Package 'coMethDMR'

March 4, 2019

Title An unsupervised approach for identifying differentially methylated regions in Illumina arrays

Version 0.0.0.9000

Maintainer Lissette Gomez <lxg255@miami.edu>

Description coMethDMR identifies genomic regions associated with continuous phenotypes by optimally leverages covariations among CpGs within predefined genomic regions. Instead of testing all CpGs within a genomic region, coMethDMR carries out an additional step that selects comethylated sub-regions first without using any outcome information. Next, coMethDMR tests association between methylation within the sub-region and continuous phenotype using a random coefficient mixed effects model, which models both variations between CpG sites within the region and differential methylation simultaneously.

Depends R (>= 3.1.0), IlluminaHumanMethylation450kanno.ilmn12.hg19, IlluminaHumanMethylationEPICanno.ilm10b2.hg19

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports bumphunter, psych, pathwayPCA, lmerTest, GenomicRanges, IRanges

R topics documented:

`betaMatrixChr22_df` *Prefrontal Cortex (PFC) Methylation Data from Alzheimer's Disease subjects*

Description

Subset of an Alzheimer's methylation dataset, with beta values for CpGs.

Usage

```
betaMatrixChr22_df
```

Format

A data frame containing beta values for 8552 CpGs in Chr22 for a subset of 20 subjects.

Source

GEO accession: GSE59685

`betaMatrix_ex1` *Alzheimer's Prefrontal Cortex (PFC) Methylation Data*

Description

Subset of an Alzheimer's Disease methylation data set, with beta values for CpGs.

Usage

```
betaMatrix_ex1
```

Format

A data frame containing beta values for 4 CpGs in one CpG islands for 110 subjects. Each column is a sample, each row is a CpG.

Source

GEO accession: GSE59685

betaMatrix_ex2 *Alzheimer's Prefrontal Cortex (PFC) Methylation Data*

Description

Subset of an Alzheimer's Disease methylation data set, with beta values for CpGs.

Usage

```
betaMatrix_ex2
```

Format

A data frame containing beta values for 4 CpGs in one CpG islands for 110 subjects. Each column is a sample, each row is a CpG.

Source

GEO accession: GSE59685

betaMatrix_ex3 *Alzheimer's Prefrontal Cortex (PFC) Methylation Data*

Description

Subset of an Alzheimer's Disease methylation data set, with beta values for CpGs.

Usage

```
betaMatrix_ex3
```

Format

A data frame containing beta values for 6 CpGs in one CpG islands for 110 subjects. Each column is a sample, each row is a CpG.

Source

GEO accession: GSE59685

betaMatrix_ex4 *Alzheimer's Prefrontal Cortex (PFC) Methylation Data*

Description

Subset of an Alzheimer's Disease methylation data set, with beta values for CpGs.

Usage

```
betaMatrix_ex4
```

Format

A data frame containing beta values for 52 CpGs in one CpG islands for 110 subjects. Each column is a sample, each row is a CpG.

Source

GEO accession: GSE59685

CloseBySingleRegion *Extract clusters of CpGs located closely in a genomic region.*

Description

Extract clusters of CpGs located closely in a genomic region.

Usage

```
CloseBySingleRegion(CpGs_char, arrayType = c("450k", "EPIC"),
  maxGap = 200, minCpGs = 3)
```

Arguments

CpGs_char	a list of CpG IDs
arrayType	Type of array, 450k or EPIC
maxGap	an integer, genomic locations within maxGap from each other are placed into the same cluster
minCpGs	an integer, minimum number of CpGs for the resulting CpG cluster

Details

Note that this function depends only on CpG locations, and not on any methylation data. The algorithm is based on the clusterMaker function in the bumphunter R package. Each cluster is essentially a group of CpG locations such that two consecutive locations in the clsuter are separated by less than maxGap.

