Using R and Bioconductor for Proteomics Data Analysis

Laurent Gatto[1*] and Sebastian Gibb[2]

[1]Computational Proteomics Unit, University of Cambridge, UK
[2]Institute for Medical Informatics, Statistics and Epidemiology, University of Leipzig, Germany

July 24, 2014

This vignette shows and executes the code presented in the manuscript *Using R for proteomics data analysis*. It also aims at being a general overview for users who wish to explore the R environment and programming language for the analysis of proteomics data.

Keywords: proteomics, mass spectrometry, tutorial.

*lg390@cam.ac.uk

Contents

1 Introduction

This document illustrates some existing `R` infrastructure for the analysis of proteomics data. It presents the code for the use cases taken from [8]. A pre-print of the manuscript is available on arXiv[1].

There are however numerous additional `R` resources distributed by the Bioconductor[2] and CRAN[3] repositories, as well as packages hosted on personal websites. Section 7 on page 54 tries to provide a wider picture of available packages, without going into details.

1.1 General `R` resources

The reader is expected to have basic `R` knowledge to find the document helpful. There are numerous `R` introductions freely available, some of which are listed below.

From the `R` project web-page:

- **An Introduction to R** is based on the former *Notes on R*, gives an introduction to the language and how to use R for doing statistical analysis and graphics. [browse HTML — download PDF]
- Several introductory tutorials in the contributed documentation section.
- The `TeachingMaterial` repository[4] contains several sets of slides and vignettes about `R` programming.

Relevant background on the `R` software and its application to computational biology in general and proteomics in particular can also be found in [8]. For details about the Bioconductor project, the reader is referred to [10].

1.2 Getting help

All `R` packages come with ample documentation. Every command (function, class or method) a user is susceptible to use is documented. The documentation can be accessed by preceding the command by a `?` in the `R` console. For example, to obtain help about the `library` function, that will be used in the next section, one would type `?library`. In addition, all Bioconductor packages come with at least one vignette (this document is

[1] `http://arxiv.org/abs/1305.6559`
[2] `http://www.bioconductor.org`
[3] `http://cran.r-project.org/web/packages/`
[4] https://github.com/lgatto/TeachingMaterial

the vignette that comes with the **RforProteomics** package), a document that combines text and R code that is executed before the pdf is assembled. To look up all vignettes that come with a package, say **RforProteomics** and then open the vignette of interest, one uses the `vignette` function as illustrated below. More details can be found in `?vignette`.

```
## list all the vignettes in the RforProteomics package
vignette(package = "RforProteomics")
## Open the vignette called RforProteomics
vignette("RforProteomics", package = "RforProteomics")
## or just
vignette("RforProteomics")
```

R has several mailing lists[5]. The most relevant here being the main `R-help` list, *for discussion about problem and solutions using R* . This one is for general R content and is not suitable for bioinformatics or proteomics questions. Bioconductor also offers several mailing lists[6] dedicated to bioinformatics matters and Bioconductor packages. The main `Bioconductor` list is the most relevant one. It is possible to post[7] questions without subscribing to the list. Finally, the dedicated **RforProteomics** Google group[8] welcomes questions/comments/annoucements related to R and mass-spectrometry/proteomics.

It is important to read and comply to the posting guides (here and here) to maximise the chances to obtain good responses. It is important to specify the software versions using the `sessionInfo()` functions (see an example output at the end of this document, on page 58). It the question involves some code, make sure to isolate the relevant portion and report it with your question, trying to make your code/example reproducible[9].

All lists have browsable archives.

1.3 Installation

The package should be installed using as described below:

[5] `http://www.r-project.org/mail.html`
[6] `http://bioconductor.org/help/mailing-list/`
[7] `http://bioconductor.org/help/mailing-list/mailform/`
[8] `https://groups.google.com/forum/#!forum/rbioc-sig-proteomics`
[9] `https://github.com/hadley/devtools/wiki/Reproducibility`

```r
## only first time you install Bioconductor packages
source("http://www.bioconductor.org/biocLite.R")
## else
library("BiocInstaller")
biocLite("RforProteomics")
```

To install all dependencies and reproduce the code in the vignette, replace the last line in the code chunk above with:)

```r
biocLite("RforProteomics", dependencies = TRUE)
```

Finally, the package can be loaded with

```r
library("RforProteomics")
```

See also the **RforProteomics** web page[10] for more information on installation.

1.4 External dependencies

Some packages used in the document depend on external libraries that need to be installed prior to the `R` packages:

mzR depends on the Common Data Format[11] (CDF) to CDF-based raw mass-spectrometry data. On Linux, the `libcdf` library is required. On Debian-based systems, for instance, one needs to install the `libnetcdf-dev` package.

IPPD (and others) depend on the **XML** package which requires the `libxml2` infrastructure on Linux. On Debian-based systems, one needs to install `libxml2-dev`.

biomaRt performs on-line requests using the `curl`[12] infrastructure. On Debian-based systems, you one needs to install `libcurl-dev` or `libcurl4-openssl-dev`.

[10]http://lgatto.github.io/RforProteomics/
[11]http://cdf.gsfc.nasa.gov/
[12]http://curl.haxx.se/

1.5 Obtaining the code

The code in this document describes all the examples presented in [8] and can be copy, pasted and executed. It is however more convenient to have it in a separate text file for better interaction with R (using ESS[13] for Emacs or RStudio[14] for instance) to easily modify and explore it. This can be achieved with the `Stangle` function. One needs the Sweave source of this document (a document combining the narration and the R code) and the `Stangle` then specifically extracts the code chunks and produces a clean R source file. If the package is installed, the following code chunk will create a `RforProteomics.R` file in your working directory containing all the annotated source code contained in this document.

```
## gets the vignette source
rnwfile <- system.file("doc/vigsrc/RforProteomics.Rnw",
                       package = "RforProteomics")
## produces the R file in the working directory
library("knitr")
purl(rnwfile, quiet = TRUE)

## NULL
```

Alternatively, you can obtain the Rnw file on the github page https://github.com/lgatto/RforProteomics/blob/master/inst/doc/vigsrc/RforProteomics.Rnw.

1.6 Prepare the working environment

The packages that we will depend on to execute the examples will be loaded in the respective sections. Here, we pre-load packages that provide general functionality used throughout the document.

```
library("RColorBrewer") ## Color palettes
library("ggplot2")  ## Convenient and nice plotting
library("reshape2") ## Flexibly reshape data
```

[13] http://ess.r-project.org/
[14] http://rstudio.org/

2 Data standards and input/output

2.1 The mzR package

The **mzR** package [4] provides a unified interface to various mass spectrometry open formats. This code chunk, taken mainly from the `openMSfile` documentation illustrated how to open a connection to an raw data file. The example `mzML` data is taken from the **msdata** data package. The code below would also be applicable to an `mzXML`, `mzData` or `netCDF` file.

```r
## load the required packages
library("mzR") ## the software package
library("msdata") ## the data package
## below, we extract the releavant example file
## from the local 'msdata' installation
filepath <- system.file("microtofq", package = "msdata")
file <- list.files(filepath, pattern="MM14.mzML",
                   full.names=TRUE, recursive = TRUE)
## creates a commection to the mzML file
mz <- openMSfile(file)
## demonstraction of data access
basename(fileName(mz))

## [1] "MM14.mzML"

isInitialized(mz)

## [1] TRUE

runInfo(mz)

## $scanCount
## [1] 112
##
## $lowMz
## [1] 0
##
## $highMz
## [1] 0
##
## $dStartTime
## [1] 0
##
## $dEndTime
## [1] 0
##
## $msLevels
## [1] 1
```

```
instrumentInfo(mz)
```

```
## $manufacturer
## [1] "Unknown"
##
## $model
## [1] "instrument model"
##
## $ionisation
## [1] "electrospray ionization"
##
## $analyzer
## [1] "mass analyzer type"
##
## $detector
## [1] "detector type"
```

```
## once finished, it is good to explicitely
## close the connection
close(mz)
```

mzR is used by other packages, like **MSnbase** [9], **TargetSearch** [6] and **xcms** [12, 1, 13], that provide a higher level abstraction to the data.

2.2 Handling MS^2 identification data with mzID

The **mzID** package allows to load and manipulate MS^2 data in the `mzIdentML` format. The main `mzID` function reads such a file and constructs an instance of class `mzID`.

```
library("mzID")
id <- mzID("http://psi-pi.googlecode.com/svn/trunk/examples/1_1examples/55merge_tande

## reading 55merge_tandem.mzid... DONE!

id

## An mzID object
##
## Software used:   X\!Tandem (version: x! tandem CYCLONE (2010.06.01.5))
##
## Rawfile:         D:/TestSpace/NeoTestMarch2011/55merge.mgf
##
## Database:        D:/Software/Databases/Neospora_3rndTryp/Neo_rndTryp_3times.fasta.
##
## Number of scans: 169
## Number of PSM's: 170
```

Peptides, scans, parameters, ... can be extracted with the respective `pepetides`, `scans`, `parameters`, ... functions. The `mzID` object can also be converted into a `data.frame` using the `flatten` function.

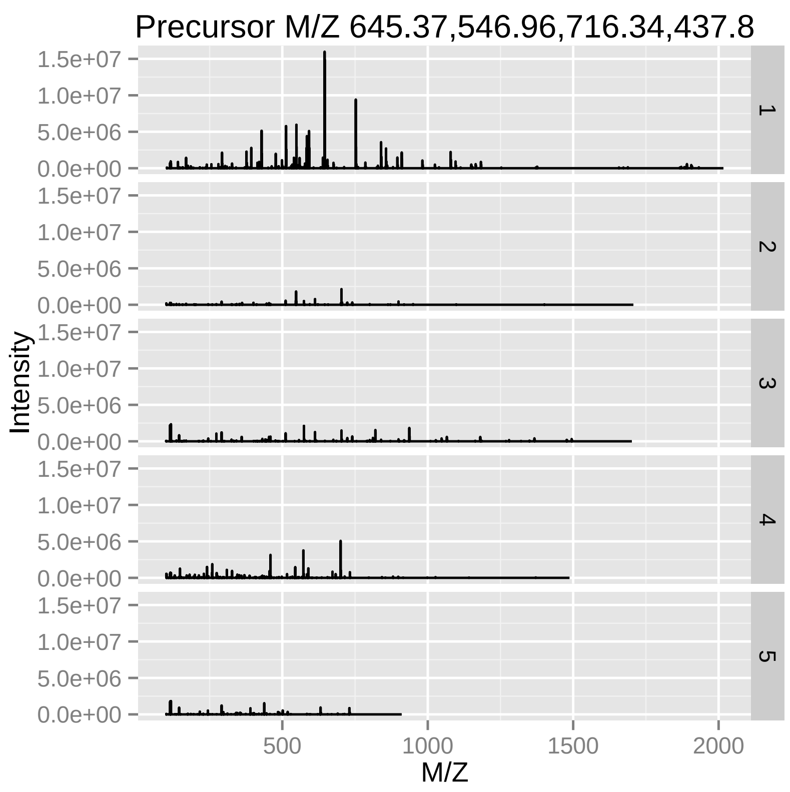
```
fid <- flatten(id)
names(fid)

##  [1] "spectrumid"                 "acquisitionnum"
##  [3] "passthreshold"              "rank"
##  [5] "calculatedmasstocharge"     "experimentalmasstocharge"
##  [7] "chargestate"                "x\\!tandem:expect"
##  [9] "x\\!tandem:hyperscore"      "pepseq"
## [11] "modified"                   "modification"
## [13] "isdecoy"                    "post"
## [15] "pre"                        "end"
## [17] "start"                    "accession"
## [19] "length"                   "sequence"
## [21] "databaseFile"

dim(fid)

## [1] 171  21
```

