Package ‘EpiMix’

February 19, 2024

Title EpiMix: an integrative tool for the population-level analysis of DNA methylation

Version 1.4.0

Description
EpiMix is a comprehensive tool for the integrative analysis of high-throughput DNA methylation data and gene expression data. EpiMix enables automated data downloading (from TCGA or GEO), preprocessing, methylation modeling, interactive visualization and functional annotation. To identify hypo- or hypermethylated CpG sites across physiological or pathological conditions, EpiMix uses a beta mixture modeling to identify the methylation states of each CpG probe and compares the methylation of the experimental group to the control group. The output from EpiMix is the functional DNA methylation that is predictive of gene expression. EpiMix incorporates specialized algorithms to identify functional DNA methylation at various genetic elements, including proximal cis-regulatory elements of protein-coding genes, distal enhancers, and genes encoding microRNAs and IncRNAs.

Depends R (>= 4.2.0), EpiMix.data (>= 1.2.2)

License GPL-3

Encoding UTF-8

Imports AnnotationHub, AnnotationDbi, Biobase, biomaRt, data.table, doParallel, doSNOW, downloader, dplyr, ELMER.data, ExperimentHub, foreach, GenomeInfoDb, GenomicFeatures, GenomicRanges, ggplot2, graphics, grDevices, impute, IRanges, limma, methods, parallel, plyr, progress, R.matlab, RColorBrewer, RCurl, rlang, RPMM, S4Vectors, stats, SummarizedExperiment, tibble, tidyr, utils

Suggests BiocStyle, clusterProfiler, GEOquery, karyoploteR, knitr, org.Hs.eg.db, regioneR, Seurat, survival, survminer, TxDb.Hsapiens.UCSC.hg19.knownGene, RUnit, BiocGenerics, multiMiR, miRBaseConverter

biocViews Software, Epigenetics, Preprocessing, DNAMethylation, GeneExpression, DifferentialMethylation

RoxygenNote 7.2.3

VignetteBuilder knitr

BugReports https://github.com/gevaertlab/EpiMix/issues
R topics documented:

.git_url https://git.bioconductor.org/packages/EpiMix
.git_branch RELEASE_3_18
.git_last_commit dd3578d
.git_last_commit_date 2023-10-24
.Repository Bioconductor 3.18
.Date/Publication 2024-02-19
.Author Yuanning Zheng [aut, cre],
John Jun [aut],
Olivier Gevaert [aut]
.Maintainer Yuanning Zheng <eric2021@stanford.edu>

R topics documented:

.extractPriMiRNA ........................................ 4
.getComp .................................................. 4
.getMetGroup .............................................. 5
.mapProbeGene .............................................. 5
.splitMetData .............................................. 6
.addDistNearestTSS ...................................... 6
.addGeneNames ............................................. 7
.BatchCorrection_Combat ................................ 7
.BatchCorrection_Seurat ................................ 8
.betaEst_2 ............................................... 8
.blc_2 .................................................. 9
.calcDistNearestTSS ...................................... 9
.ClusterProbes ........................................... 10
.ComBat_NoFiles .......................................... 11
.combineForEachOutput ................................... 12
.convertAnnotToDF ....................................... 12
.convertGeneNames ....................................... 13
.CorrectBatchEffect .................................... 13
.EpiMix .................................................. 14
.EpiMix_getInfiniumAnnotation ......................... 18
.EpiMix_PlotGene ........................................ 19
.EpiMix_PlotModel ....................................... 21
.EpiMix_PlotProbe ....................................... 23
.EpiMix_PlotSurvival .................................... 25
.filterLinearProbes ..................................... 26
.filterMethMatrix ....................................... 27
.filterProbes ............................................ 28
.find_miRNA_targets ..................................... 29
.functionEnrich .......................................... 30
.generateFunctionalPairs ............................... 31
.GEO_Download_DNAMethylation ......................... 32
.GEO_Download_GeneExpression .......................... 33
The `.extractPriMiRNA` function

**Description**
Utility function to convert mature miRNA names to pri-miRNA names

**Usage**
`.extractPriMiRNA(str)`

**Arguments**
- `str` (a character string for a mature miRNA name (e.g. "hsa-miR-34a-3p"))

**Value**
a character string for the corresponding pri-miRNA name (e.g. "hsa-mir-34a")

The `.getComp` function

**Description**
Helper function to get a string indicating the comparison made for gene expression

**Usage**
`.getComp(state)`

**Arguments**
- `state` (character string indicating the methylation state, can be either "Hyper", "Hypo", "Dual")

**Value**
a list of sample names split by methylation group
.getMetGroup

The .getMetGroup function

Description

Helper function to get sample names split by methylation group based on DM values

Usage

.getMetGroup(state, DM_values)

Arguments

state character string indicating the methylation state, can be either "Hyper", "Hypo", "Dual"
DM_values a vector of DM values for the probe. The names of the vector are sample names.

Value

a list of sample names split by methylation group

.mapProbeGene

The .mapProbeGene function

Description

since in the original probe annotation, a specific probe can be mapped to multiple genes, this function splits the rows and maps each probe to a single gene in a row.

Usage

.mapProbeGene(df.annot)

Arguments

df.annot a dataframe with probe annotation, can be the object returned from the convertAnnotToDF function.

Value

a dataframe with 1:1 mapping of probe and gene
**.splitMetData**  
*The .splitMetData* function

**Description**  
Helper function to split the methylation data matrix into the experimental group and the control group.

**Usage**  
`.splitMetData(methylation.data, sample.info, group.1, group.2)`

**Arguments**

- `methylation.data`: methylation data matrix
- `sample.info`: sample information matrix
- `group.1`: name of group.1
- `group.2`: name of group.2

**Value**  
a list with methylation data of group.1 and group.2

**addDistNearestTSS**  
*Calculate the distance between probe and gene TSS*

**Description**  
Calculate the distance between probe and gene TSS.

**Usage**  
`addDistNearestTSS(data, NearGenes, genome, met.platform, cores = 1)`

**Arguments**

- `data`: A multi Assay Experiment with both DNA methylation and gene Expression objects
- `NearGenes`: A list or a data frame with the pairs gene probes
- `genome`: Which genome build will be used: hg38 (default) or hg19.
- `met.platform`: DNA methylation platform to retrieve data from: EPIC or 450K (default)
- `cores`: Number fo cores to be used. Deafult: 1

**Value**  
a dataframe of nearest genes with distance to TSS.
**addGeneNames**

*The addGeneNames function*

**Description**

Given a dataframe with a column of probe names, add the gene names.

**Usage**

```
addGeneNames(df_data, ProbeAnnotation)
```

**Arguments**

- `df_data`: a dataframe with a column named `Probe`
- `ProbeAnnotation`: a dataframe with `ProbeAnnotation`, including one column named 'probe' and another column named 'gene'

**Value**

a dataframe with added gene names

---

**BatchCorrection_Combat**

*The BatchCorrection_Combat function*

**Description**

The BatchCorrection_Combat function

**Usage**

```
BatchCorrection_Combat(GEN_Data, BatchDataSelected)
```

**Arguments**

- `GEN_Data`: matrix with methylation.data or gene.expression.data
- `BatchDataSelected`: BatchData after filtering out the small batches and selecting for overlapped samples

**Details**

correct batch effects with Combat

**Value**

corrected data matrix
BatchCorrection_Seurat

The BatchCorrection_Seurat function

Description

The BatchCorrection_Seurat function

Usage

BatchCorrection_Seurat(GEN_Data, BatchDataSelected)

Arguments

<table>
<thead>
<tr>
<th>GEN_Data</th>
<th>matrix with methylation.data or gene.expression.data</th>
</tr>
</thead>
<tbody>
<tr>
<td>BatchDataSelected</td>
<td>BatchData after filtering out the small batches and selecting for overlapped samples.</td>
</tr>
</tbody>
</table>

Details

Correct batch effects with the Seurat data integration functions.

Value

corrected data matrix

betaEst_2

The betaEst_2 function

Description

Internal. Estimates a beta distribution via Maximum Likelihood. Adapted from RPMM package.

Usage

betaEst_2(Y, w, weights)

Arguments

| Y | data vector. |
| w | posterior weights. |
| weights | Case weights. |

Value

(a,b) parameters.
**blc_2**

*The blc_2 function*

**Description**

Internal. Fits a beta mixture model for any number of classes. Adapted from RPMM package.

**Usage**

```r
blc_2(Y, w, maxiter = 25, tol = 1e-06, weights = NULL, verbose = TRUE)
```

**Arguments**

- **Y**: Data matrix (n x j) on which to perform clustering.
- **w**: Initial weight matrix (n x k) representing classification.
- **maxiter**: Maximum number of EM iterations.
- **tol**: Convergence tolerance.
- **weights**: Case weights.
- **verbose**: Verbose output.

**Value**

A list of parameters representing mixture model fit, including posterior weights and log-likelihood.

---

**calcDistNearestTSS**

*Calculate distance from region to nearest TSS*

**Description**

Idea For a given region R linked to X genes G merge R with nearest TSS for G (multiple) this will increase nb of lines i.e R1 - G1 - TSS1 - DIST1 R1 - G1 - TSS2 - DIST2 To vectorize the code: make a granges from left and one from right and find distance collapse the results keeping min distance for equals values

**Usage**

```r
calcDistNearestTSS(links, TRange, tssAnnot)
```

**Arguments**

- **links**: Links to calculate the distance
- **TRange**: Genomic coordinates for Target region
- **tssAnnot**: TSS annotation
Description
This function uses the annotation for Illumina methylation arrays to map each probe to a gene. Then, for each gene, it clusters all its CpG sites using hierarchical clustering and Pearson correlation as distance and complete linkage. If data for normal samples is provided, only overlapping probes between cancer and normal samples are used. Probes with SNPs are removed. This function is prepared to run in parallel if the user registers a parallel structure, otherwise it runs sequentially. This function also cleans up the sample names, converting them to the 12 digit format.

Usage
ClusterProbes(MET_data, ProbeAnnotation, CorThreshold = 0.4)

Arguments
MET_data data matrix for methylation.
ProbeAnnotation GRange object for probe annotation.
CorThreshold correlation threshold for cutting the clusters.

Value
List with the clustered data sets and the mapping between probes and genes.
The ComBat_NoFiles function

Description
Internal. Performs batch correction.

Usage
ComBat_NoFiles(
  dat,
  saminfo,
  type = "txt",
  write = FALSE,
  covariates = "all",
  par.prior = FALSE,
  filter = FALSE,
  skip = 0,
  prior.plots = TRUE
)

Arguments
| dat       | currently supports two data file types 'txt' for a tab-delimited text file and 'csv' for an Excel .csv file (sometimes R handles the .csv file better, so use this if you have problems with a .txt file!). |
| saminfo   | if 'T' ComBat writes adjusted data to a file, and if 'F' and ComBat outputs the adjusted data matrix if 'F' (so assign it to an object! i.e. NewData <- ComBat('my expression.xls','Sample info file.txt', write=F)). |
| type      | 'covariates=all' will use all of the columns in your sample info file in the modeling (except array/sample name), if you only want use a some of the columns in your sample info file, specify these columns here as a vector (you must include the Batch column in this list). |
| write     | if 'T' uses the parametric adjustments, if 'F' uses the nonparametric adjustments--if you are unsure what to use, try the parametric adjustments (they run faster) and check the plots to see if these priors are reasonable. |
| covariates | 'filter=value' filters the genes with absent calls in > 1-value of the samples. The default here (as well as in dchip) is .8. Filter if you can as the EB adjustments work better after filtering. Filter must be numeric if your expression index file contains presence/absence calls (but you can set it >1 if you don’t want to filter any genes) and must be 'F' if your data doesn’t have presence/absence calls; |
| par.prior | is the number of columns that contain probe names and gene information, so 'skip=5' implies the first expression values are in column 6 |
prior.plots if true will give prior plots with black as a kernal estimate of the empirical batch effect density and red as the parametric estimate.

Value
Results.

combineForEachOutput The combineForEachOutput function

Description
Internal. Function to combine results from the foreach loop.

Usage
combineForEachOutput(out1, out2)

Arguments
out1 result from one foreach loop.
out2 result from another foreach loop.