Value

a list, each item in the list is a character vector of CpG IDs located closely (i.e. in the same cluster)

Examples

```
CpGs_char <- c(
  "cg02505293", "cg03618257", "cg04421269", "cg17885402", "cg19890033",
  "cg20566587", "cg27505880"
)

cluster_ls <- CloseBySingleRegion(
  CpGs_char, arrayType = "450k", maxGap = 100, minCpGs = 3
)
```

`CoMethAllRegions` *Extract contiguous co-methylated genomic regions from a list of pre-defined genomic regions*

Description

Extract contiguous co-methylated genomic regions from a list of pre-defined genomic regions

Usage

```
CoMethAllRegions(betaMatrix, regionType = c("ISLAND", "NSHORE", "NSHELF",
  "SSHORE", "SSHELF", "TSS1500", "TSS200", "UTR5", "EXON1", "GENEBODY",
  "UTR3"), arrayType = c("450k", "EPIC"), file = NULL,
  fileType = c("gmt", "RDS"), rDropThresh_num = 0.4,
  returnAllCpGs = FALSE, ...)
```

Arguments

`betaMatrix`	matrix (or data frame) of beta values, with row names = CpG IDs, column names = sample IDs. This is typically genome-wide methylation beta values.
`regionType`	Type of input genomic regions (e.g. "ISLAND" for CpG island)
`arrayType`	Type of array, can be "450k" or "EPIC"
`file`	an RDS or gmt file with clusters of CpG locations (i.e. CpGs located closely to each other on the genome). This file can be generated by the `WriteCloseByAllRegions` function. If RDS file, it would contain a list, where each item is a character vector of CpGs IDs.
`fileType`	file extension for input file, can be "gmt" or "RDS"
`rDropThresh_num`	thershold for min correlation between a cpg with sum of the rest of the CpGs
`returnAllCpGs`	When there is not a contiguous comethylated region in the inputing pre-defined region, `returnAllCpGs = 1` indicates outputting all the CpGs in the input regions, while `returnAllCpGs = 0` indicates not returning any CpG.
`...`	Dots for internal arguments. Currently unused.

Value

A list of two components:

- Contiguous_Regions : A data frame with CpG (CpG name), Chr (chromosome number), MAPINFO (genomic position), r_drop(correlation between the CpG with rest of the CpGs), keep (indicator for co-methylated CpG), keep_contiguous (index for contiguous comethylated subregions)
- CpGsSubregions : results from all the regions, each item is a list of CpGs in the contiguous co-methylated subregion

Examples

```
data(betaMatrixChr22_df)

exampleFile <- readRDS(
                  system.file ("extdata",
                               "CpGislandsChr22_ex.RDS",
                               package = 'coMethDMR',
                               mustWork = TRUE
                               )
)


CoMethAllRegions (
  betaMatrix = betaMatrixChr22_df,
  file = exampleFile,
  fileType = "RDS",
  arrayType = "450k",
  returnAllCpGs = FALSE
)


## Not run:

CoMethAllRegions (
     betaMatrix = betaMatrixChr22_df,
     regionType = "ISLAND",
     arrayType = "450k",
     returnAllCpGs = FALSE
   )


## End(Not run)
```

CoMethSingleRegion *Wrapper function to find contiguous and comethyalted sub-regions within a pre-defined genomic region*

Description

Wrapper function to find contiguous and comethyalted sub-regions within a pre-defined genomic region

Usage

```
CoMethSingleRegion(CpGs_char, betaMatrix, rDropThresh_num = 0.4,
  minCpGs = 3, arrayType = c("450k", "EPIC"), returnAllCpGs = FALSE)
```

Arguments

CpGs_char	vector of CpGs in the inputting pre-defined genomic region.
betaMatrix	matrix (or data frame) of beta values, with row names = CpG ids, column names = sample ids. This should include the CpGs in CpGs_char, as well as additional CpGs.
rDropThresh_num	thershold for min correlation between a cpg with sum of the rest of the CpGs
minCpGs	mininum number of CpGs to be considered a "region". Only regions with more than minCpGs will be returned.
arrayType	Type of array, can be "450k" or "EPIC"
returnAllCpGs	When there is not a contiguous comethylated region in the inputing pre-defined region, returnAllCpGs = 1 indicates outputting all the CpGs in the input region, while returnAllCpGs = 0 indicates not returning any CpG.

