

# Package ‘methylPipe’

February 20, 2025

**Type** Package

**Title** Base resolution DNA methylation data analysis

**Version** 1.41.0

**Date** 2022-03-15

**Description** Memory efficient analysis of base resolution DNA methylation data in both the CpG and non-CpG sequence context. Integration of DNA methylation data derived from any methodology providing base- or low-resolution data.

**License** GPL(>=2)

**LazyLoad** yes

**Imports** marray, gplots, IRanges, BiocGenerics, Gviz,  
GenomicAlignments, Biostrings, parallel, data.table,  
GenomeInfoDb, S4Vectors

**Depends** R (>= 3.2.0), methods, grDevices, graphics, stats, utils,  
GenomicRanges, SummarizedExperiment (>= 0.2.0), Rsamtools

**Suggests** BSgenome.Hsapiens.UCSC.hg18,  
TxDb.Hsapiens.UCSC.hg18.knownGene, knitr, MethylSeekR

**VignetteBuilder** knitr

**biocViews** MethylSeq, DNAMethylation, Coverage, Sequencing

**git\_url** <https://git.bioconductor.org/packages/methylPipe>

**git\_branch** devel

**git\_last\_commit** 9ff081a

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2025-02-20

**Author** Mattia Pelizzola [aut],  
Kamal Kishore [aut],  
Mattia Furlan [ctb, cre]

**Maintainer** Mattia Furlan <[mattia.furlan@iit.it](mailto:mattia.furlan@iit.it)>

## Contents

methylPipe-package . . . . .	2
BSdata-class . . . . .	3
BSdataSet-class . . . . .	4
BSprepare . . . . .	5
chiCombP . . . . .	6
consolidateDMRs . . . . .	7
extractBinGRanges . . . . .	8
findDMR . . . . .	9
findPMDs . . . . .	11
GEcollection-class . . . . .	12
GElis-class . . . . .	13
getCpos . . . . .	14
getCposDensity . . . . .	15
mapBSdata2GRanges . . . . .	16
mCsmoothing . . . . .	17
meth.call . . . . .	18
methstats . . . . .	19
plotMeth . . . . .	20
pool.reads . . . . .	22
process.hmc . . . . .	23
profileDNAmethBin . . . . .	24
splitChrs . . . . .	25
tabixdata2GR . . . . .	25

<b>Index</b>	<b>27</b>
--------------	-----------

---

methylPipe-package	<i>Analysis of base-pair resolution DNA methylation data.</i>
--------------------	---

---

### Description

Analysis of base-pair resolution DNA methylation data.

### Details

Package:	methylPipe
Type:	Package
Version:	1.0.5
Date:	2015-02-25
License:	GPL
Depends:	methods

The package offers the following functionalities:

- BSdata-class : This class is used in to point to a TABIX compressed file containing base-resolution DNA-methylation data and reference genome sequence
- mCsmoothing : Smoothing and plotting methylation data, even chromosome wide
- findPMDs : Find partially methylated regions for a given sample
- mapBSdata2GRanges : Retrieve mC calls for a GRanges from a BSdata object for a sample
- BSdataSet-class : This class is a set of BSdata objects
- findDMR : Identifying differentially methylated regions for pairwise or multiple samples comparison
- methstats : Descriptive methylation statistics of samples within BSdataSet object
- consolidateDMRs : Joins differentially methylated regions according to their proximity to each other, statistical significance, methylation difference and DMR type
- GEcollection-class : This class is used to define and manipulate a set of genomic regions and the associated DNA methylation patterns
- getCpos : Get genomic Cxx positons for a series of genomic regions
- getCposDensity : Determines the density of genomic Cxx positions for a series of genomic regions
- profileDNAmethBin : Profile DNA methylation data for a set of genomic regions
- plotMeth : Plot the methylation/omics data of a GEcollection object
- BSprepare : Preparing tabular data to be used to feed a BSdata object
- meth.call : Reads the methylation information from the sorted SAM files generated from BISMARCK aligner
- pool.reads : Combine reads of replicates within a group
- GElist-class : This class is a set of GEcollection objects

### Author(s)

Computational Epigenomics Unit at the Center for Genomic Sciences of IIT@SEMM, Milan, Italy  
<kamal.fartiyal84@gmail.com>

<http://genomics.iit.it/groups/computational-epigenomics.html>

---

BSdata-class

Class "BSdata"

---

### Description

This class is used in the methylPipe library to point to a TABIX compressed file containing base-resolution DNA-methylation data and reference genome sequence

### Objects from the Class

Objects can be created by calls of the form `new("BSdata", ...)` or using the function `BSdata(file, uncov, org)` whose arguments are described in the next section Slots.

**Slots**

**file:** Object of class "character" : the TABIX compressed file containing base-resolution DNA-methylation data. This file can be generated through the [BSprepare](#) function, and it must contain the following columns: chromosome assignment (in the form chr1, chr2...), genomic position (positive integer), strand (either - or +), methylation sequence context (either CG, CHG or CHH), number (>0) of sequencing reads with C calls at that genomic position, number of sequencing reads with T calls at that genomic position, binomial pvalue ( $-10 \cdot \log_{10}(\text{pvalue})$ ) for calling a mC at that position.

**uncov:** Object of class [GRanges](#) : this [GRanges](#) object consists of the list of genomic regions with sequencing coverage information; this information is used to distinguish which methylation sites are unmethylated, but covered, from those that are missing data since they have no sequencing coverage. This object is automatically generated by the [meth.call](#) function while processing aligned files generated from the aligner.

