanges object to specify a plot region.

Value

returns the grid viewport layout information

Author(s)

Pan Du

See Also

plotTracks, plotTracksWithDataTrackInfo, plotMethylationHeatmapByGene

Examples

```
data(exampleMethyGenoSet)
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene) && require(Gviz)) {
## define data track
exampleMethyGenoSet <- checkChrName(exampleMethyGenoSet)

## build annotation tracks
selGene <- '1826'
annotationTracks <- buildAnnotationTracks(selGene, includeGeneBody = FALSE, genomicFeature = "TxDb.Hsapiens.UCSC
heatmapByChromosome(exampleMethyGenoSet, selGene, annotationTracks = annotationTracks)
}
```

---

identifyCpG *Identify the CpG-site locations from a genome library*

Description

Identify the CpG-site locations from a genome library

Usage

```
identifyCpG(bsgenome = "Hsapiens", seqnames, genomeLib = "BSgenome.Hsapiens.UCSC.hg19", pattern = "CG
```

Arguments

bsgenome	a BSgenome object or variant name in the genomeLib
seqnames	chromosome names, if missing all chromosomes will be used.
genomeLib	the BSgenome library in Bioconductor
pattern	the sequence pattern to be matched.

Value

a GRanges object with CpG-site locations

Author(s)

Pan Du

See Also

filterBisulfiteVariant

Examples

```
# library(GenomicFeatures)
# library(BSgenome)
# seqnames <- paste('chr', c(1:22, 'X', 'Y', 'M'), sep='')
```

---

identifySigDMR *Identify significantly DMR (Differentially Methylated Region)*

Description

Identify significantly DMR (Differentially Methylated Region)

Usage

```
identifySigDMR(detectResult, p.adjust.method = "fdr", pValueTh = 0.01, fdrTh = pValueTh, diffTh = 1, m
```

Arguments

detectResult	A GRanges object or a data.frame with "PROBEID", "CHROMOSOME" and "POSITION" columns
p.adjust.method	p.value FDR adjust method
pValueTh	The threshold of p.value
fdrTh	The threshold of FDR (adjusted p.value)
diffTh	The threshold of difference between two conditions
minProbeNum	Minimum number of probes in each DMR
maxGap	The maximum gap allowed between two nearby probes to be considered within a same DMR
minGap	If two nearby DMRs have a gap less than or equal to the minGap, they will be merged as a single DMR
oppositeMethylation	Whether require the averaged methylation levels in the DMR are in opposite direction
topNum	Whether only returns the top number of probes (ranked by p.value)

Details

(with "status" column)

We define a differentially methylated region (DMR) as a region, in which most measured CpG-sites are differentially methylated. To identify DMRs, the function first determines the differential methylation status of each probe, then merge them as a continuous region. The identifySigDMR function returns a list of two GRanges objects. The sigDMRInfo includes the identified DMRs, and the sigDataInfo includes all differentially methylated probe information.

Value

A list of GRanges objects, sigDMRInfo and sigDataInfo. sigDMRInfo contains DMR information, while sigDataInfo includes the probe level information within the DMRs.

Author(s)

Pan Du

See Also

getContinuousRegion, annotateDMRInfo

Examples

```
data(exampleMethyGenoSet)

sampleType <- pData(exampleMethyGenoSet)$SampleType

## Do differential test
```

```
allResult <- detectDMR.slideWin(exampleMethyGenoSet, sampleType=sampleType, testMethod='ttest')

## Identify the DMR (Differentially Methylated Region) by setting proper parameters.
## Here we just use default ones
allDMRInfo <- identifySigDMR(allResult)
```

MethyGenoSet-class *Class MethyGenoSet: contain and describe Illumina Infinium methylation data in GenoSet-class*

Description

This is a class representation for Illumina Infinium methylation microarray data. It directly extends GenoSet. The purpose of this class is to make the high-density methylation microarray data MethyLumiM-class compatible with the Biocoductor infrastructure packages designed for sequencing analysis.

Extends

Directly extends class GenoSet.

Creating Objects

MethyGenoSet(locData, exprs, methylated, unmethylated, detection = NULL, pData = NULL, annotation = "", universe = NULL, assayData=NULL, ...)

MethyGenoSet instances are usually created through converting from MethyLumiM object using MethyLumiM2GenoSet function or calling MethyGenoSet function as shown above. The arguments, locData, exprs, methylated and unmethylated, are required; others can be missing. Please check GenoSet for more details of other parameters.

