Package ‘phantasus’

March 16, 2024

Title  Visual and interactive gene expression analysis
Version  1.22.2
Description  Phantasus is a web-application for visual and interactive gene expression analysis. Phantasus is based on Morpheus – a web-based software for heatmap visualisation and analysis, which was integrated with an R environment via OpenCPU API. Aside from basic visualization and filtering methods, R-based methods such as k-means clustering, principal component analysis or differential expression analysis with limma package are supported.

URL  https://genome.ifmo.ru/phantasus,
https://artyomovlab.wustl.edu/phantasus

BugReports  https://github.com/ctlab/phantasus/issues
Depends  R (>= 3.5)
biocViews  GeneExpression, GUI, Visualization, DataRepresentation, Transcriptomics, RNASeq, Microarray, Normalization, Clustering, DifferentialExpression, PrincipalComponent, ImmunoOncology
Imports  ggplot2, protolite, Biobase, GEOquery, Rook, htmltools, httpuv, jsonlite, limma, edgeR, opencpu, assertthat, methods, httr, rhdf5, utils, parallel, stringr, fgsea (>= 1.9.4), svglite, tibble, stats, Matrix, pheatmap, scales, ccaPP, grid, grDevices, AnnotationDbi, DESeq2, data.table, curl
License  MIT + file LICENSE
Encoding  UTF-8
LazyData  true
RoxygenNote  7.2.3
Suggests  testthat, BiocStyle, knitr, rmarkdown
VignetteBuilder  knitr
NeedsCompilation  no

git_url  https://git.bioconductor.org/packages/phantasus
git_branch  RELEASE_3_18
git_last_commit  64188e7
topics documented: git_last_commit_date 2023-11-15

Repository Bioconductor 3.18

Date/Publication 2024-03-15

Author Daria Zenkova [aut],
Vladislav Kamenev [aut],
Rita Sabлина [ctb],
Maxim Kleverov [ctb],
Maxim Artyomov [aut],
Alexey Sergushichev [aut, cre]

Maintainer Alexey Sergushichev <alsergbox@gmail.com>

R topics documented:

adjustDataset .................................................. 3
annotationDBMeta .............................................. 4
calcPCA .......................................................... 4
calculatedAnnotation .......................................... 5
checkGPLsFallback ............................................. 6
collapseDataset ................................................ 6
colMeansByGroups ............................................. 7
convertByAnnotationDB ........................................ 8
createES ........................................................... 8
es ................................................................. 9
fgseaExample ................................................... 10
generatePreloadedSession .................................. 10
getArchs4Files ................................................ 11
getCountsMetaPart ............................................. 11
getES ............................................................. 12
getGDS ............................................................ 13
getGSE ............................................................ 14
gseaPlot .......................................................... 14
limmaAnalysis ................................................... 15
loadCounts ....................................................... 16
loadFromARCHS4 ................................................. 17
loadGEO ........................................................... 17
loadPreloaded .................................................... 18
performKmeans .................................................. 19
queryAnnotationDBMeta ...................................... 19
read.gct .......................................................... 20
reparseCachedESs .............................................. 20
reproduceInR .................................................... 21
servePhantasus .................................................. 22
shinyGAMAnalysis ............................................. 23
subsetES ........................................................... 23
updateARCHS4 ................................................... 24
updateARCHS4meta ............................................. 24
Description

Adjust dataset

Usage

adjustDataset(
  es,
  scaleColumnSum = NULL,
  log2 = FALSE,
  onePlusLog2 = FALSE,
  inverseLog2 = FALSE,
  quantileNormalize = FALSE,
  zScore = FALSE,
  robustZScore = FALSE,
  sweep = NULL
)

Arguments

es Expression set to perform adjustment on
scaleColumnSum perform sum scaling of columns (default FALSE)
log2 perform logarithm2 adjustment (default FALSE)
onePlusLog2 perform log2(1+x) adjustment (default FALSE)
inverseLog2 perform 2^x adjustment (default FALSE)
quantileNormalize perform quantile normalization (default FALSE)
zScore perform zScore adjustment: subtract mean, divide by std (default FALSE)
robustZScore perform robustZScore adjustment: subtract median, divide by MAD (default FALSE)
sweep perform sweep adjustment on rows/columns (default FALSE)