Value

A list with two components:

- Contiguous_Regions : a data frame with CpG (CpG ID), Chr (chromosome number), MAPINFO (genomic position), r_drop (correlation between the CpG with rest of the CpGs), keep (indicator for co-methylated CpG), keep_contiguous (index for contiguous comethylated subregion)
- CpGs_subregions : lists of CpGs in each contiguous co-methylated subregion

Examples

```
data(betaMatrixChr22_df)
CpGsChr22_char <- c(
  "cg02953382", "cg12419862", "cg24565820", "cg04234412", "cg04824771",
  "cg09033563", "cg10150615", "cg18538332", "cg20007245", "cg23131131",
  "cg25703541"
)
CoMethSingleRegion(
  CpGs_char = CpGsChr22_char,
  betaMatrix = betaMatrixChr22_df
)

data(betaCluster_mat_example3)
betaMatrix_ex3 <- t(betaMatrix_ex3)
CpGsEx3_char <- c(
  "cg14221598", "cg02433884", "cg07372974", "cg13419809", "cg26856676",
  "cg25246745"
)
CoMethSingleRegion(
  CpGs_char = CpGsEx3_char,
  betaMatrix = betaMatrix_ex3,
  returnAllCpGs = TRUE
)
```

CpGsInfoAllRegions *Test associations of individual CpGs in multiple genomic regions with a continuous phenotype*

Description

Test associations of individual CpGs in multiple genomic regions with a continuous phenotype

Usage

```
CpGsInfoAllRegions(AllRegionNames_char, betas_df, pheno_df, contPheno_char,
  covariates_char, arrayType = c("450k", "EPIC"))
```

Arguments

AllRegionNames_char	vector of character strings with location info for all the genomic regions. Each region should be specified in this format: "chrxx:xxxxxx-xxxxxx"
betas_df	data frame of beta values for all genomic regions, with row names = CpG IDs, column names = sample IDs
pheno_df	a data frame with phenotype and covariate variables, with variable "Sample" for sample IDs.
contPheno_char	character string of the continuous phenotype, to be tested against methylation values
covariates_char	character vector of covariate variables names
arrayType	Type of array, can be "450k" or "EPIC"

Value

a data frame with locations of the genomic region (Region), CpG ID (cpg), chromosome (chr), position (pos), and results for testing association of methylation in individual CpGs with continuous phenotype (slopeEstimate, slopePval)

Examples

```
data(betaMatrixChr22_df)
data(pheno_df)
AllRegionNames_char <- c("chr22:18267969-18268249", "chr22:18531243-18531447")

CpGsInfoAllRegions(
  AllRegionNames_char,
  betas_df = betaMatrixChr22_df,
  pheno_df, contPheno_char = "stage",
  covariates_char = c("age.brain", "sex")
)
```

CpGsInfoOneRegion *Test associations of individual CpGs in a genomic region with a continuous phenotype*

Description

Test associations of individual CpGs in a genomic region with a continuous phenotype

Usage

```
CpGsInfoOneRegion(regionName_char, betas_df, pheno_df, contPheno_char,
  covariates_char, arrayType = c("450k", "EPIC"))
```

Arguments

regionName_char	character string of location information for a genomic region, specified in the format of "chrxx:xxxxxx-xxxxxx"
betas_df	data frame of beta values with row names = CpG IDs, column names = sample IDs
pheno_df	a data frame with phenotype and covariate variables, with variable "Sample" for sample IDs.
contPheno_char	character string of the continuous phenotype, to be tested against methylation values
covariates_char	character vector of covariate variables names
arrayType	Type of array, can be "450k" or "EPIC"

Details

This function implements linear models that test association between methylation values in a genomic region with a continuous phenotype. Note that methylation M values are used as regression outcomes in these models. The model for each CpG is:

```
methylation M value ~ contPheno_char + covariates_char
```

Value

a data frame with location of the genomic region (Region), CpG ID (cpg), chromosome (chr), position (pos), results for testing association of methylation in individual CpGs with continuous phenotype (slopeEstimate, slopePval) and annotations for the regions

