Package ‘MethCP’

October 17, 2020

Type Package

Title Differential methylation anlsysis for bisulfite sequencing data

Version 1.2.0

Description MethCP is a differentially methylated region (DMR) detecting method for whole-genome bisulfite sequencing (WGBS) data, which is applicable for a wide range of experimental designs beyond the two-group comparisons, such as time-course data. MethCP identifies DMRs based on change point detection, which naturally segments the genome and provides region-level differential analysis.

License Artistic-2.0

Encoding UTF-8

LazyData TRUE

RoxygenNote 7.1.0

biocViews DifferentialMethylation, Sequencing, WholeGenome, TimeCourse

Imports methods, utils, stats, S4Vectors, bsseq, DSS, methylKit, DNAcopy, GenomicRanges, IRanges, GenomeInfoDb, BiocParallel

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

Depends R (>= 3.6.0)

BugReports https://github.com/boyinggong/methcp/issues

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

git_branch RELEASE_3_11

git_last_commit 41cf2b6

git_last_commit_date 2020-04-27

Date/Publication 2020-10-16

Author Boying Gong [aut, cre]

Maintainer Boying Gong <jorothy_gong@berkeley.edu>

R topics documented:

calcLociStat *Calculate the per-cytosine statistics for simple two-group comparisons.*

Description

calcLociStat calculates per-cytosine based statistics between two population groups.

Usage

```
calcLociStat(
    bs.object, group1, group2, test = c("DSS", "methylKit"),
    BPPARAM = bpparam())
```

Arguments

bs.object	a BSseq object from the bsseq package.
group1	a character vector containing the sample names of the treatment group.
group2	a character vector containing the sample names of the control group.
test	a character string containing the names of the test to be performed per cytosine.
BPPARAM	An optional BiocParallelParam instance determining the parallel back-end to be used during evaluation, or a list of BiocParallelParam instances, to be applied in sequence for nested calls to BiocParallel functions. Default bpparam().

Details

For each cytosine, calcLociStat calculates a statistics using either package DSS or methylKit to test the differences between two groups, and returns a MethCP object. For customized per-cytosine statistics, please use the function methcpFromStat. The input bs.object is a BSseq object from the bsseq package which contains the raw data including coverges, methylated counts and position infomation for every cytosine in the dataset.

Value

a MethCP object that is not segmented.

Examples

```
library(bsseq)
library(GenomicRanges)
library(IRanges)

set.seed(0286374)

# Similate a small dataset with 11 cyotsine and 6 samples,
# 3 in the treatment group and 3 in the control group. The
# methylation ratio are generated using Binomial distribution
# with probability 0.3.
nC <- 2000
sim_cov <- rnbinom(6*nC, 5, 0.5) + 5
sim_M <- vapply(
    sim_cov, function(x) rbinom(1, x, 0.3),
    FUN.VALUE = numeric(1))
sim_cov <- matrix(sim_cov, ncol = 6)
sim_M <- matrix(sim_M, ncol = 6)
# methylation ratios in the DMRs in the treatment group are
# generated using Binomial(0.7)
DMRs <- c(600:622, 1089:1103, 1698:1750)
sim_M[DMRs, 1:3] <- vapply(
    sim_cov[DMRs, 1:3], function(x) rbinom(1, x, 0.7),
    FUN.VALUE = numeric(1))
# sample names
sample_names <- c(paste0("treatment", 1:3), paste0("control", 1:3))
colnames(sim_cov) <- sample_names
colnames(sim_M) <- sample_names

# create a bs.object
bs_object <- BSseq(gr = GRanges(
    seqnames = "Chr01", IRanges(
        start = (1:nC)*10, width = 1)),
    Cov = sim_cov, M = sim_M, sampleNames = sample_names)
# methcp_obj1 <- calcLociStat(
#     bs_object,
#     group1 = paste0("treatment", 1:3),
#     group2 = paste0("control", 1:3),
#     test = "DSS")
methcp_obj2 <- calcLociStat(
    bs_object,
    group1 = paste0("treatment", 1:3),
    group2 = paste0("control", 1:3),
    test = "methylKit")
```

---

calcLociStatTimeCourse

Calculate the per-cytosine statistics for time-course data.

Description

For each cytosine, calcLociStatTimeCourse fits a linear model on the arcsin-tranformed methylation ratios, and test the differences of the slope between the treatment and the control group.

Usage

```
calcLociStatTimeCourse(
    bs.object, meta, force.slope = FALSE,
    BPPARAM = bpparam())
```

Arguments

bs.object	a BSseq object from the bsseq package.
meta	a data.frame. See details.
force.slope	if TRUE, we force the slope in the linear model to be the same between two conditions. Otherwise, the slopes are fitted separately but not tested.
BPPARAM	An optional BiocParallelParam instance determining the parallel back-end to be used during evaluation, or a list of BiocParallelParam instances, to be applied in sequence for nested calls to BiocParallel functions. Default bpparam().

