Package: artMS (via r-universe)

November 25, 2026

Type Package

Title Analytical R tools for Mass Spectrometry

Version 1.32.0

Date 2026-11-16

Description artMS provides a set of tools for the analysis of proteomics label-free datasets. It takes as input the MaxQuant search result output (evidence.txt file) and performs quality control, relative quantification using MSstats, downstream analysis and integration. artMS also provides a set of functions to re-format and make it compatible with other analytical tools, including, SAINTq, SAINTexpress, Phosfate, and PHOTON. Check http://artms.org for details.

License GPL (>= 3) + file LICENSE

URL http://artms.org

BugReports https://github.com/biodavidjm/artMS/issues

Depends R (>= 4.1.0)

Imports AnnotationDbi, bit64, circlize, cluster, corrplot, data.table, dplyr, getopt, ggdendro, ggplot2, gplots, ggrepel, graphics, grDevices, grid, limma, MSstats, openxlsx, org.Hs.eg.db, pheatmap, plotly, plyr, RColorBrewer, scales, seqinr, stats, stringr, tidyr, UpSetR, utils, VennDiagram, yaml

Suggests BiocStyle, ComplexHeatmap, factoextra, FactoMineR, gprofiler2, knitr, PerformanceAnalytics, org.Mm.eg.db, rmarkdown, testthat

Encoding UTF-8

LazyLoad yes

LazyData true

biocViews Proteomics, DifferentialExpression, BiomedicalInformatics, SystemsBiology, MassSpectrometry, Annotation, QualityControl, GeneSetEnrichment, Clustering, Normalization, ImmunoOncology, MultipleComparison

NeedsCompilation no

VignetteBuilder knitr

RoxygenNote 7.1.1

Roxygen list(markdown = TRUE)

Config/pak/sysreqs cmake make libicu-dev libpng-dev libuv1-dev libssl-dev zlib1g-dev

Repository Bioconductor 3.24

Date/Publication 2026-11-23 22:41:50 UTC

RemoteUrl https://github.com/bioc/artMS

RemoteRef RELEASE_3_24

RemoteSha 318273c04260bf9199afd95116446eee36ca4eb0

Contents

artms_config . . . 3
artms_data_corum_mito_database . . . 4
artms_data_pathogen_LPN . . . 5
artms_data_pathogen_TB . . . 6
artms_data_ph_config . . . 6
artms_data_ph_contrast . . . 7
artms_data_ph_evidence . . . 8
artms_data_ph_keys . . . 8
artms_data_ph_msstats_modelqc . . . 9
artms_data_ph_msstats_results . . . 9
artms_data_randomDF . . . 10
artmsAnalysisQuantifications . . . 10
artmsAnnotateSpecie . . . 13
artmsAnnotationUniprot . . . 14
artmsAvgIntensityRT . . . 15
artmsChangeColumnName . . . 16
artmsConvertMetabolomics . . . 16
artmsDataPlots . . . 17
artmsEnrichLog2fc . . . 18
artmsEnrichProfiler . . . 19
artmsEvidenceToSaintExpress . . . 21
artmsEvidenceToSAINTq . . . 22
artmsFilterEvidenceContaminants . . . 23
artmsGeneratePhSiteExtended . . . 24
artmsIsEvidenceNewVersion . . . 25
artmsIsSpeciesSupported . . . 26
artmsLeaveOnlyUniprotEntryID . . . 27
artmsMapUniprot2Entrez . . . 28
artmsMergeEvidenceAndKeys . . . 28
artmsMsstatsSummary . . . 29
artmsPhosfateOutput . . . 30
artmsPhotonOutput . . . 31

artmsPlotHeatmapQuant 32
artmsProtein2SiteConversion 33
artmsQualityControlEvidenceBasic 35
artmsQualityControlEvidenceExtended 37
artmsQualityControlMetabolomics 40
artmsQualityControlSummaryExtended 41
artmsQuantification 43
artmsResultsWide 44
artmsSILACtoLong 45
artmsSpectralCounts 46
artmsVolcanoPlot 46
artmsWriteConfigYamlFile 47

Index **49**

`artms_config` *artMS configuration template*

Description

The configuration file in `yaml` format contains the configuration details required to run `artmsQuantification()`, which includes quality control functions

Usage

```
artms_config
```

Format

The configuration (`yaml`) file contains the following sections:

files
- `evidence` : /path/to/the/evidence.txt
- `keys` : /path/to/the/keys.txt
- `contrasts` : /path/to/the/contrast.txt
- `summary` : /path/to/the/summary.txt
- `output` : /path/to/the/output/results/results.txt

qc
- basic: 1 # 1 = yes; 0 = no
- extended: 1 # 1 = yes; 0 = no
- extendedSummary: 0 # 1 = yes; 0 = no

data
- enabled : 1 # 1 = yes; 0 = no
- silac:
  - enabled : 0 # 1 for SILAC experiments
- filters:
  - enabled : 1
- contaminants : 1
- protein_groups : remove #remove, keep

- modifications : ab # PH, UB, AB, APMS
- sample_plots : 1 # correlation plots

msstats

- enabled : 1
- msstats_input : # blank if not previous msstats input file is available
- profilePlots : none # before, after, before-after, none
- normalization_method : equalizeMedians # globalStandards (include a reference protein(s)), equalizeMedians, quantile, 0
- normalization_reference : #should be a value in the Protein column
- summaryMethod : TMP # "TMP"(default) means Tukey's median polish, which is robust estimation method. "linear" uses linear mixed model. "logOfSum" conducts log2 (sum of intensities) per run.
- censoredInt : NA # Missing values are censored or at random. 'NA' (default) assumes that all 'NA's in 'Intensity' column are censored. '0' uses zero intensities as censored intensity. In this case, NA intensities are missing at random. The output from Skyline should use '0'. Null assumes that all NA intensites are randomly missing.
- MBimpute : 1 # only for summaryMethod="TMP" and censoredInt='NA' or '0'. TRUE (default) imputes 'NA' or '0' (depending on censoredInt option) by Accelated failure model. FALSE uses the values assigned by cutoffCensored
- For all othe features, please check documentation for MSstats' dataProcess function

output_extras

- output_extras :
  - enabled : 1 # if 0, it wont do anything in this section
- annotate :
  - enabled: 1 # 1|0 whether to annotate the proteins in the results or not
- species : HUMAN # Supported species: HUMAN, MOUSE, ANOPHELES, ARABIDOPSIS, BOVINE, WORM, CANINE, FLY, ZEBRAFISH, ECOLI_STRAIN_K12, ECOLI_STRAIN_SAKAI, CHICKEN, RHESUS, MALARIA, CHIMP, RAT, YEAST, PIG, XENOPUS
- plots:
  - volcano: 1
  - heatmap: 1
  - LFC : -1.5 1.5 # Range of minimal log2fc
  - FDR : 0.05
  - heatmap_cluster_cols : 0
  - heatmap_display : log2FC # log2FC or pvalue

`artms_data_corum_mito_database`

CORUM Protein Complexes database use for complex enrichment analysis

---

Description

The list of protein complexes has been enriched with mitochondria proteins from mouse, as described in this paper:

2018 - Ruchi Masand, Esther Paulo, Dongmei Wu , Yangmeng Wang, Danielle L. Swaney, David Jimenez-Morales, Nevan J. Krogan, and Biao Wang Proteome Imbalance of Mitochondrial Electron Transport Chain in Brown Adipocytes Leads to Metabolic Benefits. Cell Metab. 2018 Mar 06; 27(3):616-629.e4

Usage

```
artms_data_corum_mito_database
```

Format

Tab delimited file.

To find out more about the format and columns available at CORUM, please visit this link

Details

LAST CORUM DOWNLOAD DATE: 2017-08-01

`artms_data_pathogen_LPN`

LPN PATHOGEN: Legionella pneumophila subsp. pneumophila (strain Philadelphia 1 / ATCC 33152 / DSM 7513) UNIPROT IDS

Description

LPN PATHOGEN: Legionella pneumophila subsp. pneumophila (strain Philadelphia 1 / ATCC 33152 / DSM 7513) UNIPROT IDS

Usage

```
artms_data_pathogen_LPN
```

Format

A data.frame of Entry IDs

artms_data_pathogen_TB

TB PATHOGEN: Mycobacterium tuberculosis (strain ATCC 35801 / TMC 107 / Erdman) UNIPROTS IDS

Description

TB PATHOGEN: Mycobacterium tuberculosis (strain ATCC 35801 / TMC 107 / Erdman) UNIPROTS IDS

Usage

```
artms_data_pathogen_TB
```

Format

A data.frame of Entry IDs

artms_data_ph_config *artMS configuration for the available PH dataset*

Description

The configuration file with default options to run the available PH dataset with 'artmsQuantification()"

Usage

```
artms_data_ph_config
```

Format

The configuration (`yaml`) file contains the following sections:

files
- `evidence` : Empty. To test an example, run `artms_data_ph_config$files$evidence <- artms_data_ph_evidence`
- `keys` : Empty To test an example datasets run `artms_data_ph_config$files$keys <- artms_data_ph_keys`
- `contrasts` : Empty. To test the example datasets, run `artms_data_ph_config$files$contrasts <- artms_data_ph_contrast`
- `summary` :
- `output` : "results.txt"

qc
- basic: 0
- extended: 0
- extendedSummary: 0

data
- enabled : 1
- silac:
  - enabled : 0
- filters:
  - enabled : 1
- contaminants : 1
- protein_groups : remove
- modifications : PH
- sample_plots : 1

msstats
- enabled : 1
- msstats_input : # blank if not previous msstats input file is available
- profilePlots : none # before, after, before-after, none
- normalization_method : equalizeMedians
- normalization_reference : #should be a value in the Protein column
- summaryMethod : TMP
- censoredInt : NA
- cutoffCensored : minFeature
- MBimpute : 1
- feature_subset: all

output_extras
- output_extras :
  - enabled : 1
- annotate :
  - enabled: 1
- species : HUMAN
- plots:
  - volcano: 1
  - heatmap: 1
  - LFC : -1 1
  - FDR : 0.05
  - heatmap_cluster_cols : 0
  - heatmap_display : log2FC

`artms_data_ph_contrast`

Contrast example for the PH dataset

Description

Contrast file with the relative quantification to be performed for the two conditions available in the example dataset: "Cal33-HSC6". See vignette for more details on how to prepare the contrast file.

