

MSnbase development

Laurent Gatto and Johannes Rainer

January 4, 2017

Abstract

This vignette describes the classes implemented in *MSnbase* package. It is intended as a starting point for developers or users who would like to learn more or further develop/extend mass spectrometry and proteomics data structures.

Contents

1	Introduction	4
2	<i>MSnbase</i> classes	4
2.1	<i>pSet</i> : a virtual class for raw mass spectrometry data and meta data	4
2.2	<i>MSnExp</i> : a class for MS experiments	5
2.3	<i>OnDiskMSnExp</i> : a on-disk implementation of the <i>MSnExp</i> class	6
2.4	<i>MSnSet</i> : a class for quantitative proteomics data 8	
2.5	<i>MSnProcess</i> : a class for logging processing meta data.	10
2.6	<i>MIAPe</i> : Minimum Information About a Proteomics Experiment	11
2.7	<i>Spectrum et al.</i> : classes for MS spectra	12
2.8	<i>ReporterIons</i> : a class for isobaric tags.	14
2.9	<i>NAnnotatedDataFrame</i> : multiplexed <i>Annotated-DataFrames</i>	15
2.10	Other classes	16
3	Miscellaneous	16
4	Session information	17

Foreword

MSnbase is under active development; 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 (<https://github.com/lgatto/MSnbase/issues>) 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/>.

1 Introduction

This document is not a replacement for the individual manual pages, that document the slots of the *MSnbase* classes. It is a centralised high-level description of the package design.

MSnbase aims at being compatible with the *Biobase* infrastructure [1]. Many meta data structures that are used in *eSet* and associated classes are also used here. As such, knowledge of the *Biobase development and the new eSet* vignette would be beneficial; the vignette can directly be accessed with `vignette("BiobaseDevelopment", package="Biobase")`.

The initial goal is to use the *MSnbase* infrastructure for MS² labelled (iTRAQ [2] and TMT [3]) and label-free (spectral counting, index and abundance) quantitation - see the documentation for the `quantify` function for details.

2 MSnbase classes

All classes have a `.__classVersion__` slot, of class *Versioned* from the *Biobase* package. This slot documents the class version for any instance to be used for debugging and object update purposes. Any change in a class implementation should trigger a version change.

2.1 *pSet*: a virtual class for raw mass spectrometry data and meta data

This virtual class is the main container for mass spectrometry data, i.e spectra, and meta data. It is based on the *eSet* implementation for genomic data. The main difference with *eSet* is that the `assayData` slot is an environment containing any number of *Spectrum* instances (see section 2.7).

MSnbase devel

One new slot is introduced, namely `processingData`, that contains one *MSnProcess* instance (see section 2.5). and the `experimentData` slot is now expected to contain *MIAPe* data (see section 2.6). The `annotation` slot has not been implemented, as no prior feature annotation is known in shotgun proteomics.

```
getClass("pSet")
```

```
Virtual Class "pSet" [package "MSnbase"]
```

```
Slots:
```

Name:	assayData	phenoData
Class:	environment	NAnnotatedDataFrame

Name:	featureData	experimentData
Class:	AnnotatedDataFrame	MIAXE

Name:	protocolData	processingData
Class:	AnnotatedDataFrame	MSnProcess

Name:	.cache	.__classVersion__
Class:	environment	Versions

```
Extends: "Versioned"
```

```
Known Subclasses:
```

```
Class "MSnExp", directly
```

```
Class "OnDiskMSnExp", by class "MSnExp", distance 2, with explicit coerce
```

