Package ‘MAGAR’

March 22, 2024

Type Package
Title MAGAR: R-package to compute methylation Quantitative Trait Loci (methQTL) from DNA methylation and genotyping data
Version 1.10.0
Date 2021-10-20
Description ```Methylation-Aware Genotype Association in R'' (MAGAR) computes methQTL from DNA methylation and genotyping data from matched samples. MAGAR uses a linear modeling strategy to call CpGs/SNPs that are methQTLs. MAGAR accounts for the local correlation structure of CpGs.
License GPL-3
Imports doParallel, igraph, bigstatsr, rjson, plyr, data.table, UpSetR, reshape2, jsonlite, methods, ff, argparse, impute, RnBeads.hg19, utils, stats
Depends R (>= 4.1), HDF5Array, RnBeads, snpStats, crlmm
Suggests gridExtra, VennDiagram, qqman, LOLA, RUnit, rmutil, rmarkdown, JASPAR2018, TFBSTools, seqLogo, knitr, devtools, BiocGenerics, BiocManager
Encoding UTF-8
RoxygenNote 7.1.1
VignetteBuilder knitr
biocViews Regression, Epigenetics, DNAMethylation, SNP, GeneticVariability, MethylationArray, Microarray, CpGIsland, MethylSeq, Sequencing, mRNAMicroarray, Preprocessing, CopyNumberVariation, TwoChannel, ImmunoOncology, DifferentialMethylation, BatchEffect, QualityControl, DataImport, Network, Clustering, GraphAndNetwork
BugReports https://github.com/MPIIComputationalEpigenetics/MAGAR/issues
URL https://github.com/MPIIComputationalEpigenetics/MAGAR
git_url https://git.bioconductor.org/packages/MAGAR
git_branch RELEASE_3_18
git_last_commit  b2f223a

R topics documented:

computeCorrelationBlocks  ........................................... 3
doImport  ................................................................. 4
doMethQTL ................................................................. 5
doMethQTL.Chromosome ................................................ 6
filterPval,MethQTLResult-method .................................. 7
getAnno,MethQTLInput-method ...................................... 8
getCorrelationBlocks,MethQTLResult-method .................... 9
getGeno,MethQTLInput-method ..................................... 10
getMethData,MethQTLInput-method ................................ 10
getResult,MethQTLResult-method ................................ 11
getResultGWASMap,MethQTLResult-method ..................... 11
getSamples,MethQTLInput-method ................................ 12
getOverlappingQTL .................................................... 12
getOverlapUniverse .................................................. 13
getPheno,MethQTLInput-method .................................... 13
getResult,MethQTLResult-method ................................ 14
getResultGWASMap,MethQTLResult-method ..................... 14
getSamples,MethQTLInput-method ................................ 15
getSpecificQTL .......................................................... 16
imputeMeth,MethQTLInput-method ................................ 16
joinMethQTLResult ..................................................... 17
loadMethQTLInput ...................................................... 18
loadMethQTLResult ..................................................... 18
MethQTLInput-class ................................................... 19
MethQTLResult-class .................................................. 20
overlapInputs ........................................................... 21
overlapQTL ............................................................... 21
QTL.OPTIONS ............................................................. 22
qtlAnnotationEnrichment ............................................ 23
qtlBaseSubstitutionEnrichment ................................... 24
qtlDistanceScatterplot .............................................. 25
qtlGetOption ............................................................ 25
qtlJSON2options ....................................................... 26
qtlManhattanPlot ...................................................... 27
qtlOptions2JSON ........................................................ 28
qtlPlotBaseSubstitution ............................................. 28
qtlPlotClusterSize ..................................................... 29
qtlPlotSNPCpGInteraction .......................................... 30
qtlSetOption ............................................................ 31
qtlTFBSMotifEnrichment ............................................. 34
computeCorrelationBlocks

This function computes CpG correlation blocks from correlations of CpGs across samples by Louvian clustering.

Usage

```r
computeCorrelationBlocks(
  meth.data,
  annotation,
  cor.threshold = qtlGetOption("cluster.cor.threshold"),
  sd.gauss = qtlGetOption("standard.deviation.gauss"),
  absolute.cutoff = qtlGetOption("absolute.distance.cutoff"),
  max.cpgs = qtlGetOption("max.cpgs"),
  assembly = "hg19",
  chromosome = "chr1"
)
```

Arguments

- `meth.data`: A `data.frame` containing the methylation data with CpGs in the rows and samples in the columns.
- `annotation`: The genomic annotations of the CpG positions.
- `cor.threshold`: The correlation threshold used to discard edges from the correlation-based network.
- `sd.gauss`: Standard deviation of the Gauss distribution used to weight the distance.
- `absolute.cutoff`: Absolute distance cutoff after which no methQTL interaction is to be considered.
- `max.cpgs`: Maximum number of CpGs used in the computation (used to save memory). 40,000 is a reasonable default for machines with ~128GB of main memory. Should be smaller for smaller machines and larger for larger ones.
- `assembly`: The assembly used.
- `chromosome`: The chromosome for which correlation block calling is to be performed.
doImport

details

This method performs clustering of the correlation matrix obtained from the DNA methylation matrix. Correlations are computed for each pair of CpGs across all the samples. We then compute a similarity matrix from this correlation matrix and set correlations lower than the given threshold to 0. In the next step, we weight the correlations by the distance between the CpGs: smaller distances get higher weights according to Gaussian distribution with mean 0 and standard deviation as specified above. Furthermore, similarities of CpGs that are further than absolute.distance.cutoff away from one another are discarded.

We then compute the associated weighted, undirected graph from the similarity matrix and execute Louvain clustering on the graph. The resulting clusters of CpGs are returned.

value

A list representing the clustering of CpGs into correlation blocks. Each element is a cluster, which contains row indices of the DNA methylation matrix that correspond to this cluster.

author(s)

Michael Scherer

examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.data <- getMethData(meth.qtl)
anno.meth <- getAnno(meth.qtl,"meth")
cor.blocks <- computeCorrelationBlocks(meth.data[seq(1,10),],annotation=anno.meth[seq(1,10),])

Description

Performs input for the given DNA methylation and genotyping data.