Usage

```
artms_data_ph_contrast
```

Format

list with one comparison: "Cal33-HSC6"

`artms_data_ph_evidence` *Evidence file example*

Description

Evidence file from a PH experiment consisting of two head and neck cancer cell lines ("Conditions" `"Cal33"` and `"HSC6"`).

Unfortunately, the number of lines was reduced to 1/20 due to bioconductor limitations on data size, but it should be enough to test the qc and quantification functions. The number of total columns from the original evidence file was also reduced to 36 (out of the original 66 columns). Check `colnames(artms_data_ph_evidence)` for details

Usage

```
artms_data_ph_evidence
```

Format

A data frame with all the columns available in an evidence file generated with MaxQuant version 1.6.2.3

`artms_data_ph_keys` *Keys File Example*

Description

the `artMS` keys file provides the details of the experimental design for any given proteomics experiment.

This particular example belongs to a PH experiment consisting of two head and neck cancer cell lines ("Conditions" `"Cal33"` and `"HSC6"`), with 2 biological replicates each (in this reduced version)

Usage

```
artms_data_ph_keys
```

Format

Tab delimited file with the following columns:

Raw.file Raw file processed. Each one should be a unique biological (or technical) replicate

IsotopeLabelType Type of labeling. `L` is used for label free experiments

Condition Label for conditions. VERY IMPORTANT: Only alpha-numeric characters and `underscore (_)` are allowed

BioReplicate Label for the Biological replicates. VERY IMPORTANT: Use the same labeling for bioreplicate as the Condition, but adding a `dash (-)` corresponding to the number of biological replicate. For example, for `Condition "Cal"`, use `Cal-1`, `Cal-2`, `Cal-3`, etc for the bioreplicates

Run The MS run number

---

artms_data_ph_msstats_modelqc

MSstats modelQC example

Description

Normalized data obtained from the `artmsQuantification()` step of the PH dataset (global analysis)

Usage

```
artms_data_ph_msstats_modelqc
```

Format

A data frame resulting from running the latest version of `MSstats::groupComparison` function required as input for artmsAnalysisQuantifications()

---

artms_data_ph_msstats_results

MSstats results example

Description

Relative quantification results obtained running MSstats on the available PH datasets (global analysis). Changes in protein phosphorylation were quantified between two conditions (check `artms_data_ph_contrast`)

Usage

```
artms_data_ph_msstats_results
```

Format

A data frame resulting from running the latest version of MSstats

artms_data_randomDF *Random data set*

Description

Dataset randomly generated for testing purposes

Usage

```
artms_data_randomDF
```

Format

A data frame with 100 rows and 10 variables:

Dataset generated using this code

```
data.frame(replicate(10,sample(0:1,100,rep=TRUE)))
```

artmsAnalysisQuantifications *Analysis of the Relative Quantifications*

Description

Analysis of relative quantifications, including:

- Annotations
- Summary files in different format (xls, txt) and shapes (long, wide)
- Numerous summary plots
- Enrichment analysis using Gprofiler
- PCA of quantifications
- Clustering analysis
- Basic imputation of missing values

To run this function, the following packages must be installed on your system:

- From bioconductor: `BiocManager::install(c("ComplexHeatmap", "org.Mm.eg.db"))`
- From CRAN: `install.packages(c("factoextra", "FactoMineR", "gprofiler2", "PerformanceAnalytics"))`

Usage

```
artmsAnalysisQuantifications(
  log2fc_file,
  modelqc_file,
  species,
  output_dir = "analysis_quant",
  outliers = c("keep", "iqr", "std"),
  enrich = TRUE,
  l2fc_thres = 1,
  choosePvalue = c("adjpvalue", "pvalue"),
  isBackground = "nobackground",
  isPtm = "global",
  mnbr = 2,
  pathogen = "nopathogen",
  plotPvaluesLog2fcDist = TRUE,
  plotAbundanceStats = TRUE,
  plotReproAbundance = TRUE,
  plotCorrConditions = TRUE,
  plotCorrQuant = TRUE,
  plotPCAabundance = TRUE,
  plotFinalDistributions = TRUE,
  plotPropImputation = TRUE,
  plotHeatmapsChanges = TRUE,
  plotTotalQuant = TRUE,
  plotClusteringAnalysis = TRUE,
  data_object = FALSE,
  printPDF = TRUE,
  verbose = TRUE
)
```

Arguments

`log2fc_file`	(char) MSstats results file location
`modelqc_file`	(char) MSstats modelqc file location
`species`	(char) Select one species. Species currently supported for a full analysis (including enrichment analysis): • HUMAN • MOUSE To find out species supported only for annotation check `?artmsIsSpeciesSupported()`
`output_dir`	(char) Name for the folder to output the results from the function. Default is current directory (recommended to provide a new folder name).
`outliers`	(char) It allows to keep or remove outliers. Options: • `keep` (default): it keeps outliers 'keep', 'iqr', 'std' • `iqr` (recommended): remove outliers +/- 6 x Interquartile Range (IQR) • `std` : 6 x standard deviation
`enrich`	(logical) Performed enrichment analysis using GprofileR? Only available for species HUMAN and MOUSE. `TRUE` (default if "human" or "mouse" are the species) or `FALSE`

`l2fc_thres` (int) log2fc cutoff for enrichment analysis (default, `l2fc_thres = 1.5`)

`choosePvalue` (char) specify whether `pvalue` or `adjpvalue` should use for the analysis. The default option is `adjpvalue` (multiple testing correction). But if the number of biological replicates for a given experiment is too low (for example n = 2), then `choosePvalue = pvalue` is recommended.

`isBackground` (char) background of gene names for enrichment analysis. `nobackground` (default) will use the total number of genes detected. Alternatively provided the file path name to the background gene list.

`isPtm` (char) Is a ptm-site quantification?

- `global` (default),
- `ptmsites` (for site specific analysis),
- `ptmph` (Jeff Johnson script output evidence file)

`mnbr` (int) **PARAMETER FOR NAIVE IMPUTATION**: "minimal number of biological replicates" for "naive imputation" and filtering. Default: `mnbr = 2`. Details: Intensity values for proteins/PTMs that are completely missed in one of the two conditions compared ("condition A"), but are found in at least 2 biological replicates (`mnbr = 2`) of the other "condition B", are imputed (values artificially assigned) and the log2FC values calculated. The goal is to keep those proteins/PTMs that are consistently found in one of the two conditions (in this case "condition B") and facilitate the inclusion in downstream analysis (if wished). The imputed intensity values are sampled from the lowest intensity values detected in the experiment, and (**WARNING**) the p-values are just randomly assigned between 0.05 and 0.01 for illustration purposes (when generating a volcano plot with the output of `artmsAnalysisQuantifications`) or to include them when making a cutoff of p-value < 0.05 for enrichment analysis **CAUTION**: `mnbr` would also add the constraint that any protein must be identified in at least `nmbr` biological replicates of the **same** condition or it will be filtered out. That is, if `mnbr = 2`, a protein found in two conditions but only in one biological replicate in each of them, it would be removed.

`pathogen` (char) Is there a pathogen in the dataset as well? if it does not, then use `pathogen = nopathogen` (default). Pathogens available: `tb` (Tuberculosis), `lpn` (Legionella)

`plotPvaluesLog2fcDist` (logical) If `TRUE` (default) plots pvalues and log2fc distributions

`plotAbundanceStats` (logical) If `TRUE` (default) plots stats graphs about abundance values

`plotReproAbundance` (logical) If `TRUE` plots reproducibility based on normalized abundance values

`plotCorrConditions` (logical) If `TRUE` plots correlation between the different conditions

`plotCorrQuant` (logical) if `TRUE` plots correlation between the available quantifications (comparisons)

`plotPCAabundance` (logical) if `TRUE` performs PCA analysis of conditions using normalized abundance values

`plotFinalDistributions` (logical) if `TRUE` plots distribution of both log2fc and pvalues

plotPropImputation
: (logical) if TRUE plots proportion of overall imputation

plotHeatmapsChanges
: (logical) if TRUE plots heatmaps of quantified changes (both all and significant only). Only if printPDF is also TRUE

plotTotalQuant
: (logical) if TRUE plots barplot of total number of quantifications per comparison

plotClusteringAnalysis
: (logical) if TRUE performs clustering analysis between quantified comparisons (more than 1 comparison required)

data_object
: (logical) flag to indicate whether the required files are data objects. Default is FALSE

printPDF
: If TRUE (default) prints out the pdfs. Warning: plot objects are not returned due to the large number of them.

verbose
: (logical) TRUE (default) shows function messages

Value

(data.frame) summary of quantifications, including annotations, enrichments, etc

Examples

```
# Testing that the files cannot be empty
artmsAnalysisQuantifications(log2fc_file = NULL,
                             modelqc_file = NULL,
                             species = NULL,
                             output_dir = NULL)
```

---

artmsAnnotateSpecie *Adding a column with the species name*

Description

Adding the species name to every protein. This makes more sense if there are more than one species in the dataset, which must be specified in the pathogen option. Influenza is a special case that it does not need to be specified, as far as the proteins were originally annotated as INFLUENZAGENE_STRAIN (strains covered H1N1, H3N2, H5N1), as for example, NS1_H1N1

Usage

```
artmsAnnotateSpecie(df, pathogen = "nopathogen", species)
```

Arguments

df
: (data.frame) with a Protein column (of uniprot ids)

pathogen
: (char) Is there a pathogen in the dataset as well? if it does not, then use pathogen = nopathogen (default). Supportedtb (Tuberculosis), lpn (Legionella)

species
: (char) Host organism (supported for now: human or mouse)

