Package 'BiSeq'

May 16, 2024

Type Package

Title Processing and analyzing bisulfite sequencing data

Version 1.44.0

Date 2022-01-14

Author Katja Hebestreit, Hans-Ulrich Klein

Maintainer Katja Hebestreit <katja.hebestreit@gmail.com>

Depends R (>= 2.15.2), methods, S4Vectors, IRanges (>= 1.17.24), GenomicRanges, SummarizedExperiment (>= 0.2.0), Formula

Imports methods, BiocGenerics, Biobase, S4Vectors, IRanges, GenomeInfoDb, GenomicRanges, SummarizedExperiment, rtracklayer, parallel, betareg, lokern, Formula, globaltest

Description The BiSeq package provides useful classes and functions to handle and analyze targeted bisulfite sequencing (BS) data such as reduced-representation bisulfite sequencing (RRBS) data. In particular, it implements an algorithm to detect differentially methylated regions (DMRs). The package takes already aligned BS data from one or multiple samples.

License LGPL-3

biocViews Genetics, Sequencing, MethylSeq, DNAMethylation

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

git_branch RELEASE_3_19

git_last_commit 0babe62

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-16

Contents

annotateGRanges 2
betaRegression 4

betaResults . . . 5
betaResultsNull . . . 6
binomLikelihoodSmooth . . . 7
BSraw-class . . . 8
BSrel-class . . . 9
clusterSites . . . 11
clusterSitesToGR . . . 12
compareTwoSamples . . . 13
covBoxplots . . . 14
covStatistics . . . 15
DMRs . . . 15
estLocCor . . . 16
filterByCov . . . 17
filterBySharedRegions . . . 18
findDMRs . . . 19
globalTest . . . 20
limitCov . . . 22
logisticRegression . . . 23
makeVariogram . . . 24
plotBindingSites . . . 25
plotMeth . . . 26
plotMethMap . . . 27
plotSmoothMeth . . . 28
predictedMeth . . . 29
predictMeth . . . 30
promoters . . . 31
rawToRel . . . 32
readBismark . . . 32
rrbs . . . 34
smoothVariogram . . . 34
summarizeRegions . . . 36
testClusters . . . 37
trimClusters . . . 38
vario . . . 40
writeBED . . . 40

Index **42**

`annotateGRanges` *Annotates a* `GRanges` *object by means of a second* `GRanges` *object*

Description

Each genomic location of `object` is checked for overlapping with genomic ranges of `regions`. In case of an overlapping, this genomic location is marked as `TRUE`, or with the identifier of respective the `regions` object (if any).

Usage

```
annotateGRanges(object, regions, name, regionInfo)
```

Arguments

object	A GRanges.
regions	A GRanges.
name	A string specifying the name of the metadata column with the overlapping information to be added to object. Usually the type of the regions object, e.g. 'Promoter'.
regionInfo	OPTIONAL: A string or integer specifying the metadata column column of regions object containing the identifiers, e.g. entrez gene IDs of the promoters. If not specified, the genomic locations of object are labeled as TRUE (overlap) or FALSE (no overlap).

Details

If multiple ranges of regions overlap with a genomic region in object, the identifier names of the overlapping regions are seperated by ','.

Value

A GRanges object similar to object containing an additional metadata column with the overlapping information.

Author(s)

Katja Hebestreit

See Also

GRanges-class

Examples

```
# load detected DMRs:
data(DMRs)

# annotate the DMRs with a GRanges object:
data(promoters)
DMRs.anno <- annotateGRanges(object = DMRs,
                             regions = promoters,
                             name = 'Promoter',
                             regionInfo = 'acc_no')
DMRs.anno
```

betaRegression *A function to estimate and test a group factor within a beta regression*

Description

This function models the methylation level within a beta regression. The first independent variable in formula is tested to be unequal to zero.

Usage

```
betaRegression(formula, link, object, mc.cores, ...)
```

Arguments

formula	Symbolic description of the model. For the first independent variable the P value (Wald test) and the effect on methylation is returned. For details see below.
link	A character specifying the link function in the mean model (mu). Currently, "logit", "probit", "cloglog", "log", "loglog" are supported.
object	A BSrel object.
mc.cores	Passed to mclapply.
...	Other parameters passed to the betareg function.

Details

See betareg function for details.

mclapply

Value

A data.frame containing the position, chromosome, P value, estimated methylation level in group 1 and group 2 and methylation difference of group 1 and group 2.

Author(s)

Katja Hebestreit

References

Hebestreit, K., Dugas, M., and Klein HU. Detection of significantly differentially methylated regions in targeted bisulfite sequencing Data. In preparation. Bioinformatics. 2013 Jul 1;29(13):1647-53.

See also reference list in the documentation of betareg.

See Also

betareg

Examples

```
# load RRBS data, subset to save time, find CpG clusters and smooth methylation data:
data(rrbs)
rrbs.small <- rrbs[1:1000,]
rrbs.clust.unlim <- clusterSites(object = rrbs.small,
                                 groups = colData(rrbs)$group,
                                 perc.samples = 4/5,
                                 min.sites = 20,
                                 max.dist = 100)

ind.cov <- totalReads(rrbs.clust.unlim) > 0
quant <- quantile(totalReads(rrbs.clust.unlim)[ind.cov], 0.9)
rrbs.clust.lim <- limitCov(rrbs.clust.unlim, maxCov = quant)

# with a small subset to save calculation time:
rrbs.part <- rrbs.clust.lim[1:100,]

predictedMeth <- predictMeth(object=rrbs.part)

betaResults <- betaRegression(formula = ~group, link = "probit",
                              object = predictedMeth, type="BR")
```

---

betaResults *The output of* betaRegression

Description

Please see the package vignette for description.

Usage

```
data(betaResults)
```

Format

A data frame with 4276 observations on the following 10 variables:

`chr` a factor with levels `chr1` `chr2`

`pos` a numeric vector

`p.val` a numeric vector

`meth.group1` a numeric vector

`meth.group2` a numeric vector

`meth.diff` a numeric vector

`estimate` a numeric vector

`std.error` a numeric vector

`pseudo.R.sqrt` a numeric vector

`cluster.id` a character vector

Examples

```
data(betaResults)
head(betaResults)
```

---

betaResultsNull *The output of* betaRegression *for resampled data*

Description

Please see the package vignette for description.

Usage

```
data(betaResultsNull)
```

Format

A data frame with 4276 observations on the following 10 variables.

`chr` a factor with levels `chr1` `chr2`

`pos` a numeric vector

`p.val` a numeric vector

`meth.group1` a numeric vector

`meth.group2` a numeric vector

`meth.diff` a numeric vector

`estimate` a numeric vector

`std.error` a numeric vector

`pseudo.R.sqrt` a numeric vector

`cluster.id` a character vector

Examples

```
data(betaResultsNull)
head(betaResultsNull)
```

`binomLikelihoodSmooth` *Calculates local likelihood estimations for binomial random variables*

Description

For a given set of binomial random variables with 1-dimensional coordinates, this function calculates the local likelihood estimation of the success probability p at a given point. For this purpose, a weighted likelihood estimation with weights obtained by a triangular kernel with given bandwidth is used. This can be used to predict values at points where no variable has been observed and/or to smooth observations using neighboured observations.

Usage

```
binomLikelihoodSmooth(pred.pos, pos, m, n, h)
```

Arguments

`pred.pos`	A vector of positions where p should be estimated.
`pos`	A vector of positions where binomial variables have been observed.
`m`	A vector of length `pos` with the number of successfull experiments.
`n`	A vector of length `pos` with the number of experiments.
`h`	The bandwidth of the kernel.