Value
List with the combined results.

convertAnnotToDF The convertAnnotToDF function

Description
convert the probe annotation from the GRang eobject to a dataframe

Usage
convertAnnotToDF(annot)

Arguments
annot a GRanger object of probe annotation, can be the object returned from the get-InfiniumAnnotation function.

Value
a dataframe with chromosome, beginning and end position, mapped gene information for each CpG probe
**convertGeneNames**

The `convertGeneNames` function

**Description**

auxiliary function to translate ensembl_gene_ids or ensembl_transcript_ids to human gene symbols (HGNC)

**Usage**

`convertGeneNames(gene.expression.data)`

**Arguments**

- `gene.expression.data` gene expression data matrix with the rownames to be the ensembl_gene_ids or ensembl_transcript_ids

**Value**

gene expression matrix with rownames translated to human gene symbols (HGNC)

**CorrectBatchEffect**

The `CorrectBatchEffect` function

**Description**

top-level wrapper function for batch correction.

**Usage**

```
CorrectBatchEffect(
    GEN_Data, BatchData, batch.correction.method, MinInBatch = 5, featurePerSet = 50000)
```
Arguments

**GEN_Data** matrix with methylation.data or gene.expression.data with genes in rows and samples in columns

**BatchData** dataframe with two columns: the first column indicates the sample names, and the second column indicates the batch ids.

**batch.correction.method** character string. Should be either ‘Seurat’ or ‘Combat’.

**MinInBatch** integer indicating the batch size threshold. Batches smaller than this threshold will be removed. Default: 5

**featurePerSet** integer indicating the row numbers to split the GEN_Data into small subsets. Default: 50,000

Details

1. filters the batch data and the molecular data to keep only the overlapped samples.
2. removes extremely small batches.
3. if the molecular data have over 50,000 features (rows), it splits the data into subsets, with 50,000 features in each subset, and perform batch correction on each subset.
4. identify overlapped samples in batch corrected subsets, and merge the subsets into one matrix.

Value

matrix with corrected data

---

**EpiMix**

*The EpiMix function*

Description

EpiMix uses a model-based approach to identify functional changes DNA methylation that affect gene expression.

Usage

```r
EpiMix(
  methylation.data,
  gene.expression.data,
  sample.info,
  group.1,
  group.2,
  mode = "Regular",
  promoters = FALSE,
  correlation = "negative",
  met.platform = "HM450",
  genome = "hg38",
  cluster = FALSE,
  listOfGenes = NULL,
) ```
filter = TRUE,
raw.pvalue.threshold = 0.05,
adjusted.pvalue.threshold = 0.05,
numFlankingGenes = 20,
roadmap.epigenome.groups = NULL,
roadmap.epigenome.ids = NULL,
chromatin.states = c("EnhA1", "EnhA2", "EnhG1", "EnhG2"),
NoNormalMode = FALSE,
cores = 1,
MixtureModelResults = NULL,
OutputRoot = "."
)

Arguments

methylation.data
Matrix of the DNA methylation data with CpGs in rows and samples in columns.
gene.expression.data
Matrix of the gene expression data with genes in rows and samples in columns.
sample.info
Dataframe that maps each sample to a study group. Should contain two columns: the first column (named 'primary') indicates the sample names, and the second column (named 'sample.type') indicating which study group each sample belongs to (e.g., “Cancer” vs. “Normal”, “Experiment” vs. “Control”). Sample names in the 'primary' column must coincide with the column names of the methylation.data.
group.1
Character vector indicating the name(s) for the experiment group.
group.2
Character vector indicating the name(s) for the control group.
mode
Character string indicating the analytic mode to model DNA methylation. Should be one of the followings: 'Regular', 'Enhancer', 'miRNA' or 'lncRNA'. Default: 'Regular'. See details for more information.
promoters
Logic indicating whether to focus the analysis on CpGs associated with promoters (2000 bp upstream and 1000 bp downstream of the transcription start site). This parameter is only used for the Regular mode.
correlation
Character vector indicating the expected correlation between DNA methylation and gene expression. Can be either 'negative' or 'positive'. Default: 'negative'.
met.platform
Character string indicating the microarray type for collecting the DNA methylation data. The value should be either 'HM27', 'HM450' or 'EPIC'. Default: 'HM450'
genome
Character string indicating the genome build version to be used for CpG annotation. Should be either 'hg19' or 'hg38'. Default: 'hg38'.
cluster
Logic indicating whether to cluster CpG site based on methylation levels using hierarchical clustering
listOfGenes
Character vector used for filtering the genes to be evaluated.
filter
Logic indicating whether to use a linear regression filter to pre-filter the CpGs whose methylation correlates with gene expression. Used in the Regular mode. Default: TRUE.
raw.pvalue.threshold
Numeric value indicating the threshold of the raw P value for selecting the functional CpG-gene pairs. Default: 0.05.

adjusted.pvalue.threshold
Numeric value indicating the threshold of the adjusted P value for selecting the functional CpG-gene pairs. Default: 0.05.

numFlankingGenes
Numeric value indicating the number of flanking genes whose expression is to be evaluated for selecting the functional enhancers. Default: 20.

roadmap.epigenome.groups
(parameter used for the 'Enhancer' mode) Character vector indicating the tissue group(s) to be used for selecting the enhancers. See details for more information. Default: NULL.

roadmap.epigenome.ids
(parameter used for the 'Enhancer' mode) Character vector indicating the epigenome ID(s) to be used for selecting the enhancers. See details for more information. Default: NULL.

chromatin.states
(parameter used for the 'Enhancer' mode) Character vector indicating the chromatin states to be used for selecting the enhancers. To get the available chromatin states, please run the list.chromatin.states() function. Default: c('EnhA1', 'EnhA2', 'EnhG1', 'EnhG2').

NoNormalMode
Logical indicating if the methylation states found in the experiment group should be compared to the control group. Default: FALSE.

cores
Number of CPU cores to be used for computation. Default: 1.

MixtureModelResults
Pre-computed EpiMix results, used for generating functional probe-gene pair matrix. Default: NULL

OutputRoot
File path to store the EpiMix result object. Default: '.' (current directory)

Details
mode: EpiMix incorporates four alternative analytic modes for modeling DNA methylation: “Regular,” “Enhancer”, “miRNA” and “lncRNA”. The four analytic modes target DNA methylation analysis on different genetic elements. The Regular mode aims to model DNA methylation at proximal cis-regulatory elements of protein-coding genes. The Enhancer mode targets DNA methylation analysis on distal enhancers. The miRNA or lncRNA mode focuses on methylation analysis of miRNA- or lncRNA-coding genes.

roadmap.epigenome.groups & roadmap.epigenome.ids:
Since enhancers are cell-type or tissue-type specific, EpiMix needs to know the reference tissues or cell types in order to select the proper enhancers. EpiMix identifies enhancers from the RoadmapEpigenomic project (Nature, PMID: 25693563), which enhancers were identified by ChromHMM in over 100 tissue and cell types. Available epigenome groups (a group of relevant cell types) or epigenome ids (individual cell types) can be obtained from the original publication (Nature, PMID: 25693563, figure 2). They can also be retrieved from the list.epigenomes() function. If both roadmap.epigenome.groups and roadmap.epigenome.ids are specified, EpiMix will select all the epigenomes from the combination of the inputs.
Value

The results from EpiMix is a list with the following components:

- **MethylationDrivers**: CpG probes identified as differentially methylated by EpiMix.
- **NrComponents**: The number of methylation states found for each driver probe.
- **MixtureStates**: A list with the DM-values for each driver probe. Differential Methylation values (DM-values) are defined as the difference between the methylation mean of samples in one mixture component from the experiment group and the methylation mean in samples from the control group, for a given probe.
- **MethylationStates**: Matrix with DM-values for all driver probes (rows) and all samples (columns).
- **Classifications**: Matrix with integers indicating to which mixture component each sample in the experiment group was assigned to, for each probe.
- **Models**: Beta mixture model parameters for each driver probe.
- **group.1**: sample names in group.1 (experimental group).
- **group.2**: sample names in group.2 (control group).
- **FunctionalPairs**: Dataframe with the prevalence of differential methylation for each CpG probe in the sample population, and fold change of mRNA expression and P values for each significant probe-gene pair.

Examples

```r
# Example #1: Regular mode
EpiMixResults <- EpiMix(methylation.data = MET.data,
            gene.expression.data = mRNA.data,
            sample.info = LUAD.sample.annotation,
            group.1 = 'Cancer',
            group.2 = 'Normal',
            met.platform = 'HM450',
            OutputRoot = tempdir())

# Example #2: Enhancer mode
EpiMixResults <- EpiMix(methylation.data = MET.data,
            gene.expression.data = mRNA.data,
            sample.info = LUAD.sample.annotation,
            mode = 'Enhancer',
            group.1 = 'Cancer',
            group.2 = 'Normal',
            met.platform = 'HM450',
            OutputRoot = tempdir())
```
EpiMix_getInfiniumAnnotation

The EpiMix_getInfiniumAnnotation function

Description
fetch the Infinium probe annotation from the seasameData library

Usage
EpiMix_getInfiniumAnnotation(plat = "EPIC", genome = "hg38")

Arguments
- plat character string indicating the methylation platform
- genome character string indicating the version of genome build

Value
a GRanges object of probe annotation
**Description**

plot the genomic coordinate, DM values and chromatin state for each CpG probe of a specific gene.

**Usage**

```r
EpiMix_PlotGene(
  gene.name,
  EpiMixResults,
  met.platform = "HM450",
  roadmap.epigenome.id = "E002",
  left.gene.margin = 10000,
  right.gene.margin = 10000,
  gene.name.font = 0.7,
  show.probe.name = TRUE,
  probe.name.font = 0.6,
  plot.transcripts = TRUE,
  plot.transcripts.structure = TRUE,
  y.label.font = 0.8,
  y.label.margin = 0.1,
  axis.number.font = 0.5,
  chromatin.label.font = 0.7,
  chromatin.label.margin = 0.02
)
```

**Arguments**

- **gene.name**: character string indicating the name of the gene to be plotted.
- **EpiMixResults**: the resulting list object returned from the function of EpiMix.
- **met.platform**: character string indicating the type of the microarray where the DNA methylation data were collected. The value should be either 'HM27', 'HM450' or 'EPIC'. Default: 'HM450'
- **roadmap.epigenome.id**: character string indicating the epigenome id (EID) for a reference tissue or cell type. Default: 'E002'. If the value is empty ('"'), no histone modifications plot will show. Note: Keep this value empty if using the Windows system, since this feature is not supported in Windows.
- **left.gene.margin**: numeric value indicating the number of extra nucleotide bases to be plotted on the left side of the target gene. Default: 10000.
- **right.gene.margin**: numeric value indicating the number of extra nucleotide bases to be plotted on the right side of the target gene. Default: 10000.
EpiMix_PlotGene

gene.name.font  numeric value indicating the font size for the gene name. Default: 0.7.
show.probe.name  logic indicating whether to show the name(s) for each differentially methylated Cpg probe. Default: TRUE
probe.name.font  numeric value indicating the font size of the name(s) for the differentially methylated probe(s) in pixels. Default: 0.6.
plot.transcripts  logic indicating whether to plot each individual transcript of the gene. Default: TRUE. If False, the gene will be plotted with a single rectangle, without showing the structure of individual transcripts.
plot.transcripts.structure  logic indicating whether to plot the transcript structure (introns and exons). Non-coding exons are shown in green and the coding exons are shown in red. Default: TRUE.
y.label.font  font size of the y axis label
y.label.margin  distance between y axis label and y axis
axis.number.font  font size of axis ticks and numbers
chromatin.label.font  font size of the labels of the histone proteins
chromatin.label.margin  distance between the histone protein labels and axis

Details

this function requires R package dependencies: karyoploteR, TxDb.Hsapiens.UCSC.hg19.knownGene, org.Hs.eg.db
roadmap.epigenome.id: since the chromatin state is tissue or cell-type specific, EpiMix needs to know the reference tissue or cell type in order to retrieve the proper DNase-seq and histone ChIP-seq data. Available epigenome ids can be obtained from the Roadmap Epigenomic study (Nature, PMID: 25693563, figure 2). They can also be retrieved from the list.epigenomes() function.

Value

plot of the genomic coordinate, DM values and chromatin state for each CpG probe of a specific gene.

Examples

library(karyoploteR)
library(TxDB.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
library(regioneR)
data(Sample_EpiMixResults.Regular)
gene.name = 'CCND2'
roadmap.epigenome.id = 'E096'

EpiMix_PlotGene(gene.name = gene.name,
    EpiMixResults = Sample_EpiMixResults-Regular,
    met.platform = 'HM450',
    roadmap.epigenome.id = roadmap.epigenome.id)

EpiMix_PlotModel

The EpiMix_PlotModel function.

Description

Produce the mixture model and the gene expression plots representing the EpiMix results.

Usage

EpiMix_PlotModel(
    EpiMixResults,
    Probe,
    methylation.data,
    gene.expression.data = NULL,
    GeneName = NULL,
    axis.title.font = 20,
    axis.text.font = 16,
    legend.title.font = 18,
    legend.text.font = 18,
    plot.title.font = 20
)

Arguments

EpiMixResults resulting list object from the EpiMix function.
Probe character string indicating the name of the CpG probe for which to create a mixture model plot.
methylation.data Matrix with the methylation data with genes in rows and samples in columns.
gene.expression.data Gene expression data with genes in rows and samples in columns (optional). Default: NULL.
GeneName character string indicating the name of the gene whose expression will be plotted with the EpiMix plot (optional). Default: NULL.
axis.title.font font size for the axis legend.
axis.text.font  font size for the axis label.
legend.title.font
  font size for the legend title.
legend.text.font
  font size for the legend label.
plot.title.font
  font size for the plot title.

Details

The violin plot and the scatter plot will be NULL if the gene expression data or the GeneName is not provided

Value

A list of EpiMix plots:

MixtureModelPlot
  a histogram of the distribution of DNA methylation data
ViolinPlot
  a violin plot of gene expression levels in different mixtures in the MixtureModelPlot
CorrelationPlot
  a scatter plot between DNA methylation and gene expression

Examples

{
  data(MET.data)
  data(mRNA.data)
  data(Sample_EpiMixResults-Regular)

  probe = "cg14029001"
  gene.name = "CCND3"
  plots <- EpiMix_PlotModel(
    EpiMixResults = Sample_EpiMixResults-Regular,
    Probe = probe,
    methylation.data = MET.data,
    gene.expression.data = mRNA.data,
    GeneName = gene.name
  )

  plots$MixtureModelPlot
  plots$ViolinPlot
  plots$CorreilationPlot
}
**EpiMix_PlotProbe**

The `EpiMix_PlotProbe` function

Description

plot the genomic coordinate and the chromatin state of a specific CpG probe and the nearby genes.

Usage