3 Raw data abstraction with `MSnExp` objects

MSnbase [9] provides base functions and classes for MS-based proteomics that allow facile data and meta-data processing, manipulation and plotting (see for instance figure 1 on page 12).

```
library("MSnbase")
## uses a simple dummy test included in the package
mzXML <- dir(system.file(package="MSnbase",dir="extdata"),
             full.name=TRUE,
             pattern="mzXML$")
basename(mzXML)

## [1] "dummyiTRAQ.mzXML"

## reads the raw data into and MSnExp instance
raw <- readMSData(mzXML, verbose = FALSE)
raw

## Object of class "MSnExp"
##  Object size in memory: 0.2 Mb
## - - - Spectra data - - -
##  MS level(s): 2
##  Number of MS1 acquisitions: 1
##  Number of MSn scans: 5
##  Number of precursor ions: 5
##  4 unique MZs
##  Precursor MZ's: 437.8 - 716.34
##  MSn M/Z range: 100 2017
##  MSn retention times: 25:1 - 25:2 minutes
## - - - Processing information - - -
## Data loaded: Thu Jul 24 16:32:04 2014
##  MSnbase version: 1.12.1
## - - - Meta data  - - -
## phenoData
##   rowNames: 1
##   varLabels: sampleNames
##   varMetadata: labelDescription
## Loaded from:
##   dummyiTRAQ.mzXML
## protocolData: none
## featureData
##   featureNames: X1.1 X2.1 ... X5.1 (5 total)
##   fvarLabels: spectrum
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'

## Extract a single spectrum
raw[[3]]

## Object of class "Spectrum2"
##  Precursor: 645.4
##  Retention time: 25:2
##  Charge: 2
##  MSn level: 2
##  Peaks count: 2125
##  Total ion count: 150838188
```

```
plot(raw, full=TRUE)
```



```
plot(raw[[3]], full=TRUE, reporters=iTRAQ4)
```

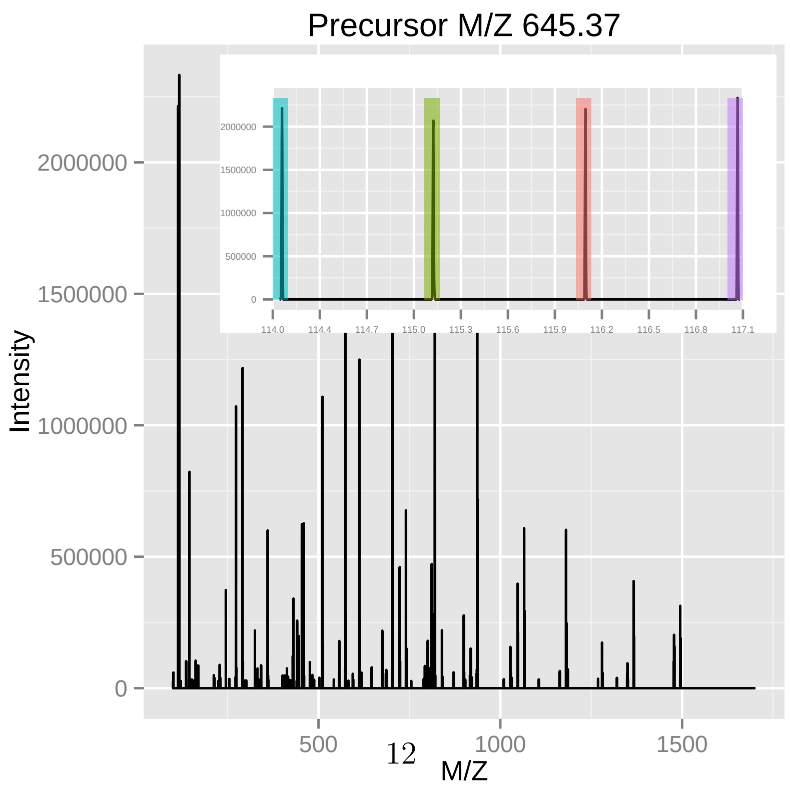


Figure 1: The `plot` method can be used on experiments, i.e. spectrum collections (left), or individual spectra (right).

3.1 `mgf` read/write support

Read and write support for data in the `mgf`[15] and `mzTab`[16] formats are available via the `readMgfData`/`writeMgfData` and `readMzTabData`/`writeMzTabData` functions, respectively. An example for the latter is shown in the next section.

4 Quantitative proteomics

As an running example throughout this document, we will use a TMT 6-plex data set, `PXD000001` to illustrate quantitative data processing. The code chunk below first downloads this data file from the ProteomeXchange server using the **rpx** package.

4.1 The `mzTab` format

The first code chunk downloads the `mzTab` data from the ProteomeXchange repository [16].

```
## Experiment information
library("rpx")
px1 <- PXDataset("PXD000001")
px1

## Object of class "PXDataset"
##  Id: PXD000001 with 8 files
##  [1] 'F063721.dat' ... [8] 'erwinia_carotovora.fasta'
##  Use 'pxfiles(.)' to see all files.

pxfiles(px1)

## [1] "F063721.dat"
## [2] "F063721.dat-mztab.txt"
## [3] "PRIDE_Exp_Complete_Ac_22134.xml.gz"
## [4] "PRIDE_Exp_mzData_Ac_22134.xml.gz"
## [5] "PXD000001_mztab.txt"
## [6] "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01.mzXML"
```

[15] http://www.matrixscience.com/help/data_file_help.html#GEN
[16] https://code.google.com/p/mztab/

```
## [7] "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01.raw"
## [8] "erwinia_carotovora.fasta"

## Downloading the mzTab data
mztab <- pxget(px1, "PXD000001_mztab.txt")

## Downloading 1 file
## PXD000001_mztab.txt already present.

mztab

## [1] "PXD000001_mztab.txt"
```

The code below loads the mzTab file into R and generates an MSnSet instance, removes missing values and calculates protein intensities by summing the peptide quantitation data. Figure 2 illustrates the intensities for 5 proteins.

```
## Load mzTab peptide data
qnt <- readMzTabData(mztab, what = "PEP")

## Warning:  Support for mzTab version 0.9 only.  Support will be added soon.
## Detected a metadata section
## Detected a peptide section
## Warning:  NAs introduced by coercion

sampleNames(qnt) <- reporterNames(TMT6)
head(exprs(qnt))

##   TMT6.126 TMT6.127 TMT6.128 TMT6.129 TMT6.130 TMT6.131
## 1       NA       NA       NA       NA       NA       NA
## 2 10630132 11238708 12424917 10997763  9928972 10398534
## 3       NA       NA       NA       NA       NA       NA
## 4       NA       NA       NA       NA       NA       NA
## 5 11105690 12403253 13160903 12229367 11061660 10131218
## 6  1183431  1322371  1599088  1243715  1306602  1159064

## remove missing values
qnt <- filterNA(qnt)
processingData(qnt)
```

```
## - - - Processing information - - -
## mzTab read: Thu Jul 24 16:32:15 2014
## Subset [2351,6][1504,6] Thu Jul 24 16:32:15 2014
## Removed features with more than 0 NAs: Thu Jul 24 16:32:15 2014
## Dropped featureData's levels Thu Jul 24 16:32:15 2014
##  MSnbase version: 1.12.1

## combine into proteins
## - using the 'accession' feature meta data
## - sum the peptide intensities
protqnt <- combineFeatures(qnt,
                           groupBy = fData(qnt)$accession,
                           fun = sum)

## Combined 1504 features into 399 using user-defined function
```

```
qntS <- normalise(qnt, "sum")
qntV <- normalise(qntS, "vsn")
qntV2 <- normalise(qnt, "vsn")

acc <- c("P00489", "P00924",
         "P02769", "P62894",
         "ECA")

idx <- sapply(acc, grep, fData(qnt)$accession)
idx2 <- sapply(idx, head, 3)
small <- qntS[unlist(idx2), ]

idx3 <- sapply(idx, head, 10)
medium <- qntV[unlist(idx3), ]

m <- exprs(medium)
colnames(m) <- c("126", "127", "128",
                 "129", "130", "131")
rownames(m) <- fData(medium)$accession
```

```
cls <- brewer.pal(5, "Set1")
matplot(t(tail(exprs(protqnt), n = 5)), type = "b",
        lty = 1, col = cls,
        ylab = "Protein intensity (summed peptides)",
        xlab = "TMT reporters")
legend("topright", tail(featureNames(protqnt), n=5),
       lty = 1, bty = "n", cex = .8, col = cls)
```

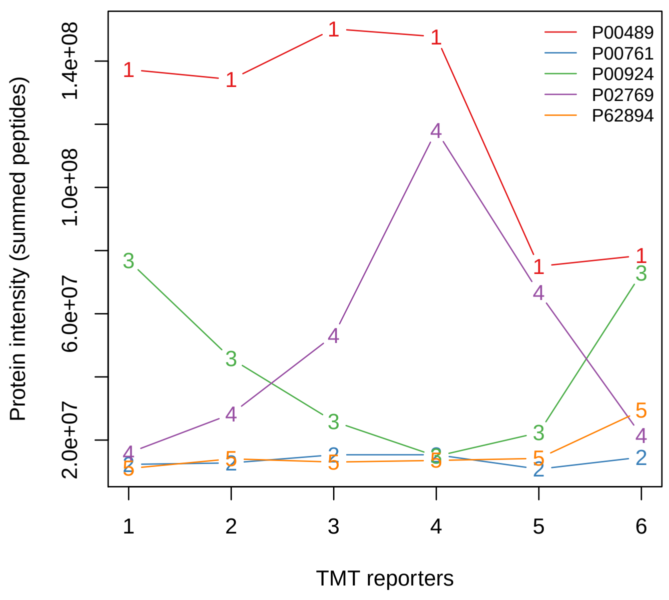


Figure 2: Protein quantitation data.

```
rownames(m)[grep("CYC", rownames(m))] <- "CYT"
rownames(m)[grep("ENO", rownames(m))] <- "ENO"
rownames(m)[grep("ALB", rownames(m))] <- "BSA"
rownames(m)[grep("PYGM", rownames(m))] <- "PHO"
rownames(m)[grep("ECA", rownames(m))] <- "Background"

cls <- c(brewer.pal(length(unique(rownames(m)))-1, "Set1"),
         "grey")
names(cls) <- unique(rownames(m))
wbcol <- colorRampPalette(c("white", "darkblue"))(256)
```

```
heatmap(m, col = wbcol, RowSideColors=cls[rownames(m)])
```

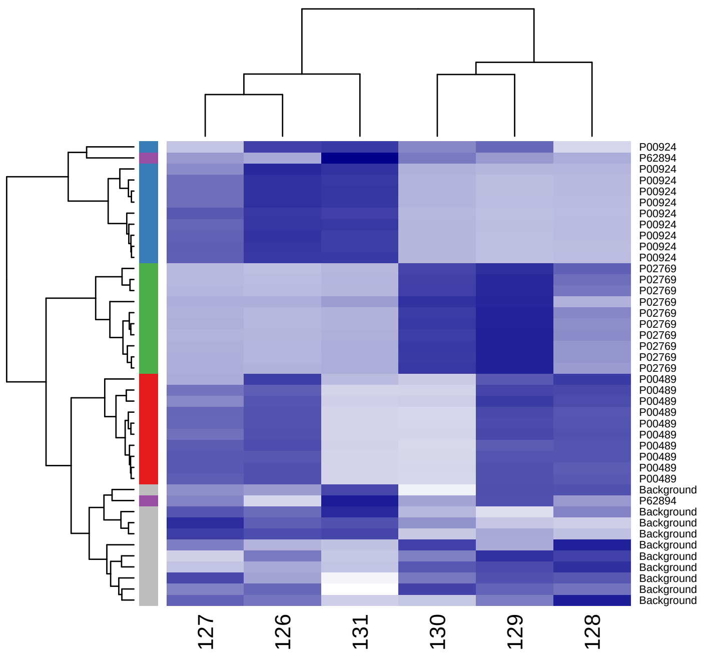


Figure 3: A heatmap.

```
dfr <- data.frame(exprs(small),
                  Protein = as.character(fData(small)$accession),
                  Feature = featureNames(small),
                  stringsAsFactors = FALSE)
colnames(dfr) <- c("126", "127", "128", "129", "130", "131",
                   "Protein", "Feature")
dfr$Protein[dfr$Protein == "sp|P00924|ENO1_YEAST"] <- "ENO"
dfr$Protein[dfr$Protein == "sp|P62894|CYC_BOVIN"]  <- "CYT"
dfr$Protein[dfr$Protein == "sp|P02769|ALBU_BOVIN"] <- "BSA"
dfr$Protein[dfr$Protein == "sp|P00489|PYGM_RABIT"] <- "PHO"
dfr$Protein[grep("ECA", dfr$Protein)] <- "Background"
dfr2 <- melt(dfr)

## Using Protein, Feature as id variables

ggplot(aes(x = variable, y = value, colour = Protein),
       data = dfr2) +
  geom_point() +
  geom_line(aes(group=as.factor(Feature)), alpha = 0.5) +
  facet_grid(. ~ Protein) + theme(legend.position="none") +
  labs(x = "Reporters", y = "Normalised intensity")
```