Details

For a given position x, the weighted likelihood for parameter p

$$L(p; m, n, w) = \prod_{i=1}^{k} B(m_i|n_i, p)^{w_i}$$

is maximized. B denotes the binomial probability function. The weights w_i are calculated using a triangular kernel with bandwidth h:

$$w_i = K(x_i) = (1 - (|x - x_i|)/h)\mathbf{1}_{((|x-x_i|)/h)\leq 1}$$

Value

A vector of length `pred.pos` giving the local likelihood estimation of the success probability p at the given positions.

Author(s)

Hans-Ulrich Klein

See Also

`predictMeth`

Examples

```
n = rpois(100, lambda=10)
E = c(rep(0.4, 30), rep(0.8, 40), rep(0.1, 30))
m = rbinom(100, n, E)
pos = 1:100
p_10 = binomLikelihoodSmooth(pos, pos, m, n, h=10)
p_20 = binomLikelihoodSmooth(pos, pos, m, n, h=20)

## Not run: plot(x=pos, y=m/n)
points(x=pos, y=p_10, col="green")
lines(x=pos, y=p_10, col="green")
points(x=pos, y=p_20, col="red")
lines(x=pos, y=p_20, col="red")
## End(Not run)
```

BSraw-class *Class to contain raw Bisulfite Sequencing (BiSeq) Data*

Description

The BSraw class is derived from RangedSummarizedExperiment and contains a SimpleList of matrices named methReads and totalReads as assays.

Objects from the Class

Objects can be created by calls of the form BSraw(metadata = list(), rowRanges, colData = DataFrame(row.names=colnames(methReads)), methReads, totalReads, ...).

However, one will most likely create a BSraw object when use readBismark to load data.

Slots

metadata: An optional list of arbitrary content describing the overall experiment.

rowRanges: Object of class "GRanges" containing the genome positions of CpG-sites covered by bisulfite sequencing. WARNING: The accessor for this slot is rowRanges, not rowRanges!

colData: Object of class "DataFrame" containing information on variable values of the samples.

assays: Object of class SimpleList of two matrices, named totalReads and methReads. The matrix totalReads contains the number of reads spanning a CpG-site. The rows represent the CpG sites in rowRanges and the columns represent the samples in colData. The matrix methReads contains the number of methylated reads spanning a CpG-site.

Extends

Class "RangedSummarizedExperiment", directly.

Methods

totalReads `signature(x = "BSraw")`: Gets the `totalReads` slot.

totalReads<- `signature(x = "BSraw", value = "matrix")`: Sets the `totalReads` slot.

methReads `signature(x = "BSraw")`: Gets the `methReads` slot.

methReads<- `signature(x = "BSraw", value = "matrix")`: Sets the `methReads` slot.

combine `signature(x = "BSraw", y = "BSraw")`: Combines two `BSraw` objects.

Author(s)

Katja Hebestreit

See Also

RangedSummarizedExperiment, `BSrel-class`, `readBismark`

Examples

```
showClass("BSraw")

## How to create a BSraw object by hand:
metadata <- list(Sequencer = "Sequencer", Year = "2013")
rowRanges <- GRanges(seqnames = "chr1",
                  ranges = IRanges(start = c(1,2,3), end = c(1,2,3)))
colData <- DataFrame(group = c("cancer", "control"),
                     row.names = c("sample_1", "sample_2"))
totalReads <- matrix(c(rep(10L, 3), rep(5L, 3)), ncol = 2)
methReads <- matrix(c(rep(5L, 3), rep(5L, 3)), ncol = 2)
BSraw(metadata = metadata,
      rowRanges = rowRanges,
      colData = colData,
      totalReads = totalReads,
      methReads = methReads)

## A more realistic example can be loaded:
data(rrbs)
rrbs

head(totalReads(rrbs))
head(methReads(rrbs))
```

`BSrel-class` *Class to contain Bisulfite Sequencing (BiSeq) Data*

Description

The `BSrel` class is derived from `RangedSummarizedExperiment` and contains a `SimpleList` of one matrix named `methLevel` as `assays`.

Objects from the Class

Objects can be created by calls of the form `BSrel(metadata = list(), rowRanges, colData = DataFrame(row.names=colnames(methLevel)), methLevel, ...)`.

However, one will most likely create a `BSraw` object when use `readBismark` to load data.

Slots

`metadata`: An optional `list` of arbitrary content describing the overall experiment.

`rowRanges`: Object of class `"GRanges"` containing the genome positions of CpG-sites covered by bisulfite sequencing. WARNING: The accessor for this slot is `rowRanges`, not `rowRanges`!

`colData`: Object of class `"DataFrame"` containing information on variable values of the samples.

`assays`: Object of class `SimpleList` of a matrix, named `methLevel` containing the methylation levels (between 0 and 1) per CpG site. The rows represent the CpG sites in `rowRanges` and the columns represent the samples in `colData`.

Extends

Class `"RangedSummarizedExperiment"`, directly.

Methods

methLevel `signature(x = "BSrel")`: Gets the `methLevel` slot.

methLevel<- `signature(x = "BSrel", value = "matrix")`: Sets the `methLevel` slot.

combine `signature(x = "BSrel", y = "BSrel")`: Combines two `BSrel` objects.

Author(s)

Katja Hebestreit

See Also

RangedSummarizedExperiment, `BSraw-class`, `readBismark`

Examples

```
showClass("BSrel")

## How to create a BSrel object by hand:
metadata <- list(Sequencer = "Sequencer", Year = "2013")
rowRanges <- GRanges(seqnames = "chr1",
                  ranges = IRanges(start = c(1,2,3), end = c(1,2,3)))
colData <- DataFrame(group = c("cancer", "control"),
                     row.names = c("sample_1", "sample_2"))
methLevel <- matrix(c(rep(0.5, 3), rep(1, 3)), ncol = 2)
BSrel(metadata = metadata,
      rowRanges = rowRanges,
      colData = colData,
      methLevel = methLevel)
```

```
# Or get a BSrel object out of a BSraw object:
data(rrbs)
rrbs.rel <- rawToRel(rrbs)
```

---

clusterSites *Assigns CpG cluster memberships on CpG sites within* BSraw *objects*

Description

Within a BSraw object clusterSites searches for agglomerations of CpG sites across all samples. In a first step the data is reduced to CpG sites covered in round(perc.samples*ncol(object)) samples, these are called 'frequently covered CpG sites'. In a second step regions are detected where not less than min.sites frequently covered CpG sites are sufficiantly close to each other (max.dist). Note, that the frequently covered CpG sites are considered to define the boundaries of the CpG clusters only. For the subsequent analysis the methylation data of all CpG sites within these clusters are used.

Usage

```
clusterSites(object, groups, perc.samples, min.sites, max.dist,
mc.cores, ...)
```

Arguments

object	A BSraw.
groups	OPTIONAL. A factor specifying two or more sample groups within the given object. See Details.
perc.samples	A numeric between 0 and 1. Is passed to filterBySharedRegions.
min.sites	A numeric. Clusters should comprise at least min.sites CpG sites which are covered in at least perc.samples of samples, otherwise clusters are dropped.
max.dist	A numeric. CpG sites which are covered in at least perc.samples of samples within a cluster should not be more than max.dist bp apart from their nearest neighbors.
mc.cores	Passed to mclapply Default is 1.
...	Further arguments passed to the filterBySharedRegions function. closer than

Details

There are three parameters that are important: perc.samples, min.sites and max.dist. For example, if perc.samples=0.5, the algorithm detects all CpG sites that are covered in at least 50% of the samples. Those CpG sites are called frequently covered CpG sites. In the next step the algorithm determines the distances between neighboured frequently covered CpG sites. When they are closer than (or close as) max.dist base pairs to each other, those frequently covered CpG sites and all other, less frequently covered CpG sites that are in between, belong to the same cluster.