Examples

```
data(betaMatrixChr22_df)
data(pheno_df)

CpGsInfoOneRegion(
  regionName_char = "chr22:19709548-19709755",
  betas_df = betaMatrixChr22_df,
  pheno_df, contPheno_char = "stage",
  covariates_char = c("age.brain", "sex"),
```

```
  arrayType = "450k"
)

# not adjusting for covariates
CpGsInfoOneRegion(
  regionName_char = "chr22:18267969-18268249",
  betas_df = betaMatrixChr22_df,
  pheno_df, contPheno_char = "stage",
  covariates_char = NULL
)
```

`FindComethylatedRegions` *Find contiguous comethylated regions based on output file from function* `MarkComethylatedCpGs`

Description

Find contiguous comethylated regions based on output file from function `MarkComethylatedCpGs`

Usage

```
FindComethylatedRegions(CpGs_df, minCpGs_int = 3)
```

Arguments

`CpGs_df`	an output dataframe from function `MarkComethylatedCpGs`, with variables `CpG, keep, ind, r_drop`. See details in documentation for `MarkComethylatedCpGs`.
`minCpGs_int`	an integer, indicates minimum nubmer of CpGs for output genomic regions

Value

A data frame with variables `ProbeID` and `Subregion` (index for each output contiguous comethylated regions)

Examples

```
data(betaMatrix_ex4)

CpGs_df <- MarkComethylatedCpGs(betaCluster_mat = betaMatrix_ex4)

FindComethylatedRegions(CpGs_df)
```

GetCpGsInRegion *Extract probe IDs for CpGs located in a genomic region*

Description

Extract probe IDs for CpGs located in a genomic region

Usage

```
GetCpGsInRegion(regionName_char, arrayType = c("450k", "EPIC"))
```

Arguments

regionName_char	character string with location information for one region in this format: "chrxx:xxxxxx-xxxxxx"
arrayType	Type of array, 450k or EPIC

Value

vector of CpG probe IDs mapped to the genomic region

Examples

```
GetCpGsInRegion(
  regionName_char = "chr22:18267969-18268249",
  arrayType = "450k"
)
```

lmmTest *Fit mixed model to methylation values in one genomic region*

Description

Fit mixed model to methylation values in one genomic region

Usage

```
lmmTest(betaOne_df, pheno_df, contPheno_char, covariates_char,
  modelType = c("randCoef", "simple"), arrayType = c("450k", "EPIC"))
```

Arguments

betaOne_df	matrix of beta values for one genomic region, with row names = CpG IDs, column names = sample IDs
pheno_df	a data frame with phenotype and covariates, with variable Sample indicating sample IDs.
contPheno_char	character string of the main effect (a continuous phenotype) to be tested for association with methylation values in the region

`covariates_char`
: character vector for names of the covariate variables

`modelType`
: type of mixed model, can be `randCoef` for random coefficient mixed model, or `simple` for simple linear mixed model.

`arrayType`
: Type of array, can be "450k" or "EPIC"

Details

This function implements a mixed model to test association between methylation values in a genomic region with a continuous phenotype.

When `randCoef` is selected, the model is

`methylation M value ~ contPheno_char + covariates_char + (1|Sample) + (contPheno_char|CpG)`. The last two terms are random intercepts and slopes for each CpG.

When `simple` is selected, the model is

`methylation M value ~ contPheno_char + covariates_char + (1|Sample)`

In our simulation studies, we found both models are conservative, so p-values are estimated from normal distributions instead of t-distributions.

Value

A dataframe with one row for association result of one region: `Estimate`, `StdErr`, and `pvalue` for the association of methylation values in the genomic region tested vs. continuous phenotype `contPheno_char`

Examples

```
data(betaMatrixChr22_df)

CpGsChr22_char<-c("cg02953382", "cg12419862", "cg24565820", "cg04234412",
    "cg04824771", "cg09033563", "cg10150615", "cg18538332", "cg20007245",
    "cg23131131", "cg25703541")

coMethCpGs <- CoMethSingleRegion(CpGsChr22_char, betaMatrixChr22_df)

# test only the first co-methylated region
coMethBetaMatrix <- betaMatrixChr22_df[coMethCpGs$CpGsSubregions[[1]], ]

data(pheno_df)

res <- lmmTest (betaOne_df = coMethBetaMatrix,
         pheno_df,
         contPheno_char = "stage",
         covariates_char = c("age.brain", "sex"),
         modelType = "randCoef",
         arrayType = "450k")
```