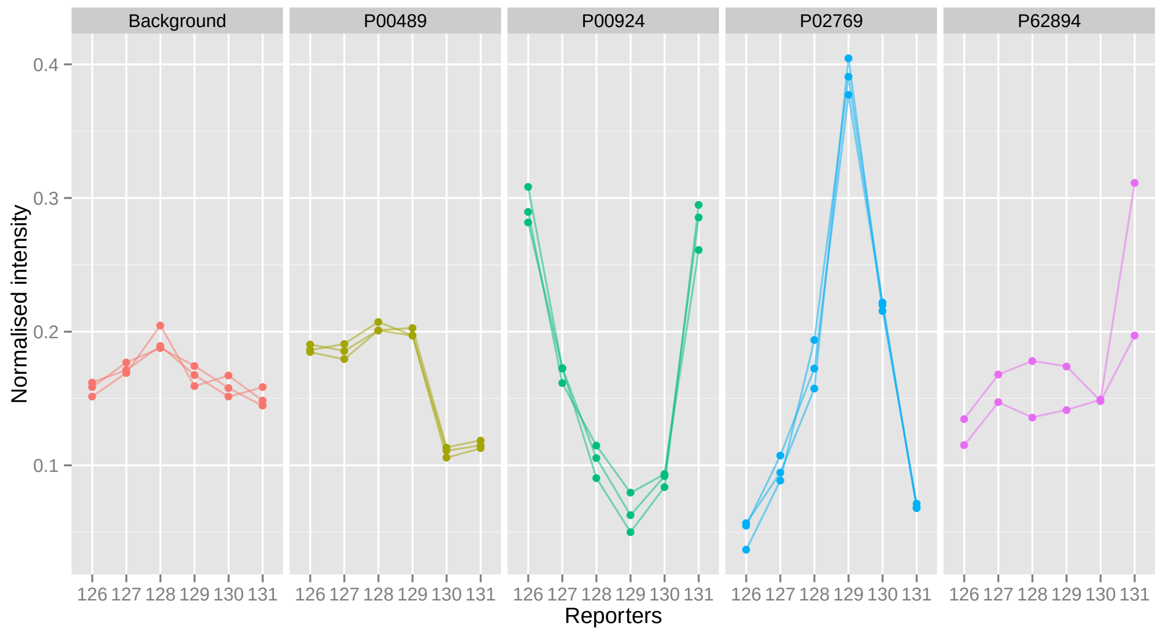


Figure 4: Spikes plot using **ggplot2**.

4.2 Working with raw data

We reuse our dedicated `px1` ProteomeXchange data object to download the raw data (in `mzXML` format) and load it with the `readMSData` from the **MSnbase** package that produces a raw data experiment object of class `MSnExp`. The raw data is then quantified using the `quantify` method specifying the TMT 6-plex isobaric tags and a 7th peak of interest corresponding to the un-dissociated reporter tag peaks (see the `MSnbase-demo` vignette in **MSnbase** for details).

```
mzxml <- pxget(px1, "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01.mzXML")

## Downloading 1 file

rawms <- readMSData(mzxml, centroided = TRUE, verbose = FALSE)
qntms <- quantify(rawms, reporters = TMT7, method = "max")

##
  |
  |                                                                 |   0%
  |
  |                                                                 |   1%
  |
  |=                                                                |   1%
  |
  |=                                                                |   2%
  |
  |=                                                                |   3%
  |
  |==                                                               |   3%
  |
  |==                                                               |   4%
  |
  |==                                                               |   5%
  |
  |===                                                              |   5%
  |
  |===                                                              |   6%
  |
  |===                                                                   |   7%
  |
  |====                                                                  |   7%
  |
  |====                                                                  |   8%
  |
  |====                                                                  |   9%
  |
  |=====                                                                 |   9%
  |
  |=====                                                                 |  10%
  |
  |=====                                                                 |  11%
  |
  |======                                                                |  11%
  |
  |======                                                                |  12%
  |
  |======                                                                |  13%
  |
  |=======                                                               |  13%
  |
  |=======                                                               |  14%
  |
  |=======                                                               |  15%
  |
  |========                                                              |  15%
  |
  |========                                                              |  16%
  |
  |========                                                              |  17%
  |
  |=========                                                             |  17%
  |
  |=========                                          |  18%
  |
  |=========                                          |  19%
  |
  |==========                                         |  19%
  |
  |==========                                         |  20%
  |
  |==========                                         |  21%
  |
  |===========                                        |  21%
  |
  |===========                                        |  22%
  |
  |===========                                        |  23%
  |
  |============                                       |  23%
  |
  |============                                       |  24%
  |
  |============                                       |  25%
  |
  |=============                                      |  25%
  |
  |=============                                      |  26%
  |
  |=============                                      |  27%
  |
  |==============                                     |  27%
  |
  |==============                                     |  28%
  |
  |==============                                     |  29%
  |
  |===============                                    |  29%
  |
  |===============                                   |  30%
  |
  |===============                                   |  31%
  |
  |================                                  |  31%
  |
  |================                                  |  32%
  |
  |================                                  |  33%
  |
  |=================                                 |  33%
  |
  |=================                                 |  34%
  |
  |=================                                 |  35%
  |
  |==================                                |  35%
  |
  |==================                                |  36%
  |
  |==================                                |  37%
  |
  |===================                               |  37%
  |
  |===================                               |  38%
  |
  |===================                               |  39%
  |
  |====================                              |  39%
  |
  |====================                              |  40%
  |
  |====================                              |  41%
  |
  |=====================                             |  41%
  |
  |=====================                             |  42%
  |
  |=====================                             |  43%
  |
  |======================                            |  43%
  |
  |======================                            |  44%
  |
  |======================                            |  45%
  |
  |=======================                           |  45%
  |
  |=======================                           |  46%
  |
  |=======================                           |  47%
  |
  |========================                          |  47%
  |
  |========================                          |  48%
  |
  |========================                          |  49%
  |
  |=========================                         |  49%
  |
  |=========================                         |  50%
  |
  |=========================                         |  51%
  |
  |==========================                        |  51%
  |
  |==========================                        |  52%
  |
  |==========================                        |  53%
  |
  |===========================                       |  53%
  |
  |===========================                       |  54%
  |
  |===========================                       |  55%
  |
  |============================                      |  55%
  |
  |============================                      |  56%
  |
  |============================                      |  57%
  |
  |=============================                     |  57%
  |
  |=============================                     |  58%
  |
  |=============================                     |  59%
  |
  |==============================                    |  59%
  |
  |==============================                    |  60%
  |
  |==============================                    |  61%
  |
  |===============================                   |  61%
  |
  |===============================                   |  62%
  |
  |===============================                   |  63%
  |
  |================================                  |  63%
  |
  |================================                  |  64%
  |
  |================================                  |  65%
  |
  |=================================                 |  65%
  |
  |=================================                 |  66%
  |
  |=================================                 |  67%
  |
  |==================================                |  67%
  |
  |==================================                |  68%
  |
  |==================================                |  69%
  |
  |===================================               |  69%
  |
  |===================================               |  70%
  |
  |===================================               |  71%
  |
  |====================================              |  71%
  |
  |====================================              |  72%
  |
  |====================================              |  73%
  |
  |=====================================             |  73%
  |
  |=====================================             |  74%
  |
  |=====================================             |  75%
  |
  |======================================            |  75%
  |
  |======================================            |  76%
```

```
|
|======================================             |  77%
|
|=======================================            |  77%
|
|=======================================            |  78%
|
|=======================================            |  79%
|
|========================================           |  79%
|
|========================================           |  80%
|
|========================================           |  81%
|
|=========================================          |  81%
|
|=========================================          |  82%
|
|=========================================          |  83%
|
|==========================================         |  83%
|
|==========================================         |  84%
|
|==========================================         |  85%
|
|===========================================        |  85%
|
|===========================================        |  86%
|
|===========================================        |  87%
|
|============================================       |  87%
|
  |=============================================      |  88%
  |
  |=============================================      |  89%
  |
  |==============================================     |  89%
  |
  |==============================================     |  90%
  |
  |==============================================     |  91%
  |
  |===============================================    |  91%
  |
  |===============================================    |  92%
  |
  |===============================================    |  93%
  |
  |================================================   |  93%
  |
  |================================================   |  94%
  |
  |================================================   |  95%
  |
  |=================================================  |  95%
  |
  |=================================================  |  96%
  |
  |=================================================  |  97%
  |
  |================================================== |  97%
  |
  |================================================== |  98%
  |
  |================================================== |  99%
  |
  |===================================================|  99%
  |
  |=================================================| 100%

## Preparing meta-data
## Original MSnExp and new MSnSet have different number of samples in phenoData.
Dropping original.
## Creating 'MSnSet' object

qntms

## MSnSet (storageMode: lockedEnvironment)
## assayData: 6103 features, 7 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: TMT7.126 TMT7.127 ... TMT7.230 (7
##     total)
##   varLabels: mz reporters
##   varMetadata: labelDescription
## featureData
##   featureNames: X1.1 X10.1 ... X999.1 (6103 total)
##   fvarLabels: spectrum file ... collision.energy (12
##     total)
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'
## Annotation: No annotation
## - - - Processing information - - -
## Data loaded: Thu Jul 24 14:41:40 2014
## TMT7 quantification by max: Thu Jul 24 15:20:24 2014
##  MSnbase version: 1.12.1
```

Identification data in the `mzIdentML` format can be added to `MSnExp` or `MSnSet` instances with the `addIdentificationData` function. See the function documentation for examples.

```
d <- data.frame(Signal = rowSums(exprs(qntms)[, 1:6]),
                Incomplete = exprs(qntms)[, 7])
d <- log(d)
cls <- rep("#00000050", nrow(qnt))
pch <- rep(1, nrow(qnt))
cls[grep("P02769", fData(qnt)$accession)] <- "gold4" ## BSA
cls[grep("P00924", fData(qnt)$accession)] <- "dodgerblue" ## ENO
cls[grep("P62894", fData(qnt)$accession)] <- "springgreen4" ## CYT
cls[grep("P00489", fData(qnt)$accession)] <- "darkorchid2" ## PHO
pch[grep("P02769", fData(qnt)$accession)] <- 19
pch[grep("P00924", fData(qnt)$accession)] <- 19
pch[grep("P62894", fData(qnt)$accession)] <- 19
pch[grep("P00489", fData(qnt)$accession)] <- 19
```

```
mzp <- plotMzDelta(rawms, reporters = TMT6, verbose = FALSE) + ggtitle("")
```

```
mzp
```

