Package 'signatureSearchData'

October 12, 2020

Title Datasets for signatureSearch package

Description CMAP/LINCS hdf5 databases and other annotations used for signatureSearch software package.

Version 1.2.2

Author Yuzhu Duan <yduan004@ucr.edu>, Thomas Girke <thomas.girke@ucr.edu>

Maintainer Yuzhu Duan <yduan004@ucr.edu>

Depends R (>= 3.6)

Imports ExperimentHub, utils, affy, limma, Biobase, magrittr, dplyr, R.utils, stats, signatureSearch, rhdf5

Suggests knitr, rmarkdown

License Artistic-2.0

biocViews ExperimentHub, ExperimentData, ExpressionData

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.1.1

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

git_branch RELEASE_3_11

git_last_commit 8bf572a

git_last_commit_date 2020-08-19

Date/Publication 2020-10-12

R topics documented:

---

cmap *CMap2 LFC Signature Database*

Description

The MAS5 normalized CEL files from the help file of cmap_expr can be used for differential expression (DE) analysis with 'limma' package to get a matrix containing the LFC (log2 fold change) scores. The columns of the matrix are the treatment v.s. control instances, which is defined in the 'cmap_instances_02.txt' file downloaded from CMap project website. The same as the 'cmap_rank' database, only one treatment condition is selected for a compound in a cell. The resulting matrix represents LFC of 12,403 genes for 1,281 compound treatments in up to 5 cells (3,478 signatures in total). The latter was stored in an HDF5 file, which is referred to as the cmap database. Note, The number of compound treatments in cmap database is slightly different from that of the cmap_expr database. The reason is that some of the compound treatment is discarded if the number of control and treatment samples are less than 3 during the DE analysis.

Details

The cmap database can be downloaded as HDF5 file from Bioconductor's ExperimentHub as shown in the Example section.

The same as the cmap_expr dataset, the loaded cmap data object is also generated from the raw CEL files downloaded from the CMap project site and processed as describe above.

Examples

```
library(ExperimentHub)
# eh <- ExperimentHub()
# query(eh, c("signatureSearchData", "cmap"))
# cmap_path <- eh[["EH3223"]]
# rhdf5::h5ls(cmap_path)
```

cmap_expr *CMap Intensity Signature Database*

Description

To search CMap2 with signatureSearch's correlation based GESS methods (gess_cor), normalized gene expression values (here intensities) are required where the biological replicate information has been collapsed to mean values. For this, the `cmap_expr` database has been created from CEL files, which are the raw data of the Affymetrix technology. To obtain normalized expression data, the CEL files were downloaded from the CMap project site, and then processed with the MAS5 algorithm. Gene level expression data was generated the same way as described in help file of `cmap_rank`. Next, the gene expression values for different concentrations and treatment times of each compound and cell were averaged. Subsequently, the expression matrix was saved to an HDF5 file, referred to as the cmap_expr database. It represents mean expression values of 12,403 genes for 1,309 compound treatments in up to 5 cells (3,587 signatures in total).

Details

The cmap_expr database can be downloaded as HDF5 file from Bioconductor's ExperimentHub as shown in the Example section.

The loaded `cmap_expr` data object is generated from the raw CEL files downloaded from the CMap project site. For documentation and code of generating the cmap_expr databases from sources, please refer to the vignette of this package by running `browseVignettes("signatureSearchData")` in R.

References

CMap project site: https://portals.broadinstitute.org/cmap

Examples

```
library(ExperimentHub)
# eh <- ExperimentHub()
# query(eh, c("signatureSearchData", "cmap_expr"))
# cmap_expr_path <- eh[["EH3224"]]
# rhdf5::h5ls(cmap_expr_path)
```

cmap_rank *Cmap Rank Signature Database*

Description

CMap2 (Version build02) contains GESs for 1,309 drugs and eight cell lines that were generated with Affymetrix Gene Chips as expression platform. In some cases this includes drug treatments at different concentrations and time points. For consistency, the CMap2 data was reduced to drug treatments with concentrations and time points that are comparable to those used for the LINCS data. CMap2 data can be downloaded from GEO or its project site either in raw format or as rank transformed matrix. The ranks are based on DEG analyses of drug treatments (drug vs. no-drug) where the resulting Z-scores were used to generate the rank matrix. The latter was used here and

