Package ‘gINTomics’

May 29, 2024

Title Multi-Omics data integration

Version 1.0.0

Description gINTomics is an R package for Multi-Omics data integration and visualization. gINTomics is designed to detect the association between the expression of a target and of its regulators, taking into account also their genomics modifications such as Copy Number Variations (CNV) and methylation. What is more, gINTomics allows integration results visualization via a Shiny-based interactive app.

License AGPL-3

biocViews GeneExpression, RNASEq, Microarray, Visualization, CopyNumberVariation, GeneTarget

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports BiocParallel, biomaRt, OmnipathR, edgeR, ggplot2, ggridges, gtools, MultiAssayExperiment, plyr, stringi, stringr, SummarizedExperiment, methods, stats, reshape2, randomForest, limma, org.Hs.eg.db, org.Mm.eg.db, BiocGenerics, GenomicFeatures, ReactomePA, clusterProfiler, dplyr, AnnotationDbi, TxDb.Hsapiens.UCSC.hg38.knownGene, TxDb.Mmusculus.UCSC.mm10.knownGene, shiny, GenomicRanges, ggtree, shinydashboard, plotly, DT, MASS, InteractiveComplexHeatmap, ComplexHeatmap, visNetwork, shiny.gosling, ggvenn, RColorBrewer, utils, grDevices, callr, circlize

Depends R (>= 4.4.0)

LazyData false

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr
BugReports https://github.com/angelovelle96/gINTomics/issues

URL https://github.com/angelovelle96/gINTomics
git_url https://git.bioconductor.org/packages/gINTomics
git_branch RELEASE_3_19
git_last_commit 6525a11
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29

Author Angelo Velle [cre, aut] (<https://orcid.org/0000-0002-4010-6390>),
Francesco Patane' [aut] (<https://orcid.org/0009-0001-8619-447X>),
Chiara Romualdi [aut] (<https://orcid.org/0000-0003-4792-9047>)

Maintainer Angelo Velle <angelo.velle@unipd.it>

Contents

gINTomics-package ......................................................... 3
create_multiassay ........................................................... 3
dot_plotly ................................................................. 4
extract_model_res ......................................................... 5
mirna_hsa ................................................................. 7
mmultiassay_ov ............................................................ 7
MultiClass-class .......................................................... 8
MultiOmics-class ........................................................ 8
plot_chr_distribution .................................................... 9
plot_heatmap .............................................................. 10
plot_network .............................................................. 11
plot_ridge ................................................................. 12
plot_tf_distribution ..................................................... 13
plot_venn ................................................................. 13
plot_volcano .............................................................. 14
run_cnv_integration ..................................................... 15
run_genomic_enrich ..................................................... 16
run_genomic_integration ................................................. 17
run_met_integration ..................................................... 19
run_multiomics .......................................................... 20
run_shiny ................................................................. 22
run_tf_enrich ............................................................. 23
run_tf_integration ....................................................... 24

Index 26
gINTomics-package  

**gINTomics: Multi-Omics data integration**

**Description**

gINTomics is an R package for Multi-Omics data integration and visualization. gINTomics is designed to detect the association between the expression of a target and of its regulators, taking into account also their genomics modifications such as Copy Number Variations (CNV) and methylation. What is more, gINTomics allows integration results visualization via a Shiny-based interactive app.

**Author(s)**

**Maintainer:** Angelo Velle <angelo.velle@unipd.it> (ORCID)

Authors:

- Francesco Patane’ <francesco.patane@unipd.it> (ORCID)
- Chiara Romualdi <chiara.romualdi@unipd.it> (ORCID)

**See Also**

Useful links:

- [https://github.com/angelovelle96/gINTomics](https://github.com/angelovelle96/gINTomics)
- Report bugs at [https://github.com/angelovelle96/gINTomics/issues](https://github.com/angelovelle96/gINTomics/issues)

---

**create_multiassay**  

**MultiAssayExperiment generation**

**Description**

This function will generate a proper MultiAssayExperiment suitable for the `run_multiomics` function.