2.2 MSnExp: a class for MS experiments

MSnExp extends *pSet* to store MS experiments. It does not add any new slots to *pSet*. Accessors and setters are all inherited from *pSet* and new ones should be implemented for *pSet*. Methods that manipulate actual data in experiments are implemented for *MSnExp* objects.

```
getClass("MSnExp")
```

```

Class "MSnExp" [package "MSnbase"]

Slots:

Name:          assayData          phenoData
Class:          environment NAnnotatedDataFrame

Name:          featureData        experimentData
Class:  AnnotatedDataFrame        MIAxE

Name:          protocolData        processingData
Class:  AnnotatedDataFrame        MSnProcess

Name:          .cache      .__classVersion__
Class:          environment          Versions

Extends:
Class "pSet", directly
Class "Versioned", by class "pSet", distance 2

Known Subclasses:
Class "OnDiskMSnExp", directly, with explicit coerce

```

2.3 *OnDiskMSnExp*: a on-disk implementation of the *MSnExp* class

The *OnDiskMSnExp* class extends *MSnExp* and inherits all of its functionality but is aimed to use as little memory as possible based on a balance between memory demand and performance. Most of the spectrum-specific data, like retention time, polarity, total ion current are stored within the object's `featureData` slot. The actual M/Z and intensity values from the individual spectra are, in contrast to *MSnExp* objects, not kept in memory (in the `assayData` slot), but are fetched from the original files on-demand. Because *mzR* package to read the relevant spectrum data is fast and only moderately slower than for in-memory *MSnExp*¹.

¹The *benchmarking* vignette compares data size and operation speed of the two implementations.

MSnbase devel

To keep track of data manipulation steps that are applied to spectrum data (such as performed by methods `removePeaks` or `clean`) a *lazy execution* framework was implemented. Methods that manipulate or subset a spectrum's M/Z or intensity values can not be applied directly to a *OnDiskMSnExp* object, since the relevant data is not kept in memory. Thus, any call to a processing method that changes or subset M/Z or intensity values are added as *ProcessingStep* items to the object's `spectraProcessingQueue`. When the spectrum data is then queried from an *OnDiskMSnExp*, the spectra are read in from the file and all these processing steps are applied on-the-fly to the spectrum data before being returned to the user.

The operations involving extracting or manipulating spectrum data are applied on a per-file basis, which enables parallel processing. Thus, all corresponding method implementations for *OnDiskMSnExp* objects have an argument `BPPARAM` and users can set a `PARALLEL_THRESH` option flag ² that enables to define how and when parallel processing should be performed (using the *BiocParallel* package).

²see [?MSnbaseOptions](#) for details.

Note that all data manipulations that are not applied to M/Z or intensity values of a spectrum (e.g. sub-setting by retention time etc) are very fast as they operate directly to the object's *featureData* slot.

```
getClass("OnDiskMSnExp")  
  
Class "OnDiskMSnExp" [package "MSnbase"]  
  
Slots:  
  
Name:   spectraProcessingQueue      backend  
Class:   list                       character  
  
Name:   assayData                   phenoData  
Class:   environment               NAnnotatedDataFrame  
  
Name:   featureData                 experimentData  
Class:   AnnotatedDataFrame         MIAxE  
  
Name:   protocolData                processingData  
Class:   AnnotatedDataFrame         MSnProcess  
  
Name:   .cache                      .__classVersion__
```

```

Class:                environment                Versions

Extends:
Class "MSnExp", directly
Class "pSet", by class "MSnExp", distance 2
Class "Versioned", by class "MSnExp", distance 3

```

The distinction between *MSnExp* and *OnDiskMSnExp* is often not explicitly stated as it should not matter, from a user's perspective, which data structure they are working with, as both behave in equivalent ways. Often, they are referred to as *in-memory* and *on-disk MSnExp* implementations.

2.4 *MSnSet*: a class for quantitative proteomics data

This class stores quantitation data and meta data after running `quantify` on an *MSnExp* object or by creating an *MSnSet* instance from an external file, as described in the `MSnbase-io` vignette and in `?readMSnSet`, `readMzTabData`, etc. The quantitative data is in form of a $m \times n$ matrix, where m is the number of features/spectra originally in the *MSnExp* used as parameter in `quantify` and n is the number of reporter ions (see section 2.8). If read from an external file, n corresponds to the number of features (protein groups, proteins, peptides, spectra) in the file and m is the number of columns with quantitative data (samples) in the file.

