Package 'MultiSEp'

August 27, 2026

Type Package

Title Predict Synthetic Lethality and Other Gene Dependency Relationships from Multiomics Data

Version 4.1.3

BuildVignettes TRUE

Description Predicts gene dependency relationships (GDRs) from functional genomics data. Regularised Gaussian mixture modelling may be applied for unsupervised clustering in addition to other data partitioning strategies. GDR analysis tools include discovery of synthetic lethal relationships and predicting population coverage for candidate drug targets from cancer patient mutational profiles. Functionality for visualisation is also available. 'MultiSEp' is applicable to data from a variety of sources including clinical cohorts, organoids and cell lines.

License LGPL-2.1

Depends R (>= 3.5.0)

Imports pracma, pbapply, pbmcapply, reshape2, ggplot2, gtools, testthat, igraph, grid, patchwork, scales

Encoding UTF-8

LazyData true

LazyDataCompression bzip2

NeedsCompilation no

VignetteBuilder knitr, rmarkdown

Suggests rmarkdown, kableExtra, magrittr, knitr, dplyr

Language en-GB

Author Adeline McKie [aut], Mark Wappett [aut], Ian Overton [aut, cre]

Maintainer Ian Overton <i.overton@qub.ac.uk>

Repository CRAN

Date/Publication 2026-08-27 10:30:02 UTC

Contents

`depMapCRISPRscores_subset`

Example CRISPR score input matrix

Description

Log2 CRISPR scores where row names correspond to genes and column names correspond to samples.

Usage

```
data("depMapCRISPRscores_subset")
```

Format

A data frame with 50 observations (genes) across 1054 variables (cell lines).

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(depMapCRISPRscores_subset)
depMapCRISPRscores_subset[1:10,1:10]
```

---

depMapMUT_small *Example mutation data input matrix*

Description

Discrete mutation values where row names correspond to genes and column names correspond to samples. Wild type denoted by 'WT' and mutations are indicated by amino acid changes.

Usage

```
data("depMapMUT_small")
```

Format

A data frame with 6 observations (genes) across 1299 variables (cell lines).

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(depMapMUT_small)
depMapMUT_small[1:10,1:10]
```

---

depMapMUT_subset *Example mutation data input matrix*

Description

Discrete mutation values where row names correspond to genes and column names correspond to samples. Wild type denoted by 'WT' and mutations are indicated by amino acid changes.

Usage

```
data("depMapMUT_subset")
```

Format

A data frame with 132 observations (genes) across 1738 variables (cell lines).

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(depMapMUT_subset)
depMapMUT_subset[1:10,1:10]
```

depMapTissue_subset *Example tissue data input matrix*

Description

A data frame with 789 observations (samples) across 2 variables (tissues).

Usage

```
data("depMapTissue_subset")
```

Format

A data frame where column 1 corresponds to the sample ID and column 2 corresponds to the tissue value.

`cell_line` a character vector

`tissue` a character vector

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(depMapTissue_subset)
depMapTissue_subset[1:10,]
```

depMapXPR_small *Example gene expression input matrix*

Description

Log2 gene expression data for 11 observations (genes) across 1299 variables (cell lines).

Usage

```
data("depMapXPR_small")
```

Format

A data frame where rownames correspond to genes and column names correspond to samples.

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(depMapXPR_small)
depMapXPR_small[1:10,1:10]
```

---

depMapXPR_subset *Example gene expression input matrix*

Description

Log2 gene expression data for 50 observations (genes) across 1304 variables (cell lines).

Usage

```
data("depMapXPR_subset")
```

Format

A data frame where rownames correspond to genes and column names correspond to samples.

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(depMapXPR_subset)
depMapXPR_subset[1:10,1:10]
```

Formatted_MUTimpactMatrix

The output of `mts_formatMatrix()` *function for an example dataset.*

Description

List of 23 dataframes where each element corresponds to a gene. The mutation impact data per sample is given for each gene, correctly formatted for analysis with the `mts_patternDetection` function.

Usage

```
data("Formatted_MUTimpactMatrix")
```

Format

A data frame with 3 columns: 'Samples', 'Impact' and 'impact Group'.

Examples

```
data(Formatted_MUTimpactMatrix)
```

mixModelClusters_depMapCRISPR

Clusters of samples (cell lines) from the `mts_mixModelCluster` *function for a small dataset of DepMap CRISPR scores.*

Description

List of 5 data frames where each element corresponds to a gene.

Usage

```
data("mixModelClusters_depMapCRISPR")
```

Format

Each data frame in the list has 3 columns: gene ID, log2 gene expression and cluster assignment. Also provides an example of a gene, EIF1AX, that was not able to receive cluster assignments.

Source

Derived from analysis of data taken from DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(mixModelClusters_depMapCRISPR)
```

`mixModelClusters_depMapXPR`

Clusters output from the `mts_mixModelCluster` *function for a toy DepMap expression dataset.*

Description

List of 5 data frames where each element corresponds to a gene.

Usage

```
data("mixModelClusters_depMapXPR")
```

Format

A data frame with 3 columns: gene ID, log2 gene expression and cluster assignment.

Source

Derived from analysis of data taken from DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(mixModelClusters_depMapXPR)
```

mixModelClusters_vignette

Clusters output from the `mts_mixModelCluster` *function for a DepMap expression data that is required for the vignette.*

Description

List of 9 data frames where each element corresponds to a gene.

Usage

```
data("mixModelClusters_vignette")
```

Format

Each data frame has 3 columns: gene ID, log2 gene expression and cluster assignment.

Source

Derived from analysis of data taken from DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(mixModelClusters_vignette)
```

mRes

Example results for the `mts_Mutation` *function.*

Description

A data frame containing example predicted gene dependencies produced by the `mts_Mutation` function (which takes as input gene expression and mutation data).

Usage

```
data("mRes")
```

Format

A data frame with 3039 observations on the following 5 variables.

`mRNA_gene` a character vector

`mutation_gene` a character vector

`tissue` a character vector

`chiSqPvalue` a numeric vector

`mutation_per_mode` a character vector

References

Wappett *et al.* (2021) [*Nucleic Acids Research* 49, W613-W618] doi:10.1093/nar/gkab338

Examples

```
data(mRes)
mRes[1:10,]
```

`mts_clusterAvg` *Determine average CRISPR score values for gene expression clusters*

Description

MultiSEp applies regularised Gaussian Mixture Modelling (GMM) with expectation-maximisation to discover clusters, or modes, from functional genomics data; for example gene expression (such as RNA-seq), or CRISPR scores. Cardinality (the number of clusters) is determined by regularisation with Bayesian Information Criterion, considering at least two and up to five clusters. Each sample is assigned a probability of belonging to each cluster by MultiSEp, for example some samples may be positioned in a region of overlap between two clusters formed from the mixture of Gaussians used to model the distribution of the functional genomics data. The `mts_clusterAvg` function generates Gaussian mixture models for gene expression data, assigning samples to clusters for each gene. Subsequently, the average CRISPR score value is calculated per cluster of samples for all of the genes with CRISPR data. Accordingly, each gene with expression data is paired with all of the genes with CRISPR data across the samples analysed and these results provide the basis for quantification of the differences in CRISPR scores between gene expression clusters in downstream functions.

Usage

```
mts_clusterAvg(exprsMatrix, crisprMatrix, cores)
```

Arguments

exprsMatrix	A log2 gene expression matrix in gene by sample format, where rownames correspond to genes (or probesets) and colnames correspond to sample names.
crisprMatrix	A crispr score matrix in gene by sample format, where rownames correspond to genes (or guide RNA's) and colnames correspond to sample names. The score should be derived from a pooled or arrayed CRISPR screen where negative scores correspond to dependency and positive scores correspond to outgrowth.
cores	The number of compute cores to use. Defaults to 1.

Value

A list of sublists corresponding to each gene in the exprsMatrix object. Each sublist has the following components:

AverageCrisprScore	The average CRISPR score of each gene in the crisprMatrix object in each cluster assigned by Multisep to each gene in the exprsMatrix object. Column headings 'Cluster 1', 'Cluster 2' etc.
Sample	The sample name, for example cell line identifiers.
Log2 Expression	The log2 gene expression value for the sample.
Expression Cluster	The cluster assigned to the sample by Multisep

References

Wappett *et al.* (2021) [*Nucleic Acids Research* 49, W613-W618] doi:10.1093/nar/gkab338

See Also

mts_plotCRISPRGeneCluster, mts_Crispr, mts_CrisprTS, mts_InducedDependency

Examples

```
data(depMapXPR_subset)
data(depMapCRISPRscores_subset)

mR1 <- mts_clusterAvg(
  exprsMatrix=depMapXPR_subset[1,],
  crisprMatrix=depMapCRISPRscores_subset[1:10],
  cores=1
)
```

`mts_Crispr` *Investigate Gene Dependency Relationships with CRISPR and gene expression data*

Description

Performs statistical evaluation of gene dependency relationships for CRISPR scores between consecutive mRNA expression clusters.

Usage