**Usage**

```r
create_multiassay(
  methylation = NULL,
  cnv_data = NULL,
  gene_exp = NULL,
  miRNA_exp = NULL,
  miRNA_cnv_data = NULL,
  ...
)
```

---
Arguments

methylation  Matrix or SummarizedExperiment for Methylation data
cnv_data    Matrix or SummarizedExperiment for genes’ Copy Number Variation data
gene_exp    Matrix or SummarizedExperiment for Gene expression data
miRNA_exp   Matrix or SummarizedExperiment for miRNA expression data
miRNA_cnv_data Matrix or SummarizedExperiment for miRNA’s Copy Number Variations data
...

Additional arguments to be passed to the function

Value

A MultiAssayExperiment object containing the provided assays.

Examples

# Example usage:
library(MultiAssayExperiment)
data('mmultiassay_ov')
gene_exp_matrix <- as.matrix(assay(mmultiassay_ov[['gene_exp']]))
miRNA_exp_matrix <- as.matrix(assay(mmultiassay_ov[['miRNA_exp']]))
meth_matrix <- as.matrix(assay(mmultiassay_ov[['methylation']]))
gene_cnv_matrix <- as.matrix(assay(mmultiassay_ov[['cnv_data']]))
miRNA_cnv_matrix <- as.matrix(assay(mmultiassay_ov[['miRNA_cnv_data']]))
create_multiassay(methylation=meth_matrix, cnv_data=gene_cnv_matrix,
gene_exp=gene_exp_matrix, miRNA_exp=miRNA_exp_matrix,
miRNA_cnv_data=miRNA_cnv_matrix)

Description

plotting enrichment

Usage

dot_plotly(
  enrich_result,
  title = NULL,
  showCategory = 10,
  width = 800,
  height = 700
)
extract_model_res

Arguments

- `enrich_result`: Enrichment analysis results.
- `title`: Title of the plot.
- `showCategory`: Number of categories to display.
- `width`: Width of the plot.
- `height`: Height of the plot.

Value

A plotly object containing the dot plot.

Examples

```r
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
#multiomics_integration <- run_multiomics(data = mmultiassay_ov)
#gen_enr <- run_genomic_enrich(multiomics_integration,
#   qvalueCutoff = 1,
#   pvalueCutoff = 0.05,
#   pAdjustMethod = "none")
#dot_plotly(gen_enr, title = "Enrichment Analysis", showCategory = 10)
```

extract_model_res

Setting method for extracting results

Description

Setting method for extracting results

Usage

```r
extract_model_res(model_results, ...)
```

## S4 method for signature 'list'
```r
eXtract_model_res(
    model_results,
    outliers = TRUE,
    species = "Hsa",
    filters = c("hgnc_symbol", "ensembl_gene_id", "entrezgene_id"),
    genes_info = NULL,
    ...
)
```
## S4 method for signature 'MultiClass'

```r
extract_model_res(
  model_results,
  outliers = TRUE,
  species = "hsa",
  filters = c("hgnc_symbol", "ensembl_gene_id", "entrezgene_id"),
  genes_info = NULL,
  ...
)
```

## S4 method for signature 'MultiOmics'

```r
extract_model_res(
  model_results,
  outliers = TRUE,
  species = "hsa",
  filters = c("hgnc_symbol", "ensembl_gene_id", "entrezgene_id"),
  genes_info = NULL,
  ...
)
```

### Arguments

- `model_results`  The model results object from which to extract results.
- `outliers` if TRUE (by default), it removes outliers
- `species` species for the analysis
- `filters` Specific filters to apply
- `genes_info` genes info

### Value

A dataframe containing the results of all the integration models provided

### Examples

```r
# example code
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
gene_cnv_matrix <- t(as.matrix(assay(mmultiassay_ov["cnv_data" ge])))
gene_exp_matrix <- t(as.matrix(assay(mmultiassay_ov["gene_exp" ge])))
cnv_integration <- run_cnv_integration(
  expression = gene_exp_matrix,
  cnv_data = gene_cnv_matrix
)
data_table <- extract_model_res(cnv_integration)
head(data_table)
```
**mirna_hsa**

| mirna_hsa | miRNA IDs. Dataset containing lastly definition of miRNAs (Names, Accessions, Sequences, Families and others) from different miRBase versions (From miRBase version 6 to version 22). |

**Description**

miRNA IDs. Dataset containing lastly definition of miRNAs (Names, Accessions, Sequences, Families and others) from different miRBase versions (From miRBase version 6 to version 22).

