MSnbase, efficient R-based access and manipulation of raw mass spectrometry data

Laurent Gatto,[*,†] Sebastian Gibb,[‡] and Johannes Rainer[¶]

†*Computational Biology Unit, de Duve Institute, Université catholique de Louvain, Brussels, Belgium*

‡*Department of Anaesthesiology and Intensive Care of the University Medicine Greifswald, Germany*

¶*Institute for Biomedicine, Eurac Research, Affiliated Institute of the University of Lübeck, Bolzano, Italy*

E-mail: laurent.gatto@uclouvain.be

Abstract

We present version 2 of the `MSnbase` R/Bioconductor package. `MSnbase` provides infrastructure for the manipulation, processing and visualisation of mass spectrometry data. Here we present how the new *on-disk* infrastructure allows the handling of hundreds on commodity hardware and present some application of the package.

Introduction

Mass spectrometry is a powerful technology to assays chemical and biological samples. It is used routinely, with well characterised protocol, as well a development platform, to improve on existing methods and devise new ones to analyse ever more complex sample in greater details. The complexity and diversity of mass spectrometry yields data that is itself complex

and often times of considerable size, that requires non trivial processing before producing interpretable results. This is particularly relevant, and can constitue a significant challenge for method developers that, in addition to the development of sample processing and mass spectrometry methods, need to process and analyse these new data to demonstrate the improvement in their technical and analytical work.

There exists a very diverse catalogue of software tools to explore, process and interpret mass spectrometry data. These range from low level software libraries such as vendor libraries, jmzML (ref), proteowizard (ref), ... that are aimed at programmers to develop new applications, to user-oriented applications, such as ProteomeDiscoverer, MaxQuant, ... that provide a limited and fixed set of functionality. The former are used through application programming interfaces exclusively, while the latter generally featuring graphical user interfaces (GUI).

TODO: Give examples of libraries re-used in user/gui focused application...

In this software note, we present version 2 of the `MSnbase`[1] R/Bioconductor software package. `MSnbase` offers a platform that lies between low level libraries and end-use software. It provides a flexible command line environment for metabolomics and proteomics mass spectrometry-based application, that allows a detailed step-by-step processing, analysis and exploration of the data and development of novel computational mass spectrometry methods.

Software functionality

In `MSnbase`, mass spectrometry experiments are handled as `MSnExp` objects. While the implementation is more complex, it is useful to schematise a raw data experiment as being composed of raw data, i.e. a collection of individual spectra, as well as spectra-level metadata. Each spectrum is composed of m/z values and associated intensities. The metadata are represented by a table with variables along the columns and each row associated to a spectrum. Among the metadata available for each spectrum, we there is its MS level, ac-

quisition number, retention time, precursor m/z and intensity (for MS level 2 and above), and many more. `MSnbase` provides a rich interface to manipulate such objects. The code chunk below illustrates such an object as displayed in the R console and an enumeration of the metadata.

```
> show(ms)
MSn experiment data ("OnDiskMSnExp")
Object size in memory: 0.54 Mb
- - - Spectra data - - -
 MS level(s): 1 2 3
 Number of spectra: 994
 MSn retention times: 45:27 - 47:6 minutes
- - - Processing information - - -
Data loaded [Sun Apr 26 15:40:58 2020]
 MSnbase version: 2.13.6
- - - Meta data  - - -
phenoData
  rowNames: MS3TMT11.mzML
  varLabels: sampleNames
  varMetadata: labelDescription
Loaded from:
  MS3TMT11.mzML
protocolData: none
featureData
  featureNames: F1.S001 F1.S002 ... F1.S994 (994 total)
  fvarLabels: fileIdx spIdx ... spectrum (35 total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
```

```
> fvarLabels(ms)
 [1] "fileIdx"                    "spIdx"
 [3] "smoothed"                   "seqNum"
 [5] "acquisitionNum"             "msLevel"
 [7] "polarity"                   "originalPeaksCount"
 [9] "totIonCurrent"              "retentionTime"
[11] "basePeakMZ"                 "basePeakIntensity"
[13] "collisionEnergy"            "ionisationEnergy"
[15] "lowMZ"                      "highMZ"
[17] "precursorScanNum"           "precursorMZ"
[19] "precursorCharge"            "precursorIntensity"
[21] "mergedScan"                 "mergedResultScanNum"
[23] "mergedResultStartScanNum"   "mergedResultEndScanNum"
[25] "injectionTime"              "filterString"
[27] "spectrumId"                 "centroided"
[29] "ionMobilityDriftTime"       "isolationWindowTargetMZ"
[31] "isolationWindowLowerOffset" "isolationWindowUpperOffset"
[33] "scanWindowLowerLimit"       "scanWindowUpperLimit"
[35] "spectrum"
```

