Package 'MsBackendMgf'

December 4, 2024

Title Mass Spectrometry Data Backend for Mascot Generic Format (mgf)
Files

Version 1.15.1

Description Mass spectrometry (MS) data backend supporting import and
export of MS/MS spectra data from Mascot Generic Format
(mgf) files. Objects defined in this package are supposed to be
used with the Spectra Bioconductor package. This package thus
adds mgf file support to the Spectra package.

Depends R (>= 4.0), Spectra (>= 1.5.14)

Imports ProtGenerics (>= 1.35.3), BiocParallel, S4Vectors, IRanges,
MsCoreUtils, methods, stats

Suggests testthat, knitr (>= 1.1.0), roxygen2, BiocStyle (>= 2.5.19),
rmarkdown

License Artistic-2.0

LazyData yes

Encoding UTF-8

VignetteBuilder knitr

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

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

biocViews Infrastructure, Proteomics, MassSpectrometry, Metabolomics,
DataImport

Roxygen list(markdown=TRUE)

RoxygenNote 7.3.2

Collate 'hidden_aliases.R' 'MsBackendMgf.R' 'functions-mgf.R'

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

git_branch devel

git_last_commit 40c14e0

git_last_commit_date 2024-11-20

Repository Bioconductor 3.21

Author RforMassSpectrometry Package Maintainer [cre],
Laurent Gatto [aut] (ORCID: <https://orcid.org/0000-0002-1520-2268>),
Johannes Rainer [aut] (ORCID: <https://orcid.org/0000-0002-6977-7147>),
Sebastian Gibb [aut] (ORCID: <https://orcid.org/0000-0001-7406-4443>),
Michael Witting [ctb] (ORCID: <https://orcid.org/0000-0002-1462-4426>),
Adriano Rutz [ctb] (ORCID: <https://orcid.org/0000-0003-0443-9902>)

Maintainer
RforMassSpectrometry Package Maintainer <maintainer@rformassspectrometry.org>

Contents

---

hidden_aliases   *Internal page for hidden aliases*

Description

For S4 methods that require a documentation entry but only clutter the index.

MsBackendMgf   *MS data backend for mgf files*

Description

The MsBackendMgf class supports import and export of MS/MS spectra data from/to files in Mascot Generic Format (mgf) files. After initial import, the full MS data is kept in memory. MsBackendMgf extends the MsBackendDataFrame() backend directly and supports thus the applyProcessing() function to make data manipulations persistent.

New objects are created with the MsBackendMgf function. The backendInitialize method has to be subsequently called to initialize the object and import MS/MS data from (one or more) mgf files.

The MsBackendMgf backend provides an export method that allows to export the data from the Spectra object (parameter x) to a file in mgf format. See the package vignette for details and examples.

Default mappings from fields in the MGF file to spectra variable names are provided by the spectraVariableMapping function. This function returns a named character vector were names are the spectra variable names and the values the respective field names in the MGF files. This named character vector is submitted to the import and export function with parameter mapping. It is also possible to pass own mappings (e.g. for special MGF dialects) with the mapping parameter.

Usage

```
## S4 method for signature 'MsBackendMgf'
backendInitialize(
  object,
  files,
  mapping = spectraVariableMapping(object),
  nlines = -1L,
  ...,
  BPPARAM = SerialParam()
)

MsBackendMgf()

## S4 method for signature 'MsBackendMgf'
spectraVariableMapping(object, format = c("mgf"))

## S4 method for signature 'MsBackendMgf'
export(
  object,
  x,
  file = tempfile(),
  mapping = spectraVariableMapping(object),
  exportTitle = TRUE,
  ...
)
```

Arguments

object	Instance of `MsBackendMgf` class.
files	`character` with the (full) file name(s) of the mgf file(s) from which MS/MS data should be imported.
mapping	for `backendInitialize` and `export`: named `character` vector allowing to specify how fields from the MGF file should be renamed. Names are supposed to be the spectra variable name and values of the vector the field names in the MGF file. See output of `spectraVariableMapping()` for the expected format and examples below or description above for details.
nlines	for `backendInitialize`: `integer(1)` defining the number of lines that should be imported and processed from the MGF file(s). By default (`nlines = -1L`) the full file is imported and processed at once. If set to a positive integer, the data is imported and processed *chunk-wise* using [readMgfSplit()]().
...	Currently ignored.
BPPARAM	Parameter object defining the parallel processing setup. If parallel processing is enabled (with BPPARAM different than `SerialParam()`, the default) and length of `files` is larger than one, import is performed in parallel on a per-file basis. If data is to be imported from a single file (i.e., length of `files` is one), parsing of the imported file is performed in parallel. See also [SerialParam()]() for information on available parallel processing setup options.