```r
EpiMix_PlotProbe(
  probe.name, 
  EpiMixResults, 
  met.platform = "HM450", 
  roadmap.epigenome.id = "E002", 
  numFlankingGenes = 20, 
  left.gene.margin = 10000, 
  right.gene.margin = 10000, 
  gene.name.pos = 2, 
  gene.name.size = 0.5, 
  gene.arrow.length = 0.05, 
  gene.line.width = 2, 
  plot.chromatin.state = TRUE, 
  y.label.font = 0.8, 
  y.label.margin = 0.1, 
  axis.number.font = 0.5, 
  chromatin.label.font = 0.7, 
  chromatin.label.margin = 0.02
)
```

Arguments

- **probe.name**: character string indicating the CpG probe name.
- **EpiMixResults**: resulting list object returned from EpiMix.
- **met.platform**: character string indicating the type of micro-array where the DNA methylation data were collected. Can be either 'HM27', 'HM450' or 'EPIC'. Default: 'HM450'
- **roadmap.epigenome.id**: character string indicating the epigenome id (EID) for a reference tissue or cell type. Default: 'E002'. If the value is empty ('"'), no histone modifications plot will show. Note: Keep this value empty if using the Windows system, since this feature is not supported in Windows.
- **numFlankingGenes**: numeric value indicating the number of flanking genes to be plotted with the CpG probe. Default: 20 (10 gene upstream and 10 gene downstream).
- **left.gene.margin**: numeric value indicating the number of extra nucleotide bases to be plotted on the left side of the image. Default: 10000.
right.gene.margin
numeric value indicating the number of extra nucleotide bases to be plotted on the right side of the image. Default: 10000.
gene.name.pos
integer indicating the position for plotting the gene name relative to the gene structure. Should be 1 or 2 or 3 or 4, indicating bottom, left, top, and right, respectively.
gene.name.size
numeric value indicating the font size of the gene names in pixels.
gene.arrow.length
numeric value indicating the size of the arrow which indicates the positioning of the gene.
gene.line.width
numeric value indicating the line width for the genes.
plot.chromatin.state
logical indicating whether to plot the DNase-seq and histone ChIP-seq signals. Warnings: If the 'numFlankingGenes' is a larger than 15, plotting the chromatin state may flood the internal memory.
y.label.font
font size of the y axis label.
y.label.margin
distance between y axis label and y axis.
axis.number.font
font size of axis ticks and numbers.
chromatin.label.font
font size of the labels of the histone proteins.
chromatin.label.margin
distance between the histone protein labels and axis.

Details
this function requires additional dependencies: karyplotR, TxDb.Hsapiens.UCSC.hg19.knownGene, org.Hs.eg.db
roadmap.epigenome.id: since the chromatin state is tissue or cell-type specific, EpiMix needs to know the reference tissue or cell type in order to retrieve the proper DNase-seq and histone ChIP-seq data. Available epigenome ids can be obtained from the Roadmap Epigenomic study (Nature, PMID: 25693563, figure 2). They can also be retrieved from the list.epigenomes() function.

Value
plot with CpG probe and nearby genes. Genes whose expression is significantly negatively associated with the methylation of the probe are shown in red, while the others are shown in black.

Examples
library(karyplotR)
library(TxDB.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
library(regioneR)
data(Sample_EpiMixResults_Regula)
EpiMix_PlotSurvival

# The CpG site to plot
probe.name = 'cg00374492'

# The number of adjacent genes to be plotted
numFlankingGenes = 10

# Set up the reference cell/tissue type
roadmap.epigenome.id = 'E096'

# Generate the plot
EpiMix_PlotProbe(probe.name = probe.name,
                 EpiMixResults = Sample_EpiMixResults_Regular,
                 met.platform = 'HM450',
                 roadmap.epigenome.id = roadmap.epigenome.id,
                 numFlankingGenes = numFlankingGenes)

EpiMix_PlotSurvival  

EpiMix_PlotSurvival function

Description

function to plot Kaplan-meier survival curves for patients with different methylation state of a specific probe.

Usage

EpiMix_PlotSurvival(
    EpiMixResults,
    plot.probe,
    TCGA_CancerSite = NULL,
    clinical.df = NULL,
    font.legend = 16,
    font.x = 16,
    font.y = 16,
    font.tickslab = 14,
    legend = c(0.8, 0.9),
    show.p.value = TRUE
)

Arguments

EpiMixResults  List of objects returned from the EpiMix function
plot.probe    Character string with the name of the probe
TCGA_CancerSite  TCGA cancer code (e.g. 'LUAD')
filterLinearProbes

**Description**

use a linear regression filter to screen for probes that were negatively associated with gene expression.

**Usage**

```r
filterLinearProbes(
    methylation.data,
    gene.expression.data,
    ProbeAnnotation,
    cores,
    clinical.df
)
```

(If the TCGA_CancerSite parameter has been specified, this parameter is optional) Dataframe with survival information. Must contain at least three columns: 'sample.id', 'days_to_death', 'days_to_last_follow_up'.

**Value**

Kaplan-meier survival curve showing the survival time for patients with different methylation states of the probe.

**Examples**

```r
library(survival)
library(survminer)
data(Sample_EpiMixResults_miRNA)

EpiMix_PlotSurvival(EpiMixResults = Sample_EpiMixResults_miRNA, 
plot.probe = 'cg00909706', 
TCGA_CancerSite = 'LUAD')
```
filterMethMatrix

filter, cluster, correlation = "negative"
)

Arguments
methylation.data
methylation data matrix.
gene.expression.data
gene expression data matrix.
ProbeAnnotation
dataframe of probe annotation
cores
number of CPU cores used for computation
filter
logical indicating whether to perform a linear regression to select functional probes
cluster
logical indicating whether the CpGs were clustered using hierarchical clustering
correlation
Character vector indicating the expected correlation between DNA methylation and gene expression. Can be either 'negative' or 'positive'. Default: 'negative'.

Value
a character vector of probe names.

Description
The filterMethMatrix function

Usage
filterMethMatrix(MET_matrix, control.names, gene.expression.data)

Arguments
MET_matrix
a matrix of methylation states from the EpiMix results
control.names
a character vector of control sample names
gene.expression.data
a matrix with gene expression data

Details
This function filters methylation states from the beta mixture modeling for each probe. The filtered probes can be used to model gene expression by Wilcoxon test.
Value

A matrix of methylation states for each differentially methylated probe with probes in rows and patient in columns.

Description

Filter CpG sites based on user-specified conditions.

Usage

```
filterProbes(
  mode,
  gene.expression.data,
  listOfGenes,
  promoters,
  met.platform,
  genome
)
```

Arguments

- **mode**: analytic mode
- **gene.expression.data**: matrix of gene expression data
- **listOfGenes**: list of genes of interest
- **promoters**: logic indicating whether to filter CpGs on promoters
- **met.platform**: methylation platform
- **genome**: genome build version

Value

Filtered ProbeAnnotation
The `find_miRNA_targets` function

**Description**

Detection potential target protein-coding genes for the differentially methylated miRNAs using messenger RNA expression data.

**Usage**

```r
find_miRNA_targets(
  EpiMixResults,
  geneExprData,
  database = "mirtarbase",
  raw.pvalue.threshold = 0.05,
  adjusted.pvalue.threshold = 0.2,
  cores = 1
)
```

**Arguments**

- `EpiMixResults`: List of the result objects returned from the EpiMix function.
- `geneExprData`: Matrix of the messenger RNA expression data with genes in rows and samples in columns.
- `database`: Character string indicating the database for retrieving miRNA targets. Default: "mirtarbase".
- `raw.pvalue.threshold`: Numeric value indicating the threshold of the raw P value for selecting the miRNA targets based on gene expression. Default: 0.05.
- `adjusted.pvalue.threshold`: Numeric value indicating the threshold of the adjusted P value for selecting the miRNA targets based on gene expression. Default: 0.2.
- `cores`: Number of CPU cores to be used for computation. Default: 1.

**Value**

Matrix indicating the miRNA-target pairs, with fold changes of target gene expression and P values.

**Examples**

```r
library(multiMiR)
library(miRBaseConverter)

data(mRNA.data)
data(Sample_EpiMixResults_miRNA)

miRNA_targets <- find_miRNA_targets(
  # Add arguments as needed
)```
functionEnrich

The *functionEnrich* function

Description

Perform functional enrichment analysis for the differentially methylated genes occurring in the significant CpG-gene pairs.

Usage

```r
functionEnrich(
  EpiMixResults,
  methylation.state = "all",
  enrich.method = "GO",
  ont = "BP",
  simplify = TRUE,
  cutoff = 0.7,
  pvalueCutoff = 0.05,
  pAdjustMethod = "BH",
  qvalueCutoff = 0.2,
  save.dir = "."
)
```

Arguments

- **EpiMixResults**: List of the result objects returned from the EpiMix function.
- **methylation.state**: character string indicating whether to use all the differentially methylated genes or only use the hypo- or hyper-methylated genes for enrichment analysis. Can be either 'all', 'Hyper' or 'Hypo'.
- **enrich.method**: character string indicating the method to perform enrichment analysis, can be either 'GO' or 'KEGG'.
- **ont**: character string indicating the aspect for GO analysis. Can be one of 'BP' (i.e., biological process), 'MF' (i.e., molecular function), and 'CC' (i.e., cellular component) subontologies, or 'ALL' for all three.
- **simplify**: boolean value indicating whether to remove redundancy of enriched GO terms.
- **cutoff**: if simplify is TRUE, this is the threshold for similarity cutoff of the adjusted p value.
- **pvalueCutoff**: adjusted pvalue cutoff on enrichment tests to report
The generateFunctionalPairs function

Description

Wrapper function to get functional CpG-gene pairs, used for Regular, miRNA and LncRNA modes

Usage

```r
generateFunctionalPairs(
  MET_matrix,
  control.names,
  gene.expression.data,
  ProbeAnnotation,
  raw.pvalue.threshold,
  adjusted.pvalue.threshold,
  cores,
  mode = "Regular",
  correlation = "negative"
)
```
**GEO_Download_DNAMethylation**

**Arguments**

- **MET_matrix**
  matrix of methylation states
- **control.names**
  character vector indicating the samples names in the control group
- **gene.expression.data**
  matrix of gene expression data
- **ProbeAnnotation**
  dataframe of probe annotation
- **raw.pvalue.threshold**
  raw p value threshold
- **adjusted.pvalue.threshold**
  adjusted p value threshold
- **cores**
  number of computational cores
- **mode**
  character string indicating the analytic mode
- **correlation**
  the expected relationship between DNAme and gene expression

**Value**

a dataframe of functional CpG-gene matrix

---

**GEO_Download_DNAMethylation**

*The GEO_Download_DNAMethylation function*

**Description**

Download the methylation data and the associated sample phenotypic data from the GEO database.

