

# Package ‘phantasusLite’

February 21, 2025

**Type** Package

**Title** Loading and annotation RNA-seq counts matrices

**Version** 1.5.0

**Description** PhantasusLite – a lightweight package with helper functions of general interest extracted from phantasus package. In particular it simplifies working with public RNA-seq datasets from GEO by providing access to the remote HSDS repository with the precomputed gene counts from ARCHS4 and DEE2 projects.

**Depends** R (>= 4.2)

**Imports** data.table, rhdf5client(>= 1.25.1), httr, stringr, stats, utils, Biobase, methods

**biocViews** GeneExpression, Transcriptomics, RNASeq

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, BiocStyle, rhdf5, GEOquery

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**URL** <https://github.com/ctlab/phantasusLite/>

**BugReports** <https://github.com/ctlab/phantasusLite/issues>

**git\_url** <https://git.bioconductor.org/packages/phantasusLite>

**git\_branch** devel

**git\_last\_commit** 13aa8ea

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2025-02-20

**Author** Rita Sablina [aut],  
 Maxim Kleverov [aut],  
 Alexey Sergushichev [aut, cre]

**Maintainer** Alexey Sergushichev <alsergbox@gmail.com>

## Contents

createH5 . . . . .	2
createIndexH5 . . . . .	3
createIndexH5Remote . . . . .	3
createMetaH5 . . . . .	4
createPriorityH5 . . . . .	4
getCountsMetaPart . . . . .	5
getHSDSFileList . . . . .	5
getIndexRemote . . . . .	6
gsmToChunk . . . . .	6
inferCondition . . . . .	7
inferConditionImpl . . . . .	7
loadCountsFromH5FileHSDS . . . . .	8
loadCountsFromHSDS . . . . .	8
readGct . . . . .	9
removeRepeatWords . . . . .	10
updateARCHS4meta . . . . .	10
updateDEE2meta . . . . .	11
updateIndexH5 . . . . .	11
validateCountsCollection . . . . .	12
writeGct . . . . .	12
<b>Index</b>	<b>13</b>

---

createH5	<i>Creates metafiles for HDF5-files</i>
----------	---

---

### Description

Creates metafiles for HDF5-files

### Usage

```
createH5(data, file, dataset_name)
```

### Arguments

data	contains metadata
file	contains file name
dataset_name	contains dataset name

**Value**

Returns NULL

---

createIndexH5	<i>Writes indexes to the file</i>
---------------	-----------------------------------

---

**Description**

Writes indexes to the file

**Usage**

```
createIndexH5(data, file)
```

**Arguments**

data	contains metadata
file	contains the file name

**Value**

Returns NULL

---

createIndexH5Remote	<i>Creates HDF5-file with indexes</i>
---------------------	---------------------------------------

---

**Description**

Creates HDF5-file with indexes

**Usage**

```
createIndexH5Remote(
  url,
  collections = c("archs4", "dee2"),
  destfile = "index.h5"
)
```

**Arguments**

url	contains URL to the root of counts files
collections	vector of collection names to process
destfile	where to put resulting index file

**Value**

Returns NULL

---

createMetaH5	<i>Converts collection meta.txt files to meta.h5, putting them to the respective collection folders</i>
--------------	---

---

**Description**

Converts collection meta.txt files to meta.h5, putting them to the respective collection folders

**Usage**

```
createMetaH5(counts_dir)
```

**Arguments**

counts_dir	contains directory name
------------	-------------------------

**Value**

Returns NULL

---

createPriorityH5	<i>Creates HDF5-File with priority</i>
------------------	--

---

**Description**

Creates HDF5-File with priority

**Usage**

```
createPriorityH5(counts_dir, force = FALSE, verbose = FALSE)
```

**Arguments**

counts_dir	contains counts directory
force	logical value which lets function replace existing priority file
verbose	logical value which determines a content of the output.