format	for spectraVariableMapping: character(1) defining the format to be used. Currently only format = "mgf" is supported.
x	for export: an instance of Spectra() class with the data that should be exported.
file	character(1) with the (full) file name to which the data should be exported.
exportTitle	logical(1) whether the *TITLE* field should be included in the exported MGF file. If TRUE (the default) a spectraVariable called "TITLE" will be used, if no such variable is present either the spectraNames(object) will be used or, if they are empty, a title will be generated including the MS level, retention time and acquisition number of the spectrum.

Value

See description above.

Author(s)

Laurent Gatto and Johannes Rainer

Examples

```
library(BiocParallel)
fls <- dir(system.file("extdata", package = "MsBackendMgf"),
    full.names = TRUE, pattern = "mgf$")

## Create an MsBackendMgf backend and import data from test mgf files.
be <- backendInitialize(MsBackendMgf(), fls)
be

be$msLevel
be$intensity
be$mz

## The spectra variables that are available; note that not all of them
## have been imported from the MGF files.
spectraVariables(be)

## The variable "TITLE" represents the title of the spectrum defined in the
## MGF file
be$TITLE

## The default mapping of MGF fields to spectra variables is provided by
## the spectraVariableMapping function
spectraVariableMapping(MsBackendMgf())

## We can provide our own mapping e.g. to map the MGF field "TITLE" to a
## variable named "spectrumName":
map <- c(spectrumName = "TITLE", spectraVariableMapping(MsBackendMgf()))
map

## We can then pass this mapping with parameter `mapping` to the
```

```
## backendInitialize method:
be <- backendInitialize(MsBackendMgf(), fls, mapping = map)

## The title is now available as variable named spectrumName
be$spectrumName

## Next we create a Spectra object with this data
sps <- Spectra(be)

## We can use the 'MsBackendMgf' also to export spectra data in mgf format.
out_file <- tempfile()
export(sps, backend = MsBackendMgf(), file = out_file, map = map)

## The first 20 lines of the generated file:
readLines(out_file, n = 20)

## Next we add a new spectra variable to each spectrum
sps$spectrum_idx <- seq_along(sps)

## This new spectra variable will also be exported to the mgf file:
export(sps, backend = MsBackendMgf(), file = out_file, map = map)
readLines(out_file, n = 20)
```

---

readMgf   *Reading MGF files*

Description

The readMgf function imports the data from a file in MGF format reading all specified fields and returning the data as a [DataFrame()](#).

For very large MGF files the readMgfSplit function might be used instead. In contrast to the readMgf functions, readMgfSplit reads only nlines lines from an MGF file at once reducing thus the memory demand (at the cost of a lower performance, compared to readMgf).

Usage

```
readMgf(
  f,
  msLevel = 2L,
  mapping = spectraVariableMapping(MsBackendMgf()),
  ...,
  BPPARAM = SerialParam()
)

readMgfSplit(
  f,
  msLevel = 2L,
  mapping = spectraVariableMapping(MsBackendMgf()),
```

```
    nlines = 1e+05,
    BPPARAM = SerialParam(),
    ...
)
```

Arguments

`f`	`character(1)` with the path to an mgf file.
`msLevel`	`numeric(1)` with the MS level. Default is 2.
`mapping`	named `character` vector to rename mgf fields to spectra variables.
`...`	Additional parameters, currently ignored.
`BPPARAM`	parallel processing setup that should be used. Only the parsing of the imported MGF file is performed in parallel.
`nlines`	for `readMgfSplit`: `integer(1)` with the number of lines that should be imported and parsed in each iteration.

Value

A `DataFrame` with each row containing the data from one spectrum in the MGF file. m/z and intensity values are available in columns `"mz"` and `"intensity"` in a list representation.

Author(s)

Laurent Gatto, Johannes Rainer, Sebastian Gibb

Examples

```
fls <- dir(system.file("extdata", package = "MsBackendMgf"),
    full.names = TRUE, pattern = "mgf$")[1L]

readMgf(fls)
```

Index