On-disk backend

The main feature in version 2 of the `MSnbase` package was the addition of different backends for raw data storage, namely *in-memory* and *on-disk*. The following code chunk demonstrates how to create two `MSnExp` objects, tailored to manage mass spectrometry experiments, storing data in-memory or on-disk.

```
library("MSnbase")
raw_mem <- readMSData("file.mzML", mode = "inMemory")
raw_dsk <- readMSData("file.mzML", mode = "onDisk")
```

The former is the legacy storage mode, implemented in the first version of the package, that loads all the raw data and the metadata in memory. This solution doesn't scale for modern large dataset, and was complemented by the on-disk backend, that only loads metadata into memory and accesses the spectra in the original files when needed. There are two direct benefits using the on-disk backend, namely faster reading and reduced memory footpring. Figure 1 shows 5-fold faster reading times (a) and over a 10-fold reduction in memory usage (b).

The on-disk backend also offers efficient data manipulation by way of *lazy processing*. Operations on the raw data are stored in a processing queue and only effectively applied when raw data is accessed on disk. As an example, the following short analysis pipeline, that can equally be applied to on in-memory or on-disk data retains MS2 spectra acquired between 1000 and 3000 seconds, extract the m/z range corresponding to the TMT 6-plex range and focuses on the MS2 spectra with a precursor intensity greater than 11×10^6 (the median precursor intensity).

```
x <- x_dsk %>%
    filterRt(c(1000, 3000)) %>%
    filterMz(120, 135)
x[precursorIntensity(x) > 11e6, ]
```

As shown on Figure 1 (c), this lazy mechanism si significantly faster than its application on in-memory data. The advantageous reading and execution times and memory footprint of the on-disk backend are possible by avoiding unnecessary access to the raw data. Once access to the spectra m/z and intensity values become mandatory (for example for plotting), then the in-memory backend becomes more efficient, as illustrated on Figure 1 (d). This