Value

Nothing. Adjusted dataset will be assigned as ES in global environment
Examples

```r
## Not run:
es <- gseGSE('GSE53986')[[1]]
adjustDataset(es, log2 = T, quantileNormalize = T)

## End(Not run)
```

---

**annotationDBMeta**

Create meta file for AnnotationDB

**Description**

createES function creates an rds file containing meta information of provided sqlite files for AnnotationDB

**Usage**

`annotationDBMeta(cacheDir)`

**Arguments**

- `cacheDir` cacheDir for phantasus

**Value**

nothing

**Examples**

```r
## Not run:
annotationDBMeta('/var/phantasus/cache')

## End(Not run)
```

---

**calcPCA**

Principal Component Analysis.

**Description**

calcPCA calculates PCA-matrix for the given ExpressionSet and returns this matrix encoded to JSON.

**Usage**

`calcPCA(es, replacena = "mean")`
calculatedAnnotation

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>es</td>
<td>ExpressionSet object, should be normalized</td>
</tr>
<tr>
<td>replacena</td>
<td>method for replacing NA values (mean by default)</td>
</tr>
</tbody>
</table>

Value

json with full description of the plot for plotly.js

Examples

```r
## Not run:
data(es)calcPCA(es)
## End(Not run)
```

Description

calculatedAnnotation adds a column calculated by operation

Usage

```r
calculatedAnnotation(
es, operation, rows = c(), columns = c(), isColumns = FALSE, name = NULL)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>es</td>
<td>ExpressionSet object.</td>
</tr>
<tr>
<td>operation</td>
<td>Name of the operation to perform calculation</td>
</tr>
<tr>
<td>rows</td>
<td>List of specified rows' indices (optional), indices start from 0</td>
</tr>
<tr>
<td>columns</td>
<td>List of specified columns' indices (optional), indices start from 0</td>
</tr>
<tr>
<td>isColumns</td>
<td>Apply fn to columns</td>
</tr>
<tr>
<td>name</td>
<td>Name of the new annotation</td>
</tr>
</tbody>
</table>

Value

Nothing. Annotated dataset will be assigned to es in environment
checkGPLsFallback  

Check possible annotations for GEO Dataset.

Description
checkGPLs returns GPL-names for the specified GEO identifier.

Usage
checkGPLsFallback(name)

Arguments
name  
String, containing GEO identifier of the dataset.

Value
Vector of filenames serialized in JSON format. If there is only one GPL for that dataset, the function will return name.

Examples
## Not run:
checkGPLs('GSE27112')
checkGPLs('GSE14308')
## End(Not run)

collapseDataset  

Collapse dataset

Description
collapseDataset performs a collapse action on expression set

Usage
collapseDataset(
es,
isRows = TRUE,
selectOne = FALSE,
fn,
fields,
removeEmpty = TRUE
)

colMeansByGroups

Arguments

- `es`: Expression set
- `isRows`: Work with rows. False if columns (default True - row mode)
- `selectOne`: select best match or merge duplicates
- `fn`: select/merge function
- `fields`: fields to unique on
- `removeEmpty`: remove unannotated genes

Value

Nothing. Collapsed dataset will be assigned to es in environment

Examples

```r
## Not run:
es <- getGSE('GSE53986')[[1]]
collapseDataset(es, isRows = TRUE, selectOne = TRUE,
fn = mean, fields = c('Gene ID', 'Gene symbol'))
## End(Not run)
```

---

colMeansByGroups  

*Calculate column averages in row groups*

Description

Calculate column averages in row groups

Usage

