

# Package ‘phantasus’

October 16, 2019

**Title** Visual and interactive gene expression analysis

**Version** 1.4.0

**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, opencpu, assertthat, methods, httr, rhdf5, utils, parallel, stringr, fgsea (>= 1.9.4), svglite, gtable, stats, Matrix, Matrix.utils, pheatmap, scales, ccaPP, grid, grDevices, AnnotationDbi

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.1

**Suggests** testthat, BiocStyle, knitr, rmarkdown, data.table

**VignetteBuilder** knitr

**NeedsCompilation** no

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

**git\_branch** RELEASE\_3\_9

**git\_last\_commit** 86a00a4

**git\_last\_commit\_date** 2019-05-02

**Date/Publication** 2019-10-15

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

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

## R topics documented:

|                       |    |
|-----------------------|----|
| adjustDataset         | 2  |
| annotationDBMeta      | 3  |
| calcPCA               | 4  |
| calculatedAnnotation  | 4  |
| checkGPLs             | 5  |
| collapseDataset       | 6  |
| convertByAnnotationDB | 6  |
| createES              | 7  |
| es                    | 8  |
| getArchs4Files        | 8  |
| getES                 | 9  |
| getGDS                | 9  |
| getGSE                | 10 |
| gseaPlot              | 11 |
| limmaAnalysis         | 11 |
| loadFromARCHS4        | 12 |
| loadGEO               | 12 |
| loadPreloaded         | 13 |
| performKmeans         | 14 |
| queryAnnotationDBMeta | 14 |
| read.gct              | 15 |
| reparseCachedESs      | 15 |
| servePhantasia        | 16 |
| shinyGAMAnalysis      | 16 |
| subsetES              | 17 |
| write.gct             | 17 |

|              |           |
|--------------|-----------|
| <b>Index</b> | <b>19</b> |
|--------------|-----------|

---

|               |                       |
|---------------|-----------------------|
| adjustDataset | <i>Adjust dataset</i> |
|---------------|-----------------------|

---

### 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 $\log_2(1+x)$ 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**

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

## End(Not run)
```

---

|                  |  |
|------------------|--|
| annotationDBMeta | <i>Create meta file for AnnotationDB</i> |
|------------------|--|

---

**Description**

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

**Usage**

```
annotationDBMeta(cacheDir)
```

**Arguments**

|          |                         |
|----------|-------------------------|
| cacheDir | cacheDir for phantastus |
|----------|-------------------------|

**Value**

nothing

**Examples**

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

## End(Not run)
```

---

|         |                                      |
|---------|--------------------------------------|
| calcPCA | <i>Principal Component Analysis.</i> |
|---------|--------------------------------------|

---

**Description**

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

**Usage**

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

**Arguments**

|           |  |
|-----------|--|
| es        | an ExpressionSet object, should be normalized    |
| replacena | method for replacing NA values (mean by default) |

**Value**

json with full description of the plot for plotly.js

**Examples**

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

## End(Not run)
```

---

|                      |                                     |
|----------------------|-------------------------------------|
| calculatedAnnotation | <i>Create calculated annotation</i> |
|----------------------|-------------------------------------|

---

**Description**

calculatedAnnotation adds a column calculated by operation

**Usage**

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

**Arguments**

|           |   |
|-----------|---|
| es        | ExpressionSet object.   |
| operation | Name of the operation to perform calculation                          |
| rows      | List of specified rows' indices (optional), indices start from 0      |
| columns   | List of specified columns' indices (optional), indices start from 0#' |
| isColumns | Apply fn to columns   |
| name      | Name of the new annotation  |

**Value**

Nothing. Annotated dataset will be assigned to es in environment

---

|           |  |
|-----------|--|
| checkGPLs | <i>Check possible annotations for GEO Dataset.</i> |
|-----------|--|

---

**Description**

checkGPLs returns GPL-names for the specified GEO identifier.

**Usage**

```
checkGPLs(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 | <i>Collapse dataset</i> |
|-----------------|-------------------------|

---

**Description**

collapseDataset performs a collapse action on expression set

**Usage**

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

**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**

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

## End(Not run)
```

---

|                       |  |
|-----------------------|--|
| convertByAnnotationDB | <i>Map indexes using Annotation DB</i> |
|-----------------------|--|

---

**Description**

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

**Usage**

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

**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                                |

**Value**

JSON object with a vector of converted IDs

---

|          |                              |
|----------|------------------------------|
| createES | <i>Create ExpressionSet.</i> |
|----------|------------------------------|

---

**Description**

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

**Usage**

```
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 | <i>Example dataset</i> |
|----|------------------------|

---

**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)
```

---

|                |   |
|----------------|---|
| getArchs4Files | <i>Returns list of ARCHS4 hdf5 files with expression data</i> |
|----------------|---|

---

**Description**

Returns list of ARCHS4 hdf5 files with expression data

**Usage**

```
getArchs4Files(cacheDir)
```

**Arguments**

cacheDir      base directory for cache

**Value**

list of .h5 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(),  
      mirrorPath = "https://ftp.ncbi.nlm.nih.gov")
```

**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.

**Examples**

```
## Not run:  
  getES('GSE14308', type = 'GSE', destdir = 'cache')  
  getES('GSE27112')  
  
## End(Not run)  
  getES('GDS4922')
```

---

getGDS *Load ExpressionSet from GEO Datasets*

---

**Description**

getGDS return the ExpressionSet object corresponding to GEO Dataset identifier.

