Package 'MsBackendMetaboLights'

February 2, 2026

Title Retrieve Mass Spectrometry Data from MetaboLights

Version 1.5.1

Description MetaboLights is one of the main public repositories for storage of metabolomics experiments, which includes analysis results as well as raw data. The MsBackendMetaboLights package provides functionality to retrieve and represent mass spectrometry (MS) data from MetaboLights. Data files are downloaded and cached locally avoiding repetitive downloads. MS data from metabolomics experiments can thus be directly and seamlessly integrated into R-based analysis workflows with the Spectra and MsBackendMetaboLights package.

Depends R (>= 4.2.0), Spectra (>= 1.15.12)

Imports curl, ProtGenerics, BiocFileCache, S4Vectors, methods, progress

Suggests testthat, rmarkdown, mzR, knitr, BiocStyle

License Artistic-2.0

Encoding UTF-8

VignetteBuilder knitr

BugReports https://github.com/RforMassSpectrometry/MsBackendMetaboLights/issues

URL https://github.com/RforMassSpectrometry/MsBackendMetaboLights

biocViews Infrastructure, MassSpectrometry, Metabolomics, DataImport, Proteomics

Roxygen list(markdown=TRUE)

RoxygenNote 7.3.3

git_url https://git.bioconductor.org/packages/MsBackendMetaboLights

git_branch devel

git_last_commit d9f698a

git_last_commit_date 2025-11-04

Repository Bioconductor 3.23

Date/Publication 2026-02-01

Author Johannes Rainer [aut, cre] (ORCID:
<https://orcid.org/0000-0002-6977-7147>),
Philippine Louail [aut] (ORCID:
<https://orcid.org/0009-0007-5429-6846>)

Maintainer Johannes Rainer <Johannes.Rainer@eurac.edu>

Contents

MetaboLights-utils 2
MsBackendMetaboLights 4
retry 7

Index **9**

`MetaboLights-utils` *Utility functions for the MetaboLights repository*

Description

MetaboLights is one of the main public repositories for deposition of metabolomics experiments including (raw) mass spectrometry (MS) and NMR data files and experimental/analysis results. The experimental metadata and results are stored as plain text files in ISA-tab format. Each MetaboLights experiment must provide a file describing the samples analyzed and at least one *assay* file that links between the experimental samples and the (raw and processed) data files with quantification of metabolites/features in these samples.

Each experiment in MetaboLights is identified with its unique identifier, starting with *MTBLS* followed by a number. The data (metadata files and MS/NMR data files) of an experiment are available through the repository's ftp server.

The functions listed here allow to query and retrieve information of a data set/experiment from MetaboLights.

- `mtbls_ftp_path()`: returns the FTP path for a provided MetaboLights ID. With `mustWork = TRUE` (the default) the function throws an error if the path is not accessible (either because the data set does not exist or no internet connection is available). The function returns a `character(1)` with the FTP path to the data set folder.
- `mtbls_cached_data_files()`: lists locally cached data files from MetaboLights. Since this function evaluates only local content it does not require an internet connection. With the default parameters all available data files are listed. The parameters can be used to restrict the lookup.
- `mtbls_list_files()`: returns the available files (and directories) for the specified MetaboLights data set (i.e., the FTP directory content of the data set). The function returns a `character` vector with the relative file names to the absolute FTP path (`mtbls_ftp_path()`) of the data set. Parameter `pattern` allows to filter the file names and define which file names should be returned.

- `mtbls_sync_data_files()`: synchronize data files of a specifies MetaboLights data set eventually downloading and locally caching them. Parameter `fileName` allows to specify names of selected data files to sync.
- `mtbls_delete_cache()`: removes all local content for the MetaboLights data set with ID `mtblsId`. This will delete eventually present locally cached data files for the specified data set. This does not change any other data eventually present in the local `BiocFileCache`.

Usage

```
mtbls_ftp_path(x = character(), mustWork = TRUE)

mtbls_list_files(x = character(), pattern = NULL)

mtbls_sync_data_files(
  mtblsId = character(),
  assayName = character(),
  pattern = "mzML$|CDF$|cdf$|mzXML$",
  fileName = character()
)

mtbls_cached_data_files(
  mtblsId = character(),
  assayName = character(),
  pattern = "*",
  fileName = character()
)

mtbls_delete_cache(mtblsId = character())
```