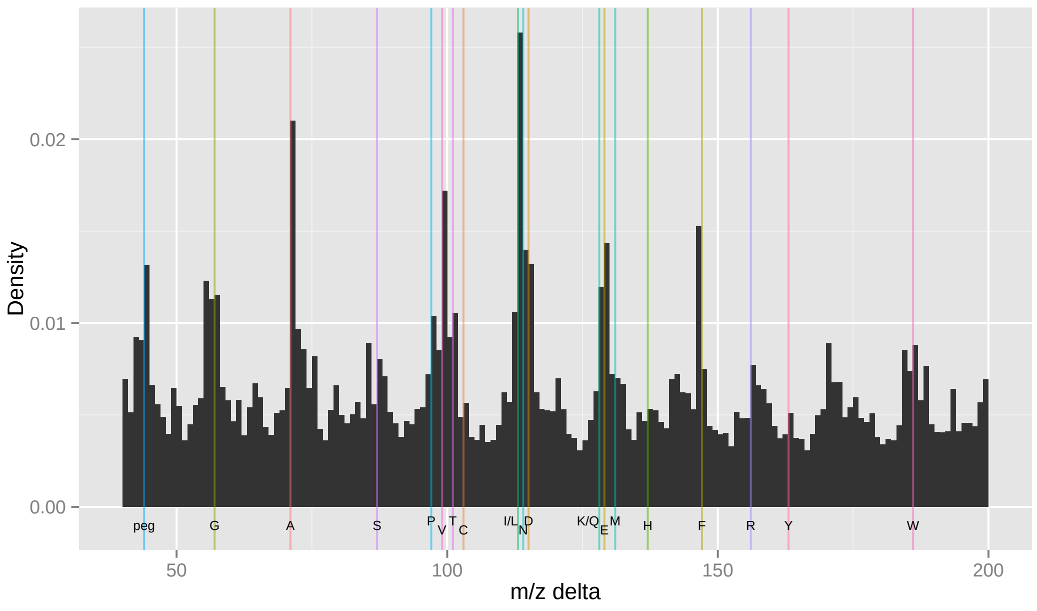


Figure 5: A m/z delta plot.

```
plot(Signal ~ Incomplete, data = d,
     xlab = expression(Incomplete~dissociation),
     ylab = expression(Sum~of~reporters~intensities),
     pch = 19,
     col = "#4582B380")
grid()
abline(0, 1, lty = "dotted")
abline(lm(Signal ~ Incomplete, data = d), col = "darkblue")
```

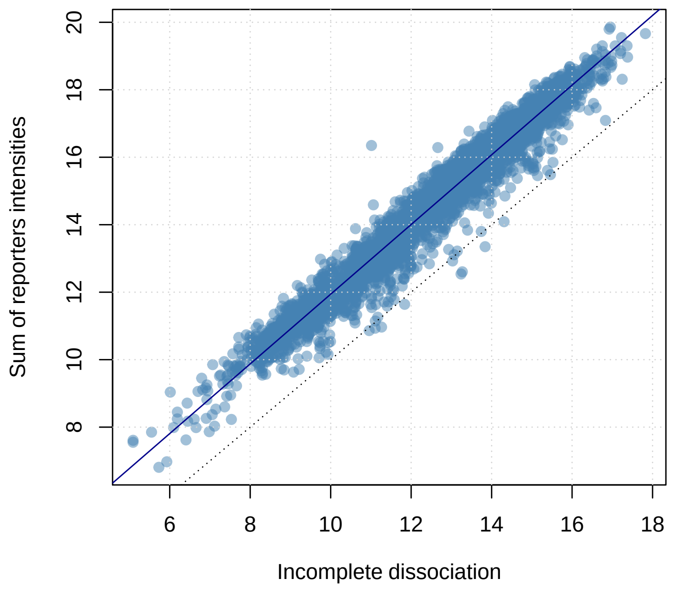


Figure 6: Incomplete dissociation.

```
MAplot(qnt[, c(4, 2)], cex = .9, col = cls, pch = pch, show.statistics = FALSE)
```

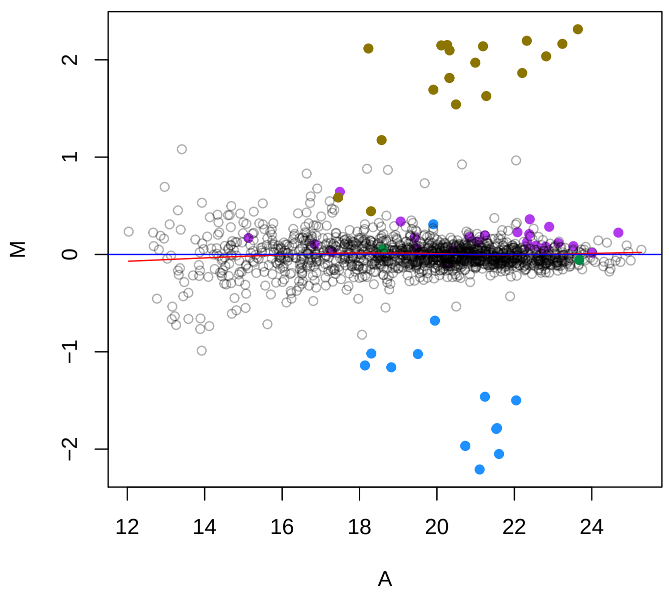


Figure 7: MAplot on an `MSnSet` instance.

4.3 The MALDIquant package

This section illustrates some of **MALDIquant**'s data processing capabilities [11]. The code is taken from the `processing-peaks.R` script downloaded from the package homepage[17].

Loading the data

```
## load packages
library("MALDIquant")
library("MALDIquantForeign")
## getting test data
datapath <-
  file.path(system.file("Examples",
                        package = "readBrukerFlexData"),
            "2010_05_19_Gibb_C8_A1")
dir(datapath)

## [1] "0_A1" "0_A2"

sA1 <- importBrukerFlex(datapath, verbose=FALSE)
# in the following we use only the first spectrum
s <- sA1[[1]]

summary(mass(s))

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1000    2370    4330    4720    6870   10000

summary(intensity(s))

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4     180    1560    2840    4660   32600

head(as.matrix(s))
```

[17]http://strimmerlab.org/software/maldiquant/

```
##         mass  intensity
## [1,]   999.9      11278
## [2,]  1000.1      11350
## [3,]  1000.3      10879
## [4,]  1000.5      10684
## [5,]  1000.7      10740
## [6,]  1000.9      10947
```

```
plot(s)
```

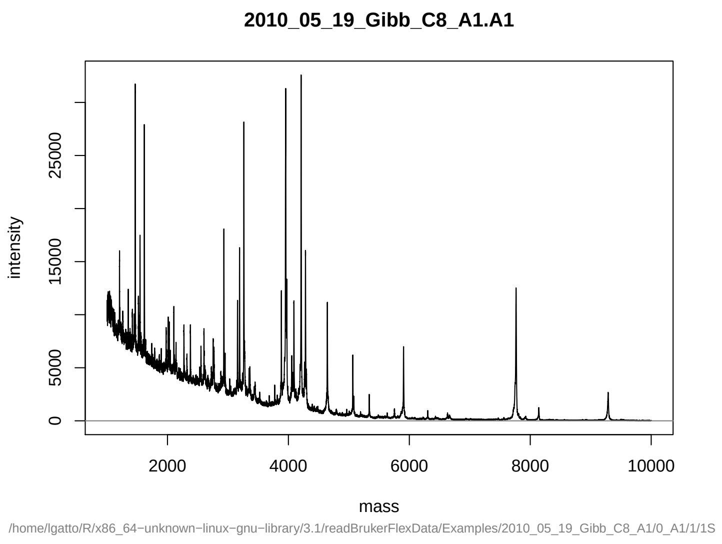


Figure 8: Spectrum plotting in **MALDIquant**.

Preprocessing

```
## sqrt transform (for variance stabilization)
s2 <- transformIntensity(s, method="sqrt")
s2

## S4 class type            : MassSpectrum
## Number of m/z values     : 22431
## Range of m/z values      : 999.939 - 10001.925
## Range of intensity values: 2e+00 - 1.805e+02
## Memory usage                : 359.875 KiB
## Name                        : 2010_05_19_Gibb_C8_A1.A1
## File                        : /home/lgatto/R/x86_64-unknown-linux-gnu-library/3.1/rea

## smoothing - 5 point moving average
s3 <- smoothIntensity(s2, method="MovingAverage", halfWindowSize=2)
s3

## S4 class type               : MassSpectrum
## Number of m/z values        : 22431
## Range of m/z values         : 999.939 - 10001.925
## Range of intensity values: 3.606e+00 - 1.792e+02
## Memory usage                : 359.875 KiB
## Name                        : 2010_05_19_Gibb_C8_A1.A1
## File                        : /home/lgatto/R/x86_64-unknown-linux-gnu-library/3.1/rea

## baseline subtraction
s4 <- removeBaseline(s3, method="SNIP")
s4

## S4 class type               : MassSpectrum
## Number of m/z values        : 22431
## Range of m/z values         : 999.939 - 10001.925
## Range of intensity values: 0e+00 - 1.404e+02
## Memory usage                : 359.875 KiB
## Name                        : 2010_05_19_Gibb_C8_A1.A1
## File                        : /home/lgatto/R/x86_64-unknown-linux-gnu-library/3.1/rea
```

Peak picking

```
## peak picking
p <- detectPeaks(s4)
length(p) # 181

## [1] 186

peak.data <- as.matrix(p) # extract peak information
```

```
par(mfrow=c(2,3))
xl <- range(mass(s))
# use same xlim on all plots for better comparison
plot(s, sub="", main="1: raw", xlim=xl)
plot(s2, sub="", main="2: variance stabilisation", xlim=xl)
plot(s3, sub="", main="3: smoothing", xlim=xl)
plot(s4, sub="", main="4: base line correction", xlim=xl)
plot(s4, sub="", main="5: peak detection", xlim=xl)
points(p)
top20 <- intensity(p) %in% sort(intensity(p), decreasing=TRUE)[1:20]
labelPeaks(p, index=top20, underline=TRUE)
plot(p, sub="", main="6: peak plot", xlim=xl)
labelPeaks(p, index=top20, underline=TRUE)
```

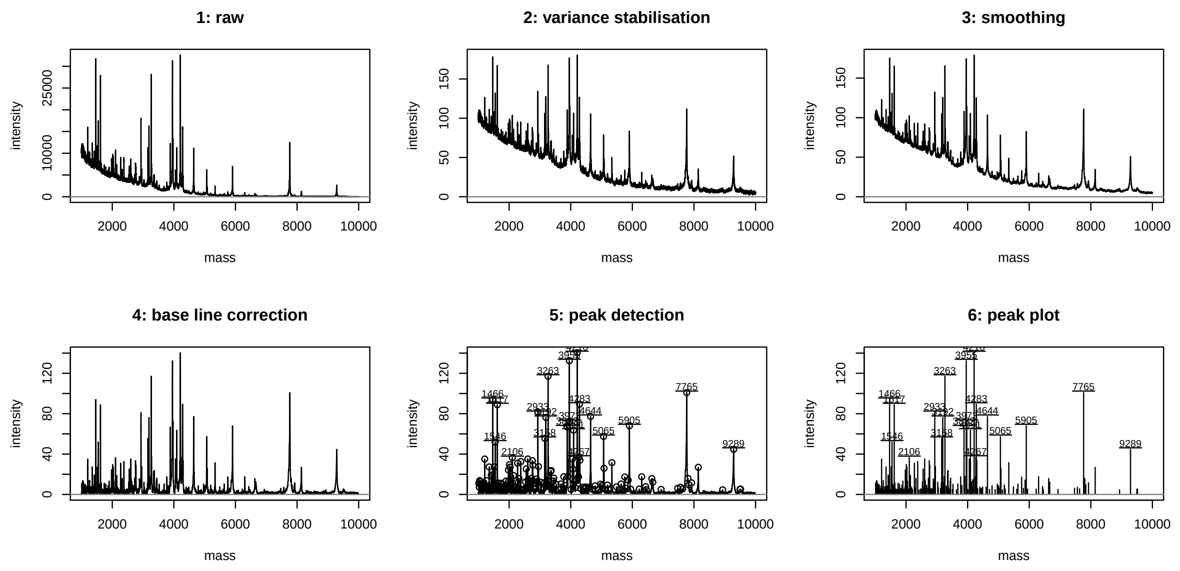


Figure 9: Spectrum plotting in **MALDIquant**.

