

Package ‘mzR’

March 26, 2013

Type Package

Title parser for netCDF, mzXML, mzData and mzML files (mass spectrometry data)

Version 1.4.6

Author Bernd Fischer, Steffen Neumann, Laurent Gatto

Maintainer Bernd Fischer <bernd.fischer@embl.de>, Steffen Neumann
<sneumann@ipb-halle.de>, Laurent Gatto <lg390@cam.ac.uk>

Description mzR provides a unified API to the common file formats
and parsers available for mass spectrometry data.

It comes with a wrapper for the ISB random access parser for mass
spectrometry mzXML, mzData and mzML files. The package
contains the original code written by the ISB, and a
subset of the proteowizard library for mzML. The netCDF reading
code has previously been used in XCMS.

License Artistic-2.0

LazyLoad yes

Depends Rcpp (>= 0.10.1), methods, utils

Imports Biobase

Suggests msdata (>= 0.1.9), RUnit, faahKO

LinkingTo Rcpp

RcppModules Ramp

SystemRequirements GNU make, NetCDF, zlib

biocViews Infrastructure, Bioinformatics, DataImport, Proteomics, Metabolomics, MassSpectrometry

R topics documented:

mzR-package	2
metadata	2
mzR-class	3
openMSfile	5
peaks	6

Index	8
--------------	----------

mzR-package	<i>parser for mzXML, mzData and mzML files (mass spectrometry data)</i>
-------------	---

Description

The mzR package is a library purely for accessing mass spectrometry data in a wide range of formats. Several backend libraries are used, such as the ISB random access parser (RAMP) for mass spectrometry mzXML, mzData and mzML files. The package contains the original RAMP code written by the ISB, and a subset of the proteowizard library for mzML.

Details

Further information is available in the following vignette:

mzR mzR, Ramp, mzXML, mzData, mzML (source, pdf)

Author(s)

Bernd Fischer, Steffen Neumann, Laurent Gatto

Maintainers: Bernd Fischer <bernd.fischer@embl.de>, Steffen Neumann <sneumann@ipb-halle.de>, Laurent Gatto <lg390@cam.ac.uk>

References

<http://dx.doi.org/10.1074/mcp.R110.000133>

metadata	<i>Access the metadata from an mzR object.</i>
----------	--

Description

Accessors to the analytical setup metadata of a run. runInfo will show a summary of the experiment as a named list, including scanCount, lowMZ, highMZ, startMZ, endMZ, dStartTime and dEndTime. The instrumentInfo method returns a named list including instrument manufacturer, model, ionisation technique, analyzer and detector. These individual pieces of information can also be directly accessed by the specific methods.

Usage

```
runInfo(object)
analyzer(object)
detector(object)
instrumentInfo(object)
ionisation(object)
manufacturer(object)
model(object)
```

Arguments

object An instantiated mzR object.

Author(s)

Steffen Neumann and Laurent Gatto

See Also

See for example [peaks](#) to access the data for the spectra in a "mzR" class.

Examples

```
library(msdata)
filepath <- system.file("microtofq", package = "msdata")
file <- list.files(filepath, pattern="MM14.mzML",
                  full.names=TRUE, recursive = TRUE)
mz <- openMSfile(file)
fileName(mz)
runInfo(mz)
close(mz)
```

mzR-class

Class mzR and sub-classes

Description

The class mzR is the main class for the common mass spectrometry formats. It is a virtual class and thus not supposed to be instantiated directly.

The sub-classes implement specific APIs to access the underlying data and metadata in the files. Currently, mzRramp is the only available implementation. It uses the ISB 'RAMP' random access C/C++ API to access the relevant information in mzData, mzXML and mzML files.

Additional sub-classes using the proteowizard API and netCDF are planned.

Objects from the Class

mzR is a virtual class, so instances cannot be created.

Objects can be created by calls of the form `new("mzRramp", ...)`, but more often they will be created with [openMSfile](#).

Slots

fileName: Object of class character storing the original filename used when the instance was created.

backend: One of the implemented backends or NULL.

.___classVersion__: Object of class "Versioned", from Biobase.