**org:** refrence genome of class BSgenome

**Author(s)**

Mattia Pelizzola

**See Also**

[BSprepare](#), [mCsmoothing](#)

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
```

---

BSdataSet-class

*Class "BSdataSet"*

---

**Description**

This class is used in the methylPipe library to store a set of BSdata objects

**Objects from the Class**

Objects can be created by calls of the form `new("BSdataSet", ...)` or using the function `BSdataSet(org,Objlist,names)`, see below.

**Slots**

**Objlist:** Object of class "list" : a list with more than one item, where each item is a [BSdata](#) object

**names:** Object of class "character" : vector of the names of the objects, i.e. the sample names

**group:** Object of class "character" : indicating conditions and replicates; replicated samples within the same condition have to be assigned the same group name; if object has only two groups, "C"(control) and "E" (Experiment) should be specified as groups name

**org:** refrence genome of class BSgenome

**Methods**

"[" signature(x = "BSdataSet"): subsets the [BSdataSet](#) returning a specific [BSdata](#) object

"[<-" signature(x = "BSdataSet"): replaces the specific [BSdata](#) object in the [BSdataSet](#)

"[" signature(x = "BSdataSet"): subsets the [BSdataSet](#) returning another [BSdataSet](#)

**Author(s)**

Mattia Pelizzola, Kamal Kishore

**See Also**

[BSdata-class](#), [findDMR](#)

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
IMR90data <- system.file('extdata', 'IMR90_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
IMR90.db <- BSdata(file=IMR90data, uncov=uncov_GR, org=Hsapiens)
H1.IMR90.set <- BSdataSet(org=Hsapiens, group=c("C","E"), IMR90=IMR90.db, H1=H1.db)
```

---

BSprepare

*Preparing tabular data to be used to feed a BSdata object*

---

**Description**

Appending p-values and TABIX compressing tabular data containing DNA-methylation data so that they can be used to create a BSdata object.

**Usage**

```
BSprepare(files_location, output_folder, tabixPath, bc=1.5/100)
```

### Arguments

files\_location character; the path to the files  
output\_folder character; the path to the output files  
tabixPath character; the path to the Tabix application folder  
bc numeric; combined bisulfite conversion and sequencing error rate

### Details

This function can be used to convert tabular files containing DNA-methylation base-resolution data so that they can be used to create a BSdata object. Genomic coordinates in the 1-base system are required, i.e. the first base of each chromosome should be at position 1. Files have to be tab-delimited and they have to contain the following columns: chromosome assignment (in the form chr1, ..., chr22, chrX, chrY, chrM, chrC), genomic position (positive integer), strand (either - or +), methylation sequence context (either CG, CHG or CHH), number of sequencing reads with C calls (>0) at that genomic position, number of sequencing reads with T calls at that genomic position. Each file has to be sorted by genomic coordinate.

### Value

Binomial p-values are added and a compressed file is created together with the Tabix index. P-values indicate for each Cytosine the probability of observing by chance the occurrence of unmethylated reads. The lower the p-value the higher the confidence in calling that Cytosine methylated.

### Author(s)

Mattia Pelizzola, Kamal Kishore

### See Also

[BSdata-class](#)

### Examples

```
#BSprepare("/path-to-input/", "/path-to-output/", tabix="/path-to-tabix/tabix-0.2.6")
```

---

chiCombP

*Fisher's method implementation*

---

### Description

Fisher's method implementation, used to combine the results from several independent tests bearing upon the same overall hypothesis.

### Usage

```
chiCombP(Pvalues)
```

**Arguments**

Pvalues            an array of pvalues

**Details**

Pvalues will not be corrected for multiple testing. The sum of the log of the pvalues is determined (S).  $-2*S$  has a chi-square distribution with  $2k$  degrees of freedom, where  $k$  is the number of tests being combined. A chi-square test is then performed.

**Value**

The chi-square final pvalue

**Author(s)**

Mattia Pelizzola

**Examples**

```
chiCombP(c(1e-3, 1e-5, 1e-2))
```

---

consolidateDMRs            *Consolidating Differentially Methylated Regions (DMRs)*

---

**Description**

Joins differentially methylated regions according to their proximity to each other, statistical significance and methylation difference

**Usage**

```
consolidateDMRs(DmrGR, pvThr=0.05, MethDiff_Thr=NULL, log2Er_Thr =NULL, GAP=0, type=NULL, correct=FALSE)
```

**Arguments**

DmrGR	the GRanges object resulting from the <a href="#">findDMR</a> function
pvThr	numeric; the minimum pvalue for a DMR to be selected
MethDiff_Thr	numeric; the absolute value of minimum methylation difference percentage (for both hyper- and hypo-methylation) cutoff for the selection of a DMR
log2Er_Thr	numeric; the absolute value of minimum log2Enrichment (for both hyper- and hypo-methylation) cutoff for the selection of a DMR
GAP	numeric; the minimum distance between two adjacent DMRs; DMRs closer than that will be joined, resulting DMRs will be updated mean methylation difference and Pvalues combined using the Fisher's Method
type	character; one of the "hyper" or "hypo", specifies the type of differentially methylated regions selected
correct	logical; whether to correct the pvalues using the Benjamini-Hochberg multiple testing correction method

## Details

After the DMRs are identified by [findDMR](#) method, a consolidation can be applied on them. This functions allows to select hyper/hypo differentially methylated regions based on P-value and absolute methylation change thresholds. Moreover, DMRs closer than a given distance can be joined. The final [GRanges](#) object with the set of final DMRs will be provided with updated mean methylation difference and Pvalues combined using the Fisher's Method.

## Value

Either NULL or a [GRanges](#) object containing the differential methylated regions satisfying the criteria.