This prompted to keep a similar implementation as the *ExpressionSet* class, while adding the proteomics-specific annotation slot introduced in the *pSet* class, namely `processingData` for objects of class *MSnProcess* (see section 2.5).

```

getClass("MSnSet")

Class "MSnSet" [package "MSnbase"]

Slots:

```


MSnbase devel

```
Name:      experimentData      processingData      qual
Class:      MIAPE              MSnProcess              data.frame

Name:      assayData          phenoData          featureData
Class:      AssayData AnnotatedDataFrame AnnotatedDataFrame

Name:      annotation          protocolData      ___classVersion___
Class:      character AnnotatedDataFrame              Versions

Extends:
Class "eSet", directly
Class "VersionedBiobase", by class "eSet", distance 2
Class "Versioned", by class "eSet", distance 3
```

The *MSnSet* class extends the virtual *eSet* class to provide compatibility for *ExpressionSet*-like behaviour. The experiment meta-data in `experimentData` is also of class *MIAPE* (see section 2.6). The `annotation` slot, inherited from *eSet* is not used. As a result, it is easy to convert *ExpressionSet* data from/to *MSnSet* objects with the coercion method `as`.

```
data(msnset)
class(msnset)

[1] "MSnSet"
attr(,"package")
[1] "MSnbase"

class(as(msnset, "ExpressionSet"))

[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"

data(sample.ExpressionSet)
class(sample.ExpressionSet)

[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"

class(as(sample.ExpressionSet, "MSnSet"))
```

```
[1] "MSnSet"  
attr(,"package")  
[1] "MSnbase"
```

2.5 MSnProcess: a class for logging processing meta data

This class aims at recording specific manipulations applied to *MSnExp* or *MSnSet* instances. The `processing` slot is a `character` vector that describes major processing. Most other slots are of class `logical` that indicate whether the data has been centroided, smoothed, ... although many of the functionality is not implemented yet. Any new processing that is implemented should be documented and logged here.

It also documents the raw data file from which the data originates (`files` slot) and the *MSnbase* version that was in use when the *MSnProcess* instance, and hence the *MSnExp*/*MSnSet* objects, were originally created.

```
getClass("MSnProcess")  
  
Class "MSnProcess" [package "MSnbase"]  
  
Slots:  
  
Name:      files      processing      merged  
Class:     character   character     logical  
  
Name:      cleaned    removedPeaks    smoothed  
Class:     logical     character       logical  
  
Name:      trimmed    normalised      MSnbaseVersion  
Class:     numeric     logical         character  
  
Name:      __classVersion__  
Class:     Versions  
  
Extends: "Versioned"
```

2.6 *MIAPe*: Minimum Information About a Proteomics Experiment

The Minimum Information About a Proteomics Experiment [4, 5] *MIAPe* class describes the experiment, including contact details, information about the mass spectrometer and control and analysis software.

```
getClass("MIAPe")
```

```
Class "MIAPe" [package "MSnbase"]
```

```
Slots:
```

Name:	title	url
Class:	character	character

Name:	abstract	pubMedIds
Class:	character	character

Name:	samples	preprocessing
Class:	list	list

Name:	other	dateStamp
Class:	list	character

Name:	name	lab
Class:	character	character

Name:	contact	email
Class:	character	character

Name:	instrumentModel	instrumentManufacturer
Class:	character	character

Name:	instrumentCustomisations	softwareName
Class:	character	character

Name:	softwareVersion	switchingCriteria
Class:	character	character

Name:	isolationWidth	parameterFile
Class:	numeric	character
Name:	ionSource	ionSourceDetails
Class:	character	character
Name:	analyser	analyserDetails
Class:	character	character
Name:	collisionGas	collisionPressure
Class:	character	numeric
Name:	collisionEnergy	detectorType
Class:	character	character
Name:	detectorSensitivity	.___classVersion__
Class:	character	Versions

Extends:

