Package 'MEAL'

October 14, 2021

Title Perform methylation analysis

Version 1.22.0

Maintainer Xavier Escribà Montagut <xavier.escriba@isglobal.org>

Description Package to integrate methylation and expression data. It can also perform methylation or expression analysis alone. Several plotting functionalities are included as well as a new region analysis based on redundancy analysis. Effect of SNPs on a region can also be estimated.

Depends R (>= 3.6.0), Biobase, MultiDataSet

License Artistic-2.0

biocViews DNAMethylation, Microarray, Software, WholeGenome

LazyData true

Imports GenomicRanges, limma, vegan, BiocGenerics, minfi, IRanges, S4Vectors, methods, parallel, ggplot2 (>= 2.0.0), permute, Gviz, missMethyl, isva, SummarizedExperiment, SmartSVA, graphics, stats, utils, matrixStats

Suggests testthat, IlluminaHumanMethylationEPICanno.ilm10b2.hg19, IlluminaHumanMethylation450kanno.ilmn12.hg19, knitr, minfiData, BiocStyle, rmarkdown, brgedata

VignetteBuilder knitr

RoxygenNote 7.1.1

Encoding UTF-8

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

git_branch RELEASE_3_13

git_last_commit bd59164

git_last_commit_date 2021-05-19

Date/Publication 2021-10-14

Author Carlos Ruiz-Arenas [aut, cre], Juan R. Gonzalez [aut]

R topics documented:

`computeRDAR2` *Compute signification of RDA test*

Description

Compare R2 obtained in our region of interest with the global R^2 and the R^2 of regions with the same number of probes.

Usage

```
computeRDAR2(
  fullMat,
  varsmodel,
  covarsmodel = NULL,
  featNum,
  R2,
  num_permutations = 1e+05 - 1
)
```

Arguments

fullMat	Matrix with the whole genome expression or methylation values
varsmodel	Matrix with the model
covarsmodel	Matrix with the covariables model
featNum	Numeric with the number of features of the RDA model
R2	Numeric with the R2 of the RDA model
num_permutations	Numeric with the number of permutations.

Value

Numeric vector with the probability of finding a region with the same number of probes with a bigger R2 and the global R2.

---

correlationMethExprs *Computes the correlation between methylation and expression*

Description

Estimates the correlation between methylation and expression. When there are known variables that affect methylation and/or expression, their effect can be substracted using a linear model and then the residuals are used.

Usage

```
correlationMethExprs(
  multiset,
  meth_set_name = NULL,
  exprs_set_name = NULL,
  vars_meth = NULL,
  vars_exprs = NULL,
  sel_cpgs,
  flank = 250000,
  betas = TRUE,
  num_cores = 1,
  verbose = TRUE
)
```

Arguments

multiset	MultiDataSet containing a methylation and an expression slots.
meth_set_name	Character vector with the name of the MultiDataSet's slot containing methylation data.
exprs_set_name	Character vector with the name of the MultiDataSet's slot containing expression data.

vars_meth	Character vector with the names of the variables that will be used to obtain the methylation residuals. By default, none is used and residuals are not computed.
vars_exprs	Character vector with the names of the variables that will be used to obtain the expression residuals. By default, none is used and residuals are not computed.
sel_cpgs	Character vector with the name of the CpGs used in the analysis. If empty, all the CpGs of the methylation set will be used.
flank	Numeric with the number of pair bases used to define the cpg-expression probe pairs.
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)
num_cores	Numeric with the number of cores to be used.
verbose	Logical value. If TRUE, it writes out some messages indicating progress. If FALSE nothing should be printed.

Details

For each cpg, a range is defined by the position of the cpg plus the flank parameter (upstream and downstream). Only those expression probes that are entirely in this range will be selected. For these reason, it is required that the ExpressionSet contains a featureData with the chromosome and the starting and ending positions of the probes.

Value

Data.frame with the results of the linear regression:

- cpg: Name of the cpg
- exprs: Name of the expression probe
- beta: coefficient of the methylation change
- se: standard error of the beta
- P.Value: p-value of the beta coefficient
- adj.P.Val: q-value computed using B&H

---

exportResults *Exports results data.frames to csv files.*

Description

Exports results to csv files. If more than one variable is present, subfolders with the name of the variable are created. For each variable, four files will be generated: probeResults.csv, dmrCateResults.csv, bumphunterResults.csv and blockFinderResults.csv

Usage

```
exportResults(
  object,
  dir = "./",
  prefix = NULL,
  fNames = c("chromosome", "start")
)
```

Arguments

object	ResultSet
dir	Character with the path to export.
prefix	Character with a prefix to be added to all file names.
fNames	Names of the columns of object fData that will be added to the results data.frame.