**Usage**

```r
GEO_Download_DNAMethylation(
  AccessionID,
  targetDirectory = ".",
  DownloadData = TRUE
)
```

**Arguments**

- **AccessionID**
  character string indicating GEO accession number. Currently support the GEO series (GSE) data type.
- **targetDirectory**
  character string indicating the path to save the data. Default: `"."` (current directory).
- **DownloadData**
  logical indicating whether the actual data should be downloaded (Default: `TRUE`). If False, the desired directory where the downloaded data should have been saved is returned.
Value

a list with two elements. The first element (‘$MethylationData’) indicating the file path to the downloaded methylation data. The second element (‘$PhenotypicData’) indicating the file path to the sample phenotypic data.

Examples

```r
METdirectories <- GEO_Download_DNAMethylation(AccessionID = 'GSE114134',
                                               targetDirectory = tempdir())
```

---

**GEO_Download_GeneExpression**

*The GEO_Download_GeneExpression function*

---

**Description**

Download the gene expression data and the associated sample phenotypic data from the GEO database.

**Usage**

```r
GEO_Download_GeneExpression(
  AccessionID,
  targetDirectory = '.',
  DownloadData = TRUE
)
```

**Arguments**

- **AccessionID**
  character string indicating the GEO accession number. Currently support the GEO series (GSE) data type.
- **targetDirectory**
  character string indicating the file path to save the data. Default: ‘.’ (current directory)
- **DownloadData**
  logical indicating whether the actual data should be downloaded (Default: TRUE). If False, the desired directory where the downloaded data should have been saved is returned.

**Value**

a list with two elements. The first element (‘$GeneExpressionData’) indicating the file path to the downloaded methylation data. The second element (‘$PhenotypicData’) indicating the file path to the sample phenotypic data.
Examples

GEdirectories <- GEO_Download_GeneExpression(AccessionID = 'GSE114065',
                                            targetDirectory = tempdir())

GEO_EstimateMissingValues_Methylation

The GEO_EstimateMissingValues_Methylation function

Description

Internal. Removes samples and probes with more missing values than the MissingValueThreshold, and imputes remaining missing values using Tibshirani’s KNN method.

Usage

GEO_EstimateMissingValues_Methylation(
    MET_Data,
    MissingValueThresholdGene = 0.3,
    MissingValueThresholdSample = 0.3
)

Arguments

MET_Data methylation data or gene expression data matrix.

MissingValueThresholdGene threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.

MissingValueThresholdSample threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.

Value

the dataset with imputed values and possibly some genes or samples deleted.
The GEO_EstimateMissingValues_Molecular function

**Description**

Internal. Removes samples and genes with more missing values than the MissingValueThreshold, and imputes remaining missing values using Tibshirani’s KNN method.

**Usage**

```r
gEO_EstimateMissingValues_Molecular(
  MET_Data,
  MissingValueThresholdGene = 0.3,
  MissingValueThresholdSample = 0.1
)
```

**Arguments**

- **MET_Data**  
  methylation data or gene expression data matrix.
- **MissingValueThresholdGene**  
  threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.
- **MissingValueThresholdSample**  
  threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.

**Value**

the dataset with imputed values and possibly some genes or samples deleted.

The GEO_GetSampleInfo function

**Description**

auxiliary function to generate a sample information dataframe that indicates which study group each sample belongs to.

**Usage**

```r
gEO_GetSampleInfo(METdirectories, group.column, targetDirectory = ".")
```
Arguments

METdirectories  list of the file paths to the downloaded DNA methylation data, which can be the output from the GEO_Download_DNAmethylation function.

group.column  character string indicating the column in the phenotypic data that defines the study group of each sample. The values in this column will be used to split the experiment and the control group.

targetDirectory  file path to save the output. Default: '.' (current directory)

Value

a dataframe with two columns: a 'primary' column indicating the actual sample names, a 'sample.type' column indicating the study group for each sample.

Description

auxiliary function to generate a sample map for DNA methylation data and gene expression data

Usage

GEO_getSampleMap(METdirectories, GEdirectories, targetDirectory = ".")

Arguments

METdirectories  list of the file paths to the downloaded DNA methylation datasets, which can be the output from the GEO_Download_DNAmethylation function.

GEdirectories  list of the file paths to the downloaded gene expression datasets, which can be the output from the GEO_Download_GeneExpression function.

targetDirectory  file path to save the output. Default: '.' (current directory)

Value

dataframe with three columns: $assay (character string indicating the type of the experiment, can be either 'DNA methylation' or 'Gene expression'), $primary(character string indicating the actual sample names), $colnames (character string indicating the actual column names for each samples in DNA methylation data and gene expression data)
get.chromosome

The get.chromosome function

Description

given a list of genes, get the chromosomes of these genes.

Usage

get.chromosome(genes, genome)

Arguments

genes character vector with the gene names

genome character string indicating the genome build version, can be either 'hg19' or 'hg38'

Value

a dataframe for the mapping between genes and their chromosomes.

get.prevalence

The get.prevalence function

Description

Helper function to get the methylation state and the prevalence of the differential methylation of a CpG sites in the study population.

Usage

get.prevalence(MethylMixResults)

Arguments

METH_matrix matrix of methylation states

Value

a list of prevalence for the abnormal methylation
getFeatureProbe

to select probes within promoter regions or distal regions.

Description

getFeatureProbe is a function to select the probes falling into distal feature regions or promoter regions.
This function selects the probes on HM450K that either overlap distal biofeatures or TSS promoter.

Usage

getFeatureProbe(
    feature = NULL,
    TSS,
    genome = "hg38",
    met.platform = "HM450",
    TSS.range = list(upstream = 2000, downstream = 2000),
    promoter = FALSE,
    rm.chr = NULL
)
getFunctionalGenes

Arguments

- **feature**: A GRRange object containing biofeature coordinate such as enhancer coordinates. If NULL only distal probes (2Kbp away from TSS will be selected) feature option is only usable when promoter option is FALSE.

- **TSS**: A GRRange object contains the transcription start sites. When promoter is FALSE, Union.TSS in ELMER.data will be used for default. When promoter is TRUE, UCSC gene TSS will be used as default (see detail). User can specify their own preference TSS annotation.

- **genome**: Which genome build will be used: hg38 (default) or hg19.

- **met.platform**: DNA methylation platform to retrieve data from: EPIC or 450K (default)

- **TSS.range**: A list specify how to define promoter regions. Default is upstream =2000bp and downstream=2000bp.

- **promoter**: A logical. If TRUE, function will output the promoter probes. If FALSE, function will output the distal probes overlapng with features. The default is FALSE.

- **rm.chr**: A vector of chromosome need to be remove from probes such as chrX chrY or chrM

Details

In order to get real distal probes, we use more comprehensive annotated TSS by both GENCODE and UCSC. However, to get probes within promoter regions need more accurate annotated TSS such as UCSC. Therefore, there are different settings for promoter and distal probe selection. But user can specify their own favorable TSS annotation. Then there won’t be any difference between promoter and distal probe selection. @return A GRanges object contains the coordinate of probes which locate within promoter regions or distal feature regions such as union enhancer from REMC and FANTOM5. @usage getFeatureProbe(feature, TSS, TSS.range = list(upstream = 2000, downstream = 2000), promoter = FALSE, rm.chr = NULL)

Value

A GRRange object containing probes that satisfy selecting criteria.

getFunctionalGenes  The getFunctionalGenes function

Description

Helper function to assess if the methylation of a probe is reversely correlated with the expression of its nearby genes.
getFunctionalGenes

Usage

getFunctionalGenes(
  target.probe,
  target.genes,
  MET_matrix,
  gene.expression.data,
  ProbeAnnotation,
  correlation = "negative",
  raw.pvalue.threshold = 0.05,
  adjusted.pvalue.threshold = 0.01
)

Arguments

target.probe character string indicating the probe to be evaluated.
target.genes character vector indicating the nearby genes of the target probe.
MET_matrix methylation data matrix for CpGs from group.1 and group.2.
gene.expression.data gene expression data matrix.
ProbeAnnotation GRange object of CpG probe annotation.
raw.pvalue.threshold raw p value from testing DNA methylation and gene expression
adjusted.pvalue.threshold adjusted p value from testing DNA methylation and gene expression

details

This function is probe-centered, which is used in the enhancer mode and the miRNA mode of EpiMix.

Value

dataframe with functional probe-gene pair and p values from the Wilcoxon test for methylation and gene expression.

Examples

data(Sample_EpiMixResults_Enhancer)
data(mRNA.data)
EpiMixResults <- Sample_EpiMixResults_Enhancer
target.probe <- EpiMixResults$FunctionalPairs$ Probe[1]
target.genes <- EpiMixResults$FunctionalPairs$ Gene
MET_matrix <- EpiMixResults$MethylationStates
ProbeAnnotation <- ExperimentHub::ExperimentHub()[["EH3675"]]
res <- getFunctionalGenes(target.probe, target.genes, MET_matrix, mRNA.data, ProbeAnnotation)
getLncRNAData

The getLncRNAData function

Description
Helper function to retrieve the lncRNA expression data from Experiment Hub

Usage
getLncRNAData(CancerSite)

Arguments
CancerSite TCGA cancer code

Value
local file path where the lncRNA expression data are saved

getMethStates

The getMethStates function

Description
Helper function that adds a methylation state label to each driver probe

Usage
gETCHMSStates(MethylMixResults, DM.probes)

Arguments
MethylMixResults the list object returned from the EpiMix function
DM.probes character vector of differentially methylated probes.

Value
a character vector with the methylation state ('Hypo', 'Hyper' or 'Dual') for each probe. The names for the vector are the probe names and the values are the methylation state.
getMethStates_Helper

The `getMethStates_Helper` function

Description

helper function to determine the methylation state based on DM values

Usage

```r
getMethStates_Helper(DMValues)
```

Arguments

- `DMValues`: a character vector indicating the DM values of a CpG site

Value

a character string indicating the methylation state of the CpG

GetNearGenes

`GetNearGenes` to collect nearby genes for one locus.

Description

GetNearGenes is a function to collect equal number of gene on each side of one locus. It can receive either multi Assay Experiment with both DNA methylation and gene Expression matrix and the names of probes to select nearby genes, or it can receive two granges objects TRange and geneAnnot.

Usage

```r
GetNearGenes(
  data = NULL,
  probes = NULL,
  geneAnnot = NULL,
  TRange = NULL,
  numFlankingGenes = 20
)
```
**getProbeAnnotation**

**Arguments**

- **data**: A multi Assay Experiment with both DNA methylation and gene Expression objects.
- **probes**: Name of probes to get nearby genes (it should be rownames of the DNA methylation object in the data argument object).
- **geneAnnot**: A GRanges object or Summarized Experiment object that contains coordinates of promoters for human genome.
- **TRange**: A GRanges object or Summarized Experiment object that contains coordinates of a list of targets loci.
- **numFlankingGenes**: A number determines how many gene will be collected totally. Then the number divided by 2 is the number of genes collected from each side of targets (number should be even) Default to 20.

**Value**

A data frame of nearby genes and information: genes’ IDs, genes’ symbols, distance with target and side to which the gene locate to the target.

**References**


---

**getProbeAnnotation** *The getProbeAnnotation function*

**Description**

Helper function to get the probe annotation based on mode.