is referred to as rankMatrix. The Affymetrix probe set identifiers stored in the row name slot of this matrix were translated into gene identifies. To obtain a matrix with unique gene identifiers, the ranks for genes represented by more than one probe set were averaged and then re-ranked accordingly. This final gene level rank matrix, referred to as `cmap_rank`, contains rank profiles for 12,403 genes from 1,309 compound treatments in up to 5 cells corresponding to a total of 3,587 treatment signatures. This matrix can be used for all GESS methods in the signatureSearch package that are compatible with rank data, such as the `gess_cmap` method.

Details

The `cmap_rank` data can be downloaded from Bioconductor's ExperimentHub as HDF5 file. Since CMap2 is much smaller than LINCS, it can be imported in its entirety into a SummarizedExperiment object without excessive memory requirements as shown in the Examples section.

The loaded `cmap_rank` data object is generated from the `rankMatrix` downloaded from the CMap project site. For documentation and code of generating the cmap_rank databases from sources, please refer to the vignette of this package by running `browseVignettes("signatureSearchData")` in R.

References

CMap project site: https://portals.broadinstitute.org/cmap

Examples

```
library(ExperimentHub)
# eh <- ExperimentHub()
# query(eh, c("signatureSearchData", "cmap_rank"))
# cmap_rank_path <- eh[["EH3225"]]
# rhdf5::h5ls(cmap_rank_path)
```

`combineNormRes` *Combine Normalized Data from Different Chip Types*

Description

Function assembles normalization results from different chip types into a single data frame. Note, this is possible since the three chip types used by the CMap2 project contain nearly identical probe set ids. Typically, the input for this step is generated by the upstream `normalizeCel` and `combineResults` functions. For more context, please consult the vignette of this package.

Usage

```
combineNormRes(chiptype_dir, norm_method)
```

Arguments

`chiptype_dir`	character vector containing paths to chiptype directories
`norm_method`	normalization method, one of "MAS5" or "rma".

Value

Object of class `data.frame` containing normalization values

Examples

```
# chiptype_dir <- unique(readRDS("./data/chiptype.rds"))
combineNormRes(chiptype_dir, norm_method="not run")
```

---

combineResults *Combine Batch-processed Normalization Results*

Description

Function combines batch-processed normalization results from same chip type into a single data frame. Typically, the input for this step is generated by the upstream `normalizeCel` function. For more context, please consult the vignette of this package.

Usage

```
combineResults(chiptype_dir, rerun = TRUE)
```

Arguments

`chiptype_dir`	character vector containing paths to chiptype directories
`rerun`	TRUE or FALSE, whether to run the function

Value

File storing normalization values for each CEL file type

Examples

```
# chiptype_dir <- unique(readRDS("./data/chiptype.rds"))
combineResults(chiptype_dir, rerun=FALSE)
```

---

dtlink_db_clue_sti *Drug-Target Annotation Resources*

Description

This is a SQLite database containing drug-target links (drug names to gene SYMBOL ids) obtained from DrugBank, CLUE, and STITCH annotation databases.

Details

It is an intermediate database used for functionality in the `signatureSearch` software package to get target genes/proteins of query drugs in DrugBank (https://www.drugbank.ca/), CLUE (https://clue.io/) and STITCH (http://stitch.embl.de/) databases.

The target proteins of drugs are mapped to their encoding genes by using identifier mapping resources from R/Bioconductor such as the `org.Hs.eg.db` annotation package. Thus, the drug-target links in this database are drug names to gene SYMBOLs mapping.

The STITCH database provides confidence scores for each drug-target interaction. Only interactions with confidence scores >=0.7 were selected.