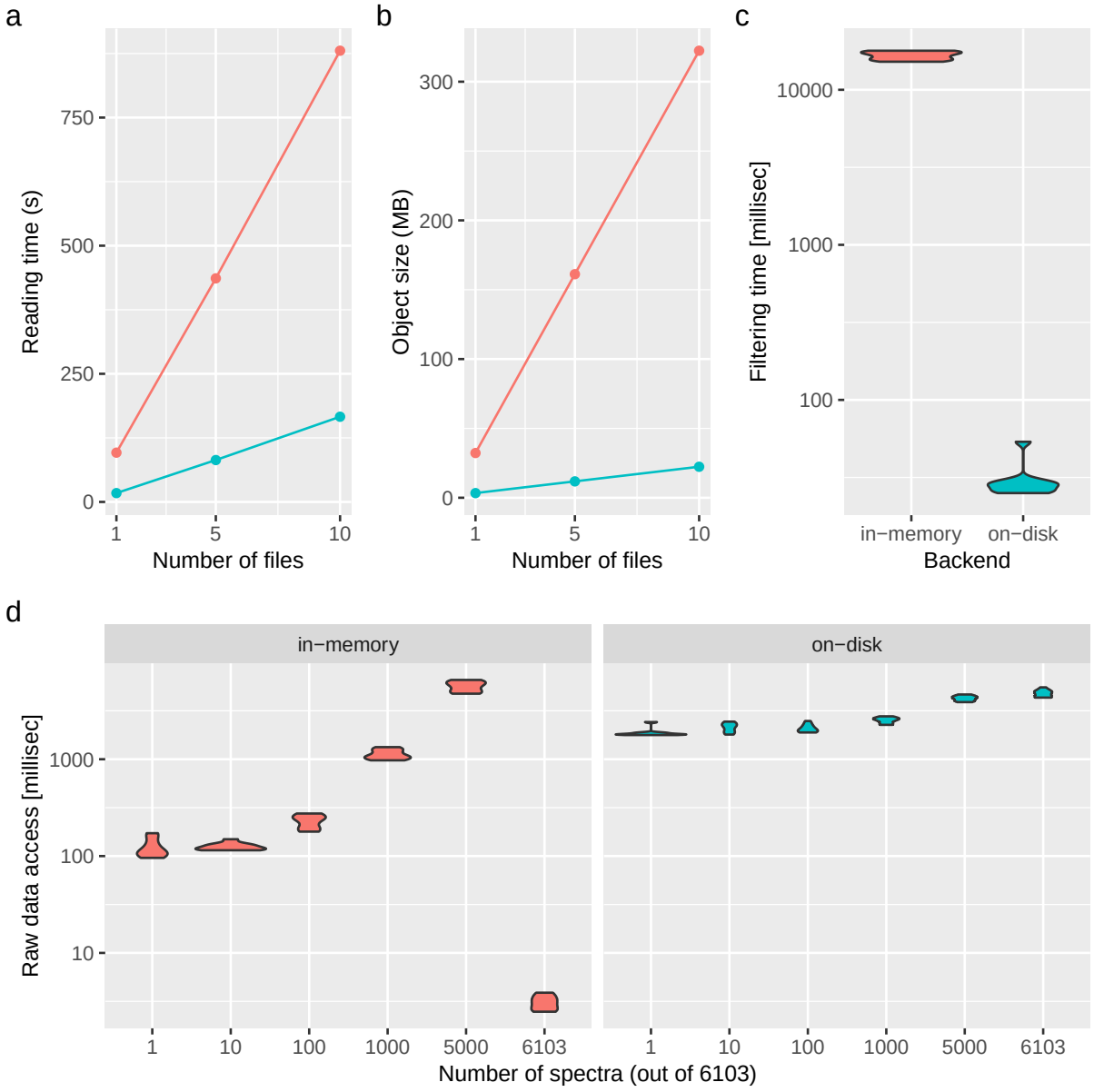

Figure 1: (a) Reading time (in seconds) and (b) data size in memory (in MB) to read/store 1, 5 and 10 files containing 1431 MS1 (on-disk only) and 6103 MS2 (on-disk and in-memory) spectra. (c) Filtering benchmark assessed over 10 interations on in-memory and on-disk data containing 6103 MS2 spectra. (d) Access time to spectra for the in-memory (left) and on-disk (right) backends for 1, 10, 100 1000, 5000 and all 6103 spectra. On-disk backend: blue. In-memory backend: red.

gain is maximal when the whole dataset is the be accessed (i.e. all spectra are already in memory) and negligeable when large fractions of the data need to be subset.

This new on-disk infrastructure enables large scale data analyses using `MSnbase` (metabolomics example, see Johannes).

Prototyping

The `MSnExp` data structure and its interface constitute a efficient prototyping environment for computational method development. We illustrate this by demonstrating how to implement the BoxCar[2] acquisition method. In a nutshell, BoxCar acquisition aims at improving the detection of intact precursor ions by distributing the charge capacity over multiple narrow m/z segments and thus limiting the proportion of highly abundant precursors in each segment. A full scan is reconstructed by combining the respective adjacent segments of the BoxCar acquisitions. The `MSnbaseBoxCar` package[3] is a small package that demonstrates this. The simple method is composed of three steps:

1. Identify and filter the groups of spectra that represent adjacent BoxCar acquisitions (Figure 2 (b)). This can be done using the 'filterString' metadata variable (see code chunk above) that identifies BoxCar spectra by their adjacent M/Z segments with the `bc_groups()` function from `MSnbaseBoxCar`.

2. Remove any signal outside the BoxCar segnments using the `bc_zero_out_box()` function from `MSnbaseBoxCar` (Figures 2 (c) and (d)).

3. Using the `combineSpectra` function from the `MSnbase`, combine the cleaned BoxCar spectra into a new, full spectrum (Figure 2 (e)).