Usage

doImport(
  data.location,
  s.anno = NULL,
  assembly.meth = "hg19",
  assembly.geno = "hg19",
  tab.sep = ",",
  s.id.col = "sample_id",
  out.folder = tempdir(),
  ...
)
Arguments

data.location  Named character vector specifying the data location. The names correspond to:
    idat.dir   Path to the DNA methylation data. Can be in the form of IDAT files or in any other format accepted by RnBeads
    geno.dir   Path to the genotyping data file directory, either containing PLINK files (i.e. with bed, bim and fam files), imputed, (dosage) files (i.e. with dos and txt files), or idat files
s.anno        Path to the sample annotation sheet. If NULL, the program searches for potential sample annotation sheets in the data location directories.
assembly.meth Assembly used for the DNA methylation data. Typically is "hg19" for Illumina BeadArray data.
assembly.geno Assembly used for the genotyping data. If it is not the same as assembly.geno, the positions will be matched using liftOver.
tab.sep       The table separator used for the sample annotation sheet.
s.id.col      The column name of the sample annotation sheet that specifies the sample identifier.
out.folder    The output directory to store diagnostic plots
...           Further parameters passed to e.g. doGenoImport

Details

Import of DNA methylation and genotyping data is done separately:

DNA methylation data  DNA methylation data is imported using the RnBeads package. We use a default option setting commonly used for DNA methylation data obtained from the Illumina BeadArray series. If you want to specify further options, we refer to the rnb.options.

Genotyping data  Genotyping data is processed using PLINK. We focus on genotyping data generated with the Illumina BeadArray series and use default options. For further option settings, you should consult the qtlSetOption documentation.

You can specify the import type using qtlSetOption through the options meth.data.type and geno.data.type.

This function internally uses doMethImport and doGenoImport

Value

An object of type MethQTLInput-class with the methylation and genotyping information added.

Author(s)

Michael Scherer

Examples

meth.qtl.in <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.qtl.in
doMethQTL

Description

Function to compute methQTL given DNA methylation and genotyping data.

Usage

doMethQTL(
  meth.qtl,
  sel.covariates = NULL,
  p.val.cutoff = 1e-05,
  ncores = 1,
  cluster.submit = FALSE,
  out.dir = getwd(),
  default.options = TRUE
)

Arguments

- **meth.qtl**: An object of type `MethQTLInput-class` on which methQTL computation is to be performed.
- **sel.covariates**: Covariates as column names of the sample annotation sheet stored in `meth.qtl` to be used for covariate adjustment.
- **p.val.cutoff**: The p-value cutoff used for methQTL calling.
- **ncores**: The number of cores used.
- **cluster.submit**: Flag indicating if jobs are to be distributed among a SGE compute cluster.
- **out.dir**: Output directory.
- **default.options**: Flag indicating if default options for `cluster.cor.threshold`, `standard.deviation.gauss`, and `absolute.distance.cutoff` should be loaded for the data set used. See the option settings in `inst/extdata`.

Details

The process is split into 4 steps:

1. First the two matrices are split according to the chromosomes.
2. We then compute correlations among the CpGs and compute CpG correlation blocks.
3. In each of the CpG correlation blocks, linear models according to the `linear.model.type` `qt1SetOption` with the CpG methylation state of the reference CpG specified by `representative.cpg.computation` as output and the SNP genotype state and all possible covariates as input are computed.
4. For each of the CpG correlation blocks, we report the p-value of the representative CpG.
Currently, if `qtlGetOption('cluster.architecture')=='sge'` the function does not return a `MethQTLResult` object, but `NULL`, since monitoring finished jobs is hard through SLURM. After the jobs are finished (checked using `squeue`), the results can be loaded from `out.dir` using `loadMethQTLResult`.

**Value**

An object of type `MethQTLResult-class` containing the called methQTL interactions.

**Author(s)**

Michael Scherer

**See Also**

`doMethQTLChromosome`

**Examples**

```r
meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.qtl.res <- doMethQTL(meth.qtl,p.val.cutoff=0.01)
```

---

**Description**

This functions computes the methQTL interactions for a single chromosome

**Usage**

```r
doMethQTLChromosome(
  meth.qtl, 
  chrom, 
  sel.covariates = NULL, 
  p.val.cutoff = 1e-05, 
  out.dir = NULL, 
  ncores = 1
)
```

**Arguments**

- **meth.qtl**: An Object of type `MethQTLInput-class`
- **chrom**: Character vector representing the chromosome to be investigated.
- **sel.covariates**: Covariates as column names of the sample annotation sheet stored in `meth.qtl` to be used for covariate adjustment.
- **p.val.cutoff**: The p-value used for methQTL calling
- **out.dir**: Optional argument specifying the output directory
- **ncores**: The number of cores to be used
Value

A data frame with seven columns:

- **CpGs** The CpG ID chosen to be the representative CpG in the methQTL
- **SNP** The SNP ID (as rsNNNNNN) involved in the methQTL
- **Beta** The coefficient estimate of the linear model
- **P.value** The p-value associated with the coefficient estimate
- **Chromosome** The chromosome name
- **Position.CpG** The genomic position of the CpG
- **Position.SNP** The genomic position of the SNP
- **Distance** The distance between the CpG and the SNP

Author(s)

Michael Scherer

See Also

dоМethQTL

Examples

```r
meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.qtl.res <- doMethQTLChromosome(meth.qtl,chrom="chr18",p.val.cutoff=0.01)
```

Description

This functions filters the methQTL results according to a given p-value cutoff

Usage

```r
## S4 method for signature 'MethQTLResult'
filterPval(object, p.val.cutoff = 0.01)
```

Arguments

- **object** The *MethQTLResult*-class object to be filtered
- **p.val.cutoff** The p-value cutoff to be employed

Value

The filtered *MethQTLResult*-class object
Author(s)

Michael Scherer

Examples

```r
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res <- filterPval(meth.qtl.res)
meth.qtl.res
```

Description

Returns genomic annotation information for the given dataset.