The drug-target interaction annotations from the above three databases are combined. To minimize noise, e.g. from promiscuous binders, drugs with more than 100 distinct targets are removed. Similarly, targets with more than 100 annotated drugs are excluded.

See Also

get_targets

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
qr <- query(eh, c("signatureSearchData", "dtlink_db_clue_sti"))
dtlink <- eh["EH3228"]
# dtlink <- eh[["EH3228"]]
```

ES_NULL *Null Distribution of WTCS from* gess_lincs *Method*

Description

This pre-computed WTCS null distribution is generated with 1000 random queries (150 randomly sampled up and down gene labels sets) searching against the lincs database. It is used for computing nominal p-values of WTCS scores in the gess_lincs result from signatureSearch package.

References

For detailed description of the gess_lincs method and scores, please refer to the vignette of the signatureSearch package by running browseVignettes("signatureSearch") in your R session.

See Also

gess_lincs

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
es_null <- eh[["EH3234"]]
```

---

getCmapCEL *Download CMap2 CEL Files*

Description

This function will download the 7,056 CEL files from the CMap2 project site (http://www.broadinstitute.org/cmap), and save each of them to a subdirectory named CEL under data. Since this download step will take a long time, the rerun argument has been assigned FALSE in the example code below to avoid running it accidentally. If the raw data are not needed, users can skip this time consuming step and work with the preprocessed cmap or cmap_expr database downloaded from the ExperimentHub instead.

Usage

```
getCmapCEL(rerun = TRUE)
```

Arguments

rerun	TRUE or FALSE, whether to download the data

Value

download files

Examples

```
getCmapCEL(rerun=FALSE) # set 'rerun' to TRUE if download
```

---

goAnno *GO Term to Gene SYMBOLs Mapping Annotation*

Description

It is a `data.frame` containing the GO term ID to gene SYMBOLs mapping information used for `dtnetplot` function in `signatureSearch` package to get the annotated gene labels set if the 'set' argument is a GO term ID.

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
goAnno <- eh["EH3229"]
# goAnno <- eh[["EH3229"]]
```

---

goAnno_drug *GO Term ID to Drug Name Mapping Annotation*

Description

It is a `data.frame` containing the GO term ID to drug name mapping information used for `dsea_*` functions in `signatureSearch` package.

Details

The original GO annotation system contains GO term ID to gene SYMBOL id mappings, the latter are converted to drug names mappings via drug to target protein-encoding genes links in DrugBank, CLUE and STITCH databases.

See Also

- `dsea_hyperG`
- `dsea_GSEA`

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
goAnno_drug <- eh["EH3230"]
# goAnno_drug <- eh[["EH3230"]]
```

GO_DATA *GO Annotation Environment*

Description

It is an RDS file storing the GO annotation environment (GO term ID to gene SYMBOLs mapping, gene SYMBOL to annotated GO terms, GO term ID to its description, GO term ID to its ontology) used for `tsea_*` functions in `signatureSearch` package with GO enrichment analysis. This environment is saved to an RDS file to accelerate the analysis speed by avoiding building this environment from scratch every time running the `tsea` functions.

See Also

`tsea`

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
qr <- query(eh, c("signatureSearchData", "GO_DATA"))
go_data <- eh["EH3231"]
# go_data <- eh[["EH3231"]]
```

GO_DATA_drug *GO to Drug Annotation Environment*

Description

It is an RDS file storing the GO annotation environment where the gene SYMBOLs are mapped to drug names via drug-target interactions. The GO to drug annotation environment contains GO term ID to drug names, drug name to annotated GO terms, GO term ID to its description and GO term ID to its ontology mappings. This environment is used for `dsea_*` methods in the `signatureSearch` package to do GO enrichment analysis directly with test drug set. This environment is stored in an RDS file to accelerate the speed by avoiding building this environment from scratch every time running the `dsea` functions.

See Also

`dsea`

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
qr <- query(eh, c("signatureSearchData", "GO_DATA"))
go_data_drug <- eh["EH3232"]
# go_data_drug <- eh[["EH3232"]]
```

inst_filter *Filter LINCS Level 3 by Condition*

Description

Function to filter Level 3 data from LINCS by specific conditions, such as compound concentrations and treatment times.