Slots

locData: a GRanges or RangedData object, inherited from GenoSet

assayData: contains equal dimensional matrices: exprs (contains the methylation M-value, same as MethyLumiM-class), methylated (contains the methylated probe intensities. Same as MethyLumiM-class), unmethylated (contains the unmethylated probe intensities. Same as MethyLumiM-class), detection (records the detection p-value of the probe. Same as MethyLumiM-class). For more details of assayData, please see ExpressionSet

featureData: See eSet

phenoData: See eSet

experimentData: See eSet

protocolData: See eSet

annotation: See eSet

.__classVersion__: See eSet

history: a data.frame recording the operation history of the MethyGenoSet object.

Methods

Class-specific methods:

`exprs(MethyGenoSet)`, `exprs(MethyGenoSet,matrix)<-`: Access and set elements named `exprs` in the `AssayData-class` slot.

`methylated(MethyGenoSet)`, `methylated(MethyGenoSet)<-`: Access and set elements named `methylated` in the `AssayData-class` slot.

`unmethylated(MethyGenoSet)`, `unjmethylated(MethyGenoSet)<-`: Access and set elements named `unmethylated` in the `AssayData-class` slot.

`detection(MethyGenoSet)`, `detection(MethyGenoSet)<-`: Access and set elements named `detection` in the `AssayData-class` slot.

`as(methyGenoSet,"MethyLumiM")` Coerce objects of `MethyGenoSet-class` to `MethyLumiM`

`as(genoSet,"MethyGenoSet")` Coerce objects of `GenoSet-class` to `MethyGenoSet`

`getHistory(MethyGenoSet)`: Access the operation history of `MethyGenoSet` object.

Derived from `GenoSet`:

`locData(MethyGenoSet)`: return a RangedData object, which contains the chromosome location information

Derived from `ExpressionSet` (For the directly inherited methods, please see `ExpressionSet` and `eSet`):

`combine(MethyGenoSet,missing)`: Combine two MethyGenoSet objects, including `history` slot. See `eSet`

`exprs(MethyGenoSet)`, `exprs(MethyGenoSet,matrix)<-`: Access and set elements named `exprs` in the `AssayData-class` slot.

`object[(i,j)`: Conduct subsetting of the data in a MethyGenoSet object

Standard generic methods Please check `ExpressionSet` and `eSet` for other inherited methods,

Author(s)

Pan Du

See Also

`MethyLumiM2GenoSet`)

Examples

```
## load example data
data(exampleMethyGenoSet)
class(exampleMethyGenoSet)
```

MethyLumiM2GenoSet *Coerce objects of MethyLumiM-class to MethyGenoSet*

Description

Coerce objects of MethyLumiM-class to MethyGenoSet

Usage

```
MethyLumiM2GenoSet(methyLumiM, lib = "FDb.InfiniumMethylation.hg19", bigMatrix=FALSE, dir.bigMatrix=
```

Arguments

methyLumiM	a MethyLumiM object
lib	lib is a annotation library
bigMatrix	whether to save the data as BigMatrix (designed for very large dataset)
dir.bigMatrix	the parent directory to save the BigMatrix data files
savePrefix.bigMatrix	the folder name prefix of the directory to save the BigMatrix data files. The fold name will be like this: paste(savePrefix.bigMatrix, '_bigmat', sep="")

Value

a MethyGenoSet object

Author(s)

Pan Du

See Also

MethyGenoSet

Examples

```
if (require(FDb.InfiniumMethylation.hg19)) {
data(exampleMethyGenoSet)
## set as MethyLumiM object
methyLumiM <- as(exampleMethyGenoSet, 'MethyLumiM')
## set back as MethyGenoSet object
methyGenoSet <- MethyLumiM2GenoSet(methyLumiM, lib = "FDb.InfiniumMethylation.hg19")
class(methyGenoSet)
}
```

plotHeatmapByGene *plot methylation heatmap by genes*

Description

plot methylation heatmap by genes

Usage

```
plotHeatmapByGene(selGene, genoSet,  phenoData = NULL, sortBy=c(NA, 'phenoData', 'data'), includeGene
sortByTx = FALSE, CpGInfo = NULL, genomicFeature = NULL, phenoColor = list(gradient=c("green", "black'
title.suffix = NULL, addLegend = TRUE, genoSetLegendTitle = NULL, gradient = c("blue", "white", "red")
ncolor = 16, main = NULL, newPlot = TRUE, ylim = NULL, ...)
```

