This item is the archived peer-reviewed author-version of:

Open-source, platform-independent library and online scripting environment for accessing thermo scientific RAW files

Reference:

Kelchtermans Pieter, Silva Ana S.C., Argentini Andrea, Staes An, Vandenbussche Jonathan, Laukens Kris, Valkenborg Dirk, Martens Lennart.- Open-source, platform-independent library and online scripting environment for accessing thermo scientific RAW files

Journal of proteome research - ISSN 1535-3893 - 14:11(2015), p. 4940-4943

Full text (Publishers DOI): <http://dx.doi.org/doi:10.1021/acs.jproteome.5b00778>

Full text (Publishers DOI): <http://dx.doi.org/doi:10.1021/acs.jproteome.5600778>

To cite this reference: <http://hdl.handle.net/10067/1295520151162165141>

This document is confidential and is proprietary to the American Chemical Society and its authors. Do not copy or disclose without written permission. If you have received this item in error, notify the sender and delete all copies.

**An open source, platform-independent library and online
scripting environment for accessing Thermo Scientific RAW
files**

Journal:	<i>Journal of Proteome Research</i>
Manuscript ID	pr-2015-00778w.R1
Manuscript Type:	Technical Note
Date Submitted by the Author:	n/a
Complete List of Authors:	<p>Kelchtermans, Pieter; VIB, Medical Biotechnology Center; Ghent University, Department of Biochemistry; Flemish Institute for Technological Research (VITO); Ghent University, Bioinformatics Institute Ghent</p> <p>C. Silva, Ana Silvia; VIB, Medical Biotechnology Center; Ghent University, Department of Biochemistry; Ghent University, Bioinformatics Institute Ghent</p> <p>Argentini, Andrea; VIB, Medical Biotechnology Center; Ghent University, Department of Biochemistry; Ghent University, Bioinformatics Institute Ghent</p> <p>Staes, An; VIB, Medical Biotechnology Center; Ghent University, Department of Biochemistry</p> <p>Vandenbussche, Jonathan; VIB, Medical Biotechnology Center; Ghent University, Department of Biochemistry</p> <p>Laukens, Kris; Antwerp University</p> <p>Valkenburg, Dirk; VITO, Applied Bio & molecular Systems</p> <p>Martens, Lennart; UGent / VIB, Department of Biochemistry</p>

SCHOLARONE™
Manuscripts

An open source, platform-independent library and online scripting environment for accessing Thermo Scientific RAW files

*Pieter Kelchtermans^{1,2,3,4}, Ana S. C. Silva^{1,2,3}, Andrea Argentini^{1,2,3}, An Staes^{1,2}, Jonathan
Vandenbussche^{1,2}, Kris Laukens^{5,6}, Dirk Valkenburg^{4,7,8}, Lennart Martens^{1,2,3,*}*

¹ Medical Biotechnology Center, VIB, Albert Baertsoenkaai 3, 9000 Ghent, Belgium

² Department of Biochemistry, Faculty of Medicine and Health Sciences, Ghent University,
Albert Baertsoenkaai 3, 9000 Ghent, Belgium

³ Bioinformatics Institute Ghent, Ghent University, Ghent, Belgium

⁴ Flemish Institute for Technological Research (VITO), Boeretang 200, B-2400 Mol Belgium

⁵ Department of Mathematics and Computer Science, University of Antwerp, Antwerp, Belgium

⁶ Biomedical Informatics Research Center Antwerp (biomina), University of Antwerp / Antwerp
University Hospital, Antwerp, Belgium

⁷ CFP, University of Antwerp, Groenenborgerlaan 171, 2020 Antwerp, Belgium

⁸ I-BioStat, Hasselt University, Agoralaan - Building D, 3590 Diepenbeek, Belgium

* Correspondence: Professor Lennart Martens, Faculty of Medicine and Health Sciences, Ghent
University, A. Baertsoenkaai 3, B-9000 Ghent, Belgium Email: lennart.martens@vib-ugent.be
Tel.: +32-92649358 Fax: +32-92649484

Abbreviations: MS: Mass Spectrometry; URL: Uniform Resource Locator

Abstract

Mass spectrometers typically output data in proprietary binary formats. While converter suites and standardized XML formats have been developed in response, these conversion steps come with non-negligible overheads in compute time and storage space. As a result, simple, everyday data inspection tasks are often beyond the skills of the mass spectrometrists, who are unable to freely access the acquired data. We therefore here describe the *unthermo* library for convenient, platform-independent access to Thermo Scientific RAW files, and the associated online playground to transform small and easily understandable scriptlets into executable programs for end-users. By fostering the provision of code examples and snippet exchange, the interested mass spectrometrists or researchers can use this playground to quickly assemble custom scripts for their particular purpose. In this way, the data in these RAW files can be mined much more readily and directly by the user, and fast, automated raw data extraction or analysis can finally become part of the daily routine of the mass spectrometrists.