Usage

```r
## S4 method for signature 'MethQTLInput'
getAnno(object, type = "meth")

## S4 method for signature 'MethQTLResult'
getAnno(object, type = "meth")
```

Arguments

- **object**
  An object of class `MethQTLInput-class` or `MethQTLResult-class`
- **type**
  The type of annotation to be returned. Can either be 'meth' or 'geno' for methylation, and genotyping information, respectively.

Value

The genomic annotation as a data.frame.

Examples

```r
meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
head(getAnno(meth.qtl,"meth"))
head(getAnno(meth.qtl,"geno"))
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
head(getAnno(meth.qtl.res,"meth"))
head(getAnno(meth.qtl.res,"geno"))
```
getCorrelationBlocks, MethQTLResult-method

Description
Returns the correlation blocks defined for the given dataset

Usage
## S4 method for signature 'MethQTLResult'
getCorrelationBlocks(object)

Arguments
object An object of class MethQTLResult-class.

Value
A list object containing the correlation blocks.

Examples
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
head(getCorrelationBlocks(meth.qtl.res))

getGeno, MethQTLInput-method

Description
Returns genotyping information for the given dataset.

Usage
## S4 method for signature 'MethQTLInput'
getGeno(object, site = NULL, sample = NULL)

Arguments
object An object of class MethQTLInput-class.
site The sites to be selected either as a numeric or logical vector. If NULL all sites are returned.
sample The samples to be selected either as a numeric or logical vector. If NULL all samples are returned.
Value

The genotyping matrix either as a matrix of HDF5Matrix.

Examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
head(getGeno(meth.qtl))

getMethData,MethQTLInput-method

Description

Returns methylation information for the given dataset.

Usage

## S4 method for signature 'MethQTLInput'
getMethData(object, site = NULL, sample = NULL)

Arguments

object An object of class MethQTLInput-class.
site The sites to be selected either as a numeric or logical vector. If NULL all sites are returned.
sample The samples to be selected either as a numeric or logical vector. If NULL all samples are returned.

Value

The methylation matrix either as a matrix of HDF5Matrix.

Examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
head(getMethData(meth.qtl))
getOverlappingQTL

Description
This function merges the QTLs given and returns the methQTL table in a merged format.

Usage
getOverlappingQTL(meth.qtl.list, type = "SNP")

Arguments
- **meth.qtl.list**: A list of MethQTLResult-class objects to be merged
- **type**: The type of annotation to be overlapped. Needs to be 'SNP', 'CpG' or 'cor.block'

Value
A data.frame with the methQTL interactions and an additional column specifying where the interaction displayed has been found. This value is generated from the names() argument of meth.qtl.list.

Author(s)
Michael Scherer

Examples
```r
meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
res <- getOverlappingQTL(list(A=meth.qtl.res.1,B=meth.qtl.res.2),type="SNP")
```

getOverlapUniverse

Description
This function overlaps results from a list of MethQTLResults and returns the union of all the input data points used.

Usage
getOverlapUniverse(meth.qtl.res, type)
getPheno, MethQTLInput-method

Arguments

meth.qtl.res  An object of type MethQTLResult-class or a list of such objects
type  The type of annotation to be overlapped. Needs to be 'SNP', 'CpG' or 'cor.block'

Value

A list with two GRanges objects, one containing the overlapped set and the other the union of input data points as elements 'all.qtl' and 'all.input'

Author(s)

Michael Scherer

Examples

meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
res <- getOverlapUniverse(list(A=meth.qtl.res.1,B=meth.qtl.res.2),type="SNP")

getPheno

Description

Returns phenotypic information for the given dataset.

Usage

## S4 method for signature 'MethQTLInput'
getPheno(object)

Arguments

object  An object of class MethQTLInput-class.

Value

The phenotypic data either as a data.frame.

Examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
head(getPheno(meth.qtl))
**Description**

Returns the methQTL results stores in the object.

**Usage**

```r
## S4 method for signature 'MethQTLResult'
getResult(object, cor.blocks = NULL, na.rm = FALSE)
```

**Arguments**

- `object`: An of type `MethQTLResult-class`.
- `cor.blocks`: Correlation blocks as obtained using `getCorrelationBlocks`. Please note that the correlation blocks need to contain the CpG identifiers, so the `MethQTLInput-class` object needs to be provided to `getCorrelationBlocks`.
- `na.rm`: Flag indicating if rows containing NA values are to be removed from the result.

**Value**

The methQTL results as a `data.frame` with each row being a methQTL.

**Examples**

```r
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
head(getResult(meth.qtl.res))
```

---

**Description**

Returns the methQTL results in the format used as input to GWAS-map and stores in the object.

**Usage**

```r
## S4 method for signature 'MethQTLResult'
getResultGWASMap(object, meth.qtl)
```

**Value**

The methQTL results in the format used as input to GWAS-map.
Arguments

object      An object of type MethQTLResult-class.
meth.qtl    An object of type MethQTLInput-class containing further information about the QTLs

Value

The methQTL results as a data.frame with each row being a methQTL.

Examples

meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
head(getResultGWASMap(meth.qtl.res,meth.qtl))

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
getSamples(meth.qtl)

Description

Returns the samples of the given dataset.

Usage

## S4 method for signature 'MethQTLInput'
getSamples(object)

Arguments

object      An object of class MethQTLInput-class.

Value

The samples of the dataset as a character vector.

Examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
getSamples(meth.qtl)
getSpecificQTL

Description
This function returns the methQTL interactions specific for a result

Usage
getSpecificQTL(meth.qtl.res, meth.qtl.background, type = "SNP")

Arguments
- `meth.qtl.res` An object of type `MethQTLResult-class` for which specific QTLs are to be obtained.
- `meth.qtl.background` The background set as a list of `MethQTLResult-class` objects.
- `type` The type of annotation to be overlapped. Needs to be 'SNP', 'CpG' or 'cor.block'

Value
A data.frame of methQTL interactions sorted by the effect size.

Author(s)
Michael Scherer

Examples
meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
res <- getSpecificQTL(meth.qtl.res.1,list(A=meth.qtl.res.1,B=meth.qtl.res.2),type="SNP")

imputeMeth,MethQTLInput-method

Description
Replaces missing values in the DNA methylation data matrix by imputed values

Usage
## S4 method for signature 'MethQTLInput'
imputeMeth(object)
joinMethQTLResult

Arguments

object An object of class MethQTLInput-class.

Value

The object with imputed values.

Examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.qtl.imp <- imputeMeth(meth.qtl)

joinMethQTLResult

Description

This function combines a list of MethQTLResult-class objects.

Usage

joinMethQTLResult(obj.list)

Arguments

obj.list A list of MethQTLResult-class objects to be joined

Value

An object of type MethQTLResult-class containing the combined information

Author(s)

Michael Scherer

Examples

meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
meth.qtl.res <- joinMethQTLResult(list(meth.qtl.res.1,meth.qtl.res.2))
loadMethQTLInput

Description
This functions load a MethQTLInput-class object from disk.

Usage
loadMethQTLInput(path)

Arguments
path Path to the directory that has been created by saveMethQTLInput,MethQTLInput-method.

Value
The object of type MethQTLInput-class that has been stored on disk.

Author(s)
Michael Scherer

Examples
meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.qtl

loadMethQTLResult

Description
This functions load a MethQTLResult-class object from disk.

Usage
loadMethQTLResult(path)

Arguments
path Path to the directory that has been created by saveMethQTLResult,MethQTLResult-method.

Value
The object of type MethQTLResult-class that has been stored on disk.
MethQTLInput-class

Author(s)

Michael Scherer

Examples

meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res

Description

Class storing methQTL input data, such as DNA methylation and genotyping data, as well as sample metadata

Details

This class is the basis for computing methQTLs in the methQTL-package. It stores all the relevant information including methylation data and genotype data as a matrix or HDF5Matrix, the phenotypic data as a data frame and the genomic annotation of both the methylation sites and the SNP data.

Slots

meth.data The methylation data as a numeric matrix of beta values or as an object of type HDF5Matrix
geno.data The genotyping data as a numeric matrix of SNP genotypes (0=homozygote reference, 1=heterozygote, 2=homozygote alternative allele) or as an object of type HDF5Matrix
pheno.data Phenotypic data describing the samples used in the study. Matches the dimensions of both meth.data and geno.data
anno.meth Genomic annotation of the methylation sites as a data.frame. Has the same number of rows as meth.data.
anno.geno Genomic annotation of the SNPs as a data.frame. Has the same number of rows as geno.data.
samples The sample identifiers used both for meth.data and geno.data, and as the rownames of pheno.data.
assembly The genome assembly used.
platform The platform used to compute the methylation data.
disk.dump Flag indicating if the matrices are stored on disk rather than in memory.
imputed Flag indicating if genotype dataset has been imputed.
**MethQTLResult-class**

**Methods**

- **getMeth** Returns the methylation matrix.
- **getGeno** Returns the genotyping matrix.
- **getPheno** Returns the phenotypic information.
- **getAnno** Returns the genomic annotation.
- **saveMethQTLInput** Stores the object on disk.
- **imputeMeth** Imputes the DNA methylation data matrix

**Author(s)**

Michael Scherer

---

**Description**

Class storing methQTL analysis results and the associated genomic annotations

**Details**

This class stores the results of the methQTL analysis. It stores a `data.frame` with the methQTL results, and associated genomic annotations for both the methylation sites and SNPs.

**Slots**

- **result.frame** The methQTL results as a `data.frame`
- **anno.meth** Genomic annotation of the methylation sites as a `data.frame`.
- **anno.geno** Genomic annotation of the SNPs as a `data.frame`.
- **correlation.blocks** Correlation blocks determined from the methylation matrix.
- **method** The method used to call methQTL.
- **rep.type** Method used to determine representative CpGs from correlation blocks.
- **chr** Optional argument specifying if methQTL were called on a single chromosome.

**Methods**

- **getResult** Returns the methQTL results.
- **getAnno** Returns the genomic annotation.

**Author(s)**

Michael Scherer
**overlapInputs**

**Description**
Overlaps the input annotations

**Usage**
overlapInputs(meth.qtl.list, type)

**Arguments**
- **meth.qtl.list**: A list of *MethQTLInput-class* or *MethQTLResult-class* objects to be overlapped
- **type**: The type of annotation to be overlapped. Needs to be 'SNP', 'CpG' or 'cor.block'

**Value**
A data frame containing the annotations of the unique input values.

**Author(s)**
Michael Scherer

**Examples**
```r
meth.qtl.1 <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
meth.qtl.2 <- meth.qtl.1
res <- overlapInputs(list(A=meth.qtl.1,B=meth.qtl.2),type="SNP")
```

**overlapQTLs**

**Description**
This function overlaps a list of methQTLs to determine which interactions are common.

**Usage**
overlapQTLs(meth.qtl.result.list, type)
Arguments

meth.qtl.result.list
A named list with each entry being an object of type MethQTLResult-class. The names are used in the visualization.

type
Determines if either the SNP (default), the CpG, or the correlation block 'cor.block' is to be visualized

Value

A list with length(meth.qtl.result.list) elements, containing IDs of methQTL interactions according to the option type.

Author(s)

Michael Scherer

Examples

meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
res <- overlapQTLs(list(A=meth.qtl.res.1,B=meth.qtl.res.2),type="SNP")

Description

This file contains code to generate the options of the methQTL package.

