`MALDIquantForeign`: Import/Export routines for `MALDIquant`

Sebastian Gibb*

January 5, 2022

Abstract

`MALDIquantForeign` provides routines for importing/exporting different file formats into/from `MALDIquant`.
This vignette describes the usage of the `MALDIquantForeign` package.

*`mail@sebastiangibb.de`

Contents

Foreword

`MALDIquantForeign` is free and open source software for the `R` (R Core Team, 2014) environment and under active development. If you use it, please support the project by citing it in publications:

> Gibb, S. and Strimmer, K. (2012). MALDIquant: a versatile R package for the analysis of mass spectrometry data. *Bioinformatics*, 28(17):2270–2271

If you have any questions, bugs, or suggestions do not hesitate to contact me (`mail@sebastiangibb.de`).
Please visit `http://strimmerlab.org/software/maldiquant/`.

1 Introduction

`MALDIquant` should be device and platform independent. That's why it has not any import/export functions.
`MALDIquantForeign` fills this gap and provides import/export routines for various file formats:

```
> supportedFileFormats()
```

```
$import
 [1] "txt"       "tab"       "csv"       "fid"       "ciphergen" "mzxml"
 [7] "mzml"      "imzml"     "analyze"   "cdf"       "msd"

$export
[1] "tab"   "csv"   "msd"   "mzml"  "imzml"
```

2 Setup

After starting `R` we could install `MALDIquant` and `MALDIquantForeign` directly from `CRAN` using `install.packages`:

```
> install.packages(c("MALDIquant", "MALDIquantForeign"))
```

Before we can use `MALDIquant` and `MALDIquantForeign` we have to load the packages.

```
> library("MALDIquant")
> library("MALDIquantForeign")
```

3 Import

`MALDIquantForeign` provides an `import` function that tries to auto-detect the correct file type. Because this would never be perfect `MALDIquantForeign` offers also many `import*` functions like `importBrukerFlex`, `importMzMl`, etc. Please see the manual page of `import` for a complete list (`?import`).

First we try to import some example data in Bruker Daltonics *flex-series file format using the `import` function.

```
> ## get the example directory
> exampleDirectory <- system.file("exampledata",
+                                 package="MALDIquantForeign")
>
```

```
> spectra <- import(file.path(exampleDirectory,
+                             "brukerflex"),
+                   verbose=FALSE)
> spectra[[1]]

S4 class type            : MassSpectrum
Number of m/z values     : 5
Range of m/z values      : 226.762 - 230.51
Range of intensity values: 1e+00 - 5e+00
Memory usage             : 8.719 KiB
Name                     : brukerflex.
File                     : /tmp/RtmpVoJHFZ/Rinst2cd45b46cf50/MALDIquantForeign/exa
```

Next we use the `importBrukerFlex` function (the result is the same as above).

```
> spectra <- importBrukerFlex(file.path(exampleDirectory,
+                                       "brukerflex"),
+                             verbose=FALSE)
> spectra[[1]]

S4 class type            : MassSpectrum
Number of m/z values     : 5
Range of m/z values      : 226.762 - 230.51
Range of intensity values: 1e+00 - 5e+00
Memory usage             : 8.719 KiB
Name                     : brukerflex.
File                     : /tmp/RtmpVoJHFZ/Rinst2cd45b46cf50/MALDIquantForeign/exa
```

`MALDIquantForeign` supports compressed files, too (*zip*, *tar.*{*bz2*, *gz*,*xz*}).

```
> spectra <- importCsv(file.path(exampleDirectory, "compressed",
+                                "csv.tar.gz"), verbose=FALSE)
> spectra[[1]]

S4 class type            : MassSpectrum
Number of m/z values     : 5
```

```
Range of m/z values      : 1 - 5
Range of intensity values: 6 - 10
Memory usage             : 1.492 KiB
File                     : /tmp/Rtmpb9xThK/MALDIquantForeign_uncompress/csv_2d1339

> spectra <- importCsv(file.path(exampleDirectory, "compressed",
+                                "csv.zip"), verbose=FALSE)
> spectra[[1]]

S4 class type            : MassSpectrum
Number of m/z values     : 5
Range of m/z values      : 1 - 5
Range of intensity values: 6 - 10
Memory usage             : 1.492 KiB
File                     : /tmp/Rtmpb9xThK/MALDIquantForeign_uncompress/csv_2d1341
```