## Author(s)

Kamal Kishore

## See Also

[findDMR](#)

## Examples

```
DMRs_file <- system.file('extdata', 'DMRs.Rdata', package='methylPipe')
load(DMRs_file)
hyper.DMRs.conso <- consolidateDMRs(DmrGR=DMRs, pvThr=0.05, GAP=100, type="hyper", correct=TRUE)
hyper.DMRs.conso
```

---

extractBinGRanges	<i>Extract genomic ranges for a given bin</i>
-------------------	---

---

## Description

For genomic ranges with N bins, extract the Genomic ranges for a given bin.

## Usage

```
extractBinGRanges(GenoRanges, bin, nbins)
```

## Arguments

GenoRanges	An object of class <a href="#">GRanges</a>
bin	numeric; the bin corresponding to which data has to be extracted
nbins	numeric; the number of bins in which genomic regions are divided

## Value

A [GRanges](#) Object



**Author(s)**

Mattia Pelizzola

**See Also**[mapBSdata2GRangesBin](#)**Examples**

```
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
extractBinGRanges(GR_chr20, 2, 5)
```

findDMR

*Identifying Differentially Methylated Regions (DMRs)***Description**

Identifying differentially methylated regions for pairwise or multiple samples comparison.

**Usage**

```
## S4 method for signature 'methylPipe,BSdataSet'
findDMR(object, Nproc=NULL, ROI=NULL,
pmdGRanges=NULL, MCClass='mCG', dmrSize=10, dmrBp=1000, binsize=0,
eprop=0.3, coverage=1, Pvalue=NULL, SNPs=NULL)
```

**Arguments**

object	An object of class <a href="#">BSdataSet</a>
Nproc	numeric; the number of processors to use, one chromosome is ran for each processor
ROI	character; either NULL or an object of class <a href="#">GRanges</a> consisting of genomic regions of interest for which DMRs are identified
pmdGRanges	a <a href="#">GRanges</a> object containing the genomic coordinates of Partially Methylated Domains that will be masked
MCClass	character; the mC sequence context to be considered, one of all, mCG, mCHG or mCHH
dmrSize	numeric; the number of consecutive mC to be simulataneously considered; atleast 5
dmrBp	numeric; the max number of bp containing the dmrSize mC
binsize	numeric; the size of the bin used for smoothing the methylation levels, useful for nonCG methylation in human
eprop	numeric; the max - min methylation level is determined for each mC, or for each bin, and only mC (or bins) with difference greater than eprop are considered

coverage	numeric; the minimum number of total reads at a given cytosine genomic position
Pvalue	numeric; to select only those mC with significant p-value
SNPs	GRanges; if SNPs information is provided those cytosine are removed from DMR computation

### Details

Typically for nonCG methylation in human a dmrSize of 50, a dmrBp of 50000 and a binsize of 1000 are used. For CpG methylation in human and both CpG and nonCG methylation in plants the default settings are usually fine. Partially Methylated Domains are usually found in differentiated cells and can constitute up to one third of the genome (Lister R et al, Nature 2009). Usually DMRs are not selected within those regions especially when comparing differentiated and pluripotent cells. Eprop is used to speed up the analysis. According to the number of samples different test are used to compare the methylation levels (percentage of methylated reads for each mC). In case of two samples the non parametric wilcoxon test is used. In case of more than two samples the kruskal wallis non parametric test is used. Check [consolidatedDMRs](#) to further process and finalize the list of DMRs.

### Value

A [GRanges](#) object of DMRs with the metadata slots for pValue, MethDiff\_Perc and log2Enrichment. When two samples are compared, MethDiff\_Perc is the difference between percentage methylation between the conditions compared. However, log2Enrichment is the log2ratio between the mean for the samples.

### Author(s)

Mattia Pelizzola, Kamal Kishore

### See Also

[consolidatedDMRs](#)

### Examples

```
require(BSgenome.Hsapiens.UCSC.hg18)
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
IMR90data <- system.file('extdata', 'IMR90_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
IMR90.db <- BSdata(file=IMR90data, uncov=uncov_GR, org=Hsapiens)
H1.IMR90.set <- BSdataSet(org=Hsapiens, group=c("C","E"), IMR90=IMR90.db, H1=H1.db)
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
DMRs <- findDMR(object= H1.IMR90.set, Nproc=1, ROI=GR_chr20, MCClass='mCG',
dmrSize=10, dmrBp=1000, eprop=0.3)
head(DMRs)
```

---

findPMDs	<i>Identifying Partially Methylated Domains (PMDs)</i>
----------	--

---

**Description**

This function is a wrapper function to identify partially methylated domains (PMDs) in Bis-seq data.

**Usage**

```
## S4 method for signature 'methylPipe,BSdata'  
findPMDs(Object, Nproc=1, Chrs=NULL)
```

**Arguments**

Object	An object of class <a href="#">BSdataSet</a>
Nproc	numeric; the number of processors to use, one chromosome is ran for each processor
Chrs	character; Chromosome on which PMDs are identified

**Details**

This functions is a wrapper function of segmentPMDs method of package MethylSeekR. This function trains a Hidden Markov Model (HMM) to detect partially methylated domains (PMDs) in Bis-seq data.

**Value**

A GRangesList object containing segments that partition the genome into PMDs and regions outside of PMDs. The object contains two metadata columns indicating the type of region (PMD/notPMD) and the number of covered (by at least 5 reads) CpGs (nCG) in the region.