In the third step, each cluster is checked for the number of frequently covered CpG sites. If this number is less than `min.sites`, the cluster is discarded.

In other words: 1. The `perc.samples` parameter defines which are the frequently covered CpG sites. 2. The frequently covered CpG sites determine the boundaries of the clusters, depending on their distance to each other. 3. Clusters are discarded if they have too less frequently covered CpG sites.

If argument `group` is given, `perc.samples`, or `no.samples`, are applied for all group levels.

Value

A `BSraw` object reduced to CpG sites within CpG cluster regions. A `cluster.id` metadata column on the `rowRanges` assigns cluster memberships per CpG site.

Author(s)

Katja Hebestreit

See Also

`filterBySharedRegions`, `mclapply`

Examples

```
data(rrbs)
rrbs.clust <- clusterSites(object = rrbs, groups = colData(rrbs)$group,
                           perc.samples = 4/5, min.sites = 20,
                           max.dist = 100)
```

---

clusterSitesToGR *A function to obtain a* `GRanges` *object of CpG clusters from* `BSraw` *and* `BSrel` *objects*

Description

This function allows to get the start and end positions of CpG clusters from a `BSraw` or `BSrel` object, when there is a `cluster.id` column in the `rowRanges` slot.

Usage

```
clusterSitesToGR(object)
```

Arguments

`object`	A `BSraw` or `BSrel` object with a `cluster.id` column in the `rowRanges` slot. Usually the output of `clusterSites`.

Value

An object of class `GRanges` is returned.

Author(s)

Katja Hebestreit

See Also

clusterSites

Examples

```
data(rrbs)
rrbs.clustered <- clusterSites(rrbs)

clusterSitesToGR(rrbs.clustered)
```

---

compareTwoSamples *Detects DMRs by comparing two samples*

Description

Determines the differences of (smoothed) methylation levels between two samples and aggregates the sites surpassing a minimum difference to DMRs.

Usage

```
compareTwoSamples(object, sample1, sample2, minDiff, max.dist)
```

Arguments

object	A BSrel.
sample1	A numeric or character specifying the first sample to be used.
sample2	A numeric or character specifying the second sample to be used.
minDiff	A numeric greater than 0 and smaller or equal to 1.
max.dist	Numeric. The maximum distance between two CpG sites (or grid points) with absolute methylation differences greater or equal than minDiff in a DMR. If grid points are used: should be the same as grid.dist in predictMeth.

Details

This function determines the differences between the methylation levels of sample1 and sample2 for each site. Successive sites with methylation differences smaller or equal to minDiff are summarized.

Value

A GRanges object.

Author(s)

Katja Hebestreit

See Also

predictMeth

Examples

```
data(rrbs)
rrbs <- rrbs[, c(1,6)]
CpG.clusters <- clusterSites(object = rrbs, perc.samples = 1,
                             min.sites = 20, max.dist = 100)
predictedMeth <- predictMeth(object = CpG.clusters)
DMRs <- compareTwoSamples(predictedMeth, sample1 = 1, sample2 = 2,
                          minDiff = 0.3, max.dist = 100)
```

---

covBoxplots *Creates boxplots of coverages per sample*

Description

A boxplot per sample is plotted for the coverages of CpG-sites. It is constrained to CpG-sites which are covered in the respective sample (coverage != 0 and not NA).

Usage

```
R
covBoxplots(object, ...)
```

Arguments

object	A BSraw.
...	Other graphical parameters passed to the boxplot function.

Author(s)

Katja Hebestreit

See Also

boxplot

Examples

```
data(rrbs)
covBoxplots(rrbs)
```

covStatistics *Prints a short summary of coverage statistics per sample*

Description

This function produces information per samples about 1.) the covered CpG-sites 2.) the median of their coverages.

Usage

```
covStatistics(object)
```

Arguments

object	A BiSeq object.

Author(s)

Katja Hebestreit

Examples

```
data(rrbs)
covStatistics(rrbs)
```

DMRs *The output of* findDMRs

Description

Please see the package vignette for description.

Usage

```
data(DMRs)
```

Format

A GRanges of the chromosomes, start and end positions of the detected DMRs together with information (in the metadata columns) on DMRs: median.p, codemedian.meth.group1, codemedian.meth.group2, median.meth.diff.

Examples

```
data(DMRs)
head(DMRs)
```

estLocCor *Estimates the correlations of the z-scores*

Description

For each location the correlation of this location's z-score to $\bar{Z}$ of its CpG cluster is estimated.

Usage

```
estLocCor(vario.sm)
```

Arguments

vario.sm	Output of smoothVariogram.

Value

A list:

variogram	A variogram matrix, usually created by smoothVariogram beforehand.
pValsList	A list of the test results per CpG cluster.
sigma.cluster	The standard deviations of z-scores within each cluster.
Z.cluster	The arithmetic means of the z-scores for each cluster.
length.cluster	The widths (number of pase pairs) of each cluster.

Author(s)

Katja Hebestreit

References

Yoav Benjamini and Ruth Heller (2007): False Discovery Rates for Spatial Signals. American Statistical Association, 102 (480): 1272-81.

See Also

makeVariogram, smoothVariogram

Examples

```
data(betaResultsNull)

vario <- makeVariogram(betaResultsNull)
vario.sm <- smoothVariogram(vario, sill = 1)

locCor <- estLocCor(vario.sm)
```

filterByCov *Filters regions (or single CpGs) of a* BSraw *object with a minimum coverage*

Description

This method reduces a BSraw object to its regions (or single CpGs) with a minimum number of reads.

Usage

```
filterByCov(object, minCov, global)
```

Arguments

object	A BSraw.
minCov	Minimum number of reads overlapping the CpG sites.
global	A logical indicating whether the regions should achieve the minimum coverage in each sample. If global = TRUE the filtered object will consist of the regions achieving the minimum coverage in all samples. If global = FALSE (default) this function filters the regions for each sample separately, irrespectively of the coverages in other samples. totalReads and methReads are set to zero, if the minimum coverage is not obtained. Regions covered too sparse in all samples are dropped.

Value

A BSraw object containing the CpGs or regions achieving the minimum coverage in all (if global=TRUE) or at least one (if global=FALSE) samples.

Author(s)

Katja Hebestreit

See Also

filterBySharedRegions

Examples

```
data(rrbs)
rrbs.reduced <- filterByCov(object=rrbs, minCov=10, global=TRUE)
```

`filterBySharedRegions` *Reduces a* `BSraw` *or* `BSrel` *object to regions (or single CpGs) shared by a fraction of samples*

Description

This method determines the regions which are covered in a given fraction of samples and reduces the `BSraw` or `BSrel` object to these regions.

Usage

```
filterBySharedRegions(object, groups, perc.samples, no.samples, minCov)
```

Arguments

`object`	A `BSraw` or `BSrel` object.
`groups`	OPTIONAL. A factor specifying two or more groups within the given object. See Details.
`perc.samples`	A numeric vector with elements between 0 and 1 of length 1 or of the same length as levels in group. Default is 1.
`no.samples`	Alternative to `perc.samples`. An integer vector of length 1 or of the same length as levels in group.
`minCov`	A numeric: If `object` is a `BSraw` object the minimum coverage may be set. Default is 1.

