R and Bioconductor

BS3033 Data Science for Biologists

Dr Wilson Goh
School of Biological Sciences

Learning Objectives

By the end of this topic, you should be able to:

- Describe the BioConductor project.
- Describe a workflow.
- Describe how a proteomics data analysis can be conducted using Bioconductor.

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

What is Bioconductor?

BS3033 Data Science for Biologists

Dr Wilson Goh
School of Biological Sciences

What is Bioconductor?

Bioconductor provides tools for the analysis and comprehension of high-throughput biological data.

It is based on the R statistical programming language, and is open source and open development.

It has two releases each year, 1560 software packages, and an active user community.

Installing Bioconductor

Bioconductor provides over a thousand packages for performing biodata analysis. You do not need to install all packages at once.

Find the relevant package you will need from: https://www.bioconductor.org/packages/release/BiocViews.html#___Software.

Run the R console and type the following:

```
## try http:// if https:// URLs are not supported source("https://bioconductor.org/biocLite.R");
biocLite(<"Package Name">)
##Do this to install multiple packages
biocLite(c("package1","package2"))
```

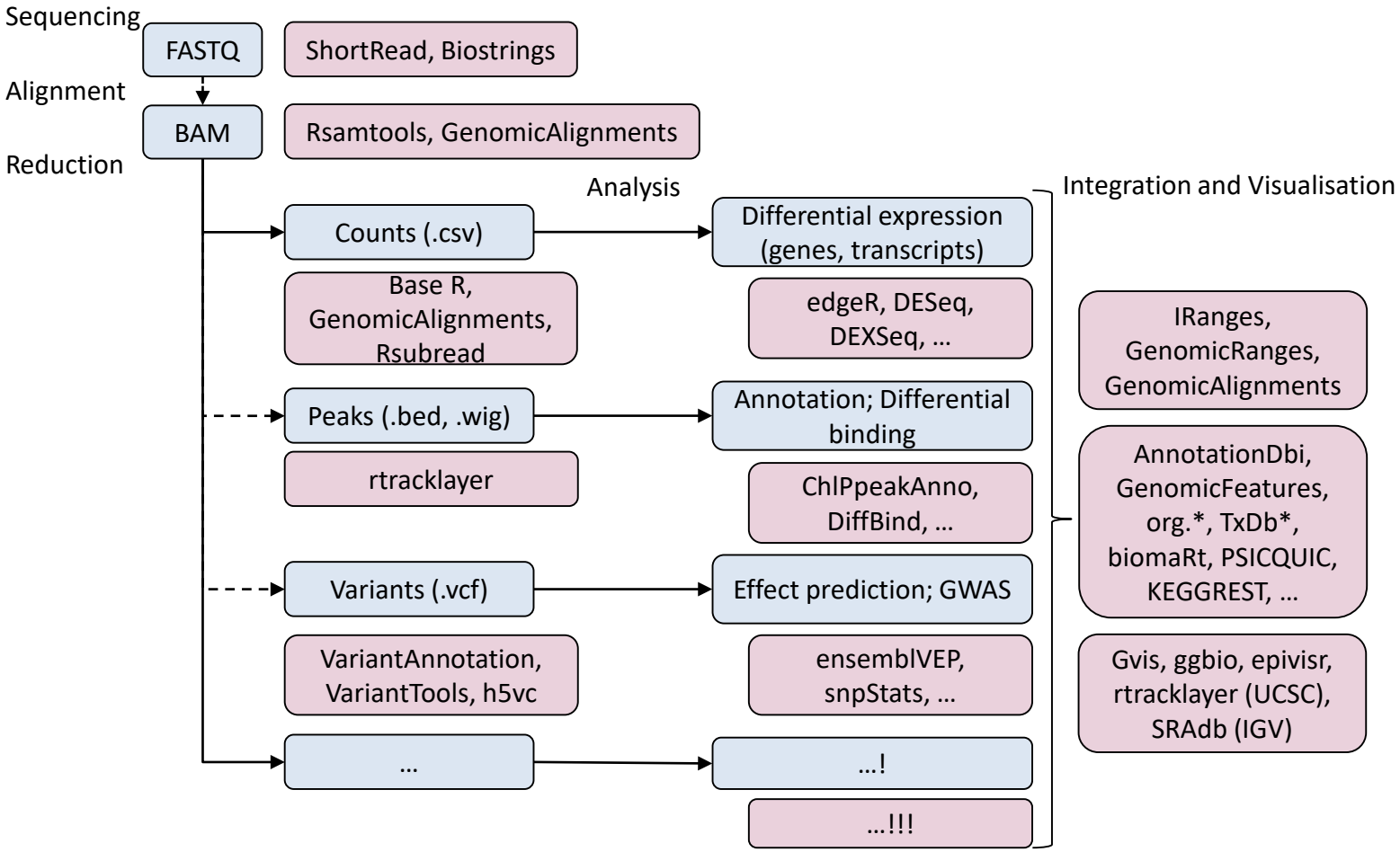
Learning More

You can access videos and up to date learning materials on Bioconductor from the below website:

https://www.bioconductor.org/help/course-materials/

Bioconductor Ecosystem for Genomics Analysis

Many software packages have overlapping functionalities.



But you can also see here that analysis has to be run in sequence. So how do we deal with this?

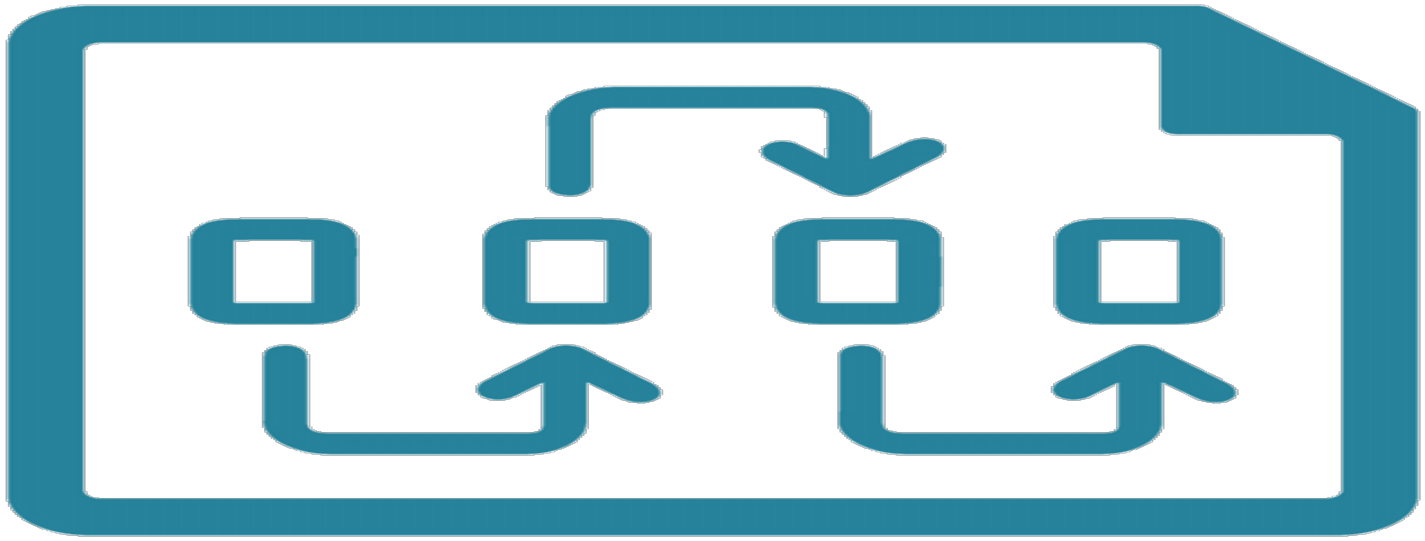
NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Workflows

BS3033 Data Science for Biologists

Dr Wilson Goh
School of Biological Sciences

What is a workflow?



Analysis of biological data requires “shepherding” files through a series of transformations, called a **pipeline** or a **workflow**.