**Usage**

```
getGDS(name, destdir = tempdir(),  
      mirrorPath = "https://ftp.ncbi.nlm.nih.gov")
```

**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 | <i>Load ExpressionSet from GEO Series</i> |
|--------|---|

---

**Description**

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

**Usage**

```
getGSE(name, destdir = tempdir(),
        mirrorPath = "https://ftp.ncbi.nlm.nih.gov")
```

**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**

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

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

---

|          |   |
|----------|---|
| gseaPlot | <i>Returns path to an svg file with enrichment plot</i> |
|----------|---|

---

**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 | <i>Differential Expression analysis.</i> |
|---------------|--|

---

**Description**

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

**Usage**

```
limmaAnalysis(es, fieldValues)
```

**Arguments**

|             |   |
|-------------|---|
| es          | ExpressionSet object. It should be normalized for more accurate analysis. |
| fieldValues | Vector of comparison values, mapping categories' names to columns/samples |

**Value**

Name of the file containing serialized de-matrix.

**Examples**

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

## End(Not run)
```

---

|                |   |
|----------------|---|
| loadFromARCHS4 | <i>Loads expression data from ARCHS4 count files. Only samples with counted expression are kept. If es already contains expression data it is returned as is.</i> |
|----------------|---|

---

**Description**

Loads 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 | <i>Load GEO Dataset.</i> |
|---------|--------------------------|

---

**Description**

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

**Usage**

```
loadGEO(name, type = NA)
```

**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.   |

**Value**

File with ProtoBuf-serialized ExpressionSet-s that were downloaded by this identifier. For GSE-datasets there can be multiple annotations, so in file will be a list mapping name with GPL to ExpressionSet.

**Examples**

```
## Not run:
  loadGEO("GSE27112")
  loadGEO("GDS4922")

## End(Not run)
```

---

|               |                          |
|---------------|--------------------------|
| loadPreloaded | <i>Load GEO Dataset.</i> |
|---------------|--------------------------|

---

**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

|           |   |
|-----------|---|
| es        | ExpressionSet object.   |
| k         | Expected number of clusters.                                      |
| replacena | Method for replacing NA values in series matrix (mean by default) |

### Value

Vector of corresponding clusters, serialized to JSON.

### Examples

```
## 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

```
## Not run:
queryAnnotationDBMeta()

## End(Not run)
```

---

|          |   |
|----------|---|
| read.gct | <i>Reads ExpressionSet from a GCT file.</i> |
|----------|---|

---

**Description**

Only versions 1.2 and 1.3 are supported.

**Usage**

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

**Arguments**

|     |                                 |
|-----|---------------------------------|
| gct | Path to gct file                |
| ... | additional options for read.csv |

**Value**

ExpressionSet object

**Examples**

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

---

|                  |   |
|------------------|---|
| reparseCachedESs | <i>Reparse cached expression sets from GEO.</i> |
|------------------|---|

---

**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**

```
reparseCachedESs(destdir, mirrorPath = "https://ftp.ncbi.nlm.nih.gov")
```

**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**

```
reparseCachedESs(destdir=tempdir())
```

---

|                |                         |
|----------------|-------------------------|
| servePhantasus | <i>Serve phantasus.</i> |
|----------------|-------------------------|

---

**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**

```
## Not run:
servePhantasus()

## End(Not run)
```

---

|                  |   |
|------------------|---|
| shinyGAMAnalysis | <i>Constructs data frame with gene annotations and submits it into Shiny GAM web-server</i> |
|------------------|---|

---

**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 | <i>Subsets es, if rows or columns are not specified, all are retained</i> |
|----------|---|

---

**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'

---

|           |   |
|-----------|---|
| write.gct | <i>Saves ExpressionSet to a GCT file (version 1.3).</i> |
|-----------|---|

---

**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#'  
...                    additional options for read.csv

**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

## \*Topic **datasets**

es, [8](#)

adjustDataset, [2](#)

annotationDBMeta, [3](#)

calcPCA, [4](#)

calculatedAnnotation, [4](#)

checkGPLs, [5](#)

collapseDataset, [6](#)

convertByAnnotationDB, [6](#)

createES, [7](#)

es, [8](#)

getArchs4Files, [8](#)

getES, [9](#)

getGDS, [9](#)

getGSE, [10](#)

gseaPlot, [11](#)

limmaAnalysis, [11](#)

loadFromARCHS4, [12](#)

loadGEO, [12](#)

loadPreloaded, [13](#)

performKmeans, [14](#)

queryAnnotationDBMeta, [14](#)

read.gct, [15](#)

reparseCachedESs, [15](#)

servePhantastus, [16](#)

shinyGAMAnalysis, [16](#)

subsetES, [17](#)

write.gct, [17](#)