Usage

QTL.OPTIONS

Format

An object of class environment of length 36.
qtlAnnotationEnrichment

Description

This function performs enrichment analysis using the Fisher’s test for the methQTLs detected with respect to different genomic annotations.

Usage

qtlAnnotationEnrichment(meth.qtl.res, type = "SNP", annotation = "cpgislands")

Arguments

- `meth.qtl.res`: An object of type `MethQTLResult-class` or a list of such objects.
- `type`: The type of methQTL to be visualized. Can be either 'SNP', 'CpG', or 'cor.block'.
- `annotation`: The genomic annotation to be used. Can be the ones available in `rnb.region.types` or "ctcf", "distal", "dnase", "proximal", "tfbs", "tss"

Details

We use all data points that have been used to calculate methQTLs as the background and compare the overlaps with the annotation of interest in comparison to the methQTLs that have been computed in case a `MethQTLResult-class` is provided. If a list of `MethQTLResult-class` objects is provided, the intersection between the methQTLs from all objects in the list is compared with the union of all interactions that have been tested.

Value

A list of two p-values named 'enrichment' for overrepresentation and 'depletion' for underrepresentation

Author(s)

Michael Scherer

Examples

```r
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
res <- qtlAnnotationEnrichment(meth.qtl.res)
```
Description

This function tests for enrichment of a specific base substitution in the methQTL interactions.

Usage

qtlBaseSubstitutionEnrichment(meth.qtl.res, merge = FALSE)

Arguments

meth.qtl.res  An object of type MethQTLResult-class or a list of such objects.
merge  Flag indicating if 5’ and 3’ substitutions are to be merged or to be analyzed separately.

Details

The names of the list are e.g. A\textsubscript{G}, which refers to a replacement of the reference base A with an A. Enrichment is computed using Fisher’s exact test, using all SNP that have been used as input as the background.

Value

A list with one element for each potential base substitution containing the enrichment p-value.

Author(s)

Michael Scherer

Examples

meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
res <- qtlBaseSubstitutionEnrichment(meth.qtl.res)
**qtlDistanceScatterplot**

*Description*

Computes a scatterplot between CpG-SNP distance with both effect size and p-value

*Usage*

```r
qtlDistanceScatterplot(meth.qtl.result, out.dir = NULL, out.name = NULL)
```

*Arguments*

- `meth.qtl.result`: An object of type `MethQTLResult-class` containing called methQTL
- `out.dir`: If specified, the plot is stored as a pdf in this directory
- `out.name`: Optional name for the resulting plot

*Value*

An object of type `ggplot` comparing the distance between CpG and SNP. Negative values indicate that the SNP is downstream of the CpG.

*Author(s)*

Michael Scherer

*Examples*

```r
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
qtlDistanceScatterplot(meth.qtl.res)
```

**qtlGetOption**

*Description*

`qtlGetOption` Print the value of the global option

*Usage*

```r
qtlGetOption(names)
```
Arguments

names string or character vector containing the names of the options to be printed. All options are listed in \texttt{qtlSetOption}

Value

the option for the specified option

Author(s)

Michael Scherer

Examples

\begin{verbatim}
qtlGetOption("cluster.cor.threshold")
\end{verbatim}

\begin{verbatim}
qtlJSON2options(path)
\end{verbatim}

\begin{verbatim}
qtlJSON2options(system.file("extdata","qtl_options_probesEPIC.json",package="MAGAR"))
\end{verbatim}

Description

This function reads an option setting from a JSON file and applies them to the current session

Usage

\begin{verbatim}
qtlJSON2options(path)
\end{verbatim}

Arguments

path Path to a JSON file containing the options to be specified

Value

None

Author(s)

Michael Scherer

Examples

\begin{verbatim}
qtlJSON2options(system.file("extdata","qtl_options_probesEPIC.json",package="MAGAR"))
\end{verbatim}
Description

This function creates a manhattan plot for the given methQTL result.

Usage

$qtlManhattanPlot(meth.qtl.result, type = "CpG", stat = "p.val.adj.fdr")$

Arguments

- **meth.qtl.result**: An object of type `MethQTLResult-class` containing the methQTL
- **type**: Determines if either the CpG (default) or the SNP is to be visualized
- **stat**: Determines the statistic that is to be visualized. Can be either `P.value`, `Beta` or `p.val.adj.fdr`

Details

A plot is shown that contains chromosome-wise interactions.

Value

None

Author(s)

Michael Scherer

Examples

```r
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
qtlManhattanPlot(meth.qtl.res)
```
**qtlOptions2JSON**

Description

This function stores the current options setting as a JSON file at the specified path.

Usage

```r
qtlOptions2JSON(path = file.path(getwd(), "methQTL_options.json"))
```

Arguments

- `path`: A filename, to which the option setting is to be saved.

Value

None

Author(s)

Michael Scherer

Examples

```r
qtlSetOption('cluster.cor.threshold'=0.5)
qtlOptions2JSON("my_opts.json")
qtlJSON2options("my_opts.json")
```

---

**qtlPlotBaseSubstitution**

Description

This function returns an enrichment plot for the different base substitutions.

Usage

```r
qtlPlotBaseSubstitution(meth.qtl.res, ...)
```

Arguments

- `meth.qtl.res`: An object of type `MethQTLResult-class` or a list of such objects.
- `...`: Further parameters passed to `qtlBaseSubstitutionEnrichment`.
Value

None

Author(s)

Michael Scherer

See Also

qtlBaseSubstitutionEnrichment

Examples

meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
qtlPlotBaseSubstitution(meth.qtl.res)

qtlPlotClusterSize

Description

This functions returns a histogram comprising the (genomic) sizes of the correlation blocks in the given objet.