Value

Files are saved into the given folder.

Examples

```
if (require(minfiData)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
methyOneVar <- runPipeline(set, variable_names = "sex")
exportResults(methyOneVar)
}
```

---

filterResults *Filter the data.frame obtained from probe analysis*

Description

Filter the data.frame obtained from probe analysis

Usage

```
filterResults(results, range, position = "position", chr = "chromosome")
```

Arguments

results	Data.frame with the results of probe analysis
range	GenomicRanges with the desired range.
position	Character with the name of the column containing the positions
chr	Character with the name of the column containing the chromosome

Value

Data.frame with the results of the probes of the range

getGeneVals *Get all probes related to a gene*

Description

Given a ResultSet and a gene name returns the results of the analysis of all the probes of the gene.

Usage

```
getGeneVals(
  object,
  gene,
  rid = 1,
  genecol = "genes",
  fNames = c("chromosome", "start"),
  ...
)
```

Arguments

object	ResultSet
gene	Character with the name of the gene
rid	Name of the results: "DiffMean" for mean differences, "DiffVar" for variance differences. (Default: DiffMean)
genecol	Character with the column of object fData with the gene information
fNames	Names of the columns of object fData that will be added to the results data.frame.
...	Further arguments passed to getProbeResults

Value

data.frame with the results of the analysis of the probes belonging to the gene

Examples

```
## Not run:
if (require(minfiData)){
 set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
methyOneVar <- runPipeline(set, variable_names = "sex")
getGeneVals(methyOneVar, "TSPY4")
}

## End(Not run)
```

getProbeResults *Obtain probe results from a* ResultSet

Description

It computes the statistics from the MArrayLM computed with DiffMeanAnalysis or DiffVarAnalysis. This function allows to specify the contrasts and to get F-statistics for a group of variables.

Usage

```
getProbeResults(
  object,
  rid = "DiffMean",
  coef = 2,
  contrast = NULL,
  fNames = c("chromosome", "start"),
  robust = FALSE,
  ...
)
```

Arguments

object	ResultSet
rid	Name of the results: "DiffMean" for mean differences, "DiffVar" for variance differences. (Default: DiffMean)
coef	Number of the coefficient used to compute the statistics. If a vector is supplied, F-statistics evaluating the global effect of the coefficients are computed. (Default: 2).
contrast	Matrix of contrasts
fNames	Names of the columns of object fData that will be added to the results data.frame.
...	Further arguments passed to getAssociation.

Value

data.frame with the probe results.

`getRDAresults` *Get a summary of RDA results*

Description

Get statistics from RDA result.

Usage

```
getRDAresults(object)
```

Arguments

`object`	`ResultSet`

Value

Numeric vector with the RDA statistics

`MEAL` *MEAL (Methylation and Expression AnaLizer): Package for analysing methylation and expression data*

Description

MEAL is a package designed to facilitate the analysis methylation and expression data. The package can analyze one dataset and can find correlations between methylation and expression data. MEAL has a vignette that explains the main functionalities of the package.

`MEAL-defunct` *Defunct functions*

Description

These functions are defunct and no longer available.

Details

Defunct functions are: multiCorrMethExprs, DAPipeline, DAProbe, DARegion, RDAset, filterSet, plotBestFeatures, preparePhenotype, createRanges, prepareMethylationSet, calculateRelevantSNPs, correlationMethSNPs, explainedVariance, normalSNP, plotLM

Defunct classes are: analysisRegionResults, analysisResults

plotFeature *Plot values of a feature*

Description

Plot values of a feature splitted by one or two variables.

Usage

```
plotFeature(set, feat, variables = colnames(pheno)[1], betas = TRUE)
```

Arguments

set	ExpressionSet, GenomicRatioSet or SummarizedExperiment.
feat	Numeric with the index of the feature or character with its name.
variables	Character vector with the names of the variables to be used in the splitting. Two variables is the maximum allowed.
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)

Value

A plot is generated on the current graphics device.