Extends

Class "[Versioned](#)", directly.

Methods

Methods currently implemented for mzR

fileName signature(object = "mzR"): ...

Methods currently implemented for mzRramp

analyzer signature(object = "mzRramp"): ...

close signature(con = "mzRramp"): ...

detector signature(object = "mzRramp"): ...

fileName signature(object = "mzRramp"): ...

get3Dmap signature(object = "mzRramp"): ...

header signature(object = "mzRramp", scans = "missing"): ...

header signature(object = "mzRramp", scans = "numeric"): ...

header signature(object = "mzRnetCDF", scans = "missing"): ...

header signature(object = "mzRnetCDF", scans = "numeric"): ...

initializeRamp signature(object = "mzRramp"): ...

instrumentInfo signature(object = "mzRramp"): ...

ionisation signature(object = "mzRramp"): ...

isInitialized signature(object = "mzRramp"): ...

length signature(x = "mzRramp"): ...

manufacturer signature(object = "mzRramp"): ...

model signature(object = "mzRramp"): ...

peaksCount signature(object = "mzRramp", scans = "missing"): ...

peaksCount signature(object = "mzRramp", scans = "numeric"): ...

peaks signature(object = "mzRramp", scans = "missing"): ...

peaks signature(object = "mzRramp", scans = "numeric"): ...

peaks signature(object = "mzRnetCDF", scans = "missing"): ...

peaks signature(object = "mzRnetCDF", scans = "numeric"): ...

runInfo signature(object = "mzRramp"): ...

Author(s)

Steffen Neumann and Laurent Gatto

References

RAMP: <http://tools.proteomecenter.org/wiki/index.php?title=Software:RAMP> Proteowizard: <http://proteowizard.sourceforge.net/>

Examples

```
library(msdata)
filepath <- system.file("microtofq", package = "msdata")
file <- list.files(filepath, pattern="MM14.mzML",
                  full.names=TRUE, recursive = TRUE)
mzml <- openMSfile(file)
close(mzml)
```

openMSfile	<i>Create and check mzR objects from netCDF, mzXML, mzData or mzML files.</i>
------------	---

Description

The openMSfile constructor will create a new format-specific mzR object, open 'filename' file and all header information is loaded as a Rcpp module and made accessible through the ramp slot of the resulting object.

Usage

```
openMSfile(filename, backend=c("Ramp", "netCDF"), verbose = FALSE)
initializeRamp(object)
isInitialized(object)
fileName(object)
```

Arguments

filename	Path name of the netCDF, mzData, mzXML or mzML file to read.
backend	A character specifying with backend API to use. Currently only 'Ramp' and 'netCDF' are available. 'pwiz' will be added in future version.
object	An instantiated mzR object.
verbose	Enable verbose output.

Author(s)

Steffen Neumann and Laurent Gatto

Examples

```
library(msdata)
filepath <- system.file("microtofq", package = "msdata")
file <- list.files(filepath, pattern="MM14.mzML",
                  full.names=TRUE, recursive = TRUE)
mz <- openMSfile(file)
fileName(mz)
isInitialized(mz)
runInfo(mz)
close(mz)
```

peaks

Access the raw data from an mzR object.

Description

Access the MS raw data. The `peaks` and `peaksCount` functions return the (m/z,intensity) pairs and the number peaks in the spectrum/spectra. `peaks` returns a single matrix if `scans` is a numeric of length 1 and a list of matrices if several scans are asked for or no `scans` argument is provided (i.e all spectra in the object are returned). `peaksCount` will return a numeric of length n.

The `header` function returns a list containing `seqNum`, `acquisitionNum`, `msLevel`, `peaksCount`, `totIonCurrent`, `retentionTime`, `basePeakMZ`, `basePeakIntensity`, `collisionEnergy`, `ionisationEnergy`, `lowM`, `highMZ`, `precursorScanNum`, `precursorMZ`, `precursorCharge`, `precursorIntensity`, `mergedScan`, `mergedResultScanNum`, `mergedResultStartScanNum` and `mergedResultEndScanNum`, when available in the original file. If multiple scans are queried, a `data.frame` is returned with the scans reported along the rows.