What are the options?

A series of scripts designed to work in tandem (controlled using Unix shell script).

Basically a set of Unix commands saved in a file for convenience.

Do task 1 <input> <output>
Do task 2 <input> <output>
Do task 3 <input> <output> <arg 1> <arg 2>

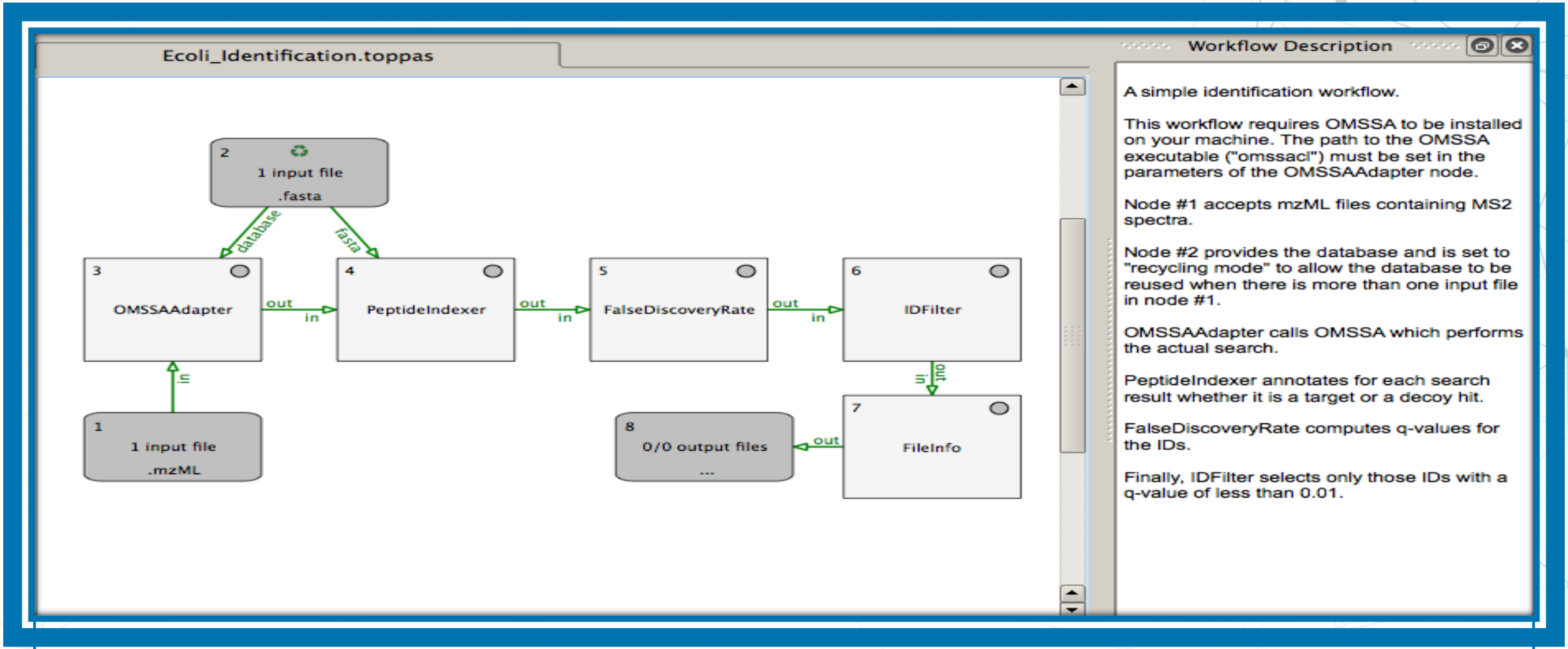
Each task is written as a separate program.
In the above example, the output of 1 task, is the input of another.

You can also specify arguments (arg1, arg2, …) to the scripts controlling each task so as to change the behaviors or analysis parameters.

For example, if task 3 is a statistical analysis tool. Arg 1 is the statistical threshold, so we can set the value of Arg 1 to 0.05 or 0.01.

R scripts can be executed in Unix sequentially using the **Rscript** command.