**Usage**

data(mirna_hsa)
"mirna_hsa"

**Value**

An object of class data.frame.

**Examples**

# example code
data(mirna_hsa)
head(mirna_hsa)

---

**mmultiassay_ov**

Example data for a standard workflow. This is an example dataset containing a MultiAssayExperiment of 20 ovarian cancer (OVC) patients extracted from the Cancer Genome Atlas (TCGA) database. The object contains all the available input data types: Gene expression data, miRNA expression data, gene methylation data, gene Copy Number Variations and miRNA Copy Number Variations.

**Description**

Example data for a standard workflow. This is an example dataset containing a MultiAssayExperiment of 20 ovarian cancer (OVC) patients extracted from the Cancer Genome Atlas (TCGA) database. The object contains all the available input data types: Gene expression data, miRNA expression data, gene methylation data, gene Copy Number Variations and miRNA Copy Number Variations.

**Usage**

data(mmultiassay_ov)
"mmultiomics_ov"
Value

An object of class `MultiAssayExperiment`.

Examples

```r
# example code
data(multiassay_ov)
multiassay_ov
```

MultiClass-class

Description

S4 class containing the output of a single integration, for which classes has been provided. It’s a list in which each element represents the result of the integration for a given class. The length will be equal to the number of classes defined.

Value

MultiOmics Class

MultiOmics-class

Description

S4 class containing the output of a multiomics integration. It’s a list in which each element represents the result of an integration. If all the available omics are provided, it will be a list of integrations: `gene_genomic_res`, `mirna_cnv_res`, `tf_res`, `tf_mirna_res` and `mirna_target_res`

Value

MultiOmics Class
plot_chr_distribution

Description

plotting chr distribution

Usage

plot_chr_distribution(
  data_table,
  class = NULL,
  omics = NULL,
  cnv_met = NULL,
  pval = 0.05
)

Arguments

data_table The data table containing information for plotting chromosome distribution.
class Optional. The class of interactions to include in the plot.
omics Optional. The type of omics data for the plot.
cnv_met Optional. The type of copy number variation or methylation data.
pval Optional. The p-value threshold for significance. Default is 0.05.

Value

A histogram plot showing chromosome distribution.

Examples

# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
# multiomics_integration <- run_multiomics(data = mmultiassay_ov)
# data_table <- extract_model_res(multiomics_integration)
# plot_chr_distribution(data_table, omics = "gene_genomic_res")
plot_heatmap

Description
plotting heatmap

Usage
plot_heatmap(
  multiomics_integration,
  data_table,
  omics,
  scale = "none",
  genes_number = 50,
  samples_number = 50,
  class = NULL,
  pval = 0.05
)

Arguments
multiomics_integration
  The multiomics integration object.
data_table
  The data table containing information for the heatmap.
omics
  The type of omics data for the heatmap.
scale
  Optional. The scale type for the heatmap. Default is "none".
genes_number
  Optional. The number of genes to include in the heatmap. Default is 50.
samples_number
  Number of samples to include in the heatmap. If this number is inferior to the total number of samples, the n most variable samples will be selected.
class
  Optional. The class of interactions to include in the heatmap.
pval
  Optional. The p-value threshold for significance in the heatmap. Default is 0.05.

Value
A heatmap plot.

