MSnbase input/output capabilities

Laurent Gatto[*]

June 23, 2016

Abstract

This vignette describes *MSnbase*'s input and output capabilities.

Keywords: Mass Spectrometry (MS), proteomics, infrastructure, IO.

Foreword

MSnbase is under active developed; current functionality is evolving and new features will be added. This software is free and open-source software. If you use it, please support the project by citing it in publications:

> Laurent Gatto and Kathryn S. Lilley. *MSnbase - an R/Bioconductor package for isobaric tagged mass spectrometry data visualization, processing and quantitation.* Bioinformatics 28, 288-289 (2011).

Questions and bugs

You are welcome to contact me directly about *MSnbase*. For bugs, typos, suggestions or other questions, please file an issue in our tracking system[1] providing as much information as possible, a reproducible example and the output of `sessionInfo()`.

If you wish to reach a broader audience for general questions about proteomics analysis using R, you may want to use the Bioconductor support site: https://support.bioconductor.org/.

* lg390@cam.ac.uk

[1] https://github.com/lgatto/MSnbase/issues

1 Overview

MSnbase's aims are to facilitate the reproducible analysis of mass spectrometry data within the *R*environment, from raw data import and processing, feature quantification, quantification and statistical analysis of the results [1]. Data import functions for several formats are provided and intermediate or final results can also be saved or exported. These capabilities are presented below.

2 Data input

Raw data Data stored in one of the published XML-based formats. i.e. `mzXML` [2], `mzData` [3] or `mzML` [4], can be imported with the `readMSData` method, which makes use of the *mzR* package to create `MSnExp` objects. The files can be in profile or centroided mode. See `?readMSData` for details.

Peak lists Peak lists in the `mgf` format[2] can be imported using the `readMgfData`. In this case, the peak data has generally been pre-processed by other software. See `?readMgfData` for details.

Quantitation data Third party software can be used to generate quantitative data and exported as a spreadsheet (generally comma or tab separated format). This data as well as any additional meta-data can be imported with the `readMSnSet` function. See `?readMSnSet` for details.

MSnbase also supports the `mzTab` format[3], a light-weight, tab-delimited file format for proteomics data developed within the Proteomics Standards Initiative (PSI). `mzTab` files can be read into *R*with `readMzTabData` to create and `MSnSet` instance.

3 Data output

RData files *R*objects can most easily be stored on disk with the `save` function. It creates compressed binary images of the data representation that can later be read back from the file with the `load` function.

Peak lists `MSnExp` instances as well as individual spectra can be written as `mgf` files with the `writeMgfData` method. Note that the meta-data in the original *R*object can not be included in the file. See `?writeMgfData` for details.

[2] http://www.matrixscience.com/help/data_file_help.html#GEN
[3] https://github.com/HUPO-PSI/mzTab

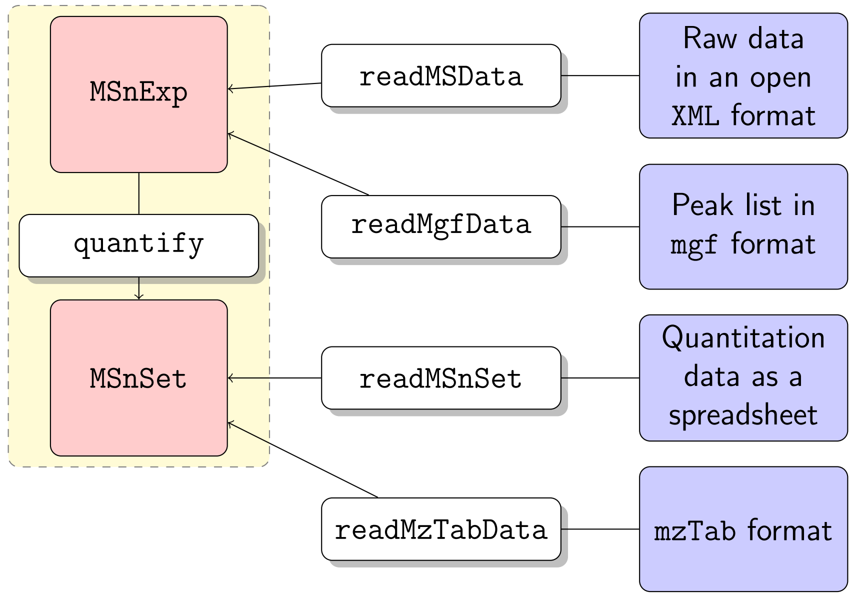

Figure 1: Illustration of MSnbase input capabilities. The white and red boxes represent *R*functions/methods and objects respectively. The blue boxes represent different disk storage formats.