Usage

```
inst_filter(
  meta,
  pert_type = "trt_cp",
  dose = 10,
  dose_unit = "um",
  time = 24,
  time_unit = "h"
)
```

Arguments

meta	tibble containing experimental conditions of LINCS data
pert_type	perturbation type, 'trt_cp' refers to treatment ('trt') with compound ('cp'). Description of other perturbation types ('pert_type') can be found in the GEO CMap LINCS User Guide v2.1 URL: https://docs.google.com/document/d/1rbHBy3DKekFm9lZouRZcfLmCsfkUKzGPxxjqxPlYw/edit#
dose	concentration of compound used for treatment, needs to match elements in 'pert_dose' column of 'meta'
dose_unit	unit of dose of compound treatment, needs to match elements in 'pert_dose_unit' column of 'meta'
time	compound treatment time, needs to match elements in 'pert_time' column of 'meta'
time_unit	unit of time, needs to match elements in 'pert_time_unit' column of 'meta'

Value

tibble

Examples

```
meta <- data.frame(pert_dose=c(2,4,10), pert_dose_unit="um",
                   pert_time=24, pert_time_unit="h",
                   pert_type="trt_cp", pert_iname=c("p1","p2","p3"),
                   cell_id="MCF7")
inst_filter(meta)
```

`lincs` *LINCS Z-score Signature Database*

Description

The L1000 assay, used for generating the LINCS data, measures the expression of 978 landmark genes and 80 control genes by loading amplified mRNA populations onto beads and then detecting their abundance with a fluorescent-based method. The expression of 11,350 additional genes is imputed from the landmark genes by using as training data a large collection of Affymetrix gene chips.

The LINCS data have been pre-processed by the Broad Institute to 5 different levels and are available for download from GEO. Level 1 data are the raw mean fluorescent intensity values that come directly from the Luminex scanner. Level 2 data are the expression intensities of the 978 landmark genes. They have been normalized and used to impute the expression of the additional 11,350 genes, forming Level 3 data. A robust z-scoring procedure was used to generate differential expression values from the normalized profiles (Level 4). Finally, a moderated z-scoring procedure was applied to the replicated samples of each experiment (mostly 3 replicates) to compute a weighted average signature (Level 5). For a more detailed description of the preprocessing methods used by the LINCS project, readers want to refer to the LINCS user guide.

Disregarding replicates, the LINCS data set contains 473,647 signatures with unique cell type and treatment combinations. This includes 19,811 drug-like small molecules tested on different cell lines at multiple concentrations and treatment times. In addition to compounds, several thousand genetic perturbations (gene knock-downs and over expressions) have been tested. Currently, the data stored in this package is restricted to signatures of small molecule treatments across different cells lines. However, users have the option to assemble any custom collection of the LINCS data. For consistency, only signatures at one specific concentration (10uM) and one time point (24h) have been selected for each small molecule in the default collection. These choices are similar to the conditions used in primary high-throughput compound screens of cell lines. Since the selected compound concentrations and treatment duration have not been tested by LINCS across all cell types yet, a subset of compounds had to be selected that best met the chosen treatment requirements. This left us with 8,104 compounds that were uniformly tested at the chosen concentration and treatment time, but across variable numbers of cell lines. The total number of expression signatures meeting this requirement is 45,956, while the total number of cell lines included in this data set is 30.

Details

The LINCS sub-dataset, filtered and assembled according to the above criteria, can be downloaded from Bioconductor's `ExperimentHub` as HDF5 file. In this case the loaded data instance includes moderated Z-scores from DE analyses of 12,328 genes for 8,140 compound treatments across a total of 30 cell lines corresponding to 45,956 expression signatures. This data set can be used by all set-based and correlation-based GESS methods provided by the signatureSearch package.

The loaded `lincs` data object is filtered and generated from the original LINCS level 5 data stored at GEO: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE92742. For documentation and code of generating the lincs databases from sources, please refer to the vignette of this package by running `browseVignettes("signatureSearchData")` in R.