Examples
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
# multiomics_integration <- run_multiomics(data = mmultiassay_ov)
# data_table <- extract_model_res(multiomics_integration)
# data_table <- data_table[!is.na(data_table$cnv_met),]
# plot_heatmap(multiomics_integration, data_table, omics = "gene_genomic_res")

---

**Description**

Plotting network

**Usage**

```r
plot_network(data_table, num_interactions = 300, class = NULL, pval = 0.05)
```

**Arguments**

- `data_table`: The data table containing network information.
- `num_interactions`: The number of interactions to display in the network (default: 300).
- `class`: Optional. The class of interactions to include in the plot.
- `pval`: The p-value threshold for selecting interactions (default: 0.05).

**Value**

A network plot.

**Examples**

```r
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
# multiomics_integration <- run_multiomics(data = mmultiassay_ov)
# data_table <- extract_model_res(multiomics_integration)
# plot_network(data_table)
```
**plot_ridge**

Description

plotting ridge

Usage

plot_ridge(data_table, class = NULL, omics = NULL, cnv_met = NULL)

Arguments

data_table The data table containing information for the ridge plot.
class Optional. The class of interactions to include in the ridge plot.
omics Optional. The omics type for the ridge plot.
cnv_met Optional. Indicates whether the ridge plot is for CNV or MET omics (only applicable if omics is specified).

Value

A ridge plot.

Examples

# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
gene_cnv_matrix <- t(as.matrix(assay(mmultiassay_ov["cnv_data"])))
gene_exp_matrix <- t(as.matrix(assay(mmultiassay_ov["gene_exp"])))

# Run CNV integration

cnv_integration <- run_cnv_integration(
  expression = gene_exp_matrix,
  cnv_data = gene_cnv_matrix)

data_table <- extract_model_res(cnv_integration)

data_table <- data_table[data_table$cov!="(Intercept)",]

plot_ridge(data_table)
plot_tf_distribution

Description
plotting TF distribution

Usage
plot_tf_distribution(data_table, class = NULL, pval = 0.05)

Arguments
- data_table: The data table containing TF information.
- class: Optional. The class of interactions to include in the distribution plot.
- pval: Optional. The p-value threshold for significance in the distribution plot. Default is 0.05.

Value
A TF distribution plot.

Examples
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
# multiomics_integration <- run_multiomics(data = mmultiassay_ov)
# data_table <- extract_model_res(multiomics_integration)
# plot_tf_distribution(data_table, pval=0.5)

plot_venn

Description
plotting venn

Usage
plot_venn(data_table, class = NULL)
Arguments

- `data_table`: The data table containing information for the Venn diagram.
- `class`: Optional. The class of interactions to include in the Venn diagram.

Value

A Venn diagram plot.

Examples

```r
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
# multiomics_integration <- run_multiomics(data = mmultiassay_ov)
# data_table <- extract_model_res(multiomics_integration)
# plot_venn(data_table)
```

Description

plotting volcano

Usage

```r
plot_volcano(data_table, class = NULL, omics = NULL, cnv_met = NULL)
```

Arguments

- `data_table`: The data table containing information for the volcano plot.
- `class`: Optional. The class of interactions to include in the volcano plot.
- `omics`: Optional. The omics type for the volcano plot.
- `cnv_met`: Optional. Indicates whether the volcano plot is for CNV or MET omics (only applicable if omics is specified).

Value

A volcano plot.
**Examples**

```r
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
multiomics_integration <- run_multiomics(data = mmultiassay_ov)
data_table <- extract_model_res(multiomics_integration)
plot_volcano(data_table, omics = "gene_genomic_res", cnv_met = "cnv")
```

---

**run_cnv_integration**  
*Integration of expression and Copy Number Variations*

**Description**

This function will perform an integration of expression data and Copy Number Variations data

**Usage**

```r
run_cnv_integration(
  expression,
  cnv_data,
  sequencing_data = TRUE,
  normalize = TRUE,
  norm_method = "TMM",
  class = NULL,
  run_deg = TRUE,
  BPPARAM = SerialParam(),
  ...
)
```

**Arguments**

- **expression**  
  Matrix or data.frame containing the expression values for each model. Rows represent samples, while each column represents the different response variables of the models.