**Usage**

getProbeAnnotation(mode, met.platform, genome)

**Arguments**

- **mode**: analytic mode
- **met.platform**: methylation platform
- **genome**: genome build version

**Value**

A ProbeAnnotation dataframe consisting of two columns: probe, gene
getRandomGenes

### Description

Helper function to get a set of random genes located on different chromosomes of the target CpG.

#### Usage

```r
getRandomGenes(
  target.probe,
  gene.expression.data,
  ProbeAnnotation,
  genome = "hg38",
  perm = 1000
)
```

#### Arguments

- `target.probe`: character string indicating the target CpG for generating the permutation p values.
- `gene.expression.data`: a matrix of gene expression data.
- `ProbeAnnotation`: GRanges object of probe annotation.
- `genome`: character string indicating the genome build version, can be either ‘hg19’ or ‘hg38’.
- `perm`: the number of permutation tests. Default: 1000

#### Value

A dataframe for the permutation genes and p values for the target CpG site.

---

getRegionNearGenes

### Description

Auxiliary function for GetNearGenes This will get the closest genes (n=numFlankingGenes) for a target region (TRange) based on a genome of reference gene annotation (geneAnnot). If the transcript level annotation (tssAnnot) is provided the Distance will be updated to the distance to the nearest TSS.
**Usage**

```
getRegionNearGenes(
  TRange = NULL,
  numFlankingGenes = 20,
  geneAnnot = NULL,
  tssAnnot = NULL
)
```

**Arguments**

- `TRange` A GRange object contains coordinate of targets.
- `numFlankingGenes` A number determine how many gene will be collected from each
- `geneAnnot` A GRange object contains gene coordinates of for human genome.
- `tssAnnot` A GRange object contains tss coordinates of for human genome.

**Value**

A data frame of nearby genes and information: genes’ IDs, genes’ symbols,

**Author(s)**

Tiago C Silva (maintainer: tiagochst@usp.br)

---

**getRoadMapEnhancerProbes**

```
getRoadMapEnhancerProbes
```

**Description**

**Usage**

```
getRoadMapEnhancerProbes(
  met.platform = "EPIC",
  genome = "hg38",
  functional.regions = c("EnhA1", "EnhA2"),
  listOfEpigenomes = NULL,
  ProbeAnnotation
)
```
GetSurvivalProbe

Arguments

- **met.platform**: character string indicating the methylation platform, can be either 'EPIC' or 'HM450'
- **genome**: character string indicating the genome build version, can be either 'hg19' or 'hg38'
- **functional.regions**: character vector indicating the MNEMONIC chromatin states that will be retrieved from the Roadmap epigenomics. Default values are the active enhancers: 'EnhA1', 'EnhA2'.
- **listOfEpigenomes**: character vector indicating which epigenome(s) to use for finding enhancers.
- **ProbeAnnotation**: GRange object of probe annotation.

Details

get the CpG probes that locate at the enhancer regions identified by the Roadmap epigenomics project

Value

a dataframe with enhancer probes and their chromosome coordinates

Examples

```r
met.platform = 'EPIC'
genome = 'hg38'
listOfEpigenomes = c('E034', 'E045', 'E047')
functional.regions = c('EnhA1', 'EnhA2', 'EnhG1', 'EnhG2')
df.enhancer.probes <- getEnhancerProbes(met.platform = met.platform,
                                        genome = genome,
                                        functional.regions = functional.regions,
                                        listOfEpigenomes = listOfEpigenomes)
```

---

GetSurvivalProbe  The GetSurvivalProbe function

Description

Get probes whose methylation state is predictive of patient survival
Usage

GetSurvivalProbe(
  EpiMixResults,  # List of objects returned from the EpiMix function
  TCGA_CancerSite = NULL,  # String indicating the TCGA cancer code (e.g. 'LUAD')
  clinical.data = NULL, # (If the TCGA_CancerSite is specified, this parameter is optional) Dataframe with survival information. Must contain at least three columns: 'sample.id', 'days_to_death', 'days_to_last_follow_up'.
  raw.pval.threshold = 0.05,  # numeric value indicating the raw p value threshold for selecting the survival predictive probes. Survival time is compared by log-rank test. Default: 0.05
  p.adjust.method = "none",  # character string indicating the statistical method for adjusting multiple comparisons, can be either of 'holm', 'hochberg', 'hommel', 'bonferroni', 'BH', 'BY', 'fdr', 'none'. Default: 'fdr'
  adjusted.pval.threshold = 0.05,  # numeric value indicating the adjusted p value threshold for selecting the survival predictive probes. Default: 0.05
  OutputRoot = ""  # path to save the output. If not null, the return value will be saved as 'SurvivalProbes.csv'.
)

Arguments

EpiMixResults List of objects returned from the EpiMix function
TCGA_CancerSite String indicating the TCGA cancer code (e.g. 'LUAD')
clinical.data (If the TCGA_CancerSite is specified, this parameter is optional) Dataframe with survival information. Must contain at least three columns: 'sample.id', 'days_to_death', 'days_to_last_follow_up'.
raw.pval.threshold numeric value indicating the raw p value threshold for selecting the survival predictive probes. Survival time is compared by log-rank test. Default: 0.05
p.adjust.method character string indicating the statistical method for adjusting multiple comparisons, can be either of 'holm', 'hochberg', 'hommel', 'bonferroni', 'BH', 'BY', 'fdr', 'none'. Default: 'fdr'
adjusted.pval.threshold numeric value indicating the adjusted p value threshold for selecting the survival predictive probes. Default: 0.05
OutputRoot path to save the output. If not null, the return value will be saved as 'SurvivalProbes.csv'.

Value

da dataframe with probes whose methylation state is predictive of patient survival and the p value.

Examples

library(survival)

data('Sample_EpiMixResults_miRNA')

survival.CpGs <- GetSurvivalProbe(EpiMixResults = Sample_EpiMixResults_miRNA,
  TCGA_CancerSite = 'LUAD')
getTSS

getTSS to fetch GENCODE gene annotation (transcripts level) from Bioconductor package biomaRt. If upstream and downstream are specified in TSS list, promoter regions of GENCODE gene will be generated.

Description

getTSS to fetch GENCODE gene annotation (transcripts level) from Bioconductor package biomaRt. If upstream and downstream are specified in TSS list, promoter regions of GENCODE gene will be generated.

Usage

getTSS(genome = "hg38", TSS = list(upstream = NULL, downstream = NULL))

Arguments

genome

Which genome build will be used: hg38 (default) or hg19.

TSS

A list. Contains upstream and downstream like TSS=list(upstream, downstream). When upstream and downstream is specified, coordinates of promoter regions with gene annotation will be generated.

Value

GENCODE gene annotation if TSS is not specified. Coordinates of GENCODE gene promoter regions if TSS is specified.

Author(s)

Lijing Yao (maintainer: lijingya@usc.edu)

get_firehoseData

The get_firehoseData function

Description

Gets data from TCGA's firehose.
Usage

get_firehoseData(
    downloadData = TRUE,
    saveDir = "/",
    TCGA_acronym_uppercase = "LUAD",
    dataType = "stddata",
    dataFileTag = "mRNAseq_Preprocess.Level_3",
    FFPE = FALSE,
    fileType = "tar.gz",
    gdacURL = "https://gdac.broadinstitute.org/runs/",
    untarUngzip = TRUE,
    printDisease_abbr = FALSE
)

Arguments

downloadData logical indicating if data should be downloaded (default: TRUE). If false, the url of the desired data is returned.

saveDir path to directory to save downloaded files.

TCGA_acronym_uppercase TCGA's cancer site code.

dataType type of data in TCGA (default: 'stddata').

dataFileTag name of the file to be downloaded (the default is to download RNAseq data, but this can be changed to download other data).

FFPE logical indicating if FFPE data should be downloaded (default: FALSE).

fileType type of downloaded file (default: 'fileType', other type not admitted at the moment).

gdacURL gdac url.

untarUngzip logical indicating if the gzip file downloaded should be untarred (default: TRUE).

printDisease_abbr if TRUE data is not downloaded but all the possible cancer sites codes are shown (default: FALSE).

Value

DownloadedFile path to directory with downloaded files.

mapTranscriptToGene

mapTranscriptToGene

Description

map the miRNA precursor names to HGNC
Usage

mapTranscriptToGene(transcripts)

Arguments

transcripts vector with the name of miRNA precursors

Value

a dataframe with two columns: 'Transcript' indicating the miRNA precursor names, 'Gene_name' indicating the actual human gene names (HGNC)

MethylMix_MixtureModel

*The MethylMix_MixtureModel function*

Description

Internal. Prepares all the structures to store the results and calls in a foreach loop a function that fits the mixture model in each gene.

Usage

MethylMix_MixtureModel(
  METcancer,
  METnormal = NULL,
  FunctionalGenes,
  NoNormalMode = FALSE
)

Arguments

METcancer matrix with methylation data for cancer samples (genes in rows, samples in columns).

METnormal matrix with methylation data for normal samples (genes in rows, samples in columns). If NULL no comparison to normal samples will be done.

FunctionalGenes vector with genes names to be considered for the mixture models.

NoNormalMode logical, if TRUE no comparison to normal samples is performed. Defaults to FALSE.
The MethylMix_ModelSingleGene function

Description

Internal. For a given gene, this function fits the mixture model, selects the number of components and defines the respective methylation states.

Usage

MethylMix_ModelSingleGene(
  GeneName,
  METdataVector,
  METdataNormalVector = NULL,
  NoNormalMode = FALSE,
  maxComp = 3,
  PvalueThreshold = 0.01,
  MeanDifferenceThreshold = 0.1,
  minSamplesPerGroup = 1
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeneName</td>
<td>character string with the name of the gene to model</td>
</tr>
<tr>
<td>METdataVector</td>
<td>vector with methylation data for cancer samples.</td>
</tr>
<tr>
<td>METdataNormalVector</td>
<td>vector with methylation data for normal samples. It can be NULL and then no normal mode will be used.</td>
</tr>
<tr>
<td>NoNormalMode</td>
<td>logical, if TRUE no comparison to normal samples is performed. Defaults to FALSE.</td>
</tr>
<tr>
<td>maxComp</td>
<td>maximum number of mixture components admitted in the model (3 by default).</td>
</tr>
<tr>
<td>PvalueThreshold</td>
<td>threshold to consider results significant.</td>
</tr>
</tbody>
</table>
MeanDifferenceThreshold
threshold in beta value scale from which two methylation means are considered different.

minSamplesPerGroup
minimum number of samples required to belong to a new mixture component in order to accept it. Default is 1 (not used). If -1, each component has to have at least 5% of all cancer samples.

Details
maxComp, PvalueThreshold, METDiffThreshold, minSamplesPerGroup are arguments for this function but are fixed in their default values for the user because they are not available in the main MethylMix function, to keep it simple. It would be easy to make them available to the user if we want to.

Value
NrComponents number of components identified.
Models an object with the parameters of the model fitted.
MethylationStates vector with DM values for each sample.
MixtureStates vector with DM values for each component.
Classifications a vector indicating to which component each sample was assigned.
FlipOverState

MethylMix_Predict

The MethylMix_Predict function

Description
Given a new data set with methylation data, this function predicts the mixture component for each new sample and driver gene. Predictions are based on posterior probabilities calculated with MethylMix'x fitted mixture model.

Usage
MethylMix_Predict(newBetaValuesMatrix, MethylMixResult)

Arguments
newBetaValuesMatrix
Matrix with new observations for prediction, genes/cpg sites in rows, samples in columns. Although this new matrix can have a different number of genes/cpg sites than the one provided as METcancer when running MethylMix, naming of genes/cpg sites should be the same.

MethylMixResult
Output object from MethylMix
**MethylMix_RemoveFlipOver**

**Value**

A matrix with predictions (indices of mixture component), driver genes in rows, new samples in columns.

**Description**

Internal. The estimated densities for each beta component can overlap, generating samples that look like being separated from their group. This function re classifies such samples.