```
mts_Crispr(resultList = resultList, exprsMatrix = exprsMatrix,
crisprMatrix = crisprMatrix, fcVal = fcVal, pVal = pVal, cores = cores)
```

Arguments

`resultList`	A result list object produced by the `mts_clusterAvg` function.
`exprsMatrix`	A log2 gene expression matrix in gene by sample format, where rownames correspond to genes (or probesets) and colnames correspond to sample names.
`crisprMatrix`	A crispr score matrix in gene by sample format, where rownames correspond to genes (or guide RNA's) and colnames correspond to sample names. The score should be derived from a pooled or arrayed CRISPR screen where negative scores correspond to dependency and positive scores correspond to outgrowth.
`fcVal`	A log 2 fold change value filter. Defaults to -0.1.
`pVal`	A p value filter. Defaults to 0.1.
`cores`	The number of compute cores to use. Defaults to 1.

Details

Gene expression clusters enable partitioning of effect scores from CRISPR screen data, and therefore may reveal pairwise gene dependencies. Statistical evaluation of dependency relationships is carried out by calculating a log2 fold-change and two-tailed T-test p-value between the CRISPR scores for each consecutive pair of mRNA expression clusters. The user can define the fold change (default:-0.1) and p-value (default: 0.1) cutoff values. Q-values are also calculated; we strongly recommend taking the q-values as a baseline for determining statistical significance, indeed it is crucially important to correct for multiple hypothesis testing to ensure the statistical validity of results.

Value

A data frame containing the significantly enriched gene expression and CRISPR gene pair results. The data frame contains the following columns:

`mrna_gene`	Gene id of the gene expression gene
`crispr_gene`	Gene id of the crispr gene
`num_modes`	Number of gene expression clusters

nMode1-5	Number of cell lines in each gene expression cluster
mMode1-5	Mean CRISPR score of the crispr gene in each gene expression cluster
shift2_1	Fold-change of CRISPR scores between expression modes 2 and 1
shift3_2	Fold-change of CRISPR scores between expression modes 3 and 2. NA if mRNA gene has 2 gene expression clusters
shift4_3	Fold-change of CRISPR scores between expression modes 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
shift5_4	Fold-change of CRISPR scores between expression modes 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters
pvalue2_1	P-value of CRISPR scores between expression modes 2 and 1
pvalue3_2	P-value of CRISPR scores between expression modes 3 and 2. NA if mRNA gene has 2 gene expression clusters
pvalue4_3	P-value of CRISPR scores between expression modes 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
pvalue5_4	P-value of CRISPR scores between expression modes 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters
qvalue2_1	Q-value of CRISPR scores between expression modes 2 and 1
qvalue3_2	Q-value of CRISPR scores between expression modes 3 and 2. NA if mRNA gene has 2 gene expression clusters
qvalue4_3	Q-value of CRISPR scores between expression modes 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
qvalue5_4	Q-value of CRISPR scores between expression modes 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters

See Also

mts_clusterAvg

Examples

```
data(depMapXPR_subset)
data(depMapCRISPRscores_subset)
data(multisepList)

resultListSmall <- lapply(multisepList["DNAJC15"], function(g) {
    keep <- intersect(rownames(g[[1]]), rownames(depMapCRISPRscores_subset))
    g[[1]] <- g[[1]][keep, , drop = FALSE]
    g
  })

mCrisprOut <- mts_Crispr(
  resultList = resultListSmall,
  exprsMatrix = depMapXPR_subset,
  crisprMatrix = depMapCRISPRscores_subset
)
```

`mts_crisprPartition` *Partition CRISPR gene effect scores*

Description

Partitions the CRISPR gene effect scores using the supplied threshold. For example a score of -0.5 is the default threshold, which aims to partition samples into 'lethal' (representative of cell death or stasis) and 'non-lethal' groups.

Usage

```
mts_crisprPartition(
  dataMatrix,
  NumSampleThreshold = 20,
  partitionThreshold=-0.5,
  verbose=FALSE,
  cores)
```

Arguments

`dataMatrix`	CRISPR score matrix in gene by sample format, where row names correspond to genes and column names correspond to sample names.
`NumSampleThreshold`	The minimum number of samples in which a gene must have a CRISPR score for it to be retained; genes with scores in more than `NumSampleThreshold` samples are kept. Defaults to 20.
`partitionThreshold`	The score value used to split data into two partitions, default is -0.5.
`verbose`	Controls the production of reporting messages, default is `FALSE` (i.e. no verbose messages).
`cores`	The number of compute cores to use, default is 1.

Value

A list object for each gene in the `dataMatrix` object

`Samples`	Sample identifiers.
`Values`	The analysed value for each sample, for example the gene effect chronos score.
`Cluster_Assignment`	The assigned cluster value for each sample.

See Also

`mts_patternDetection`

Examples

```
data("depMapCRISPRscores_subset")
crisprClusters_thresholded <- mts_crisprPartition(
  dataMatrix=depMapCRISPRscores_subset[1:5,], cores=1
)
```

`mts_CrisprTS` *Predict tissue-specific gene dependency relationships, evaluating differences in CRISPR scores for gene A between expression clusters for gene B (SynLeGG).*

Description

Subsets the output from the `mts_Crispr` function according to tissue type and performs the analysis again in a tissue specific manner. Running with negative fold-change values (<0) allows prediction of synthetic lethal relationships, specifying positive fold-change values (>0) corresponds to 'induced dependency' relationships.

Usage

```
mts_CrisprTS(resultList = resultList, crisprMatrix = crisprMatrix, fcVal = fcVal,
pVal = pVal, cores = cores, tissueMatrix = tissueMatrix, allDisRes = allDisRes)
```

Arguments

`resultList`	A result list object produced by the `mts_clusterAvg` function.
`crisprMatrix`	A crispr score matrix in gene by sample format, where rownames correspond to genes (or guide RNA's) and colnames correspond to sample names. The score should be derived from a pooled or arrayed CRISPR screen where negative scores correspond to dependency and positive scores correspond to outgrowth.
`fcVal`	The log2 fold-change threshold, default is -0.1.
`pVal`	The p-value threshold, default is 0.1.
`cores`	The number of compute cores to use, defaults to 1.
`tissueMatrix`	A data frame containing 2 columns where column 1 is the sample id and column 2 is the tissue id.
`allDisRes`	A result list object produced by the `mts_Crispr` function.

Value

A data frame containing the significantly enriched gene expression and CRISPR gene pair results. The data frame contains the following columns:

`mrna_gene`	Gene id of the gene expression gene
`crispr_gene`	Gene id of the crispr gene
`tissue`	Tissue ID of the gene pair comparison

num_modes	Number of gene expression clusters
nMode1-5	Number of cell lines in each gene expression cluster
mMode1-5	Mean CRISPR score of the crispr gene in each gene expression cluster
shift2_1	Fold-change of CRISPR scores between expression modes 2 and 1
shift3_2	Fold-change of CRISPR scores between expression modes 3 and 2. NA if mRNA gene has 2 gene expression clusters
shift4_3	Fold-change of CRISPR scores between expression modes 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
shift5_4	Fold-change of CRISPR scores between expression modes 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters
pvalue2_1	P-value of CRISPR scores between expression modes 2 and 1
pvalue3_2	P-value of CRISPR scores between expression modes 3 and 2. NA if mRNA gene has 2 gene expression clusters
pvalue4_3	P-value of CRISPR scores between expression modes 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
pvalue5_4	P-value of CRISPR scores between expression modes 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters
qvalue2_1	Q-value of CRISPR scores between expression modes 2 and 1
qvalue3_2	Q-value of CRISPR scores between expression modes 3 and 2. NA if mRNA gene has 2 gene expression clusters
qvalue4_3	Q-value of CRISPR scores between expression modes 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
qvalue5_4	Q-value of CRISPR scores between expression modes 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters

References

Wappett *et al.* (2021) [*Nucleic Acids Research* 49, W613-W618] doi:10.1093/nar/gkab338

Examples

```
data(depMapCRISPRscores_subset)
data(depMapTissue_subset)
data(multisepList)
data(multisepCrisprList)
allDis <- multisepCrisprList[multisepCrisprList$crispr_gene %in%
                               rownames(depMapCRISPRscores_subset), ][1:15, ]

mCrisprTS <- mts_CrisprTS(
  resultList = multisepList,
  crisprMatrix = depMapCRISPRscores_subset,
  tissueMatrix = depMapTissue_subset,
  fcVal = -0.1,
  pVal = 0.25,
  allDisRes = allDis)
```

`mts_formatMatrix` *Format a matrix into a list of data frames that capture gene functional status*

Description

This function transforms an 'impact matrix' into a list of data frames, where each gene is represented by a separate data frame to provide information on the sample name, functional impact status (e.g. high/low impact mutations; high/low methylation levels) and impact group (1 for HIGH, 2 for LOW). This function should be used to format any gene by sample matrix to be used as input to the `mts_patternDetection` function in order to predict gene dependency relationship patterns.

Usage

```
mts_formatMatrix(matrix, cores)
```

Arguments

`matrix`	A binary matrix in gene by sample format, where rownames correspond to genes (or probesets) and colnames correspond to sample names. The matrix must have either 1 or 2 values to represent impact groups (1 for HIGH, 2 for LOW).
`cores`	The number of compute cores to use. Defaults to 1.

Details

Formats an input matrix into the correct format for input into the `mts_patternDetection()` function.

Value

A list of dataframes, where each gene corresponds to a separate dataframe and contains the impact groups for samples (cell lines). Each dataframe has the following three columns:

`Samples`	Sample/cell line names associated with the gene (or probesets etc.).
`Impact`	An indicator for the impact status, which can be "HIGH" or "LOW".
`ImpactGroup`	The numeric impact assignment for each sample, values of 1 represent "HIGH" impact and 2 represents "LOW" impact.

See Also

`mts_patternDetection`

Examples