Arguments

selGene	a Entrez Gene ID
genoSet	a GenoSet object or a list of GenoSet objects
phenoData	a data.frame for phenotype information
sortBy	whether to sort samples based on the phenoData, cluster of genoSet data or NA (no sorting)
includeGeneBody	if FALSE, then only shows the promoter region
sortByTx	if TRUE, sort the genoset columns based on Gene Model track. (only valid when the genoset column names are matching transcript IDs.)
CpGInfo	a bed file or GRanges for CpG island information
genomicFeature	used by buildAnnotationTracks function
phenoColor	a list of colors corresponding to phenotype
title.suffix	a string attached to the end of the title
addLegend	whether to add a legend or not
genoSetLegendTitle	title for methylation colorbar legend
gradient	the gradient color to show the DataTrack
ncolor	the number of color levels
main	title of the plot. If it is null, then the Gene Symbol will be the plot title
newPlot	whether to create a new plot or add it to previous plot
ylim	ylim for the genoSet data, which is also used for plotting the legend.
...	other parameters used by heatmapByChromosome

Details

Function, plotHeatmapByGene, is specifically designed for the methylation data. It plots one gene or genomic range each time. Users can add phenotypes or matched gene expression data to the right panel of the plot. Figure legends can be also added. By default, the plotHeatmapByGene plots methylation Beta-values (in the range of 0 to 1) instead of M-values. Users can set useBetaValue as FALSE if they want to change to M-values.

Value

returns the grid viewport information

Author(s)

Pan Du

See Also

See also `heatmapByChromosome`

Examples

```
data('exampleMethyGenoSet')
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene)) {
  genomicFeature <- 'TxDb.Hsapiens.UCSC.hg19.knownGene'
  selGene <- '1826'
  plotHeatmapByGene(selGene, genoSet=exampleMethyGenoSet, phenoData=pData(exampleMethyGenoSet), genomicFeature=
}
```

---

plotMethylationHeatmapByGene

plot methylation heatmap by genes

Description

plot methylation heatmap by genes

Usage

```
plotMethylationHeatmapByGene(selGene, methyGenoSet, gene2tx = NULL, expression.tx = NULL, expression.
phenoData = NULL, sortBy=c('expression', 'methylation', NA), scaledExpression = FALSE, labelPrefix.ex
showAllTx = TRUE, useBetaValue = TRUE, includeGeneBody = FALSE, CpGInfo = NULL, genomicFeature = NULL,
phenoColor = list(gradient=c("green", "black", "red")), th = 0.99, title.suffix = NULL, addLegend = TR
methylationLegendTitle = NULL, expressionLegendTitle = "Expression\n(log2-RPKM)",
gradient = c("blue", "white", "red"), ncolor = 16, main = NULL, newPlot = TRUE, selSample = NULL, ...)
```

Arguments

selGene	a vector of EntrezIDs or a list of gene2tx
methyGenoSet	a GenoSet object for methylation data
gene2tx	a gene to transcript mapping list, used for retrieving expression.tx data
expression.tx	an ExpressionSet or data matrix for transcript expression
expression.other	an ExpressionSet or data matrix for other types of expression, whose dimnames matches expression.tx
phenoData	a data.frame for phenoData information
sortBy	whether to sort samples based on the mean of expression profiles, methylation cluster or NA (no sorting)
scaledExpression	whether to scale the expression values based on maximum expression (to the range of 0 to 1)
labelPrefix.expression.other	the labelPrefix for the "expression.other" colnames
showAllTx	whether to show all transcript in gene2tx or just those provided in selGene
useBetaValue	whether to use methylation Beta-value in the plot.
includeGeneBody	if FALSE, then only shows the promoter region
CpGInfo	a bed file or GRanges for CpG island information
genomicFeature	used by buildAnnotationTracks function
phenoColor	a list of colors corresponding to phenoData
th	the quantile threshold used to remove outlier, which affects the plot color ranges.
title.suffix	a string attached to the end of the title
addLegend	whether to add a legend or not
methylationLegendTitle	title for methylation colorbar legend
expressionLegendTitle	title for expression colorbar legend
gradient	the gradient color to show the DataTrack
ncolor	the number of color levels
main	title of the plot. If it is null, then the Gene Symbol will be the plot title
newPlot	whether to create a new plot or add it to previous plot
selSample	subset of samples for plotting. It is designed for BigMatrix, which have to extract the data at the last moment.
...	other parameters used by heatmapByChromosome

Details

Function, plotMethylationHeatmapByGene, is specifically designed for the methylation data. It plots one gene or genomic range each time. Users can add phenotypes or matched gene expression data to the right panel of the plot. Figure legends can be also added. By default, the plotMethylationHeatmapByGene plots methylation Beta-values (in the range of 0 to 1) instead of M-values. Users can set useBetaValue as FALSE if they want to change to M-values.