Details

If argument `group` is given `perc.samples` or `no.samples` are applied for all group levels.

Value

An object of the same class as `object` storing methylation information solely for regions or single CpGs covered in at least `round(perc.samples*ncol(object))` samples, if `perc.samples` is given. Alternatively, the number of samples can be given directly by using `no.samples`.

Author(s)

Katja Hebestreit

See Also

`filterByCov`

Examples

```
data(rrbs)
rrbs.reduced <- filterBySharedRegions(object = rrbs, perc.samples = 1)

# Or filter CpG sites where at least 50% of cancer and 50% of normal samples have coverage:
rrbs.reduced <- filterBySharedRegions(object=rrbs, groups = colData(rrbs)$group,
                                      perc.samples = 0.5)
```

`findDMRs` *Aggregates CpG sites to DMRs*

Description

This function aggregates CpG sites to DMRs on the basis of their P values.

Usage

```
findDMRs(test.out, alpha, max.dist, diff.dir)
```

Arguments

`test.out`	An object returned by `betaRegression`.
`alpha`	OPTIONAL. A DMR contains CpG sites with P values smaller or equal than `alpha`.
`max.dist`	Numeric. The maximum distance between two P values smaller than `alpha` in a DMR. Should be the same as `grid.dist` in `predictMeth`.
`diff.dir`	Logical. Should DMRs be seperated if the direction of methylation differences changes? If `TRUE` (default), than resulting DMRs will consist of sites which are all hypomethylated, or hypermethylated respectively.

Value

A `GRanges` object storing the start and end positions of the DMRs with information in metadata columns:

`median.p`	median of P values
`median.meth.group1`	median of modeled methylation level of group1.
`median.meth.group1`	median of modeled methylation level of group2.
`median.meth.diff`	median of difference of modeled methylation levels of group1 and group2.

Author(s)

Katja Hebestreit

See Also

predictMeth, betaRegression

Examples

```
## Variogram under Null hypothesis (for resampled data):
data(vario)

plot(vario$variogram$v)
vario.sm <- smoothVariogram(vario, sill=0.9)

# auxiliary object to get the pValsList for the test
# results of interest:
data(betaResults)
vario.aux <- makeVariogram(betaResults, make.variogram=FALSE)

# Replace the pValsList slot:
vario.sm$pValsList <- vario.aux$pValsList

## vario.sm contains the smoothed variogram under the Null hypothesis as
## well as the p Values that the group has an effect on DNA methylation.

locCor <- estLocCor(vario.sm)

clusters.rej <- testClusters(locCor, FDR.cluster = 0.1)

clusters.trimmed <- trimClusters(clusters.rej, FDR.loc = 0.05)

DMRs <- findDMRs(clusters.trimmed, max.dist=100, diff.dir=TRUE)
```

`globalTest` *Test whether at least one CpG is differentially methylated in a given genomic region*

Description

This method is a wrapper for conveniently invoking the globaltest method `gt` on a BSrel-class object. The globaltest can be applied to test against a high dimensional alternative in various regression models. E.g., it can be used to test whether at least one CpG is differentially methylated between two groups.

Usage

```
globalTest(response, alternative, ...)
```

Arguments

response	The response vector of the regression model. May be supplied as a vector or as a formula object. In the latter case, the right hand side of response defines the null hypothesis. The default null hypopthesis is ~1, i.e. only an intercept.
alternative	An object of BSrel-class defining the alternative. The CpGs are used as explanatory variable in the alternative regression model. The null hypothesis is that the coefficients of all CpGs are zero. If response is given as formula, colData(alternative) is used to obtain the respective data.
...	Other arguments passed to the gt method. The argument subsets can be given as GRanges-class object. Then, the globaltest is applied for each region using only the CpGs lying within the respective region. This is useful for, e.g., testing all promoter regions within function call.

Details

For details see the documentation of the gt method in package globaltest.

Value

The function returns an object of class gt.object. Several operations and diagnostic plots for this class are provided by the globaltest package.

Author(s)

Hans-Ulrich Klein

References

Goeman, J. J., van de Geer, S. A., and van Houwelingen, J. C. (2006). Testing against a high-dimensional alternative. Journal of the Royal Statistical Society Series B- Statistical Methodology, 68(3):477-493.

See Also

link{gt}, link{BSrel}

Examples

```
data(rrbs)
rrbs <- rawToRel(rrbs)
regions <- GRanges(IRanges(start=c(850000, 1920000, 500), end=c(879000, 1980000, 600)),
                   seqnames=c("chr1", "chr2", "chr3"))

globalTest(group~1, rrbs)
globalTest(group~1, rrbs, subsets=regions)
```

limitCov *Limits the coverage of a* BSraw *object*

Description

Number of methylated and unmethylated reads of a CpG site with coverage above maxCov are reduced such that the methylation level remains unchanged.

Usage

```
limitCov(object, maxCov)
```

Arguments

object	A BSraw.
maxCov	The maximum number of reads a CpG should have. All coverages above this threshold are limited. (Default is 50)

Details

This function might be useful prior to the use of predictMeth to limit the weights of CpGs with extremly high coverages. See binomLikelihoodSmooth for details.

Value

A BSraw object.

Author(s)

Katja Hebestreit

See Also

predictMeth, binomLikelihoodSmooth

Examples

```
data(rrbs)

rrbs.clust.unlim <- clusterSites(object = rrbs,
                                 groups = colData(rrbs)$group,
                                 perc.samples = 4/5,
                                 min.sites = 20, max.dist = 100)
covBoxplots(rrbs.clust.unlim)

# 90% quantile of coverage is 39x
quantile(totalReads(rrbs.clust.unlim)[totalReads(rrbs.clust.unlim)>0],
0.9)
```

```
rrbs.clust.lim <- limitCov(rrbs.clust.unlim, maxCov = 39)
covBoxplots(rrbs.clust.lim)
```

`logisticRegression` *Estimates and tests a group factor*

Description

It is used to fit a linear model on the log odds of each (smoothed) methylation level. The first independent variable in `formula` is tested to be unequal to zero.

Usage

```
logisticRegression(formula, link, object, mc.cores)
```

Arguments

`formula`	An object of class `formula`. For the first independent variable the p-value and the effect on methylation is returned.
`link`	A character specifying the link function. Currently, `"logit"`, `"probit"`, `"cloglog"`, `"log"`, `"loglog"` are supported.
`object`	A `BSrel` object.
`mc.cores`	Passed to `mclapply`.

Value

A `data.frame` containing the position, chromosome, P value, estimated methylation level in group 1 and group 2 and methylation difference of group 1 and group 2.

Author(s)

Katja Hebestreit

See Also

`mclapply`, `glm`

Examples

```
data(predictedMeth)

logisticResults <- logisticRegression(formula = ~group, link = "logit",
                                      object = predictedMeth)
```

makeVariogram *Variogram estimator.*

Description

A function which estimates the variogram of the z-scores in the given data frame.

Usage

```
makeVariogram(test.out, make.variogram, sample.clusters, max.dist)
```

Arguments

test.out	A data.frame. Usually the output of betaRegression. Must contain columns chr, pos, p.val and cluster.id.
make.variogram	A logical. Default is TRUE.
sample.clusters	Can speed up variogram estimation significantly. Default is NULL, and all data is used to estimate the variogram. If set to numeric, the variogram will be estimated on the basis of the data of randomly selected sample.clusters only. Especially useful if there are many clusters.
max.dist	Can speed up variogram estimation significantly. The variogram is estimated for distances until this threshold. Default is 500 base pairs, since the variogram usually does not change for distances larger than 100 base pairs, because methylation of CpG sites further away are not correlated anymore. Especially useful if there are large clusters.