What are the options?



A graphical user interface (GUI) for controlling sequence and behaviors.

What are the options?

A library of tools that can be executed within 1 script.

Line 1: x <- Function 1(<data>, args 1, args 2,)
Line 2: y <- Function 2(x, args 1, args 2,)
Line 3: z <- Function 3(y, args 1, args 2,)

This is a self-contained script calling a series of functions within R itself (without using the Unix command line). Similar to the Unix shellscript, it also involves calling a series of functions with the output of 1 line becoming the input of the next line.

Workflows

Refer to the lecture notes for links to help on various workflows

https://bioconductor.org/help/workflows/

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Bioconductor in Action: Proteomics Workflow

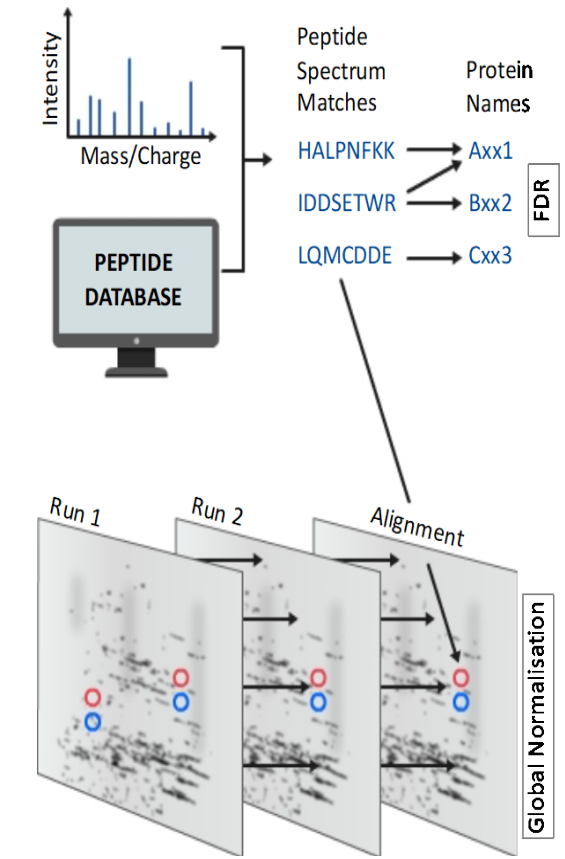
BS3033 Data Science for Biologists

Dr Wilson Goh
School of Biological Sciences

Objectives of Proteomics

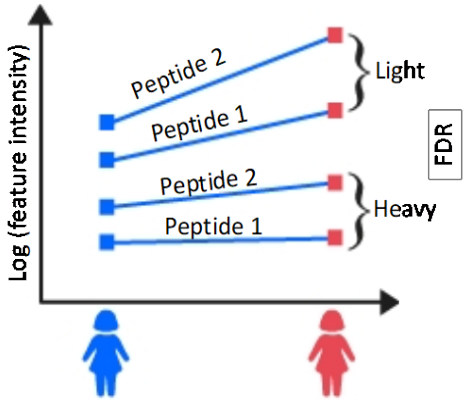
Technology-dependent