Quantitation data Quantitation data can be exported to spreadsheet files with the `write.exprs` method. Feature meta-data can be appended to the feature intensity values. See `?writeMgfData` for details.

`MSnSet` instances can also be exported to `mzTab` files using the `writeMzTabData` function.

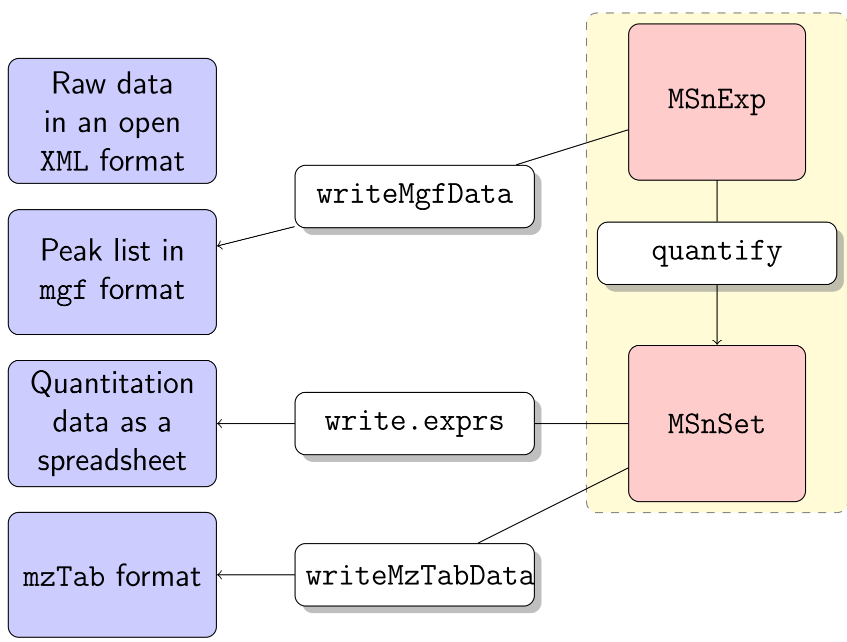

Figure 2: Illustration of MSnbase output capabilities. The white and red boxes represent *R*functions/methods and objects respectively. The blue boxes represent different disk storage formats.

4 Creating MSnSet from text spread sheets

This section describes the generation of MSnSet objects using data available in a text-based spreadsheet. This entry point into *R*and *MSnbase* allows to import data processed by any of the third party mass-spectrometry processing software available and proceed with data exploration, normalisation and statistical analysis using functions available in *R*and the numerous Bioconductor packages.

4.1 A complete work flow

The following section describes a work flow that uses three input files to create the MSnSet. These files respectively describe the quantitative expression data, the sample meta-data and the feature meta-data. It is taken from the *pRoloc* tutorial and uses example files from the *pRolocdata* package.

4.1.1 The original data file

We start by describing the csv to be used as input using the read.csv function.

```
> ## The original data for replicate 1, available
> ## from the pRolocdata package
> f0 <- dir(system.file("extdata", package = "pRolocdata"),
+           full.names = TRUE,
+           pattern = "pr800866n_si_004-rep1.csv")
> csv <- read.csv(f0)
```

The three first lines of the original spreadsheet, containing the data for replicate one, are illustrated below (using the function head). It contains 888 rows (proteins) and 16 columns, including protein identifiers, database accession numbers, gene symbols, reporter ion quantitation values, information related to protein identification, ...

```
> head(csv, n=3)

  Protein.ID        FBgn Flybase.Symbol No..peptide.IDs Mascot.score
1    CG10060 FBgn0001104    G-ialpha65A               3       179.86
2    CG10067 FBgn0000044         Act57B               5       222.40
3    CG10077 FBgn0035720        CG10077               5       219.65
  No..peptides.quantified area.114 area.115 area.116 area.117
1                       1 0.379000 0.281000 0.225000 0.114000
2                       9 0.420000 0.209667 0.206111 0.163889
3                       3 0.187333 0.167333 0.169667 0.476000
  PLS.DA.classification Peptide.sequence Precursor.ion.mass
1                    PM
2                    PM
3
  Precursor.ion.charge pd.2013 pd.markers
```

```
1                             PM    unknown
2                             PM    unknown
3                        unknown    unknown
```