References

LINCS source data at 5 levels from GEO: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE92742

LINCS User Guide v2.1: https://docs.google.com/document/d/1q2gciWRhVCAAnlvF2iRLuJ7whrGP6QjpsCMq1yWz7

Examples

```
library(ExperimentHub)
# eh <- ExperimentHub()
# query(eh, c("signatureSearchData", "lincs"))
# lincs_path <- eh[['EH3226']]
# rhdf5::h5ls(lincs_path)
```

`lincs_expr` *LINCS Intensity Signature Database*

Description

The LINCS Level 3 data can be downloaded from GEO the same way as described for the Level 5 data available at help file of `lincs`. The Level 3 data contain normalized gene expression values across all treatments and cell lines used by LINCS. The Level 3 signatures were filtered using the same dosage and duration criteria as the Level 5 data. The biological replicate information included in the Level 3 data were collapsed to mean values. Subsequently, the resulting matrix of mean expression values was written to an HDF5 file. The latter is referred to as `lincs_expr` database containing 38,824 signatures for a total of 5,925 small molecule treatments and 30 cell lines. Although the LINCS Level 3 and 5 data are filtered here the same way, the number of small molecules represented in the Level 3 data (5,925) is smaller than in the Level 5 data (8,140). The reason for this inconsistency is most likely that the Level 3 dataset, downloadable from GEO, is incomplete.

Details

The filtered and processed LINCS Level3 data (lincs_expr) can be loaded from Bioconductor's `ExperimentHub` interface as shown in Examples section. In this case the loaded lincs_expr instance includes mean expression values of 12,328 genes for 5,925 compound treatments across a total of 30 cell lines (38,824 signatures in total). This data set can be used by all correlation-based GESS methods provided by the signatureSearch package.

The loaded `lincs_expr` data object is filtered and generated from the original LINCS level 3 data stored at GEO: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE92742. For documentation and code of generating the lincs_expr databases from sources, please refer to the vignette of this package by running `browseVignettes("signatureSearchData")` in R.

References

LINCS source data at 5 levels from GEO: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE92742

Examples

```
library(ExperimentHub)
# eh <- ExperimentHub()
# query(eh, c("signatureSearchData", "lincs_expr"))
# lincs_expr_path <- eh[["EH3227"]]
# rhdf5::h5ls(lincs_expr_path)
```

meanExpr *Calculate Mean Value for Replicated Samples*

Description

Function averages the normalized gene expression values for different concentrations, treatment times and replicates of compounds and cell types. For more context on this step, please consult the corresponding section in the vignette of this package.

Usage

```
meanExpr(expr_df, cmap_inst)
```

Arguments

expr_df	data.frame containing normalized expression values
cmap_inst	data.frame defining experimental conditions

Value

data.frame with mean expression values of replicates

Examples

```
path <- system.file("extdata", "cmap_instances_02.txt", package="signatureSearchData")
cmap_inst <- read.delim(path, check.names=FALSE)
expr_df <- as.data.frame(matrix(runif(30), ncol=5))
colnames(expr_df) <- paste0(cmap_inst$perturbation_scan_id[seq_len(5)],".CEL")
mexpr <- meanExpr(expr_df, cmap_inst[seq_len(5),])
```

meanExpr2h5 *Calculate Mean Expression Values of LINCS Level 3 Data*

Description

Function calculates mean expression values for replicated samples of LINCS Level 3 data that have been treated by the same compound in the same cell type at a chosen concentration and treatment time. Usually, the function is used after filtering the Level 3 data with inst_filter. The results (here matrix with mean expression values) are saved to an HDF5 file. The latter is referred to as the 'lincs_expr' database.