Remote files are supported as well. Data are taken from Tan et al. (2006).

```
> spectra <- import(paste0("http://www.meb.ki.se/",
+                          "~yudpaw/papers/spikein_xml.zip"),
+                   centroided=FALSE, verbose=TRUE)
```

If you want to read peak lists (centroided data) instead of spectra data you have to set `centroided=TRUE`.

```
> peaks <- import(file.path(exampleDirectory, "ascii.txt"),
+                 centroided=TRUE, verbose=FALSE)
> peaks

[[1]]
S4 class type            : MassPeaks
Number of m/z values     : 5
Range of m/z values      : 1 - 5
Range of intensity values: 6 - 10
Range of snr values      : NA - NA
Memory usage             : 1.695 KiB
File                     : /tmp/RtmpVoJHFZ/Rinst2cd45b46cf50/MALDIquantForeign/exa
```

4 Export

The export routines in `MALDIquantForeign` are very similar to the import routines. Please see manual page of `export` for a complete list of supported export routines (`?export`).

First we create a simple list of `MassSpectrum` objects using `createMassSpectrum`.

```
> spectra <- list(
+   createMassSpectrum(mass=1:5, intensity=1:5),
+   createMassSpectrum(mass=1:5, intensity=6:10))
```

Now we want to export the first spectrum into a CSV file.

```
> export(spectra[[1]], file="spectrum1.csv")
> import("spectrum1.csv")

[[1]]
S4 class type            : MassSpectrum
Number of m/z values     : 5
Range of m/z values      : 1 - 5
Range of intensity values: 1 - 5
Memory usage             : 1.492 KiB
File                     : /tmp/RtmpVoJHFZ/Rbuild2cd4724df9ad/MALDIquantForeign/vi
```

Exporting every file by hand is cumbersome. We want to export the whole list of spectra. Instead of `file` we use `path` now to specify a directory. Please note that we have to add the file type/format information now (we can use the `type` argument or the corresponding `export*` function). If the path doesn't exists we will get an error. To force `export` to create/overwrite the given path, we set the argument `force=TRUE`.

```
> export(spectra, type="csv", path="spectra", force=TRUE)
> list.files("spectra")

[1] "1.csv" "2.csv"
```

5 Analyse Mass Spectrometry Data

Please have a look at the corresponding vignette shipped with `MALDIquant` and the `MALDIquant` website: `http://strimmerlab.org/software/maldiquant/`.

```
> vignette(topic="MALDIquant", package="MALDIquant")
```

6 Session Information

- R version 4.1.2 (2021-11-01), `x86_64-pc-linux-gnu`
- Running under: `Debian GNU/Linux 11 (bullseye)`
- Matrix products: default
- BLAS: `/usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0`
- LAPACK: `/usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: MALDIquant 1.20, MALDIquantForeign 0.13, knitr 1.37
- Loaded via a namespace (and not attached): XML 3.99-0.8, base64enc 0.1-3, compiler 4.1.2, digest 0.6.29, evaluate 0.14, highr 0.9, magrittr 2.0.1, parallel 4.1.2, readBrukerFlexData 1.8.5, readMzXmlData 2.8.1, stringi 1.7.6, stringr 1.4.0, tools 4.1.2, xfun 0.29

References

Gibb, S. and Strimmer, K. (2012). MALDIquant: a versatile R package for the analysis of mass spectrometry data. *Bioinformatics*, 28(17):2270–2271.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Tan, C. S., Ploner, A., Quandt, A., Lehtiö, J., and Pawitan, Y. (2006). Finding regions of significance in SELDI measurements for identifying protein biomarkers. *Bioinformatics*, 22(12):1515–1523.