a) Peptide and Protein Identification from PSMs

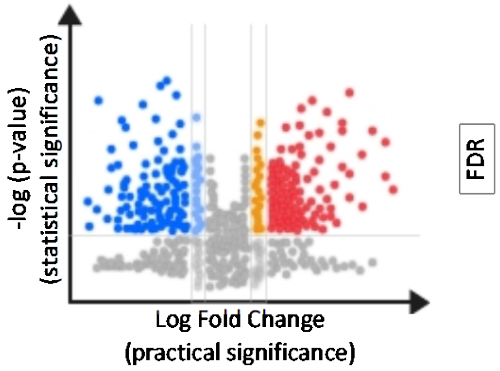


b) Feature Detection, Quantification Annotation, and Alignment

c) Peptide Significance Analysis

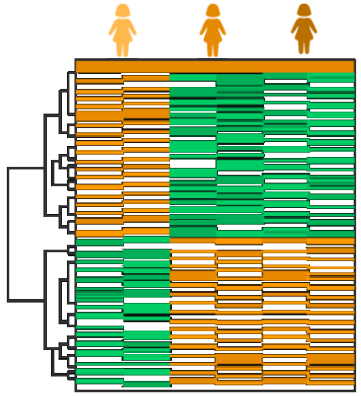


d) Protein Significance Analysis

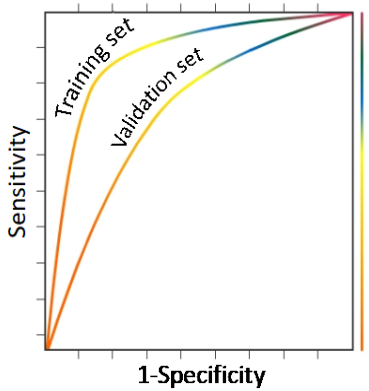


Technology-independent

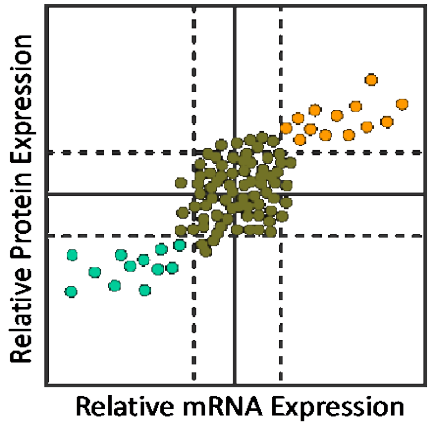
e) Class Discovery



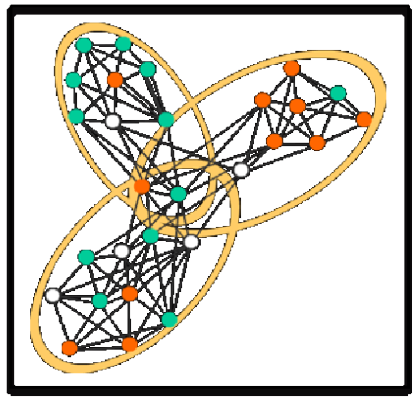
f) Class Prediction



g) Data Integration

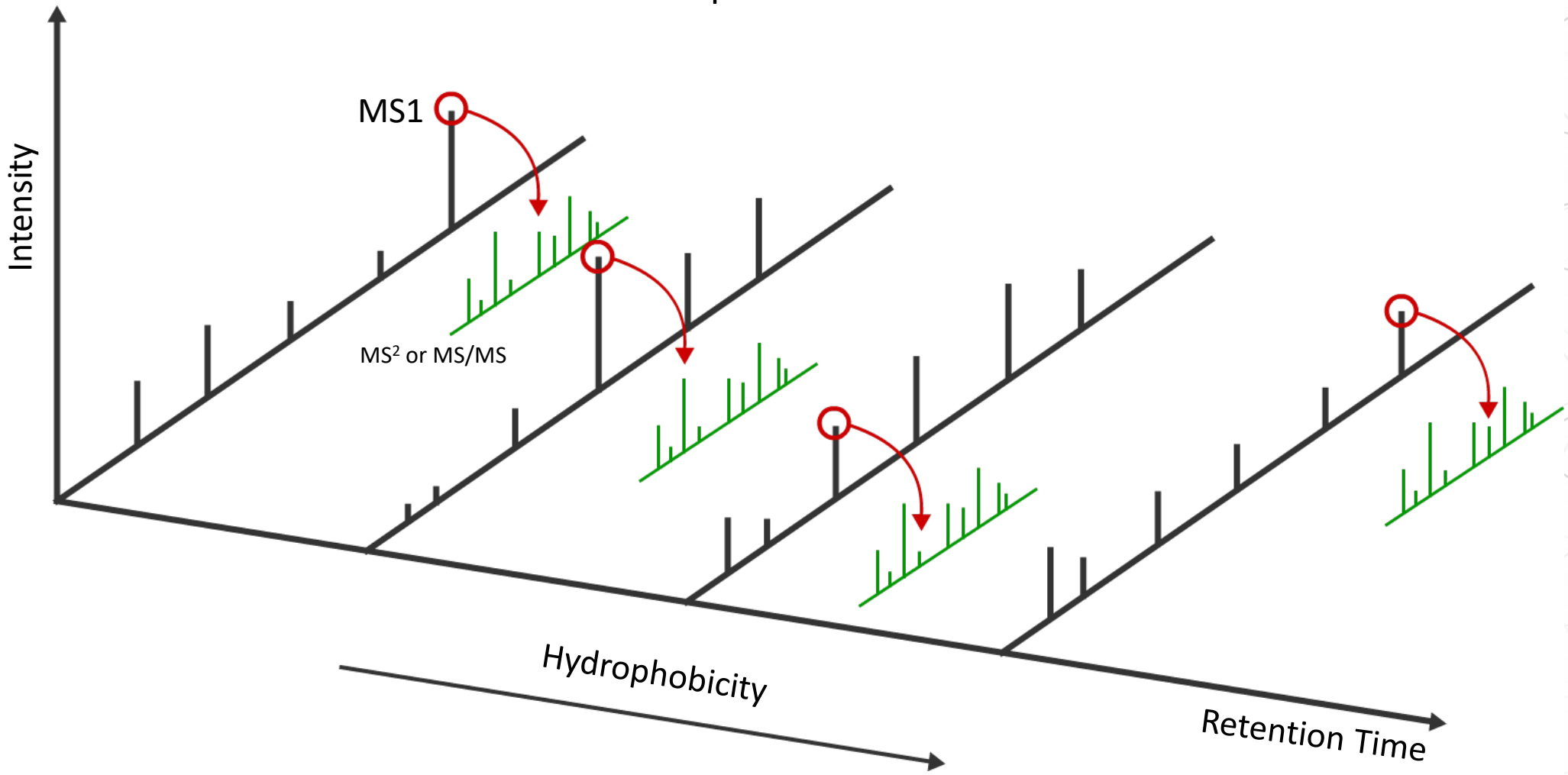


h) Pathway Analysis



Layout of the Data

There is a one-to-one correspondence between each MS1 and MS2.



Installing the R Bioconductor Proteomics Pipeline

To install everything in one shot:

```
library("BiocInstaller")
biocLite("RforProteomics", dependencies = TRUE)
```

To install individually:

```
biocLite(c("mzR",
"mzID",
"MSnID",
"MSnbase",
"rpx",
"MLInterfaces",
"pRoloc",
"pRolocdata",
"MSGFplus",
"rols",
"hpar"), dependencies = TRUE)
```

Installing the R Bioconductor Proteomics Pipeline

First we load all the libraries we need into memory:

```
library("mzR")
library("mzID")
library("MSnID")
library("MSnbase")
library("rpx")
library("MLInterfaces")
library("pRoloc")
library("pRolocdata")
library("MSGFplus")
library("rols")
library("hpar")
```

What can this pipeline do?

Exploring Available Infrastructure

In Bioconductor version 3.7, there are respectively 105 proteomics, 69 mass spectrometry software packages and 19 mass spectrometry experiment packages. These respective packages can be extracted with the proteomicsPackages(), massSpectrometryPackages() and massSpectrometryDataPackages(), and explored interactively.

```
library("RforProteomics")
pp <- proteomicsPackages()
display(pp)
```

Mass Spectrometry Data

R/Bioconductor support many open access formats:		
Type	**Format**	**Package**
Raw	mzML,mzXML, netCDF, mzData	mzR (read)
Identification	mzIdentML	mzR (read) and mZID (read)
Quantitation	mzQuantML	
Peak Lists	mgf	MSnbase (read/write)
Other	mzTab	MSnbase (read)