Value

returns the grid viewport information

Author(s)

Pan Du

See Also

See also heatmapByChromosome

Examples

```
data('exampleMethyGenoSet')
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene)) {
  genomicFeature <- 'TxDb.Hsapiens.UCSC.hg19.knownGene'
  selGene <- '1826'
  plotMethylationHeatmapByGene(selGene, methyGenoSet=exampleMethyGenoSet, phenoData=pData(exampleMethyGenoSet),

  ## use different color map for expression data
  es.example <- matrix(runif(ncol(exampleMethyGenoSet), max=10), nrow=1)
  rownames(es.example) <- selGene
  colnames(es.example) <- colnames(exampleMethyGenoSet)
  plotMethylationHeatmapByGene(selGene, methyGenoSet=exampleMethyGenoSet, expression.tx=es.example, genomicFeat
}
```

---

plotTracksWithDataTrackInfo

plot Tracks with additional DataTrack information added to the left of the plot

---

Description

plot Tracks with additional DataTrack information added to the left of the plot

Usage

```
plotTracksWithDataTrackInfo(trackList, labels = NULL, grange2show = NULL, dataTrackName = NULL, dataI
dataInfoRange = NULL, dataBackground = gray(0.9), minHeatmapColumnWidth = 2, labelWidth = 0.1,
gradient = c("blue", "white", "red"), ncolor = 16, main = "", newPlot = FALSE, sizes = NULL, ...)
```

Arguments

trackList	a list of tracks supported by plotTracks function
labels	the sample labels. By default, it will use the rownames of dataTrack. It can also be a list if there are multiple dataTracks. And the list names should be consistent with dataTrack names. Providing a subset of dataTrack labels is allowed.
grange2show	a GRanges to indicate the plot range
dataTrackName	the name of the DataTrack
dataInfo	a data matrix or data.frame to show the related sample information, e.g. its expression profile
dataColorMap	the color map to plot the dataInfo
dataInfoRange	the range of dataInfo to control the range of color map
dataBackground	the background color for the data tracks
minHeatmapColumnWidth	the minimum width (points) of the heatmap data column
labelWidth	the width of the label, which is the ratio of the entire plot width
gradient	the gradient color to show the DataTrack
ncolor	the number of color levels
main	the title of the plot
newPlot	whether to create a new plot or add it to previous plot
sizes	the track sizes used by plotTracks function
...	other parameters used by plotTracks

Details

This function is adapted based on the plotTracks function in Gviz package. It adds sample labels to the heatmap dataTracks.

Value

Grid viewport layout information

Author(s)

Pan Du

See Also

See Also plotTracks, heatmapByChromosome

Examples

```
data(exampleMethyGenoSet)
if (require(TxDb.Hsapiens.UCSC.hg19.knownGene) && require(Gviz)) {
## define data track
exampleMethyGenoSet <- checkChrName(exampleMethyGenoSet)
dTrack <- DataTrack(range=suppressWarnings(as(locData(exampleMethyGenoSet), 'GRanges')), data=t(exprs(exampleMe
chromosome='chr21', type='heatmap')

## build annotation tracks
annotationTracks <- buildAnnotationTracks('1826', includeGeneBody = FALSE, genomicFeature = "TxDb.Hsapiens.UCSC.
trackList <- c(annotationTracks, list(dTrack))
plotTracksWithDataTrackInfo(trackList, labels=colnames(exampleMethyGenoSet), grange2show = attr(annotationTrack
}
```

`smoothMethyData` *Smooth the methylation data*

Description

Smooth the methylation data by a sliding window with fixed width in bp unit

Usage

```
smoothMethyData(methyData, winSize = 250, lib = "FDb.InfiniumMethylation.hg19", p.value.detection.th
    bigMatrix=FALSE, dir.bigMatrix='.', savePrefix.bigMatrix=NULL, ...)
```

Arguments

`methyData`	A GenoSet or MethyLumiM object
`winSize`	Half sliding window size in bp unit at each side of the probe
`lib`	Methylation annotation library
`p.value.detection.th`	the threshold of detection p.value used to determine the failed probes, which will be set as NAs.
`bigMatrix`	whether to save the data as BigMatrix (designed for very large dataset)
`dir.bigMatrix`	the parent directory to save the BigMatrix data files
`savePrefix.bigMatrix`	the folder name prefix of the directory to save the BigMatrix data files. The fold name will be like this: paste(savePrefix.bigMatrix, '_bigmat', sep='')
`...`	other parameters

Details

The function basically averages the probes within a local window (within winSize bp at each side of the probe).

Value

An object with the methylation values smoothed

Author(s)

Pan Du

See Also

detectDMR.slideWin

Examples

```
data(exampleMethyGenoSet)
smoothData <- smoothMethyData(exampleMethyGenoSet)
```

Index