Class "MIAxE", directly

Class "Versioned", by class "MIAxE", distance 2

2.7 *Spectrum et al.*: classes for MS spectra

Spectrum is a virtual class that defines common attributes to all types of spectra. MS1 and MS2 specific attributes are defined in the *Spectrum1* and *Spectrum2* classes, that directly extend *Spectrum*.

```
getClass("Spectrum")
```

Virtual Class "Spectrum" [package "MSnbase"]

Slots:

Name:	msLevel	peaksCount	rt
Class:	integer	integer	numeric
Name:	acquisitionNum	scanIndex	tic

```

Class:          integer          integer          numeric

Name:           mz              intensity         fromFile
Class:          numeric          numeric          integer

Name:           centroided      smoothed      polarity
Class:          logical          logical          integer

Name:  .__classVersion__
Class:          Versions

Extends: "Versioned"

Known Subclasses: "Spectrum2", "Spectrum1"

```

```

getClass("Spectrum1")

Class "Spectrum1" [package "MSnbase"]

Slots:

Name:           msLevel          peaksCount          rt
Class:          integer          integer          numeric

Name:           acquisitionNum    scanIndex          tic
Class:          integer          integer          numeric

Name:           mz              intensity         fromFile
Class:          numeric          numeric          integer

Name:           centroided      smoothed      polarity
Class:          logical          logical          integer

Name:  .__classVersion__
Class:          Versions

Extends:
Class "Spectrum", directly
Class "Versioned", by class "Spectrum", distance 2

```

```
getClass("Spectrum2")
```

```
Class "Spectrum2" [package "MSnbase"]
```

```
Slots:
```

Name:	merged	precScanNum	precursorMz
Class:	numeric	integer	numeric

Name:	precursorIntensity	precursorCharge	collisionEnergy
Class:	numeric	integer	numeric

Name:	msLevel	peaksCount	rt
Class:	integer	integer	numeric

Name:	acquisitionNum	scanIndex	tic
Class:	integer	integer	numeric

Name:	mz	intensity	fromFile
Class:	numeric	numeric	integer

Name:	centroided	smoothed	polarity
Class:	logical	logical	integer

Name:	.__classVersion__
Class:	Versions

```
Extends:
```

```
Class "Spectrum", directly
```

```
Class "Versioned", by class "Spectrum", distance 2
```

2.8 ReporterIons: a class for isobaric tags

The iTRAQ and TMT (or any other peak of interest) are implemented *ReporterIons* instances, that essentially defines an expected MZ position for the peak and a width around this value as well a names for the reporters.

```
getClass("ReporterIons")
Class "ReporterIons" [package "MSnbase"]

Slots:

Name:          name      reporterNames      description
Class:         character  character      character

Name:          mz        col        width
Class:         numeric   character   numeric

Name:  .__classVersion__
Class:  Versions

Extends: "Versioned"
```

2.9 *NAnnotatedDataFrame*: multiplexed *AnnotatedDataFrames*

The simple expansion of the *AnnotatedDataFrame* classes adds the `multiplex` and `multiLabel` slots to document the number and names of multiplexed samples.

```
getClass("NAnnotatedDataFrame")
Class "NAnnotatedDataFrame" [package "MSnbase"]

Slots:

Name:          multiplex      multiLabels      varMetadata
Class:         numeric      character      data.frame

Name:          data      dimLabels  .__classVersion__
Class:         data.frame  character      Versions

Extends:
Class "AnnotatedDataFrame", directly
```

```
Class "Versioned", by class "AnnotatedDataFrame", distance 2
```

2.10 Other classes

Lists of *MSnSet* instances

When several *MSnSet* instances are related to each other and should be stored together as different objects, they can be grouped as a list into an *MSnSetList* object. In addition to the actual *list* slot, this class also has basic logging functionality and enables iteration over the *MSnSet* instances using a dedicated `lapply` methods.

```
getClass("MSnSetList")  
  
Class "MSnSetList" [package "MSnbase"]  
  
Slots:  
  
Name:                x                log __classVersion__  
Class:                list             list             Versions  
  
Extends: "Versioned"
```

3 Miscellaneous

Unit tests *MSnbase* implements unit tests with the *testthat* package.