Obtaining Data from Database

The ***rpx*** is an interface to **ProteomeXchange** (PX) and provides a basic access to its data. PX is a massive proteomics database which also takes data from **PRIDE**, **PASSEL** and **MassIVE**.

```
library("rpx")
pxannounced()
```

```
## 15 new ProteomeXchange announcements
```

```
## Data.Set Publication.Data Message
## 1 PXD008710 2018-04-25 07:30:20 Updated information
## 2 PXD008376 2018-04-25 07:25:40 New
## 3 PXD006838 2018-04-25 07:19:31 New
## 4 PXD006836 2018-04-25 07:12:42 New
## 5 PXD002273 2018-04-25 07:07:05 New
…
```

Obtaining Data from Database

Metadata can be retrieved remotely using the PXDataset() function and calling its specific identifier e.g. PXD000001.

```
px <- PXDataset("PXD000001")
px #to find out what is in there
```

```
## Object of class "PXDataset"
## Id: PXD000001 with 12 files
## [1] 'F063721.dat' ... [12] 'generated' ## Use 'pxfiles(.)' to see all files.
```

Obtaining Data from Database

```
pxfiles(px) #take a look at the specific files inside (those 12)
```

```
## [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] "README.txt"
## [7] "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01-20141210.mzML" ##
[8] "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01-20141210.mzXML" ##
…
```

Obtaining Data from Database

Data files can then be downloaded with the pxget function. The file is downloaded into the working directory and the name of the file is return by the function and stored in the mzf variable for later use.

```
fn <-"TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01-20141210.mzML"
mzf <- pxget(px, fn)
```

```

```

```
## Downloading 1 file
```

```
mzf
```

```
## [1]
"/tmp/RtmpSUTMZy/Rbuild77de6f913a8c/proteomics/vignettes/TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01-20141210.mzML"
```

```

```

Handling Raw MS Data

The mzR package is useful for reading raw MS formats including mzML, mzXML, netCDF, and mzData. The three main functions are openMSfile to create a file handle to a raw data file, header to extract data about the spectra contained in the file and peaks to extract one or multiple spectra of interest.

```
library("mzR")
ms <- openMSfile(mzf)
ms
```

```
## Mass Spectrometry file handle.
## Filename: TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01-20141210.mzML
## Number of scans: 7534
```

The header function returns the metadata of all available peaks:

```
hd <- header(ms) dim(hd)
```

```
## [1] 7534 25
```

Handling Raw MS Data

```
names(hd)
```

```
## [1] "seqNum" "acquisitionNum"
## [3] "msLevel" "polarity"
## [5] "peaksCount" "totIonCurrent"
## [7] "retentionTime" "basePeakMZ"
## [9] "basePeakIntensity" "collisionEnergy"
## [11] "ionisationEnergy" "lowMZ"
## [13] "highMZ" "precursorScanNum"
## [15] "precursorMZ" "precursorCharge"
## [17] "precursorIntensity" "mergedScan"
## [19] "mergedResultScanNum" "mergedResultStartScanNum"
## [21] "mergedResultEndScanNum" "injectionTime"
## [23] "filterString" "spectrumId"
## [25] "centroided"
```

Handling Raw MS Data

Extract metadata and scan data for scan 1000 by simply calling it by its row.

```
hd[1000, ]
```

```
## seqNum acquisitionNum msLevel polarity peaksCount totIonCurrent
## 1000 1000 1000 2 1 274 1048554
## retentionTime basePeakMZ basePeakIntensity collisionEnergy ## 1000 1106.916 136.061 164464 45
## ionisationEnergy lowMZ highMZ precursorScanNum precursorMZ ## 1000 0 104.5467 1370.758 992 683.0817
## precursorCharge precursorIntensity mergedScan mergedResultScanNum
## 1000 2 689443.7 0 0
## mergedResultStartScanNum mergedResultEndScanNum injectionTime
## 1000 0 0 55.21463
## filterString
…
```