**Author(s)**

Kamal Kishore

**See Also**

[findDMR](#)

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)  
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))  
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')  
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)  
PMDs <- findPMDs(H1.db, Nproc=1, Chrs="chr20")
```

---

GEcollection-class      *Class "GEcollection"*

---

### Description

This class is used in the methylPipe library to define and manipulate a set of genomic regions and the associated DNA methylation patterns

### Objects from the Class

This class is an extension of the [RangedSummarizedExperiment](#) class from the **SummarizedExperiment** package. Objects can be created using the function `profileDNAMetBin` which determines the absolute and relative methylation level by filling the `binC`, `binmC` and `binrC` slots. The assays slot of the [RangedSummarizedExperiment](#) class here consists of four matrices:

- `binC`: each genomic region is divided in one or more bins and for each bin the density (per bp) of potential methylation sites is determined.
- `binmC`: each genomic region is divided in one or more bins and for each bin the density (per bp) of methylation events is determined.
- `binrC`: each genomic region is divided in one or more bins and for each bin the relative mC/C content is determined.
- `binscore`: each genomic region is divided in one or more bins and scores can be assigned to them. In particular, it can be convenient for storing reads count for each bin of each genomic region.

The minimal set of data to create a [GEcollection](#) object is a set of genomic regions to be provided as a [GRanges](#) object and a dataset of class [BSdata](#).

### Methods

**chr** signature(object = "GEcollection"): extracts the chr assignment of the genomic regions

**Strand** signature(object = "GEcollection"): extracts the strand assignment of the genomic regions

**binC** signature(object = "GEcollection"): extracts the binC matrix

**binmC** signature(object = "GEcollection"): extracts the binmC matrix

**binrC** signature(object = "GEcollection"): extracts the binrC matrix

**binscore** signature(object = "GEcollection"): extracts the binscore matrix

**binscore<-** signature(object = "GEcollection"): replaces the binscore matrix

**nbins** signature(object = "GEcollection"): returns the number of bins

### Author(s)

Kamal Kishore

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
gec.H1 <- profileDNAMetBin(GenoRanges=GR_chr20, Sample=H1.db, mcCLASS='mCG')
gec.H1
```

GElis-class

*Class "GElis"***Description**

This class is used in the methylPipe library to collect a set of GEcollection objects

**Objects from the Class**

Objects can be created by calls of the form `new("GElis", ...)` or using the function `GElis(Objlist,names)`, see below. GElis are a collection of GEcollection objects (see [GElis-class](#)).

**Slots**

`Objlist`: Object of class "list" : a list where each item is a [GEcollection](#) object

`names`: Object of class "character" : vector of the names of the objects

**Methods**

`"["` signature(`x = "GElis"`): subsets the [GElis](#) returning a specific [GEcollection](#) object

`"[["` signature(`x = "GElis"`): replaces the specific [GEcollection](#) object in the [GElis](#)

`"["` signature(`x = "GElis"`): subsets the [GElis](#) returning another [GElis](#)

**Author(s)**

Mattia Pelizzola

**See Also**

[GElis-class](#)

**Examples**

```
gecollect_file <- system.file('extdata', 'gec.H1.Rdata', package='methylPipe')
load(gecollect_file)
gec1 <- gec.H1[start(gec.H1) < 153924]
gec2 <- gec.H1[start(gec.H1) > 153924]
gel.set <- GElis(g1=gec1, g2=gec2)
```

---

getCpos

*Get genomic Cxx positons for a series of genomic regions*


---

### Description

getCpos retrieves genomic Cxx positions, possible target of DNA methylation for a series of genomic regions (and bins thereof) and a given organism. getCposChr is a Helper function which performs the same task for any given DNASTring sequence and is not intended for the user to call directly.

### Usage

```
getCpos(GenoRanges, seqContext='all', nbins, org)
getCposChr(GenoRanges, seqContext, chrseq, nbins)
```

### Arguments

GenoRanges	An object of class <a href="#">GRanges</a>
seqContext	character; one of: all, CG, CHG or CHH
org	an object of class BSgenome; typically the genome sequences of a given organism
chrseq	an object of class <a href="#">DNASTring</a> ; typically a chromosome sequence of a given organism
nbins	numeric; the number of bins each region of genomic regions is divided

### Value

A list is returned with the position of the Cxx in the [GRanges](#) regions. The length of the list is equal to the length of the [GRanges](#). For each list item a list with length equal to the number of bins of the [GRanges](#) is returned. For each bin the position of the Cxx relative to the genomic coordinates of that bin is returned.

### Author(s)

Mattia Pelizzola

### See Also

[getCposDensity](#), [profileDNAMetBin](#)

### Examples

```
require(BSgenome.Hsapiens.UCSC.hg18)
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
res <- getCpos(GR_chr20, seqContext='CG', nbins=1, org=Hsapiens)
res[[1]]
```

---

getCposDensity	<i>Determines the density of genomic Cxx positions for a series of genomic regions</i>
----------------	--

---

### Description

After having used `getCpos` (or `getCposChr`), `getCposDensity` determines the density of Cxx sites for each bin of each genomic region.

### Usage

```
getCposDensity(GenoRanges, Cpos, nbins)
```

### Arguments

GenoRanges	an object of class <a href="#">GRanges</a> used to generate the Cpos list
Cpos	list returned by <code>getCpos</code> or <code>getCposChr</code> methods
nbins	numeric; the number of bins each region of genomic regions is divided

### Value

Returns a list with the number of Cxx sites per bp of bin size for each region of the [GRanges](#).

### Author(s)

Mattia Pelizzola

### See Also

[getCpos](#), [profileDNAmetBin](#)

### Examples

```
require(BSgenome.Hsapiens.UCSC.hg18)
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
resC <- getCposChr(GenoRanges=GR_chr20, seqContext='CG', chrseq=unmasked(Hsapiens[['chr20']]), nbins=3)
resd <- getCposDensity(GenoRanges=GR_chr20, Cpos= resC, nbins=3)
```

---

mapBSdata2GRanges	<i>Retrieve mC calls for a GRanges set of genomic regions given a BSdata object for a sample</i>
-------------------	--

---

### Description

mapBSdata2GRanges retrieves mC calls for a [GRanges](#) given a [BSdata](#) object for a sample. mapBSdata2GRangesBin does the same for each bin of each genomic region.