**Value**

Returns NULL

---

getCountsMetaPart	<i>Gets list with metadata</i>
-------------------	--------------------------------

---

**Description**

Gets list with metadata

**Usage**

```
getCountsMetaPart(counts_dir, collection_name, verbose)
```

**Arguments**

counts_dir	contains counts directory
collection_name	contains name of the collection
verbose	logical value which determines a content of the output.

**Value**

list with metadata

---

getHSDSFileList	<i>Returns list of all HDF5-files on HSDS-server</i>
-----------------	--

---

**Description**

Returns list of all HDF5-files on HSDS-server

**Usage**

```
getHSDSFileList(  
  url = "https://alserglab.wustl.edu/hsds/?domain=/counts",  
  directory = NULL  
)
```

**Arguments**

url	containing url of the server and root domain.
directory	containing name of the directory

**Value**

List of all HDF5-files on the server or all files of the collection

**Examples**

```
url <- 'https://alserglab.wustl.edu/hsds/?domain=/counts'
getHSDSFileList(url)
```

---

getIndexRemote	<i>Creates a data table with indexes and chunks of samples in remote HDF5-files</i>
----------------	---

---

**Description**

Creates a data table with indexes and chunks of samples in remote HDF5-files

**Usage**

```
getIndexRemote(url, collections)
```

**Arguments**

url	contains url to the root of counts files
collections	contains names of the collections

**Value**

table with samples, indexes and chunks in all HDF5-files

---

gsmToChunk	<i>Gets chunk from GSE identifiers.</i>
------------	---

---

**Description**

Gets chunk from GSE identifiers.

**Usage**

```
gsmToChunk(samples)
```

**Arguments**

samples	containing a list of samples
---------	------------------------------

**Value**

list of chunks

---

inferCondition	<i>Adds condition to the annotation.</i>
----------------	--

---

**Description**

Adds condition to the annotation.

**Usage**

```
inferCondition(es)
```

**Arguments**

es                    contains ExpressionSet object

**Value**

Annotated ExpressionSet with conditions and replicates

**Examples**

```
ess <- GEOquery::getGEO("GSE143903")
es <- ess[[1]]
es <- inferCondition(es)
es$condition # contains inferred groups
es$replicate # contains inferred replicate numbers
```

---

inferConditionImpl	<i>Creates condition from the samples titles</i>
--------------------	--

---

**Description**

Creates condition from the samples titles

**Usage**

```
inferConditionImpl(gse_titles)
```

**Arguments**

gse\_titles            contains titles

**Value**

List of conditions and replicates

loadCountsFromH5FileHSDS

*Load count matrix from remote HDF5-file*

---

### Description

Load count matrix from remote HDF5-file

### Usage

```
loadCountsFromH5FileHSDS(  
  es,  
  url = "https://alserglab.wustl.edu/hsds/?domain=/counts",  
  file,  
  sampleIndexes = NULL  
)
```

### Arguments

es	containing ExpressionSet loaded from GEO. Contains empty expression matrix.
url	containing url of the server and root domain.
file	containing name of the file (relative to the root domain)
sampleIndexes	containing sample indexes list

### Value

ExpressionSet object with loaded count matrix

### Examples

```
ess <- GEOquery::getGEO("GSE53053")  
es <- ess[[1]]  
url <- 'https://alserglab.wustl.edu/hsds/?domain=/counts'  
file <- "/dee2/mmusculus_star_matrix_20240409.h5"  
es <- loadCountsFromH5FileHSDS(es, url, file)
```

---

loadCountsFromHSDS

*Load count matrix from HDF5-files.*

---

### Description

Load count matrix from HDF5-files.



**Usage**

```
loadCountsFromHSDS(
  es,
  url = "https://alserglab.wustl.edu/hsds/?domain=/counts"
)
```

**Arguments**

`es` containing ExpressionSet loaded from GEO. Contains empty expression matrix.  
`url` containing url of the server and root domain.