Keywords: Thermo raw data, software library, data access

Main Text

A key milestone in any mass spectrometry experiment is the transition between the wet-lab and the dry-lab workflows. This transition is marked by the acquisition of the raw data by the mass spectrometer, resulting in the output of a vendor-specific binary file. In the case of Thermo Scientific instrumentation, the typical output file is a ‘.RAW’ file. Immediate access to the information in this file is quite useful, for instance as a means to perform rapid quality assessment of the data ¹ or for quantification purposes ². Yet since these raw data formats are usually proprietary, vendor software is required to view or analyze their contents ³. Fortunately, vendors are increasingly open about making their relevant libraries accessible to third parties (e.g., MSFileReader for Thermo Scientific, WiffReader for AB SCIEX, CompassXtract for Bruker, MHDAC for Agilent, and WRDAC for Waters). These software interfaces allow tools or workflows to be built that rely on direct access to raw data files ⁴, even though access can still be cumbersome due to operating system limitations (vendor libraries are typically only developed for Microsoft Windows), inadequately commented libraries or lack of detailed developer information ^{3, 5}. Rather than work with the raw file directly, a conversion can therefore be performed of (parts of) vendor-specific data files to a standard format (notably mzML ⁶) by software tools such as ProteoWizard ⁷. These standard files can then in turn be read by generic readers such as the ms-data-core-api⁸. However, conversion comes with its own limitations. Perhaps most importantly, there is always a processing overhead involved, and standard data formats come with a notable size increase compared to the original raw file. These overheads become particularly noticeable when performing simple tasks such as quality checks or rapid visualizations.

1
2
3 Indeed, despite the availability of the abovementioned approaches for accessing raw data, very
4
5 few mass spectrometrists today are able to interact directly with the data they obtain from their
6
7 instrument, hampering their ability to make the most of their analytics. In order to provide
8
9 convenient access to the raw data, we here therefore describe our platform independent system
10
11 for straightforward access to Thermo Scientific RAW files. Based on the approach followed by
12
13 the *unfinnigan* project ⁹, we have built the *unthermo* open source Go library to read Thermo
14
15 Scientific raw data. While our library provides full access to the RAW files, our library is not
16
17 intended to provide a perfect duplication of all existing calls in other libraries such as the Thermo
18
19 Visual Basic API, or the ProteoWizard API (which is built on top of the Thermo API).
20
21 Furthermore, we have provided an online code playground around this library that can be used to
22
23 transform simple scriptlets to standalone, platform independent executables. The playground is
24
25 also accompanied by a code snippet wiki, to stimulate development and sharing of common
26
27 code.
28
29
30
31
32
33
34
35

36
37 Importantly, our choice to re-implement *unfinnigan* in Go (instead of forking the existing Perl
38
39 code) is motivated by the much easier path from Go to compiled and statically linked
40
41 executables. This allows our *unthermo* library to be used on many different platforms or even
42
43 entire compute clusters ¹⁰ without requiring any installation of libraries or interpreters. A
44
45 thorough description of the Go Playground is already published by the Go Authors ¹¹. Our MS
46
47 Playground starts from this basic functionality and adds three major features: (i) it can serve
48
49 downloadable binaries for a variety of platforms, (ii) it can graphically display results, and (iii) it
50
51 also hosts a version of *unthermo* that runs on specific server-hosted test RAW files.
52
53
54
55
56
57
58
59
60

Upon opening a RAW file with the library, only scan indices are read, so random access to any indexed part of the binary files is possible within *unthermo*. Details about the internal data structures read by the library are described in the documentation, which can be found online at <https://godoc.org/bitbucket.org/proteinspector/ms/unthermo>. It is important to note that the *unthermo* library is a subpackage of *ms*, a general purpose mass spectrometry library. This means that other instrument file readers can be implemented in a similar way under the umbrella of this general purpose library. It should however, be noted that the *unthermo* library does not yet support the parsing of RAW files from the latest Orbitrap Fusion instruments.