Details

bs.object is a BSseq object from the bsseq package, which contains the raw data including coverges, methylated counts and position infomation for every cytosine in the dataset. meta must contain columns named Condition, Time and SampleName in the dataframe. They are used to fit the linear model.

Value

A MethCP object that is not segmented.

Examples

```
library(bsseq)
# Simulate a small dataset with 2000 cyotsine and 10 samples,
# 5 in the treatment group and 5 in the control group. The
# methylation ratio are generated using Binomial distribution
# with probability 0.3, 0.4, 0.5, 0.6 and 0.7 for 5 time points.
nC <- 2000
nsamples <- 5
sim_cov <- rnbinom(10*nC, 5, 0.5) + 5
sim_cov <- matrix(sim_cov, ncol = 10)
time_point <- rep(1:nsamples, 2)
ratios <- time_point/10 + 0.2
sim_M <- sapply(1:(2*nsamples), function(i){
    sapply(sim_cov[, i], function(j) rbinom(1, j, ratios[i]))
})
sim_M <- matrix(sim_M, ncol = 2*nsamples)
# methylation ratios in the DMRs in the treatment group are
# generated using Binomial(0.3)
DMRs <- c(600:622, 1089:1103, 1698:1750)
sim_M[DMRs, 1:5] <- sapply(
    sim_cov[DMRs, 1:5], function(x) rbinom(1, x, 0.3))
# sample names
sample_names <- c(paste0("treatment", 1:nsamples),
paste0("control", 1:nsamples))
colnames(sim_cov) <- sample_names
colnames(sim_M) <- sample_names
```

```
# create a bs.object
bs_object_ts <- BSseq(gr = GRanges(
    seqnames = "Chr01", IRanges(
        start = (1:nC)*10, width = 1)),
    Cov = sim_cov, M = sim_M, sampleNames = sample_names)
DMRs_pos_ts <- DMRs*10
meta <- data.frame(
    Condition = rep(
        c("treatment", "control"),
        each = nsamples),
    SampleName = sample_names,
    Time = time_point)
obj_ts <- calcLociStatTimeCourse(bs_object_ts, meta)
obj_ts
```

createBsseqObject *Helper function to read text files and create a bsseq object.*

Description

Create a bsseq object when the data for each sample is stored in a separate text file.

Usage

```
createBsseqObject(
    files, sample_names,
    chr_col, pos_col, m_col, cov_col, header = TRUE)
```

Arguments

files	a charactor vector of file names with full path to the file.
sample_names	a charactor vector of sample names. It should have the same length as files vector.
chr_col	name or index of the chromosome column in data files.
pos_col	name or index of the position column in data files.
m_col	name or index of the methylated counts column.
cov_col	name or index of the coverage counts column.
header	a logical value indicating whether the file contains the names of the variables as its first line. dedault TRUE.

Value

a MethCP object that is not segmented.

Examples

```
library(bsseq)
# The dataset is consist of 6 samples. 3 samples are H2A.Z mutant
# plants, and 3 samples are controls.
sample_names <- c(
    paste0("control", seq_len(3)),
    paste0("treatment", seq_len(3))
)

# Get the vector of file path and names
raw_files <- system.file(
    "extdata", paste0(sample_names, ".txt"), package = "MethCP")

# load the data
bs_object <- createBsseqObject(
    files = raw_files, sample_names = sample_names,
    chr_col = 'Chr', pos_col = 'Pos', m_col = "M", cov_col = 'Cov')
```

getSigRegion *Obtain the significant DMRs.*

Description

getSigRegion returns the significant DMRs giving the segmented MethCP object.

Usage

```
getSigRegion(
    object, sig.level = 0.01, mean.coverage = 1,
    mean.diff = 0.1, nC.valid = 10)
```

Arguments

object	a MethCP object that is segmented using function segmentMethCP.
sig.level	significance level to call a region DMR.
mean.coverage	The minimum average coverage required for the reported DMRs.
mean.diff	The minimum differences between groups required for the reported DMRs.
nC.valid	number of valid cytosines required for the reported DMRs.

Value

a data.frame containing the DMRs.