Value

(data.frame) The same data.frame but with an extra column specifying the species

Examples

```
# Adding a new column with the main species of the data. Easy.
# But the main functionality is to add both the host-species and a pathogen,
# which is not illustrated in this example
data_with_specie <- artmsAnnotateSpecie(df = artms_data_ph_msstats_results,
                                        species = "human")
```

`artmsAnnotationUniprot`

Annotate table with Gene Symbol and Name based on Uniprot ID(s)

Description

Annotate gene name and symbol based on uniprot ids. It will take the column from your data.frame specified by the `columnid` argument, search for the gene symbol, name, and entrez based on the species (`species` argument) and merge the information back to the input data.frame

Usage

```
artmsAnnotationUniprot(x, columnid, species, verbose = TRUE)
```

Arguments

`x`	(data.frame) to be annotated (or file path and name)
`columnid`	(char) The column with the uniprotkb ids
`species`	(char) The species name. Check `?artmsMapUniprot2Entrez` to find out more about supported species.
`verbose`	(logical) `TRUE` (default) shows function messages

Value

(data.frame) with two new columns: `Gene` and `Protein.name`

Examples

```
# This example adds annotations to the example evidence file included in
# artMS, based on the column 'Proteins'.

evidence_anno <- artmsAnnotationUniprot(x = artms_data_ph_evidence,
                                        columnid = 'Proteins',
                                        species = 'human')
```

`artmsAvgIntensityRT` *Summarize average intensity and retention time per protein*

Description

Input an evidence file from MaxQuant and a file containing a list of proteins of interest (optional). The function will summarize from the evidence file and report back the average intensity, average retention time, and the average caliberated retention time. If a list of proteins is provided, then only those proteins will be summarized and returned.

Usage

```
artmsAvgIntensityRT(
  evidence_file,
  protein_file = NULL,
  output_file = FALSE,
  verbose = TRUE
)
```

Arguments

`evidence_file`	(char) The filepath to the MaxQuant searched data (evidence) file (txt tab delimited file).
`protein_file`	(char) The file path to a file or vector containing a list of proteins of interest.
`output_file`	(char) The file name for the results (must have the extension `.txt`). If empty, then the results will be returned as an R object.
`verbose`	(logical) `TRUE` (default) shows function messages

Value

An R object with the results and a file with the results (if the output_file argument is provided). It contains averages of Intensity, Retention Time, Caliberated Retention Time

Examples

```
ave_int <- artmsAvgIntensityRT(evidence_file = artms_data_ph_evidence)
```

artmsChangeColumnName *Change a specific column name in a given data.frame*

Description

Making easier to change a column name in any data.frame

Usage

```
artmsChangeColumnName(dataset, oldname, newname)
```

Arguments

dataset	(data.frame) with the column name you want to change
oldname	(char) the old column name
newname	(char) the new name for that column

Value

(data.frame) with the new specified column name

Examples

```
artms_data_ph_evidence <- artmsChangeColumnName(
                                dataset = artms_data_ph_evidence,
                                oldname = "Phospho..STY.",
                                newname = "PH_STY")
```

artmsConvertMetabolomics *Convert Markview Metabolomics file (alignment table) into a artMS compatible format*

Description

artMS enables the relative quantification of untargeted polar metabolites using the alignment table generated by Markview. MarkerView is an ABSciex software that supports the files generated by Analyst software (.wiff) used to run our specific mass spectrometer (ABSciex Triple TOF 5600+). It also supports .t2d files generated by the Applied Biosystems 4700/4800 MALDI-TOF. MarkerView software is used to align mass spectrometry data from several samples for comparison. Using the import feature in the software, .wiff files (also .t2d MALDI-TOF files and tab-delimited .txt mass spectra data in mass-intensity format) are loaded for retention time alignment. Once the data files are selected, a series of windows will appear wherein peak finding, alignment, and filtering options can be entered and selected. These options include minimum spectral peak width, minimum

retention time peak width, retention time and mass tolerance, and the ability to filter out peaks that do not appear in more than a user selected number of samples.

'artmsConvertMetabolomics" processes the markview file to enable QC analysis and relative quantification using the artMS functions

Usage

```
artmsConvertMetabolomics(input_file, out_file, id_file = NULL, verbose = TRUE)
```

Arguments

input_file	(char) Markview input file
out_file	(char) Output file name
id_file	(char) KEGG database
verbose	(logical) TRUE (default) shows function messages

Value

(text file) Outputs the converted output name

Examples

```
# Testing that the arguments cannot be null
artmsConvertMetabolomics(input_file = NULL,
                         out_file = NULL)
```

---

artmsDataPlots *Individual Normalized abundance dot plots for every protein*

Description

Protein abundance dot plots for each unique uniprot id. It can take a long time

Usage

```
artmsDataPlots(input_file, output_file, verbose = TRUE)
```

Arguments

input_file	(char) File path and name to the -normalized.txt output file from MSstats
output_file	(char) Output file (path) name (add the .pdf extension)
verbose	(logical) TRUE (default) shows function messages

Value

(pdf) file with each individual protein abundance plot for each conditions

Examples

```
## Not run:
artmsDataPlots(input_file = "results/ab-results-mss-normalized.txt",
               output_file = "results/ab-results-mss-normalized.pdf")

## End(Not run)
```

`artmsEnrichLog2fc` *Enrichment of changes in protein abundance or PTMs*

Description

Enrichment analysis of the selected proteins

Usage

```
artmsEnrichLog2fc(
  dataset,
  species,
  background,
  heatmaps = FALSE,
  output_name = "enrichment.txt",
  verbose = TRUE
)
```

Arguments

`dataset`	(data.frame) with a `Gene` and `Comparison or Label` (with the name of the comparisons specified in the contrast file) columns
`species`	(char) Specie, only supported "human" or "mouse"
`background`	(vector) Background genes for the enrichment analysis.
`heatmaps`	(logical) if `TRUE` generates heatmaps (pdf), `FALSE` (default) otherwise.
`output_name`	(char) Name of the annotation files, which will be used as well for the heatmaps (if `heatmaps` is selected) Default `output_name = "enrichment.txt"`
`verbose`	(logical) `TRUE` (default) shows function messages

Value

(data.frame) Results from the enrichment analysis using Gprofiler and heatmaps (if selected)

Examples

```
## Not run:
# The data must be annotated (Protein and Gene columns)
data_annotated <- artmsAnnotationUniprot(
                      x = artms_data_ph_msstats_results,
                      columnid = "Protein",
                      species = "human")
# And then the enrichment
enrich_set <- artmsEnrichLog2fc(
                  dataset = data_annotated,
                  species = "human",
                  background = unique(data_annotated$Gene))

## End(Not run)
```

`artmsEnrichProfiler` *Enrichment analysis using g:Profiler (gprofiler2)*

Description

This function simplifies the enrichment analysis performed by the excellent tool g:Profiler, using the `gost()` function of the `gprofiler2` package (which replaces the deprecated `gProfileR` package). An internet connection is required.

Usage

```
artmsEnrichProfiler(
  x,
  categorySource = c("GO"),
  species,
  background = NA,
  verbose = TRUE
)
```

Arguments

`x`	(list, data.frame) List of protein ids. It can be anything: either a list of ids, or you could also send a data.frame and it will find the columns with the IDs. Is not cool? Multiple list can be also sent simultaneously, as for example running: `tmp <- split(enrichment$Gene, enrichment$cl_number, drop= TRUE)`
`categorySource`	(vector) Resources providing the terms on which the enrichment will be performed. The resources supported by g:Profiler are: • GO (GO:BP, GO:MF, GO:CC): Gene Ontology. `GO` selects the three of them • KEGG: Biological pathways • REAC: Biological pathways (Reactome) • WP: Biological pathways (WikiPathways)

- TF: Regulatory motifs in DNA (TRANSFAC TFBS)
- MIRNA: Regulatory motifs in DNA (miRTarBase microRNAs)
- CORUM: protein complexes database
- HP: Human Phenotype Ontology
- HPA: Protein databases (Human Protein Atlas) `OMIM` is no longer available in g:Profiler and it is ignored. Electronic GO annotations (IEA) are excluded.

`species` (char) Specie code: Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'. Check g:Profiler to find out more about supported species.

`background` (vector) gene list to use as background for the enrichment analysis. Default: `NA` (no custom background: all known genes are used)

`verbose` (logical) `TRUE` (default) shows function messages

Details

This function calls `gprofiler2::gost()` with these arguments:

- ordered_query = FALSE
- significant = TRUE
- exclude_iea = TRUE
- measure_underrepresentation = FALSE
- evcodes = TRUE (needed to report the `intersection` column)
- user_threshold = 0.05
- correction_method = "g_SCS" (named "analytical" in gProfileR)
- domain_scope = "known", or "custom" when a `background` is provided

The column names of the results are kept as in previous versions of artMS (based on `gProfileR`): `query.number`, `significant`, `p.value`, `term.size`, `query.size`, `overlap.size`, `recall`, `precision`, `term.id`, `domain`, `term.name`, `intersection`. Note that `domain` now uses the g:Profiler source names (e.g. `GO:BP`, `KEGG`, `REAC`).

Value

(data.frame) The significant enrichment results (zero rows if there are none)

Examples

```
## Not run:
# annotate the MSstats results to get the Gene name
data_annotated <- artmsAnnotationUniprot(
                                    x = artms_data_ph_msstats_results,
                                    columnid = "Protein",
                                    species = "human")

# Filter the list of genes with a log2fc > 2
filtered_data <-
```

```
unique(data_annotated$Gene[which(data_annotated$log2FC > 2)])

# And perform enrichment analysis
data_annotated_enrich <- artmsEnrichProfiler(
                                  x = filtered_data,
                                  categorySource = c('KEGG'),
                                  species = "hsapiens",
                                  background = unique(data_annotated$Gene))

## End(Not run)
```

`artmsEvidenceToSaintExpress` *MaxQuant evidence file to SAINTexpress format*

Description

Converts the MaxQuant evidence file to the 3 required files by SAINTexpress. One can choose to either use the `spectral counts` (use `msspc`) or the `intensities` (use `msint`) for the analysis.