`lmmTestAllRegions` *Fit mixed model to test association between a continuous phenotype and methylation values in a list of genomic regions*

Description

Fit mixed model to test association between a continuous phenotype and methylation values in a list of genomic regions

Usage

```
lmmTestAllRegions(beta_df, region_ls, pheno_df, contPheno_char,
  covariates_char, modelType = c("randCoef", "simple"),
  arrayType = c("450k", "EPIC"), outFile = NULL)
```

Arguments

`beta_df`	data frame of beta values for all genomic regions, with row names = CpG IDs, column names = sample IDs. This is often the genome-wide array data.
`region_ls`	a list of genomic regions, each item is a vector of CpG IDs within a genomic region. The co-methylated regions can be obtained by function `CoMethAllRegions`.
`pheno_df`	a data frame with phenotype and covariates, with variable `Sample` indicating sample IDs.
`contPheno_char`	character string of the main effect (a continuous phenotype) to be tested for association with methylation values in each region
`covariates_char`	character vector for names of the covariate variables
`modelType`	type of mixed model, can be `randCoef` for random coefficient mixed model, or `simple` for simple linear mixed model.
`arrayType`	Type of array, can be "450k" or "EPIC"
`outFile`	output .csv file with the results for the mixed model analysis

Value

csv file with location of the genomic region (`chrom, start, end`), number of CpGs (`nCpGs`), `Estimate`, Standard error (`StdErr`) of the test statistic, p-value and False Discovery Rate (FDR) for association between methylation values in each genomic region with phenotype (`pValue`).

Examples

```
data(betaMatrixChr22_df)

data(pheno_df)

CpGisland_ls <- system.file(
  "extdata", "CpGislandsChr22_ex.RDS",
   package = 'coMethDMR', mustWork = TRUE
)

coMeth_ls <- CoMethAllRegions(
```

```
                betaMatrix = betaMatrixChr22_df,
                file = CpGisland_ls,
                fileType = "RDS",
                arrayType = "450k",
                rDropThresh_num = 0.4,
                returnAllCpGs = FALSE
            )

lmmTestAllRegions(
  beta_df = betaMatrixChr22_df,
  region_ls = coMeth_ls$CpGsSubregions,
  pheno_df,
  contPheno_char = "stage",
  covariates_char = "age.brain",
  modelType = "randCoef",
  arrayType = "450k"
)
```

`MarkComethylatedCpGs` *Mark CpGs in contiguous and co-methylated region*

Description

Mark CpGs in contiguous and co-methylated region

Usage

```
MarkComethylatedCpGs(betaCluster_mat, rDropThresh_num = 0.4)
```

Arguments

`betaCluster_mat`
: matrix of beta values, with rownames = sample ids, column names = CpG ids. Note that the CpGs need to be ordered by their genomic positions, this can be accomplished by the `OrderCpGbyLocation` function.

`rDropThresh_num`
: thershold for min correlation between a cpg with sum of the rest of the CpGs

Details

An outlier CpG in a genomic region will typically have low correlation with the rest of the CpGs in a genomic region. On the other hand, in a cluster of co-methylated CpGs, we expect each CpG to have high correlation with the rest of the CpGs. The `r.drop` statistic is used to identify these co-methylated CpGs here.

Value

A data frame with the following columns:

- `CpG` : CpG ID
- `keep` : The CpGs with `keep = 1` belong to the contiguous and co-methylated region
- `ind` : Index for the CpGs
- `r_drop` : The correlation between each CpG with the sum of the rest of the CpGs

Examples

```
data(betaMatrix_ex1)
MarkComethylatedCpGs(betaCluster_mat = betaMatrix_ex1)

data(betaMatrix_ex2)
MarkComethylatedCpGs(betaCluster_mat = betaMatrix_ex2)

data(betaMatrix_ex3)
MarkComethylatedCpGs(betaCluster_mat = betaMatrix_ex3)

data(betaMatrix_ex4)
MarkComethylatedCpGs(betaCluster_mat = betaMatrix_ex4, rDropThresh_num = 0.6)
```