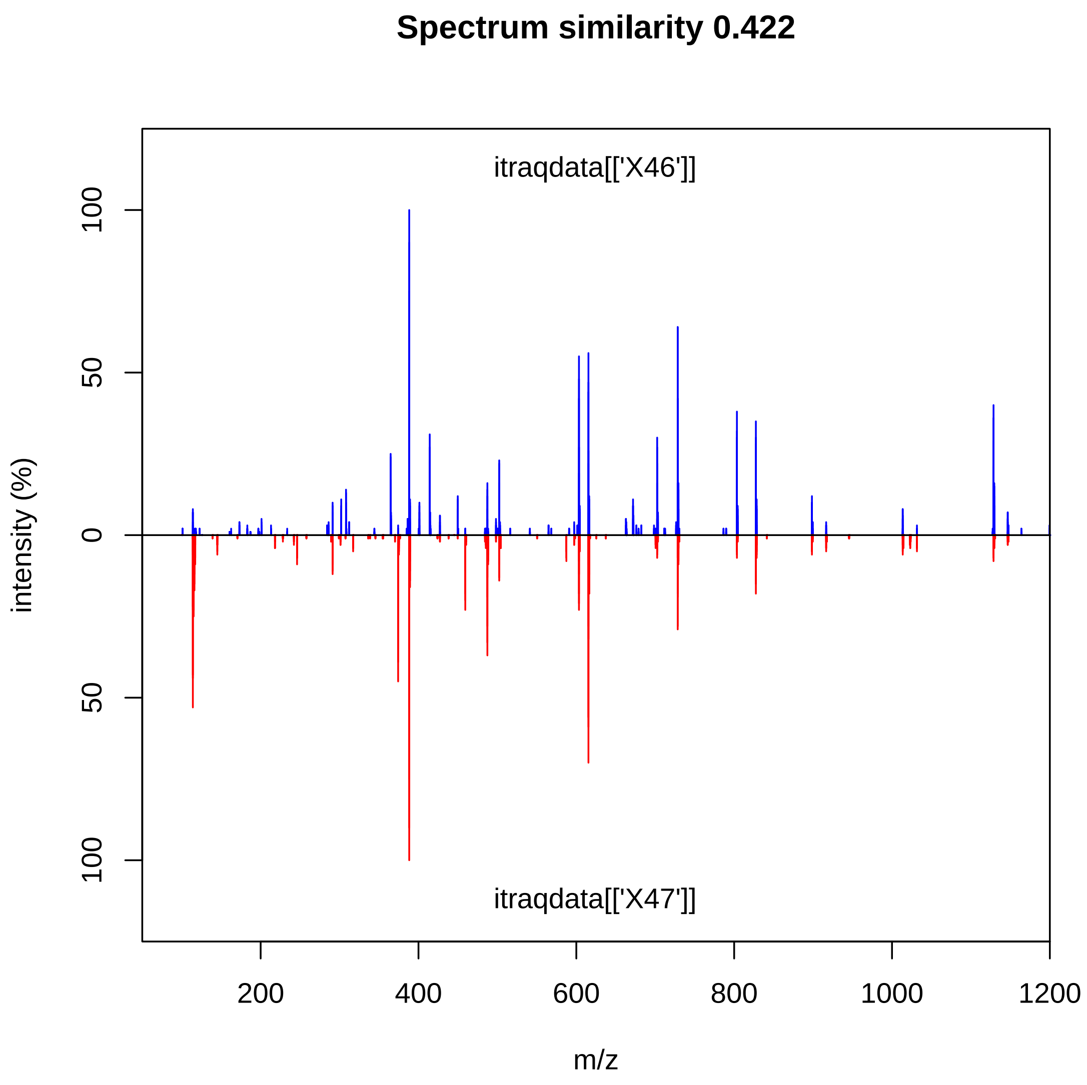
4.4 Working with peptide sequences

```
library(IPPD)
library(BRAIN)
atoms <- getAtomsFromSeq("SIVPSGASTGVHEALEMR")
unlist(atoms)

##   C   H   N   O   S
##  77 129  23  27   1

library(Rdisop)
pepmol <- getMolecule(paste0(names(atoms),
                             unlist(atoms),
                             collapse = ""))
pepmol

## $formula
## [1] "C77H129N23O27S"
##
## $score
## [1] 1
##
## $exactmass
## [1] 1840
##
## $charge
## [1] 0
##
## $parity
## [1] "e"
##
## $valid
## [1] "Valid"
##
## $DBE
## [1] 25
```

```
##
## $isotopes
## $isotopes[[1]]
##            [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 1839.9149 1840.9177 1841.9197 1.843e+03 1.844e+03
## [2,]    0.3427    0.3353    0.1961 8.474e-02 2.953e-02
##            [,6]      [,7]      [,8]      [,9]     [,10]
## [1,] 1.845e+03 1.846e+03 1.847e+03 1.848e+03 1.849e+03
## [2,] 8.692e-03 2.226e-03 5.066e-04 1.040e-04 1.950e-05
```

```
##
library(OrgMassSpecR)
data(itraqdata)

simplottest <-
  itraqdata[featureNames(itraqdata) %in% paste0("X", 46:47)]
sim <- SpectrumSimilarity(as(simplottest[[1]], "data.frame"),
                          as(simplottest[[2]], "data.frame"),
                          top.lab = "itraqdata[['X46']]",
                          bottom.lab = "itraqdata[['X47']]",
                          b = 25)
```

```
##        mz intensity.top intensity.bottom
## 1   114.1             0               44
## 2   114.1             0               53
## 3   114.1             0               43
## 4   115.1             0               25
## 5   364.7            25                0
## 6   374.2             0               39
## 7   374.2             0               45
## 8   374.2             0               35
## 9   388.2             0               35
## 10  388.3             0               75
## 11  388.3             0              100
## 12  388.3             0               90
## 13  388.3            35               53
## 14  388.3           100                 53
## 15  388.3            90                 53
## 16  388.3            53                 53
## 17  388.3            75                 53
## 18  414.3            31                  0
## 19  414.3            27                  0
## 20  487.3             0                 33
## 21  487.3             0                 37
## 22  487.3             0                 28
## 23  603.3            42                  0
## 24  603.4            55                  0
## 25  603.4            48                  0
## 26  603.4            27                  0
## 27  615.3             0                 28
## 28  615.3             0                 56
## 29  615.4             0                 70
## 30  615.4             0                 59
## 31  615.4            26                 32
## 32  615.4            44                 32
## 33  615.4            56                 32
## 34  615.4            47                 32
## 35  702.4            27                  0
## 36  702.4            30                  0
## 37  728.4             0                 28
## 38  728.5            64                 29
## 39  728.5            64                 29
## 40  728.5            42                 29
## 41  728.5            42                 29
## 42  803.4            30                  0
## 43  803.5            38                  0
## 44  803.5            32                  0
## 45  827.5            28                  0
## 46  827.5            35                  0
## 47  827.5            30                  0
## 48 1128.6            36                  0
## 49 1128.6             40                  0
## 50 1128.7             29                  0

title(main = paste("Spectrum similarity", round(sim, 3)))
```



```
MonoisotopicMass(formula = list(C = 2, O = 1, H=6))

## [1] 46.04

molecule <- getMolecule("C2H5OH")
molecule$exactmass
```

```
## [1] 46.04

## x11()
## plot(t(.pepmol£isotopes[[1]]), type = "h")

## x <- IsotopicDistribution(formula = list(C = 2, O = 1, H=6))
## t(molecule£isotopes[[1]])
## par(mfrow = c(2,1))
## plot(t(molecule£isotopes[[1]]), type = "h")
## plot(x[, c(1,3)], type = "h")

## data(myo500)
## masses <- c(147.053, 148.056)
## intensities <- c(93, 5.8)
## molecules <- decomposeIsotopes(masses, intensities)

## experimental eno peptides
exppep <-
  as.character(fData(qnt[grep("ENO", fData(qnt)[, 2]), ])[, 1]) ## 13
minlength <- min(nchar(exppep))

eno <- download.file("http://www.uniprot.org/uniprot/P00924.fasta",
                     destfile = "P00924.fasta")
eno <- paste(readLines("P00924.fasta")[-1], collapse = "")
enopep <- Digest(eno, missed = 1)
nrow(enopep) ## 103

## [1] 103

sum(nchar(enopep$peptide) >= minlength) ## 68

## [1] 0

pepcnt <- enopep[enopep[, 1] %in% exppep, ]
nrow(pepcnt) ## 13

## [1] 0
```

The following code chunks demonstrate how to use the **cleaver** package for in-silico cleavage of polypeptides, e.g. cleaving of *Gastric juice peptide 1 (P01358)* using *Trypsin*:

```
library(cleaver)
cleave("LAAGKVEDSD", enzym = "trypsin")

## $LAAGKVEDSD
## [1] "LAAGK" "VEDSD"
```

Sometimes cleavage is not perfect and the enzym miss some cleavage positions:

```
## miss one cleavage position
cleave("LAAGKVEDSD", enzym = "trypsin", missedCleavages = 1)

## $LAAGKVEDSD
## [1] "LAAGKVEDSD"

## miss zero or one cleavage positions
cleave("LAAGKVEDSD", enzym = "trypsin", missedCleavages = 0:1)

## $LAAGKVEDSD
## [1] "LAAGK"      "VEDSD"      "LAAGKVEDSD"
```

Example code to generate an Texshade image to be included directly in a Latex document or R vignette is presented below. The R code generates a Texshade environment and the annotated sequence display code that is written to a TEX file that can itself be included into a LATEX of Sweave document.

```
seq1file <- "seq1.tex"
cat("\\begin{texshade}{Figures/P00924.fasta}
     \\setsize{numbering}{footnotesize}
     \\setsize{residues}{footnotesize}
     \\residuesperline*{70}
     \\shadingmode{functional}
     \\hideconsensus
     \\vsepspace{1mm}
     \\hidenames
     \\noblockskip\n", file = seq1file)
tmp <- sapply(1:nrow(pepcnt), function(i) {
  col <- ifelse((i %% 2) == 0, "Blue", "RoyalBlue")
  cat("\\shaderegion{1}{", pepcnt$start[i], "..", pepcnt$stop[i], "}{White}{", col, "}\n",
      file = seq1file, append = TRUE)
})
cat("\\end{texshade}
    \\caption{Visualising observed peptides for the Yeast enolase protein. Peptides are shaded in blue and black.
              The last peptide is a mis-cleavage and overlaps with \\texttt{IEEELGDNAVFAGENFHHGDK}.}
    \\label{fig:seq}
  \\end{center}
\\end{figure}\n\n",
    file = seq1file, append = TRUE)
```

^{15}N incorporation

```
## 15N incorporation rates from 0, 0.1, ..., 0.9, 0.95, 1
incrate <- c(seq(0, 0.9, 0.1), 0.95, 1)
inc <- lapply(incrate, function(inc)
              IsotopicDistributionN("YEVQGEVFTKPQLWP", inc))
par(mfrow = c(4,3))
for (i in 1:length(inc))
  plot(inc[[i]][, c(1, 3)], xlim = c(1823, 1848), type = "h",
       main = paste0("15N incorporation at ", incrate[i]*100, "%"))
```