Using a simple example that prints every spectrum in a RAW file, we will illustrate the use of our library. When navigating a web browser to <http://ms-playground.appspot.com/>, a code box and a set of buttons are presented as displayed in Figure 1. In this code box, anyone can edit Go code that can subsequently be either run online or compiled for offline use. Note that installation of a local copy of MS Playground is also possible using the tools mentioned above, but this procedure is rather technical and will require local support from a system administrator or developer.

The default program provided, is tasked with loading the required libraries and opening and closing the RAW file. It stops with an error message if the file cannot be opened. Function *printspectrum* calls *unthermo* methods, looping over all peaks in the provided scan's spectrum. For every peak, it simply prints the *m/z* value and intensity. A thoroughly annotated copy of the code can be found in the Supplementary Material.

The program in the code box can be compiled by selecting the “Run”-button which will yield output in the form of two floating point numbers per line. The first column corresponds to the

m/z value and the second column to the intensity value. Clicking the “Download”-button opens a list box wherein command-line executables can be downloaded for multiple platforms (i.e., Linux, MacOS X, or Windows) and hardware architectures (i.e. 32-bit or 64-bit). The playground also offers a scatter plot generator for this type of two-column output by enabling the “Plot” checkbox and selecting the “Run” button again. This action will plot the first column on the horizontal axis, and the second column on the vertical axis. If a user wants to store the program in the code box for later use, or to share it with colleagues, then the “Share”-button should be used. This button provides a URL that will link to the adjusted code, a functionality that is especially convenient for data provenance.

Do note that the name of the RAW file is hard-coded in the example program. This file “small.RAW” is stored on the server, so users of the MS Playground can test their algorithms directly online by using this example file. Obviously, hardcoding file names would be very impractical in practice as users typically want to run the command-line executable on different RAW files. Go therefore provides the “flag” package that allows command-line flags to be supplied to executables such that users are able to define the RAW file on which the executable should operate at invocation time. An example of the *printspectrum* program that uses flags can be found on the MS Playground website and in the Supplementary Material.

It should also be noted that additional test RAW files can be submitted to the developers by users who may have a specific type of RAW file that would be useful to have available in the online environment. Adding such a RAW file to the system is on a per-request basis however, to avoid duplication and to avoid the creation of a *de facto* RAW file repository, which is not the purpose of MS Playground.

In the next two paragraphs we will shortly introduce three other, more advanced applications that are all implemented in the MS playground appspot. The code can be found in the Supplementary Material, and on the *github* wiki page for the MS playground: <https://github.com/pkelchte/ms-playground/wiki>. This wiki page also contains more commonly used snippets. It is encouraged for github users to extend this wiki with new examples, which is facilitated by the lack of access restrictions on that part of the repository. A typical use of the wiki would be a developer going through the wiki on a user's computer, picking relevant examples on the wiki and extending these examples to the user's liking in the MS Playground, ultimately yielding a shareable and downloadable tool.

The first more advanced example tool extracts an ion chromatogram from all MS1 scans. It loops over all scans, and loads the spectrum for MS1 scans only. From each MS1 spectrum, the algorithm selects the highest peak within a certain tolerance around a specific m/z value. This m/z tolerance is specified at the start of the program. The output is a list of the times a peak was detected with this m/z value, along with the intensity of the peak.

The second example program inserts iTRAQ (or TMT) reporter ions from HCD scans into CID spectra to simplify identification and quantification, as described in ¹². The mass spectrometer is here set up to perform both HCD and CID scans for each selected precursor, with the CID spectra intended for identification, while the HCD scans will contain quantitative data in the form of iTRAQ (or TMT) reporter ions. Ideally, the quantitative data from the reporter ions should be added to the CID spectra, allowing identification and quantification to occur on the same spectra. The program therefore loops over all scans in a file, accumulating CID scans and

HCD reporter ion peaks in the respective hash maps *cidScans* and *hcdPeakSpectra*. These maps are indexed by precursor m/z , which is identical for the pairs of HCD and CID MS2 scans following an MS1 scan. If a complete set of MS2 scans is collected, the HCD-derived reporter ion peaks are merged into the corresponding CID scans. This extended scan is subsequently printed in MGF format. The HCD-derived reporter ion peaks, *reporterIons(scan.Spectrum())*, are selected as the highest peaks in the m/z windows centered on the theoretical reporter ion m/z and with a width corresponding to a given tolerance in parts per million.