**Usage**

```r
MethylMix_RemoveFlipOver(
  OrigOrder,
  MethylationState,
  classification,
  METdataVector,
  NrComponents,
  UseTrainedFlipOver = FALSE,
  FlipOverState = 0
)
```

**Arguments**

- `OrigOrder`: order of sorted values in the methylation vector.
- `MethylationState`: methylation states for this gene.
- `classification`: vector with integers indicating to which component each sample was classified into.
- `METdataVector`: vector with methylation values from the cancer samples.
- `NrComponents`: number of components in this gene.
- `UseTrainedFlipOver`:.
- `FlipOverState`.

**Value**

Corrected vectors with methylation states and classification.
**predictOneGene**  
*The predictOneGene function*

**Description**

Auxiliar function. Given a new vector of beta values, this function calculates a matrix with posterior prob of belonging to each mixture component (columns) for each new beta value (rows), and return the number of the mixture component with highest posterior probabilit

**Usage**

```r
predictOneGene(newVector, mixtureModel)
```

**Arguments**

- `newVector`: vector with new beta values
- `mixtureModel`: beta mixture model object for the gene being evaluated.

**Value**

A matrix with predictions (indices of mixture component), driver genes in rows, new samples in columns

---

**Preprocess_CancerSite_Methylation27k**  
*The Preprocess_CancerSite_Methylation27k function*

**Description**

Internal. Pre-processes DNA methylation data from TCGA from Illymina 27k arrays.

**Usage**

```r
Preprocess_CancerSite_Methylation27k(
   CancerSite,
   METdirectory,
   doBatchCorrection,
   batch.correction.method,
   MissingValueThreshold
)
```
Preprocess_DNAMethylation

Arguments

CancerSite  character of length 1 with TCGA cancer code.
METdirectory  character with directory where a folder for downloaded files will be created. Can be the object returned by the Download_DNAmethylation function.
MissingValueThreshold  threshold for removing samples or genes with missing values.

Value

List with pre processed methylation data for cancer and normal samples.

Preprocess_DNAMethylation  
The Preprocess_DNAMethylation function

Description

Preprocess DNA methylation data from the GEO database.

Usage

Preprocess_DNAMethylation(
  methylation.data,
  met.platform = "EPIC",
  genome = "hg38",
  sample.info = NULL,
  group.1 = NULL,
  group.2 = NULL,
  sample.map = NULL,
  rm.chr = c("chrX", "chrY"),
  MissingValueThresholdGene = 0.2,
  MissingValueThresholdSample = 0.2,
  doBatchCorrection = FALSE,
  BatchData = NULL,
  batch.correction.method = "Seurat",
  cores = 1
)

Arguments

methylation.data  matrix of DNA methylation data with CpG in rows and sample names in columns.
met.platform  character string indicating the type of the Illumina Infinium BeadChip for collecting the methylation data. Should be either 'HM450' or 'EPIC'. Default: 'EPIC'
Preprocess_DNAMethylation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>genome</td>
<td>character string indicating the genome build version for retrieving the probe annotation. Should be either 'hg19' or 'hg38'. Default: 'hg38'.</td>
</tr>
<tr>
<td>sample.info</td>
<td>dataframe that maps each sample to a study group. Should contain two columns: the first column (named: 'primary') indicating the sample names, and the second column (named: 'sample.type') indicating which study group each sample belongs to (e.g., “Experiment” vs. “Control”, “Cancer” vs. “Normal”). Sample names in the 'primary' column must coincide with the column names of the methylation.data. Please see details for more information. Default: NULL.</td>
</tr>
<tr>
<td>group.1</td>
<td>character vector indicating the name(s) for the experiment group. The values must coincide with the values in the 'sample.type' of the sample.info dataframe. Please see details for more information. Default: NULL.</td>
</tr>
<tr>
<td>group.2</td>
<td>character vector indicating the names(s) for the control group. The values must coincide with the values in the 'sample.type' of the sample.info dataframe. Please see details for more information. Default: NULL.</td>
</tr>
<tr>
<td>sample.map</td>
<td>dataframe for mapping the GEO accession ID (column names) to the actual sample names. Can be the output from the GEO_getSampleMap function. Default: NULL.</td>
</tr>
<tr>
<td>rm.chr</td>
<td>character vector indicating the probes on which chromosomes to be removed. Default: 'chrX', 'chrY'.</td>
</tr>
</tbody>
</table>

MissingValueThresholdGene

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MissingValueThresholdGene</td>
<td>threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default: 0.3.</td>
</tr>
<tr>
<td>MissingValueThresholdSample</td>
<td>threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default: 0.1.</td>
</tr>
</tbody>
</table>

DoBatchCorrection

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doBatchCorrection</td>
<td>logical indicating whether to perform batch correction. If TRUE, the batch data need to be provided.</td>
</tr>
<tr>
<td>BatchData</td>
<td>dataframe with batch information. Should contain two columns: the first column indicating the actual sample names, the second column indicating the batch. Users are expected to retrieve the batch information from the GEO on their own, but this can also be done using the GEO_getSampleInfo function with the 'group.column' as the column indicating the batch for each sample. Default: NULL.</td>
</tr>
<tr>
<td>batch.correction.method</td>
<td>character string indicating the method that will be used for batch correction. Should be either 'Seurat' or 'Combat'. Default: 'Seurat'.</td>
</tr>
<tr>
<td>cores</td>
<td>number of CPU cores to be used for batch effect correction. Default: 1.</td>
</tr>
</tbody>
</table>

Details

The data preprocessing pipeline includes: (1) eliminating samples and genes with too many NAs, imputing NAs. (2) (optional) mapping the column names of the DNA methylation data to the actual sample names based on the information from 'sample.map'. (3) (optional) removing CpG probes on the sex chromosomes or the user-defined chromosomes. (4) (optional) doing Batch correction. If both sample.info and group.1 and group.2 information are provided, the function will perform missing value estimation and batch correction on group.1 and group.2 separately. This will ensure that
the true difference between group.1 and group.2 will not be obscured by missing value estimation and batch correction.

Value

DNA methylation data matrix with probes in rows and samples in columns.

Examples

```r
# data from GEO
data(MET.data)
data(LUAD.sample.annotation)

Preprocessed_Data <- Preprocess_DNAMethylation(MET.data,
                                           met.platform = 'HM450',
                                           sample.info = LUAD.sample.annotation,
                                           group.1 = 'Cancer',
                                           group.2 = 'Normal')
```

Preprocess_GeneExpression

The **Preprocess_GeneExpression** function

Description

Preprocess the gene expression data from the GEO database.

Usage

```r
Preprocess_GeneExpression(
    gene.expression.data,
    sample.info = NULL,
    group.1 = NULL,
    group.2 = NULL,
    sample.map = NULL,
    MissingValueThresholdGene = 0.3,
    MissingValueThresholdSample = 0.1,
    doBatchCorrection = FALSE,
    BatchData = NULL,
    batch.correction.method = "Seurat",
    cores = 1
)
```
Arguments

gene.expression.data
a matrix of gene expression data with gene in rows and samples in columns.

sample.info
dataframe that maps each sample to a study group. Should contain two columns:
the first column (named: 'primary') indicating the sample names, and the sec-
ond column (named: 'sample.type') indicating which study group each sample
belongs to (e.g., “Experiment” vs. “Control”, “Cancer” vs. “Normal”). Sample
names in the 'primary' column must coincide with the column names of the
methylation.data. Please see details for more information. Default: NULL.

group.1
character vector indicating the name(s) for the experiment group. The values
must coincide with the values in the 'sample.type' of the sample.info dataframe. Please
see details for more information. Default: NULL.

group.2
character vector indicating the names(s) for the control group. The values must
coincide with the values in the 'sample.type' of the sample.info dataframe. Please
see details for more information. Default: NULL.

sample.map
dataframe for mapping the GEO accession ID (column names) to the actual sam-
ple names. Can be the output from the GEO_getSampleMap function. Default:
NULL.

MissingValueThresholdGene
threshold for missing values per gene. Genes with a percentage of NAs greater
than this threshold are removed. Default is 0.3.

MissingValueThresholdSample
threshold for missing values per sample. Samples with a percentage of NAs
greater than this threshold are removed. Default is 0.1.

doBatchCorrection
logical indicating whether to perform batch correction. If TRUE, the batch data
need to be provided.

BatchData
dataframe with batch information. Should contain two columns: the first col-
umn indicating the actual sample names, the second column indicating the batch.
Users are expected to retrieve the batch information from GEO on their own, but
this can also be done using the GEO_getSampleInfo function with the 'group.column'
as the column indicating the batch for each sample. Default: NULL.

batch.correction.method
character string indicating the method that be used for batch correction. Should
be either 'Seurat' or 'Combat'. Default: 'Seurat'.

cores
number of CPU cores to be used for batch effect correction. Default: 1

Details

The preprocessing pipeline includes: (1) eliminating samples and genes with too many NAs and
imputing NAs. (2) if the gene names (rownames) in the gene expression data are ensembl_gene_ids
or ensembl_transcript_ids, translate the gene names or the transcript names to human gene symbols
(HGNC). (3) mapping the column names of the gene expression data to the actual sample names
based on the information from 'sample.map'. (4) doing batch correction.
Value

gene expression data matrix with genes in rows and samples in columns.

Examples

```
{  
data(mRNA.data)
data(LUAD.sample.annotation)
Preprocessed_Data <- Preprocess_GeneExpression(gene.expression.data = mRNA.data,
sample.info = LUAD.sample.annotation,
   group.1 = 'Cancer',
   group.2 = 'Normal')
}
```

Preprocess_MAdata_Cancer

The Preprocess_MAdata_Cancer function

Description

Internal. Pre-process gene expression data for cancer samples.

Usage

```
Preprocess_MAdata_Cancer(  
CancerSite,  
Directory,  
File,  
MissingValueThresholdGene = 0.3,  
MissingValueThresholdSample = 0.1,  
doBatchCorrection,  
batch.correction.method,  
BatchData
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CancerSite</td>
<td>TCGA code for the cancer site.</td>
</tr>
<tr>
<td>Directory</td>
<td>Directory.</td>
</tr>
<tr>
<td>File</td>
<td>File.</td>
</tr>
<tr>
<td>MissingValueThresholdGene</td>
<td>threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.</td>
</tr>
<tr>
<td>MissingValueThresholdSample</td>
<td>threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.</td>
</tr>
</tbody>
</table>
Preprocess_MAdata_Normal

**The Preprocess_MAdata_Normal function**

**Description**

Internal. Pre-process gene expression data for normal samples.

**Usage**

```r
Preprocess_MAdata_Normal(
  CancerSite, 
  Directory, 
  File, 
  MissingValueThresholdGene, 
  MissingValueThresholdSample, 
  doBatchCorrection, 
  batch.correction.method, 
  BatchData
)
```

**Arguments**

- `CancerSite`  
  TCGA code for the cancer site.
- `Directory`  
  Directory.
- `File`  
  File.
- `MissingValueThresholdGene`  
  threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.
- `MissingValueThresholdSample`  
  threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.

**Value**

The data matrix.
removeDuplicatedGenes

**Description**

sum up the transcript expression values if a gene has multiple transcripts

**Usage**

removeDuplicatedGenes(GEN_data)

**Arguments**

GEN_data  gene expression data matrix

**Value**

gene expression data matrix with duplicated genes removed

splitmatrix

**Description**

The splitmatrix function

**Usage**

splitmatrix(x, by = "row")

**Arguments**

x  A matrix

by  A character specify if split the matrix by row or column.

**Value**

A list each of which is the value of each row/column in the matrix.
**TCGA_Download_DNAmethylation**

*The TCGA_Download_DNAmethylation function*

**Description**

Download DNA methylation data from TCGA.

**Usage**

```
TCGA_Download_DNAmethylation(CancerSite, TargetDirectory, downloadData = TRUE)
```

**Arguments**

- **CancerSite**: character of length 1 with TCGA cancer code.
- **TargetDirectory**: character with directory where a folder for downloaded files will be created.
- **downloadData**: logical indicating if data should be downloaded (default: TRUE). If false, the url of the desired data is returned.

**Value**

list with paths to downloaded files for both 27k and 450k methylation data.

**Examples**

```
METdirectories <- TCGA_Download_DNAmethylation(CancerSite = 'OV', TargetDirectory = tempdir())
```

---

**TCGA_Download_GeneExpression**

*The TCGA_Download_GeneExpression function*

**Description**

Download gene expression data from TCGA.

**Usage**