Usage

```
artmsEvidenceToSaintExpress(
  evidence_file,
  keys_file,
  ref_proteome_file,
  quant_variable = c("msspc", "msint"),
  output_file,
  verbose = TRUE
)
```

Arguments

`evidence_file`	(char) The evidence file path and name
`keys_file`	(char) Keys file with a SAINT column specifying test (`T`) and control (`C`) conditions
`ref_proteome_file`	(char) Reference proteome path file name in fasta format
`quant_variable`	(char) choose either • `msspc` (spectral counts, default) or • `msint` (MS Intensity)
`output_file`	(char) Output file name (must have extension .txt)
`verbose`	(logical) `TRUE` (default) shows function messages

Value

The 3 required files by SAINTexpress:

- `interactions.txt`
- `preys.txt`
- `baits.txt`

Examples

```
# Testing that the files cannot be empty
artmsEvidenceToSaintExpress(evidence_file = NULL,
keys_file = NULL, ref_proteome_file = NULL)
```

`artmsEvidenceToSAINTq` *MaxQuant evidence file to SAINTq format*

Description

Converts the MaxQuant evidence file to the required files by SAINTq. The user can choose to use either peptides with `spectral counts` (use `msspc`) or the all the peptides (use `all`) for the analysis. The quantitative can be also chosen (either MS Intensity or Spectral Counts)

Usage

```
artmsEvidenceToSAINTq(
  evidence_file,
  keys_file,
  output_dir = "artms_saintq",
  sc_option = c("all", "msspc"),
  fractions = FALSE,
  quant_variable = c("msint", "msspc"),
  verbose = TRUE
)
```

Arguments

`evidence_file`	(char or data.frame) The evidence file path and name, or data.frame
`keys_file`	(char) Keys file with a SAINT column specifying test (`T`) and control (`C`) conditions
`output_dir`	(char) New directory to create and save files. Default is current directory (recommended to provide a new folder name).
`sc_option`	(char). Filter peptides with spectral counts only. Two options: • `msspc`: use only peptides with spectral_counts • `all` (default): all peptides detected (including the one resulting from the MaxQuant 'Match between run' algorithm)

`fractions`	(logical) `TRUE` for 2D proteomics (fractions). Default: `FALSE`
`quant_variable`	(char) Select the quantitative variable. Two options available: • `msint`: MS Intensity (default) • `msspc`: MS.MS.count (Spectral Counts)
`verbose`	(logical) `TRUE` (default) shows function messages

Details

After running the script, the new specified folder should contain the folling files:

- saintq-config-peptides
- saintq-config-proteins
- saintq_input_peptides.txt
- saintq_input_proteins.txt

Then `cd` into the new folder and run either of the following two options (assuming that `saintq` is installed in your linux/unix/mac os x system):

```
> saintq config-saintq-peptides
```

or

```
> saintq config-saintq-proteins
```

Value

The input files requires to run SAINTq

Examples

```
# Testing that the files cannot be empty
artmsEvidenceToSAINTq   (evidence_file = NULL,
                                keys_file = NULL,
                                output_dir = NULL)
```

---

artmsFilterEvidenceContaminants

Remove contaminants and empty proteins from the MaxQuant evidence file

---

Description

Remove contaminants and erronously identified 'reverse' sequences by MaxQuant, in addition to empty protein ids

Usage

```
artmsFilterEvidenceContaminants(x, verbose = TRUE)
```

Arguments

x	(data.frame) of the Evidence file
verbose	(logical) TRUE (default) shows function messages

Value

(data.frame) without REV__ and CON__ Protein ids

Examples

```
ef <- artmsFilterEvidenceContaminants(x = artms_data_ph_evidence)
```

---

artmsGeneratePhSiteExtended

Generate ph-site specific detailed file

Description

Generate extended detailed ph-site file, where every line is a ph site instead of a peptide. Therefore, if one peptide has multiple ph sites it will be breaking down in each of the sites. This file will help generate input files for tools as Phosfate or PHOTON

Usage

```
artmsGeneratePhSiteExtended(
  df,
  pathogen = "nopathogen",
  species,
  ptmType,
  output_name
)
```

Arguments

df	(data.frame) of log2fc and imputed values
pathogen	(char) Is there a pathogen in the dataset as well? Available pathogens are tb (Tuberculosis), lpn (Legionella). If it is not, then use nopathogen (default).
species	(char) Main organism (supported for now: human or mouse)
ptmType	(char) It must be a ptm-site quantification dataset. Either: yes: ptmsites (for site specific analysis), or ptmph (Jeff's script output evidence file).
output_name	(char) A output file name (extension .txt required)

Value

(data.frame) extended version of the ph-site

Examples

```
## Not run:
artmsGeneratePhSiteExtended(df = dfobject,
                            species = "mouse",
                            ptmType = "ptmsites",
                            output_name = log2fc_file)

## End(Not run)
```

`artmsIsEvidenceNewVersion` *Check if a given evidencee file was generated by a new version of MaxQuant (v>1)*

Description

MaxQuant introduced changes in the column names and number of columns for the evidence file in version 1 (we think). This function check whether the evidence comes from the latest version of MaxQuant.

Usage

```
artmsIsEvidenceNewVersion(evidence_file)
```

Arguments

`evidence_file`	the evidence file name

Value

(logical) `TRUE` if it is a newer version of MaxQuant, `FALSE` otherwise

Examples

```
artmsIsEvidenceNewVersion(evidence_file = artms_data_ph_evidence)
```

artmsIsSpeciesSupported *Check if a species is supported and available*

Description

Given a species name, it checkes whether is supported, and if supported, check whether the annotation package is installed.

Usage

```
artmsIsSpeciesSupported(species, verbose = TRUE)
```

Arguments

species	(char) The species name. Species currently supported as part of artMS: • HUMAN • MOUSE And the following species can be used as well, but the user needs to install the corresponding org.db package: • ANOPHELES (`install.packages(org.Ag.eg.db)`) • BOVINE (`install.packages(org.Bt.eg.db)`) • WORM (`install.packages(org.Ce.eg.db)`) • CANINE (`install.packages(org.Cf.eg.db)`) • FLY (`install.packages(org.Dm.eg.db)`) • ZEBRAFISH (`install.packages(org.Dr.eg.db)`) • CHICKEN (`install.packages(org.Gg.eg.db)`) • RHESUS (`install.packages(org.Mmu.eg.db)`) • CHIMP (`install.packages(org.Pt.eg.db)`) • RAT (`install.packages(org.Rn.eg.db)`) • YEAST (`install.packages(org.Sc.sgd.db)`) • PIG (`install.packages(org.Ss.eg.db)`) • XENOPUS (`install.packages(org.Xl.eg.db)`)
verbose	(logical) `TRUE` (default) shows function messages

Value

(string) Name of the package for the given species

Examples

```
# Should return TRUE
artmsIsSpeciesSupported(species = "HUMAN")
artmsIsSpeciesSupported(species = "CHIMP")
```

`artmsLeaveOnlyUniprotEntryID` *Leave only the Entry ID from a typical full Uniprot IDs in a given column*

Description

Downloading a Reference Uniprot fasta database includes several Uniprot IDs for every protein. If the regular expression available in Maxquant is not activated, the full id will be used in the Proteins, Lead Protein, and Leading Razor Protein columns. This script leaves only the Entry ID.

For example, values in a Protein column like this:
`sp|P12345|Entry_name;sp|P54321|Entry_name2`

will be replace by

'P12345;P54321"

Usage

```
artmsLeaveOnlyUniprotEntryID(x, columnid)
```

Arguments

`x`	(data.frame) that contains the `columnid`
`columnid`	(char) Column name with the full uniprot ids

Value

(data.frame) with only Entry IDs.

Examples

```
# Example of data frame with full uniprot ids and sequences
p <- c("sp|A6NIE6|RN3P2_HUMAN;sp|Q9NYV6|RRN3_HUMAN",
       "sp|A7E2V4|ZSWM8_HUMAN",
       "sp|A5A6H4|ROA1_PANTR;sp|P09651|ROA1_HUMAN;sp|Q32P51|RA1L2_HUMAN",
       "sp|A0FGR8|ESYT2_HUMAN")
s <- c("ALENDFFNSPPRK", "GWGSPGRPK", "SSGPYGGGGQYFAK", "VLVALASEELAK")
evidence <- data.frame(Proteins = p, Sequences = s, stringsAsFactors = FALSE)

# Replace the Proteins column with only Entry ids
evidence <- artmsLeaveOnlyUniprotEntryID(x = evidence, columnid = "Proteins")
```

artmsMapUniprot2Entrez

Map GENE SYMBOL, NAME, AND ENTREZID to a vector of Uniprot IDS

Description

Map GENE SYMBOL, NAME, AND ENTREZID to a vector of Uniprot IDS

Usage

```
artmsMapUniprot2Entrez(uniprotkb, species)
```

Arguments

uniprotkb	(vector) Vector of UniprotKB IDs
species	(char) The species name. Species currently supported as part of artMS: check ?artmsIsSpeciesSupported() to find out the list of supported species‘

Value

(data.frame) with ENTREZID and GENENAMES mapped on UniprotKB ids

Examples

```
# Load an example with human proteins
exampleID <- c("Q6P996", "B1N8M6")
artmsMapUniprot2Entrez(uniprotkb = exampleID,
                       species = "HUMAN")
```

artmsMergeEvidenceAndKeys

Merge evidence.txt (or summary.txt) with keys.txt files

Description

Merge the evidence and keys files on the given columns

Usage

```
artmsMergeEvidenceAndKeys(
  x,
  keys,
  by = c("RawFile"),
  isSummary = FALSE,
  verbose = TRUE
)
```

Arguments

x	(data.frame or char) The evidence data, either as data.frame or the file name (and path). It also works for the summary.txt file
keys	The keys data, either as a data.frame or file name (and path)
by	(vector) specifying the columns use to merge the evidence and keys. Default: by=c('RawFile')
isSummary	(logical) TRUE or FALSE (default)
verbose	(logical) TRUE (default) shows function messages

Value

(data.frame) with the evidence and keys merged

Examples

```
evidenceKeys <- artmsMergeEvidenceAndKeys(x = artms_data_ph_evidence,
                                          keys = artms_data_ph_keys)
```

---

artmsMsstatsSummary *Summarize the MSStats results and data quantification*

Description

Converts the MSStats results file to wide format (unique Protein ID and columns are the comparisons), as well as adds BioReplicate information about

- the Number of Unique Peptides,
- Spectral Counts
- Intensities for each protein. In cases where there are multiple values for a Protein-BioReplicate pair due to minute changes in sequence, the maximum value is taken for the pair. Any pairs without a value are assigned a value of NA.