### Usage

```
mapBSdata2GRanges(GenoRanges, Sample, context='all', mC=1, depth=0, pValue=1)
mapBSdata2GRangesBin(GenoRanges, Sample, context='all', mC=1, depth=0, pValue=1, nbins)
```

### Arguments

GenoRanges	An object of class <a href="#">GRanges</a>
Sample	An object of class <a href="#">BSdata</a>
context	character; one of: all, CG, CHG or CHH
mC	numeric; the minimum number of reads with C (DNA-methylation events) at a given cytosine genomic position
depth	numeric; the minimum number of total reads at a given cytosine genomic position
pValue	numeric; the minimum binomial pValue for an mC call at a given cytosine genomic position
nbins	numeric; the number of bins in which Genomic regions are divided

### Details

DNA-methylation data contained for a sample within a [BSdata](#) object is extracted for the set of genomic regions of a [GRanges](#) (and in particular for each bin using the mapBSdata2GRangesBin method). It is also possible to restrict the mC sequence context, to specify the minimal number of reads with C events at a given cytosine genomic position, to specify the minimum depth of sequencing and binomial pValue for the mC calls. A region with no mC will be defined unmethylated (0 is returned for that region). However, if it is overlapping with at least one uncovered region then it is defined non evaluable (NA is returned).

### Value

A list is returned. The length of the list is equal to the length of the [GRanges](#). For each list item either NA, 0 or a data frame are returned. 0 means that the region contains unmethylated DNA methylation sites, whereas NA indicates that the region or some part of region was not covered by the sequencing. If a data frame is returned, it has the following columns: chromosome assignment (in the form chr1, ..., chr22, chrX, chrY, chrM, chrC), genomic position (positive integer), strand (either - or +), methylation sequence context (either CG, CHG or CHH), number (>0) of sequencing reads with C calls at that genomic position, number of sequencing reads with T calls at that genomic position, binomial pvalue ( $-10 \cdot \log_{10}(\text{pvalue})$ ) for calling a mC at that position.



**Author(s)**

Mattia Pelizzola, Kamal Kishore

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
res <- mapBSdata2GRanges(GenoRanges=GR_chr20, Sample=H1.db, context='CG', mC=1, depth=0, pValue=1)
resbin <- mapBSdata2GRangesBin(GenoRanges=GR_chr20, Sample=H1.db, context='CG', mC=1, depth=0, pValue=1, nbins=2)
```

---

mCsmoothing

*Smoothing and plotting methylation data*


---

**Description**

Smoothing and plotting methylation data, even chromosome wide.

**Usage**

```
## S4 method for signature 'methylPipe,BSdata'
mCsmoothing(Object, refgr, Scorefun='sum', Nbins=20,
Context="CG", plot=TRUE)
```

**Arguments**

Object	An object of class <a href="#">BSdata</a>
refgr	GRanges; Genomic Ranges to plot the data
Scorefun	character; either sum or mean for smoothing
Nbins	numeric; the number of interval each range is divided
Context	character; either all or a combination of CG, CHG, and CHH
plot	logical; whether the smoothed profile has to be plotted

**Details**

The sum or the mean methylation level is determined on each window of size Binsize and smoothed with the smooth.spline function.

**Value**

A list with three components: pos (the left most point of each window), score (either the sum or the mean methylation levels), smoothed (the smoothed methylation levels).

**Author(s)**

Mattia Pelizzola

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
gr <- GRanges("chr20",IRanges(1,5e5))
sres <- mCsmoothing(H1.db, gr, Scorefun='sum', Nbins=50, Context="CG", plot=TRUE)
```

meth.call

*Function to read methylation calls***Description**

Reads the methylation calls from sorted SAM files generated from Bismark aligner.

**Usage**

```
meth.call(files_location, output_folder, no_overlap, read.context, Nproc)
```

**Arguments**

files_location	character; the path(s) to the folder location consisting of sorted SAM files
output_folder	character; the path(s) to the folder location where the output files are written
no_overlap	character; if set to TRUE and the SAM file has paired-end reads, then one read of the overlapping paired-end read pair will be ignored for methylation calling
read.context	character; One of the 'CpG' or 'All'. Determines what type of methylation context will be read. If given as 'all', cytosine methylation information in all sequence context will be read.
Nproc	numeric; the number of processors to use, one sample is processed by each processor.

**Details**

The function reads methylation calls from the sorted SAM file so that they can be used to create a [BSdata](#) object. SAM files must be sorted based on chr and start of reads. The user can specify the sequence context in which the methylation information is read from these files either "CpG" or "All". If "All" is specified, cytosine methylation in all context (CG, CHG or CHH) will be read. The methylation calls is saved as a text file in the output folder. These text files are tab-delimited and contain the following columns: chromosome assignment (in the form chr1, chr2..), genomic position (positive integer), strand (either - or +), methylation sequence context (either CG, CHG or CHH), number (>0) of sequencing reads with C calls at that genomic position, number of sequencing reads with T calls at that genomic position. In addition a GRanges object consisting of

uncovered genomic regions is generated and saved in the output folder for each sample. This information is used to distinguish unmethylated cytosines from those that are not covered by sequencing. This GRanges object is used further to provide uncovered regions information while creating BSdata object by [BSdata](#) method.

### Value

A text file of methylation calls and a [GRanges](#) object consisting of uncovered genomic regions for each sample are generated in the "output\_folder" folder. The files are prefixed with sample name.