Details

For each CpG site the z-score is determined by qnorm(1 - P value). The variogram of the z-scores of locations k and l within one cluster is estimated robustly by

$$2\hat{\gamma}(h) = [median(Z_k - Z_l)^2 : (s_k, s_l) \in N(h)]/.455$$

.

Value

A list:

variogram	A list of two: A matrix, called v with columns h and v, and a numeric, called h.est. v comprises the data that was used to estimate the variogram. h.est comprises the distances seen in the data. If sample.clusters=NULL, h.est is identical to v$h.
pValsList	A list of data frames. Each data frame corresponds to a CpG cluster and contains same information as test.out plus the columns z.score and pos.new (position corresponding to the respective CpG cluster).

Author(s)

Katja Hebestreit

References

Yoav Benjamini and Ruth Heller (2007): False Discovery Rates for Spatial Signals. American Statistical Association, 102 (480): 1272-81.

See Also

betaRegression

Examples

```
data(betaResults)

vario <- makeVariogram(betaResults)

plot(vario$variogram$v)
```

---

plotBindingSites *Plots the mean methylation of given regions*

Description

plotBindingSites takes several genomic regions (e.g. protein binding sites), centers them such that the position 0 refers to the center of each region and finally calculates the mean methylation of all regions for each given sample. If several samples are given, the median of the samples' methylation values and optionally other quantiles are plotted.

Usage

```
plotBindingSites(object, regions, width, groups, quantiles, bandwidth, ...)
```

Arguments

object	An object of class BSraw or BSrel.
regions	Regions given a GRanges object. The regions may have different widths.
width	The width of the genomic region that is plotted. Default value is the width of the largest given region.
groups	An optional factor defining two or more groups within the given object. The mean methylation is than plotted for each group separately.
quantiles	Other quantiles to be plotted besides the median. Default are the 25% and the 75% quantiles.
bandwidth	The bandwidth of the kernel smoother used for smoothing methylation values. Default value is 1/8 width.
...	Other graphical parameters passed to the plot function.

Details

First, all regions were expanded or shrinked to the given `width` by adding or removing base pairs symmetrically at both ends of the regions (not by scaling). A new coordinate system is centered at the middle of the equally sized regions. Next, the relative methylation values for each sample are averaged accross all regions. That means, if there are several CpGs from different regions lying the same position, the mean methylation value is calculated for that position. Then, the median of these methylation values across all samples is calculated. Optionally, other quantiles are calculated, too. The median of the methylation is then plotted for each position after smoothing using a gaussian kernel with the given bandwidth.

If the given regions correspond to binding sites of a certain protein, the plot can be used to discover whether the protein induces changes in the DNA methylation in the proximity of its binding sites.

Author(s)

Hans-Ulrich Klein

See Also

BSraw-class, BSrel-class

Examples

```
data(rrbs)
data(promoters)
plotBindingSites(object=rrbs,
                 regions=promoters,
                 width=4000,
                 groups=colData(rrbs)$group)
```

`plotMeth` *Plots raw and smoothed methylation data for a given region*

Description

This function plots the raw and the smoothed methylation data for one sample and a given region. The smoothed data is shown as a line (one line per CpG cluster) and the raw data is shown as points with color intensities proportional to the coverage.

Usage

```
plotMeth(object.raw, object.rel, region, col.lines, lwd.lines, col.points, ...)
```

Arguments

object.raw	A BSraw with only one sample.
object.rel	A BSrel with only one sample.
region	A GRanges of length one.
col.lines	OPTIONAL. The color for the line representing the smoothed methylation values.
lwd.lines	OPTIONAL. The line width for the line representing the smoothed methylation values.
col.points	OPTIONAL. The color for the points representing the raw methylation levels.
...	Other graphical parameters passed to the plot function.

Author(s)

Katja Hebestreit

See Also

plotSmoothMeth, plot

Examples

```
data(rrbs)
data(predictedMeth)

region <- GRanges(seqnames="chr1",
                  ranges=IRanges(start = 875200,
                                 end = 875500))
plotMeth(object.raw = rrbs[,6],
         object.rel = predictedMeth[,6],
         region = region)
```

---

plotMethMap *Plots methylation values of multiple samples in a given region*

Description

A heatmap like plot is generated showing the relative methylation of single CpG sites. Samples are clustered hierarchically.

Usage

```
plotMethMap(object, region, groups, intervals, ...)
```

Arguments

object	A BSraw or BSrel object storing the methylation values.
region	A GRanges object giving the region of interest.
groups	OPTIONAL. A factor that will be encoded by a color bar.
intervals	OPTIONAL. A logical indicating whether neighboured CpG sites should be placed side by side (if FALSE) or whether the intervals between CpG sites should be preserved (if TRUE).
...	Further arguments passed to the heatmap function.

Details

The relative methylation values are passed to the heatmap function. Default colors are green (not methylated), black and red (methylated). To ensure that a relative methylation of 0 corresponds to green, 0.5 to black and 1 to red, the default value for the zlim argument of the heatmap function is set to c(0,1). And the default for the scale parameter is set to "none".
If argument intervals is set to TRUE, region should not be too large (< 1kb) and respect the resolution of your screen.

Author(s)

Hans-Ulrich Klein

See Also

heatmap, BSraw-class, BSrel-class, filterBySharedRegions, filterByCov

Examples

```
data(rrbs)
data(predictedMeth)
data(DMRs)

plotMethMap(rrbs, region = DMRs[4], groups = colData(rrbs)[, "group"])

plotMethMap(predictedMeth, region = DMRs[4],
            groups = colData(rrbs)[,"group"], intervals = FALSE)
```

---

plotSmoothMeth *Plots smoothed methylation values for a bunch of samples and a given region*

Description

This function plots the smoothed methylation data as lines for a given region and all given samples. It is also possible to average the data for groups of samples.

Usage

```
plotSmoothMeth(object.rel, region, groups, group.average, ...)
```

Arguments

object.rel	A BSrel.
region	A GRanges of length one.
groups	OPTIONAL. A factor defining two or more sample groups within the given object.
group.average	OPTIONAL. A logical. If TRUE, then the data is averaged for the groups given in groups. Default is FALSE..
...	Other graphical parameters passed to the plot function.

Author(s)

Katja Hebestreit

See Also

plotMeth, plot

Examples

```
data(predictedMeth)
data(DMRs)

plotSmoothMeth(object.rel = predictedMeth,
               region = DMRs[3] + 200,
               groups = colData(predictedMeth)$group,
               col=c("magenta", "blue"))
legend("topright",
       lty=1,
       legend=levels(colData(predictedMeth)$group),
       col=c("magenta", "blue"))
```

---

predictedMeth *The output of* predictMeth

Description

Please see the package vignette for description.

Usage

```
data(predictedMeth)
```

Format

A BSrel object with the smoothed methylation data.

Examples

```
data(predictedMeth)
show(predictedMeth)
```

---

predictMeth *Predicts methylation levels along CpG sites or for a grid of sites in CpG clusters.*

Description

Uses local regression to predict methylation levels per sample.

Usage

```
predictMeth(object, h, grid.dist, mc.cores)
```

Arguments

object	A BSraw with a cluster.id metadata column on the rowRanges, usually the output of clusterSites.
h	Bandwidth in base pairs. Large values produce a smoother curve. Default is 80.
grid.dist	OPTIONAL. If numeric, than methylation values are predicted at intervals of grid.dist base pairs. By default, methylation is smoothed at each CpG site.
mc.cores	Passed to mclapply. Default is 1.

