rawDiag – an R package supporting rational LC-MS method optimization for bottom-up proteomics

Christian Trachsel[◇], Christian Panse[◇], Tobias Kockmann[◇], Witold E. Wolski[◇], Jonas Grossmann[◇], Laura Kunz[◇], Jay Tracy[◇], Claudia Fortes[◇], Paolo Nanni[◇] and Ralph Schlapbach[◇]

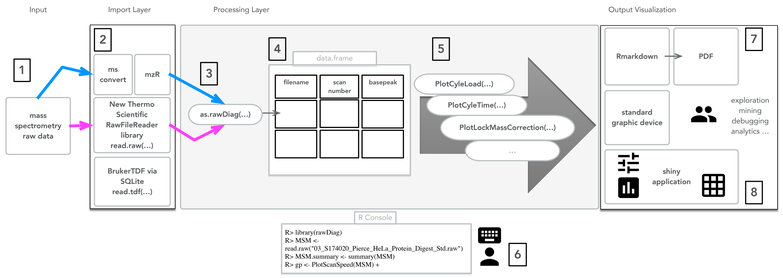


Figure 1: `rawDiag` R package architecture. 1 LC-MS scan files in vendor-specific binary format. 2 `rawDiag` reads scan metadata through vendor library (magenta = Thermo Fisher Scientific RawFileReader, black = Bruker Daltonics .tdf/SQLite), or by conversion to PSI formats using MSconvert (blue = mzML). 3 Utility function coerces scan metadata from different sources to common format. 4 Central data structure used for visualization/modeling: tidy data frame [5] in wide format (rows: scans, columns: scan attributes) 5 The R-package provides a variety of functions to visualize mass spectrum-related metadata (see Table 1 for details). 6 Typical R command line usage for experienced R users. 7 Reporting functionality is provided through R markdown. 8 Interactive GUI-driven exploration is possible through the R shiny package.

1 Introduction

Mass spectra, as well as their metadata, are considered as the raw measurement data and usually recorded in a vendor specific binary format. During a measurement, the mass spectrometer applies internal heuristics which enables the instrument to adapt to sample properties like sample complexity or amount in near real time. Still, method parameters controlling these heuristics, need to be set before the measurement. An optimal measurement result requires a carefully balanced set of parameters, but their complex interactions with each other make LC-MS method optimization a challenging task.

Here we present `rawDiag`, a platform-independent software tool implemented in R that supports LC-MS operators during the process of empirical method optimization. Our work builds on the ideas of the discontinued software "rawMeat" (vastScientific). Our application is currently tailored towards spectral data acquired on Thermo Fisher Scientific instruments (raw format), with a particular focus on Orbitrap mass analyzers (Exactive or Fusion instruments) and the PSI open proteomics file standard [1]. `rawDiag` is meant to run after mass spectrometry acquisition, optimally as an interactive R shiny application and produces a series of diagnostic plots.

2 Implementation

The entire software is implemented as a R [**?**] package providing full documentation and includes example data. A rough sketch of the package can be seen in Figure 1. All diagnostic plots are generated by R functions using the ggplot2 [4] graphical system, based on "The Grammar of Graphics" [6]. The package ships with an adapter function *`read.raw`* which returns an R *`data.frame`* object from the raw data input file. In its current implementation, the adapter functions default input method is set for reading Thermo Fisher Scientific raw files, using a C# programmed executable, based on the platform-independent New RawFileReader .Net assembly [2]. The package also ships with an S3 utility function *`as.rawDiag`* to coerce data from the PSI open proteomics file standard, e.g., by using the code snippet *`as.rawDiag.mzR(openMSfile(mzML))`*, into the by `rawDiag` used tidy data frame [5]. Figure 2 graphs a comparison of both methods.