Usage

```
meanExpr2h5(gctx, inst, h5file, chunksize = 2000, overwrite = TRUE)
```

Arguments

gctx	character(1), path to the LINCS Level 3 gctx file
inst	tibble, LINCS Level 3 instances after filtering for specific concentrations and times
h5file	character(1), path to the destination HDF5 file
chunksize	number of columns of the matrix to be processed at a time to limit memory usage
overwrite	TRUE or FALSE, whether to overwrite or append data to an existing 'h5file'

Value

HDF5 file, representing the lincs_expr database

Examples

```
gctx <- system.file("extdata", "test_sample_n2x12328.gctx", package="signatureSearchData")
h5file <- tempfile(fileext=".h5")
inst <- data.frame(inst_id=c("ASG001_MCF7_24H:BRD-A79768653-001-01-3:10",
                             "CPC012_SKB_24H:BRD-K81418486:10"),
    pert_cell_factor=c('sirolimus__MCF7__trt_cp', 'vorinostat__SKB__trt_cp'))
meanExpr2h5(gctx, inst, h5file, overwrite=TRUE)
```

---

normalizeCel *Normalize CEL Files*

Description

Function processes the CEL files from each chip type (three for CMap2 data) separately using the MAS5 normalization algorithm. The results will be written to subdirectores (under a data parent directory) that are named after the chip type names. To reduce the memory consumption of this step, the CEL files are processed in user definable batch sizes. For details on the overall workflow, please consult the vignette of this package. The normalization takes about 10 hours without parallelization. To save time, this process can be easily accelerated on a computer cluster.

Usage

```
normalizeCel(chiptype_list, batchsize = 300, rerun = TRUE)
```

Arguments

chiptype_list	list, storing CEL files in each chiptype
batchsize	number of CEL files to import and to normalize at once
rerun	TRUE or FALSE, whether to run the function

Value

Files storing normalized expression values in matrix, here one matrix file per chip type

Examples

```
# chiptype_list <- split(names(chiptype), as.character(chiptype))
normalizeCel(chiptype_list, rerun=FALSE)
```

probe2gene *Transform Probe Set to Gene Level Data*

Description

Function to transform expression values from probe set to gene level data. If genes are represented by several probe sets then their mean intensities are used. Probe sets not matching any gene ids are removed.

Usage

```
probe2gene(expr_df, annot)
```

Arguments

expr_df	data.frame with probe set level expression data
annot	data.frame containing probe set id (in row name slot) to gene entrez id (in first column) mapping information

Value

data.frame with gene level expression values

Examples

```
expr_df <- data.frame(t1=runif(3), t2=runif(3), row.names=paste0("p", 1:3))
annot <- data.frame(ENTREZID=c("123","123","124"),
                    SYMBOL=c("g1","g1","g2"),
                    row.names=paste0("p", 1:3))
gdf <- probe2gene(expr_df, annot)
```

runLimma *DEG Analysis with Limma*

Description

This function runs DEG analysis with Limma at user defined FDR and LFC cutoff by providing CMAP02 normalized expression values and annotation of treatment v.s. control samples. For more context of the CMAP02 database, please consult the vignette of this package.

Usage

```
runLimma(df, comp_list, fdr = 0.05, foldchange = 1, verbose = TRUE)
```

Arguments

df	data.frame containing normalized intensity values of CMAP02 samples
comp_list	list of CEL file ids of treatment and control samples for each compound treatment. The list for CMAP02 data is generated from the sampleList function.
fdr	cutoff of false discovery rate (FDR) for defining DEGs
foldchange	cutoff of log2 fold change (LFC) for defining DEGs
verbose	TRUE or FALSE

Value

list containing DEGs and log2FC matrix

Examples

```
path <- system.file("extdata", "cmap_instances_02.txt", package="signatureSearchData")
cmap_inst <- read.delim(path, check.names=FALSE)
comp_list <- sampleList(cmap_inst, myby="CMP_CELL")
df <- as.data.frame(matrix(runif(70), ncol=7))
colnames(df) <- unlist(comp_list[1])
degList <- runLimma(df, comp_list[1])
```

---

sampleList *Generate List of Treatment vs. Control Samples*

Description

This function generates a list of treatment samples and control samples CEL ids for each compound treatment from the experiment annotation of CMAP02 instances. For more context of the CMAP02 database, please consult the vignette of this package.