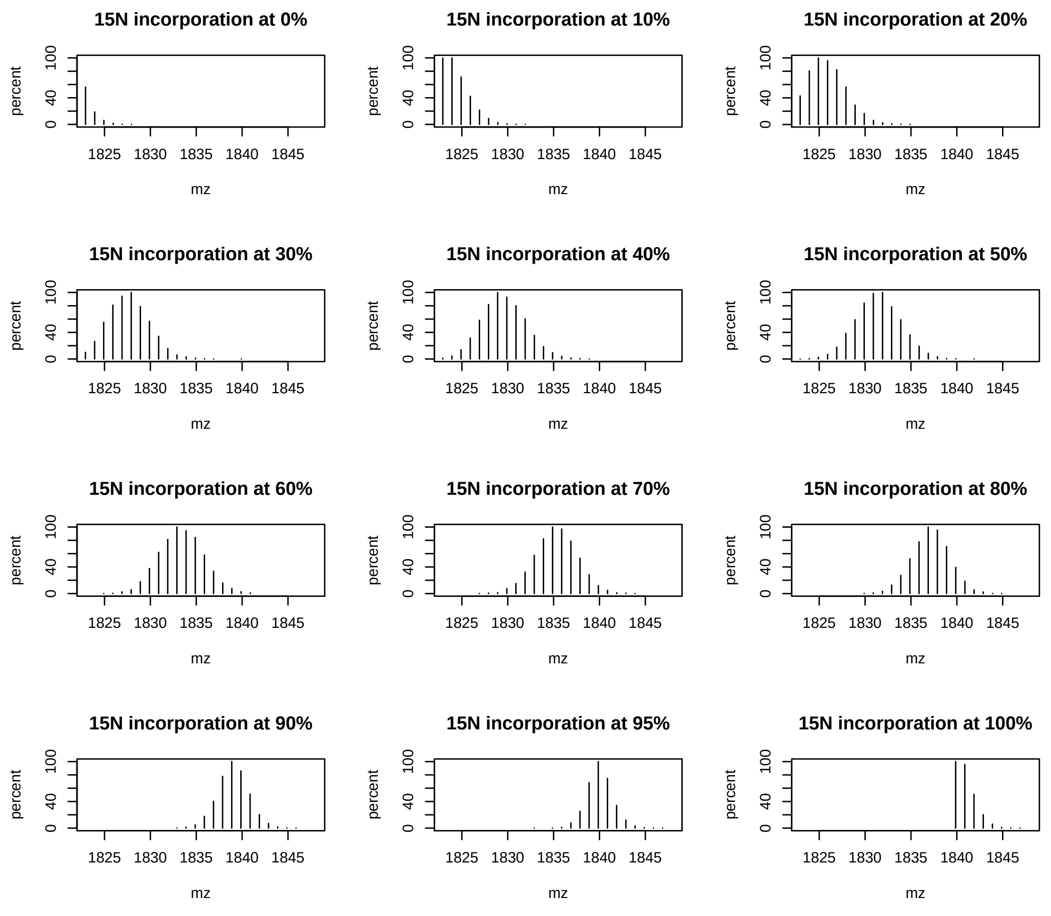


Figure 10: Isotopic envelope for the `YEVQGEVFTKPQLWP` peptide at different ^{15}N incorporation rates.

4.5 The isobar package

The **isobar** package [3] provides methods for the statistical analysis of isobarically tagged MS^2 experiments.

```
library(isobar)

## Prepare the PXD000001 data for isobar analysis
.ions <- exprs(qnt)
.mass <- matrix(mz(TMT6), nrow(qnt), byrow=TRUE, ncol = 6)
colnames(.ions) <- colnames(.mass) <-
  reporterTagNames(new("TMT6plexSpectra"))
rownames(.ions) <- rownames(.mass) <-
  paste(fData(qnt)$accession, fData(qnt)$sequence, sep = ".")
pgtbl <- data.frame(spectrum = rownames(.ions),
                    peptide = fData(qnt)$sequence,
                    modif = ":",
                    start.pos = 1,
                    protein = fData(qnt)$accession,
                    accession = fData(qnt)$accession)
x <- new("TMT6plexSpectra", pgtbl, .ions, .mass)

##  data.frame columns OK
## Creating ProteinGroup ...  done

featureData(x)$proteins <- as.character(fData(qnt)$accession)


x <- correctIsotopeImpurities(x) ## using identity matrix here

## LOG: isotopeImpurities.corrected:  TRUE

x <- normalize(x, per.file = FALSE)

## LOG: is.normalized:  TRUE
## LOG: normalization.multiplicative.factor channel 126:  0.8846
## LOG: normalization.multiplicative.factor channel 127:  0.9244
## LOG: normalization.multiplicative.factor channel 128:  1
## LOG: normalization.multiplicative.factor channel 129:  0.9421
## LOG: normalization.multiplicative.factor channel 130:  0.8593
## LOG: normalization.multiplicative.factor channel 131:  0.889

## spikes
spks <- c(protein.g(proteinGroup(x), "P00489"),
          protein.g(proteinGroup(x), "P00924"),
          protein.g(proteinGroup(x), "P02769"),
          protein.g(proteinGroup(x), "P62894"))

cls2 <- rep("#00000040", nrow(x))
pch2 <- rep(1, nrow(x))
cls2[grep("P02769", featureNames(x))] <- "gold4" ## BSA
cls2[grep("P00924", featureNames(x))] <- "dodgerblue" ## ENO
cls2[grep("P62894", featureNames(x))] <- "springgreen4" ## CYT
cls2[grep("P00489", featureNames(x))] <- "darkorchid2" ## PHO
pch2[grep("P02769", featureNames(x))] <- 19
pch2[grep("P00924", featureNames(x))] <- 19
pch2[grep("P62894", featureNames(x))] <- 19
pch2[grep("P00489", featureNames(x))] <- 19

nm <- NoiseModel(x)



ib.background <- subsetIBSpectra(x, protein=spks,
                                 direction = "exclude")

## Creating ProteinGroup ...  done

nm.background <- NoiseModel(ib.background)



ib.spks <- subsetIBSpectra(x, protein = spks,
                           direction="include",
                           specificity="reporter-specific")

## Creating ProteinGroup ...  done
```

```
nm.spks <- NoiseModel(ib.spks, one.to.one=FALSE, pool=TRUE)

## 4 proteins with more than 10 spectra, taking top 50.
## [1] 1.000e-10 6.193e+00 6.721e-01

ratios <- 10^estimateRatio(x, nm,
                           channel1="127", channel2="129",
                           protein = spks,
                           combine = FALSE)[, "lratio"]


res <- estimateRatio(x, nm,
                     channel1="127", channel2="129",
                     protein = unique(fData(x)$proteins),
                     combine = FALSE,
                     sign.level = 0.01)[, c(1, 2, 6, 8)]
res <- as.data.frame(res)
res$lratio <- -(res$lratio)

cls3 <- rep("#00000050", nrow(res))
pch3 <- rep(1, nrow(res))
cls3[grep("P02769", rownames(res))] <- "gold4" ## BSA
cls3[grep("P00924", rownames(res))] <- "dodgerblue" ## ENO
cls3[grep("P62894", rownames(res))] <- "springgreen4" ## CYT
cls3[grep("P00489", rownames(res))] <- "darkorchid2" ## PHO
pch3[grep("P02769", rownames(res))] <- 19
pch3[grep("P00924", rownames(res))] <- 19
pch3[grep("P62894", rownames(res))] <- 19
pch3[grep("P00489", rownames(res))] <- 19

rat.exp <- c(PHO = 2/2,
             ENO = 5/1,
             BSA = 2.5/10,
             CYT = 1/1)
```

```
maplot(x,
       noise.model = c(nm.background, nm.spks, nm),
       channel1="127", channel2="129",
       pch = 19, col = cls2,
       main = "Spectra MA plot")
abline(h = 1, lty = "dashed", col = "grey")
legend("topright",
       c("BSA", "ENO", "CYT", "PHO"),
       pch = 19, col = c("gold4", "dodgerblue",
                   "springgreen4", "darkorchid2"),
       bty = "n", cex = .7)
```

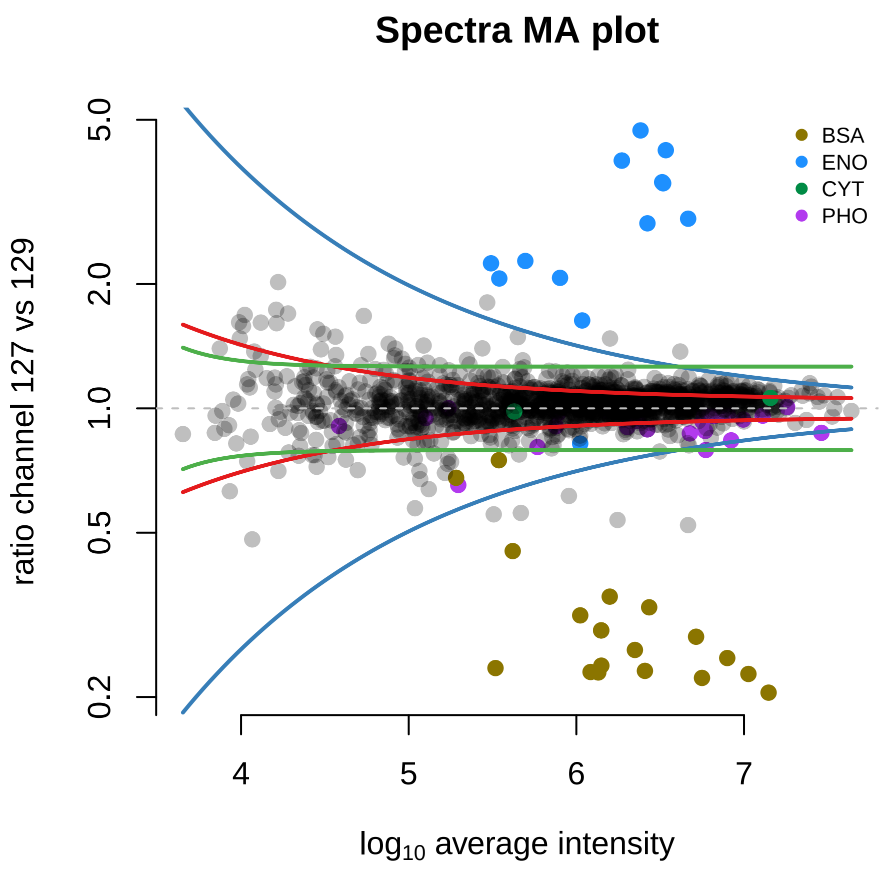


Figure 11: Result from the **isobar** pipeline.

4.6 The synapter package

The **synapter** [2] package comes with a detailed vignette that describes how to prepare the MS^E data and then process it in `R` . Several interfaces are available provided the user with maximum control, easy batch processing capabilities or a graphical user interface. The conversion into `MSnSet` instances and filter and combination thereof as well as statistical analysis are also described.

```
## open the synapter vignette
library("synapter")
synapterGuide()
```

5 MS^2 spectra identification

A recent addition to Bioconductor 2.12 is the **rTANDEM** package, that provides a direct interface to the X!Tandem software [5]. A typical **rTANDEM** pipeline comprises

1. Prepare the input data.
2. Run the search.
3. Import the search results and extract the peptides and proteins

Using example code/data from the **rTANDEM** vignette/package, these steps are executed as described below.