### Author(s)

Kamal Kishore

### See Also

[BSprepare](#)

### Examples

```
file_loc <- system.file('extdata', 'test_methcall', package='methylPipe')
meth.call(files_location=file_loc, output_folder=tempdir(), no_overlap=TRUE, read.context="CpG", Nproc=1)
```

---

methstats

*Exploratory statistics of samples in BSdataSet object*

---

### Description

Exploratory methylation statistics of samples in BSdataSet object.

### Usage

```
## S4 method for signature 'methylPipe,BSdataSet'
methstats(object, chrom, mcClass='mCG', minC=1, coverage=1, pval=1, Nproc=1)
```

### Arguments

object	An object of class <a href="#">BSdataSet</a>
chrom	character; either NULL or an object of class character
mcClass	character; the mC sequence context to be considered, one of all, mCG, mCHG or mCHH
minC	numeric; the minimum number of reads with C (DNA-methylation events) at a given cytosine genomic position
coverage	numeric; the minimum number of total reads at a given cytosine genomic position
pval	numeric; the minimum binomial pValue for an mC call at a given cytosine genomic position
Nproc	numeric; the number of processors to use, one chromosome is ran for each processor

**Details**

The function provides basic statistical methods which computes descriptive statistics, correlation matrix and clustering of samples within the BSdataSet.

**Value**

A list with slots named descriptive\_stats and correlation\_mat.

**Author(s)**

Kamal Kishore

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
uncov_GR <- GRanges('chr20', IRanges(14350, 18349))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
IMR90data <- system.file('extdata', 'IMR90_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
IMR90.db <- BSdata(file=IMR90data, uncov=uncov_GR, org=Hsapiens)
H1.IMR90.set <- BSdataSet(org=Hsapiens, group=c("C","C","E","E"), IMR_1=IMR90.db,
IMR_2=IMR90.db, H1_1=H1.db,H1_2=H1.db)
stats_res <- methstats(H1.IMR90.set,chrom="chr20",mcClass='mCG', Nproc=1)
stats_res
```

---

plotMeth

*Plot DNA methylation together with other omics, or annotation data for a genomic region*

---

**Description**

Plot DNA methylation data (either high- or low-resolution) together with other omics data (ChIP-seq, RNA-seq), or annotation tracks for one genomic region (genome-browser like view based on gviz).

**Usage**

```
plotMeth(grl=NULL, colors=NULL, datatype=NULL, yLim, brmeth=NULL, mcContext="CG", annodata=NULL,
transcriptDB, chr, start, end, org)
```

**Arguments**

grl	An object of class <a href="#">GEList</a> or a potentially mixed <a href="#">list</a> of <a href="#">GRanges</a> or <a href="#">GEcollection</a> objects
colors	character of length equal to grl; name of colors to display data tracks from the grl object
datatype	character of length equal to grl; one of C, mC, rC, density or cols

<code>yLim</code>	numeric vector with the same length of <code>grl</code> setting maximum values
<code>brmeth</code>	A list of object of class <code>BSdata</code>
<code>mcContext</code>	character; one of all, CG, CHG or CHH
<code>annodata</code>	An object of class <code>GRangesList</code>
<code>transcriptDB</code>	An object of class <code>TranscriptDb</code>
<code>chr</code>	character; chromosome name
<code>start</code>	numeric; chromosome start
<code>end</code>	numeric; chromosome end
<code>org</code>	<code>BSgenome</code> ; an object of class <code>BSgenome</code>

### Details

This function can be used to display for one genomic region (genome-browser like) DNA methylation data together with other omics data or static annotation info. The genomic region is indicated by `chr`, `start` and `end`. Specifically, `grl` is used to display binned high- or low-resolution data, while `brmeth` is used to point to (unbinned) base-resolution data. Each component of `grl` can either be a `GEcollection` or a `GRanges`.

In case of `GEcollection`, `binC`, `binmC` or `binrC` components will be extracted as indicated in `datatype` (setting `C`, `mC` or `rC`, respectively), and if more than 1 bin is present the average value will be considered for each range. `Datatype` can be set to `density` to extract the `binscore` component of the `GEcollection`, which can be used to store low-resolution or other omics data attached to a base-resolution dataset.

In case of a `GRanges` (suitable for low-resolution or other omics data independently from base-resolution data), only the 1st column of the `mcols` will be considered. Eventually, for both `GEcollection` and `GRanges` tracks, a bar with the specific value will be displayed for the ranges occurring in the considered region (if any).

Regarding unbinned base-resolution data, `mcContext` defines the sequence context to be considered for the methyl-cytosines for each component of the `brmeth` object; a bar with height equal to the methylation level of each cytosine will be displayed for each sample (track).

`Annodata` is an optional `GRangesList` that can be used to display co-occurring annotation data, such as CpG islands (presence or absence of the regions only). `transcriptDB` and `BSgenome` are used to overlay the structure of annotated genes and chromosome ideogram, respectively.

### Author(s)

Kamal Kishore

### Examples

```
require(TxDb.Hsapiens.UCSC.hg18.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg18.knownGene
require(BSgenome.Hsapiens.UCSC.hg18)
gecH1_file <- system.file('extdata', 'gec.H1.Rdata', package='methylPipe')
load(gecH1_file)
gecIMR_file <- system.file('extdata', 'gec.IMR90.Rdata', package='methylPipe')
load(gecIMR_file)
```