Processing methods Methods that process raw data, i.e. spectra should be implemented for *Spectrum* objects first and then `eapply`'ed (or similar) to the *assayData* slot of an *MSnExp* instance in the specific method.

4 Session information

- R version 3.3.2 (2016-10-31), 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 1.36.0, Biobase 2.34.0,
BiocGenerics 0.20.0, BiocParallel 1.8.1, BiocStyle 2.2.1,
IRanges 2.8.1, MLInterfaces 1.54.0, MSnbase 2.0.2,
ProtGenerics 1.6.0, Rcpp 0.12.8, RcppClassic 0.9.6,
Rdisop 1.34.0, S4Vectors 0.12.1, XML 3.98-1.5,
annotate 1.52.1, cluster 2.0.5, ggplot2 2.2.1, gplots 3.0.1,
knitr 1.15.1, microbenchmark 1.4-2.1, msdata 0.14.0,
mzR 2.8.0, pRoloc 1.14.5, pRolocdata 1.12.0, pryr 0.1.2,
reshape2 1.4.2, zoo 1.7-14
- Loaded via a namespace (and not attached):
BiocInstaller 1.24.0, DBI 0.5-1, DEoptimR 1.0-8, FNN 1.1,
KernSmooth 2.23-15, MALDIquant 1.16, MASS 7.3-45,
Matrix 1.2-7.1, MatrixModels 0.4-1, ModelMetrics 1.1.0,
R6 2.2.0, RColorBrewer 1.1-2, RCurl 1.95-4.8, RSQLite 1.1-1,
SparseM 1.74, TH.data 1.0-7, affy 1.52.0, affyio 1.44.0,
assertthat 0.1, backports 1.0.4, base64enc 0.1-3,
biomaRt 2.30.0, bitops 1.0-6, caTools 1.17.1, car 2.1-4,
caret 6.0-73, class 7.3-14, codetools 0.2-15, colorspace 1.3-2,
dendextend 1.3.0, digest 0.6.11, diptest 0.75-7,
doParallel 1.0.10, dplyr 0.5.0, e1071 1.6-7, evaluate 0.10,
flexmix 2.3-13, foreach 1.4.3, fpc 2.1-10, gbm 2.1.1,
gdata 2.17.0, genefilter 1.56.0, ggvis 0.4.3, gridExtra 2.2.1,
gtable 0.2.0, gtools 3.5.0, highr 0.6, htmltools 0.3.5,
htmlwidgets 0.8, httpuv 1.3.3, hwriter 1.3.2, impute 1.48.0,
iterators 1.0.8, jsonlite 1.2, kernlab 0.9-25, labeling 0.3,
lattice 0.20-34, lazyeval 0.2.0, limma 3.30.7, lme4 1.1-12,
lpSolve 5.6.13, magrittr 1.5, mclust 5.2.1, memoise 1.0.0,
mgcv 1.8-16, mime 0.5, minqa 1.2.4, mlbench 2.1-1,
modeltools 0.2-21, multcomp 1.4-6, munsell 0.4.3,

mvtnorm 1.0-5, mzID 1.12.0, nlme 3.1-128, nloptr 1.0.4,
nnet 7.3-12, pbkrtest 0.4-6, pcaMethods 1.66.0, pls 2.6-0,
plyr 1.8.4, prabclus 2.2-6, preprocessCore 1.36.0, proxy 0.4-16,
quantreg 5.29, randomForest 4.6-12, rda 1.0.2-2,
rmarkdown 1.3, robustbase 0.92-7, rpart 4.1-10, rprojroot 1.1,
sampling 2.8, sandwich 2.3-4, scales 0.4.1, sfsmisc 1.1-0,
shiny 0.14.2, snow 0.4-2, splines 3.3.2, stringi 1.1.2,
stringr 1.1.0, survival 2.40-1, threejs 0.2.2, tibble 1.2,
tools 3.3.2, trimcluster 0.1-2, vsn 3.42.3, whisker 0.3-2,
xtable 1.8-2, yaml 2.1.14, zlibbioc 1.20.0