```
data("MUT_impactMatrix")
Formatted_MUTimpactMatrix = mts_formatMatrix(matrix = MUT_impactMatrix, cores = 1)
```

`mts_GeneDepCrispr` *CRISPR and gene expression gene-dependency prediction (SynLeGG)*

Description

Run the CRISPR and expression gene-dependency pipeline, optionally with tissue data. Minimally, an expression matrix and a CRISPR matrix must be provided.

Usage

```
mts_GeneDepCrispr(exprsMatrix = exprsMatrix, crisprMatrix = crisprMatrix,
cores = cores, fcVal = fcVal, pVal = pVal, tissueMatrix = tissueMatrix)
```

Arguments

`exprsMatrix`	Log2 gene expression matrix in a gene by sample format where rownames are genes and the column names are samples.
`crisprMatrix`	Log2 crispr score matrix in a gene by sample format where rownames are genes and the column names are samples.
`cores`	The number of compute cores to use, default is 1.
`fcVal`	A log 2 fold change value filter, default is -0.1. Running with negative fold-change values (<0) allows prediction of synthetic lethal relationships, specifying positive fold-change values (>0) corresponds to 'induced dependency' relationships.
`pVal`	A p value filter, default is 0.1.
`tissueMatrix`	Matrix of two columns where column 1 is a sample ID and column 2 is a tissue ID.

Value

A list containing up to two tables:

`CRISPR_Results`	Cluster assignment table output from the `mts_Crispr` function.
`CRISPR_TS_Results`	Output from the `mts_CrisprTS` function (generated if a `tissueMatrix` is supplied).

Examples

```
data(depMapCRISPRscores_subset)
data(depMapXPR_subset)
data(depMapTissue_subset)
exprsSub  <- depMapXPR_subset["PCYT1B",]
crisprSub <- depMapCRISPRscores_subset["PCYT1A",]
mts_GeneDepCrispr(
exprsMatrix = exprsSub, crisprMatrix = crisprSub,
cores = 1, fcVal = -0.5, pVal = 0.25, tissueMatrix = depMapTissue_subset)
```

mts_GeneDepID *CRISPR vs gene expression induced dependency pipeline (SynLeGG)*

Description

Run the MultiSEp CRISPR vs gene expression induced dependency pipeline, optionally with tissue data. Minimally, a gene expression matrix and a CRISPR matrix must be provided.

Usage

```
mts_GeneDepID(exprsMatrix = exprsMatrix, crisprMatrix = crisprMatrix,
cores = cores, fcVal = fcVal, pVal = pVal, tissueMatrix = tissueMatrix)
```

Arguments

exprsMatrix	Log2 gene expression matrix in a gene by sample format where rownames are genes and the column names are samples.
crisprMatrix	Log2 CRISPR score matrix in a gene by sample format where rownames are genes and the column names are samples.
cores	The number of compute cores to use, defaults to 1.
fcVal	A log 2 fold change value filter, defaults to 0.1 (positive fold-change values are expected to reflect induced dependency relationships).
pVal	A p value filter, defaults to 0.1.
tissueMatrix	Matrix with two columns where column 1 is a sample ID and column 2 is a tissue ID.

Value

A list containing up to 2 tables:

CRISPR_Results	Cluster assignment table output from the mts_InducedDependency function.
CRISPR_TS_Results	Output from the mts_CrisprTS function (generated if a tissueMatrix is supplied).

Examples

```
data(depMapCRISPRscores_subset)
data(depMapXPR_subset)
data(depMapTissue_subset)
mtsGeneDep_ID_Results <- mts_GeneDepID(exprsMatrix=depMapXPR_subset[1:3,],
crisprMatrix=depMapCRISPRscores_subset[1:10,], cores=1, fcVal=0.1, pVal=0.25,
tissueMatrix=depMapTissue_subset)
```

`mts_GeneDepMutation` *Mutation vs gene expression gene-dependency prediction*

Description

Run the MultiSEp mutation gene-dependency pipeline, with tissue data. As a minimum an expression matrix, mutation matrix and a tissue matrix must be provided (please see details below).

Usage

```
mts_GeneDepMutation(exprsMatrix = exprsMatrix, mutMatrix= mutMatrix,
cores = cores, pVal = pVal, tissueMatrix = tissueMatrix)
```

Arguments

`exprsMatrix`	Log2 gene expression matrix in a gene by sample format where row names are genes and the column names are samples.
`mutMatrix`	A mutation score matrix in gene by sample format, where row names correspond to genes and column names correspond to sample names. The value should be either 'WT' or a mutation detail (e.g. amino acid change or coding sequence change). For example, please see the depMapMUT_subset object supplied with this package.
`cores`	The number of compute cores to use, default is 1.
`pVal`	A p value filter, default is 0.1.
`tissueMatrix`	Matrix of two columns where column 1 is a sample ID and column 2 is a tissue ID.

Value

A data frame containing the results of MultiSEp analysis of mutations by gene expression clusters, with the following columns:

`mrna_gene`	Identifier of the gene expression gene
`mutation_gene`	Identifier of the mutation gene
`tissue`	Description of the tissue(s) evaluated
`chiSqPvalue`	p-value from chi-squared test, assessing the distribution of mutation classes across the gene expression clusters
`mutation_per_mode`	Total number of mutated samples in the multisep gene expression cluster (or 'mode'), for each mutation_gene.

Examples

```
data(depMapXPR_subset)
data(depMapMUT_subset)
data(depMapTissue_subset)

mtsGeneDepMutResults <- mts_GeneDepMutation(
  exprsMatrix=depMapXPR_subset[1,],
  tissueMatrix = depMapTissue_subset,
  mutMatrix = depMapMUT_subset[1:2, ], pVal=0.1)
```

---

mts_genepairsChunkGeneration

Generate `.RData` *files containing subsets of gene pairs*

Description

Generates `.RData` files ('chunks') containing subsets of gene pairs. These subsets are useful for MultiSEp applications in high-performance computing (HPC) environments, where each file ('chunk') may be analysed by one task in an array job. This function takes one or two list objects containing cluster assignments and generates `.RData` files containing subsets of the full gene pair list.

Each file contains a portion of the gene pairs that can be processed independently, enabling `mts_patternDetection` to run in parallel across array jobs. This facilitates scalable analysis of large gene pair sets.

Usage

```
mts_genepairsChunkGeneration(
  genepairs = NULL,
  mixModelClusters1,
  mixModelClusters2=NULL,
  num_tasks,
  output_dir = tempdir(),
  cores
)
```

Arguments

`genepairs`	Optional. A data frame containing two columns of gene names. If `genepairs` is not supplied or is set to `NULL`, gene pairs are generated automatically from the genes present in the supplied cluster assignment object or objects. For unidirectional input using a single cluster assignment object, both genes in each pair must be present in `mixModelClusters1`. For bidirectional input using two cluster assignment objects, the first column must contain genes from `mixModelClusters1`, and the second column must contain genes from `mixModelClusters2`.

mixModelClusters1
: A list object containing cluster assignments generated by mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition, or mts_formatMatrix. This object provides the gene set used to generate gene pairs when genepairs is not supplied or is set to NULL.

mixModelClusters2
: Optional. A second list object containing cluster assignments generated by mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition or mts_formatMatrix. If supplied, gene pairs are generated between genes in mixModelClusters1 and genes in mixModelClusters2. If not supplied, gene pairs are generated within mixModelClusters1.

num_tasks
: The number of .RData files into which the gene pairs are divided.

output_dir
: Directory where the .RData files are saved. If the specified directory does not exist, an attempt is made to create the directory according to the given path. Defaults to a temporary directory.

cores
: Number of compute cores to use. Defaults to 1.

Value

Splits the gene pairs into num_tasks chunks and writes each to the output directory as a 'genepairs_chunk_<i>.RData' file. Returns a list of length num_tasks whose elements are each NULL.

See Also

mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition, mts_formatMatrix, mts_patternDetection

Examples

```
data(depMapXPR_subset)
mixModelClusters = mts_mixModelCluster_XPR(
  dataMatrix = depMapXPR_subset[1:5,],
  cores=1
)

# Using a custom output directory to save the .RData files (chunks)
# Here we write to a temporary directory for the example
output_dir <- tempdir()

# Generate Gene Pairs and Save to 3 Files
mts_genepairsChunkGeneration(
  mixModelClusters1 = mixModelClusters,
  num_tasks = 3,
  output_dir = output_dir
)
```

`mts_InducedDependency` *Predict 'induced dependency' relationships, evaluating differences in CRISPR scores for gene A between expression clusters for gene B*

Description

Gene expression clusters enable partitioning of effect scores from CRISPR screen data, and therefore may reveal pairwise gene dependencies. `mts_InducedDependency` performs statistical evaluation of induced dependency relationships (enrichment) by calculating a log2 fold-change and two-tailed T-test p-value between the CRISPR scores for each consecutive pair of mRNA expression clusters. 'Induced dependency' relationships in this function are defined by lower CRISPR scores (corresponding to reduced cell survival/growth) with higher gene expression values; corresponding to filtering by positive fold-change values (>0). Q-values are also calculated (using false discovery rate correction). We strongly recommend taking the q-values as a baseline for determining statistical significance, indeed it is crucially important to correct for multiple hypothesis testing to ensure the statistical validity of results.

Usage