Usage

```
artmsMsstatsSummary(
  evidence_file,
  prot_group_file,
  keys_file,
  results_file,
  return_df = FALSE,
  verbose = TRUE
)
```

Arguments

evidence_file	(char or data.frame) The filepath to the MaxQuant searched data (evidence) file (txt tab delimited file). Only works for the newer versions of the evidence file.
prot_group_file	(char) The filepath to the MaxQuant proteinGroups.txt file (txt tab delimited file) or data.frame
keys_file	(char) The filepath to the keys file used with MSStats (txt tab delimited file).
results_file	(char) The filepath to the MSStats results file in t he default long format (txt tab delimited file or data.frame).
return_df	(data.frame) Whether or not to return the results to the R environment upon completion. This is useful if this is being used in an R pipeline and you want to feed the results directly into the next stage of analysis via an R environment/terminal. Regardless, the results will be written to file. Default = FALSE
verbose	(logical) TRUE (default) shows function messages

Value

(data.frame or txt file) with the summary

Examples

```
# Testing warning if files are not submitted
test <- artmsMsstatsSummary(evidence_file = NULL,
                      prot_group_file = NULL,
                      keys_file = NULL,
                      results_file = NULL)
```

---

artmsPhosfateOutput *Generate Phosfate Input file*

Description

It takes as input the imputedL2fcExtended.txt results generated by the artmsAnalysisQuantifications() function and generates the Phosfate input file (or data.frame) Please, notice that the only species suported by Phosfate is humans.

Usage

```
artmsPhosfateOutput(inputFile, output_dir = ".", verbose = TRUE)
```

Arguments

inputFile	(char) the imputedL2fcExtended.txt file name and location
output_dir	(char) Name of the folder to output results (Default: current directory. Recommended: phosfate_input)
verbose	(logical) TRUE (default) to show function messages

Value

Multiple output files (inputs of phosfate)

Examples

```
## Not run:
artmsPhosfateOutput(inputFile)

## End(Not run)
```

`artmsPhotonOutput` *Generate PHOTON Input file*

Description

It takes as input the `imputedL2fcExtended.txt` results generated by the `artmsAnalysisQuantifications()` function and generates the PHOTON input file. Please, notice that the only species suported by PHOTON is humans.

Usage

```
artmsPhotonOutput(inputFile, output_dir = ".", verbose = TRUE)
```

Arguments

`inputFile`	(char) the `imputedL2fcExtended.txt` file name and location
`output_dir`	(char) Name of the folder to output results (Default: current. Recommended: "photon_input_files" or similar)
`verbose`	(logical) `TRUE` (default) to show function messages

Value

Multiple output files (inputs of phosfate)

Examples

```
## Not run:
artmsPhotonOutput(inputFile)

## End(Not run)
```

`artmsPlotHeatmapQuant` *Outputs a heatmap of the MSStats results created using the log2fold changes*

Description

Heatmap of the Relative Quantifications (MSStats results)

Usage

```
artmsPlotHeatmapQuant(
  input_file,
  output_file = "quantifications_heatmap.pdf",
  species,
  labels = "*",
  cluster_cols = FALSE,
  display = "log2FC",
  lfc_lower = -2,
  lfc_upper = 2,
  whatPvalue = "adj.pvalue",
  FDR = 0.05,
  verbose = TRUE
)
```

Arguments

`input_file`	(char) MSstats `results.txt` file and location (or data.frame of resuts)
`output_file`	(char) Output file name (pdf format) and location. Default:"quantifications_heatmap.pdf"
`species`	(char). Specie name to be able to add the Gene name. To find out more about the supported species check `?artmsMapUniprot2Entrez`
`labels`	(vector) of uniprot ids if only specific labes would like to be plotted. Default: all labels
`cluster_cols`	(boolean) `True` or `False` to cluster columns. Default: FALSE
`display`	Metric to be displayed. Options: • `log2fc` (default) • `adj.pvalue` • `pvalue`
`lfc_lower`	(int) Lower limit for the log2fc. Default: -2
`lfc_upper`	(int) Upper limit for the log2fc. Default: +2
`whatPvalue`	(char) `pvalue` or `adj.pvalue` (default)
`FDR`	(int) Upper limit false discovery rate (or pvalue). Default: 0.05
`verbose`	(logical) `TRUE` (default) shows function messages

Value

(pdf or ggplot2 object) heatmap of the MSStats results using the selected metric

Examples

```
# Unfortunately, the example does not contain any significant hits
# Use for illustration purposes
artmsPlotHeatmapQuant(input_file = artms_data_ph_msstats_results,
                      species = "human",
                      output_file = NULL,
                      whatPvalue = "pvalue",
                      lfc_lower = -1,
                      lfc_upper = 1)
```

---

artmsProtein2SiteConversion

Converts the Protein ID column of the evidence file selected by the user to mod-site-specific notation: `ProteinID` *to* `ProteinID_AAnumber` *notation*

Description

It enables the modified-peptide specific quantification by converting the Protein column of the evidence file selected by the user to an `ProteinID_AAnumber`notation. In this way, each of the modified peptides can be quantified independently across conditions.

!!

WARNING: we have detected a version of MaxQuant (>1.6.3.0) outputs a‘ "Modified sequence" column of the evidence file that has two important changes for the annotation of phosphorylation:

- Uses `p` instead of `(ph)`
- The modified residue (i.e. `STY`) is the residue on the right of the `p`, instead of the residue to the left of `(ph)`, as usual. We have introduced a modification to detect and address this issue, but we advice the user to double check both the new evidence file with the introduce new notation and the `-mapping.txt` file and check that there are no NA values for the notation of phophopeptides.

!!

Usage

```
artmsProtein2SiteConversion(
  evidence_file,
  ref_proteome_file,
  column_name = c("Leading razor protein", "Leading proteins", "Proteins"),
  output_file,
  mod_type,
  overwrite_evidence = FALSE,
  verbose = TRUE
)
```

Arguments

`evidence_file` (char) The evidence file name and location

`ref_proteome_file`
(char) The reference proteome used as database to search the `evidence.txt` file with MaxQuant. It will be used to map the modified peptide to the protein sequence and find the site location. Therefore, it does not use the MaxQuant's `Phospho (STY)Sites.txt`

`column_name` (char) The Protein Column Name to map. Options:

- `Leadind razor protein` (default)
- `Leading protein`
- `Proteins` It only supports Uniprot Entry IDs and RefSeq, but it might work for other database IDs

`output_file` (char) Output file name (`ptmsites-evidence.txt` recommended)

`mod_type` (char) The posttranslational modification. Options:

- `UB`: Protein Ubiquitination
- `PH`: Protein Phosphorylation
- `AC`: Protein Acetylation
- `PTM:XXX:yy` : User defined PTM. Replace XXX with 1 or more 1-letter amino acid codes on which to find modifications (all uppercase). Replace yy with modification name used within the evidence file (require lowercase characters). Example: `PTM:STY:ph` will find modifications on aa S,T,Y with this format `_AAGGAPS(ph)PPPPVR_`. This would be equivalent to `mod_type = PH`

`overwrite_evidence`
(logical) if <output_file> is the same as <evidence_file>, `overwrite_evidence = FALSE` (default) doesn't allow to overwrite the evidence file. Otherwise, `overwrite_evidence = TRUE` allows to overwrite the evidence_file (this option might be activated if the user allows to use the same `ptm-sites-evidence.txt` file to re-annotate all the Protein IDs columns)

`verbose` (logical) `TRUE` (default) shows function messages

Value

(file) Return a new evidence file with the specified Protein id column modified by adding the sequence site location(s) + postranslational modification(s) to the uniprot entry / refseq id.

Output ID examples: `A34890_ph3`; `Q64890_ph24_ph456`; `Q64890_ub34_ub129_ub234`; `Q64890_ac35`.