```
TCGA_Download_GeneExpression(
    CancerSite,
    TargetDirectory,
    mode = "Regular",
    downloadData = TRUE
)
```

---
TCGA_EstimateMissingValues_MolecularData

Arguments

CancerSite character string indicating the TCGA cancer code.
TargetDirectory character with directory where a folder for downloaded files will be created.
mode character string indicating whether we should download the gene expression data for miRNAs or IncRNAs, instead of for protein-coding genes. See details for more information.
downloadData logical indicating if the data should be downloaded (default: TRUE). If False, the url of the desired data is returned.

Details

mode: when mode is set to 'Regular', this function downloads the level 3 RNAseq data (file tag 'mRNAseq_Preprocess.Level_3'). Since there is not enough RNAseq data for OV and GBM, the micro array data is downloaded. If you plan to run the EpiMix on miRNA- or lncRNA-coding genes, please specify the 'mode' parameter to 'miRNA' or 'lncRNA'.

Value

list with paths to downloaded files for gene expression.

Examples

# Example #1 : download regular gene expression data for ovarian cancer
GEdirectories <- TCGA_Download_GeneExpression(CancerSite = 'OV', TargetDirectory = tempdir())

# Example #2 : download miRNA expression data for ovarian cancer
GEdirectories <- TCGA_Download_GeneExpression(CancerSite = 'OV', TargetDirectory = tempdir(),
mode = 'miRNA')

# Example #3 : download lncRNA expression data for ovarian cancer
GEdirectories <- TCGA_Download_GeneExpression(CancerSite = 'OV', TargetDirectory = tempdir(),
mode = 'lncRNA')

Description

Internal. Deletes samples and genes with more NAs than the respective thresholds. Imputes other NAs values.
Usage

TCGA_EstimateMissingValues_MolecularData(
    MET_Data,
    MissingValueThresholdGene = 0.3,
    MissingValueThresholdSample = 0.1
)

Arguments

MET_Data matrix of gene expression data
MissingValueThresholdGene threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.
MissingValueThresholdSample threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.

Value
gene expression data with no missing values.

---------

TCGA_GENERIC_CheckBatchEffect

The TCGA_GENERIC_CheckBatchEffect function

---------

Description

Internal. Checks if batch correction is needed.

Usage

TCGA_GENERIC_CheckBatchEffect(GEN_Data, BatchData)

Arguments

GEN_Data matrix with data to be corrected for batch effects.
BatchData Batch data.

Value

the p value from ANOVA test on PCA values.
TCGA_GENERIC_CleanUpSampleNames

The TCGA_GENERIC_CleanUpSampleNames function

Description

Internal. Cleans the samples IDs into the 12 digit format and removes doubles.

Usage

TCGA_GENERIC_CleanUpSampleNames(GEN_Data, IDlength = 12)

Arguments

GEN_Data data matrix.
IDlength length of samples ID.

Value

data matrix with cleaned sample names.

TCGA_GENERIC_GetSampleGroups

The TCGA_GENERIC_GetSampleGroups function

Description

Internal. Looks for the group of the samples (normal/cancer).

Usage

TCGA_GENERIC_GetSampleGroups(SampleNames)

Arguments

SampleNames vector with sample names.

Value

a list.
TCGA_GENERIC_LoadIlluminaMethylationData

**Description**

Internal. Read in an illumina methylation file with the following format: header row with sample labels, 2nd header row with 4 columns per sample: beta-value, geneSymbol, chromosome and GenomicCoordinate. The first column has the probe names.

**Usage**

```
TCGA_GENERIC_LoadIlluminaMethylationData(Filename)
```

**Arguments**

- **Filename**
  - name of the file with the data.

**Value**

methyltion data.

---

TCGA_GENERIC_MergeData

**Description**

Internal.

**Usage**

```
TCGA_GENERIC_MergeData(NewIDListUnique, DataMatrix)
```

**Arguments**

- **NewIDListUnique**
  - unique rownames of data.
- **DataMatrix**
  - data matrix.

**Value**

data matrix.
The TCGA_GENERIC_MET_ClusterProbes_Helper_ClusterGenes_with_hclust function

**Description**

Internal. Cluster probes into genes.

**Usage**

```r
TCGA_GENERIC_MET_ClusterProbes_Helper_ClusterGenes_with_hclust(
  Gene,
  ProbeAnnotation,
  MET_Cancer,
  MET_Normal = NULL,
  CorThreshold = 0.4
)
```

**Arguments**

- **Gene**: gene.
- **ProbeAnnotation**: data set matching probes to genes.
- **MET_Cancer**: data matrix for cancer samples.
- **MET_Normal**: data matrix for normal samples.
- **CorThreshold**: correlation threshold for cutting the clusters.

**Value**

List with the clustered data sets and the mapping between probes and genes.

---

The TCGA_GetData function

**Description**

This function wraps the functions for downloading, pre-processing and analysis of the DNA methylation and gene expression data from the TCGA project.
Usage

TCGA_GetData(
  CancerSite,
  mode = "Regular",
  outputDirectory = ".",
  doBatchCorrection = FALSE,
  batch.correction.method = "Seurat",
  roadmap.epigenome.ids = NULL,
  roadmap.epigenome.groups = NULL,
  forceUse450K = FALSE,
  cores = 1
)

Arguments

CancerSite character string indicating the TCGA cancer code. The information can be found at: https://gdc.cancer.gov/resources-tcga-users/tcga-code-tables/tcga-study-abbreviations

mode character string indicating the analytic mode to model DNA methylation. Should be one of the followings: 'Regular', 'Enhancer', 'miRNA' or 'lncRNA'. Default: 'Regular'. See details for more information.

outputDirectory character string indicating the file path to save the output.

doBatchCorrection logical indicating whether to do batch effect correction during preprocessing. Default: False.

batch.correction.method character string indicating the method to perform batch effect correction. The value should be either 'Seurat' or 'Combat'. Seurat is much faster than the Combat. Default: 'Seurat'.

roadmap.epigenome.ids character vector indicating the epigenome ID(s) to be used for selecting enhancers. See details for more information. Default: NULL.

roadmap.epigenome.groups character vector indicating the tissue group(s) to be used for selecting enhancers. See details for more information. Default: NULL.

forceUse450K logic indicating whether force to use only 450K methylation data. Default: FALSE

cores Number of CPU cores to be used for computation.

Details

mode: EpiMix incorporates four alternative analytic modes for modeling DNA methylation: “Regular,” “Enhancer”, “miRNA” and “lncRNA”. The four analytic modes target DNA methylation analysis on different genetic elements. The Regular mode aims to model DNA methylation at proximal cis-regulatory elements of protein-coding genes. The Enhancer mode targets DNA methylation analysis on distal enhancers. The miRNA or lncRNA mode focuses on methylation analysis of miRNA- or lncRNA-coding genes.
roadmap.epigenome.groups & roadmap.epigenome.ids:

Since enhancers are cell-type or tissue-type specific, EpiMix needs to know the reference tissues or cell types in order to select proper enhancers. EpiMix identifies enhancers from the RoadmapEpigenomic project (Nature, PMID: 25693563), in which enhancers were identified by ChromHMM in over 100 tissue and cell types. Available epigenome groups (a group of relevant cell types) or epigenome ids (individual cell types) can be obtained from the original publication (Nature, PMID: 25693563, figure 2). They can also be retrieved from the list.epigenomes() function. If both roadmap.epigenome.groups and roadmap.epigenome.ids are specified, EpiMix will select all the epigenomes from the combination of the inputs.

Value

The results from EpiMix is a list with the following components:

- **MethylationDrivers**
  - CpG probes identified as differentially methylated by EpiMix.
- **NrComponents**
  - The number of methylation states found for each driver probe.
- **MixtureStates**
  - A list with the DM-values for each driver probe. Differential Methylation values (DM-values) are defined as the difference between the methylation mean of samples in one mixture component from the experiment group and the methylation mean in samples from the control group, for a given probe.
- **MethylationStates**
  - Matrix with DM-values for all driver probes (rows) and all samples (columns).
- **Classifications**
  - Matrix with integers indicating to which mixture component each sample in the experiment group was assigned to, for each probe.
- **Models**
  - Beta mixture model parameters for each driver probe.
- **group.1**
  - sample names in group.1 (experimental group).
- **group.2**
  - sample names in group.2 (control group).
- **FunctionalPairs**
  - Dataframe with the prevalence of differential methylation for each CpG probe in the sample population, and fold change of mRNA expression and P values for each significant probe-gene pair.

Examples

```r
# Example #1 - Regular mode
EpiMixResults <- TCGA_GetData(CancerSite = 'LUAD',
                               outputDirectory = tempdir(),
                               cores = 8)

# Example #2 - Enhancer mode
EpiMixResults <- TCGA_GetData(CancerSite = 'LUAD',
                               mode = 'Enhancer',
                               roadmap.epigenome.ids = 'E097',
                               outputDirectory = tempdir(),
                               cores = 8)
```
Example #3 - miRNA mode
EpiMixResults <- TCGA_GetData(CancerSite = 'LUAD',
    mode = 'miRNA',
    outputDirectory = tempdir(),
    cores = 8)

# Example #4 - lncRNA mode
EpiMixResults <- TCGA_GetData(CancerSite = 'LUAD',
    mode = 'lncRNA',
    outputDirectory = tempdir(),
    cores = 8)

TCGA_GetSampleInfo

The TCGA_GetSampleInfo function

Description
The TCGA_GetSampleInfo function

Usage
TCGA_GetSampleInfo(METProcessedData, CancerSite = "LUAD", TargetDirectory = "")

Arguments

METProcessedData
Matrix of preprocessed methylation data.

CancerSite
Character string of TCGA study abbreviation.

TargetDirectory
Path to save the sample.info. Default: "."

Details
Generate the 'sample.info' dataframe for TCGA data.

Value
A dataframe for the sample groups. Contains two columns: the first column (named: 'primary') indicating the sample names, and the second column (named: 'sample.type') indicating whether each sample is a Cancer or Normal tissue.

Examples
{
data(MET.data)
sample.info <- TCGA_GetSampleInfo(MET.data, CancerSite = 'LUAD')
}
**TCGA_Load_MethylationData**

*The TCGA_Load_MethylationData function*

---

**Description**

The TCGA_Load_MethylationData function

**Usage**

```r
TCGA_Load_MethylationData(METdirectory, ArrayType)
```

**Arguments**

- **METdirectory**  
  path to the 27K or 450K data

- **ArrayType**  
  character string indicating the array type, can be either '27K' or '450K'

**Details**

load 27K or 450K methylation data into memory

**Value**

matrix of methylation data with probes in rows and patient in columns

---

**TCGA_Load_MolecularData**

*The TCGA_Load_MolecularData function*

---

**Description**

Internal. Reads in gene expression data. Deletes samples and genes with more NAs than the respective thresholds. Imputes other NAs values.

**Usage**

```r
TCGA_Load_MolecularData(Filename)
```

**Arguments**

- **Filename**  
  name of the file with the data.

- **MissingValueThresholdGene**  
  threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.

- **MissingValueThresholdSample**  
  threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.
Value
gene expression data.

---

### TCGA_Preprocess_DNAmethylation

#### The TCGA_Preprocess_DNAmethylation function

**Description**
Pre-processes DNA methylation data from TCGA.

**Usage**

```r
TCGA_Preprocess_DNAmethylation(
  CancerSite,
  METdirectories,
  doBatchCorrection = FALSE,
  batch.correction.method = "Seurat",
  MissingValueThreshold = 0.2,
  cores = 1,
  use450K = FALSE
)
```

**Arguments**

- **CancerSite** character string indicating the TCGA cancer code.
- **METdirectories** character vector with directories with the downloaded data. It can be the object returned by the TCGA_Download_DNAmethylation function.
- **doBatchCorrection** logical indicating whether to perform batch correction. Default: False.
- **batch.correction.method** character string indicating the method to perform batch correction. The value should be either 'Seurat' or 'Combat'. Default: 'Seurat'. Note: Seurat is much faster than the Combat.
- **MissingValueThreshold** numeric values indicating the threshold for removing samples or genes with missing values. Default: 0.2.
- **cores** integer indicating the number of cores to be used for performing batch correction with Combat.
- **use450K** logic indicating whether to force use 450K, instead of 27K data.

**Details**
Pre-process includes eliminating samples and genes with too many NAs, imputing NAs, and doing Batch correction. If there are samples with both 27k and 450k data, the 27k data will be used only if the sample number in the 27k data is greater than the 450k data and there is more than 50 samples in the 27k data. Otherwise, the 450k data is used and the 27k data is discarded.
The TCGA_Preprocess_GeneExpression function

**Description**

Pre-processes gene expression data from TCGA.

**Usage**

