Package 'knowYourCG'

September 12, 2024

Type Package

Title Functional analysis of DNA methylome datasets

Version 1.1.0

Description knowYourCG automates the functional analysis of DNA methylation data. The package tests the enrichment of discrete CpG probes across thousands of curated biological and technical features. GSEA-like analysis can be performed on continuous methylation data query sets. knowYourCG can also take beta matrices as input to perform feature aggregation over the curated database sets.

Depends R (>= 4.4.0)

URL https://github.com/zhou-lab/knowYourCG

BugReports https://github.com/zhou-lab/knowYourCG/issues

License MIT + file LICENSE

VignetteBuilder knitr

Encoding UTF-8

LazyData false

RoxygenNote 7.2.3

Imports sesameData, dplyr, methods, rlang, GenomicRanges, IRanges, reshape2, S4Vectors, stats, stringr, utils

biocViews Epigenetics, DNAMethylation, MethylationArray

Suggests testthat (>= 3.0.0), SummarizedExperiment, rmarkdown, knitr, sesame, gprofiler2

Config/testthat/edition 3

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

git_branch devel

git_last_commit 4077bd9

git_last_commit_date 2024-04-30

Repository Bioconductor 3.20

Date/Publication 2024-09-11

Author Zhou Wanding [aut],
Goldberg David [aut, cre] (<https://orcid.org/0000-0002-9622-4708>),
Moyer Ethan [ctb]

Maintainer Goldberg David <golddc72@pennmedicine.upenn.edu>

Contents

`aggregateTestEnrichments` *Aggregate test enrichment results*

Description

Aggregate test enrichment results

Usage

```
aggregateTestEnrichments(result_list, column = "estimate", return_df = FALSE)
```

Arguments

`result_list`	a list of results from testEnrichment
`column`	the column name to aggregate (Default: estimate)
`return_df`	whether to return a merged data frame

Value

a matrix for all results

Examples

```
## pick some big TFBS-overlapping CpG groups
sesameData::sesameDataCache(data_titles=
c("KYCG.MM285.TFBSconsensus.20220116","KYCG.MM285.chromHMM.20210210",
"probeIDSignature", "MM285.address"))
cg_lists <- getDBs("MM285.TFBS")
queries <- cg_lists[(sapply(cg_lists, length) > 40000)]
result_list <- lapply(queries, testEnrichment, "MM285.chromHMM")
mtx <- aggregateTestEnrichments(result_list)
```

---

annoProbes *Annotate Probe IDs using KYCG databases*

Description

see sesameData_annoProbes if you'd like to annotate by genomic coordinates (in GRanges)

Usage

```
annoProbes(
  probeIDs,
  databases,
  db_names = NULL,
  platform = NULL,
  sep = ",",
  indicator = FALSE,
  silent = FALSE
)
```

Arguments

probeIDs	probe IDs in a character vector
databases	character or actual database (i.e. list of probe IDs)
db_names	specific database (default to all databases)
platform	EPIC, MM285 etc. will infer from probe IDs if not given
sep	delimiter used in paste
indicator	return the indicator matrix instead of a concatenated annotation (in the case of have multiple annotations)
silent	suppress message

Value

named annotation vector, or indicator matrix

Examples

```
sesameData::sesameDataCache(data_titles=
c("MM285.address","probeIDSignature","KYCG.MM285.designGroup.20210210"))
probes <- names(sesameData::sesameData_getManifestGRanges("MM285"))
anno <- annoProbes(probeIDs=probes, "designGroup", silent = TRUE)
```

`buildGeneDBs` *build gene-probe association database*

Description

build gene-probe association database

Usage

```
buildGeneDBs(
  probeIDs = NULL,
  platform = NULL,
  genome = NULL,
  max_distance = 10000,
  silent = FALSE
)
```

Arguments

`probeIDs`	the query probe list. If NULL, use all the probes on the platform
`platform`	HM450, EPIC, MM285, Mammal40, will infer from query if not given
`genome`	hg38, mm10, ..., will infer if not given.
`max_distance`	probe-gene distance for association
`silent`	suppress messages

Value

gene databases

Examples

```
sesameData::sesameDataCache(data_titles=
c("EPIC.address","genomeInfo.hg38","probeIDSignature"))
query <- c("cg04707299", "cg13380562", "cg00480749")
dbs <- buildGeneDBs(query, platform = "EPIC")
testEnrichment(query, dbs, platform = "EPIC")
```

dbStats *dbStats aggregates methylation of a given betas matrix over specified database set features*