Examples

```
if (require(minfiData)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
 plotFeature(set, 1, variables = "Sample_Group")
 }
```

plotRDA *Plot RDA results*

Description

Plot RDA results

Usage

```
plotRDA(object, pheno = data.frame(), n_feat = 5, main = "RDA plot", alpha = 1)
```

Arguments

object	ResultSet
pheno	data.frame with the variables used to color the samples.
n_feat	Numeric with the number of cpgs to be highlighted. Default: 5.
main	Character with the plot title.
alpha	Numeric with the alpha level for colour transparance. Default: 1; no transparency.

Value

A plot is generated on the current graphics device.

Examples

```
if (require(minfiData)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
model <- model.matrix(~set$sex)
rda <- runRDA(set, model)
plotRDA(rda, pheno = data.frame(factor(set$sex)))
}
```

---

plotRegion *Plot results in a genomic region*

Description

Plot the results from the different analyses of a ResultSet in a specific genomic region. It can plot all the results from runPipeline.

Usage

```
plotRegion(
  rset,
  range,
  results = names(rset),
  genome = "hg19",
  rset2,
  tPV = 5,
  fNames = c("chromosome", "start", "end"),
  fNames2 = c("chromosome", "start", "end")
)
```

Arguments

rset	ResultSet
range	GenomicRanges with the region coordinates
results	Character with the analyses that will be included in the plot. By default, all analyses available are included.
genome	String with the genome used to retrieve transcripts annotation: hg19, hg38, mm10. (Default: "hg19")
rset2	Additional ResultSet
tPV	Threshold for P-Value
fNames	Names from rset fData
fNames2	Names from rset2 fData

Details

This plot allows to have a quick summary of the methylation or gene expression analyses in a given region. If we use a ResultSet obtained from methylation data, transcripts annotation is obtained from archive. If we use a ResultSet obtained from gene expression data, transcripts annotation is taken from fData.

This plot can be used to plot the results of one dataset (methylation or gene expression) or to represent the association between methylation and gene expression data. If only one dataset is used, the p-values and the coefficients of DiffMean and DiffVar analyses are plotted. If we pass two ResultSets, rset should contain methylation results and a rset2 the gene expression results.

Value

Regional plot

---

runBlockFinder *Run blockFinder*

Description

Run blockFinder

Usage

```
runBlockFinder(
  set,
  model,
  coefficient = 2,
  blockfinder_cutoff = 0.1,
  num_permutations = 0,
  resultSet = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

set	GenomicRatioSet, eSet derived object or SummarizedExperiment
model	Model matrix or formula to get model matrix from set.
coefficient	Numeric with the column of model matrix used in the analysis. (Default: 2)
blockfinder_cutoff	Numeric with the minimum cutoff to include a probe in a block. (Default: 0.1)
num_permutations	Numeric with the number of permutations run to compute the blocks p-value. (Default: 0)
resultSet	Should results be encapsulated in a resultSet? (Default: TRUE)
verbose	Logical value. Should the function be verbose? (Default: FALSE)
...	Further arguments passed to blockFinder.

Details

This function has been deprecated and will be defunct in the new version.

Value

data.frame or resultSet with the result of blockFinder

See Also

blockFinder

---

runBumphunter *Run bumphunter*

Description

Run bumphunter

Usage

```
runBumphunter(
  set,
  model,
  coefficient = 2,
  bumphunter_cutoff = 0.1,
  num_permutations = 0,
  bumps_max = 30000,
  betas = TRUE,
  check_perms = FALSE,
  verbose = FALSE,
  resultSet = FALSE,
  ...
)
```

Arguments

set	GenomicRatioSet, eSet derived object or SummarizedExperiment
model	Model matrix or formula to get model matrix from set.
coefficient	Numeric with the column of model matrix used in the analysis. (Default: 2)
bumphunter_cutoff	Numeric with the minimum cutoff to include a probe in a block. (Default: 0.1)
num_permutations	Numeric with the number of permutations run to compute the bumps p-value. (Default: 0)
bumps_max	Numeric with the maximum number of bumps used in the permutation. This parameter only applies when num_permutations is greater than 0. (Default: 30000)
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)
check_perms	Logical. Should we check that there are less bumps than bumps_max? This parameter only applies when num_permutations is greater than 0. (Default: TRUE)
verbose	Logical value. Should the function be verbose? (Default: FALSE)
resultSet	Should results be encapsulated in a resultSet? (Default: TRUE)
...	Further arguments passed to bumphunter.

Details

This function has been deprecated and will be defunct in the new version.

Value

data.frame or resultSet with the result of bumphunter

See Also

bumphunter

---

runDiffMeanAnalysis *Run differential mean analysis*

Description

Run differential mean analysis using t-moderated statistics. This function relies on lmFit from limma package.

Usage