- **cnv_data**  
  Matrix or data.frame containing the Copy Number variation status for the models. Rows represent samples, while columns represent the different covariates. If **interactions** are not provided, they will be automatically generated and for each gene contained in **expression** the model will look for the same gene in **cnv_data**

- **sequencing_data**  
  logical. Are expression data obtained from RNA sequencing? Default is set to TRUE

- **normalize**  
  logical. Should expression data be normalized? Default is set to TRUE

- **norm_method**  
  Normalization method to be used for expression data. One of "TMM" (default), "TMMwsp", "RLE", "upperquartile", "none".
run_genomic_enrich

Description

Running genomic enrichment analysis

Usage

run_genomic_enrich(
  model_results,
  species = "hsa",
  pvalueCutoff = 0.1,
  pAdjustMethod = "BH",
  qvalueCutoff = 0.1,
  ont = "all",
  BPPARAM = BiocParallel::SerialParam(),
  extracted_data = NULL,
  ...
)

Value

A list or a MultiClass object if class is provided containing the results of the CNV integration

Examples

# Example usage_multi:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
gene_cnv_matrix <- t(as.matrix(assay(multiassay_ov["cnv_data"])))
gene_exp_matrix <- t(as.matrix(assay(multiassay_ov["gene_exp"])))
cnv_integration <- run_cnv_integration(
  expression = gene_exp_matrix,
  cnv_data = gene_cnv_matrix
)
Arguments

model_results  Model integration results, typically a list containing different types of genomic results
species      Species to select for the enrichment analysis. Default is 'hsa' (Homo sapiens).
pvalueCutoff P-value cutoff for significant enrichment. Default is 0.1.
pAdjustMethod Method for adjusting p-values. Default is 'BH' (Benjamini & Hochberg).
qvalueCutoff Q-value cutoff for significant enrichment. Default is 0.1.
ont          Ontology to use for the enrichment analysis. Default is 'all'.
BPPARAM     A BiocParallelParam object specifying parallelization options. Default is BiocParallel::SerialParam().
extracted_data Pre-extracted data for enrichment analysis. If NULL, function will extract relevant data from model_results.
...          Additional arguments to be passed to the internal enrichment function.

Value

A list containing enrichment results. If CNV and methylation data are available, it returns a nested list with results for each data type.

Examples

# Example usage:
library(MultiAssayExperiment)
data(multiassay_ov)
tmp <- lapply(experiments(multiassay_ov), function(x) x[1:200,])
multiassay_ov <- MultiAssayExperiment(experiments = tmp)
#multiomics_integration <- run_multiomics(multiassay_ov)
#gen_enr <- run_genomic_enrich(multiomics_integration, qvalueCutoff = 1,
#pvalueCutoff = 0.05, pAdjustMethod = 'none')

Description

This function will perform an integration of expression data and Copy Number Variations data.
Usage

```r
run_genomic_integration(
  expression,
  cnv_data,
  methylation,
  sequencing_data = TRUE,
  normalize = TRUE,
  norm_method = "TMM",
  interactions = NULL,
  class = NULL,
  scale = TRUE,
  run_deg = TRUE,
  BPPARAM = SerialParam(),
  ...)
```

Arguments

description

- **expression**: Matrix or data.frame containing the expression values for each model. Rows represent samples, while each column represents the different response variables of the models.

- **cnv_data**: Matrix or data.frame containing the Copy Number variation status for the models. Rows represent samples, while columns represent the different covariates. If **interactions** are not provided, they will be automatically generated and for each gene contained in **expression** the model will look for the same gene in **cnv_data**

- **methylation**: Matrix or data.frame containing the methylation values for the models. Rows represent samples, while columns represent the different covariates. If **interactions** are not provided, they will be automatically generated and for each gene contained in **expression** the model will look for the same gene in **methylation**

- **sequencing_data**: logical. Are expression data obtained from RNA sequencing? Default is set to **TRUE**

- **normalize**: logical. Should expression data be normalized? Default is set to **TRUE**

- **norm_method**: Normalization method to be used for expression data. One of "TMM" (default), "TMMwp", "RLE", "upperquartile", "none".