```r
colMeansByGroups(m, groups)
```

Arguments

- `m`: matrix n x m
- `groups`: vector of size n of numbers from 1 to k

Value

- matrix k*m of column averages by groups
convertByAnnotationDB  \textit{Map indexes using Annotation DB}

**Description**

createES function creates an rds file containing meta information of provided sqlite files for AnnotationDB.

**Usage**

```r
convertByAnnotationDB(
es, 
dbName, 
columnName, 
columnType, 
keyType, 
otherOptions
)
```

**Arguments**

- `es`: source ExpressionSet
- `dbName`: name of AnnotationDB file
- `columnName`: name of column in featureData of source ExpressionSet
- `columnType`: Type of indexes in columnName
- `keyType`: Type of mapped indexes
- `otherOptions`: additional parameters for conversion. Currently only named boolean delete-DotVersion is not ignored.

**Value**

JSON object with a vector of converted IDs

---

createES  \textit{Create ExpressionSet.}

**Description**

createES function produces an ExpressionSet object from given data, and exports it to global scope.

**Usage**

```r
createES(data, pData, varLabels, fData, fvarLabels, eData)
```
Arguments

data  Gene expression matrix.
pData  Matrix with phenotypical data.
varLabels  Names of phenoData columns.
fData  Matrix with feature data.
fvarLabels  Names of featureData columns.
eData  List with experimentData

Value

produced ExpressionSet object

Examples

## Not run:
data <- matrix(1:15, 5, 3)
pData <- c("A", "B", "C")
varLabels <- "cat"
fData <- c("p", "r", "s", "t", "u")
fvarLabels <- "id"
eData <- list(name="", lab="", contact="", title="", url="", other=list(), pubMedIds="")
createES(data, pData, varLabels, fData, fvarLabels, eData)
## End(Not run)

---

es  Example dataset

Description

Small slice from GSE27112-GPL6103 for runnable examples.

Usage

data(es)

Format

An object of class ExpressionSet with 20 rows and 5 columns.

Examples

## Not run:
data(es)
performKmeans(es, k = 2)
## End(Not run)
fgseaExample  

Example pathway data.frame for fgsea tool

description

Example pathway data.frame for fgsea tool

generatePreloadedSession

Generate files for preloaded session from a session link.

description

Generate files for preloaded session from a session link.

Usage

generatePreloadedSession(sessionURL, preloadedName, preloadedDir)

Arguments

sessionURL  
String with session link produced by phantasus.

preloadedName  
String with name that should be assigned to the session.

preloadedDir  
Path to the directory with preloaded datasets and sessions.

Value

Function produces two files (preloadedName.rda with ExpressionSet and preloadedName.json with session features) in preloadedDir folder.

Examples

```r
## Not run:
sessionURL <- "https://ctlab.itmo.ru/phantasus/?session=x063c1b365b9211"  # link from 'Get dataset link...' tool in phantasus
newName <- "my_session"  # user defined name
preloadedDir <- "/preloaded"  # directory where files will be stored. In order too get access through phantasus web-app

dir.create(preloadedDir, showWarnings = FALSE)
generatePreloadedSession(sessionURL= sessionURL,
preloadedName = newName,
preloadedDir = preloadedDir)

servePhantasus(preloadedDir=preloadedDir, openInBrowser=FALSE)
# open browser manually at http://0.0.0.0:8000/phantasus/index.html?preloaded=my_session