```
runDiffMeanAnalysis(
  set,
  model,
  weights = NULL,
  method = "ls",
  max_iterations = 100,
  betas = TRUE,
  resultSet = TRUE,
  warnings = TRUE
)
```

Arguments

set	Matrix, GenomicRatioSet, SummarizedExperiment or ExpressionSet.
model	Model matrix or formula to get model matrix from set.
weights	weights used in the lmFit model.
method	String indicating the method used in the regression: "ls" or "robust". (Default: "ls")
max_iterations	Numeric indicating the maximum number of iterations done in the robust method.
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)
resultSet	Should results be encapsulated in a resultSet? (Default: TRUE)
warnings	Should warnings be displayed? (Default:TRUE)

Value

MArrayLM or resultSet with the result of the differential mean analysis.

Examples

```
if (require(minfiData)){
 mvalues <- getM(MsetEx)[1:100, ]
 model <- model.matrix(~ Sample_Group, data = pData(MsetEx))
 res <- runDiffMeanAnalysis(mvalues, model, method = "ls")
 res
}
```

---

runDiffVarAnalysis *Run differential variance analysis*

Description

Run differential variance analysis. This analysis can only be run with categorical variables. This function relies on varFit from missMethyl package.

Usage

```
runDiffVarAnalysis(
  set,
  model,
  coefficient = NULL,
  resultSet = TRUE,
  betas = TRUE,
  warnings = TRUE,
  ...
)
```

Arguments

set	Matrix, GenomicRatioSet, SummarizedExperiment or ExpressionSet.
model	Model matrix or formula to get model matrix from set.
coefficient	Numeric with the coefficients used to make the groups. If NULL, all possible groups will be computed.
resultSet	Should results be encapsulated in a resultSet? (Default: TRUE)
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)
warnings	Should warnings be displayed? (Default:TRUE)
...	Further arguments passed to varFit.

Value

MArrayLM or resultSet with the result of the differential variance analysis.

Examples

```
if (require(minfiData)){
 mvalues <- getM(MsetEx)[1:100, ]
 model <- model.matrix(~ Sample_Group, data = pData(MsetEx))
 res <- runDiffVarAnalysis(mvalues, model)
 res
}
```

---

runDMRcate *Run DMRcate*

Description

Run DMRcate

Usage

```
runDMRcate(set, model, coefficient = 2, resultSet = FALSE, ...)
```

Arguments

set	GenomicRatioSet, eSet derived object or SummarizedExperiment
model	Model matrix or formula to get model matrix from set.
coefficient	Numeric with the column of model matrix used in the analysis. (Default: 2)
resultSet	Should results be encapsulated in a resultSet? (Default: TRUE)
...	Further arguments passed to cpg.annotate or dmrcate.

Details

This function has been deprecated and will be defunct in the new version.

Value

data.frame or resultSet with the result of bumphunter

See Also

dmrcate, cpg.annotate

---

runPipeline *Perform differential methylation analysis*

Description

Wrapper for analysing differential methylation and expression at region and probe level.

Usage

```
runPipeline(
  set,
  variable_names,
  covariable_names = NULL,
  model = NULL,
  weights = NULL,
  num_vars,
  sva = FALSE,
  betas = TRUE,
  range,
  analyses = c("DiffMean"),
  verbose = FALSE,
  warnings = TRUE,
  DiffMean_params = NULL,
  DiffVar_params = list(coefficient = 1:2),
  rda_params = NULL,
  method = "ls",
  big = FALSE
)
```

Arguments

set	GenomicRatioSet, eSet derived object or SummarizedExperiment
variable_names	Character vector with the names of the variables that will be returned as result.
covariable_names	Character vector with the names of the variables that will be used to adjust the model.
model	Model matrix or formula to get model matrix from set.
weights	weights used in the lmFit model (default NULL)
num_vars	Numeric with the number of variables in the matrix for which the analysis will be performed. Compulsory if equation is not null.
sva	Logical. Should Surrogate Variable Analysis be applied? (Default: FALSE)
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)
range	GenomicRanges with the region used for RDA
analyses	Vector with the names of the analysis to be run (DiffMean and/or DiffVar).
verbose	Logical value. If TRUE, it writes out some messages indicating progress. If FALSE nothing should be printed.
warnings	Should warnings be displayed? (Default:TRUE)
DiffMean_params	List with other parameter passed to runBumphunter function.
DiffVar_params	List with other parameter passed to runBumphunter function.
rda_params	List with other parameter passed to runRDA function.
method	String indicating the method used in the regression: "ls" or "robust". (Default: "ls")
big	Logical value indicating whether SmartSVA should be instead of SVA (TRUE recommended for methylation or when having large number of samples). Default is FALSE.

Details

This function is the main wrapper of the package. First, it simplifies the the set to only contain the common samples between phenotype and features. In addition, it allows to change the class of the variables and to apply genomic models (more information on preparePhenotype). Afterwards, analysis per probe and per region are done merging the results in an AnalysisResults object.

Default linear model will contain a sum of the variables and covariables. If interactions are desired, a costum formula can be specified. In that case, variables and covariables must also be specified in order to assure the proper work of the resulting AnalysisResult. In addition, the number of variables of the model for which the calculation will be done **must** be specified.

Value

ResultSet object

Examples