Arguments

`x`	`character(1)` with the ID of the MetaboLights data set (usually starting with a *MTBLS* followed by a number).
`mustWork`	for `mtbls_ftp_path()`: `logical(1)` whether the validity of the path should be verified or not. By default (with `mustWork = TRUE`) the function throws an error if either the data set does not exist or if the folder can not be accessed (e.g. if no internet connection is available).
`pattern`	for `mtbls_list_files()`, `mtbls_sync_data_files()` and `mtbls_cached_data_files()`: `character(1)` defining a pattern to filter the file names, such as `pattern = "^a_"` to retrieve the file names of all assay files of the data set (i.e., files with a name starting with `"a_"`). This parameter is passed to the `grepl()` function.
`mtblsId`	`character(1)` with the ID of a single MetaboLights data set/experiment.
`assayName`	`character` with the file names of assay files of the data set. If not provided (`assayName = character()`, the default), MS data files of all data set's assays are loaded. Use `mtbls_list_files(<MetaboLights ID>, pattern = "^a_")` to list all available assay files of a data set `<MetaboLights ID>`.

fileName — for mtbls_sync_data_files() and mtbls_cached_data_files(): optional character defining the names of specific data files of a data set that should be downloaded and cached.

Value

- For mtbls_ftp_path(): character(1) with the ftp path to the specified data set on the MetaboLights ftp server.
- For mtbls_list_files(): character with the names of the files in the data set's base ftp directory.
- For mtbls_sync_data_files() and mtbls_cached_data_files(): a data.frame with the MetaboLights ID, the assay name(s) and remote and local file names of the synchronized data files.

Author(s)

Johannes Rainer, Philippine Louail

Examples

```
## Get the FTP path to the data set MTBLS2
mtbls_ftp_path("MTBLS2")

## Retrieve available files (and directories) for the data set MTBLS2
mtbls_list_files("MTBLS2")

## Retrieve the available assay files (file names starting with "a_").
afiles <- mtbls_list_files("MTBLS2", pattern = "^a_")
afiles

## Read the content of one file. Connections to the MetaboLights ftp server
## are limited and might fail, thus we use the `retry()` function to
## retry on failure for 5 times (waiting `i * sleep_mult` seconds in between)
a <- retry(
    read.table(paste0(mtbls_ftp_path("MTBLS2"), afiles[1L]),
    header = TRUE, sep = "\t", check.names = FALSE),
    ntimes = 5, sleep_mult = 4)
head(a)

## List all available files
mtbls_cached_data_files()
```

MsBackendMetaboLights *MsBackend representing MS data from MetaboLights*

Description

MsBackendMetaboLights retrieves and represents mass spectrometry (MS) data from metabolomics experiments stored in the MetaboLights repository. The backend directly extends the Spectra::MsBackendMzR backend from the *Spectra* package and hence supports MS data in mzML, netCDF and mzXML format. Data in other formats can not be loaded with MsBackendMetaboLights. Upon initialization with the backendInitialize() method, the MsBackendMetaboLights backend downloads and caches the MS data files of an experiment locally avoiding hence repeated download of the data. The local data cache is managed by Bioconductor's *BiocFileCache* package. See the help and vignettes from that package for details on cached data resources. Additional utility function for management of cached files are also provided by *MsBackendMetaboLights*. See help for mtbls_cached_data_files() for more information.

Usage

```
MsBackendMetaboLights()

## S4 method for signature 'MsBackendMetaboLights'
backendInitialize(
  object,
  mtblsId = character(),
  assayName = character(),
  filePattern = "mzML$|CDF$|cdf$|mzXML$",
  offline = FALSE,
  ...
)

## S4 method for signature 'MsBackendMetaboLights'
backendRequiredSpectraVariables(object, ...)

mtbls_sync(x, offline = FALSE)
```

Arguments

object	an instance of MsBackendMetaboLights.
mtblsId	character(1) with the ID of a single MetaboLights data set/experiment.
assayName	character with the file names of assay files of the data set. If not provided (assayName = character(), the default), MS data files of all data set's assays are loaded. Use mtbls_list_files(<MetaboLights ID>, pattern = "^a_") to list all available assay files of a data set <MetaboLights ID>.
filePattern	character with the pattern defining the supported (or requested) file types. Defaults to filePattern = "mzML$\|CDF$|cdf$\|mzXML$" hence restricting to mzML, CDF and mzXML files which are supported by *Spectra*'s MsBackendMzR backend.
offline	logical(1) whether only locally cached content should be evaluated/loaded.
...	additional parameters; currently ignored.
x	an instance of MsBackendMetaboLights.

Details

File names for data files are by default extracted from the column "Derived Spectral Data File" of the MetaboLights data set's *assay* table. If this column does not contain any supported file names, the assay's column "Raw Spectral Data File" is evaluated instead.

The backend uses the BiocFileCache package for caching of the data files. These are stored in the default local *BiocFileCache* cache along with additional metadata that includes the MetaboLights ID and the assay file name with which the data file is associated with. Note that at present only MS data files in *mzML*, *CDF* and *mzXML* format are supported.