5.1 Preparation of the input data

```
library(rTANDEM)
taxonomy <- rTTaxo(taxon = "yeast",
                   format = "peptide",
                   URL = system.file(
                     "extdata/fasta/scd.fasta.pro",
                     package="rTANDEM"))
param <- rTParam()
param <- setParamValue(param,
                       'protein', 'taxon',
                       value="yeast")
param <- setParamValue(param, 'list path',
                       'taxonomy information', taxonomy)
param <- setParamValue(param,
                       'list path', 'default parameters',
                       value = system.file(
                         "extdata/default_input.xml",
                         package="rTANDEM"))
param <- setParamValue(param, 'spectrum', 'path',
                       value = system.file(
                         "extdata/test_spectra.mgf",
                         package="rTANDEM"))
param <- setParamValue(param, 'output', 'xsl path',
                       value = system.file(
                         "extdata/tandem-input-style.xsl",
                         package="rTANDEM"))
param <- setParamValue(param, 'output', 'path',
                       value = paste(getwd(),
                         "output.xml", sep="/"))
```

5.2 Performing the search

The analysis is run using the `tandem` function (see also the `rtandem` function), which returns the results data file path (only the file name is displayed below).

```
resultPath <- tandem(param)

## Loading spectra
##  (mgf). loaded.
## Spectra matching criteria = 242
## Starting threads . started.
## Computing models:
##  testin
##  sequences modelled = 5 ks
## Model refinement:
##  partial cleavage ..... done.
##  unanticipated cleavage ..... done.
##  modified N-terminus ..... done.
##  finishing refinement ... done.
## Creating report:
##  initial calculations  ..... done.
##  sorting  ..... done.
##  finding repeats ..... done.
##  evaluating results ..... done.
##  calculating expectations ..... done.
##  writing results ..... done.
##
## Valid models = 40
## Unique models = 41
## Estimated false positives = 1 +/- 1

basename(resultPath)

## [1] "output.2014_07_24_16_32_47.t.xml"
```

5.3 Import and analyse results

```
res <- GetResultsFromXML(resultPath)
## the inferred proteins
proteins <- GetProteins(res,
                        log.expect = -1.3,
                        min.peptides = 2)
proteins[, -(4:5), with = FALSE]

##      uid expect.value   label description num.peptides
##   1:  576        -27.2 YCR012W     YCR012W            5
##   2: 1811        -14.5 YFR053C     YFR053C            3
##   3: 2301        -12.8 YGR254W     YGR254W            3
##   4:    4        -12.0 YAL005C     YAL005C            3
##  5: 3517        -12.0 YLL024C     YLL024C            3
##  6: 3328        -10.3 YKL152C     YKL152C            2
##  7: 3386        -10.1 YKL216W     YKL216W            2
##  8: 2281         -7.9 YGR234W     YGR234W            2
##  9: 2568         -7.5 YHR174W     YHR174W            2
## 10: 2044         -7.1 YGL253W     YGL253W            2
```

```
## the identified peptides for YFR053C
peptides <- GetPeptides(protein.uid = 1811,
                        results = res,
                        expect = 0.05)
peptides[, c(1:4, 9, 10:16), with = FALSE]
```

```
##     pep.id prot.uid spectrum.id spectrum.mh expect.value
## 1: 102.1.1     1811         102       942.5      0.00660
## 2: 250.1.1     1811         250      1212.6      0.00043
## 3:  60.1.1     1811          60       863.5      0.00870
##    tandem.score     mh   delta peak.count missed.cleavages
## 1:         31.9  942.5 -0.0220         NA                0
## 2:         35.0 1212.6  0.0079         NA                0
## 3:         21.7  863.5 -0.0052         NA                0
##    start.position end.position
## 1:            166          173
## 2:            437          447
## 3:            309          315
```

More details are provided in the vignette available with (vignette(”rTANDEM”)), for instance the extraction of degenerated peptides, i.e. peptides found in multiple proteins.

The **shinyTANDEM** package offers a web-based graphical interface to **rTANDEM**.

6 Annotation

In this section, we briefly present some Bioconductor annotation infrastructure.

We start with the **hpar** package, an interface to the *Human Protein Atlas* [14, 15], to retrieve subcellular localisation information for the `ENSG00000002746` ensemble gene.

```
id <- "ENSG00000002746"
library("hpar")
getHpa(id, "SubcellularLoc")

##                 Gene                        Main.location
## 25 ENSG00000002746 Nucleus but not nucleoli;Cytoplasm
##    Other.location Expression.type Reliability
## 25                            APE        High
```

Below, we make use of the human annotation package **org.Hs.eg.db** and the Gene Ontology annotation package **GO.db** to retrieve the same information as above.

```
library(org.Hs.eg.db)
library(GO.db)
ans <- select(org.Hs.eg.db,
              keys = id, columns = c("ENSEMBL", "GO", "ONTOLOGY"),
              keytype = "ENSEMBL")
ans <- ans[ans$ONTOLOGY == "CC", ]
ans

##           ENSEMBL         GO EVIDENCE ONTOLOGY
## 2 ENSG00000002746 GO:0005737      IEA       CC

sapply(as.list(GOTERM[ans$GO]), slot, "Term")

##  GO:0005737
## "cytoplasm"
```

Finally, this information can also be retrieved from on-line databases using the **biomaRt** package [7].

```
library("biomaRt")
ensembl <- useMart("ensembl",dataset="hsapiens_gene_ensembl")
efilter <- "ensembl_gene_id"
eattr <- c("go_id", "name_1006", "namespace_1003")
bmres <- getBM(attributes=eattr, filters = efilter, values = id, mart = ensembl)
bmres[bmres$namespace_1003 == "cellular_component", "name_1006"]
```

```
## [1] "cytoplasm" "nucleus"
```

7 Other packages

7.1 Bioconductor packages

This section provides a complete list of packages available in the relevant Bioconductor version 2.14 (as of July 24, 2014) *biocView*[18] categories. Tables 1, 2 and 3 represent the packages for the `Proteomics` (53 packages), `MassSpectrometry` (31 packages) and `MassSpectrometryData` (7 experiment packages) categories.

Package	Title	Version
ASEB	Predict Acetylated Lysine Sites	1.8.0
bioassayR	R library for Bioactivity analysis	1.2.0
BRAIN	Baffling Recursive Algorithm for Isotope distributioN calculations	1.10.0
CellNOptR	Training of boolean logic models of signalling networks using prior knowledge networks and perturbation data.	1.10.0
ChemmineR	Cheminformatics Toolkit for R	2.16.7
cisPath	Visualization and management of the protein-protein interaction networks.	1.4.4
cleaver	Cleavage of polypeptide sequences	1.2.0
clippda	A package for the clinical proteomic profiling data analysis	1.14.0
CNORdt	Add-on to CellNOptR: Discretized time treatments	1.6.0
CNORfeeder	Integration of CellNOptR to add missing links	1.4.0
CNORode	ODE add-on to CellNOptR	1.6.0
customProDB	Generate customized protein database from NGS data, with a focus on RNA-Seq data, for proteomics search.	1.4.0
deltaGseg	deltaGseg	1.4.0
eiR	Accelerated similarity searching of small molecules	1.4.4
fmcsR	Mismatch Tolerant Maximum Common Substructure Searching	1.6.4
GraphPAC	Identification of Mutational Clusters in Proteins via a Graph Theoretical Approach.	1.6.0
hpar	Human Protein Atlas in R	1.6.0
iPAC	Identification of Protein Amino acid Clustering	1.8.0
IPPD	Isotopic peak pattern deconvolution for Protein Mass Spectrometry by template matching	1.12.0
isobar	Analysis and quantitation of isobarically tagged MSMS proteomics data	1.10.0
LPEadj	A correction of the local pooled error (LPE) method to replace the asymptotic variance adjustment with an unbiased adjustment based on sample size.	1.24.0
MassSpecWavelet	Mass spectrum processing by wavelet-based algorithms	1.30.0
msmsEDA	Exploratory Data Analysis of LC-MS/MS data by spectral counts	1.2.0
msmsTests	LC-MS/MS Differential Expression Tests	1.2.0
MSnbase	MSnbase: Base Functions and Classes for MS-based Proteomics	1.12.1
MSstats	Protein Significance Analysis in DDA, SRM and DIA for Label-free or Label-based Proteomics Experiments	2.2.0
mzID	An mzIdentML parser for R	1.2.1
mzR	parser for netCDF, mzXML, mzData and mzML files (mass spectrometry data)	1.10.6
PAnnBuilder	Protein annotation data package builder	1.28.0
pathview	a tool set for pathway based data integration and visualization	1.4.2
PCpheno	Phenotypes and cellular organizational units	1.26.0

[18] http://www.bioconductor.org/packages/devel/BiocViews.html

plgem	Detect differential expression in microarray and proteomics datasets with the Power Law Global Error Model (PLGEM)	1.36.0
PLPE	Local Pooled Error Test for Differential Expression with Paired High-throughput Data	1.24.0
ppiStats	Protein-Protein Interaction Statistical Package	1.30.0
PROcess	Ciphergen SELDI-TOF Processing	1.40.0
procoil	Prediction of Oligomerization of Coiled Coil Proteins	1.14.0
ProCoNA	Protein co-expression network analysis (ProCoNA).	1.2.0
pRoloc	A unifying bioinformatics framework for spatial proteomics	1.4.0
prot2D	Statistical Tools for volume data from 2D Gel Electrophoresis	1.2.0
qcmetrics	A Framework for Quality Control	1.2.0
RCASPAR	A package for survival time prediction based on a piecewise baseline hazard Cox regression model.	1.10.0
Rchemcpp	Similarity measures for chemical compounds	2.2.0
Rcpi	Toolkit for Compound-Protein Interaction in Drug Discovery	1.0.0
RpsiXML	R interface to PSI-MI 2.5 files	2.6.0
rpx	R Interface to the ProteomeXchange Repository	1.0.1
rTANDEM	Interfaces the tandem protein identification algorithm in R	1.4.0
sapFinder	A package for variant peptides detection and visualization in shotgun proteomics.	1.0.1
ScISI	In Silico Interactome	1.36.0
shinyTANDEM	Provides a GUI for rTANDEM	1.2.0
SLGI	Synthetic Lethal Genetic Interaction	1.24.0
SpacePAC	Identification of Mutational Clusters in 3D Protein Space via Simulation.	1.2.0
spliceSites	Manages align gap positions from RNA-seq data	1.2.0
synapter	Label-free data analysis pipeline for optimal identification and quantitation	1.6.0

Table 1: Packages available under the `Proteomics` *biocViews* category.

Package	Title	Version
apComplex	Estimate protein complex membership using AP-MS protein data	2.30.0
BRAIN	Baffling Recursive Algorithm for Isotope distributioN calculations	1.10.0
CAMERA	Collection of annotation related methods for mass spectrometry data	1.20.0
flagme	Analysis of Metabolomics GC/MS Data	1.20.1
gaga	GaGa hierarchical model for high-throughput data analysis	2.10.0
iontree	Data management and analysis of ion trees from ion-trap mass spectrometry	1.10.0
isobar	Analysis and quantitation of isobarically tagged MSMS proteomics data	1.10.0
MassArray	Analytical Tools for MassArray Data	1.16.0
MassSpecWavelet	Mass spectrum processing by wavelet-based algorithms	1.30.0
metaMS	MS-based metabolomics annotation pipeline	1.0.0
msmsEDA	Exploratory Data Analysis of LC-MS/MS data by spectral counts	1.2.0
msmsTests	LC-MS/MS Differential Expression Tests	1.2.0
MSnbase	MSnbase: Base Functions and Classes for MS-based Proteomics	1.12.1
MSstats	Protein Significance Analysis in DDA, SRM and DIA for Label-free or Label-based Proteomics Experiments	2.2.0
mzID	An mzIdentML parser for R	1.2.1
mzR	parser for netCDF, mzXML, mzData and mzML files (mass spectrometry data)	1.10.6
PAPi	Predict metabolic pathway activity based on metabolomics data	1.4.0
PROcess	Ciphergen SELDI-TOF Processing	1.40.0
pRoloc	A unifying bioinformatics framework for spatial proteomics	1.4.0
qcmetrics	A Framework for Quality Control	1.2.0
Rdisop	Decomposition of Isotopic Patterns	1.24.1

Risa	Converting experimental metadata from ISA-tab into Bioconductor data structures	1.6.0
RMassBank	Workflow to process tandem MS files and build MassBank records	1.6.0
rols	An R interface to the Ontology Lookup Service	1.6.0
rpx	R Interface to the ProteomeXchange Repository	1.0.1
rTANDEM	Interfaces the tandem protein identification algorithm in R	1.4.0
sapFinder	A package for variant peptides detection and visualization in shotgun proteomics.	1.0.1
shinyTANDEM	Provides a GUI for rTANDEM	1.2.0
synapter	Label-free data analysis pipeline for optimal identification and quantitation	1.6.0
TargetSearch	A package for the analysis of GC-MS metabolite profiling data.	1.20.0
xcms	LC/MS and GC/MS Data Analysis	1.40.0

Table 2: Packages available under the `MassSpectrometry` *biocViews* category.