- **interactions**: A list of character vectors containing the interactions between response variable and covariates. The names of the list should match the response variables while the character contained in each element of the list should match the covariates. If NULL (default), the interactions will be automatically defined according to response variable’s colnames.

- **class**: Character vector specifying the classes for differential expression analysis.

- **scale**: Logical. Should the data be scaled? Default is set to **TRUE**.

- **run_deg**: Logical. Should differential expression analysis be performed? Default is set to **TRUE**.
run_met_integration

**BPPARAM**
A BiocParallelParam object specifying the parallel backend to be used.

... Additional arguments to be passed to internal functions.

**Value**
A list or a MultiClass object if `class` is provided containing the results of the Genomic integration.

**Examples**

```r
# Example usage_multi:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
meth_matrix <- t(as.matrix(assay(mmultiassay_ov["methylation"])))
gene_exp_matrix <- t(as.matrix(assay(mmultiassay_ov["gene_exp"])))
gene_cnv_matrix <- t(as.matrix(assay(mmultiassay_ov["cnv_data"])))
genomic_integration <- run_genomic_integration(
  expression = gene_exp_matrix,
  cnv_data = gene_cnv_matrix, methylation = meth_matrix
)
```

---

**run_met_integration** Integration of expression and methylation

**Description**
This function will perform an integration of expression data and methylation data.

**Usage**

```r
run_met_integration(
  expression,
  methylation,
  sequencing_data = TRUE,
  normalize = TRUE,
  norm_method = "TMM",
  class = NULL,
  run_deg = TRUE,
  BPPARAM = SerialParam(),
  ...
)
```

**Arguments**

- `expression` Matrix or data.frame containing the expression values for each model. Rows represent samples, while each column represents the different response variables of the models.
methylation Matrix or data.frame containing the methylation values for the models. Rows represent samples, while columns represent the different covariates. If interactions are not provided, they will be automatically generated and for each gene contained in expression the model will look for the same gene in methylation

sequencing_data logical. Are expression data obtained from RNA sequencing? Default is set to TRUE

normalize logical. Should expression data be normalized? Default is set to TRUE

norm_method Normalization method to be used for expression data. One of "TMM" (default), "TMMwsp", "RLE", "upperquartile", "none".

class Character vector specifying the classes for differential expression analysis.

run_deg Logical. Should differential expression analysis be performed? Default is set to TRUE.

BPPARAM A BiocParallelParam object specifying the parallel backend to be used.

... Additional arguments to be passed to internal functions.

Value

A list or a MultiClass object if class is provided containing the results of the Methylation integration

Examples

# Example usage_multi:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
meth_matrix <- t(as.matrix(assay(mmultiassay_ov["methylation"])))
gene_exp_matrix <- t(as.matrix(assay(mmultiassay_ov["gene_exp"])))
met_integration <- run_met_integration(
  expression = gene_exp_matrix,
  methylation = meth_matrix
)

run_multiomics Complete Multi-Omics integration

Description

This function will perform a complete Multi-Omics integration on a MultiAssayExperiment
run_multiomics

Usage

run_multiomics(
  data,
  interactions_met = NULL,
  interactions_miRNA_target = NULL,
  interactions_tf = NULL,
  interactions_tf_miRNA = NULL,
  RNAseq = TRUE,
  miRNAseq = TRUE,
  normalize_miRNA_expr = TRUE,
  normalize_gene_expr = TRUE,
  norm_method_gene_expr = "TMM",
  norm_method_miRNA_expr = "TMM",
  class = NULL,
  BPPARAM = SerialParam()
)