## End(Not run)
```
getArchs4Files  

Returns list of ARCHS4 hdf5 files with expression data

Description

Returns list of ARCHS4 hdf5 files with expression data

Usage

getArchs4Files(cacheDir)

Arguments

cacheDir  base directory for cache

Value

list of .h5 files

getCountsMetaPart  

Create meta-data for single counts collection

Description

Creates a part of counts collections meta-data

Usage

getCountsMetaPart(counts_dir, collection_name, verbose)

Arguments

counts_dir  path to directory with count collections

collection_name  name of collection and collection’s directory

verbose  logical value which determines a content of the output.

Details

Function assumes that collection_name contains meta.txt which is valid (in sence of validateCountsCollection). For each row in meta.txt function reads specified sample_id dataset and writes every sample id to the resulting data.table with source file name and collection name.

Value

data.table with meta-data or nothing if destdir does not exist or does not contain files.
getES

Load ExpressionSet by GEO identifier

Description

getES return the ExpressionSet object(s) corresponding to GEO identifier.

Usage

getES(
  name,
  type = NA,
  destdir = tempdir(),
)

Arguments

name String, containing GEO identifier of the dataset. It should start with 'GSE' or 'GDS' and can include exact GPL to annotate dataset, separated with dash ('-') from the identifier.

type Type of the dataset: 'GSE' or 'GDS'. If not specified, the function will take first three letters of name variable as type.

destdir Directory for caching loaded Series and GPL files from GEO database.

mirrorPath URL string which specifies the source of matrices.

Value

List of ExpressionSet objects, that were available by given in name variable GEO identifier.
getGDS

Load ExpressionSet from GEO Datasets

Description

getGDS return the ExpressionSet object corresponding to GEO Dataset identifier.

Usage


Arguments

name String, containing GEO identifier of the dataset. It should start with 'GSE' or 'GDS' and can include exact GPL to annotate dataset, separated with dash ('-') from the identifier.
destdir Directory for caching loaded Series and GPL files from GEO database.
mirrorPath URL string which specifies the source of matrices.

Value

ExpressionSet object wrapped in list, that was available by given in name variable GEO identifier.

Examples

getGDS('GDS4922')
getGSE  

*Load ExpressionSet from GEO Series*

**Description**

getGSE return the ExpressionSet object(s) corresponding to GEO Series Identifier.

**Usage**

```r
```

**Arguments**

- `name`  
  String, containing GEO identifier of the dataset. It should start with 'GSE' or 'GDS' and can include exact GPL to annotate dataset, separated with dash ('-') from the identifier.

- `destdir`  
  Directory for caching loaded Series and GPL files from GEO database.

- `mirrorPath`  
  URL string which specifies the source of matrices.

**Value**

List of ExpressionSet objects, that were available by given in name variable GEO identifier.

**Examples**

```r
## Not run:
getGSE("GSE14308", destdir = "cache")
getGSE("GSE27112")

## End(Not run)
getGSE("GSE53986")
```

gseaPlot  

*Returns path to an svg file with enrichment plot*

**Description**

Returns path to an svg file with enrichment plot.
Usage

gseaPlot(
  es,
  rankBy,
  selectedGenes,
  width,
  height,
  vertical = FALSE,
  addHeatmap = FALSE,
  showAnnotation = NULL,
  annotationColors = NULL,
  pallete = c("blue", "white", "red")
)

Arguments

es  ExpressionSet object.
rankBy  name of the numeric column used for gene ranking
selectedGenes  indexes of selected genes (starting from one, in the order of fData)
width  width of the image (in inches)
height  height of the image (in inches)
vertical  whether to use vertical orientation (default: FALSE)
addHeatmap  whether to add an expression heatmap, sorted by rankBy (default: FALSE)
showAnnotation  a name of column annotation to add to the heatmap, default: NULL (no annotation)
annotationColors  a list of colors to use in annotation
pallete  a vector of colors to draw heatmap

Value

  path to an svg file

limmaAnalysis  Differential Expression analysis.

Description

limmaAnalysis performs differential expression analysis from limma package and returns a ProtoBuf-serialized resulting de-matrix.
Usage

```r
limmaAnalysis(
  es,
  fieldValues,
  version = "One-factor design",
  contrast = list("Comparison", "Target", "Reference"),
  designData = NULL
)
```

Arguments

- `es`: ExpressionSet object. It should be normalized for more accurate analysis.
- `fieldValues`: Vector of comparison values, mapping categories' names to columns/samples
- `version`: name of the limma analysis implementation. Should be "One-factor design" or "Advanced design"
- `contrast`: a character vector with exactly three elements: the name of a factor in the design formula, the name of the numerator level for the fold change, and the name of the denominator level for the fold change
- `designData`: data.frame with design matrix

Value

Name of the file containing serialized de-matrix.

Examples