Usage

qtlPlotClusterSize(meth.qtl.res, type = "count")

Arguments

meth.qtl.res An object of type MethQTLResult-class
type Either "genomic" or "count", for genomic size of the correlation block in base pairs or as the number of CpGs

Value

An object of type ggplot containing the histogram as a plot

Author(s)

Michael Scherer

Examples

meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
qtlPlotClusterSize(meth.qtl.res)
qtlPlotSNPCpGInteraction

Description

Compares the methylation states of a given CpG for the genotype states available at the given SNP.

Usage

qtlPlotSNPCpGInteraction(
  meth.qtl,  
cpg = NULL,  
snp = NULL,  
out.dir = NULL,  
meth.qtl.res = NULL,  
out.name = NULL
)

Arguments

meth.qtl An object of type MethQTLInput-class containing the methylation and genotype information for the given CpG and the given SNP.
cpg The CpG identifier as a character (e.g. cg12345678).
snp The SNP identifier as a character (e.g. rs12345678).
out.dir If specified, the plot is stored as a pdf in this directory.
meth.qtl.res An optional argument of type MethQTLResult-class containing information on the results. If either cpg or snp are NULL, this function sorts the results by increasing p-value and uses the best results for plotting.
out.name Optional name for the resulting plot.

Value

An object of type ggplot comparing the CpG methylation states as boxplots across the different genotype states.

Author(s)

Michael Scherer

Examples

meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
qtlPlotSNPCpGInteraction(meth.qtl,cpg="cg19565884",snp="rs149871695")
qtlSetOption

Description

Change global options for methQTL calculation

Usage

qtlSetOption(
  rnbeads.options = NULL,
  meth.data.type = "idat.dir",
  geno.data.type = "plink",
  rnbeads.report = "temp",
  rnbeads.qc = FALSE,
  hdf5dump = FALSE,
  hardy.weinberg.p = 0.001,
  db.snp.ref = NULL,
  minor.allele.frequency = 0.05,
  missing.values.samples = 0.05,
  plink.geno = 0.1,
  impute.geno.data = FALSE,
  n.prin.comp = NULL,
  plink.path = NULL,
  fast.qtl.path = NULL,
  bgzip.path = NULL,
  tabix.path = NULL,
  correlation.type = "pearson",
  cluster.cor.threshold = 0.25,
  standard.deviation.gauss = 250,
  absolute.distance.cutoff = 5e+05,
  linear.model.type = "classial.linear",
  representative.cpg.computation = "row.medians",
  meth.qtl.type = "oneVSall",
  max.cpgs = 40000,
  cluster.architecture = "sge",
  cluster.config = c(h_vmem = "5G", mem_free = "5G"),
  n.permutations = 1000,
  compute.cor.blocks = TRUE,
  recode.allele.frequencies = FALSE,
  vcftools.path = NULL,
  imputation.user.token = NULL,
  imputation.reference.panel = "apps@hrc-r1.1",
  imputation.phasing.method = "shapeit",
  imputation.population = "eur"
)
Arguments

- **rnbeads.options**
  Path to an XML file specifying the RnBeads options used for data import. The default options are suitable for Illumina Beads Array data sets.

- **meth.data.type**
  Type of DNA methylation data used. Choices are listed in `rnb.execute.import`.

- **geno.data.type**
  The type of data to be imported. Can be either 'plink' for '.bed', '.bim', and '.fam' or '.dos' and 'txt' files or 'idat' for raw IDAT files.

- **rnbeads.report**
  Path to an existing directory, in which the preprocessing report of RnBeads is to be stored. Defaults to the temporary file.

- **rnbeads.qc**
  Flag indicating if the quality control module of RnBeads is to be executed.

- **hdf5dump**
  Flag indicating, if large matrices are to be stored on disk rather than in main memory using the HDF5Array package.

- **hardy.weinberg.p**
  P-value used for the markers to be excluded if they do not follow the Hardy-Weinberg equilibrium as implemented in PLINK.

- **db.snp.ref**
  Path to a locally stored version of dbSNP[3]. If this option is specified, the reference allele is determined from this file instead of from the allele frequencies of the dataset. This circumvents problems with some imputation methods. If NULL(default), recoding will not be performed.

- **minor.allele.frequency**
  Threshold for the minor allele frequency of the SNPs to be used in the analysis.

- **missing.values.samples**
  Threshold specifying how much missing values per SNP are allowed across the samples to be included in the analysis.

- **plink.geno**
  Threshold for missing values per SNP

- **impute.geno.data**
  Flag indicating if imputation of genotyping data is to be performed using the Michigan imputation server (https://imputationserver.sph.umich.edu/index.html)[2].

- **n.prin.comp**
  Number of principal components of the genetic data to be used as covariates in the methQTL calling. NULL means that no adjustment is conducted.

- **plink.path**
  Path to an installation of PLINK (also comes with the package)

- **fast.qtl.path**
  Path to an installation of fastQTL (comes with the package for Linux)

- **bgzip.path**
  Path to an installation of BGZIP (comes with the package for Linux)

- **tabix.path**
  Path to an installation of TABIX (comes with the package for Linux)

- **correlation.type**
  The type of correlation to be used. Please note that for type='pearson' (default) the more efficient implementation of correlation in the bigstatsr is used. Further available options are 'spearman' and 'kendall'.

- **cluster.cor.threshold**
  Threshold for CpG methylation state correlation to be considered as connected in the distance graph used to compute the correlation clustering.