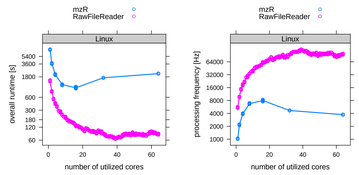


Figure 2: Import layer benchmark – The left plot shows the total processing time needed to extract scan metadata from 128 scan files as a function of utilized computing cores. The right plot shows the corresponding processing speed. Magenta: Extraction from .raw files using the New Thermo Fisher Scientific RawFileReader library [2]. Blue: Extraction from the same files after conversion to mzML format using MSconvert. Time needed for conversion is **not** considered! The benchmark was performed on a Linux Debian 8 system running on a Intel(R) Xeon(R) CPU E5-2698 v3 @ 2.30GHz hardware having 64 cores. The software versions are: R 3.4.3, mzR 2.8.1, and ThermoRawFileReader 4.0.22.

References

[1] Bernd Fischer and Steffen Neumann. mzr, 2017. URL: `https://doi.org/10.18129/b9.bioc.mzr`, `doi:10.18129/b9.bioc.mzr`.

[2] Jim Shofstahl. New RawFileReader from Thermo Fisher Scientific, 2018. URL: `http://planetorbitrap.com/rawfilereader#.WvWESK3QPmE`.

[3] Can Türker, Fuat Akal, Dieter Joho, Christian Panse, Simon Barkow-Oesterreicher, Hubert Rehrauer, and Ralph Schlapbach. B-fabric: The swiss army knife for life sciences. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 717–720, New York, NY, USA, 2010. ACM. URL: `http://doi.acm.org/10.1145/1739041.1739135`, `doi:10.1145/1739041.1739135`.

[4] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. URL: `http://ggplot2.org`, `doi:10.1007/978-0-387-98141-3`.

[5] Hadley Wickham. Tidy data. *Journal of Statistical Software*, 59(10), 2014. URL: `https://doi.org/10.18637/jss.v059.i10`, `doi:10.18637/jss.v059.i10`.

[6] Leland Wilkinson. *The Grammar of Graphics (Statistics and Computing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. `doi:10.1007/0-387-28695-0`.

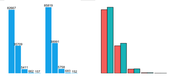
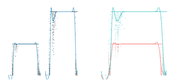
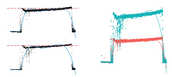
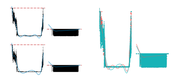
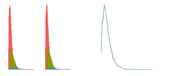
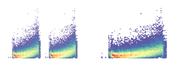
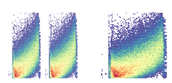
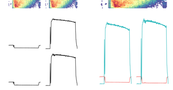
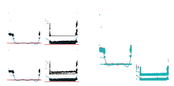
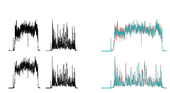
function name	trellis	overlay	violin	description
PlotChargeState			-	displays charge state distributions as biologist-friendly bar charts as absolute counts.
PlotCycleLoad				displays duty cycle load (number of MS2 scans per duty cycle) as a function of retention time (RT) (scatter plots) or its marginal distribution (violin).
PlotCycleTime				displays cycle time with respect to RT (scatter plots) or its marginal distribution (violin). A smooth curve graphs the trend. The maximum is indicated by a red dashed line.
PlotInjectionTime				displays injection time as a function of RT. A smooth curve graphs the trend. The maximum is indicated by a red dashed line.
PlotLockMassCorrection				graphs the lock mass deviations along RT (note: this example data were acquired with lock mass correction).
PlotMassDistribution				displays mass distribution using color coding according to charge state (trellis) or file (overlay, violin).
PlotMassHeatmap			-	draws a computer scientist-friendly hexagon binned heatmap of the peak count charge deconvoluted mass along RT.
PlotMzDistribution				a scatter plot of m/z versus RT on MS1 level (no density; with overplotting.). violin display the marginal m/z distribution of each file.
PlotPrecursorHeatmap			-	according to *PlotMassHeatmap* but displaying convoluted data.
PlotScanFrequency				graphs scan frequency versus RT or scan frequency marginal distribution for violin.
PlotScanTime				plots scan time as function of RT for each MSn level. A smooth curve displays the trend.
PlotTicBasepeak			-	displays the total ion chromatogram (TIC) and the base peak chromatogram.