```
mts_InducedDependency(resultList = resultList, exprsMatrix = exprsMatrix,
crisprMatrix = crisprMatrix, fcVal = fcVal, pVal = pVal, cores = cores)
```

Arguments

`resultList`	A result list object produced by the `mts_clusterAvg` function.
`exprsMatrix`	A log2 gene expression matrix in gene by sample format, where row names correspond to genes (or probesets) and column names correspond to sample names.
`crisprMatrix`	A crispr score matrix in gene by sample format, where row names correspond to genes (or guide RNAs) and column names correspond to sample names. The score should be derived from a pooled or arrayed CRISPR screen where negative scores correspond to dependency (e.g. cell death/stasis) and positive scores correspond to outgrowth.
`fcVal`	The log2 fold-change threshold, defaults to 0.1. Please note that positive fold-change values (>0) should be given for assessment of induced dependency relationships.
`pVal`	The p-value threshold, defaults to 0.1.
`cores`	The number of compute cores to use, defaults to 1.

Details

Similar to the methodology applied for CRISPR analysis in the www.overton-lab.uk/synlegg resource (Wappett et al. 2021); developments include an improved approach for discovery of the gene expression clusters.

Value

A data frame containing results for gene expression and CRISPR gene pairs that pass the p-value and log2 fold-change threshold values used. The data frame contains the following columns:

`mrna_gene`	Identifier of the gene expression gene
`crispr_gene`	Identifier of the CRISPR gene
`num_modes`	Number of gene expression clusters
`nMode1-5`	Number of cell lines in each gene expression cluster
`mMode1-5`	Mean CRISPR score of the CRISPR gene in each gene expression cluster
`shift2_1`	Fold-change of CRISPR scores between expression modes (clusters) 2 and 1
`shift3_2`	Fold-change of CRISPR scores between expression modes (clusters) 3 and 2. NA if mRNA gene has 2 gene expression clusters
`shift4_3`	Fold-change of CRISPR scores between expression modes (clusters) 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
`shift5_4`	Fold-change of CRISPR scores between expression modes (clusters) 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters
`pvalue2_1`	P-value of CRISPR scores between expression modes (clusters) 2 and 1
`pvalue3_2`	P-value of CRISPR scores between expression modes (clusters) 3 and 2. NA if mRNA gene has 2 gene expression clusters
`pvalue4_3`	P-value of CRISPR scores between expression modes (clusters) 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
`pvalue5_4`	P-value of CRISPR scores between expression modes (clusters) 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters
`qvalue2_1`	Q-value of CRISPR scores between expression modes (clusters) 2 and 1
`qvalue3_2`	Q-value of CRISPR scores between expression modes (clusters) 3 and 2. NA if mRNA gene has 2 gene expression clusters
`qvalue4_3`	Q-value of CRISPR scores between expression modes (clusters) 4 and 3. NA if mRNA gene has 2 or 3 gene expression clusters
`qvalue5_4`	Q-value of CRISPR scores between expression modes (clusters) 5 and 4. NA if mRNA gene has 2,3 or 4 gene expression clusters

References

Wappett *et al.* (2021) [*Nucleic Acids Research* 49, W613-W618] doi:10.1093/nar/gkab338

See Also

`mts_clusterAvg`

Examples

```
data(depMapXPR_subset)
data(depMapCRISPRscores_subset)
data(multisepList)

resultListSmall <- lapply(multisepList["DNAJC15"], function(g) {
    keep <- intersect(rownames(g[[1]]), rownames(depMapCRISPRscores_subset))
    g[[1]] <- g[[1]][keep, , drop = FALSE]
    g
  })

mIDOut <-  mts_InducedDependency(
    resultList   = resultListSmall,
    exprsMatrix  = depMapXPR_subset,
    crisprMatrix = depMapCRISPRscores_subset
  )
```

`mts_mixModelCluster` *Cluster continuous data with Gaussian Mixture Modelling*

Description

Assigns samples into clusters by unsupervised analysis of continuous values for a given gene, flexibly capturing between two to five different sub-populations of cell lines. Input data could be transcriptome, proteome or other continuous data that informs on gene functional status. Each sample (for example, a cell line) is assigned a probability of belonging to each cluster.

Usage

```
mts_mixModelCluster(dataMatrix, cores)
```

Arguments

`dataMatrix`	Could be gene expression, protein concentration (or other continuous data), in gene by sample format, where row names correspond to genes (or proteoforms, probesets etc.) and column names correspond to sample names.
`cores`	The number of compute cores to use, default is 1.

Details

Gaussian mixture modelling with Expectation-Maximisation (EM) discovers clusters, or modes, where cardinality is determined by Bayesian Information Criterion regularisation. The method from Lubbock et al. 2013 was adapted for application to gene dependency prediction. Contiguous clusters are ensured and the ordering of clusters is taken from the value one standard deviation below the mean rather than the mean value. For example, occasionally there is overlap between clusters

determined by EM such that the samples contained in one cluster encompass the range of values within a second cluster; hence samples from the first cluster are both below the minimum value and above the maximum value for any sample in the second cluster. Downstream analysis based upon the clusters, for example construction of contingency tables, assumes contiguous clusters and this property is ensured by assigning cluster boundaries with the average of minimum and maximum values from neighbouring clusters.

Value

A list object for each gene (or molecular species e.g. proteoform) in the dataMatrix object

`Samples` Sample identifiers.

`Values` The analysed value for each sample, for example from the input log2 gene expression value.

`Cluster_Assignment`
The assigned cluster value for each sample.

References

Lubbock *et al.* (2013) TMA Navigator: network inference, patient stratification and survival analysis with tissue microarray data [*Nucleic Acids Research* 41, W562-W568] doi:10.1093/nar/gkt529

See Also

`mts_patternDetection`, `mts_genepairsChunkGeneration`, `mts_plotClusterDistribution`, `mts_genepairsChunkGene`

`mts_plotMutation`

Examples

```
data("depMapXPR_subset")
mixModelClusters <- mts_mixModelCluster(dataMatrix = depMapXPR_subset[1,])
```

---

mts_mixModelCluster_XPR

Cluster gene expression data with Gaussian Mixture Modelling

Description

Assigns samples into clusters by unsupervised analysis of continuous values for a given gene, flexibly capturing between two to five different sub-populations of samples. This function is designed for RNA-seq data but may be adapted to other data types. Each sample (for example, a cell line) is assigned a probability of belonging to each cluster. Filtering steps include ensuring a gene is present in an adequate number of samples with an expression value above the background.

Usage

```
mts_mixModelCluster_XPR(
  dataMatrix,
  GeneXPRthresh = 3.321928,
  NumSampleThresh = 20,
  cores
)
```

Arguments

dataMatrix	A gene expression matrix in gene by sample format, where row names correspond to genes and column names correspond to sample names.
GeneXPRthresh	A gene expression value that retains genes expressed in samples above that value. Defaults to 3.321928 which corresponds to log2 transformed data. A TPM value of 10 is recommended for non-log transformed data.
NumSampleThresh	Determines how many samples a gene should be expressed in, with the expression value specified by GeneXPRthresh. Defaults to 20 samples.
cores	The number of compute cores to use, default is 1.

Details

Gaussian mixture modelling with Expectation-Maximisation (EM) discovers clusters, or modes, where cardinality is determined by Bayesian Information Criterion regularisation. The method from Lubbock et al. 2013 [*Nucleic Acids Research* 41, W562-W568] was adapted for application to gene dependency prediction. Contiguous clusters are ensured and the ordering of clusters is taken from the value one standard deviation below the mean rather than the mean value. For example, occasionally there is overlap between clusters determined by EM such that the samples contained in one cluster encompass the range of values within a second cluster; hence samples from the first cluster are both below the minimum value and above the maximum value for any sample in the second cluster. Downstream analysis based upon the clusters, for example construction of contingency tables, assumes contiguous clusters and this property is ensured by assigning cluster boundaries with the average of minimum and maximum values from neighbouring clusters.

Value

A list object for each gene (or molecular species e.g. proteoform) in the dataMatrix object

Samples	Sample identifiers.
Values	The analysed value for each sample, for example from the input log2 gene expression value.
Cluster_Assignment	The assigned cluster value for each sample.

References

Lubbock *et al.* (2013) TMA Navigator: network inference, patient stratification and survival analysis with tissue microarray data [*Nucleic Acids Research* 41, W562-W568] doi:10.1093/nar/gkt529

See Also

mts_patternDetection, mts_formatMatrix

Examples

```
data("depMapXPR_subset")
mixModelClusters <- mts_mixModelCluster_XPR(dataMatrix = depMapXPR_subset[1,])
```

---

mts_Mutation *Integrate gene cluster assignments with mutation data*

Description

MultiSEp detects mutations that are enriched in gene clusters (for example, from mRNA expression); providing insight into the genetic background relevant to candidate 'Achilles Heel' relationships suggested by analysis of the CRISPR screen and transcriptome data. A chi-squared test is performed for each gene to derive p-values for the enrichment of mutation classes across the gene expression clusters on a per-tissue basis.

Usage