```r
## Not run:
data(es)
limmaAnalysis(es, fieldValues = c("A", "A", "A", "B", "B"))
## End(Not run)
```

---

### loadCounts

**Description**

Loads expression data from .h5 count files. Only samples with counted expression are kept. If `es` already contains expression data it is returned as is.

**Usage**

```r
loadCounts(es, counts_dir)
```
loadFromARCHS4

Arguments

es ExpressionSet from GEO to check for expression in ARCHS4/dec2 or other h5 files
counts_dir directory with .h5 files collections. There must be meta.rda file in counts_dir and each collection’s sub directory must have meta.txt file with description. Also counts_dir must contain counts_priority.txt file.

Value

either original es or an ExpressionSet with loaded count data from ARCHS4

loadFromARCHS4

Load expression data from ARCHS4 count files. Only samples with counted expression are kept. If es already contains expression data it is returned as is.

Usage

loadFromARCHS4(es, archs4_files)

Arguments

es ExpressionSet from GEO to check for expression in ARCHS4
archs4_files list of available .h5 files from ARCHS4 project

Value

either original es or an ExpressionSet with loaded count data from ARCHS4

loadGEO

Load GEO Dataset.

Description

loadGEO returns the file with serialized ExpressionSet using ProtoBuf, parsed from data downloaded from GEO by identifier.

Usage

loadGEO(name, type = NA)
loadPreloaded

Description

loadPreloaded returns the file with serialized ExpressionSets using ProtoBuf, that were preloaded on server.

Usage

loadPreloaded(name)

Arguments

name String, containing filename. Assuming that in the directory with preloaded files preloadedDir exists file filename.rda with list of ExpressionSets ess.

Value

File with ProtoBuf-serialized ExpressionSet-s that were loaded from specified file.
performKmeans

K-means clusterisation.

Description

performKmeans returns a vector of corresponding clusters for each gene from a given Expression-Set.

Usage

performKmeans(es, k, replacena = "mean")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>es</td>
<td>ExpressionSet object.</td>
</tr>
<tr>
<td>k</td>
<td>Expected number of clusters.</td>
</tr>
<tr>
<td>replacena</td>
<td>Method for replacing NA values in series matrix (mean by default)</td>
</tr>
</tbody>
</table>

Value

Vector of corresponding clusters, serialized to JSON.

Examples

```r
## Not run:
data(es)
performKmeans(es, k = 2)
## End(Not run)
```

queryAnnotationDBMeta

Get meta list for annotationDB files

Description

createES Function reads an rds file containing meta information of provided sqlite files for AnnotationDB

Usage

queryAnnotationDBMeta()

Value

meta info in JSON
Examples

```r
## Not run:
queryAnnotationDBMeta()

## End(Not run)
```

read.gct

*Reads ExpressionSet from a GCT file.*

**Description**

Only versions 1.2 and 1.3 are supported.

**Usage**

```r
read.gct(gct, ...)
```

**Arguments**

- `gct`  
  Path to gct file
- `...`  
  Additional options for read.csv

**Value**

ExpressionSet object

**Examples**

```r
read.gct(system.file("extdata", "centers.gct", package = "phantasus"))
```

reparseCachedESs

*Reparse cached expression sets from GEO.*

**Description**

The function should be used on phantasus version updates that change behavior of loading datasets from GEO. It finds all the datasets that were cached and runs `getES` for them again. The function uses cached Series and other files from GEO.

**Usage**