The third example tool, *peakstats*, calculates the CPTAC quality control parameters used to monitor chromatographic peak shifts over time for peptides derived from a given standard protein (e.g., bovine serum albumin or enolase). It extracts the maximal peak intensity, the retention time at that peak, and the full width at half maximum of relevant peptide ions.

Small programs such as these illustrate the usefulness of having direct and convenient access to RAW files for quality control and data inspection purposes by mass spectrometrists. Furthermore, the convenient transition from existing examples to actual executables *via* the MS Playground makes it easy to start using the library. It should also be noted that the main intended use of our library is in the form of these bespoke executables. This because most programming languages and workflow tools in common use make it very easy to integrate executables in their operations.

We have here presented a simple, open-source interface to access Thermo Scientific RAW files. Using this library in the MS Playground can speed up tool generation, and provides

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

executables for these tools for all common platforms. Over time, we aim to keep the unthermo library and the MS Playground system up-to-date with newer RAW file versions, and are currently working on incorporating RAW file from the Fusion instrument. Moreover, the MS Playground wiki allows code to be archived for provenance, and to be exchanged and re-used by others as well. As a result, our *unthermo* library and online playground provides mass spectrometrists and proteomics researchers with the ability to directly interface with the raw data acquired by their Thermo Scientific instruments.

Supplementary Information

A document with thoroughly annotated copies of the relevant code.

Author Contributions

The manuscript was written through contributions of all authors. All authors have given approval to the final version of the manuscript.

Acknowledgments

Acknowledgments: PK, ASCS, AA, KL, DV and LM acknowledge support from the SBO grant “InSPECtor” (120025) of the Flemish agency for Innovation by Science and Technology (IWT). We thank Gene Selkov for the reverse-engineering of the Thermo raw-data object and for making the corresponding code available as open-source, and Bernard Delanghe from Thermo Fisher Scientific.

References

1. Rudnick, P. A.; Klauser, K. R.; Kilpatrick, L. E.; Tchekhovskoi, D. V.; P. Neta, N. B.; al., e. Performance metrics for liquid chromatography-tandem mass spectrometry systems in proteomic analyses. *Mol Cell Proteomics* **2010**, 2, 225-241.
2. Vaudel, M.; Sickmann, A.; Martens, L. Peptide and protein quantification: a map of the minefield. *Proteomics* **2010**, 4, 650-670.
3. Martens, L.; Nesvizhskii, A. I.; Hermjakob, H.; Adamski, M.; Omenn, G. S.; Vandekerckhove, J.; Gevaert, K. Do we want our data raw? Including binary mass spectrometry data in public proteomics data repositories. *Proteomics* **2005**, 13, 3501-3505.
4. Chambers, M. C.; Maclean, B.; Burke, R.; Amodei, D.; Ruderman, D. L.; Neumann, S.; Gatto, L.; Fischer, B.; Pratt, B.; Egertson, J.; Hoff, K.; Kessner, D.; Tasman, N.; Shulman, N.; Frewen, B.; Baker, T. A.; Brusniak, M.; Paulsen, C.; Creasy, D.; Flashner, L.; Kani, K.; Moulding, K.; Seymour, S. L.; Nuwaysir, L. M.; Lefebvre, B.; Kuhlmann, F.; Roark, J.; Rainer, P.; Detlev, S.; Hemenway, T.; Huhmer, A.; Langridge, J.; Connolly, B.; Chadick, T.; Holly, K.; Eckels, J.; Deutsch, E. W.; Moritz, R. L.; Katz, J. E.; Agus, D. B.; MacCoss, M.; Tabb, D. L.; Mallick, P. A cross-platform toolkit for mass spectrometry and proteomics. *Nature Biotechnology* **2012**, 30, 918-920.
5. Bob Amstadt, M. K. J. Wine. *Linux J.* **1994**, 3, 4es.
6. Martens, L.; Chambers, M.; Sturm, M.; Kessner, D.; Levander, F.; Shofstahl, J.; Tang, W. H.; Rompp, A.; Neumann, S.; Pizarro, A. D.; Montecchi-Palazzi, L.; Tasman, N.; Colemann, M.; Reisinger, F.; Souda, P.; Hermjakob, H.; Binz, P. A.; Deutsch, E. W. mzML - a community standar for mass spectrometry data. *Mol Cell Proteomics* **2011**, 1, R110.000133.
7. Kessner, D.; Chambers, M.; Burke, R.; Agus, D.; Mallick, P. ProteoWizard: Open Source Software for Rapid Proteomics Tools Development. *Bioinformatics* **2008**, 24, 2534-2536.
8. Perez-Riverol, Y.; Uszkoreit, J.-; Sanchez, A.; Ternent, T.; Del Toro, N.; Hermjakob, H.; Vizcaíno, J. A.; Wang, R. ms-data-core-api: an open-source, metadata-oriented library for computational proteomics. *Bioinformatics* **2015**, 31, 2903-2905.
9. Selkov, G. unfinnigan - painless extraction of mass spectra from Thermo "raw" files. <http://code.google.com/p/unfinnigan/>.
10. Verheggen, K.; Barsnes, H.; Martens, L. Distributed computing and data storage in proteomics: many hands make light work, and a stronger memory. *Proteomics* **2014**, 4, 366-377.
11. Gerrand, A. Inside the Go Playground. <http://blog.golang.org/playground> (accessed August 2014).