```
mts_Mutation(resultList = resultList, mutMatrix = mutMatrix, tissueMatrix = tissueMatrix,
cores = cores, pVal = pVal)
```

Arguments

resultList	A result list object produced by the mts_mixModelCluster function.
mutMatrix	A mutation score matrix in gene by sample format, where row names correspond to genes and column names correspond to sample names. The value should be either 'WT' or a mutation detail (e.g. amino acid change or coding sequence change).
tissueMatrix	A data frame containing 2 columns where column 1 is the sample id and column 2 is the tissue id.
pVal	A p-value filter, the default is 0.01.
cores	The number of compute cores to use, the default is 1.

Value

A data frame containing the results of MultiSEp analysis of mutations by gene expression clusters, with the following columns:

mrna_gene	Identifier for the genes or other molecular species provided in the resultList cluster object
mutation_gene	Identifier for the genes with mutation data
tissue	Description of the tissue evaluated

chiSqPvalue p-value from chi-squared test, assessing the distribution of mutation classes across the clusters

mutation_per_mode
Total number of mutated samples in the clusters (modes) for the given mutation_gene.

See Also

mts_mixModelCluster

Examples

```
data(depMapXPR_subset)
data(depMapMUT_subset)
data(depMapTissue_subset)
mixModelClusters <- mts_mixModelCluster(dataMatrix = depMapXPR_subset[1,])

multisepMut <- mts_Mutation(
  resultList = mixModelClusters[1],
  mutMatrix = depMapMUT_subset,
  tissueMatrix = depMapTissue_subset,
  pVal = 0.01
)
```

---

mts_omics *Predict synthetic lethality and other dependency relationships from mutually exclusive loss of function patterns*

Description

Compare any 'omics datatype either against itself (e.g. XPR_vs_XPR) or against another data-type (e.g. XPR_vs_CRISPR). Analysis of dependency relationships across many different data types is possible. For example: mutation, RNA-seq, proteomics, methylation data, histone marks, CRISPR scores, drug sensitivity, potentially metabolomics etc. This function takes as input either one dataset (for symmetrical analysis such as XPR_vs_XPR) or two datasets (non-symmetrical analysis such as XPR_vs_mutation).

Usage

```
mts_omics(
  dataMatrix = NULL,
  dataMatrix2 = NULL,
  mixModelClusters1 = NULL,
  mixModelClusters2 = NULL,
  categorical1 = FALSE,
  categorical2 = FALSE,
  genepairs = NULL,
```

```
  SyntheticLethalityPrediction = TRUE,
  p_adjustMethod = c("BY","BH"),
  qVal,
  directionality = c("depletion", "enrichment"),
  include_reverse_pairs = TRUE,
  effectsize = TRUE,
  effectsize_threshold = NULL,
  verbose = FALSE,
  cores)
```

Arguments

dataMatrix	A matrix in gene by sample format, where rownames typically correspond to genes (or other data features) and colnames correspond to sample names. May be omitted if mixModelClusters1 is provided.
dataMatrix2	Optional. A matrix where rownames correspond to genes (or other data features) and colnames correspond to sample names. If provided, the function will perform bidirectional analysis to assess associations between values in dataMatrix and dataMatrix2. If not provided, the function will perform one-directional analysis. May be omitted if mixModelClusters2 is provided.
mixModelClusters1	Optional. A pre-computed cluster list for data type 1, as produced by mts_mixModelCluster() (continuous data) or mts_formatMatrix() (categorical data, e.g. mutation impact). When supplied, dataMatrix and categorical1 are ignored. Exactly one of dataMatrix or mixModelClusters1 must be provided.
mixModelClusters2	Optional. A pre-computed cluster list for data type 2, as for mixModelClusters1. When supplied, dataMatrix2 and categorical2 are ignored.
categorical1	If TRUE, dataMatrix is treated as a categorical impact matrix and is processed with mts_formatMatrix() (which expects a numeric 1/2 matrix where 1 = HIGH impact / loss of function and 2 = LOW impact / wild type) instead of Gaussian mixture modelling. Ignored when mixModelClusters1 is supplied. Defaults to FALSE.
categorical2	As categorical1, but for dataMatrix2. Defaults to FALSE.
genepairs	Optional. If the argument is not called (left as NULL) then gene pairs will be generated based on the bidirectional or one directional arguments specified. If provided, genepairs must be a data frame containing two columns of gene names. - For one-directional analysis (single mixModelClusters object), both genes in each pair must exist in that object. - For bidirectional analysis (two mixModelClusters objects), the first column must contain genes from mixModelClusters1, and the second column must contain genes from mixModelClusters2. All gene pairs must have valid cluster values in the respective mixModelClusters input(s).
SyntheticLethalityPrediction	If set to TRUE the function will exclusively predict synthetic lethal patterns (only analysing samples in the combination of the lowest value clusters for each gene

pair). If set to FALSE the function will investigate all possible gene dependency patterns across all cluster combinations for each gene pair. Defaults to TRUE.

p_adjustMethod The p.adjust method to adjust p-values for multiple comparisons. The adjustment methods include "BH" (Benjamini & Hochberg (1995), doi:10.1111/j.25176161.1995.tb02031.x) or its alias "fdr", and "BY" (Benjamini & Yekutieli (2001), doi:10.1214/aos/1013699998). Defaults to "BY".

directionality Considers directionality across the entire contingency table. Two options are available: "enrichment" and "depletion". For example when Synthetic Lethality occurs with CRISPR data we expect enrichment of samples in the bottom left contingency table cell while with other datatypes such as expression the expectation for Synthetic Lethality is a depletion of samples in the bottom-left contingency table cell. Defaults to "depletion".

include_reverse_pairs
If TRUE, both GeneA-GeneB and GeneB-GeneA will be tested when two mixModelCluster objects are provided. If FALSE, only unique one-way pairs are used. Defaults to TRUE.

qVal A q value filter. Defaults to 0.05.

effectsize If TRUE, filtering based on effect size is applied. Defaults to TRUE.

effectsize_threshold
A numeric value between 0 and 1. Gene pairs with effect sizes below this threshold will be excluded from the final results. If set to FALSE, no filtering based on effect size is applied. The default value depends on the analysis type:
- One mixModelCluster object: default = 0.9621389
- Two mixModelCluster objects: default = 0.5029284

cores The number of compute cores to use. Defaults to 1.

verbose If TRUE, the function will print detailed progress messages during execution, including progress updates and any skipped gene pairs. Defaults to FALSE.

Value

A data frame with one row per evaluated gene pair, containing the gene names (Gene1, Gene2), the observed count (Actual_Count), the number of cluster combinations, the sample count, the expected count (Expected_Count), and the p_value and false-discovery-rate q_value (plus an Effect_Size column when effectsize = TRUE). An empty data frame with these columns is returned when no gene pair is valid.

See Also

mts_mixModelCluster, mts_mixModelCluster_XPR, mts_patternDetection

Examples

```
data(mixModelClusters_depMapXPR)
Unidirectional_depletion_SL_XPRvsXPR = mts_omics(
  mixModelClusters1 = mixModelClusters_depMapXPR,
  qVal=1,
  cores=1
```

```
)

data(depMapXPR_subset)
data(depMapCRISPRscores_subset)
Bidirectional_enrichment_SL_XPRvsCRISPR = mts_omics(
  dataMatrix = depMapCRISPRscores_subset[3,],
  dataMatrix2 = depMapXPR_subset[2,],
  directionality = "enrichment",
  qVal=1,
  cores=1
)

GDR_patternAnalysis_unidirectional_depletion_XPRvsXPR = mts_omics(
  dataMatrix = depMapXPR_subset[1:2,],
  SyntheticLethalityPrediction = FALSE,
  effectsize=FALSE,
  qVal=1,
  cores=1
)


# Mutation (categorical) with gene expression.
# depMapMUT_small contains string variant calls ("WT" or e.g. "p.R167W");
# convert to the 1/2 impact convention expected by mts_formatMatrix().


data(depMapMUT_small)
data(depMapXPR_small)
mutImpact <- as.data.frame(
  ifelse(as.matrix(depMapMUT_small) == "WT", 2L, 1L),
  stringsAsFactors = FALSE, check.names = FALSE)
rownames(mutImpact) <- rownames(depMapMUT_small)

SL_mut_vs_xpr = mts_omics(
  dataMatrix   = mutImpact,
  dataMatrix2  = depMapXPR_small[3,],
  categorical1 = TRUE,
  directionality = "depletion",
  qVal         = 1,
  effectsize   = FALSE,
  cores        = 1
)

# Or with pre-computed mutation 'clusters'
mutClusters <- mts_formatMatrix(matrix = mutImpact, cores = 1)
SL_mut_vs_xpr_pre = mts_omics(
  mixModelClusters1 = mutClusters,
  dataMatrix2       = depMapXPR_small[3,],
  directionality    = "depletion",
  qVal              = 1,
  effectsize        = FALSE,
  cores             = 1
)
```

```
# Using only mutation (or other categorical) data
data(depMapMUT_small)
mutImpactSmall <- as.data.frame( # make a binary matrix
  ifelse(as.matrix(depMapMUT_small) == "WT", 2L, 1L),
  stringsAsFactors = FALSE, check.names = FALSE)
rownames(mutImpactSmall) <- rownames(depMapMUT_small)

mutClasses <- mts_formatMatrix(matrix = mutImpactSmall[,1:5], cores = 1)

SL_mut_only = mts_omics(
  mixModelClusters1 = mutClasses,
  directionality    = "depletion",
  qVal              = 1,
  effectsize        = FALSE,
  cores             = 1
  )
```

`mts_pairwiseExpressionPlot` *Visualise pairwise gene expression data with MultiSEp cluster boundaries.*

Description

Produces a scatterplot of two selected genes on the x and y axis, the boundaries between MultiSEp clusters are shown as blue lines.

Usage