**Value**

ExpressionSet with loaded count matrix

**Examples**

```
ess <- GEOquery::getGEO("GSE85653")
es <- ess[[1]]
url <- 'https://alserglab.wustl.edu/hsds/?domain=/counts'
es <- loadCountsFromHSDS(es, url)
```

---

readGct	<i>Reads ExpressionSet from a GCT file.</i>
---------	---

---

**Description**

Only versions 1.2 and 1.3 are supported.

**Usage**

```
readGct(gct)
```

**Arguments**

`gct` Path to gct file

**Value**

ExpressionSet object

**Examples**

```
es <- readGct(system.file("extdata/testdata/gct/test.gct", package="phantasusLite"))
```

---

removeRepeatWords	<i>Removes repeated words from conditions</i>
-------------------	---

---

**Description**

Removes repeated words from conditions

**Usage**

```
removeRepeatWords(titles)
```

**Arguments**

titles	contains titles
--------	-----------------

**Value**

titles without repeated words

---

updateARCHS4meta	<i>Creates meta.txt file, which describes typical archs4 and archs4Zoo files.</i>
------------------	---

---

**Description**

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

**Usage**

```
updateARCHS4meta(  
  archDir = file.path(getOption("phantasusCacheDir"), "counts/archs4")  
)
```

**Arguments**

archDir	path to directory with arch4 .h5 files.
---------	---

**Value**

Returns NULL

---

updateDEE2meta	<i>Creates meta.txt file, which describes typical dee2 files.</i>
----------------	---

---

**Description**

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

**Usage**

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

**Arguments**

destDir            path to directory with DEE2 .h5 files.

**Value**

Returns NULL

---

updateIndexH5	<i>Updates indexes from HDF5-files</i>
---------------	--

---

**Description**

Updates indexes from HDF5-files

**Usage**

```
updateIndexH5(counts_dir, force = FALSE, verbose = FALSE)
```

**Arguments**

counts\_dir        contains counts directory  
force             logical value which lets function replace existing index file  
verbose           logical value which determines a content of the output.

**Value**

Returns NULL

---

validateCountsCollection  
*Validates counts collection*

---

**Description**

Validates counts collection

**Usage**

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

**Arguments**

collectionDir contains directory name  
verbose logical value which determines a content of the output.

**Value**

false if collection is not valid

---

writeGct *Saves ExpressionSet to a GCT file (version 1.3).*

---

**Description**

Saves ExpressionSet to a GCT file (version 1.3).

**Usage**

```
writeGct(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 <- readGct(system.file("extdata/testdata/gct/test.gct", package="phantasusLite"))
out <- tempfile(fileext = ".gct.gz")
writeGct(es, out, gzip=TRUE)
```

# Index

## \* **internal**

- createH5, [2](#)
- createIndexH5, [3](#)
- createIndexH5Remote, [3](#)
- createMetaH5, [4](#)
- createPriorityH5, [4](#)
- getCountsMetaPart, [5](#)
- getIndexRemote, [6](#)
- gsmToChunk, [6](#)
- inferConditionImpl, [7](#)
- removeRepeatWords, [10](#)
- updateARCHS4meta, [10](#)
- updateDEE2meta, [11](#)
- updateIndexH5, [11](#)
- validateCountsCollection, [12](#)

- createH5, [2](#)
- createIndexH5, [3](#)
- createIndexH5Remote, [3](#)
- createMetaH5, [4](#)
- createPriorityH5, [4](#)

- getCountsMetaPart, [5](#)
- getHSDSFileList, [5](#)
- getIndexRemote, [6](#)
- gsmToChunk, [6](#)

- inferCondition, [7](#)
- inferConditionImpl, [7](#)

- loadCountsFromH5FileHSDS, [8](#)
- loadCountsFromHSDS, [8](#)

- readGct, [9](#)
- removeRepeatWords, [10](#)

- updateARCHS4meta, [10](#)
- updateDEE2meta, [11](#)
- updateIndexH5, [11](#)

- validateCountsCollection, [12](#)

- writeGct, [12](#)