The `get3Dmap` function performs a simple resampling between `lowMz` and `highMz` with `resMz` resolution. A matrix of dimensions `length(scans)` times `seq(lowMz,highMz,resMz)` is returned.

Usage

```
header(object, scans, ...)
```

```
peaksCount(object, scans, ...)
```

```
peaks(object, scans, ...)
```

```
get3Dmap(object, scans, lowMz, highMz, resMz, ...)
```

Arguments

<code>object</code>	An instantiated <code>mzR</code> object.
<code>scans</code>	A numeric specifying which scans to return. Optional for the <code>header</code> , <code>peaks</code> and <code>peaksCount</code> methods. If omitted, the requested data for all peaks is returned.
<code>lowMz</code> , <code>highMz</code>	Numerics defining the m/z range to be returned.
<code>resMz</code>	a numeric defining the m/z resolution.
<code>...</code>	Other arguments. Currently ignored.

Author(s)

Steffen Neumann and Laurent Gatto

See Also

[instrumentInfo](#) for metadata access and the "`mzR`" class.

Examples

```
library(msdata)
filepath <- system.file("microtofq", package = "msdata")
file <- list.files(filepath, pattern="MM14.mzML",
                  full.names=TRUE, recursive = TRUE)
mz <- openMSfile(file)
runInfo(mz)
colnames(header(mz))
close(mz)
```

Index

- *Topic **classes**
 - mzR-class, 3
- *Topic **package, file**
 - mzR-package, 2

- analyzer (metadata), 2
- analyzer,mzRramp-method (mzR-class), 3

- class:mzR (mzR-class), 3
- class:mzRnetCDF (mzR-class), 3
- class:mzRramp (mzR-class), 3
- close (mzR-class), 3
- close,mzRnetCDF-method (mzR-class), 3
- close,mzRramp-method (mzR-class), 3

- detector (metadata), 2
- detector,mzRramp-method (mzR-class), 3

- fileName (openMSfile), 5
- fileName,mzR-method (mzR-class), 3

- get3Dmap (peaks), 6
- get3Dmap,mzRramp-method (mzR-class), 3

- header, 6
- header (peaks), 6
- header,mzRnetCDF,missing-method (mzR-class), 3
- header,mzRnetCDF,numeric-method (mzR-class), 3
- header,mzRramp,missing-method (mzR-class), 3
- header,mzRramp,numeric-method (mzR-class), 3

- initializeRamp (openMSfile), 5
- initializeRamp,mzRramp-method (mzR-class), 3
- instrumentInfo, 6
- instrumentInfo (metadata), 2
- instrumentInfo,mzRramp-method (mzR-class), 3
- ionisation (metadata), 2
- ionisation,mzRramp-method (mzR-class), 3

- isInitialized (openMSfile), 5
- isInitialized,mzRnetCDF-method (mzR-class), 3
- isInitialized,mzRramp-method (mzR-class), 3

- length (mzR-class), 3
- length,mzRnetCDF-method (mzR-class), 3
- length,mzRramp-method (mzR-class), 3

- manufacturer (metadata), 2
- manufacturer,mzRramp-method (mzR-class), 3
- metadata, 2
- model (metadata), 2
- model,mzRramp-method (mzR-class), 3
- mzR, 3, 6
- mzR (mzR-package), 2
- mzR-class, 3
- mzR-package, 2
- mzRnetCDF-class (mzR-class), 3
- mzRramp-class (mzR-class), 3

- openMSfile, 3, 5

- peaks, 3, 6
- peaks,mzRnetCDF,missing-method (mzR-class), 3
- peaks,mzRnetCDF,numeric-method (mzR-class), 3
- peaks,mzRramp,missing-method (mzR-class), 3
- peaks,mzRramp,numeric-method (mzR-class), 3
- peaksCount (peaks), 6
- peaksCount,mzRramp,missing-method (mzR-class), 3
- peaksCount,mzRramp,numeric-method (mzR-class), 3

- runInfo (metadata), 2
- runInfo,mzRramp-method (mzR-class), 3

- Versioned, 3