Description

dbStats aggregates methylation of a given betas matrix over specified database set features

Usage

```
dbStats(betas, databases, fun = mean, na.rm = TRUE, n_min = NULL, f_min = 0.1)
```

Arguments

betas	matrix of beta values where probes are on the rows and samples are on the columns
databases	List of vectors corresponding to probe locations for which the features will be extracted
fun	aggregation function, default to mean
na.rm	whether to remove NA
n_min	min number of non-NA for aggregation function to apply, overrides f_min
f_min	min fraction of non-NA for aggregation function to apply

Value

matrix with samples on the rows and database set on the columns

Examples

```
library(SummarizedExperiment)
sesameData::sesameDataCache(data_titles=
c("MM285.467.SE.tissue20Kprobes","KYCG.MM285.probeType.20210630"))
se <- sesameData::sesameDataGet("MM285.467.SE.tissue20Kprobes")
head(dbStats(assay(se), "MM285.probeType")[,1:3])
sesameData::sesameDataGet_resetEnv()
```

getDBs *Get databases by full or partial names of the database group(s)*

Description

Get databases by full or partial names of the database group(s)

Usage

```
getDBs(
  group_nms,
  db_names = NULL,
  platform = NULL,
  summary = FALSE,
  allow_multi = FALSE,
  type = NULL,
  silent = FALSE
)
```

Arguments

group_nms	database group names
db_names	name of the database, fetech only the given databases
platform	EPIC, HM450, MM285, ... If given, will restrict to that platform.
summary	return a summary of database instead of db itself
allow_multi	allow multiple groups to be returned for
type	numerical, categorical, default: all
silent	no messages each query.

Value

a list of databases, return NULL if no database is found

Examples

```
sesameData::sesameDataCache(data_titles=
c("KYCG.MM285.chromHMM.20210210","KYCG.MM285.probeType.20210630"))
dbs <- getDBs("MM285.chromHMM")
dbs <- getDBs(c("MM285.chromHMM", "MM285.probeType"))
```

listDBGroups *List database group names*

Description

List database group names

Usage

```
listDBGroups(filter = NULL, path = NULL, type = NULL)
```

Arguments

filter	keywords for filtering
path	file path to downloaded knowledgebase sets
type	categorical, numerical (default: all)

Value

a list of db group names

Examples

```
head(listDBGroups("chromHMM"))
## or listDBGroups(path = "~/Downloads")
```

testEnrichment *testEnrichment tests for the enrichment of a set of probes (query set) in a number of features (database sets).*

Description

testEnrichment tests for the enrichment of a set of probes (query set) in a number of features (database sets).

Usage

```
testEnrichment(
  probeIDs,
  databases = NULL,
  universe = NULL,
  alternative = "greater",
  include_genes = FALSE,
  platform = NULL,
  silent = FALSE
)
```

Arguments

probeIDs	Vector of probes of interest (e.g., significant probes)
databases	List of vectors corresponding to the database sets of interest with associated meta data as an attribute to each element. Optional. (Default: NA)
universe	Vector of probes in the universe set containing all of the probes to be considered in the test. If it is not provided, it will be inferred from the provided platform. (Default: NA).
alternative	"two.sided", "greater", or "less"
include_genes	include gene link enrichment testing
platform	String corresponding to the type of platform to use. Either MM285, EPIC, HM450, or HM27. If it is not provided, it will be inferred from the query set probeIDs (Default: NA).
silent	output message? (Default: FALSE)

Value

A data frame containing features corresponding to the test estimate, p-value, and type of test.

Examples

```
library(SummarizedExperiment)
sesameData::sesameDataCache(data_titles=
c("MM285.tissueSignature","KYCG.MM285.chromHMM.20210210","MM285.address"))
df <- rowData(sesameData::sesameDataGet("MM285.tissueSignature"))
probes <- df$Probe_ID[df$branch == "B_cell"]
res <- testEnrichment(probes, "chromHMM", platform="MM285")
sesameData::sesameDataGet_resetEnv()
```

---

testEnrichmentFisher *testEnrichmentFisher uses Fisher's exact test to estimate the association between two categorical variables.*

Description

Estimates log2 Odds ratio

Usage

```
testEnrichmentFisher(query, database, universe, alternative = "greater")
```

Arguments

query	Vector of probes of interest (e.g., significant probes)
database	Vectors corresponding to the database set of interest with associated meta data as an attribute to each element.
universe	Vector of probes in the universe set containing all of
alternative	greater or two.sided (default: greater) the probes to be considered in the test. (Default: NULL)

Value

A DataFrame with the estimate/statistic, p-value, and name of test for the given results.