Arguments

data A MultiAssayExperiment. It can be generated exploiting the generate_multiassay function.
interactions_met interactions as for run_met_integration
interactions_miRNA_target miRNA-target interactions as requested by run_tf_integration
interactions_tf TF-target interactions as requested by run_tf_integration
interactions_tf_miRNA TF-target interactions as requested by run_tf_integration
RNAseq logical. Are gene expression data obtained from RNA sequencing? Default is set to TRUE
miRNAseq logical. Are miRNA expression data obtained from miRNA sequencing? Default is set to TRUE
normalize_miRNA_expr logical. Should miRNA expression data be normalized? Default is set to TRUE
normalize_gene_expr logical. Should gene expression data be normalized? Default is set to TRUE
norm_method_gene_expr Normalization method to be used for gene expression data. One of "TMM" (default), "TMMwsp", "RLE", "upperquartile", "none".
norm_method_miRNA_expr Normalization method to be used for miRNA expression data. One of "TMM" (default), "TMMwsp", "RLE", "upperquartile", "none".
class Character vector specifying the classes for differential expression analysis.
BPPARAM A BiocParallelParam object specifying the parallel backend to be used.
Value

A MultiOmic object containing the results of all the possible integration models

Examples

```r
# Example usage_multiomics:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
multiomics_integration <- run_multiomics(data = mmultiassay_ov)
```

run_shiny

Start a Shiny application for integrated multi-omics data analysis.

Description

The run_shiny function launches an interactive Shiny application that allows users to explore and analyze integrated multi-omics data through various visualizations and analyses.

Usage

```r
run_shiny(multiomics_integration)
```

Arguments

- `multiomics_integration`:
  
  An object representing the integration of multi-omics data, compatible with the `extract_model_res` function.

Details

The run_shiny function extracts model results from multiomics_integration, performs preprocessing operations to prepare the data for the Shiny user interface, creates the user interface and server for the Shiny application.

Value

No return value. The function starts an interactive Shiny application.

References

Description of the multi-omics data model and integrated analysis techniques used.

See Also

- `extract_model_res`
Examples

```r
# Example usage:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
# multiomics_integration <- run_multiomics(data = mmultiassay_ov)
# app <- run_shiny(multiomics_integration)
```

run_tf_enrich

Running TF enrichment analysis

Description

Running TF enrichment analysis

Usage

```r
run_tf_enrich(
  model_results,
  species = "hsa",
  pvalueCutoff = 0.1,
  qvalueCutoff = 0.1,
  pAdjustMethod = "BH",
  ont = "all",
  BPPARAM = BiocParallel::SerialParam(),
  extracted_data = NULL,
  ...
)
```

Arguments

- **model_results**: Model integration results, typically a list containing TF data.
- **species**: Species to select for the enrichment analysis. Default is 'hsa' (Homo sapiens).
- **pvalueCutoff**: P-value cutoff for significant enrichment. Default is 0.1.
- **qvalueCutoff**: Q-value cutoff for significant enrichment. Default is 0.1.
- **pAdjustMethod**: Method for adjusting p-values. Default is 'BH' (Benjamini & Hochberg).
- **ont**: Ontology to use for the enrichment analysis. Default is 'all'.
- **BPPARAM**: A BiocParallelParam object specifying parallelization options. Default is BiocParallel::SerialParam().
- **extracted_data**: Pre-extracted data for enrichment analysis. If NULL, function will extract relevant data from model_results.
- **...**: Additional arguments to be passed to the internal enrichment function.
run_tf_integration

Value

A list containing TF enrichment results.

Examples

```r
# Example usage:
library(MultiAssayExperiment)
data(multiassay_ov)
tmp <- lapply(experiments(multiassay_ov), function(x) x[1:200,])
multiassay_ov <- MultiAssayExperiment(experiments = tmp)
#multiomics_integration <- run_multiomics(multiassay_ov)
#run_tf_enrich(multiomics_integration, qvalueCutoff = 1, pvalueCutoff = 0.05,
#pAdjustMethod = 'none')
```

---

run_tf_integration Integration of expression and Transcription Factors / Generic Regulators

Description

This function will perform an integration of gene/miRNA expression data and Transcription Factors expression. Moreover, every type of regulator can be provided to the function as covariate through the `tf_expression` argument. Interactions for TF-target, miRNA-target and TF-miRNA integration will be automatically downloaded by the function as defined by the `type` argument. Other types of interactions should be provided through the `interactions` argument.