Below read in turn the spread sheets that contain the quantitation data (`exprsFile.csv`), feature meta-data (`fdataFile.csv`) and sample meta-data (`pdataFile.csv`).

```
> ## The quantitation data, from the original data
> f1 <- dir(system.file("extdata", package = "pRolocdata"),
+           full.names = TRUE, pattern = "exprsFile.csv")
> exprsCsv <- read.csv(f1)
> ## Feature meta-data, from the original data
> f2 <- dir(system.file("extdata", package = "pRolocdata"),
+           full.names = TRUE, pattern = "fdataFile.csv")
> fdataCsv <- read.csv(f2)
> ## Sample meta-data, a new file
> f3 <- dir(system.file("extdata", package = "pRolocdata"),
+           full.names = TRUE, pattern = "pdataFile.csv")
> pdataCsv <- read.csv(f3)
```

`exprsFile.csv` containing the quantitation (expression) data for the 888 proteins and 4 reporter tags.

```
> head(exprsCsv, n=3)
         FBgn     X114     X115     X116     X117
1 FBgn0001104 0.379000 0.281000 0.225000 0.114000
2 FBgn0000044 0.420000 0.209667 0.206111 0.163889
3 FBgn0035720 0.187333 0.167333 0.169667 0.476000
```

`fdataFile.csv` containing meta-data for the 888 features (here proteins).

```
> head(fdataCsv, n=3)
         FBgn ProteinID FlybaseSymbol NoPeptideIDs MascotScore
1 FBgn0001104   CG10060   G-ialpha65A            3      179.86
2 FBgn0000044   CG10067        Act57B            5      222.40
3 FBgn0035720   CG10077       CG10077            5      219.65
  NoPeptidesQuantified PLSDA
1                    1    PM
2                    9    PM
3                    3
```

`pdataFile.csv` containing samples (here fractions) meta-data. This simple file has been created manually.

```
> pdataCsv
  sampleNames Fractions
1        X114       4/5
2        X115     12/13
3        X116        19
```

```
4        X117        21
```

The self-contained `MSnSet` can now easily be generated using the `readMSnSet` constructor, providing the respective `csv` file names shown above and specifying that the data is comma-separated (with `sep = ","`). Below, we call that object `res` and display its content.

```
> library("MSnbase")
> res <- readMSnSet(exprsFile = f1,
+                   featureDataFile = f2,
+                   phenoDataFile = f3,
+                   sep = ",")
> res

MSnSet (storageMode: lockedEnvironment)
assayData: 888 features, 4 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: X114 X115 X116 X117
  varLabels: Fractions
  varMetadata: labelDescription
featureData
  featureNames: FBgn0001104 FBgn0000044 ... FBgn0001215 (888
    total)
  fvarLabels: ProteinID FlybaseSymbol ... PLSDA (6 total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
- - - Processing information - - -
 MSnbase version: 1.20.7
```

4.1.2 The `MSnSet` class

Although there are additional specific sub-containers for additional meta-data (for instance to make the object MIAPE compliant), the feature (the sub-container, or slot `featureData`) and sample (the `phenoData` slot) are the most important ones. They need to meet the following validity requirements (see figure 3):

- the number of row in the expression/quantitation data and feature data must be equal and the row names must match exactly, and
- the number of columns in the expression/quantitation data and number of row in the sample meta-data must be equal and the column/row names must match exactly.

A detailed description of the `MSnSet` class is available by typing `?MSnSet` in the *R*console.

The individual parts of this data object can be accessed with their respective accessor methods:

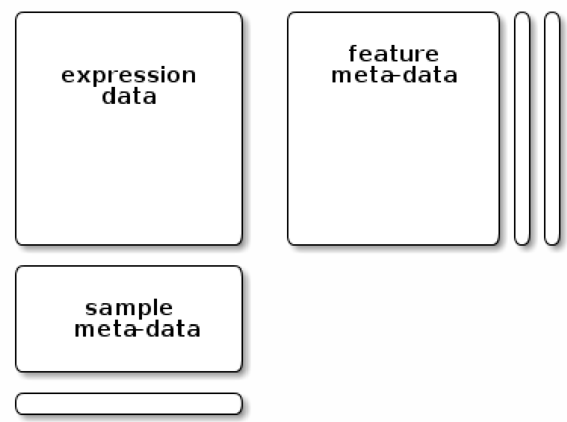

Figure 3: Dimension requirements for the respective expression, feature and sample meta-data slots.