Examples

```
# Testing warning if files are not submitted.
artmsProtein2SiteConversion(evidence_file = NULL, ref_proteome_file = NULL,
output_file = NULL)
```

`artmsQualityControlEvidenceBasic` *Quality Control analysis of the MaxQuant evidence file*

Description

Quality Control analysis of the MaxQuant evidence file

Usage

```
artmsQualityControlEvidenceBasic(
  evidence_file,
  keys_file,
  prot_exp = c("AB", "PH", "UB", "AC", "APMS", "PTM:XXX:yy"),
  output_dir = "qc_basic",
  output_name = "qcBasic_evidence",
  isSILAC = FALSE,
  plotINTDIST = FALSE,
  plotREPRO = FALSE,
  plotCORMAT = TRUE,
  plotINTMISC = TRUE,
  plotPTMSTATS = TRUE,
  printPDF = TRUE,
  verbose = TRUE
)
```

Arguments

`evidence_file`	(char or data.frame) The evidence file path and name, or data.frame
`keys_file`	(char or data.frame) The keys file path and name or data.frame
`prot_exp`	(char) Proteomics experiment. 6 options available: • `APMS`: affinity purification mass spectrometry • `AB`: protein abundance • `PH`: protein phosphorylation • `UB`: protein ubiquitination (aka ubiquitylation) • `AC`: protein acetylation • `PTM:XXX:yy` : User defined PTM. Replace XXX with 1 or more 1-letter amino acid codes on which to find modifications (all uppercase). Replace yy with modification name used within the evidence file (require lowercase characters). Example for phosphorylation: `PTM:STY:ph` will find modifications on aa S,T,Y with this example format `_AAGGAPS(ph)PPPPVR_`. This means that the user could select phosphorylation as `PH` or `PTM:STY:ph`
`output_dir`	(char) Name for the folder to output the results plots. Default is "qc_basic".
`output_name`	(char) prefix output name (no extension). Default: "qcBasic_evidence"

isSILAC	if TRUE processes SILAC input files. Default is FALSE
plotINTDIST	if TRUE plots both *Box-dot plot* and *Jitter plot* of biological replicates based on MS (raw) intensity values, otherwise FALSE (default)
plotREPRO	if TRUE plots a correlation dotplot for all the combinations of biological replicates of conditions, based on MS Intensity values using features (peptide+charge). Otherwise FALSE (default)
plotCORMAT	if TRUE (default) plots a • *Correlation matrix* for all the biological replicates using MS Intensity values, • *Clustering matrix* of the MS Intensities
plotINTMISC	if TRUE (default) plots several pages, including bar plots of *Total Sum of Intensities in BioReplicates*, *Total Sum of Intensities in Conditions*, *Total Peptide Counts in BioReplicates*, *Total Peptide Counts in conditions* separated by categories: CON: contaminants, PROT peptides, REV reversed sequences used by MaxQuant to estimate the FDR; *Box plots* of MS Intensity values per biological replicates and conditions; *bar plots* of total intensity (excluding contaminants) by bioreplicates and conditions; Barplots of *total feature counts* by bioreplicates and conditions.
plotPTMSTATS	IF TRUE (default) plots stats related to the selected modification, including: *bar plot of peptide counts and intensities*, broken by PTM/other categories; bar plots of *total sum-up of MS intensity values* by other/PTM categories.
printPDF	If TRUE (default) prints out the pdfs. Warning: plot objects are not returned due to the large number of them.
verbose	(logical) TRUE (default) shows function messages

Value

Quality control files and plots

Examples

```
artmsQualityControlEvidenceBasic(evidence_file = artms_data_ph_evidence,
                                 keys_file = artms_data_ph_keys,
                                 prot_exp =  "PH",
                                 isSILAC = FALSE,
                                 plotINTDIST = FALSE,
                                 plotREPRO = TRUE,
                                 plotCORMAT = FALSE,
                                 plotINTMISC = FALSE,
                                 plotPTMSTATS = FALSE,
                                 printPDF = FALSE,
                                 verbose = FALSE)

# But we recommend the following test:
# 1. Go to a working directory:
# setwd("/path/to/your/working/directory/")
# 2. Run the following command to print out all the pdf files
# artmsQualityControlEvidenceBasic(evidence_file = artms_data_ph_evidence,
```

```
#                                 keys_file = artms_data_ph_keys,
#                                 prot_exp =  "PH")
# 3. Check your working directory and you should find pdf files with
# all the QC plots
```

`artmsQualityControlEvidenceExtended` *Extended Quality Control of the MaxQuant evidence.txt file*

Description

Performs quality control based on the information available in the MaxQuant `evidence.txt` file.

Usage

```
artmsQualityControlEvidenceExtended(
  evidence_file,
  keys_file,
  output_dir = "qc_extended",
  output_name = "qcExtended_evidence",
  isSILAC = FALSE,
  plotPSM = TRUE,
  plotIONS = TRUE,
  plotTYPE = TRUE,
  plotPEPTIDES = TRUE,
  plotPEPTOVERLAP = TRUE,
  plotPROTEINS = TRUE,
  plotPROTOVERLAP = TRUE,
  plotPIO = TRUE,
  plotCS = TRUE,
  plotME = TRUE,
  plotMOCD = TRUE,
  plotPEPICV = TRUE,
  plotPEPDETECT = TRUE,
  plotPROTICV = TRUE,
  plotPROTDETECT = TRUE,
  plotIDoverlap = TRUE,
  plotPCA = TRUE,
  plotSP = TRUE,
  printPDF = TRUE,
  verbose = TRUE
)
```

Arguments

`evidence_file`	(char or data.frame) The evidence file path and name, or data.frame
`keys_file`	(char or data.frame) The keys file path and name or data.frame

`output_dir` (char) Name for the folder to output the results plots. Default is "qc_extended".

`output_name` (char) prefix output name (no extension). Default: "qcExtended_evidence"

`isSILAC` if `TRUE` processes SILAC input files. Default is `FALSE`

`plotPSM` (logical) `TRUE` generates peptide-spectrum-matches (PSMs) statistics plot: Page 1 shows the number of PSMs confidently identified in each BioReplicate. If replicates are present, Page 2 shows the mean number of PSMs per condition with error bar showing the standard error of the mean. Note that potential contaminant proteins are plotted separately.

`plotIONS` (logical) `TRUE` generates peptide ion statistics plot: A peptide ion is defined in the context of m/z, in other words, an unique peptide sequence may give rise to multiple ions with different charge state and/or amino acid modification. Page 1 shows the number of ions confidently identified in each BioReplicate . If replicates are present, Page 2 shows the mean number of peptide ions per condition with error bar showing the standard error of the mean. Note that potential contaminant proteins are plotted separately.

`plotTYPE` (logical) `TRUE` generates identification type statistics plot: MaxQuant classifies each peptide identification into different categories (e.g., MSMS, MULTI-MSMS, MULTI-SECPEP). Page 1 shows the distribution of identification type in each BioReplicate

`plotPEPTIDES` (logical) `TRUE` generates peptide statistics plot: Page 1 shows the number of unique peptide sequences (disregard the charge state or amino acid modifications) confidently identified in each BioReplicate. If replicates are present, Page 2 shows the mean number of peptides per condition with error bar showing the standard error of the mean. Note that potential contaminant proteins are plotted separately. Pages 3 and 4 show peptide identification intersection between BioReplicates (the bars are ordered by degree or frequency, respectively), and Page 4 shows the intersections across conditions instead of BioReplicates.

`plotPEPTOVERLAP` (logical) `TRUE` Show peptide identification intersection between BioReplicates and Conditions

`plotPROTEINS` (logical) `TRUE` generates protein statistics plot: Page 1 shows the number of protein groups confidently identified in each BioReplicate. If replicates are present, Page 2 shows the mean number of protein groups per condition with error bar showing the standard error of the mean. Note that potential contaminant proteins are plotted separately. Pages 3 and 4 show peptide identification intersection between BioReplicates (the bars are ordered by degree or frequency, respectively), and Page 4 shows the intersections across conditions instead of BioReplicates.

`plotPROTOVERLAP` (logical) `TRUE` Show protein identification intersection between BioReplicates and Conditions

`plotPIO` (logical) `TRUE` generates oversampling statistics plot: Page 1 shows the proportion of all peptide ions (including peptides matched across runs) fragmented once, twice and thrice or more. Page 2 shows the proportion of peptide ions (with intensity detected) fragmented once, twice and thrice or more. Page 3 shows the proportion of peptide ions (with intensity detected and MS/MS identification) fragmented once, twice and thrice or more

plotCS	(logical) TRUE generates charge state plot: Page 1 shows the charge state distribution of PSMs confidently identified in each BioReplicate.
plotME	(logical) TRUE generates precursor mass error plot: Page 1 shows the distribution of precursor error for all PSMs confidently identified in each BioReplicate.
plotMOCD	(logical) TRUE generates precursor mass-over-charge plot: Page 1 shows the distribution of precursor mass-over-charge for all PSMs confidently identified in each BioReplicate.
plotPEPICV	(logical) TRUE generates peptide intensity coefficient of variance (CV) plot: The CV is calculated for each feature (peptide ion) identified in more than one replicate. Page 1 shows the distribution of CV's for each condition, while Page 2 shows the distribution of CV's within 4 bins of intensity (i.e., 4 quantiles of average intensity).
plotPEPDETECT	(logical) TRUE generates peptide detection frequency plot: Page 1 summarizes the frequency that each peptide is detected across BioReplicates of each condition, showing the percentage of peptides detected once, twice, thrice, and so on (for whatever number of replicates each condition has).
plotPROTICV	(logical) TRUE generates protein intensity coefficient of variance (CV) plot: The CV is calculated for each protein (after summing the peptide intensities) identified in more than one replicate. Page 1 shows the distribution of CV's for each condition, while Page 2 shows the distribution of CV's within 4 bins of intensity (i.e., 4 quantiles of average intensity).
plotPROTDETECT	(logical) TRUE generates protein detection frequency plot: Page 1 summarizes the frequency that each protein group is detected across BioReplicates of each condition, showing the percentage of proteins detected once, twice, thrice, and so on (for whatever number of replicates each condition has). Page 2 shows the feature (peptide ion) intensity distribution within each BioReplicate (potential contaminant proteins are plot separately). Page 3 shows the density of feature intensity for different feature types (i.e., MULTI-MSMS, MULTI-SECPEP).
plotIDoverlap	(logical) TRUE generates pairwise identification heatmap overlap: Pages 1 and 2 show pairwise peptide and protein overlap between any 2 BioReplicates, respectively.
plotPCA	(logical) TRUE generates PCA and pairwise intensity correlation: Page 1 and 3 show pairwise peptide and protein intensity correlation and scatter plot between any 2 BioReplicates, respectively. Page 2 and 4 show Principal Component Analysis at the intensity level for both peptide and proteins, respectively.
plotSP	(logical) TRUE generates sample quality metrics: Page 1 shows missing cleavage distribution of all peptides confidently identified in each BioReplicate. Page 2 shows the fraction of peptides with at least one methionine oxidized in each BioReplicate.
printPDF	If TRUE (default) prints out the pdfs. Warning: plot objects are not returned due to the large number of them.
verbose	(logical) TRUE (default) shows function messages