```
mts_pairwiseExpressionPlot(exprsMatrix, gene1, gene2)
```

Arguments

`exprsMatrix`	Log2 gene expression matrix in a gene by sample format where rownames are genes and the column names are samples.
`gene1`	User specified gene 1. Must match a gene in the exprsMatrix.
`gene2`	User specified gene 2. Must match a gene in the exprsMatrix.

Value

Produces a scatter plot (base R graphics) of the two genes' log2 expression, with mixture-model cluster boundaries drawn as reference lines.

Examples

```
data(depMapXPR_subset)
mts_pairwiseExpressionPlot(exprsMatrix = depMapXPR_subset, gene1 = "STX2", gene2="VRK2")
```

`mts_patternDetection` *Predict Gene Dependency Relationship Patterns, including Synthetic Lethality*

Description

Enumerates different types of gene dependency relationships (GDRs). Can optionally detect SL where there is a depletion of samples in the bottom left cell of the contingency table (if low-scoring values indicate a reduction or loss of gene function). Performs statistical analysis of depletion with the binomial test. The expected proportion of samples in each contingency table cell is calculated with reference to the size of the clusters.

Usage

```
mts_patternDetection(
  genepairs = NULL,
  mixModelClusters1,
  mixModelClusters2 = NULL,
  SyntheticLethalityPrediction = TRUE,
  p_adjustMethod = c("BY","BH"),
  qVal,
  effectsize = TRUE,
  effectsize_threshold = NULL,
  include_reverse_pairs = FALSE,
  directionality = c("depletion", "enrichment"),
  verbose = FALSE,
  cores)
```

Arguments

`mixModelClusters1`
: List object containing multisep cluster assignments output by the `mts_mixModelCluster` function. The list object can also be generated from the `mts_mixModelCluster` function using categorical variables that have been formatted by the `mts_formatMatrix` function.

`mixModelClusters2`
: Optional. A list object produced by the `mts_mixModelCluster` function. The list object can also include categorical variables formatted by the `mts_formatMatrix` function.
If provided, bidirectional analysis is carried out to assess associations between `mixModelClusters1` and `mixModelClusters2` (i.e. two contingency tables per gene pair).
If not provided, unidirectional analysis is carried out for `mixModelClusters1` against itself (only one contingency table is required per gene pair).

`genepairs` Optional. If the argument is not called or left as `NULL` then gene pairs will be generated based on the bidirectional or unidirectional arguments specified. If provided, `genepairs` must be a data frame containing two columns of gene names.
- For one-directional analysis (single mixModelClusters object), both genes in each pair must exist in that object.
- For bidirectional analysis (two mixModelClusters objects), the first column must contain genes from `mixModelClusters1`, and the second column must contain genes from `mixModelClusters2`.
All gene pairs must have valid cluster values in the respective mixModelClusters input(s).

`SyntheticLethalityPrediction`
If set to `TRUE` synthetic lethal patterns are evaluated (by analysis of the bottom left contingency table cell). If `FALSE`, gene dependency relationships are evaluated for all combinations of clusters for each gene pair, for example including synthetic lethality and induced dependency. Defaults to `TRUE`.

`p_adjustMethod` The method for false discovery rate adjustment of p-values for multiple comparisons. The options are: `"BH"` (Benjamini & Hochberg (1995), doi:10.1111/j.25176161.1995.tb02031.x) or its alias `"fdr"`, and `"BY"` (Benjamini & Yekutieli (2001), doi:10.1214/aos/1013699998). Defaults to `"BY"` which is recommended due to the expectation that genes may participate in multiple gene pairs.

`qVal` The q value filter threshold value, defaults to 0.05.

`effectsize` If `TRUE`, filtering based on effect size is applied. Defaults to `TRUE`.

`effectsize_threshold`
A numeric value between 0 and 1. Gene pairs with effect sizes below this threshold will be excluded from the final results. If set to `FALSE`, no filtering based on effect size is applied. The default value depends on the analysis type:
- One mixModelCluster object: default = 0.9621389
- Two mixModelCluster objects: default = 0.5029284 Please note that these thresholds were calibrated in XPR_vs_XPR (one object) and XPR_vs_CRISPR analysis (two objects); please see preprint/publication for details (McKie et al 2026).

`include_reverse_pairs`
If `TRUE`, both GeneA-GeneB and GeneB-GeneA will be tested when two mixModelCluster objects are provided.
If `FALSE`, only unique one-way pairs are used. Defaults to `FALSE`

`directionality` Valid options are `"enrichment"` or `"depletion"`. Considers the directionality of gene dependency relationships: assessed as either `"depletion"` (corresponding to synthetic lethality in most data types) or `"enrichment"` in contingency table cells.
For example when synthetic lethality occurs with CRISPR data and expression we expect `"enrichment"` of samples in the bottom left contingency table cell because low CRISPR scores represent cell death (or possibly cell stasis). For other data types where low values indicate loss of (or reduced) gene function, such as gene expression, the expectation for synthetic lethal relationships is a `"depletion"` of samples in the bottom-left contingency table cell. Defaults to `"depletion"`.

verbose	If TRUE, the function will print detailed progress messages during execution, including progress updates and any skipped gene pairs. Defaults to FALSE.
cores	The number of compute cores to use, defaults to 1.

Value

A data frame with one row per evaluated gene pair, containing the gene names (Gene1, Gene2), the observed count (Actual_Count), the number of cluster combinations, the sample count, the expected count (Expected_Count), and the p_value and false-discovery-rate q_value (plus an Effect_Size column when effectsize = TRUE). An empty data frame with these columns is returned when no gene pair is valid.

See Also

mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition, mts_genepairsChunkGeneration, mts_formatMatrix, mts_plotClusterDistribution

Examples

```
data("mixModelClusters_depMapXPR")

mixturemodelClusters <- mts_mixModelCluster_XPR(dataMatrix = depMapXPR_subset[c(1:12, 20), ])
# partition into two 'clusters' at -0.5
CRISPR_clusters = mts_crisprPartition(dataMatrix = depMapCRISPRscores_subset[1:12, ])

# unidirectional (i.e. symmetrical) evaluation
# of synthetic lethality depletion relationships with expression data
predicted_XPR_SL = mts_patternDetection(
  mixModelClusters1=mixModelClusters_depMapXPR,
  qVal=1
)

# unidirectional synthetic lethal depletion prediction with expression data and
# specified gene pairs, using BH FDR correction, filtering using the q-value
# and the effect size

predicted_XPR_SL2 = mts_patternDetection(
  genepairs = data.frame(Gene1="PSMB8",Gene2="TTC7B"),
  mixModelClusters1 = mixturemodelClusters,
  SyntheticLethalityPrediction = TRUE,
  p_adjustMethod="BH",
  qVal = 0.05,  effectsize=TRUE)

# bidirectional synthetic lethal depletion prediction with expression and CRISPR data
# and including reverse pairs

Predicted_SL3 = mts_patternDetection(
  mixModelClusters1=mixturemodelClusters[1:10],
  mixModelClusters2=CRISPR_clusters[1:10],
  include_reverse_pairs = FALSE, effectsize=FALSE, qVal =0.05,
  directionality = "enrichment"
)
```

```
# tissue specific analysis
# 1. subset cell lines of interest using mapping file

Lung_Cancer_cellLines <- depMapTissue_subset$cell_line[
  depMapTissue_subset$tissue == "Lung Cancer"]

# 2. subset XPR mixture model result list
LungCancer_XPRmixturemodelClusters <- lapply(mixturemodelClusters, function(GMM) {
    GMM[GMM$Sample %in% Lung_Cancer_cellLines, ]
})

# 3. subset CRISPR mixture model result list
LungCancer_CRISPR_clusters <- lapply(CRISPR_clusters, function(GMM) {
  GMM[GMM$Sample %in% Lung_Cancer_cellLines, ]
})

TissueSpecific_SL = mts_patternDetection(mixModelClusters1 = LungCancer_CRISPR_clusters,
                         mixModelClusters2 = LungCancer_XPRmixturemodelClusters,
                         include_reverse_pairs = TRUE,
                         SyntheticLethalityPrediction = TRUE,
                         directionality = "enrichment",
                         qVal = 1)

# bidirectional analysis of all depletion gene dependency relationships
# for expression and CRISPR data with BY FDR correction

PredictedGDR = mts_patternDetection(
  mixModelClusters1=mixturemodelClusters[1:10],
  mixModelClusters2=CRISPR_clusters[1:10],
  SyntheticLethalityPrediction = FALSE,
  directionality = "enrichment",
  p_adjustMethod="BY", effectsize = FALSE, qVal = 0.01
)
```

`mts_plotClusterDistribution`

Visualise samples and pairwise gene expression clusters

Description

Produces a scatterplot with two selected genes on the x and y axis, the boundaries are determined by the cluster assignments (typically from Gaussian mixture modelling). Density plots of the clusters are also shown for each gene.

Axis scaling is chosen automatically: when a gene has a long high tail (as with many lncRNAs and other wide dynamic-range genes) that axis is square-root, or pseudo-log, transformed for improved visualisation of low-expression clusters (not compressed). This axis scaling is applied independently to each axis, for whichever gene needs it, whether it is supplied as `gene1` or `gene2`.

Usage