- the quantitation data can be retrieved with `exprs(res)`,
- the feature meta-data with `fData(res)` and
- the sample meta-data with `pData(res)`.

4.2 A shorter work flow

The `readMSnSet2` function provides a simplified import workforce. It takes a single spreadsheet as input (default is `csv`) and extract the columns identified by `ecol` to create the expression data, while the others are used as feature meta-data. `ecol` can be a `character` with the respective column labels or a numeric with their indices. In the former case, it is important to make sure that the names match exactly. Special characters like `'-'` or `'('` will be transformed by *R*into `'.'` when the `csv` file is read in. Optionally, one can also specify a column to be used as feature names. Note that these must be unique to guarantee the final object validity.

```
> ecol <- paste("area", 114:117, sep = ".")
> fname <- "Protein.ID"
> eset <- readMSnSet2(f0, ecol, fname)
> eset

MSnSet (storageMode: lockedEnvironment)
assayData: 888 features, 4 samples
  element names: exprs
protocolData: none
phenoData: none
featureData
  featureNames: CG10060 CG10067 ... CG9983 (888 total)
  fvarLabels: Protein.ID FBgn ... pd.markers (12 total)
  fvarMetadata: labelDescription
```

```
experimentData: use 'experimentData(object)'
Annotation:
- - - Processing information - - -
 MSnbase version: 1.20.7
```

The `ecol` columns can also be queried interactively from *R*using the `getEcols` and `grepEcols` function. The former return a character with all column names, given a splitting character, i.e. the separation value of the spreadsheet (typically `","` for `csv`, `"\t"` for `tsv`, ...). The latter can be used to grep a pattern of interest to obtain the relevant column indices.

```
> getEcols(f0, ",")

 [1] "\"Protein ID\""               "\"FBgn\""
 [3] "\"Flybase Symbol\""           "\"No. peptide IDs\""
 [5] "\"Mascot score\""             "\"No. peptides quantified\""
 [7] "\"area 114\""                 "\"area 115\""
 [9] "\"area 116\""                 "\"area 117\""
[11] "\"PLS-DA classification\""    "\"Peptide sequence\""
[13] "\"Precursor ion mass\""       "\"Precursor ion charge\""
[15] "\"pd.2013\""                  "\"pd.markers\""

> grepEcols(f0, "area", ",")

[1]  7  8  9 10

> e <- grepEcols(f0, "area", ",")
> readMSnSet2(f0, e)

MSnSet (storageMode: lockedEnvironment)
assayData: 888 features, 4 samples
  element names: exprs
protocolData: none
phenoData: none
featureData
  featureNames: 1 2 ... 888 (888 total)
  fvarLabels: Protein.ID FBgn ... pd.markers (12 total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
- - - Processing information - - -
 MSnbase version: 1.20.7
```

The `phenoData` slot can now be updated accordingly using the replacement functions `phenoData<-` or `pData<-` (see `?MSnSet` for details).