Details

all the plots are generated by default

Value

A number of QC plots based on the evidence file

Examples

```
# Testing warning if files are not submitted
test <- artmsQualityControlEvidenceExtended(evidence_file = NULL,
keys_file = NULL)
```

`artmsQualityControlMetabolomics` *Quality Control analysis of the evidence-like metabolomics dataset*

Description

Quality Control analysis of the evidence-like metabolomics dataset

Usage

```
artmsQualityControlMetabolomics(
  evidence_file,
  keys_file,
  met_exp = c("MV"),
  output_name = "qcPlots_metab",
  plotINTDIST = FALSE,
  plotCORMAT = TRUE,
  plotINTMISC = TRUE,
  printPDF = TRUE,
  verbose = TRUE
)
```

Arguments

`evidence_file`	(char or data.frame) The evidence file path and name, or data.frame
`keys_file`	(char or data.frame) The keys file path and name or data.frame
`met_exp`	(char) Metabolomics experiment. Only one option available (so far): • `MV`: Markview output
`output_name`	(char) prefix output name (no extension). Default: "qcPlots_metab"
`plotINTDIST`	if `TRUE` (default) plots both *Box-dot plot* and *Jitter plot* of biological replicates based on MS (raw) intensity values.
`plotCORMAT`	if `TRUE` (default) generates up to 3 pdf files for technical replicates, biological replicates, and conditions. Each pdf file contains: • *Correlation matrix* for all the biological replicates using MS Intensity values, • *Clustering matrix* of the MS Intensities and correlation distribution

- *histogram* of the distribution of correlations

plotINTMISC	if TRUE (default) plots several pages, including bar plots of *Total Sum of Intensities in BioReplicates*, *Total Sum of Intensities in Conditions*, *Total Feature Counts in BioReplicates*, *Total Feature Counts in conditions* separated by categories (INT: has a intensity value NOINT: no intensity value) *Box plots* of MS Intensity values per biological replicates and conditions; *bar plots* of total intensity by bioreplicates and conditions; Barplots of *total feature counts* by bioreplicates and conditions.
printPDF	If TRUE (default) prints out the pdfs. Warning: plot objects are not returned due to the large number of them.
verbose	(logical) TRUE (default) shows function messages

Value

Quality control files and plots for metabolomics

Examples

```
# Testing that input arguments cannot be null
artmsQualityControlMetabolomics(evidence_file = NULL,
                 keys_file = NULL,
                 met_exp = "MV")
```

`artmsQualityControlSummaryExtended`
Quality Control of the MaxQuant summary.txt file

Description

Performs quality control based on the information available in the MaxQuant summary.txt file.

Usage

```
artmsQualityControlSummaryExtended(
  summary_file,
  keys_file,
  output_dir = "qc_summary",
  output_name = "qcExtended_summary",
  isFractions = FALSE,
  plotMS1SCANS = TRUE,
  plotMS2 = TRUE,
  plotMSMS = TRUE,
  plotISOTOPE = TRUE,
  printPDF = TRUE,
  verbose = TRUE
)
```

Arguments

summary_file	(char or data.frame) The evidence file path and name, or data.frame
keys_file	(char or data.frame) The keys file path and name or data.frame
output_dir	(char) Name for the folder to output the results plots. Default is "qc_summary".
output_name	(char) prefix output name (no extension). Default: "qcExtended_summary"
isFractions	(logical) TRUE if it is a 2D experiment (fractions). Default: FALSE
plotMS1SCANS	(logical) TRUE generates MS1 scan counts plot: Page 1 shows the number of MS1 scans in each BioReplicate. If replicates are present, Page 2 shows the mean number of MS1 scans per condition with error bar showing the standard error of the mean. If isFractions TRUE, each fraction is a stack on the individual bar graphs.
plotMS2	(logical) TRUE generates MS2 scan counts plot: Page 1 shows the number of MSs scans in each BioReplicate. If replicates are present, Page 2 shows the mean number of MS1 scans per condition with error bar showing the standard error of the mean. If isFractions TRUE, each fraction is a stack on the individual bar graphs.
plotMSMS	(logical) TRUE generates MS2 identification rate (%) plot: Page 1 shows the fraction of MS2 scans confidently identified in each BioReplicate. If replicates are present, Page 2 shows the mean rate of MS2 scans confidently identified per condition with error bar showing the standard error of the mean. If isFractions TRUE, each fraction is a stack on the individual bar graphs.
plotISOTOPE	(logical) TRUE generates Isotope Pattern counts plot: Page 1 shows the number of Isotope Patterns with charge greater than 1 in each BioReplicate. If replicates are present, Page 2 shows the mean number of Isotope Patterns with charge greater than 1 per condition with error bar showing the standard error of the mean. If isFractions TRUE, each fraction is a stack on the individual bar graphs.
printPDF	If TRUE (default) prints out the pdfs. Warning: plot objects are not returned due to the large number of them.
verbose	(logical) TRUE (default) shows function messages

Value

A number of plots from the summary file

Examples

```
# Testing warning if files are not submitted
test <- artmsQualityControlSummaryExtended(summary_file = NULL,
keys_file = NULL)
```

`artmsQuantification` *Relative quantification using MSstats*

Description

Relative quantification using MSstats including:

- plots
- quantifications (log2fc, pvalues, etc)
- normalized abundance values

Usage

```
artmsQuantification(
  yaml_config_file,
  data_object = FALSE,
  printPDF = TRUE,
  printTables = TRUE,
  display_msstats = FALSE,
  return_results_object = FALSE,
  verbose = TRUE
)
```

Arguments

`yaml_config_file`
: (char, required) The yaml file name and location

`data_object`
: (logical) flag to indicate whether the configuration file is a string to a file that should be opened or config object (yaml). Default is `FALSE`. Choose `TRUE` if `yaml_config_file` is a yaml object

`printPDF`
: (logical) if `TRUE` (default), prints out pdf

`printTables`
: (logical) `TRUE` (default) print results tables

`display_msstats`
: (logical) if `TRUE`, prints MSstats outputs (default is `FALSE`)

`return_results_object`
: (logical) Default is `FALSE`. If `TRUE`, it returns a list of data frames with MSstats results, including:

  - `comparisonResult`: comparison results
  - `ModelQC`
  - `FittedModel`: fit model details
  - `power`: power calculations
  - `sample_size`: sample size estimations

`verbose`
: (logical) `TRUE` (default) shows function messages

Value

The relative quantification of the conditions and comparisons specified in the keys/contrast file resulting from running MSstats, in addition to quality control plots (if selected)

Examples

```
# Recommended
# artmsQuantification(yaml_config_file = "your-config-file.yaml")

# Example to test this function using the example dataset available in artMS
# Step 1: Add evidence, keys, and contrast to configuration object
artms_data_ph_config$files$evidence <- artms_data_ph_evidence
artms_data_ph_config$files$keys <- artms_data_ph_keys
artms_data_ph_config$files$contrasts <- artms_data_ph_contrast

# Step 2: Run the quantification step
quant_results <- artmsQuantification(yaml_config_file = artms_data_ph_config,
                                     data_object = TRUE,
                                     display_msstats = FALSE,
                                     printPDF = FALSE,
                                     printTables = FALSE)
# Check the list of data frames "quant_results". Nothing should be printed out.
```

---

artmsResultsWide *Reshape the MSstats results file from long to wide format*

Description

Converts the normal MSStats results.txt file into "wide" format where each row represents a unique protein's results, and each column represents the comparison made by MSStats. The fold change and p-value of each comparison will be its own column.

Usage

```
artmsResultsWide(
  results_msstats,
  output_file = NULL,
  select_pvalues = c("adjpvalue", "pvalue"),
  species,
  verbose = TRUE
)
```

Arguments

`results_msstats`
: (char) Input file name and location (MSstats `results.txt` file)

`output_file`
: (char) Output file name and location (e.g. `results-wide.txt`). If `NULL` (default) returns an R object (data.frame)

select_pvalues	(char) Either • pvalue or • adjpvalue (default)
species	(char) Specie name for annotation purposes. Check ?artmsMapUniprot2Entrez to find out more about the supported species (e.g species = "human")
verbose	(logical) TRUE (default) shows function messages

Value

(output file tab delimited) reshaped file with unique protein ids and as many columns log2fc and adj.pvalues as comparisons available

Examples

```
ph_results_wide <- artmsResultsWide(
                        results_msstats = artms_data_ph_msstats_results,
                        output_file = NULL,
                        species = "human")
```

---

artmsSILACtoLong *Convert the SILAC evidence file to MSstats format*

Description

Converting the evidence file from a SILAC search to a format compatible with MSstats. It basically modifies the Raw.files adding the Heavy and Light label

Usage

```
artmsSILACtoLong(evidence_file, output = NULL, verbose = TRUE)
```

Arguments

evidence_file	(char) Text filepath to the evidence file
output	(char) Text filepath of the output name. If NULL it does not write the output
verbose	(logical) TRUE (default) shows function messages

Value

(data.frame) with SILAC data processed for MSstats (and output file)

Examples

```
## Not run:
evidence2silac <- artmsSILACtoLong(evidence_file = "silac.evicence.txt",
                                   output = "silac-evidence.txt")

## End(Not run)
```

artmsSpectralCounts *Outputs the spectral counts from the MaxQuant evidence file.*

Description

Outputs the spectral counts from the MaxQuant evidence file.