Examples

```
library(bsseq)
# Simulate a small dataset with 2000 cyotsine and 6 samples,
# 3 in the treatment group and 3 in the control group. The
# methylation ratio are generated using Binomial distribution
# with probability 0.3.
nC <- 2000
```

```
sim_cov <- rnbinom(6*nC, 5, 0.5) + 5
sim_M <- vapply(
    sim_cov, function(x) rbinom(1, x, 0.3), FUN.VALUE = numeric(1))
sim_cov <- matrix(sim_cov, ncol = 6)
sim_M <- matrix(sim_M, ncol = 6)
# methylation ratios in the DMRs in the treatment group are
# generated using Binomial(0.7)
DMRs <- c(600:622, 1089:1103, 1698:1750)
sim_M[DMRs, 1:3] <- vapply(
    sim_cov[DMRs, 1:3], function(x) rbinom(1, x, 0.7),
    FUN.VALUE = numeric(1))
# sample names
sample_names <- c(paste0("treatment", 1:3), paste0("control", 1:3))
colnames(sim_cov) <- sample_names
colnames(sim_M) <- sample_names

# create a bs.object
bs_object <- BSseq(gr = GRanges(
    seqnames = "Chr01",
    IRanges(start = (1:nC)*10, width = 1)),
    Cov = sim_cov, M = sim_M, sampleNames = sample_names)
DMRs_pos <- DMRs*10
methcp_obj1 <- calcLociStat(
    bs_object,
    group1 = paste0("treatment", 1:3),
    group2 = paste0("control", 1:3),
    test = "methylKit")
methcp_obj1 <- segmentMethCP(
    methcp_obj1, bs_object,
    region.test = "fisher")
methcp_res1 <- getSigRegion(methcp_obj1)
```

MethCP-class *Class* MethCP

Description

A class for performing DMR (differentially methylated region) detection analysis using method MethCP on whole genome bisulfite sequencing data.

The constructor function for MethCP objects.

Usage

```
MethCP(
  test = NA_character_,
  group1 = NA,
  group2 = NA,
  chr,
  pos,
  pvals,
  effect.size
)
```

```
MethCP(
    test = NA_character_, group1 = NA, group2 = NA, chr, pos,
    pvals, effect.size)
```

Arguments

`test`	a character string of the name of the per-cytosine statistcis.
`group1`	a character vector containing the sample names of the treatment group.
`group2`	a character vector containing the sample names of the control group.
`chr`	a character vector containing the cytosine chromosome infomation.
`pos`	a numeric vector containing the cytosine positions.
`pvals`	a numeric vector containing the p-values for each cytosine.
`effect.size`	a numeric vector containing the effect sizes for each cytosine.

Details

If not specified by function `calcLociStatTimeCourse`, `calcLociStat`, the parameter `test` can be set to any user-specified string indicating the name of the test performed.

In the cases where the goal is not to compare between treatment and control groups, parameter `group1` and `group2` can be set to `NA`.

If generated by `calcLociStat`, parameter `stat` will be a `GRangesList` object where each element in the list contains statistics for each of the chromosome in the dataset.

Value

A `MethCP` objects.

Slots

`test` a character string of the name of the per-cytosine statistcis.

`group1` a character vector containing the sample names of the treatment group.

`group2` a character vector containing the sample names of the control group.

`stat` a `GRangesList` object containing the results of per-cytosine tests.

`segmentation` a `GRanges` object containing the segments and their infomation such as region-based statistics, coverages, etc.

Examples

```
obj <- MethCP(
    test = "myTest",
    group1 = paste0("Treatment", 1:3),
    group2 = paste0("Control", 1:3),
    chr = rep("Chr01", 5),
    pvals = c(0, 0.1, 0.9, NA, 0.02),
    pos = c(2, 5, 9, 10, 18),
    effect.size = c(1, -1, NA, 9, Inf))
# MethCP will omit the NAs and infinite values.
obj
```

`MethCPFromStat` *Create a* `MethCP` *object given the per-cytosine based tests.*

Description

Given the per-cytosine based p-values and effect sizes, `MethCPFromStat` create a `MethCP` object for segmentation.

Usage

```
MethCPFromStat(
  data,
  test.name,
  pvals.field = "pvals",
  effect.size.field = "effect.size",
  seqnames.field = c("seqnames", "seqname", "chromosome", "chrom", "chr",
    "chromosome_name", "seqid"),
  pos.field = "pos"
)
```

Arguments

`data` a data.frame or GPos or GRanges object.

`test.name` a character string containing the name of the test to be performed per cytosine.

`pvals.field` A character vector of recognized names for the column (if ‘data‘ is a data.frame) or meta data column (is ‘data‘ is GPos or GRanges object) in ‘data‘ that contains the p-value.

`effect.size.field` A character vector of recognized names for the column (if ‘data‘ is a data.frame) or meta data column (is ‘data‘ is GPos or GRanges object) in ‘data‘ that contains the effect size.

`seqnames.field` A character vector of recognized names for the column in ‘data‘ that contains the chromosome name (sequence name) associated with each position. Only the first name in seqnames.field that is found in colnames(data) is used. If no one is found, then an error is raised. This column is only used when ‘data‘ is a data.frame. Otherwise, chromosome name is obtained from GPos or GRanges object.