Visualisation

Examples: 3D MSmap, boxcar, centroiding vignette

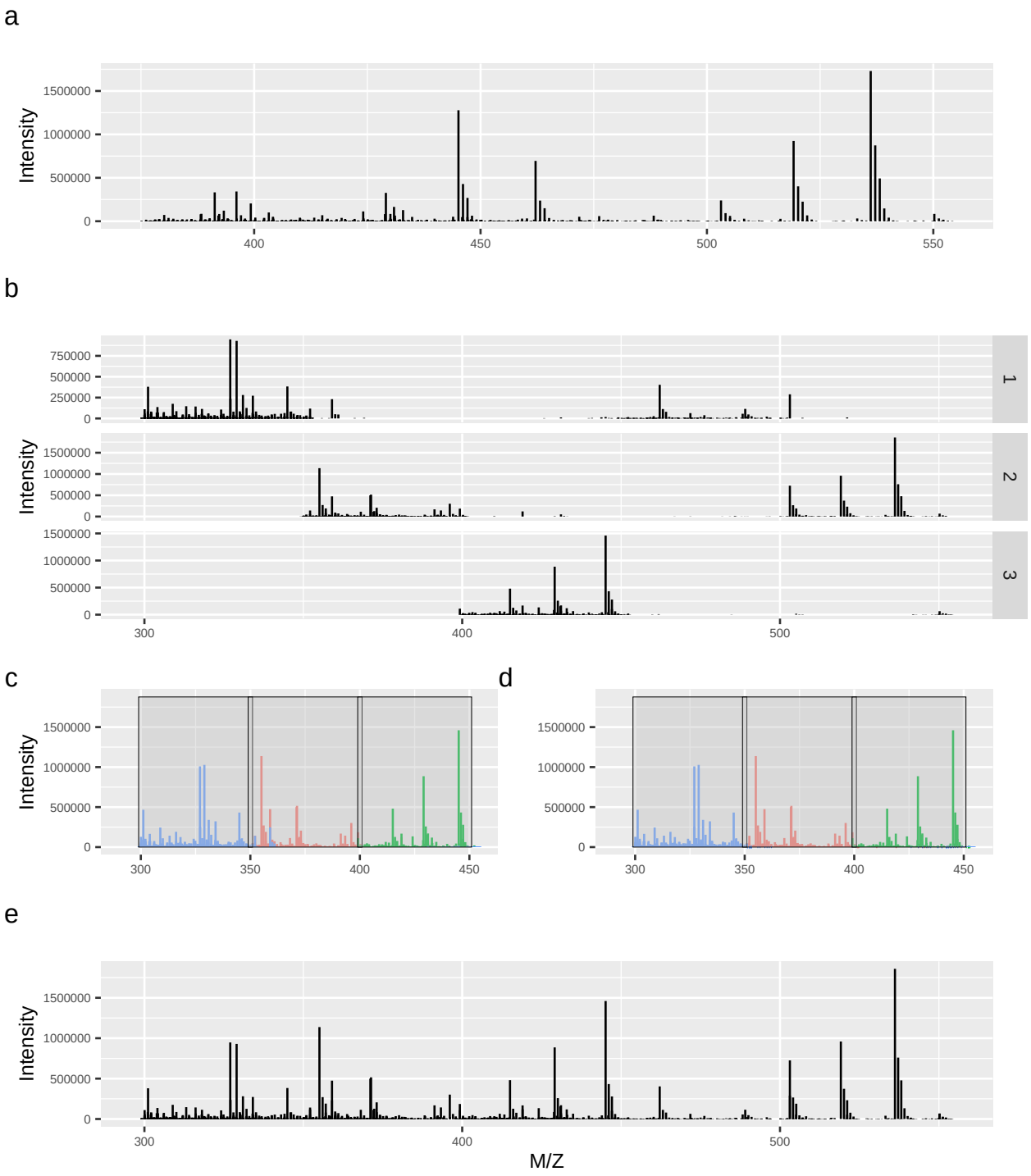

Figure 2: BoxCar processing with `MSnbase`. (a) Standard full scan with (b) three corresponding BoxCar scans showing the adjacent segments. Figure (c) shows the overlapping intact BoxCar segments and (d) the same segments after cleaning, i.e. where peaks outside of the segments were removed. The reconstructed full scan is shown on panel (e).

Discussion

To address (from guidelines):

- potential for reuse: see[4–6] for examples.
- general limitations
- system limitations
- end-user documentation
- developer documentation
- sample data
- benchmark data set
- availability
- license information
- system requirements

Collaborative development, 11 contributors since creation (see blog post).

Count packages depending on `MSnbase`.

Future developments.

The version of `MSnbase` used in this manuscritp is version 2.10.0. The main features presented here were available since version 2.0.

Acknowledgement

The authors thank the various contributors and users who have provided constructuve input and feedback that have helped, over the years, the improvement of the package. The authors declare no conflict of interest.

References

(1) Gatto, L.; Lilley, K. S. MSnbase - an R/Bioconductor package for isobaric tagged mass spectrometry data visualization, processing and quantitation. *Bioinformatics* **2012**, *28*, 288–9.

(2) Meier, F.; Geyer, P.; Virreira Winter, S.; Juergen, C.; Matthias, M. BoxCar acquisition method enables single-shot proteomics at a depth of 10,000 proteins in 100 minutes. *Nature Methods* **2018**, *15*, 440–448.

(3) Gatto, L. MSnbaseBoxCar: BoxCar Data Processing with MSnbase. 2020; R package version 0.1.0.

(4) Wieczorek, S.; Combes, F.; Lazar, C.; Giai Gianetto, Q.; Gatto, L.; Dorffer, A.; Hesse, A. M.; Couté, Y.; Ferro, M.; Bruley, C.; Burger, T. DAPAR & ProStaR: software to perform statistical analyses in quantitative discovery proteomics. *Bioinformatics* **2017**, *33*, 135–136.

(5) Griss, J.; Vinterhalter, G.; Schwämmle, V. IsoProt: A Complete and Reproducible Workflow To Analyze iTRAQ/TMT Experiments. *J Proteome Res* **2019**, *18*, 1751–1759.

(6) Smith, C. A.; Want, E. J.; O'Maille, G.; Abagyan, R.; Siuzdak, G. XCMS: processing mass spectrometry data for metabolite profiling using nonlinear peak alignment, matching, and identification. *Anal Chem* **2006**, *78*, 779–87.