```

gel <- GElist(gecH1=gec.H1, gecIMR90=gec.IMR90)
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
H1.db <- BSdata(file=H1data, uncov=uncov_GR, org=Hsapiens)
IMR90data <- system.file('extdata', 'IMR90_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
IMR90.db <- BSdata(file=IMR90data, uncov=uncov_GR, org=Hsapiens)
H1.IMR90.set <- list(H1=H1.db, IMR90=IMR90.db)
plotMeth(gel, colors=c("red","blue"), datatype=c("mC","mC"), yLim=c(.025, .025), brmeth=H1.IMR90.set, mcContext=

```

---

pool.reads

*Function to pool reads of replicates*

---

## Description

Combine reads of replicates within a group.

## Usage

```
pool.reads(files_location)
```

## Arguments

`files_location` character; the path to the folder location consisting of tab separated text files

## Details

The function reads tab separated text files of methylation calls generated from [meth.call](#) or user supplied according to the format specified for [BSprepare](#) method. It pools all the reads of the replicates within a single group for each cytosine position and creates a file consisting of the cytosines with pooled reads information.

## Value

A text file of methylation calls are generated in the "files\_location" folder.

## Author(s)

Kamal Kishore

## See Also

[BSprepare](#)

## Examples

```
#pool.reads(files_location)
```

---

`process.hmc`*Processing hmC information from the MLML output*

---

## Description

Processing MLML software output to generate files with hmC and CpG methylation information.

## Usage

```
process.hmc(file, output_folder, Coverage)
```

## Arguments

<code>file</code>	character; the path to the file
<code>output_folder</code>	character; the path to the output files
<code>Coverage</code>	<code>GRanges</code> ; the object containing coverage for each cytosine

## Details

This function allows processing of the output files from MLML software (Qu et al, Bioinformatics 2013). MLML read counts from BS-seq, oxBS-seq and TAB-seq to provide simultaneous estimates of 5hmC and 5mC levels. The input for this function is output file from this software along with a `GRanges` object consisting of coverage of each cytosine. The `GRanges` object should contain "coverage" column. This object can be generated using the `coverage` method of R package `GenomicRanges`.

## Value

The function will return two files one each for "CpG" and "hmC" for the given sample which can directly be used for `BSdata` object creation.

## Author(s)

Kamal Kishore

## See Also

[BSdata-class](#)

## Examples

```
#process.hmc(file, "/path-to-output/", Coverage)
```

---

profileDNAMetBin      *Profile DNA methylation data for a set of genomic regions*

---

### Description

Profile the absolute and relative density of mC sites for each bin of each genomic region of a [GEcollection](#) object.

### Usage

```
profileDNAMetBin(GenoRanges, Sample, mcCLASS="mCG",
  mC=1, depthThr=0, mCpv=1, minCoverage=0.75, nbins = 2)
profileDNAMetBinParallel(GenoRanges, Sample, mcCLASS="mCG", mC=1,
  depthThr=0, mCpv=1, minCoverage=0.75, Nproc=1, nbins = 2)
```

### Arguments

GenoRanges	an object of class <a href="#">GRanges</a>
Sample	an object of class <a href="#">BSdata</a>
mcCLASS	character; one of: mCG, mCHG, mCHH
mC	numeric; the minimum number of reads with C (DNA-methylation events) at a given cytosine genomic position
depthThr	numeric; the minimum number of total reads at a given cytosine genomic position
mCpv	numeric; the minimum binomial pValue for an mC call at a given cytosine genomic position
minCoverage	numeric between 0 and 1; the minimum coverage of for the genomic region to be profiled; currently ignored
Nproc	numeric; the number of processor for parallelization
nbins	numeric; the number of bins each genomic region is divided

### Details

For each bin of each genomic region of a [GRanges](#) object, the absolute and relative density of mC sites is determined and an object of class [GEcollection](#) is created.

### Value

An object of class [GRanges](#) from which an object of class [GEcollection](#) is created with the binC, binmC and binrC data slots been filled with density matrices. These matrices report the density of mC sites in the sequence context specified by mcCLASS. They are counted for each bin in each genomic region and their count is divided by the bin size in bp. The binC data slot is filled with the density of all possible methylation sites in the specified sequence context. The binmC data slot is filled with the density of mC sites in the specified sequence context for the considered sample. The binrC data slot is filled with the ratio of binC and binmC matrices, representing the relative methylation for each bin in each genomic region.



**Author(s)**

Mattia Pelizzola, Kamal Kishore

**Examples**

```
require(BSgenome.Hsapiens.UCSC.hg18)
H1data <- system.file('extdata', 'H1_chr20_CG_10k_tabix_out.txt.gz', package='methylPipe')
uncov_GR <- GRanges(Rle('chr20'), IRanges(c(14350,69251,84185), c(18349,73250,88184)))
H1.db <- BSdata(file=H1data, uncov= uncov_GR, org=Hsapiens)
gr_file <- system.file('extdata', 'GR_chr20.Rdata', package='methylPipe')
load(gr_file)
gec.H1 <- profileDNAMetBin(GenoRanges=GR_chr20, Sample=H1.db, mcCLASS='mCG', nbins=2)
head(binmC(gec.H1))
```

---

splitChrs

*Partitioning genome in chunks, for parallel computation*


---

**Description**

Helper function to partition genome chromosome-wise for parallel computation. This function is not intended for the user to call directly.

**Usage**

```
splitChrs(chrs, org)
```

**Arguments**

chrs            character; an array of chromome names in the form chr1, ..., chrX  
org             an object of class BSgenome

**Value**

A data frame with chromosome name, start and end position of each chunk.

---

tabixdata2GR

*Convert the list returned by the function scanTabix into a GRanges*


---

**Description**

Helper function to convert the list returned by the function scanTabix into a GRanges. This function is not intended for the user to call directly.

**Usage**