---

testEnrichmentSEA *uses the GSEA-like test to estimate the association of a categorical variable against a continuous variable.*

Description

estimate represent enrichment score and negative estimate indicate a test for depletion

Usage

```
testEnrichmentSEA(
  query,
  databases,
  platform = NULL,
  silent = FALSE,
  precise = FALSE,
  prepPlot = FALSE
)
```

Arguments

query	query, if numerical, expect categorical database, if categorical expect numerical database
databases	database, numerical or categorical, but needs to be different from query
platform	EPIC, MM285, ..., infer if not given
silent	suppress message (default: FALSE)
precise	whether to compute precise p-value (up to numerical limit) of interest.
prepPlot	return the raw enrichment scores and presence vectors for plotting

Value

A DataFrame with the estimate/statistic, p-value, and name of test for the given results.

Examples

```
sesameData::sesameDataCache(data_titles=
c("KYCG.MM285.designGroup.20210210","KYCG.MM285.seqContextN.20210630",
"probeIDSignature"))
query <- getDBs("KYCG.MM285.designGroup")[["TSS"]]
res <- testEnrichmentSEA(query, "MM285.seqContextN")
```

`testEnrichmentSpearman` *testEnrichmentSpearman uses the Spearman statistical test to estimate the association between two continuous variables.*

Description

testEnrichmentSpearman uses the Spearman statistical test to estimate the association between two continuous variables.

Usage

```
testEnrichmentSpearman(num_query, num_db)
```

Arguments

num_query	named numeric vector of probes of interest where names are probe IDs (e.g significant probes)
num_db	List of vectors corresponding to the database set of interest with associated meta data as an attribute to each element.

Value

A DataFrame with the estimate/statistic, p-value, and name of test for the given results.

`testGO` *tests Gene Ontology of genes overlapping CpG query*

Description

estimate represent enrichment score and negative estimate indicate a test for depletion

Usage

```
testGO(probeIDs, platform = NULL, organism = "hsapiens", gene_name = TRUE, ...)
```

Arguments

probeIDs	Vector of CpG probes IDs or a data frame with gene_name column, usually the output of testEnrichment() function
platform	EPIC, MM285, ..., infer if not given
organism	The organism corresponding to the CpG platform or genes in gene_name column
gene_name	If query is data frame from testEnrichment output, whether to use the gene_name column. If set to FALSE, TFBS will be used (default: FALSE)
...	Additional arguments to sesameData_getGenesByProbes and gost()

Value

A list of enriched terms and meta data from gprofiler2 output

Examples

```
library(SummarizedExperiment)
sesameData::sesameDataCache(data_titles=
c("MM285.tissueSignature","probeIDSignature",
"MM285.address","genomeInfo.mm10"))
df <- rowData(sesameData::sesameDataGet('MM285.tissueSignature'))
query <- df$Probe_ID[df$branch == "fetal_liver" & df$type == "Hypo"]
res <- testGO(query,platform="MM285")
```

---

testProbeProximity *testProbeProximity tests if a query set of probes share closer genomic proximity than if randomly distributed*

Description

testProbeProximity tests if a query set of probes share closer genomic proximity than if randomly distributed

Usage

```
testProbeProximity(
  probeIDs,
  gr = NULL,
  platform = NULL,
  iterations = 100,
  bin_size = 1500
)
```

Arguments

probeIDs	Vector of probes of interest (e.g., significant probes)
gr	GRanges to draw samples and compute genomic distances
platform	String corresponding to the type of platform to use. Either MM285, EPIC, HM450, or HM27. If it is not provided, it will be inferred from the query set probeIDs (Default: NA).
iterations	Number of random samples to generate null distribution (Default: 100).
bin_size	the poisson interval size for computing neighboring hits

Value

list containing a dataframe for the poisson statistics and a data frame for the probes in close proximity

Examples

```
sesameData::sesameDataCache(data_titles=
c("MM285.tissueSignature","MM285.address","probeIDSignature"))
library(SummarizedExperiment)
df <- rowData(sesameData::sesameDataGet("MM285.tissueSignature"))
probes <- df$Probe_ID[df$branch == "B_cell"]
res <- testProbeProximity(probeIDs=probes,platform="MM285")
sesameData::sesameDataGet_resetEnv()
```

Index