- **standard.deviation.gauss**
  Standard deviation of the Gauss distribution used to weight the correlation according to its distance.
absolute.distance.cutoff
Distance cutoff after which a CpG correlation is not considered anymore.

linear.model.type
Linear model type to be used. Can be either "categorical.anova" or "classical.linear". If 'meth.qtl.type'='fastQTL', this option is automatically set to 'fastQTL'. See callMethQTLBlock for more informations.

representative.cpg.computation
Option specifying how reference CpGs per correlation block are to be computed. Available options are "row.medians" for the site that is the row median across the samples within the correlation block (for ties a random selection is performed), "mean.center" for an artificial site in the geometric center of the block with the average methylation level or "best.all" for the CpG with the best p-value across all of the CpGs in the correlation block.

meth.qtl.type
Option specifying how a methQTL interaction is computed. Since the package is based on correlation blocks, a single correlation block can be associated with either one SNP (meth.qtl.type='oneVSall'), with multiple SNPs (meth.qtl.type='allVSall'), or each correlation block can be positively and once negatively correlated with a SNP genotype (meth.qtl.type='twoVSall'). Additionally, we provide the option to use (FastQTL)[1] as a methQTL mapping tool (option 'fastQTL').

max.cpgs
Maximum number of CpGs used in the computation (used to save memory). 40,000 is a reasonable default for machines with ~128GB of main memory. Should be smaller for smaller machines and larger for larger ones.

cluster.architecture
The type of HPC cluster architecture present. Currently supported are 'sge' and 'slurm'.

cluster.config
Resource parameters needed to setup an SGE or SLURM cluster job. Includes \texttt{h_vmem} and \texttt{mem_free} for SGE and \texttt{clock.limit} and \texttt{mem.size} for SLURM. An example configuration for SLURM would be \texttt{c("clock.limit"="1-0","mem.size"="10G")} for 1 day of running time (format days:hours) and 10 GB of maximum memory usage. Additionally, \texttt{\textbf{n.cpus}} can be specified as the SLURM option \texttt{\textbf{cpus-per-task}}

n.permutations
The number of permutations used to correct the p-values for multiple testing. See (http://fastqtl.sourceforge.net/) for further information.

compute.cor.blocks
Flag indicating if correlation blocks are to be called. If FALSE, each CpG is considered separately.

recode.allele.frequencies
Flag indicating if the reference allele is to be redefined according to the frequencies found in the cohort investigated.

vcftools.path
Path to the installation of VCFtools. Necessary is the vcf-sort function in this folder.

imputation.user.token
The user token that is required for authorization with the Michigan imputation server. Please have a look at https://imputationserver.sph.umich.edu, create a user account and request a user token for access in your user profile.
The reference panel used for imputation. Please see https://imputationserver.readthedocs.io/en/latest/reference-panels/ for further information which panels are supported by the Michigan imputation server.


Value

None

Author(s)

Michael Scherer

References


Examples

```r
qtlGetOption("rnbeads.report")
qtlSetOption(rnbeads.report=getwd())
qtlGetOption("rnbeads.report")
```

Description

This function performs TFBS enrichment analysis for the methQTL SNPs/CpGs detected and returns overrepresented binding motifs.
Usage

```r
qtlTFBSMotifEnrichment(
  meth.qtl.res,
  type = "SNP",
  size = 500,
  assembly = "hg19",
  subsample = 1e+05,
  out.dir = getwd(),
  ...
)
```

Arguments

- `meth.qtl.res`: An object of type `MethQTLResult-class` or a list of such objects.
- `type`: The type of methQTL to be visualized. Can be either 'SNP', 'CpG', or 'cor.block'.
- `size`: Motif enrichment is only supported for genomic regions. Therefore, we resize the individual methQTL to genomic regions using a width of this size around the site of interest.
- `assembly`: The assembly used. Only "hg19" and "hg38" supported.
- `subsample`: Integer specifying how many of the regions are to be subsamples from the universe.
- `out.dir`: The output directory in which resulting plots will be stored.
- `...`: Further parameters passed to `findMotifFgBg`.

Details

This function is in part based on the tutorial for Motif discovery in https://compgenomr.github.io/book/motif-discovery.html. We use all data points that have been used to calculate methQTLs as the background and compare the overlaps with the annotation of interest in comparison to the methQTLs that have been computed in case a `MethQTLResult-class` is provided. If a list of `MethQTLResult-class` objects is provided, the intersection between the methQTLs from all objects in the list is compared with the union of all interactions that have been tested.

Value

A plot describing the TFB motif enrichment

Author(s)

Michael Scherer

Examples

```r
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
res <- qtlTFBSMotifEnrichment(meth.qtl.res)
```
Description

This function creates an UpSet plot from the given methQTL results.

Usage

```r
qtlUpsetPlot(meth.qtl.result.list, type = "SNP", ...)
```

Arguments

- `meth.qtl.result.list`: A named list with each entry being an object of type `MethQTLResult-class`. The names are used in the visualization.
- `type`: Determines if either the SNP (default), the CpG, or the correlation block 'cor.block' is to be visualized.
- `...`: Further argument passed to `upset`.

Details

The plot is directly drawn and can be stored on disk using the known R graphic devices.

Value

None

Author(s)

Michael Scherer

Examples

```r
meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata", "MethQTLResult_chr18", package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
qtlUpsetPlot(list(A=meth.qtl.res.1,B=meth.qtl.res.2))
```
**qtlUpSetPlotCorBlocks**

Description

This function overlaps correlation blocks for a list of methQTL results.

Usage

```r
qtlUpSetPlotCorBlocks(meth.qtl.res.list, ...)
```

Arguments

- `meth.qtl.res.list`: A list of `MethQTLResult-class` objects, for which correlation blocks are to be overlapped.
- `...`: Further argument passed to `upset`.

Details

This function draws an UpSetPlot for the overlaps directly from the open graphics device.

Value

None

Author(s)

Michael Scherer

Examples

```r
meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
qtlUpSetPlotCorBlocks(list(A=meth.qtl.res.1,B=meth.qtl.res.2))
```

**qtlUpSetPlotTagCpGs**

Description

This function overlaps the tagCpGs for a list of methQTL results.