Details

Uses binomLikelihoodSmooth with pos = CpG position, m = number methylated reads and n = number of reads. pred.pos corresponds to all CpG positions, or to the grid sites respectively, within the CpG clusters.

Value

A BSrel object containing the predicted methylation levels in the methLevel slot.

Author(s)

Katja Hebestreit

See Also

clusterSites, binomLikelihoodSmooth, mclapply

Examples

```
data(rrbs)

rrbs.clust.unlim <- clusterSites(object = rrbs,
                                 groups = colData(rrbs)$group,
                                 perc.samples = 4/5,
                                 min.sites = 20, max.dist = 100)

ind.cov <- totalReads(rrbs.clust.unlim) > 0
quant <- quantile(totalReads(rrbs.clust.unlim)[ind.cov], 0.9)
rrbs.clust.lim <- limitCov(rrbs.clust.unlim, maxCov = quant)

# with a small subset to save calculation time:
rrbs.part <- rrbs.clust.lim[1:100,]

predictedMeth <- predictMeth(object=rrbs.part)
```

---

promoters *A* GRanges *of promoters of the human genome*

Description

Please see the package vignette for description.

Usage

```
data(promoters)
```

Format

A GRanges object with the chromosomes, start and end positions of defined human promoter regions together with an accesion number stored in a metadata column.

Examples

```
data(promoters)
head(promoters)
```

rawToRel *Converts a* BSraw *object to a* BSrel *object*

Description

Determines the methLevel matrix via: methReads(object) / totalReads(object).

Usage

```
rawToRel(object)
```

Arguments

object	A BSraw.

Value

A BSrel.

Author(s)

Katja Hebestreit

See Also

BSraw-class BSrel-class

Examples

```
data(rrbs)
rrbs.rel <- rawToRel(rrbs)
```

readBismark *Reads cytosine methylation stati determined by Bismark*

Description

Bismark is a bisulfite read mapper and methylation caller. This method reads Bismark's output files and returns a BSraw object.

Usage

```
readBismark(files, colData)
```

Arguments

files	A character pointing to cov files created by Bismark's methylation_extractor and bismark2bedGraph; see Details. This can be a compressed file (see file).
colData	Samples' names plus additional sample information as character, data.frame or DataFrame.

Details

Input files are created with Bismark as follows (from the command line):

bismark_methylation_extractor -s --comprehensive test_sample.sam

bismark2bedGraph -o CpG_context_test_sample.bedGraph CpG_context_test_sample.txt

This will output two files, a .bedGraph and a .cov file. We will import the CpG_context_test_sample.cov using readBismark.

The colData argument should specify the sample names as character. Alternatively, a data.frame or DataFrame can be given. Then, the row names are used as sample names and the data frame is passed to the final BSraw object.

Value

A BSraw object storing coverage and methylation information.

Author(s)

Hans-Ulrich Klein

References

http://www.bioinformatics.bbsrc.ac.uk/projects/bismark/

See Also

BSraw-class

Examples

```
file <- system.file("extdata", "CpG_context_test_sample.cov", package = "BiSeq")
rrbs <- readBismark(file,
                    colData= DataFrame(row.names="sample_1"))
```

rrbs *RRBS data of APL patient samples and controls.*

Description

RRBS data of the CpG sites CpG sites from genomic regions on p arms of chromosome 1 and 2 covered in at least one sample. Data was obtained from 5 APL patient samples and 5 control samples (APL in remission). RRBS data was preprocessed with the Bismark software version 0.5.

Usage

```
rrbs
```

Format

A BSraw-class object.

Source

Schoofs T, Rohde C, Hebestreit K, Klein HU, Goellner S, Schulze I, Lerdrup M, Dietrich N, Agrawal-Singh S, Witten A, Stoll M, Lengfelder E, Hofmann WK, Schlenke P, Buechner T, Hansen K, Berdel WE, Rosenbauer F, Dugas M, Mueller-Tidow C (2012). DNA methylation changes are a late event in acute promyelocytic leukemia and coincide with loss of transcription factor binding. Blood.

References

Krueger F, Andrews SR. Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications. Bioinformatics. 2011;27:1571-1572.

Examples

```
data(rrbs)
show(rrbs)
```

smoothVariogram *Smoothes variogram*

Description

Nonparametric smoothing with kernel regression estimators and adaptable bandwidth for variogram smoothing.

Usage

```
smoothVariogram(variogram, sill, bandwidth)
```

Arguments

variogram	A list or a matrix. Usually the output of makeVariogram.
sill	A numeric. The sill (upper bound) of the variogram. See Details.
bandwidth	A numeric vector of same length as the variogram (number of rows). Default: seq(10,1000, length.out=nrow(variogram)). See Details.

Details

It is necessary to smooth the variogram. Especially for greater h the variogram tends to oscillate strongly. This is the reason why the default bandwidth increases with increasing h. Nevertheless, the smoothed variogram may further increase or decrease after a horizontal part (sill). This is mostly due to the small number of observations for high distances. To wipe out this bias it is useful to set the smoothed variogram to a fixed value above a certain h, usually the mean value of the horizontal part. If a smoothed value $v.sm$ is greater than sill for distance h_{range}, this $v.sm$ and all other smoothed values with $h > h_{range}$ are set to sill. Internally, the function lokerns from package lokerns is used for smoothing.

Value

The variogram matrix (or a list with the variogram matrix) with an additional column of the smoothed v values.

Author(s)

Katja Hebestreit

See Also

makeVariogram, lokerns

Examples

```
data(vario)

# Find out the sill (this is more obvious for larger data sets):
plot(vario$variogram$v)

vario.sm <- smoothVariogram(vario, sill = 0.9)

plot(vario$variogram$v)
lines(vario.sm$variogram[,c("h", "v.sm")],
col = "red")
```

summarizeRegions *Aggregates methylation information of single CpG sites*

Description

This method summarizes the methylation states of single CpG sites to a single methylation state for a given genomic region.

Usage

```
summarizeRegions(object, regions, outputAll)
```

Arguments

object	An BSraw or BSrel object.
regions	A GRanges object storing the genomic regions.
outputAll	A logical. If outputAll = TRUE, all regions will be returned. If FALSE (default), regions are dropped if their coverage is zero.

Details

When the given object is of class BSraw-class, all (methylated) reads of all CpG site lying within a region are summed up and assign as total number of (methylated) reads to that region. It is recommended to use limitCov before applying summarizeRegions to an BSraw-class object in order to avoid an excessive influence of a single CpG site on the methylation value of a region. When the given object is of class BSrel-class, the mean relative methylation of all CpGs within a region is assign to that region.

The rowRanges slot of the returned object is the given object regions with all columns preserved.

Value

An BSraw or an BSrel object storing methylation information about the given regions.

Author(s)

Hans-Ulrich Klein

See Also

BSraw-class, BSrel-class, limitCov

Examples

```
data(rrbs)
rrbs.clustered <- clusterSites(rrbs)
regions <- clusterSitesToGR(rrbs.clustered)

rrbs <- limitCov(rrbs, maxCov=50)
rrbsRegion <- summarizeRegions(rrbs, regions)
totalReads(rrbsRegion)
```

---

testClusters *Tests CpG clusters*

Description

CpG clusters are tested with a cluster-wise FDR level.

Usage

```
testClusters(locCor, FDR.cluster)
```

Arguments

locCor	Output of estLocCor.
FDR.cluster	A numeric. The WFDR (weighted FDR) level at which the CpG clusters should be tested. Default is 0.05.