Usage

```r
run_tf_integration(  
expression,
  tf_expression = expression,
  interactions = NULL,
  type = "none",
  sequencing_data = TRUE,
  species = "hsa",
  normalize = TRUE,
  norm_method = "TMM",
  normalize_cov = TRUE,
  norm_method_cov = "TMM",
  class = NULL,
  run_deg = TRUE,
  BPPARAM = SerialParam(),
  ...
)
```
Arguments

expression       Matrix or data.frame containing the expression values for each model. Rows represent samples, while each column represents the different response variables of the models.

tf_expression    Matrix or data.frame containing the expression values for the models. Rows represent samples, while columns represent the different covariates. If not provided, it will be set equal to expression.

interactions     A list of character vectors containing the interactions between response variable and covariates. The names of the list should match the response variables while the character contained in each element of the list should match the covariates. If NULL (default), the interactions will be automatically downloaded according to the type argument.

type             A character defining the type of regulation under analysis. Should be one of "tf_miRNA", "tf", "miRNA_target".

sequencing_data  logical. Are expression data obtained from RNA sequencing? Default is set to TRUE

species          species information for interactions download. Fully supported species are "hsa" (default) and "mmu".

normalize        logical. Should expression data be normalized? Default is set to TRUE

norm_method      Normalization method to be used for expression data. One of "TMM" (default), "TMMwsp", "RLE", "upperquartile", "none".

normalize_cov    Same as normalize but for covariates.

norm_method_cov  Same as norm_method but for covariates.

class            Character vector specifying the classes for differential expression analysis.

run_deg          Logical. Should differential expression analysis be performed? Default is set to TRUE.

BPPARAM          A BiocParallelParam object specifying the parallel backend to be used.

...              Additional arguments to be passed to internal functions.

Value

A list or a MultiClass object if class is provided containing the results of the transcriptional integration.

Examples

# Example usage_multi:
library(MultiAssayExperiment)
data("mmultiassay_ov")
tmp <- lapply(experiments(mmultiassay_ov), function(x) x[1:20,])
mmultiassay_ov <- MultiAssayExperiment(experiments = tmp)
gene_exp_matrix <- t(as.matrix(assay(mmultiassay_ov["gene_exp"])))
tf_integration <- run_tf_integration(expression = gene_exp_matrix, type="tf")
Index

* Data analysis
  run_shiny, 22

* Function
  run_shiny, 22

* Integration
  run_shiny, 22

* Interactive
  run_shiny, 22

* Multi-omics
  run_shiny, 22

* Shiny
  run_shiny, 22

* Visualization
  run_shiny, 22

* analysis
  run_shiny, 22

* integration
  run_shiny, 22

* internal
  gINTomics-package, 3

* multiomics
  run_shiny, 22

* shiny
  run_shiny, 22

* visualization
  run_shiny, 22

create_multiassay, 3
data.frame, 7
dot_plotly, 4

extract_model_res, 5, 22
extract_model_res, list-method
(extract_model_res), 5
extract_model_res, MultiClass-method
(extract_model_res), 5
extract_model_res, MultiOmics-method
(extract_model_res), 5
gINTomics (gINTomics-package), 3

mirna_hsa, 7
mmultiassay_ov, 7
MultiAssayExperiment, 8
MultiClass, 16, 19, 20, 25
MultiClass-class, 8
MultiOmics, 22
MultiOmics-class, 8

plot_chr_distribution, 9
plot_heatmap, 10
plot_network, 11
plot_ridge, 12
plot_tf_distribution, 13
plot_venn, 13
plot_volcano, 14

run_cnv_integration, 15
run_genomic_enrich, 16
run_genomic_integration, 17
run_met_integration, 19
run_multiomics, 20
run_shiny, 22
run_tf_enrich, 23
run_tf_integration, 24

26