Table 1: The `rawDiag` cheatsheet lists the main functions of the package. Each thumbnail gives an impression of the generated graphical output using `trellis`, `overlay` or `violin` as method parameter. – indicates missing graphical output due to a discrete response variable. All thumbnails are based on a subset of two .raw files taken from the package dataset `WU163763`. All listed functions return a `ggplot` object [6].

3 Application Example

The R code snippets below show the usage of the package optimization of an LC-MS/MS method.

```
> library(rawDiag)
> data("WU163763"); df <- WU163763
```

Sample Data

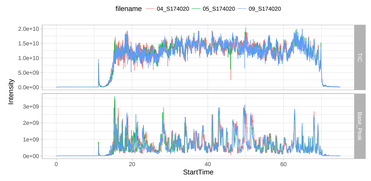
This data was recorded to investigate the optimal number of MS2 scans between two consecutive MS1 scans on a Q-Exactive HF-X mass spectrometer injecting a commercial HeLa digest. The instrument data is available through `http://www.bfabric.org` [3] workunit `WU163763` sample `S174020` or MassIVE MSV000082389. For the following demonstration we select three out of nine mass spectrometry runs.

```
> df <- df[grep("0[459]_S174020",
+   df$filename), ]
```

Total Ion Chromatogram

First, we want to inspect the TIC or base peak chromatogram. With this plot we can see if the data was recorded properly and if the signal response of the sample is the same over the three injections.

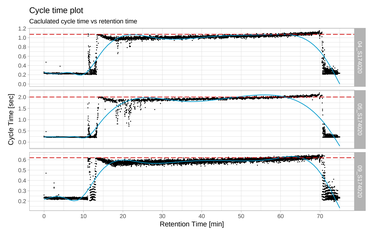
```
> PlotTicBasepeak(df,
+   method = "overlay")
```



Instrument Cycle Time

Next, we want to check the actuall cycle time of the injections.

```
> PlotCycleTime(df)
```

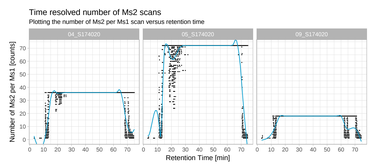


It gets obvious that all three run were recorded usimg different LC-MS methods resulting in cycle times $0.6, 1.0$ and 2 seconds.

Scans per Instrument Cycle

The next thing to check is if the instrument is actually using the available MS2 capacity. For this we plot the cycle load (the actually performed MS2 scans for each MS1 scan).

```
> PlotCycleLoad(df)
```

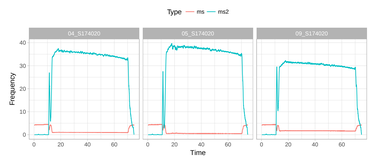


From the plot we can see that all methods actually use a different number of dependend scans (set by the topN parameter).

Scan Frequency

As expected all three mass spectrometry methods result in different scan speed.

```
> PlotScanFrequency(df,
+   method='overlay')
```



Acknowledgements

The authors thank Jim Shofstahl for his support regarding the *New Thermo Fisher RawFileReader* library. We thank Sven Brehmer from Bruker Daltonics for the discussions of timsTOF file format. We thank Lilly van de Venn for the package sticker design. We thank our colleagues at the Functional Genomics Center Zurich, the Swiss Federal Institute of Technology Zurich and the University of Zurich for the support of our work.

ASMS 2018 poster id: 291250, **ThP 375**, Contact: ◇ Functional Genomics Center Zurich, Swiss Federal Institute of Technology Zurich | University of Zurich, Winterthurerstr. 190, CH-8057 Zurich, SWITZERLAND. Phone: +41 44 635 39 12, Fax: +41 44 635 39 22, EMail: `cp@fgcz.ethz.ch`, URL: `http://fgcz.ethz.ch`.