```
mts_plotClusterDistribution(mixModelClusters,
                            mixModelClusters2 = NULL,
                            TScluster1 = NULL,
                            TScluster2 = NULL,
                            gene1, gene2,
                            suppressBlankCanvas = NULL,
                            gene1_datatype = "",
                            gene2_datatype = "")
```

Arguments

mixModelClusters	A list object containing cluster assignments generated by mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition, or mts_formatMatrix.
mixModelClusters2	Optional. A second list object containing cluster assignments generated by mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition or mts_formatMatrix. If supplied, gene pairs are generated between genes in mixModelClusters1 and genes in mixModelClusters2, where the data for gene2 is taken from this object. This functionality is useful for visualisation of more than one data type, for example integrating CRISPR gene effect scores with gene expression data.
TScluster1	Optional. List object containing tissue-specific cluster assignments (and values) for the results of mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition or mts_formatMatrix. Tissue-specific samples are coloured in red in the scatter plot. If the argument is not called or left as NULL then all samples from mixModelClusters will be coloured (according to the pairwise cluster combinations).
TScluster2	Optional. List object containing tissue-specific cluster assignments (and values) for the results of mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition or mts_formatMatrix. Tissue-specific samples are coloured in red in the scatter plot. If the argument is not called or left as NULL then all samples from mixModelClusters will be coloured (according to the pairwise cluster combinations).
gene1	User specified gene 1. Must match a gene in the list object containing cluster assignments (i.e. mixModelClusters).
gene2	User specified gene 2. Must match a gene in the list object containing cluster assignments (i.e. mixModelClusters or mixModelClusters2).
gene1_datatype	Optional. Text provided to this argument is included next to gene1 in the plot, intended to allow display of data-related information for that gene; for example 'log2 gene expression (TPM)'.
gene2_datatype	Optional. Text provided to this argument is included next to gene2 in the plot, intended to allow display of data-related information for that gene; for example 'CRISPR gene effect score'

suppressBlankCanvas
: Optional. Prevents calling plot.new, which makes a blank canvas. Can be set to TRUE if writing a plot to file in order to avoid writing a blank page in addition to the plot. Default value is NULL.

Value

A patchwork object combining a scatter plot with the two panels showing density, which can be printed or saved with ggsave. If suppressBlankCanvas is set, the plot is printed and returned invisibly.

See Also

mts_mixModelCluster, mts_mixModelCluster_XPR, mts_crisprPartition, mts_genepairsChunkGeneration, mts_formatMatrix, mts_plotClusterDistribution,

Examples

```
data("mixModelClusters_depMapXPR")
data("depMapCRISPRscores_subset")
data("depMapTissue_subset")

# Plot with gene expression data for both genes
mts_plotClusterDistribution(
  mixModelClusters = mixModelClusters_depMapXPR,
  gene1 = "NMT2",
  gene2="STX2"
)

##
## Plot gene expression with CRISPR gene effect scores
##
# partition CRISPR scores at -0.5
CRISPR_clusters = mts_crisprPartition(dataMatrix = depMapCRISPRscores_subset[4,])
mts_plotClusterDistribution(mixModelClusters = mixModelClusters_depMapXPR,
                            mixModelClusters2 = CRISPR_clusters,
                            gene1 = "NMT2", gene2 = "NMT1",
                            gene1_datatype="log2 gene expression",
                            gene2_datatype = "CRISPR gene effect score")

##
## Tissue-specific plot
##

mixturemodelClusters <- mts_mixModelCluster_XPR(dataMatrix = depMapXPR_subset[14,])
# partition CRISPR scores at -0.5
CRISPR_clusters = mts_crisprPartition(dataMatrix = depMapCRISPRscores_subset[9,])

# 1. subset cell lines of interest using mapping file
Lung_Cancer_cellLines <- depMapTissue_subset$cell_line[
  depMapTissue_subset$tissue == "Lung Cancer"]
```

```
# 2. subset XPR mixture model result list
LungCancer_XPRmixturemodelClusters <- lapply(mixturemodelClusters, function(GMM) {
    GMM[GMM$Sample %in% Lung_Cancer_cellLines, ]})

# 3. subset CRISPR clusters
LungCancer_CRISPR_Clusters <- lapply(CRISPR_clusters, function(GMM) {
  GMM[GMM$Sample %in% Lung_Cancer_cellLines, ]})

mts_plotClusterDistribution(gene1 = "ACSL1", gene2 = "PSMB5",
                            mixModelClusters = mixturemodelClusters,
                            mixModelClusters2 = CRISPR_clusters,
                            TScluster1 = LungCancer_XPRmixturemodelClusters,
                            TScluster2 = LungCancer_CRISPR_Clusters)
```

`mts_plotCRISPRGeneCluster` *Visualise CRISPR and expression data (SynLeGG methods).*

Description

Produces a boxplot of CRISPR scores split according to MultiSEp clusters (typically gene expression, but other data types may be used to produce the clusters) and with data points coloured by tissue. Alternatively, may produce a waterfall plot of CRISPR scores coloured by the MultiSEp cluster or a density plot of gene expression values (log2 data are given in the example). In the boxplot, CRISPR scores are split according to the MultiSEp gene expression cluster of the specified mrna gene.

Usage

```
mts_plotCRISPRGeneCluster(
  mrna_gene,
  crispr_gene,
  resultList,
  crisprMatrix,
  tissueMatrix,
  diseaseFilter,
  mixModelClustersXPR=NULL,
  plotType
)
```

Arguments

`mrna_gene`	User specified mRNA gene. Must match a gene in the expression matrix.
`crispr_gene`	User specified CRISPR gene. Must match a gene in the CRISPR matrix. Not required for plotType mrna_only.
`resultList`	A result list from the `mts_clusterAvg` function, which specifies gene expression clusters. Required unless the mixModelClustersXPR argument is used.

crisprMatrix	Log2 crispr score matrix in a gene by sample format where rownames are genes and the column names are samples. Not required for plotType mrna_only.
tissueMatrix	Matrix of two columns where column 1 is a sample ID and column 2 is a tissue ID. Not required for plotType mrna_only.
diseaseFilter	Optional user specified tissue filter, defaults to all. Not used with plotType mrna_only.
mixModelClustersXPR	Optional output from mts_mixModelCluster as an alternative to specifying the resultList taken from mts_cluster_Avg.
plotType	User specified plot type. Options are 'Integrated', 'mrna_only' and 'crispr_only'. Defaults to 'Integrated'.

Value

A ggplot object, which can be printed or saved with ggsave. The plot produced depends on plotType:

- "Integrated": a boxplot of crispr_gene CRISPR scores split according to the MultiSEp gene expression clusters of mrna_gene, with data points coloured by tissue (optionally restricted to a single tissue with diseaseFilter).
- "crispr_only": a waterfall plot of crispr_gene CRISPR scores coloured by the MultiSEp cluster.
- "mrna_only": a density plot of mrna_gene log2 expression values split by MultiSEp cluster.

See Also

mts_mixModelCluster, mts_clusterAvg

Examples

```
data(depMapXPR_subset)
data(depMapCRISPRscores_subset)
data(depMapTissue_subset)


clusterAssign <- mts_clusterAvg(
  exprsMatrix=depMapXPR_subset[1:5,],
  crisprMatrix=depMapCRISPRscores_subset,
  cores=1
)

# boxplots
# using clustering results from mts_clusterAvg
mts_plotCRISPRGeneCluster(
  resultList = clusterAssign,
  mrna_gene = "NMT2",
  crispr_gene = "NMT1",
  crisprMatrix = depMapCRISPRscores_subset,
  tissueMatrix = depMapTissue_subset,
```

```
  plotType = "Integrated"
)

# With clusters defined by mts_mixmodelClusters
clusters_depMapXPR <- mts_mixModelCluster(
    dataMatrix = depMapXPR_subset[1:5,],
    cores = 1
)

mts_plotCRISPRGeneCluster(
  mrna_gene = "DNAJC15",
  crispr_gene = "DNAJC19",
  crisprMatrix = depMapCRISPRscores_subset,
  tissueMatrix = depMapTissue_subset,
  mixModelClustersXPR = clusters_depMapXPR,
  plotType = "Integrated"
)

# Waterfall plot of CRISPR scores coloured by gene expression cluster

mts_plotCRISPRGeneCluster(
  mrna_gene = "DNAJC15",
  crispr_gene = "DNAJC19",
  crisprMatrix = depMapCRISPRscores_subset,
  tissueMatrix = depMapTissue_subset,
  mixModelClustersXPR = clusters_depMapXPR,
  plotType = "crispr_only"
)

# density plot of gene expression clusters
mts_plotCRISPRGeneCluster(
  resultList = clusterAssign,
  mrna_gene = "NMT2",
  plotType = "mrna_only"
)

# tissue-specific plots
# Boxplot for lung cancer cell lines
mts_plotCRISPRGeneCluster(
  mrna_gene = "DNAJC15",
  crispr_gene = "DNAJC19",
  crisprMatrix = depMapCRISPRscores_subset,
  tissueMatrix = depMapTissue_subset,
  mixModelClustersXPR = clusters_depMapXPR,
  diseaseFilter = 'Lung Cancer',
  plotType = "Integrated"
)

# waterfall plot for lung cancer cell lines
mts_plotCRISPRGeneCluster(
  mrna_gene = "DNAJC15",
  crispr_gene = "DNAJC19",
  crisprMatrix = depMapCRISPRscores_subset,
```

```
  tissueMatrix = depMapTissue_subset,
  mixModelClustersXPR = clusters_depMapXPR,
  diseaseFilter = 'Lung Cancer',
  plotType = "crispr_only"
)
```

`mts_plotMutation` *Visualise clusters for one gene with the mutation status of another gene.*

Description

Produces boxplots to visualise mutational status for clusters produced by Gaussian Mixture Modelling, for example derived from transcriptome data.

Usage