```r
```
Arguments
  destdir Directory used for caching loaded Series files from GEO database.
  mirrorPath URL string which specifies the source of matrices.

Value
  vector of previously cached GSE IDs

Examples
  reproduceCachedESs(destdir=tempdir())

---

reproduceInR Reproduce session in R code

Description
  Reproduce session in R code

Usage
  reproduceInR(sessionName, leaf = T, step = 0, savedEnv = new.env())

Arguments
  sessionName String, OCPU session name
  leaf Boolean, is it leaf (default = F)
  step Integer, step of recursion (default = 0)
  savedEnv Environment, where to store complex arguments (default = new.env())

Value
  JSON with R code

Examples
  ## Not run:
  setwd(tempdir())
  reproduceInR('x039f1672026678');

  ## End(Not run)
servePhantasus

Serve phantasus.

Description

servePhantasus starts http server handling phantasus static files and opencpu server.

Usage

servePhantasus(
  host = "0.0.0.0",
  port = 8000,
  staticRoot = system.file("www/phantasus.js", package = "phantasus"),
  cacheDir = tempdir(),
  preloadedDir = NULL,
  openInBrowser = TRUE,
  quiet = TRUE
)

Arguments

host  Host to listen.
port  Port to listen.
staticRoot  Path to static files with phantasus.js (on local file system).
cacheDir  Full path to cache directory.
preloadedDir  Full path to directory with preloaded files.
openInBrowser  Boolean value which states if application will be automatically loaded in default browser.
quiet  Boolean value which states whether the connection log should be hidden (default: TRUE)

Value

Running instance of phantasus application.

Examples

```r
## Not run:
servePhantasus()

## End(Not run)
```
shinyGAMAnalysis

Constructs data frame with gene annotations and submits it into Shiny GAM web-server

Description

Constructs data frame with gene annotations and submits it into Shiny GAM web-server

Usage

shinyGAMAnalysis(es)

Arguments

es Expression set object

Value

URL for Shiny GAM

subsetES

Subsets es, if rows or columns are not specified, all are retained

Description

Subsets es, if rows or columns are not specified, all are retained

Usage

subsetES(es, columns = c(), rows = c())

Arguments

es ExpressionSet object

columns List of specified columns’ indices (optional), indices start from 0

rows List of specified rows’ indices (optional), indices start from 0

Value

new expression set ‘es’
updateARCHS4 \hspace{1cm} \textit{Update archs4 files.}

\textbf{Description}

Download archs4 or archs4zoo counts in cacheDir. If directory does not exists function makes nothing and produce corresponding warnings.

\textbf{Usage}

\begin{verbatim}
updateARCHS4(
  cacheDir = file.path(getOption("phantasusCacheDir"), "counts/archs4"),
  organism = c("all"),
  force = FALSE
)
\end{verbatim}

\textbf{Arguments}

- **cacheDir** \hspace{1cm} file path to archs4 cache directory
- **organism** \hspace{1cm} vector which determines organisms to download: human, mouse, zoo or all as default. Also can be a genus. Possible genus:
  1. drosophila
  2. gallus
  3. bos
  4. caenorhabditis
  5. danio
  6. rattus
  7. saccharomyces
  8. arabidopsis
- **force** \hspace{1cm} logical value which let function replace current files

updateARCHS4meta \hspace{1cm} \textit{Update ARCHS4 meta files}

\textbf{Description}

Creates meta.txt file, which describes typical archs4 and archs4Zoo files.

\textbf{Usage}

\begin{verbatim}
updateARCHS4meta(
  archDir = file.path(getOption("phantasusCacheDir"), "counts/archs4")
)
\end{verbatim}
updateCountsMeta

Arguments

argDir | path to directory with arch4 .h5 files.

Details

This function produces very specific "hardcoded" meta.txt file for arch4 and archs4ZOO counts collections. See validateCountsCollection for more common information and meta.txt file structure.

See Also

validateCountsCollection

updateCountsMeta | Update meta-data for counts collections

Description

Creates meta.rda file which contain information about all samples in all collections. Also function checks priority.txt file. This file is used to manage collections with the same samples.

Usage

updateCountsMeta(
  counts_dir = file.path(getOption("phantasusCacheDir"), "counts"),
  force = FALSE,
  verbose = FALSE
)

Arguments

counts_dir | path to counts cache directory
force | logical value which lets function replace existing meta.rda file
verbose | logical value which determines a content of the output.

Details

First of all function checks validity of priority.txt file. Every Collection should have unique priority. If priority.txt is not valid function creates new one, setting priorities for each subdirectory=collection) equal to order in list.dir output.

Function updates meta.rda if this file is older than at least one .h5 file in counts files. meta.rda is data.table which is a result of union data.tables produced by getCountsMetaPart for each collection.

See Also

validateCountsCollection,updateCountsMeta @import data.table
validateCountsCollection

updateDEE2meta  

*Update DEE2 meta files*

**Description**

Creates `meta.txt` file, which describes typical dee2 files.

**Usage**