```r
TCGA_Preprocess_GeneExpression(
  CancerSite,  # character string indicating the TCGA cancer code.
  MAdirectories, # character vector with directories with the downloaded data. It can be the object returned by the GEO_Download_GeneExpression function.
  mode = "Regular", # character string indicating whether the genes in the gene expression data are miRNAs or lncRNAs. Should be either 'Regular', 'Enhancer', 'miRNA' or 'lncRNA'. This value should be consistent with the same parameter in the TCGA_Download_GeneExpression function. Default: 'Regular'.
  doBatchCorrection = FALSE, # logical indicating whether to perform batch effect correction. Default: False.
  batch.correction.method = "Seurat", # character string indicating the method to perform batch correction. The value should be either 'Seurat' or 'Combat'. Default: 'Seurat'. Seurat is much faster than the Combat.
  MissingValueThresholdGene = 0.3, # Missing value threshold for gene expression data.
  MissingValueThresholdSample = 0.1, # Missing value threshold for gene expression data.
  cores = 1 # number of cores to use when parallel processing.
)
```

**Arguments**

- `CancerSite` character string indicating the TCGA cancer code.
- `MAdirectories` character vector with directories with the downloaded data. It can be the object returned by the GEO_Download_GeneExpression function.
- `mode` character string indicating whether the genes in the gene expression data are miRNAs or lncRNAs. Should be either 'Regular', 'Enhancer', 'miRNA' or 'lncRNA'. This value should be consistent with the same parameter in the TCGA_Download_GeneExpression function. Default: 'Regular'.
- `doBatchCorrection` logical indicating whether to perform batch effect correction. Default: False.
- `batch.correction.method` character string indicating the method to perform batch correction. The value should be either 'Seurat' or 'Combat'. Default: 'Seurat'. Seurat is much faster than the Combat.
MissingValueThresholdGene

threshold for missing values per gene. Genes with a percentage of NAs greater than this threshold are removed. Default is 0.3.

MissingValueThresholdSample

threshold for missing values per sample. Samples with a percentage of NAs greater than this threshold are removed. Default is 0.1.

cores

integer indicating the number of cores to be used for performing batch correction with Combat

Details

Pre-process includes eliminating samples and genes with too many NAs, imputing NAs, and doing Batch correction. If the rownames of the gene expression data are ensembl ENSG names or ENST names, the function will convert them to the human gene symbol (HGNC).

Value

pre-processed gene expression data matrix.

Examples

# Example #1: Preprocessing gene expression for Regular mode
GEdirectories <- TCGA_Download_GeneExpression(CancerSite = 'OV',
                                            TargetDirectory = tempdir())
GEProcessedData <- TCGA_Preprocess_GeneExpression(CancerSite = 'OV',
                                                   MAdirectories = GEdirectories)

# Example #2: Preprocessing gene expression for miRNA mode
GEdirectories <- TCGA_Download_GeneExpression(CancerSite = 'OV',
                                             TargetDirectory = tempdir(),
                                             mode = 'miRNA')
GEProcessedData <- TCGA_Preprocess_GeneExpression(CancerSite = 'OV',
                                                   MAdirectories = GEdirectories,
                                                   mode = 'miRNA')

# Example #3: Preprocessing gene expression for lncRNA mode
GEdirectories <- TCGA_Download_GeneExpression(CancerSite = 'OV',
                                             TargetDirectory = tempdir(),
                                             mode = 'lncRNA')
GEProcessedData <- TCGA_Preprocess_GeneExpression(CancerSite = 'OV',
                                                   MAdirectories = GEdirectories,
                                                   mode = 'lncRNA')
The TCGA_Process_EstimateMissingValues function

Description

Internal. Removes patients and genes with more missing values than the MissingValueThreshold, and imputes remaining missing values using Tibshirani’s KNN method.

Usage

TCGA_Process_EstimateMissingValues(MET_Data, MissingValueThreshold = 0.2)

Arguments

- MET_Data: data matrix.
- MissingValueThreshold: threshold for removing samples and genes with too many missing values.

Value

the data set with imputed values and possibly some genes or samples deleted.

The TCGA_Select_Dataset function

Description

internal function to select which MET dataset to use

Usage

TCGA_Select_Dataset(CancerSite, MET_Data_27K, MET_Data_450K, use450K)

Arguments

- CancerSite: TCGA cancer code
- MET_Data_27K: matrix of MET_Data_27K
- MET_Data_450K: matrix of MET_Data_450K
- use450K: logic indicating whether to force use 450K data

Value

the selected MET data set
test_gene_expr  

The test_gene_expr function

Description

Helper function to test whether the expression levels of a gene is reversely correlated with the methylation state of a probe.

Usage

test_gene_expr(
    gene,
    probe,
    DM_values,
    gene.expr.values,
    correlation = "negative"
)

Arguments

gene  character string indicating a target gene to be modeled.
probe  character string indicating a probe mapped to the target gene.
DM_values  a vector of DM values for the probe. The names of the element should be sample names.
gene.expr.values  a vector of gene expression values for the tested gene. The names of the vector are sample names.
correlation  character indicating the direction of correlation between the methylation state of the CpG site and the gene expression levels. Can be either 'negative' or 'positive'.
raw.pvalue.threshold  raw p value from testing DNA methylation and gene expression
adjusted.pvalue.threshold  adjusted p value from testing DNA methylation and gene expression

Value

dataframe with functional probe-gene pairs and corresponding p values obtained from the Wilcoxon test for gene expression and methylation.
**translateMethylMixResults**

*The translateMethylMixResults function*

**Description**

unfold clustered MethylMix results to single CpGs

**Usage**

```
translateMethylMixResults(MethylMixResults, probeMapping)
```

**Arguments**

- `MethylMixResults` list of MethylMix output
- `probeMapping` dataframe of probe to gene-cluster mapping

**Value**

list of unfolded MethylMix results

---

**validEpigenomes**

*The validEpigenomes function*

**Description**

check user input for roadmap epigenome groups or ids

**Usage**

```
validEpigenomes(roadmap.epigenome.groups, roadmap.epigenome.ids)
```

**Arguments**

- `roadmap.epigenome.groups` epigenome groups
- `roadmap.epigenome.ids` epigenome ids

**Value**

a character vector of selected epigenome ids
Index

* cluster_probes
  ClusterProbes, 10

* download
  GEO_Download_DNA_methylation, 32
  GEO_Download_GeneExpression, 33
  TCGA_Download_DNA_methylation, 62
  TCGA_Download_GeneExpression, 62

* internal
  .getComp, 4
  .getMetGroup, 5
  .mapProbeGene, 5
  .splitMetData, 6
  BatchCorrection_Combat, 7
  BatchCorrection_Seurat, 8
  betaEst_2, 8
  blc_2, 9
  ComBat_NoFiles, 11
  combineForEachOutput, 12
  convertAnnotToDF, 12
  convertGeneNames, 13
  CorrectBatchEffect, 13
  EpiMix_getInfiniumAnnotation, 18
  filterLinearProbes, 26
  filterMethMatrix, 27
  GEO_EstimateMissingValues_Methylation, 34
  GEO_EstimateMissingValues_Molecular, 35
  get.chromosome, 37
  get.prevalence, 37
  get_firehoseData, 48
  getFunctionalGenes, 39
  getLncRNAData, 41
  getMethStates, 41
  getRandomGenes, 44
  getRoadMapEnhancerProbes, 45
  mapTranscriptToGene, 49
  MethylMix_MixtureModel, 50
  MethylMix_ModelSingleGene, 51
  MethylMix_RemoveFlipOver, 53
  Preprocess_CancerSite_Methylation27k, 54
  Preprocess_MAdapata_Cancer, 59
  Preprocess_MAdapata_Normal, 60
  splitmatrix, 61
  TCGA_EstimateMissingValues_MolecularData, 63
  TCGA_GENERIC_ChecBatchEffect, 64
  TCGA_GENERIC_CleanUpSampleNames, 65
  TCGA_GENERIC_GetSampleGroups, 65
  TCGA GENERIC_Load illuminaMethylationData, 66
  TCGA GENERIC_MergeData, 66
  TCGA GENERIC_MET_ClusterProbes_Helper_ClusterGenes_with_hclust, 67
  TCGA_Load_MethylationData, 71
  TCGA_Load_MolecularData, 71
  TCGA_Process_EstimateMissingValues, 75
  test_gene_expr, 76

* preprocess
  Preprocess_DNA_methylation, 55
  Preprocess_GeneExpression, 57
  TCGA_Preprocess_DNA_methylation, 72
  TCGA_Preprocess_GeneExpression, 73

* purpose
  GEO_GetSampleInfo, 35
  GEO_getSampleMap, 36

* testing
  GEO_GetSampleInfo, 35
  GEO_getSampleMap, 36
  .extractPriMiRNA, 4
  .getComp, 4
  .getMetGroup, 5
  .mapProbeGene, 5
  .splitMetData, 6
  addDistNearestTSS, 6
addGeneNames, 7
BatchCorrection_Combat, 7
BatchCorrection_Seurat, 8
betaEst_2, 8
blc_2, 9
calcDistNearestTSS, 9
ClusterProbes, 10
ComBat_NoFiles, 11
combineForEachOutput, 12
convertAnnotToDF, 12
convertGeneNames, 13
CorrectBatchEffect, 13
EpiMix, 14
EpiMix_getInfiniumAnnotation, 18
EpiMix_PlotGene, 19
EpiMix_PlotModel, 21
EpiMix_PlotProbe, 23
EpiMix_PlotSurvival, 25
filterLinearProbes, 26
filterMethMatrix, 27
filterProbes, 28
find_miRNA_targets, 29
functionEnrich, 30
generateFunctionalPairs, 31
GEO_Download_DNAMethylation, 32
GEO_Download_GeneExpression, 33
GEO_EstimateMissingValues_Methylation, 34
GEO_EstimateMissingValues_Molecular, 35
GEO_GetSampleInfo, 35
GEO_getSampleMap, 36
get.chromosome, 37
going.prevalence, 37
get.Pvalue.p, 38
get_firehoseData, 48
getFeatureProbe, 38
getFunctionalGenes, 39
getLncRNAData, 41
getMethStates, 41
getMethStates_Helper, 42
GetNearGenes, 42
getProbeAnnotation, 43
getRandomGenes, 44
getRegionNearGenes, 44
getRoadMapEnhancerProbes, 45
GetSurvivalProbe, 46
getTSS, 48
mapTranscriptToGene, 49
MethylMix_MixtureModel, 50
MethylMix_ModelSingleGene, 51
MethylMix_Predict, 52
MethylMix_RemoveFlipOver, 53
predictOneGene, 54
Preprocess_CancerSite_Methylation27k, 54
Preprocess_DNAmethylation, 55
Preprocess_GeneExpression, 57
Preprocess_MAdataset_Cancer, 59
Preprocess_MAdataset_Normal, 60
removeDuplicatedGenes, 61
splitmatrix, 61
TCGA_Download_DNAmethylation, 62
TCGA_Download_GeneExpression, 62
TCGA_EstimateMissingValues_MolecularData, 63
TCGA_GENERIC_CheckBatchEffect, 64
TCGA_GENERIC_CleanUpSampleNames, 65
TCGA_GENERIC_GetSampleGroups, 65
TCGA_GENERIC_LoadIlluminaMethylationData, 66
TCGA_GENERIC_MergeData, 66
TCGA_GENERIC_MET_ClusterProbes_Helper_ClusterGenes_with_hclust, 67
TCGA_GetData, 67
TCGA_GetSampleInfo, 70
TCGA_Load_MethylationData, 71
TCGA_Load_MolecularData, 71
TCGA_Preprocess_DNAmethylation, 72
TCGA_Preprocess_GeneExpression, 73
TCGA_Process_EstimateMissingValues, 75
TCGA_Select_Dataset, 75
test_gene_expr, 76
translateMethylMixResults, 77
validEpigenomes, 77