```
if (require(minfiData)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
 res <- runPipeline(set, variable_names = "Sample_Group")
 res
}
```

---

runRDA *Calculate RDA for a set*

Description

Perform RDA calculation for a AnalysisRegionResults. Feature values will be considered the matrix X and phenotypes the matrix Y. Adjusting for covariates is done using a model matrix passed in covarsmodel.

Usage

```
runRDA(
  set,
  model,
  num_vars = ncol(model),
  range,
  betas = FALSE,
  resultSet = TRUE,
  num_permutations = 10000,
  ...
)
```

Arguments

set	MethylationSet, ExpressionSet or matrix
model	Model matrix or formula to get model matrix from set.
num_vars	Numeric with the number of variables in the matrix for which the analysis will be performed. Compulsory if equation is not null.
range	GenomicRanges with the region used for RDA
betas	If set is a GenomicRatioSet, should beta values be used? (Default: TRUE)
resultSet	Should results be encapsulated in a resultSet? (Default: TRUE)
num_permutations	Numeric with the number of permutations run to compute the p-value. (Default: 1e4)
...	Further arguments passed to rda.

Value

Object of class rda or resultSet

See Also

rda

Examples

```
if (require(minfiData)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
model <- model.matrix(~set$age)
rda <- runRDA(set, model)
rda
}
```

`runRegionAnalysis` *Run different DMR detection methods*

Description

Run different DMR detection methods

Usage

```
runRegionAnalysis(
  set,
  model,
  methods = c("blockFinder", "bumphunter", "DMRcate"),
  coefficient = 2,
  bumphunter_params = NULL,
  blockFinder_params = NULL,
  dmrcate_params = NULL,
  verbose = FALSE,
  resultSet = TRUE
)
```

Arguments

`set`	`GenomicRatioSet`, `eSet` derived object or `SummarizedExperiment`
`model`	Model matrix representing a linear model.
`methods`	Character vector with the names of the methods used to estimate the regions. Valid names are: "blockFinder", "bumphunter" and "DMRcate".
`coefficient`	Numeric with the index of the model matrix used to perform the analysis.
`bumphunter_params`	List with other parameter passed to `runBumphunter` function.
`blockFinder_params`	List with other parameter passed to `runBlockFinder` function.
`dmrcate_params`	List with other parameter passed to `runDMRcate` function.
`verbose`	Logical value. Should the function be verbose? (Default: FALSE)
`resultSet`	Should results be encapsulated in a `resultSet`? (Default: TRUE)

Details

This function has been deprecated and will be defunct in the new version.

Value

List or `resultSet` with the result of the DMR detection methods.

See Also

`bumphunter`, `blockFinder`, `dmrcate`

Examples

```
if (require(minfiData)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
 model <- model.matrix(~Sample_Group, data = pData(MsetEx))
 res <- runRegionAnalysis(set, model)
 res
}
```

---

topRDAhits *Get the top features associated with the RDA*

Description

Get a list of the features significantly associated to the first two RDA components

Usage

```
topRDAhits(object, tPV = 0.05)
```

Arguments

`object`	`ResultSet`
`tPV`	numeric with the p-value threshold. Only features with a p-values below this threshold will be shown.

Value

data.frame with the features, the component, the correlation and the p-value

Examples

```
if (require(minfiData) & require(GenomicRanges)){
set <- ratioConvert(mapToGenome(MsetEx[1:10,]))
model <- model.matrix(~set$sex)
rda <- runRDA(set, model)
topRDAhits(rda)
}
```

Index