```r
updateDEE2meta(
  destDir = file.path(getOption("phantasusCacheDir"), "counts/dee2")
)
```

**Arguments**

- `destDir`  
  path to directory with DEE2 .h5 files.

**Details**

This function produces very specific "hardcoded" `meta.txt` file for dee2 counts collection. See `validateCountsCollection` for more common information and `meta.txt` file structure.

**See Also**

`validateCountsCollection`

validateCountsCollection

*Check a counts collection*

**Description**

Function checks existing and structure of `meta.txt` file in specified counts folder. Also it checks accessibility of specified datasets in corresponding .h5 files.

**Usage**

```r
validateCountsCollection(collectionDir, verbose = FALSE)
```

**Arguments**

- `collectionDir`  
  path to directory with collection
- `verbose`  
  logical value which determines a content of the output.
write.gct

Details

collectionDir should contain a bunch of .h5 files and a single meta.txt. meta.txt is .tsv-like file where for each .h5 exists a row with columns:

- **file_name** name of .h5 file in collectionDir.
- **sample_id** name of dataset in file_name which contains sample IDs (sample_geo_accession for example).
- **sample_dim** which dimension of the expression matrix in file_name corresponds to samples. Should be one of c("rows", "columns")
- **gene_id** name of dataset in file_name which contains ids for genes and the "meaning" for that ids (column name in result ES). For correct work this dataset should contain unique values. Example: ENSEMBLID:/meta/genes/ensembl_gene_id
- **genes_annot** Names of datasets and their meanings to extract gene-related metadata from file_name. Can be empty or gene_id-like values separated with semicolon(;

write.gct  Saves ExpressionSet to a GCT file (version 1.3).

Description

Saves ExpressionSet to a GCT file (version 1.3).

Usage

write.gct(es, file, gzip = FALSE)

Arguments

- **es** ExpressionSet object to save
- **file** Path to output gct file
- **gzip** Whether to gzip apply gzip-compression for the output file#

Value

Result of the closing file (as in 'close()' function')

Examples

es <- read.gct(system.file("extdata", "centers.gct", package = "phantasus"))
out <- tempfile(fileext = ".gct.gz")
write.gct(es, out, gzip=TRUE)
Index

* datasets
  es, 9
* internal
  adjustDataset, 3
  annotationDBMeta, 4
  calcPCA, 4
  calculateAnnotation, 5
  checkGPLsFallback, 6
  collapseDataset, 6
  colMeansByGroups, 7
  convertByAnnotationDB, 8
  createES, 8
  es, 9
  fgseaExample, 10
  generatePreloadedSession, 10
  getArchs4Files, 11
  getCountsMetaPart, 11, 12, 25
  getES, 12
  getGDS, 13
  getGSE, 14
  gseaPlot, 14
  limmaAnalysis, 15
  loadCounts, 16
  loadFromARCHS4, 17
  loadGEO, 17
  loadPreloaded, 18
  performKmeans, 19
  queryAnnotationDBMeta, 19
  read.gct, 20
  reparseCachedESs, 20
  reproduceInR, 21
  servePhantasus, 22
  shinyGAMAnalysis, 23
  subsetES, 23
  updateARCHS4, 24
  updateARCHS4meta, 24
  updateCountsMeta, 25, 25
  updateDEE2meta, 26
  validateCountsCollection, 11, 12, 25, 26, 26
  write.gct, 27