Usage

```
artmsSpectralCounts(
  evidence_file,
  keys_file,
  output_file = NULL,
  verbose = TRUE
)
```

Arguments

evidence_file	(char) Maxquant evidence file or data object
keys_file	(char) Keys file with the experimental design or data object
output_file	(char) Output file name (add .txt extension). If NULL (default) it returns a data.frame object
verbose	(logical) TRUE (default) shows function messages

Value

A txt file with biological replicates, protein id, and spectral count columns

Examples

```
summary_spectral_counts <- artmsSpectralCounts(
                                evidence_file = artms_data_ph_evidence,
                                keys_file = artms_data_ph_keys)
```

artmsVolcanoPlot *Volcano plot (log2fc / pvalues)*

Description

It generates a scatter-plot used to quickly identify changes

Usage

```
artmsVolcanoPlot(
  mss_results,
  output_name = "volcano_plot.pdf",
  lfc_upper = 1,
  lfc_lower = -1,
  whatPvalue = "adj.pvalue",
  FDR = 0.05,
  PDF = TRUE,
  decimal_threshold = 16,
  verbose = TRUE
)
```

Arguments

mss_results	(data.frame or file) Selected MSstats results
output_name	(char) Name for the output file (don't forget the .pdf extension)
lfc_upper	(numeric) log2fc upper threshold (positive value)
lfc_lower	(numeric) log2fc lower threshold (negative value)
whatPvalue	(char) pvalue or adj.pvalue (default)
FDR	(numeric) False Discovery Rate threshold
PDF	(logical) Option to generate pdf format. Default: T
decimal_threshold	(numeric) Decimal threshold for the pvalue. Default: 16 (10^-16)
verbose	(logical) TRUE (default) shows function messages

Value

(pdf) of a volcano plot

Examples

```
artmsVolcanoPlot(mss_results = artms_data_ph_msstats_results,
                 whatPvalue = "pvalue",
                 PDF = FALSE)
```

---

artmsWriteConfigYamlFile

Write out a template file of the artMS configuration file (yaml)

Description

Creates a template file of the artMS configuration file, which is required to run artmsQuantification. Check ?artms_config and the vignettes to find out more about the details of the structure of the file and how to fill it up

Usage

```
artmsWriteConfigYamlFile(
  config_file_name = "artms_config_file.yaml",
  overwrite = FALSE,
  verbose = TRUE
)
```

Arguments

config_file_name	(char) The name for the configuration file. It must have a `.yaml` extension. If `NULL`, it returns the config as a yaml object
overwrite	(logical) Default FALSE
verbose	(logical) `TRUE` (default) shows function messages

Value

A file (or yaml data object) of the artMS configuration file

Examples

```
config_empty <- artmsWriteConfigYamlFile(config_file_name = NULL)
```

Index

∗ **APMS**
artmsEvidenceToSaintExpress, 21
artmsEvidenceToSAINTq, 22
∗ **MSStats,**
artmsMsstatsSummary, 29
∗ **MaxQuant,**
artmsAvgIntensityRT, 15
artmsMsstatsSummary, 29
∗ **QC,**
artmsQualityControlEvidenceBasic, 35
artmsQualityControlMetabolomics, 40
∗ **SAINT,**
artmsEvidenceToSaintExpress, 21
artmsEvidenceToSAINTq, 22
∗ **SAINTexpress,**
artmsEvidenceToSaintExpress, 21
∗ **SAINTq,**
artmsEvidenceToSAINTq, 22
∗ **abundance,**
artmsDataPlots, 17
∗ **ac**
artmsProtein2SiteConversion, 33
∗ **analysis,**
artmsAnalysisQuantifications, 10
∗ **annotation,**
artmsAnnotateSpecie, 13
artmsAnnotationUniprot, 14
artmsIsSpeciesSupported, 26
artmsLeaveOnlyUniprotEntryID, 27
artmsMapUniprot2Entrez, 28
∗ **caliberated**
artmsAvgIntensityRT, 15
∗ **check**
artmsIsEvidenceNewVersion, 25
∗ **cleanup,**
artmsFilterEvidenceContaminants, 23
∗ **columns**
artmsChangeColumnName, 16
∗ **config,**
artmsWriteConfigYamlFile, 47
∗ **contaminants**
artmsFilterEvidenceContaminants, 23
∗ **control,**
artmsQualityControlEvidenceBasic, 35
artmsQualityControlMetabolomics, 40
∗ **convert,**
artmsProtein2SiteConversion, 33
artmsSILACtoLong, 45
∗ **convert**
artmsConvertMetabolomics, 16
∗ **data.frame,**
artmsChangeColumnName, 16
∗ **datasets**
artms_config, 3
artms_data_corum_mito_database, 4
artms_data_pathogen_LPN, 5
artms_data_pathogen_TB, 6
artms_data_ph_config, 6
artms_data_ph_contrast, 7
artms_data_ph_evidence, 8
artms_data_ph_keys, 8
artms_data_ph_msstats_modelqc, 9
artms_data_ph_msstats_results, 9
artms_data_randomDF, 10
∗ **dotplots,**
artmsDataPlots, 17
∗ **driver,**
artmsQuantification, 43
∗ **enrichment**
artmsEnrichLog2fc, 18
artmsEnrichProfiler, 19
∗ **evidence,**

artmsAvgIntensityRT, 15
artmsIsEvidenceNewVersion, 25
artmsMergeEvidenceAndKeys, 28
artmsMsstatsSummary, 29
artmsProtein2SiteConversion, 33
artmsQualityControlEvidenceExtended, 37
artmsQualityControlMetabolomics, 40
* **evidence**
artmsQualityControlEvidenceBasic, 35
artmsSILACtoLong, 45
artmsSpectralCounts, 46
* **external,**
artmsGeneratePhSiteExtended, 24
* **file,**
artmsIsEvidenceNewVersion, 25
* **files**
artmsPhosfateOutput, 30
artmsPhotonOutput, 31
* **function**
artmsQuantification, 43
* **generate,**
artmsPhosfateOutput, 30
artmsPhotonOutput, 31
* **heatmap,**
artmsPlotHeatmapQuant, 32
* **ids**
artmsLeaveOnlyUniprotEntryID, 27
artmsMapUniprot2Entrez, 28
* **input,**
artmsIsEvidenceNewVersion, 25
* **intensity,**
artmsAvgIntensityRT, 15
* **keys**
artmsMergeEvidenceAndKeys, 28
artmsQualityControlEvidenceExtended, 37
artmsQualityControlSummaryExtended, 41
* **log2fc**
artmsPlotHeatmapQuant, 32
* **main,**
artmsQuantification, 43
* **merge,**
artmsMergeEvidenceAndKeys, 28
* **metabolomics,**
artmsConvertMetabolomics, 16
* **metabolomics**
artmsQualityControlMetabolomics, 40
* **msstats,**
artmsResultsWide, 44
* **outputs,**
artmsPhosfateOutput, 30
artmsPhotonOutput, 31
* **ph,**
artmsProtein2SiteConversion, 33
* **phosfate**
artmsGeneratePhSiteExtended, 24
* **plot,**
artmsVolcanoPlot, 46
* **plot**
artmsDataPlots, 17
* **ptm,**
artmsProtein2SiteConversion, 33
* **qc,**
artmsQualityControlEvidenceExtended, 37
artmsQualityControlSummaryExtended, 41
* **quality,**
artmsQualityControlEvidenceBasic, 35
artmsQualityControlMetabolomics, 40
* **quantifications**
artmsAnalysisQuantifications, 10
* **rename,**
artmsChangeColumnName, 16
* **reshape**
artmsResultsWide, 44
* **results,**
artmsResultsWide, 44
* **retention**
artmsAvgIntensityRT, 15
* **silac,**
artmsSILACtoLong, 45
* **species**
artmsAnnotateSpecie, 13
artmsIsSpeciesSupported, 26
* **spectral_counts,**
artmsSpectralCounts, 46
* **summary,**
artmsAvgIntensityRT, 15

artmsMergeEvidenceAndKeys, 28
artmsQualityControlSummaryExtended, 41
* **summary**
artmsMsstatsSummary, 29
* **time,**
artmsAvgIntensityRT, 15
* **tools,**
artmsGeneratePhSiteExtended, 24
* **ub,**
artmsProtein2SiteConversion, 33
* **uniprot**
artmsAnnotationUniprot, 14
* **version**
artmsIsEvidenceNewVersion, 25
* **volcano**
artmsVolcanoPlot, 46
* **wide,**
artmsResultsWide, 44
* **yaml**
artmsWriteConfigYamlFile, 47

artms_config, 3
artms_data_corum_mito_database, 4
artms_data_pathogen_LPN, 5
artms_data_pathogen_TB, 6
artms_data_ph_config, 6
artms_data_ph_contrast, 7
artms_data_ph_evidence, 8
artms_data_ph_keys, 8
artms_data_ph_msstats_modelqc, 9
artms_data_ph_msstats_results, 9
artms_data_randomDF, 10
artmsAnalysisQuantifications, 10
artmsAnnotateSpecie, 13
artmsAnnotationUniprot, 14
artmsAvgIntensityRT, 15
artmsChangeColumnName, 16
artmsConvertMetabolomics, 16
artmsDataPlots, 17
artmsEnrichLog2fc, 18
artmsEnrichProfiler, 19
artmsEvidenceToSaintExpress, 21
artmsEvidenceToSAINTq, 22
artmsFilterEvidenceContaminants, 23
artmsGeneratePhSiteExtended, 24
artmsIsEvidenceNewVersion, 25
artmsIsSpeciesSupported, 26
artmsLeaveOnlyUniprotEntryID, 27
artmsMapUniprot2Entrez, 28
artmsMergeEvidenceAndKeys, 28
artmsMsstatsSummary, 29
artmsPhosfateOutput, 30
artmsPhotonOutput, 31
artmsPlotHeatmapQuant, 32
artmsProtein2SiteConversion, 33
artmsQualityControlEvidenceBasic, 35
artmsQualityControlEvidenceExtended, 37
artmsQualityControlMetabolomics, 40
artmsQualityControlSummaryExtended, 41
artmsQuantification, 43
artmsResultsWide, 44
artmsSILACtoLong, 45
artmsSpectralCounts, 46
artmsVolcanoPlot, 46
artmsWriteConfigYamlFile, 47