Details

CpG clusters containing at least one differentially methylated location are detected.

Value

A list is returned:

FDR.cluster	Chosen WFDR (weighted FDR) for clusters.
CpGs.clust.reject	A list of the CpG sites together with test results within clusters that were rejected.
CpGs.clust.not.reject	A list of the CpG sites together with test results within clusters that were not rejected.
clusters.reject	A GRanges of the clusters that were rejected.
clusters.not.reject	A GRanges of the clusters that were not rejected.
sigma.clusters.reject	The standard deviations for z-scores within each rejected cluster.

variogram	The variogram matrix.
m	Number of clusters tested.
k	Number of clusters rejected.
u.1	Cutoff point of the largest P value rejected.

Author(s)

Katja Hebestreit

References

Yoav Benjamini and Ruth Heller (2007): False Discovery Rates for Spatial Signals. American Statistical Association, 102 (480): 1272-81.

See Also

estLocCor, trimClusters

Examples

```
## Variogram under Null hypothesis (for resampled data):
data(vario)

plot(vario$variogram$v)
vario.sm <- smoothVariogram(vario, sill=0.9)

# auxiliary object to get the pValsList for the test
# results of interest:
data(betaResults)
vario.aux <- makeVariogram(betaResults, make.variogram=FALSE)

# Replace the pValsList slot:
vario.sm$pValsList <- vario.aux$pValsList

## vario.sm contains the smoothed variogram under the Null hypothesis as
## well as the p Values that the group has an effect on DNA methylation.

locCor <- estLocCor(vario.sm)

clusters.rej <- testClusters(locCor, FDR.cluster = 0.1)
```

---

trimClusters *Trims CpG clusters*

Description

CpG clusters rejected in a previous step are trimmed.

Usage

```
trimClusters(clusters.rej, FDR.loc)
```

Arguments

clusters.rej	Output of testClusters.
FDR.loc	Location-wise FDR level. Default is 0.2.

Details

Not differentially methylated CpG sites are removed within the CpG clusters rejected by testClusters.

Value

A data.frame containing the differentially methylated CpG sites.

Author(s)

Katja Hebestreit

References

Yoav Benjamini and Ruth Heller (2007): False Discovery Rates for Spatial Signals. American Statistical Association, 102 (480): 1272-81.

See Also

testClusters

Examples

```
## Variogram under Null hypothesis (for resampled data):
data(vario)

plot(vario$variogram$v)
vario.sm <- smoothVariogram(vario, sill=0.9)

# auxiliary object to get the pValsList for the test
# results of interest:
data(betaResults)
vario.aux <- makeVariogram(betaResults, make.variogram=FALSE)

# Replace the pValsList slot:
vario.sm$pValsList <- vario.aux$pValsList

## vario.sm contains the smoothed variogram under the Null hypothesis as
## well as the p Values that the group has an effect on DNA methylation.

locCor <- estLocCor(vario.sm)

clusters.rej <- testClusters(locCor, FDR.cluster = 0.1)
```

```
clusters.trimmed <- trimClusters(clusters.rej, FDR.loc = 0.05)
```

---

vario *Output of* makeVariogram

Description

Please see the package vignette for description.

Usage

```
data(vario)
```

Format

A list consisting of the variogram (a matrix) and the pValsList (a list of the data frames of test results).

Examples

```
data(vario)
names(vario)
```

---

writeBED *Writes* BSraw *and* BSrel *data to a bed file suitable for the IGV*

Description

The created bed files contains an entry for each CpG site. Strand information, relative methylation and absolute number of reads covering the CpG sites are stored. The relative methylation is indicated by colors: green via black to red for unmethylated to methylated.

Usage

```
writeBED(object, name, file)
```

Arguments

object	A BSraw or BSrel object.
name	Track names (sample names) written to the bed file's header.
file	Character vector with names of the bed file.

Details

The written bed file contains the following extra information:

1. score: the relative methylation of the CpG site
2. name: the coverage of the CpG site
3. itemRgb: a color value visualizing the methylation score

A separate bed file is created for each sample in the given object. The lengths of the arguments `name` and `file` should equal the number of samples.

Value

Nothing. Bed files are written.

Author(s)

Hans-Ulrich Klein

See Also

`readBismark`

Examples

```
data(rrbs)
s1 <- rrbs[,1]
out <- tempfile(, fileext = ".bed")
writeBED(s1, name = colnames(s1), file = out)
```

Index

∗ **~smooth**
binomLikelihoodSmooth, 7
∗ **classes**
BSraw-class, 8
BSrel-class, 9
∗ **datasets**
betaResults, 5
betaResultsNull, 6
DMRs, 15
predictedMeth, 29
promoters, 31
rrbs, 34
vario, 40

annotateGRanges, 2
annotateGRanges,GRanges,GRanges,character,character-method *(annotateGRanges)*, 2
annotateGRanges,GRanges,GRanges,character,integer-method *(annotateGRanges)*, 2
annotateGRanges,GRanges,GRanges,character,missing-method *(annotateGRanges)*, 2

betareg, *4*
betaRegression, 4, *20*, *25*
betaRegression,formula,character,BSrel,missing-method *(betaRegression)*, 4
betaRegression,formula,character,BSrel,numeric-method *(betaRegression)*, 4
betaResults, 5
betaResultsNull, 6
binomLikelihoodSmooth, 7, *22*, *30*
binomLikelihoodSmooth,ANY-method *(binomLikelihoodSmooth)*, 7
BSraw *(BSraw-class)*, 8
BSraw,matrix,matrix,GRanges-method *(BSraw-class)*, 8
BSraw-class, 8
BSrel *(BSrel-class)*, 9
BSrel,matrix,GRanges-method *(BSrel-class)*, 9
BSrel-class, 9

clusterSites, 11, *13*, *30*
clusterSites,BSraw,ANY,numeric,missing,missing,missing-method *(clusterSites)*, 11
clusterSites,BSraw,ANY,numeric,missing,numeric,missing-method *(clusterSites)*, 11
clusterSites,BSraw,ANY,numeric,numeric,missing,missing-method *(clusterSites)*, 11
clusterSites,BSraw,ANY,numeric,numeric,numeric,missing-method *(clusterSites)*, 11
clusterSites,BSraw,ANY,numeric,numeric,numeric,numeric-method *(clusterSites)*, 11
clusterSites,BSraw,missing,missing,missing,missing,missing *(clusterSites)*, 11
clusterSites,BSraw,missing,numeric,numeric,numeric,missing *(clusterSites)*, 11
clusterSites,BSraw,missing,numeric,numeric,numeric,numeric *(clusterSites)*, 11
clusterSitesToGR, 12
clusterSitesToGR,BSraw-method *(clusterSitesToGR)*, 12
clusterSitesToGR,BSrel-method *(clusterSitesToGR)*, 12
combine,BSraw,BSraw-method *(BSraw-class)*, 8
combine,BSrel,BSrel-method *(BSrel-class)*, 9
compareTwoSamples, 13
compareTwoSamples,BSraw,character,character,numeric,numeric *(compareTwoSamples)*, 13
compareTwoSamples,BSraw,numeric,numeric,numeric,numeric-method *(compareTwoSamples)*, 13
compareTwoSamples,BSrel,character,character,numeric,numeric *(compareTwoSamples)*, 13
compareTwoSamples,BSrel,numeric,numeric,numeric,numeric-method *(compareTwoSamples)*, 13
covBoxplots, 14
covBoxplots,BSraw-method *(covBoxplots)*, 14

covStatistics, 15
covStatistics,BSraw-method (covStatistics), 15
covStatistics,BSrel-method (covStatistics), 15