The MsBackendMetaboLights backend defines and provides additional spectra variables "mtbls_id", "mtbls_assay_name" and "derived_spectral_data_file" that list the MetaboLights ID, the name of the assay file and the original data file name on the MetaboLights ftp server for each individual spectrum. The "derived_spectral_data_file" can be used for the mapping between the experiment's samples and the individual data files, respective their spectra. This mapping is provided in the MetaboLights assay file.

The MsBackendMetaboLights backend is considered *read-only* and does thus not support changing *m/z* and intensity values directly.

Value

- For MsBackendMetaboLights(): an instance of MsBackendMetaboLights.
- For backendInitialize(): an instance of MsBackendMetaboLights with the MS data of the specified MetaboLights data set.
- For backendRequiredSpectraVariables(): character with spectra variables that are needed for the backend to provide the MS data.
- For mtbls_sync(): the input MsBackendMetaboLights with the paths to the locally cached data files being eventually updated.

Initialization and loading of data

New instances of the class can be created with the MsBackendMetaboLights() function. Data is loaded and initialized using the backendInitialize() function which can be configured with parameters mtblsId, assayName and filePattern. mtblsId must be the ID of a **single** (existing) MetaboLights data set. Parameter assayName allows to define specific *assays* of the MetaboLights data set from which the data files should be loaded. If provided, it should be the file name(s) of the respective assay(s) in MetaboLights (use e.g. mtbls_list_files(<MetaboLights ID>, pattern = "^a_") to list all available assay files for a given MetaboLights ID <MetaboLights ID>). By default, with assayName = character() MS data files from **all** assays of a data set are loaded. Optional parameter filePattern defines the pattern that should be used to filter the file names of the MS data files. It defaults to data files with file endings of supported MS data files. backendInitialize() requires an active internet connection as the function first compares the remote file content to the locally cached files and eventually synchronizes changes/updates. This can be skipped with offline = TRUE in which case only locally cached content is queried.

The backendRequiredSpectraVariables() function returns the names of the spectra variables required for the backend to provide the MS data.

The mtbls_sync() function can be used to *synchronize* the local data cache and ensure that all data files are locally available. The function will check the local cache and eventually download missing data files from the MetaboLights repository.

Note

To account for high server load and eventually failing or rejected downloads from the MetaboLights ftp server, the download functions repeatedly retry to download a file. An error is thrown if download fails for 3 consecutive attempts. Between each attemp, the function waits for an increasing time period (5 seconds between the first and second and 10 seconds between the 2nd and 3rd attempt). This time period can also be configured with the `"metabolights.sleep_mult"` option, which defines the *sleep time multiplicator* (defaults to 5).

Author(s)

Philippine Louail, Johannes Rainer

Examples

```
library(MsBackendMetaboLights)

## List files of a MetaboLights data set
mtbls_list_files("MTBLS39")

## Initialize a MsBackendMetaboLights representing all MS data files of
## the data set with the ID "MTBLS39". This will download and cache all
## files and subsequently load and represent them in R.

be <- backendInitialize(MsBackendMetaboLights(), "MTBLS39")
be

## The `mtbls_sync()` function can be used to ensure that all data files are
## available locally. This function will eventually download missing data
## files or update their paths.
be <- mtbls_sync(be)
```

`retry` *Retry expression on failure*

Description

`retry()` retries, upon failure, the evaluation of an expression `exp` for `ntimes` times waiting an increasing amount of time between tries, i.e., waiting for `Sys.sleep(i * sleep_mult)` seconds between each try `i`. If `expr` fails for `ntimes` times an error will be thrown.

Usage

```
retry(expr, ntimes = 5L, sleep_mult = 0L)
```

Arguments

`expr`	Expression to be evaluated.
`ntimes`	`integer(1)` with the number of times to try.
`sleep_mult`	`numeric(1)` multiplier to define the increasing waiting time (in seconds).

Note

Warnings are suppressed.

Author(s)

Johannes Rainer

Index

backendInitialize,MsBackendMetaboLights-method
        *(*MsBackendMetaboLights*)*, 4
backendRequiredSpectraVariables,MsBackendMetaboLights-method
        *(*MsBackendMetaboLights*)*, 4

grepl(), *3*

MetaboLights-utils, 2
MsBackendMetaboLights, 4
MsBackendMetaboLights-class
        *(*MsBackendMetaboLights*)*, 4
mtbls_cached_data_files
        *(*MetaboLights-utils*)*, 2
mtbls_cached_data_files(), *5*
mtbls_delete_cache
        *(*MetaboLights-utils*)*, 2
mtbls_ftp_path *(*MetaboLights-utils*)*, 2
mtbls_list_files *(*MetaboLights-utils*)*, 2
mtbls_sync *(*MsBackendMetaboLights*)*, 4
mtbls_sync_data_files
        *(*MetaboLights-utils*)*, 2

retry, 7

Spectra::MsBackendMzR, *5*