`pos.field` A character vector of recognized names for the column in df that contains the position integer associated with each position. Only the first name in pos.field that is found in colnames(data) is used. If no one is found, then an error is raised. This column is only used when ‘data‘ is a data.frame. Otherwise, position is obtained from GPos or GRanges object.

Value

a `MethCP` object that is not segmented.

Examples

```
# ====== construct using data frame
data <- data.frame(
    chr = rep("Chr01", 5),
    pos = c(2, 5, 9, 10, 18),
    effect.size = c(1,-1, NA, 9, Inf),
    pvals = c(0, 0.1, 0.9, NA, 0.02))
obj <- MethCPFromStat(
    data, test.name="myTest",
    pvals.field = "pvals",
    effect.size.field="effect.size",
    seqnames.field="chr",
    pos.field="pos"
)
# ====== construct using GRanges
library(GenomicRanges)
data <- GRanges(
    "Chr01", IRanges(c(2, 5, 9, 10, 18), c(2, 5, 9, 10, 18)),
    pvals=c(0, 0.1, 0.9, NA, 0.02), effect.size = c(1,-1, NA, 9, Inf))
obj <- MethCPFromStat(
    data, test.name="myTest",
    pvals.field = "pvals",
    effect.size.field="effect.size"
)
```

segmentMethCP *Perform segmentation on a* MethCP *object.*

Description

Perform CBS algorithm that segments the genome into similar levels of sigficance.

Usage

```
segmentMethCP(
    methcp.object, bs.object,
    region.test = c(
        "fisher", "stouffer", "weighted-variance", "weighted-coverage"),
    min.width = 2, sig.level = 0.01,
    presegment_dist = 600, BPPARAM = bpparam(), ...)
```

Arguments

`methcp.object`	a MethCP object.
`bs.object`	a BSseq object from the bsseq package.
`region.test`	The meta-analysis method used to create region-based test statistics.
`min.width`	the minimum width for the segments, which is used as termination rule for the segmentation algorithm.
`sig.level`	the significance level of the segments, which is used as termination rule for the segmentation algorithm.

`presegment_dist`
the maximum distance between cytosines for the presegmentation.

`BPPARAM` An optional BiocParallelParam instance determining the parallel back-end to be used during evaluation, or a list of BiocParallelParam instances, to be applied in sequence for nested calls to BiocParallel functions. Default bpparam().

`...` argument to be passed to segment function in DNAcopy package

Details

The `MethCP` object `methcp.object` can be generated from functions `calcLociStat`, `calcLociStatTimeCourse`, or `methcpFromStat`.

If `region.test` = `"fisher"`, Fisher's combined probability test is used.

If `region.test` = `stouffer` Stouffer's test is applied.

If `region.test` = `"weighted-variance"` we use the variance of the test to combine per-cytosine based statistcis into a region-based statistic.

If `region.test` = `"weighted-coverage"` we use the coverage of the test to combine per-cytosine based statistcis into a region-based statistic.

Value

a `MethCP` object that is not segmented.

Examples

```
library(bsseq)
# Simulate a small dataset with 2000 cyotsine and 6 samples,
# 3 in the treatment group and 3 in the control group. The
# methylation ratio are generated using Binomial distribution
# with probability 0.3.
nC <- 2000
sim_cov <- rnbinom(6*nC, 5, 0.5) + 5
sim_M <- vapply(
    sim_cov, function(x) rbinom(1, x, 0.3), FUN.VALUE = numeric(1))
sim_cov <- matrix(sim_cov, ncol = 6)
sim_M <- matrix(sim_M, ncol = 6)
# methylation ratios in the DMRs in the treatment group are
# generated using Binomial(0.7)
DMRs <- c(600:622, 1089:1103, 1698:1750)
sim_M[DMRs, 1:3] <- vapply(
    sim_cov[DMRs, 1:3], function(x) rbinom(1, x, 0.7),
    FUN.VALUE = numeric(1))
# sample names
sample_names <- c(paste0("treatment", 1:3), paste0("control", 1:3))
colnames(sim_cov) <- sample_names
colnames(sim_M) <- sample_names

# create a bs.object
bs_object <- BSseq(gr = GRanges(
    seqnames = "Chr01", IRanges(start = (1:nC)*10, width = 1)),
    Cov = sim_cov, M = sim_M,
    sampleNames = sample_names)
DMRs_pos <- DMRs*10
methcp_obj1 <- calcLociStat(
    bs_object,
    group1 = paste0("treatment", 1:3),
    group2 = paste0("control", 1:3),
    test = "methylKit")
methcp_obj1 <- segmentMethCP(
    methcp_obj1, bs_object,
    region.test = "fisher")
```

show,MethCP-method *The show method*

Description

Print MethCP object information.

Usage

```
## S4 method for signature 'MethCP'
show(object)
```

Arguments

object a MethCP object.

Value

No value will be returned.

Index