NameRegion *Name a region with several CpGs based on its genomic location*

Description

Name a region with several CpGs based on its genomic location

Usage

```
NameRegion(CpGsOrdered_df)
```

Arguments

CpGsOrdered_df	dataframe with columns for Probe IDs as character (cpg), chromosome number as character (chr) and genomic location as integer (pos)

Value

genome location of the CpGs, in the format of "chrxx:xxxxxx-xxxxxx"

Examples

```
 CpGs_char <- c("cg04677227", "cg07146435", "cg11632906", "cg20214853")
CpGsOrdered_df <- OrderCpGsByLocation(CpGs_char, arrayType=c("EPIC"), output = "dataframe")
 NameRegion(CpGsOrdered_df)
```

OrderCpGsByLocation *Order CpGs by genomic location*

Description

Order CpGs by genomic location

Usage

```
OrderCpGsByLocation(CpGs_char, arrayType = c("450k", "EPIC"),
  output = c("vector", "dataframe"))
```

Arguments

CpGs_char	vector of CpGs
arrayType	Type of array, 450k or EPIC
output	vector of CpGs or dataframe with CpGs, CHR, MAPINFO

Value

vector of CpGs ordered by location or dataframe with CpGs ordered by location (cpg), chromosome (chr), position (pos)

Examples

```
CpGs_char <- c("cg04677227", "cg07146435", "cg11632906", "cg20214853")
OrderCpGsByLocation(CpGs_char, arrayType=c("EPIC"), output = "dataframe")
```

pheno_df *Example phenotype data file from Prefrontal Cortex (PFC) Methylation Data of Alzheimer's Disease subjects*

Description

Subset of phenotype information for Alzheimer's methylation dataset.

Usage

```
pheno_df
```

Format

A data frame containing variables for Braak stage (stage), subject.id, Batch (Mplate), Sex, Sample, age of brain sample (age.brain)

Source

GEO accession: GSE59685

RegionsToRanges *Convert genomic regions in a data frame to GRanges format*

Description

Convert genomic regions in a data frame to GRanges format

Usage

```
RegionsToRanges(regionName_char)
```

Arguments

regionName_char	a character vector of regions, in this format: "chrxx:xxxxxx-xxxxxx"

Value

genomic regions in GRanges format

Examples

```
regions = c("chr22:19709548-19709755", "chr2:241721922-241722113")
RegionsToRanges (regions)
```

WriteCloseByAllRegions *Extract clusters of close by CpGs from a list of pre-defined genomic regions*

Description

Extract clusters of close by CpGs from a list of pre-defined genomic regions

Usage

```
WriteCloseByAllRegions(file, regionType = c("ISLAND", "NSHORE", "NSHELF",
  "SSHORE", "SSHELF", "TSS1500", "TSS200", "UTR5", "EXON1", "GENEBODY",
  "UTR3"), arrayType = c("450k", "EPIC"), maxGap = 200, minCpGs = 3,
  fileType = c("gmt", "RDS"), ...)
```

Arguments

file	file where the output genomic regions will be saved. File extension should not be supplied, it is automatically added via the fileType argument.
regionType	Type of input genomic regions (e.g. "ISLAND" for CpG island)
arrayType	Type of array, can be "450k" or "EPIC"
maxGap	an integer, genomic locations within maxGap from each other are placed into the same cluster

minCpGs	an integer, minimum number of CpGs for each resulting region
fileType	the output files can be saved as .gmt or .RDS.
...	Dots for internal arguments. Currently unused.

Details

For maxGap = 200 and minCpGs = 3, we already calculated the clusters of CpGs. They are saved in folder /inst/extdata/.

Note that for output files, .gmt files can be opened as flat text file. .RDS files are half the size of .gmt files, but they can only be read in the R enviroment.

Creating and writing the file for one type of genomic region (regionType = "ISLAND") took about 25 minutes.

Value

a file with the genomic regions containing CpGs located closely within each inputing pre-defined genomic region

Examples

```
## Not run:
  CloseByAllRegions(
     regionType = "ISLAND", arrayType = "450k", maxGap = 50,
     minCpGs = 3, fileType = "gmt", file = "closeByRegions"
  )

## End(Not run)
```

Index