5 Session information

- R version 3.3.0 (2016-05-03), `x86_64-w64-mingw32`
- Locale: `LC_COLLATE=C`, `LC_CTYPE=English_United States.1252`, `LC_MONETARY=English_United States.1252`, `LC_NUMERIC=C`, `LC_TIME=English_United States.1252`
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, stats4, utils
- Other packages: AnnotationDbi 1.34.3, Biobase 2.32.0, BiocGenerics 0.18.0, BiocParallel 1.6.2, IRanges 2.6.1, MLInterfaces 1.52.0, MSnbase 1.20.7, ProtGenerics 1.4.0, Rcpp 0.12.5, RcppClassic 0.9.6, Rdisop 1.32.0, S4Vectors 0.10.1, XML 3.98-1.4, annotate 1.50.0, cluster 2.0.4, ggplot2 2.1.0, gplots 3.0.1, knitr 1.13, mzR 2.6.2, pRoloc 1.12.4, pRolocdata 1.10.0, reshape2 1.4.1, zoo 1.7-13
- Loaded via a namespace (and not attached): BiocInstaller 1.22.2, BiocStyle 2.0.2, DBI 0.4-1, DEoptimR 1.0-4, FNN 1.1, KernSmooth 2.23-15, MALDIquant 1.14, MASS 7.3-45, Matrix 1.2-6, MatrixModels 0.4-1, R6 2.1.2, RColorBrewer 1.1-2, RCurl 1.95-4.8, RSQLite 1.0.0, SparseM 1.7, affy 1.50.0, affyio 1.42.0, assertthat 0.1, base64enc 0.1-3, biomaRt 2.28.0, bitops 1.0-6, caTools 1.17.1, car 2.1-2, caret 6.0-70, class 7.3-14, codetools 0.2-14, colorspace 1.2-6, digest 0.6.9, diptest 0.75-7, doParallel 1.0.10, dplyr 0.4.3, e1071 1.6-7, evaluate 0.9, flexmix 2.3-13, foreach 1.4.3, formatR 1.4, fpc 2.1-10, gbm 2.1.1, gdata 2.17.0, genefilter 1.54.2, ggvis 0.4.2, gtable 0.2.0, gtools 3.5.0, highr 0.6, htmltools 0.3.5, htmlwidgets 0.6, httpuv 1.3.3, hwriter 1.3.2, impute 1.46.0, iterators 1.0.8, jsonlite 0.9.22, kernlab 0.9-24, labeling 0.3, lattice 0.20-33, limma 3.28.12, lme4 1.1-12, lpSolve 5.6.13, magrittr 1.5, mclust 5.2, mgcv 1.8-12, mime 0.4, minqa 1.2.4, mlbench 2.1-1, modeltools 0.2-21, munsell 0.4.3, mvtnorm 1.0-5, mzID 1.10.2, nlme 3.1-128, nloptr 1.0.4, nnet 7.3-12, pbkrtest 0.4-6, pcaMethods 1.64.0, pls 2.5-0, plyr 1.8.4, prabclus 2.2-6, preprocessCore 1.34.0, proxy 0.4-15, quantreg 5.26, randomForest 4.6-12, rda 1.0.2-2, rgl 0.95.1441, robustbase 0.92-6, rpart 4.1-10, sampling 2.7, scales 0.4.0, sfsmisc 1.1-0, shiny 0.13.2, snow 0.4-1, splines 3.3.0, stringi 1.1.1, stringr 1.0.0, survival 2.39-4, threejs 0.2.2, tools 3.3.0, trimcluster 0.1-2, vsn 3.40.0, xtable 1.8-2, zlibbioc 1.18.0

References

[1] Laurent Gatto and Kathryn S Lilley. MSnbase – an R/Bioconductor package for isobaric tagged mass spectrometry data visualization, processing and quantitation. *Bioinformatics*, 28(2):288–9, Jan 2012. doi:10.1093/bioinformatics/btr645.

[2] Patrick G A Pedrioli, Jimmy K Eng, Robert Hubley, Mathijs Vogelzang, Eric W Deutsch, Brian Raught, Brian Pratt, Erik Nilsson, Ruth H Angeletti, Rolf Apweiler, Kei Cheung, Catherine E Costello, Henning Hermjakob, Sequin Huang, Randall K Julian, Eugene Kapp, Mark E McComb, Stephen G Oliver, Gilbert Omenn, Norman W Paton, Richard Simpson, Richard Smith, Chris F Taylor, Weimin Zhu, and Ruedi Aebersold. A common open representation of mass spectrometry data and its application to proteomics research. *Nat. Biotechnol.*, 22(11):1459–66, 2004. doi:10.1038/nbt1031.

[3] Sandra Orchard, Luisa Montechi-Palazzi, Eric W Deutsch, Pierre-Alain Binz, Andrew R Jones, Norman Paton, Angel Pizarro, David M Creasy, Jérôme Wojcik, and Henning Hermjakob. Five years of progress in the standardization of proteomics data 4th annual spring workshop of the hupo-proteomics standards initiative april 23-25, 2007 ecole nationale supérieure (ens), lyon, france. *Proteomics*, 7(19):3436–40, 2007. doi:10.1002/pmic.200700658.

[4] Lennart Martens, Matthew Chambers, Marc Sturm, Darren Kes sner, Fredrik Levander, Jim Shofstahl, Wilfred H Tang, Andreas Ro mpp, Steffen Neumann, Angel D Pizarro, Lu isa Montecchi-Palazzi, Natalie Tasman, Mike Coleman, Florian Reisinger, Pune et Souda, Henning Hermjakob, Pierre-Alain Binz, and Eric W Deutsch. mzml - a community standard for mass spectrometry data. *Molecular & Cellular Proteomics : MCP*, 2010. doi:10.1074/mcp.R110.000133.