Usage

```r
qtlUpSetPlotTagCpGs(meth.qtl.res.list, ...)
```
Arguments

meth.qtl.res.list
    A list of MethQTLResult-class objects, for which correlation blocks are to be overlapped

... Further argument passed to upset

Details

This function draws an UpSetPlot for the overlaps directly from to the open graphics device

Value

None

Author(s)

Michael Scherer

Examples

meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
qtlUpSetPlotTagCpGs(list(A=meth.qtl.res.1,B=meth.qtl.res.2))

Description

This function creates a venn plot from a list of methQTL results, showing the overlap between the interactions

Usage

qtlVennPlot(
    meth.qtl.result.list,  # MethQTLResult-class objects
    out.folder,            # folder to store the plots
    type = "SNP",         # type of interactions
    out.name = NULL,      # name of the output file
    ...                   # further arguments
  )
saveMethQTLInput.MethQTLInput-method

Arguments

meth.qtl.result.list
A named list with each entry being an object of type MethQTLResult-class. The names are used in the visualization.

cor.block
Determines if either the SNP (default), the CpG, or the correlation block 'cor.block' is to be visualized.

out.name
Optional argument for the name of the plot on disk (ending needs to be .png)

Details

The plot can be stored on disk using out.folder and out.name

Value

None

Author(s)

Michael Scherer

Examples

meth.qtl.res.1 <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
meth.qtl.res.2 <- meth.qtl.res.1
qtlVennPlot(list(A=meth.qtl.res.1,B=meth.qtl.res.2),out.folder=getwd())

saveMethQTLInput.MethQTLInput-method

saveMethQTLInput

Description

This functions stores a MethQTLInput object in disk.

Usage

## S4 method for signature 'MethQTLInput'
saveMethQTLInput(object, path)

Arguments

object
The MethQTLInput-class object to be stored on disk.

path
A path to a non-existing directory for files to be stored.
Value
None

Author(s)
Michael Scherer

Examples
meth.qtl <- loadMethQTLInput(system.file("extdata","reduced_methQTL",package="MAGAR"))
saveMethQTLInput(meth.qtl,"MethQTLInput")

description
This function stores a MethQTLInput object in disk.

Usage
## S4 method for signature 'MethQTLResult'
saveMethQTLResult(object, path)

Arguments

  object  The MethQTLResult-class object to be stored on disk.
  path    A path to a non-existing directory for files to be stored.

Value
None

Author(s)
Michael Scherer

Examples
meth.qtl.res <- loadMethQTLResult(system.file("extdata","MethQTLResult_chr18",package="MAGAR"))
saveMethQTLResult(meth.qtl.res,"MethQTLResult")
Index

* datasets
  QTL.OPTIONS, 22
computeCorrelationBlocks, 3
doImport, 4
doMethQTL, 6
doMethQTLChromosome, 7
filterPval
  (filterPval,MethQTLResult-method), 8
filterPval,MethQTLResult-method, 8
getAnno, 20
getAnno (getAnno,MethQTLInput-method), 9
getAnno,MethQTL-method
  (getAnno,MethQTLInput-method), 9
getAnno,MethQTLInput-method, 9
getAnno,MethQTLResult-method
  (getAnno,MethQTLInput-method), 9
getCorrelationBlocks
  (getCorrelationBlocks,MethQTLResult-method), 10
getCorrelationBlocks,MethQTLResult-method, 10
getGeno, 20
getGeno (getGeno,MethQTLInput-method), 10
getGeno,MethQTL-method
  (getGeno,MethQTLInput-method), 10
getGeno,MethQTLInput-method, 10
getMeth, 20
getMethData
  (getMethData,MethQTLInput-method), 11
gethData,methQTL-method
  (getMethData,MethQTLInput-method), 11
gethData,MethQTLInput-method, 11
getOverlappingQTL, 12
getOverlapUniverse, 12
getPheno, 20
getPheno
  (getPheno,MethQTLInput-method), 13
getPheno,methQTL-method
  (getPheno,MethQTLInput-method), 13
getPheno,MethQTLInput-method, 13
getResult, 20
getResult
  (getResult,MethQTLResult-method), 14
getResult,MethQTLResult-method, 14
getResultGWASMap
  (getResultGWASMap,MethQTLResult-method), 14
getResultGWASMap,MethQTLResult-method, 14
getSamples
  (getSamples,MethQTLInput-method), 15
getSamples,methQTL-method
  (getSamples,MethQTLInput-method), 15
getSamples,MethQTLInput-method, 15
getSpecificQTL, 16
HDF5Array, 32
HDF5Matrix, 11, 19
imputeMeth, 20
imputeMeth
  (imputeMeth,MethQTLInput-method), 16
imputeMeth, methQTL-method
   (imputeMeth, MethQTLInput-method), 16
imputeMeth, MethQTLInput-method, 16
joinMethQTLResult, 17
loadMethQTLInput, 18
loadMethQTLResult, 18, 19
MethQTLInput-class, 19
MethQTLResult-class, 20

overlapInputs, 21
overlapQTLs, 21

QTL.OPTIONS, 22
qtlAnnotationEnrichment, 23
qtlBaseSubstitutionEnrichment, 24, 28
qtlDistanceScatterplot, 25
qtlGetOption, 25
qtlJSON2options, 26
qtlManhattanPlot, 27
qtlOptions2JSON, 28
qtlPlotBaseSubstitution, 28
qtlPlotClusterSize, 29
qtlPlotSNPCpGInteraction, 30
qtlSetOption, 5, 6, 26, 31
qtlTFBSMotifEnrichment, 34
qtlUpsetPlot, 36
qtlUpSetPlotCorBlocks, 37
qtlUpSetPlotTagCpGs, 37
qtlVennPlot, 38

rnb.execute.import, 32
rnb.options, 5
rnb.region.types, 23
RnBeads, 5

saveMethQTLInput, 20
saveMethQTLInput
   (saveMethQTLInput, MethQTLInput-method), 39
saveMethQTLInput, methQTL-method
   (saveMethQTLInput, MethQTLInput-method), 39
saveMethQTLInput, MethQTLInput-method, 39

upset, 36–38
venn.diagram, 39