12. Köcher, T.; Pichler, P.; Schutzbier, M.; Stingl, C.; Kaul, A.; Teucher, N.; Hasenfuss, G.; Penninger, J. M.; Mechtler, K. High precision quantitative proteomics using iTRAQ on an LTQ orbitrap: a new mass spectrometric method combining the benefits of all. *Proteome Res.* **2009**, 8, 4743-4752.

Figure legends

Figure 1: The web interface for MS Playground. The yellow text box holds the Go code. The top buttons allow the user to run or download the code with textual output, or to format or share the code. In this case, the Plot option is activated and numeric results are displayed in a scatter plot that is displayed at the bottom. (This figure is a screenshot taken by the authors, and is in the public domain.)

Figures

MS Playground

Run

Download

Format

Share

Plot

About

```
1 //The printspectrum tool prints out the spectrum (m/z and intensity values) of a
2 //Thermo RAW File
3 //
4 // Every line of the output is a peak registered by the mass spectrometer
5 // characterized by an m/z value in Da and an intensity in the mass spectrometer's unit of abundance
6 package main
7
8 import (
9     "bitbucket.org/proteinspector/ms"
10    "bitbucket.org/proteinspector/ms/unthermo"
11    "flag"
12    "fmt"
13    "log"
14 )
15
16 func main() {
17     var scannumber int
18     var filename string
19
20     //Parse arguments
21     flag.IntVar(&scannumber, "sn", 1, "the scan number")
22     flag.StringVar(&filename, "raw", "small.RAW", "name of the RAW file")
23     flag.Parse()
24
25     //open RAW file
26     file, err := unthermo.Open(filename)
27     if err != nil {
28         log.Fatal(err)
29     }
30 }
```

```

package main

1 7
2 8 import (
3 9     "bitbucket.org/prot
4 10    "bitbucket.org/prot
5 11    "fmt"
6 12    "log"
7 13    "os"
8 14    "strconv"
9 15 )
10 16
11 17 func main() {
12 18     //Parse arguments
13 19     scannumber, _ := strconv.Atoi(os.Args[1])
14 20     filename := os.Args[2]
15 21
16 22     //open RAW file
17 23     rf, err := unthermo.Open(filename)
18 24     if err != nil {
19 25         log.Println(err)
20 26     }
21 27
22 28     //Print the Spectrum at the supplied scan number
23 29     printspectrum(rf.Scan(scannumber))
24 30
25 31     rf.Close()
26 32 }
27 33
28 34 //Print m/z and Intensity of every peak in the spectrum
29 35 func printspectrum(scan ms.Scan) {
30 36     for _, peak := range ms.Environment() {
31 37         fmt.Println(peak.Mz, peak.I)
32 38     }
33 39

```

[32-bit Linux](#)
[32-bit MacOS X](#)
[32-bit Windows](#)
[64-bit Linux](#)
[64-bit MacOS X](#)
[64-bit Windows](#)