```
tabixdata2GR(x)
```

**Arguments**

x list; the list returned by the function scanTabix

**Value**

An object of class data frame.

**Author(s)**

Mattia Pelizzola, Kamal Kishore

**See Also**

[BSdata-class](#)

# Index

- \* **classes**
  - BSdata-class, [3](#)
  - BSdataSet-class, [4](#)
  - GEcollection-class, [12](#)
  - GElist-class, [13](#)
- \* **package**
  - methylPipe-package, [2](#)
- [,BSdataSet,ANY,ANY-method (BSdataSet-class), [4](#)
- [,GElist,ANY,ANY-method (GElist-class), [13](#)
- [[,BSdataSet,ANY,ANY-method (BSdataSet-class), [4](#)
- [[,GElist,ANY,ANY-method (GElist-class), [13](#)
- [[<-,BSdataSet,ANY,ANY-method (BSdataSet-class), [4](#)
- [[<-,GElist,ANY,ANY-method (GElist-class), [13](#)
- \$,BSdataSet (BSdataSet-class), [4](#)
- \$,BSdataSet-method (BSdataSet-class), [4](#)
- \$,GElist (GElist-class), [13](#)
- \$,GElist-method (GElist-class), [13](#)
  
- binC (GEcollection-class), [12](#)
- binC,GEcollection-method (GEcollection-class), [12](#)
- binmC (GEcollection-class), [12](#)
- binmC,GEcollection-method (GEcollection-class), [12](#)
- binrC (GEcollection-class), [12](#)
- binrC,GEcollection-method (GEcollection-class), [12](#)
- binscore (GEcollection-class), [12](#)
- binscore,GEcollection-method (GEcollection-class), [12](#)
- binscore<- (GEcollection-class), [12](#)
- binscore<-,GEcollection-method (GEcollection-class), [12](#)
- BSdata, [5](#), [12](#), [16–19](#), [21](#), [23](#), [24](#)
- BSdata (BSdata-class), [3](#)
- BSdata-class, [3](#), [26](#)
- BSdataSet, [5](#), [9](#), [11](#), [19](#)
- BSdataSet (BSdataSet-class), [4](#)
- BSdataSet-class, [4](#)
- BSprepare, [4](#), [5](#), [19](#), [22](#)
  
- chiCombP, [6](#)
- chr (GEcollection-class), [12](#)
- chr,GEcollection-method (GEcollection-class), [12](#)
- consolidateDMRs, [7](#), [10](#)
- coverage, [23](#)
  
- DNAStrng, [14](#)
  
- extractBinGRanges, [8](#)
  
- findDMR, [5](#), [7](#), [8](#), [9](#), [11](#)
- findDMR,BSdataSet-method (findDMR), [9](#)
- findDMR,methylPipe,BSdataSet (findDMR), [9](#)
- findDMR,methylPipe,BSdataSet-method (findDMR), [9](#)
- findDMR-methods (findDMR), [9](#)
- findPMDs, [11](#)
- findPMDs,BSdata-method (findPMDs), [11](#)
- findPMDs,methylPipe,BSdata (findPMDs), [11](#)
- findPMDs,methylPipe,BSdata-method (findPMDs), [11](#)
- findPMDs-methods (findPMDs), [11](#)
  
- GEcollection, [12](#), [13](#), [20](#), [24](#)
- GEcollection (GEcollection-class), [12](#)
- GEcollection-class, [12](#)
- GElist, [13](#), [20](#)
- GElist (GElist-class), [13](#)
- GElist-class, [13](#)
- GenomicRanges, [23](#)
- getCpos, [14](#), [15](#)

- getCposChr (getCpos), [14](#)
- getCposDensity, [14](#), [15](#)
- GRanges, [4](#), [8–10](#), [12](#), [14–16](#), [19](#), [20](#), [23](#), [24](#)
- GRangesList, [21](#)
- length (GEcollection-class), [12](#)
- length, BSdataSet-method  
(BSdataSet-class), [4](#)
- length, GEcollection-method  
(GEcollection-class), [12](#)
- length, GElist-method (GElist-class), [13](#)
- list, [20](#)
- mapBSdata2GRanges, [16](#)
- mapBSdata2GRangesBin, [9](#)
- mapBSdata2GRangesBin  
(mapBSdata2GRanges), [16](#)
- mCsmoothing, [4](#), [17](#)
- mCsmoothing, BSdata-method  
(mCsmoothing), [17](#)
- mCsmoothing, methylPipe, BSdata  
(mCsmoothing), [17](#)
- mCsmoothing, methylPipe, BSdata-method  
(mCsmoothing), [17](#)
- mCsmoothing-methods (mCsmoothing), [17](#)
- meth.call, [4](#), [18](#), [22](#)
- methstats, [19](#)
- methstats, BSdataSet-method (methstats),  
[19](#)
- methstats, methylPipe, BSdataSet  
(methstats), [19](#)
- methstats, methylPipe, BSdataSet-method  
(methstats), [19](#)
- methstats-methods (methstats), [19](#)
- methylPipe (methylPipe-package), [2](#)
- methylPipe-package, [2](#)
- nbins (GEcollection-class), [12](#)
- nbins, GEcollection-method  
(GEcollection-class), [12](#)
- plotMeth, [20](#)
- pool.reads, [22](#)
- process.hmc, [23](#)
- profileDNAMetBin, [14](#), [15](#), [24](#)
- profileDNAMetBinParallel  
(profileDNAMetBin), [24](#)
- RangedSummarizedExperiment, [12](#)
- show, BSdata-method (BSdata-class), [3](#)
- show, BSdataSet-method  
(BSdataSet-class), [4](#)
- show, GEcollection-method  
(GEcollection-class), [12](#)
- show, GElist-method (GElist-class), [13](#)
- splitChrs, [25](#)
- tabixdata2GR, [25](#)