References

- [1] Robert C. Gentleman, Vincent J. Carey, Douglas M. Bates, Ben Bolstad, Marcel Dettling, Sandrine Dudoit, Byron Ellis, Laurent Gautier, Yongchao Ge, Jeff Gentry, Kurt Hornik, Torsten Hothorn, Wolfgang Huber, Stefano Iacus, Rafael Irizarry, Friedrich Leisch, Cheng Li, Martin Maechler, Anthony J. Rossini, Gunther Sawitzki, Colin Smith, Gordon Smyth, Luke Tierney, Jean Y. H. Yang, and Jianhua Zhang. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol*, 5(10):–80, 2004. URL: <http://dx.doi.org/10.1186/gb-2004-5-10-r80>, doi:10.1186/gb-2004-5-10-r80.
- [2] Philip L. Ross, Yulin N. Huang, Jason N. Marchese, Brian Williamson, Kenneth Parker, Stephen Hattan, Nikita Khainovski, Sasi Pillai, Subhakar Dey, Scott Daniels, Subhasish Purkayastha, Peter Juhasz, Stephen Martin, Michael Bartlet-Jones, Feng He, Allan Jacobson, and Darryl J. Pappin. Multiplexed protein quantitation in *saccharomyces cerevisiae* using amine-reactive isobaric tagging reagents. *Mol Cell Proteomics*, 3(12):1154–1169, Dec 2004. URL: <http://dx.doi.org/10.1074/mcp.M400129-MCP200>, doi:10.1074/mcp.M400129-MCP200.
- [3] Andrew Thompson, Jürgen Schäfer, Karsten Kuhn, Stefan Kienle, Josef Schwarz, Günter Schmidt, Thomas Neumann, R Johnstone, A Karim A Mohammed, and Christian Hamon. Tandem mass

- tags: a novel quantification strategy for comparative analysis of complex protein mixtures by MS/MS. *Anal. Chem.*, 75(8):1895–904, 2003.
- [4] Chris F. Taylor, Norman W. Paton, Kathryn S. Lilley, Pierre-Alain Binz, Randall K. Julian, Andrew R. Jones, Weimin Zhu, Rolf Apweiler, Ruedi Aebersold, Eric W. Deutsch, Michael J. Dunn, Albert J. R. Heck, Alexander Leitner, Marcus Macht, Matthias Mann, Lennart Martens, Thomas A. Neubert, Scott D. Patterson, Peipei Ping, Sean L. Seymour, Puneet Souda, Akira Tsugita, Joel Vandekerckhove, Thomas M. Vondriska, Julian P. Whitelegge, Marc R. Wilkins, Ioannis Xenarios, John R. Yates, and Henning Hermjakob. The minimum information about a proteomics experiment (miape). *Nat Biotechnol*, 25(8):887–893, Aug 2007. URL: <http://dx.doi.org/10.1038/nbt1329>, doi:10.1038/nbt1329.
- [5] Chris F Taylor, Pierre-Alain Binz, Ruedi Aebersold, Michel Affolter, Robert Barkovich, Eric W Deutsch, David M Horn, Andreas Håijhmer, Martin Kussmann, Kathryn Lilley, Marcus Macht, Matthias Mann, Dieter MÄijller, Thomas A Neubert, Janice Nickson, Scott D Patterson, Roberto Raso, Kathryn Resing, Sean L Seymour, Akira Tsugita, Ioannis Xenarios, Rong Zeng, and Randall K Julian. Guidelines for reporting the use of mass spectrometry in proteomics. *Nat. Biotechnol.*, 26(8):860–1, 2008. doi:10.1038/nbt0808-860.