```
mts_plotMutation(resultList=resultList, mrna_gene = mrna_gene, mut_gene = mut_gene,
mutMatrix = mutMatrix, tissueMatrix = tissueMatrix, diseaseFilter = diseaseFilter)
```

Arguments

`resultList`	A result list object produced by the `mts_mixModelCluster` function.
`mrna_gene`	User specified mRNA gene, must match a gene in the clusters.
`mut_gene`	User specified mutation gene. Must match a gene in the specified mutation matrix.
`mutMatrix`	A mutation score matrix in gene by sample format, where rownames correspond to genes and colnames correspond to sample names. The value should be either 'WT' or a mutation detail (e.g. amino acid change or coding sequence change).
`tissueMatrix`	Matrix of two columns where column 1 is a sample ID and column 2 is a tissue ID.
`diseaseFilter`	User specified tissue filter. Defaults to all.

Value

A `ggplot` object, which can be printed or saved with `ggsave`. The plot is a boxplot of `mrna_gene` log2 expression, split by its expression clusters, with data points coloured by the mutation status (`"WT"` or mutated) of `mut_gene`; optionally restricted to a single tissue when `diseaseFilter` is supplied.

See Also

`mts_mixModelCluster`

Examples

```
data(depMapXPR_subset)
data(depMapMUT_subset)
data(depMapTissue_subset)

mixModelClusters <- mts_mixModelCluster(dataMatrix = depMapXPR_subset[1:2,])

mts_plotMutation(
  resultList= mixModelClusters,
  mrna_gene = "EIF1AY",
  mut_gene = "TP53",
  mutMatrix = depMapMUT_subset,
  tissueMatrix = depMapTissue_subset,
  diseaseFilter = "Pancreatic Cancer")
```

---

mts_targetCoverageFromMutations

Predict the therapeutic population coverage achieved by inhibition of a candidate target gene

---

Description

This function predicts the therapeutic population coverage achieved by inhibiting each gene in the supplied network of (predicted) synthetic lethal (SL) relationships. A patient with inactivating mutations in partner genes of the candidate target gene is predicted to benefit from inhibition of the target gene because the candidate SL relationship implies that cell death will be caused by inactivation of both genes (one by mutation and one by chemical inhibition). This function can take as input a dataframe of gene pairs (edgeData) or the output from the `mts_Mutation()` function, both cases also require mutation calls as input.

Usage

```
  mts_targetCoverageFromMutations(
  mutData,
  edgeData,
  rowNum = nrow(edgeData),
  cores
)
```

Arguments

`mutData`	A mutation score matrix in gene by sample format, where rownames correspond to genes and colnames correspond to sample names. The value should be either `"WT"` or a mutation annotation (e.g. amino acid change or coding sequence change). The mutations are used to evaluate population coverage.
`edgeData`	A dataframe of (predicted) synthetic lethal gene pairs (edges), where gene1 is in column1 and gene2 is in column2. Alternatively, a data frame containing results from the `mts_Mutation` function.

rowNum	The number of gene pairs from edgeData to include in the network used for the population coverage analysis. If set to a value lower than the supplied network (edgeData), the first n edges are taken (i.e. [1:rowNum]). Defaults to nrow(edgeData).
cores	The number of compute cores to use. Defaults to 1.

Value

A data frame containing the following columns:

Gene	Candidate drug target gene.
NeighbourCount	The number of network neighbours
NumSamplesWithMutation	Number of samples with mutation(s) in any of the neighbour genes.
NeighbourMutationFrequency	The proportion of samples with one or more mutations in any of the neighbour genes.
NeighbourGenes	Identifiers of the neighbouring genes.
MutatedSamples	Identifiers of the samples with mutations

Examples

```
data(depMapMUT_subset)
data(mRes)
# An example on the first 5 edges (runs quickly):
TpopCoverage_test <- mts_targetCoverageFromMutations(
  mutData=depMapMUT_subset,
  edgeData=mRes,
  rowNum=5,
  cores=1
)
```

---

mts_tests *Unit Tests for the MultiSEp package*

Description

This function allows unit testing to help verify the MultiSEp package integrity, producing a table that summarises the purpose of each test and the outcome (passed/failed).

Usage

```
mts_tests(mode = c("All", "Demo"),
          cores = 1)
```

Arguments

mode	Perform "All" tests or the "Demo" mode runs two of the tests.
cores	The number of compute cores to use, default is 1.

Details

The unit tests are designed within the testthat framework and incorporate datasets provided in the MultiSEp package. Run time is approximately 230s using one core (4.8GHz processor). The unit tests cover all of the functions in the MultiSEp package and implement individual expectations such as:

- Expect error: verify error handling is implemented correctly.
- Expect named: ensure code will return a character vector of expected names.
- Expect type: check an object returned by a function is inherited from the expected base type.
- Expect equal: verify code returns an expected reference value.
- Expect length: verify code returns a vector with the specified length.
- Expect no error: check the function runs without returning an error.

Value

A data frame containing the test results as follows:

`Test_Case_ID` Identification number of the test case.

`Test_Objective` Purpose of the test case.

`Outcome` Result of a test case given as "Passed" or "Failed". A "Passed" value indicates to the user that the code performs as expected, "Failed" indicates that it does not.

References

Wickham, H. (2011). 'testthat: Get Started with Testing'. The R Journal (Vol. 3, pp. 5-10). https://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf

Examples

```
## Demonstration mode

testTab <- mts_tests(mode="Demo")
```

`multisepCrisprList` *Example Multisep t-test CRISPR vs XPR results object.*

Description

A data frame containing results from multiSEp t-test based analysis of gene expression and CRISPR.

Usage

```
data("multisepCrisprList")
```

Format

A data frame with 1428 observations (gene pairs) across 25 variables (columns).

`mRNA_gene` Gene id of the gene expression gene

`crispr_gene` Gene id of the crispr gene

`num_modes` Number of gene expression clusters

`nMode1` Number of samples in each gene expression cluster 1

`nMode2` Number of samples in each gene expression cluster 2

`nMode3` Number of samples in each gene expression cluster 3

`nMode4` Number of samples in each gene expression cluster 4

`nMode5` Number of samples in each gene expression cluster 5

`mMode1` Mean CRISPR score of the crispr gene in each gene expression cluster 1

`mMode2` Mean CRISPR score of the crispr gene in each gene expression cluster 2

`mMode3` Mean CRISPR score of the crispr gene in each gene expression cluster 3

`mMode4` Mean CRISPR score of the crispr gene in each gene expression cluster 4

`mMode5` Mean CRISPR score of the crispr gene in each gene expression cluster 5

`shift2_1` Fold-change of CRISPR scores between expression modes 2 and 1

`shift3_2` Fold-change of CRISPR scores between expression modes 3 and 2

`shift4_3` Fold-change of CRISPR scores between expression modes 4 and 3

`shift5_4` Fold-change of CRISPR scores between expression modes 5 and 4

`pvalue2_1` P-value of CRISPR score difference between expression modes 2 and 1

`pvalue3_2` P-value of CRISPR score difference between expression modes 3 and 2

`pvalue4_3` P-value of CRISPR score difference between expression modes 4 and 3

`pvalue5_4` P-value of CRISPR score difference between expression modes 5 and 4

`qvalue2_1` Q-value of CRISPR score difference between expression modes 2 and 1

`qvalue3_2` Q-value of CRISPR score difference between expression modes 3 and 2

`qvalue4_3` Q-value of CRISPR score difference between expression modes 4 and 3

`qvalue5_4` Q-value of CRISPR score difference between expression modes 5 and 4

Source

Derived from analysis of data taken from DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(multisepCrisprList)
multisepCrisprList[1:10,]
```

`multisepList` *Example results from mts_clusterAvg()*

Description

mts_clusterAvg() produces a list of sublists corresponding to each gene in the input exprsMatrix object. Each sublist has the following components:

Usage

```
data("multisepList")
```

Format

A named list containing ten genes, each gene's element is an unnamed list with two components as follows:

Average CRISPR Scores: The average CRISPR score of each gene in the crisprMatrix object in each cluster assigned by multisep to each gene in the exprsMatrix object.

Sample to Cluster map: The log2 gene expression value for each sample and the associated multisep cluster assignment.

Source

Derived from analysis of DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

See Also

`mts_clusterAvg`

Examples

```
data(multisepList)
multisepList[[1]][[2]][1:10,]
```

`MUT_impactMatrix` *Example mutation impact matrix*

Description

A binary/numeric matrix in gene by sample format, where row names correspond to genes (or probesets) and columns are sample names. The matrix cells must have values of '1' or '2' to represent mutation impact (1 for HIGH impact, 2 for LOW impact). This matrix has been adapted from DepMap (Tsherniak et al. 2017), the numeric values in the matrix were generated pseudo-randomly.

Usage

```
data("MUT_impactMatrix")
```

Format

A data frame with 23 observations (genes) on 26 variables (cell lines).

Source

DepMap 2024Q2 doi:10.25452/figshare.plus.25880521.v1

References

Tsherniak A., Vazquez F., Montgomery P.G., Weir B.A., Kryukov G., Cowley G.S., Gill S., Harrington W.F., Pantel S., Krill-Burger J.M. et al. . Defining a cancer dependency map. Cell. 2017; 170:564-576. doi:10.1016/j.cell.2017.06.010

Examples

```
data(MUT_impactMatrix)
MUT_impactMatrix[1:10,1:10]
```

Index