Usage

```
sampleList(cmap, myby)
```

Arguments

cmap	data.frame containing experiment information of CMAP02 instances
myby	"CMP" or "CMP_CELL", "CMP": by compound treatments in all cells; "CMP_CELL": by compound treatments in individual cell

Value

a list object containing treatment samples and control samples ids for each compound treament.

Examples

```
path <- system.file("extdata", "cmap_instances_02.txt", package="signatureSearchData")
cmap_inst <- read.delim(path, check.names=FALSE)
comp_list <- sampleList(cmap_inst, myby="CMP_CELL")
```

signatureSearchData *signatureSearchData: Reference Data for Gene Expression Signature Searching*

Description

This package provides access to the reference data used by the associated signatureSearch software package. The latter allows to search with a query gene expression signature (GES) a database of treatment GESs to identify cellular states sharing similar expression responses (connections). This way one can identify drugs or gene knockouts that induce expression phenotypes similar to a sample of interest. The resulting associations may lead to novel functional insights how perturbagens of interest interact with biological systems.

Currently, this data package includes GES data from the CMap (Connectivity Map) and LINCS (Library of Network-Based Cellular Signatures) projects that are largely based on drug and genetic perturbation experiments performed on variable numbers of human cell lines (Lamb et al. 2006; Subramanian et al. 2017). In signatureSearchData these data sets have been preprocessed to be compatible with the different gene expression signature search (GESS) algorithms implemented in signatureSearch. The preprocessed data types include but are not limited to normalized gene expression values (e.g. intensity values), log fold changes (LFC) and Z-scores, p-values or FDRs of differentially expressed genes (DEGs), rankings based on selected preprocessing routines or sets of top up/down-regulated DEGs.

The CMap data were downloaded from the CMap project site (Version build02). The latter is a collection of over 7,000 gene expression profiles (signatures) obtained from perturbation experiments with 1,309 drug-like small molecules on five human cancer cell lines. The Affymetrix Gene Chip technology was used to generate the CMAP2 data set.

In 2017, the LINCS Consortium generated a similar but much larger data set where the total number of gene expression signatures was scaled up to over one million. This was achieved by switching to a much more cost effective gene expression profiling technology called L1000 assay (Peck et al. 2006; Edgar, Domrachev, and Lash 2002). The current set of perturbations covered by the LINCS data set includes 19,811 drug-like small molecules applied at variable concentrations and treatment times to ~70 human non-cancer (normal) and cancer cell lines. Additionally, it includes several thousand genetic perturbagens composed of gene knockdown and over-expression experiments.

The data structures and search algorithms used by signatureSearch and signatureSearchData are designed to work with most genome-wide expression data including hybridization-based methods, such as Affymetrix or L1000, as well as sequencing-based methods, such as RNA-Seq. Currently, signatureSearchData does not include preconfigured RNA-Seq reference data mainly due to the lack of large-scale perturbation studies (e.g. drug-based) available in the public domain that are based on RNA-Seq. This situation may change in the near future once the technology has become more affordable for this purpose.

This package also contains other intermediate annotation datasets, such as drug-target information of small molecules in DrugBank, CLUE and STITCH databases used for getting target annotation of query drugs, GO system annotations and drugs to GO functional category mappings used for signatureSearch's functional enrichment analysis (FEA) routines or null distribution of Enrichment Scores (ES) used for computing WTCS p-values from LINCS GESS method.

Details

This package contains the following main datasets:

- cmap

- cmap_expr
- cmap_rank
- lincs
- lincs_expr
- dtlink_db_clue_sti
- taurefList

For documentation of generating the CMAP/LINCS databases from sources, please refer to the vignette of this package by running `browseVignettes("signatureSearchData")` in R.

References

Lamb, Justin, Emily D Crawford, David Peck, Joshua W Modell, Irene C Blat, Matthew J Wrobel, Jim Lerner, et al. 2006. "The Connectivity Map: Using Gene-Expression Signatures to Connect Small Molecules, Genes, and Disease." Science 313 (5795): 1929–35. http://dx.doi.org/10.1126/science.1132939.