DMRs, 15

estLocCor, 16, *38*
estLocCor,list-method (estLocCor), 16

file, *33*
filterByCov, 17, *18*, *28*
filterByCov,BSraw,missing,logical-method (filterByCov), 17
filterByCov,BSraw,missing,missing-method (filterByCov), 17
filterByCov,BSraw,numeric,logical-method (filterByCov), 17
filterByCov,BSraw,numeric,missing-method (filterByCov), 17
filterBySharedRegions, *12*, *17*, 18, *28*
filterBySharedRegions,BSraw,ANY,missing,numeric,missing-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,ANY,missing,numeric,numeric-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,ANY,numeric,missing,missing-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,ANY,numeric,missing,numeric-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,missing,missing,missing,missing-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,missing,missing,missing,numeric-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,missing,missing,numeric,missing-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,missing,missing,numeric,numeric-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,missing,numeric,missing,missing-method (filterBySharedRegions), 18
filterBySharedRegions,BSraw,missing,numeric,missing,numeric-method (filterBySharedRegions), 18
filterBySharedRegions,BSrel,ANY,missing,numeric,ANY-method (filterBySharedRegions), 18
filterBySharedRegions,BSrel,ANY,numeric,missing,ANY-method (filterBySharedRegions), 18
filterBySharedRegions,BSrel,missing,missing,missing,ANY-method (filterBySharedRegions), 18
filterBySharedRegions,BSrel,missing,missing,numeric,ANY-method (filterBySharedRegions), 18
filterBySharedRegions,BSrel,missing,numeric,missing,ANY-method (filterBySharedRegions), 18
findDMRs, 19
findDMRs,data.frame,missing,numeric,logical-method (findDMRs), 19
findDMRs,data.frame,missing,numeric,missing-method (findDMRs), 19
findDMRs,data.frame,numeric,numeric,logical-method (findDMRs), 19
findDMRs,data.frame,numeric,numeric,missing-method (findDMRs), 19

globalTest, 20
globalTest,ANY,BSrel-method (globalTest), 20
gt, *21*

heatmap, *28*

limitCov, 22, *36*
limitCov,BSraw,numeric-method (limitCov), 22
logisticRegression, 23
logisticRegression,formula,character,BSrel,missing-method (logisticRegression), 23
logisticRegression,formula,character,BSrel,numeric-method (logisticRegression), 23

makeVariogram, *16*, 24, *35*
makeVariogram,data.frame,logical,missing,missing-method (makeVariogram), 24
makeVariogram,data.frame,logical,missing,numeric-method (makeVariogram), 24
makeVariogram,data.frame,logical,numeric,missing-method (makeVariogram), 24
makeVariogram,data.frame,logical,numeric,numeric-method (makeVariogram), 24
makeVariogram,data.frame,missing,missing,missing-method (makeVariogram), 24
makeVariogram,data.frame,missing,numeric,numeric-method (makeVariogram), 24
methLevel,BSrel-method (BSrel-class), 9
methLevel<-,BSrel,ANY-method (BSrel-class), 9
methLevel<-,BSrel,matrix-method (BSrel-class), 9
methReads (BSraw-class), 8

methReads,BSraw-method *(BSraw-class)*, 8
methReads<- *(BSraw-class)*, 8
methReads<-,BSraw,matrix-method *(BSraw-class)*, 8

plotBindingSites, 25
plotBindingSites,BSraw,GRanges-method *(plotBindingSites)*, 25
plotBindingSites,BSrel,GRanges-method *(plotBindingSites)*, 25
plotMeth, 26, *29*
plotMeth,BSraw,BSrel,GRanges-method *(plotMeth)*, 26
plotMethMap, 27
plotMethMap,BSraw,GRanges,factor,logical-method *(plotMethMap)*, 27
plotMethMap,BSraw,GRanges,factor,missing-method *(plotMethMap)*, 27
plotMethMap,BSraw,GRanges,missing,logical-method *(plotMethMap)*, 27
plotMethMap,BSraw,GRanges,missing,missing-method *(plotMethMap)*, 27
plotMethMap,BSrel,GRanges,factor,logical-method *(plotMethMap)*, 27
plotMethMap,BSrel,GRanges,factor,missing-method *(plotMethMap)*, 27
plotMethMap,BSrel,GRanges,missing,logical-method *(plotMethMap)*, 27
plotMethMap,BSrel,GRanges,missing,missing-method *(plotMethMap)*, 27
plotSmoothMeth, *27*, 28
plotSmoothMeth,BSrel,GRanges,ANY,logical-method *(plotSmoothMeth)*, 28
plotSmoothMeth,BSrel,GRanges,ANY,missing-method *(plotSmoothMeth)*, 28
predictedMeth, 29
predictMeth, *7*, *14*, *20*, *22*, 30
predictMeth,BSraw,missing,missing,missing-method *(predictMeth)*, 30
predictMeth,BSraw,missing,missing,numeric-method *(predictMeth)*, 30
predictMeth,BSraw,numeric,missing,missing-method *(predictMeth)*, 30
predictMeth,BSraw,numeric,missing,numeric-method *(predictMeth)*, 30
predictMeth,BSraw,numeric,numeric,numeric-method *(predictMeth)*, 30
promoters, 31

RangedSummarizedExperiment, *8–10*
rawToRel, 32
rawToRel,BSraw-method *(rawToRel)*, 32
readBismark, *8–10*, 32, *41*
readBismark,character,character-method *(readBismark)*, 32
readBismark,character,data.frame-method *(readBismark)*, 32
readBismark,character,DataFrame-method *(readBismark)*, 32
rrbs, 34

smoothVariogram, *16*, 34
smoothVariogram,list,numeric,missing-method *(smoothVariogram)*, 34
smoothVariogram,list,numeric,numeric-method *(smoothVariogram)*, 34
smoothVariogram,matrix,numeric,missing-method *(smoothVariogram)*, 34
smoothVariogram,matrix,numeric,numeric-method *(smoothVariogram)*, 34
summarizeRegions, 36
summarizeRegions,BSraw,GRanges,logical-method *(summarizeRegions)*, 36
summarizeRegions,BSraw,GRanges,missing-method *(summarizeRegions)*, 36
summarizeRegions,BSrel,GRanges,logical-method *(summarizeRegions)*, 36
summarizeRegions,BSrel,GRanges,missing-method *(summarizeRegions)*, 36

testClusters, 37, *39*
testClusters,list,missing-method *(testClusters)*, 37
testClusters,list,numeric-method *(testClusters)*, 37
totalReads *(BSraw-class)*, 8
totalReads,BSraw-method *(BSraw-class)*, 8
totalReads<- *(BSraw-class)*, 8
totalReads<-,BSraw,matrix-method *(BSraw-class)*, 8
trimClusters, *38*, 38
trimClusters,list,missing-method *(trimClusters)*, 38
trimClusters,list,numeric-method *(trimClusters)*, 38

vario, 40

writeBED, 40

writeBED,BSraw,character,character-method
(writeBED), 40
writeBED,BSraw,character,missing-method
(writeBED), 40
writeBED,BSraw,missing,character-method
(writeBED), 40
writeBED,BSraw,missing,missing-method
(writeBED), 40
writeBED,BSrel,character,character-method
(writeBED), 40
writeBED,BSrel,character,missing-method
(writeBED), 40
writeBED,BSrel,missing,character-method
(writeBED), 40
writeBED,BSrel,missing,missing-method
(writeBED), 40