Package	Title	Version
faahKO	Saghatelian et al. (2004) FAAH knockout LC/MS data	1.4.0
gcspikelite	Spike-in data for GC/MS data and methods within flagme	1.2.0
metaMSdata	Example CDF data for the metaMS package	1.0.0
msdata	Various Mass Spectrometry raw data example files	0.2.0
RforProteomics	Companion package to the 'Using R and Bioconductor for proteomics data analysis' publication	1.2.0
RMassBankData	Test dataset for RMassBank	1.2.0
synapterdata	Data accompanying the synapter package	1.2.0

Table 3: Experimental Packages available under the `MassSpectrometryData` *biocViews* category.

The tables can easily be generated with the `proteomicsPackages`, `massSpectrometryPackages` and `massSpectrometryDataPackages` functions. The respective package tables can then be interactively explored using the `display` function.

```r
pp <- proteomicsPackages()
display(pp)
```

7.2 The Chemometrics and Computational Physics CRAN Task View

The CRAN task view on Chemometrics and Computational Physics[19] lists 76 packages, including a set of packages for mass spectrometry and proteomics, some of which are illustrated in this document. The most relevant (non Bioconductor) packages are summarised below.

[19] http://cran.r-project.org/web/views/ChemPhys.html

MALDIquant provides tools for quantitative analysis of MALDI-TOF mass spectrometry data, with support for baseline correction, peak detection and plotting of mass spectra
(http://cran.r-project.org/web/packages/MALDIquant/index.html).

OrgMassSpecR is for organic/biological mass spectrometry, with a focus on graphical display, quantification using stable isotope dilution, and protein hydrogen/deuterium exchange experiments
(http://cran.r-project.org/web/packages/OrgMassSpecR/index.html).

FTICRMS provides functions for Analyzing Fourier Transform-Ion Cyclotron Resonance Mass Spectrometry Data
(http://cran.r-project.org/web/packages/FTICRMS/index.html).

titan provides a GUI to analyze mass spectrometric data on the relative abundance of two substances from a titration series
(http://cran.r-project.org/web/packages/titan/index.html).

7.3 Other CRAN packages

Finally, **digeR**[20], which is available on CRAN but not listed in the Chemometrics and Computational Physics Task View, provides a GUI interface for analysing 2D DIGE data. It allows to perform correlation analysis, score plot, classification, feature selection and power analysis for 2D DIGE experiment data.

Suggestions for additional `R` packages are welcome and will be added to the vignette. Please send suggestions and possibly a short description and/or a example utilisation with code to lg390@cam.ac.uk. The only requirement is that the package must be available on an official package channel (CRAN, Bioconductor, R-forge, Omegahat), i.e. not only available through a personal web page.

[20]http://cran.r-project.org/web/packages/digeR/index.html

8 Session information

All software and respective versions used in this document, as returned by `sessionInfo()` are detailed below.

- R version 3.1.1 Patched (2014-07-22 r66230), `x86_64-unknown-linux-gnu`
- Base packages: base, datasets, graphics, grDevices, methods, parallel, stats, utils
- Other packages: AnnotationDbi 1.26.0, Biobase 2.24.0, BiocGenerics 0.10.0, BiocInstaller 1.14.2, Biostrings 2.32.1, bitops 1.0-6, BRAIN 1.10.0, cleaver 1.2.0, data.table 1.9.2, DBI 0.2-7, digest 0.6.4, GenomeInfoDb 1.0.2, ggplot2 1.0.0, GO.db 2.14.0, hpar 1.6.0, IPPD 1.12.0, IRanges 1.22.9, isobar 1.10.0, knitr 1.6, lattice 0.20-29, MALDIquant 1.10, MALDIquantForeign 0.8, MASS 7.3-33, Matrix 1.1-4, msdata 0.2.0, MSnbase 1.12.1, mzID 1.2.1, mzR 1.10.6, org.Hs.eg.db 2.14.0, OrgMassSpecR 0.4-4, plyr 1.8.1, PolynomF 0.94, RColorBrewer 1.0-5, Rcpp 0.11.2, RcppClassic 0.9.5, Rdisop 1.24.1, reshape2 1.4, RforProteomics 1.2.1, rols 1.6.0, rpx 1.0.0, RSQLite 0.11.4, rTANDEM 1.4.0, XML 3.98-1.1, xtable 1.7-3, XVector 0.4.0
- Loaded via a namespace (and not attached): affy 1.42.3, affyio 1.32.0, annotate 1.42.1, base64enc 0.1-2, biocViews 1.32.1, Category 2.30.0, caTools 1.17, codetools 0.2-8, colorspace 1.2-4, distr 2.5.2, doParallel 1.0.8, downloader 0.3, evaluate 0.5.5, foreach 1.4.2, formatR 0.10, genefilter 1.46.1, graph 1.42.0, grid 3.1.1, gridSVG 1.4-0, GSEABase 1.26.0, gtable 0.1.2, highr 0.3, htmltools 0.2.4, httpuv 1.3.0, impute 1.38.1, interactiveDisplay 1.2.0, iterators 1.0.7, labeling 0.2, limma 3.20.8, munsell 0.4.2, pcaMethods 1.54.0, preprocessCore 1.26.1, proto 0.3-10, R.methodsS3 1.6.1, R.oo 1.18.0, R.utils 1.32.4, RBGL 1.40.0, RCurl 1.95-4.1, readBrukerFlexData 1.7, readMzXmlData 2.7, RJSONIO 1.2-0.2, RUnit 0.4.26, scales 0.2.4, sfsmisc 1.0-26, shiny 0.10.0, splines 3.1.1, SSOAP 0.8-0, startupmsg 0.9, stats4 3.1.1, stringr 0.6.2, survival 2.37-7, SweaveListingUtils 0.6.1, tools 3.1.1, vsn 3.32.0, XMLSchema 0.7-2, zlibbioc 1.10.0

References

[1] H. P. Benton, D. M. Wong, S. A. Trauger, and G. Siuzdak. XCMS2: processing tandem mass spectrometry data for metabolite identification and structural characterization. *Anal Chem*, 80(16):6382–9, Aug 2008.

[2] N. J. Bond, P. V. Shliaha, K. S. Lilley, and L. Gatto. Improving qualitative and quantitative performance for label free proteomics. *J. Proteome Res.*, 2013.

[3] F. P. Breitwieser, A. Müller, L. Dayon, T. Köcher, A. Hainard, P. Pichler, U. Schmidt-Erfurth, G. Superti-Furga, J. C. Sanchez, K. Mechtler, K. L. Bennett,

and J. Colinge. General statistical modeling of data from protein relative expression isobaric tags. *J Proteome Res*, 10(6):2758–66, Jun 2011.

[4] M. C. Chambers, B. Maclean, R. Burke, D. Amodei, D. L. Ruderman, S. Neumann, L. Gatto, B. Fischer, B. Pratt, J. Egertson, K. Hoff, D. Kessner, N. Tasman, N. Shulman, B. Frewen, T. A. Baker, M. Y. Brusniak, C. Paulse, D. Creasy, L. Flashner, K. Kani, C. Moulding, S. L. Seymour, L. M. Nuwaysir, B. Lefebvre, F. Kuhlmann, J. Roark, P. Rainer, S. Detlev, T. Hemenway, A. Huhmer, J. Langridge, B. Connolly, T. Chadick, K. Holly, J. Eckels, E. W. Deutsch, R. L. Moritz, J. E. Katz, D. B. Agus, M. MacCoss, D. L. Tabb, and P. Mallick. A cross-platform toolkit for mass spectrometry and proteomics. *Nat Biotechnol*, 30(10):918–20, Oct 2012.

[5] R. Craig and R. C. Beavis. Tandem: matching proteins with tandem mass spectra. *Bioinformatics*, 20(9):1466–7, Jun 2004.

[6] A. Cuadros-Inostroza, C. Caldana, H. Redestig, J. Lisec, H. Pena-Cortes, L. Willmitzer, and M. A. Hannah. TargetSearch - a Bioconductor package for the efficient pre-processing of GC-MS metabolite profiling data. *BMC Bioinformatics*, 10:428, 2009.

[7] S. Durinck, Y. Moreau, A. Kasprzyk, S. Davis, B. De Moor, A. Brazma, and W. Huber. Biomart and bioconductor: a powerful link between biological databases and microarray data analysis. *Bioinformatics*, 21(16):3439–40, Aug 2005.

[8] L. Gatto and A. Christoforou. Using R and bioconductor for proteomics data analysis. *BBA - Proteins and Proteomics*, 2013.

[9] L. Gatto and K. S. Lilley. MSnbase – an R/Bioconductor package for isobaric tagged mass spectrometry data visualization, processing and quantitation. *Bioinformatics*, 28(2):288–9, Jan 2012.

[10] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, A. J. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, J. Y. H. Yang, and J. Zhang. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol*, 5(10):–80, 2004.

[11] S. Gibb and K. Strimmer. MALDIquant: a versatile R package for the analysis of mass spectrometry data. *Bioinformatics*, 28(17):2270–1, Sep 2012.

[12] C. A. Smith, E. J. Want, G. O'Maille, R. Abagyan, and G. Siuzdak. XCMS: processing mass spectrometry data for metabolite profiling using nonlinear peak alignment, matching, and identification. *Anal Chem*, 78(3):779–87, Feb 2006.

[13] R. Tautenhahn, C. Böttcher, and S. Neumann. Highly sensitive feature detection for high resolution LC/MS. *BMC Bioinformatics*, 9:504, 2008.

[14] M. Uhlén, E. Björling, C. Agaton, C. A.-K. A. Szigyarto, B. Amini, E. Andersen, A.-C. C. Andersson, P. Angelidou, A. Asplund, C. Asplund, L. Berglund, K. Bergström, H. Brumer, D. Cerjan, M. Ekström, A. Elobeid, C. Eriksson, L. Fagerberg, R. Falk, J. Fall, M. Forsberg, M. G. G. Björklund, K. Gumbel, A. Halimi, I. Hallin, C. Hamsten, M. Hansson, M. Hedhammar, G. Hercules, C. Kampf, K. Larsson, M. Lindskog, W. Lodewyckx, J. Lund, J. Lundeberg, K. Magnusson, E. Malm, P. Nilsson, J. Odling, P. Oksvold, I. Olsson, E. Oster, J. Ottosson, L. Paavilainen, A. Persson, R. Rimini, J. Rockberg, M. Runeson, A. Sivertsson, A. Sköllermo, J. Steen, M. Stenvall, F. Sterky, S. Strömberg, M. Sundberg, H. Tegel, S. Tourle, E. Wahlund, A. Waldén, J. Wan, H. Wernérus, J. Westberg, K. Wester, U. Wrethagen, L. L. L. Xu, S. Hober, and F. Pontén. A human protein atlas for normal and cancer tissues based on antibody proteomics. *Molecular & cellular proteomics : MCP*, 4(12):1920–1932, Dec. 2005.

[15] M. Uhlen, P. Oksvold, L. Fagerberg, E. Lundberg, K. Jonasson, M. Forsberg, M. Zwahlen, C. Kampf, K. Wester, S. Hober, H. Wernerus, L. Björling, and F. Ponten. Towards a knowledge-based Human Protein Atlas. *Nature biotechnology*, 28(12):1248–1250, Dec. 2010.

[16] J. A. Vizcaino, E. W. Deutsch, R. Wang, A. Csordas, F. Reisinger, D. Rios, J. A. Dianes, Z. Sun, T. Farrah, N. Bandeira, P.-A. Binz, I. Xenarios, M. Eisenacher, G. Mayer, L. Gatto, A. Campos, R. J. Chalkley, H.-J. Kraus, J. P. Albar, S. Martinez-Bartolome, R. Apweiler, G. S. Omenn, L. Martens, A. R. Jones, and H. Hermjakob. ProteomeXchange provides globally coordinated proteomics data submission and dissemination. *Nat Biotech*, 32:223–226, 2014.