Subramanian, Aravind, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, et al. 2017. "A Next Generation Connectivity Map: L1000 Platform and the First 1,000,000 Profiles." Cell 171 (6): 1437–1452.e17. http://dx.doi.org/10.1016/j.cell.2017.10.049.

Edgar, Ron, Michael Domrachev, and Alex E Lash. 2002. "Gene Expression Omnibus: NCBI Gene Expression and Hybridization Array Data Repository." Nucleic Acids Res. 30 (1): 207–10. https://www.ncbi.nlm.nih.gov/pubmed/11752295.

Peck, David, Emily D Crawford, Kenneth N Ross, Kimberly Stegmaier, Todd R Golub, and Justin Lamb. 2006. "A Method for High-Throughput Gene Expression Signature Analysis." Genome Biol. 7 (7): R61. http://dx.doi.org/10.1186/gb-2006-7-7-r61.

See Also

`signatureSearch`

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
ssd <- query(eh, c("signatureSearchData"))
ssd
ssd$title
as.list(ssd)[10]

## Retrieve CMAP databases
cmap <- eh["EH3223"] # stores LFC scores
cmap_expr <- eh["EH3224"] # stores gene expression intensity values
cmap_rank <- eh["EH3225"] # stores gene ranks

## Retrieve LINCS databases
lincs <- eh['EH3226'] # stores moderated z-scores
lincs_expr <- eh['EH3227'] # stores gene expression intensity values
```

sig_filter *Filter LINCS Level 5 by Condition*

Description

Function to filter Level 5 data from LINCS by specific conditions, such as compound concentrations and treatment times.

Usage

```
sig_filter(meta, pert_type = "trt_cp", dose, time = "24 h")
```

Arguments

meta	tibble containing experimental conditions of LINCS data
pert_type	perturbation type, 'trt_cp' refers to treatment ('trt') with compound ('cp'). Description of other perturbation types ('pert_type') can be found in the GEO CMap LINCS User Guide v2.1 URL: https://docs.google.com/document/d/1rbHBy3DKekFm9lZouRZcfLmCsfkUKzGPxxjqxPlYw/edit#
dose	concentration of compound used for treatment, needs to match elements in 'pert_idose' column of 'meta'
time	compound treatment time, needs to match elements in 'pert_itime' column of 'meta'

Value

tibble

Examples

```
meta <- data.frame(pert_idose=c("2 um","4 um", "10 um"),
                   pert_itime="24 h",
                   pert_type="trt_cp", pert_iname=c("p1","p2","p3"),
                   cell_id="MCF7")
sig_filter(meta, dose="2 um")
```

taurefList *Lookup Table for Computing Tau Scores*

Description

This dataset is generated by searching all signatures in the LINCS database against itself to obtain a background distribution of NCS values (Qref). The latter is used to compute the Tau scores of the `gess_lincs` method in the `signatureSearch` package. The Tau score compares the observed enrichment score to all others in Qref. It represents the percentage of reference queries with a lower |NCS| than |NCSq,r|, adjusted to retain the sign of NCSq,r. NCSq,r is the normalized connectivity score for signature r relative to query q. A tau of 90 indicates that only 10 percent of reference perturbations showed stronger connectivity to the query.

References

For a detailed description of the gess_lincs method and its scores, please refer to the vignette of the signatureSearch package by running browseVignettes("signatureSearch").

Subramanian, A., Narayan, R., Corsello, S. M., Peck, D. D., Natoli, T. E., Lu, X., Golub, T. R. (2017). A Next Generation Connectivity Map: L1000 Platform and the First 1,000,000 Profiles. Cell, 171(6), 1437-1452.e17. https://doi.org/10.1016/j.cell.2017.10.049

See Also

gess_lincs

Examples

```
library(ExperimentHub)
eh <- ExperimentHub()
taurefList <- eh["EH3233"]
# taurefList <- eh[["EH3233"]]
```

Index