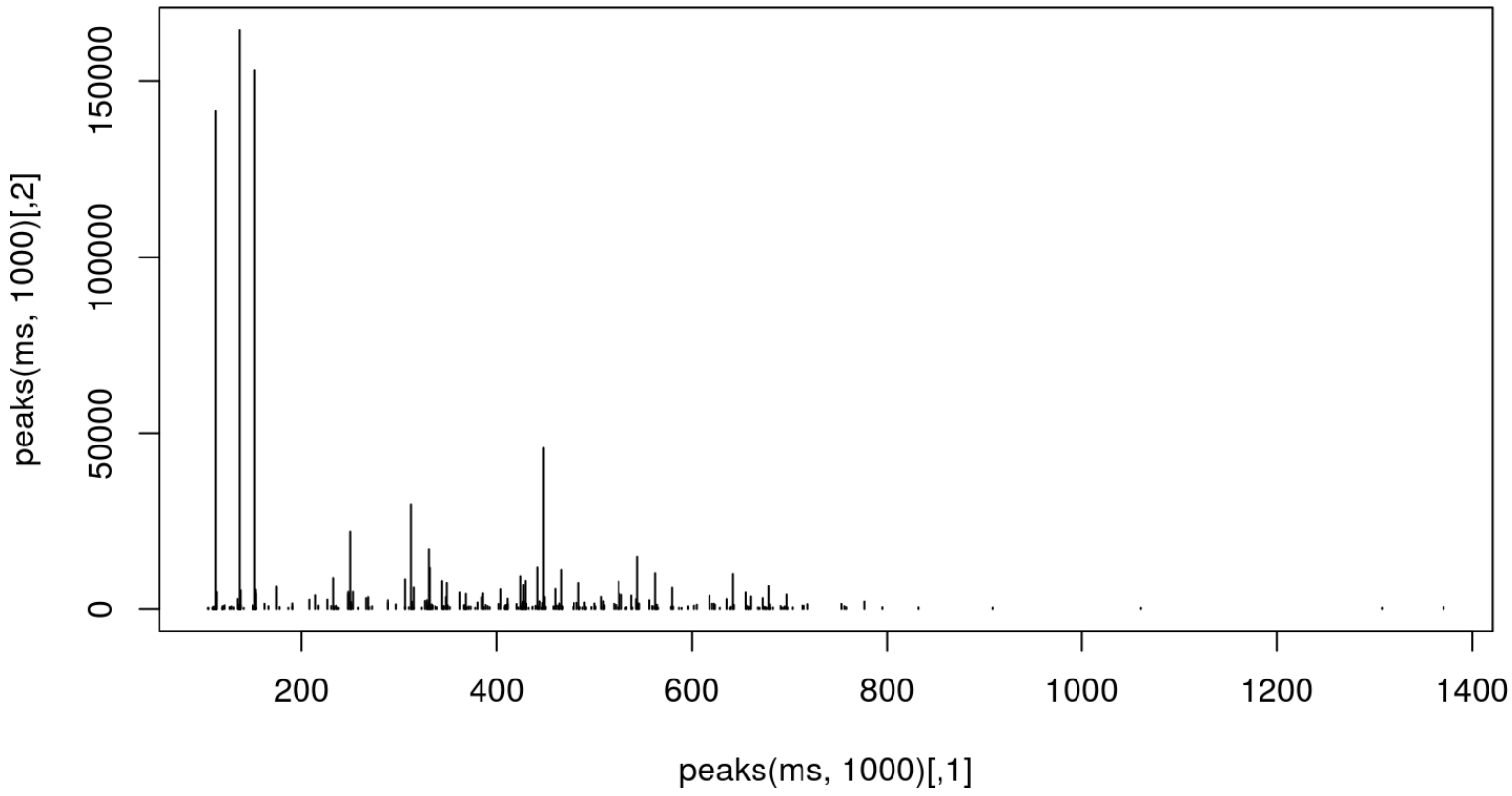
Handling Raw MS Data

```
head(peaks(ms, 1000))
```

```
##      [,1]     [,2]
## [1,] 104.5467 308.9326
## [2,] 104.5684 308.6961
## [3,] 108.8340 346.7183
## [4,] 109.3928 365.1236
## [5,] 110.0345 616.7905
## [6,] 110.0703 429.1975
```

Handling Raw MS Data

```
plot(peaks(ms, 1000), type = "h")
```



Extracting and Plotting a Slice of MS Data

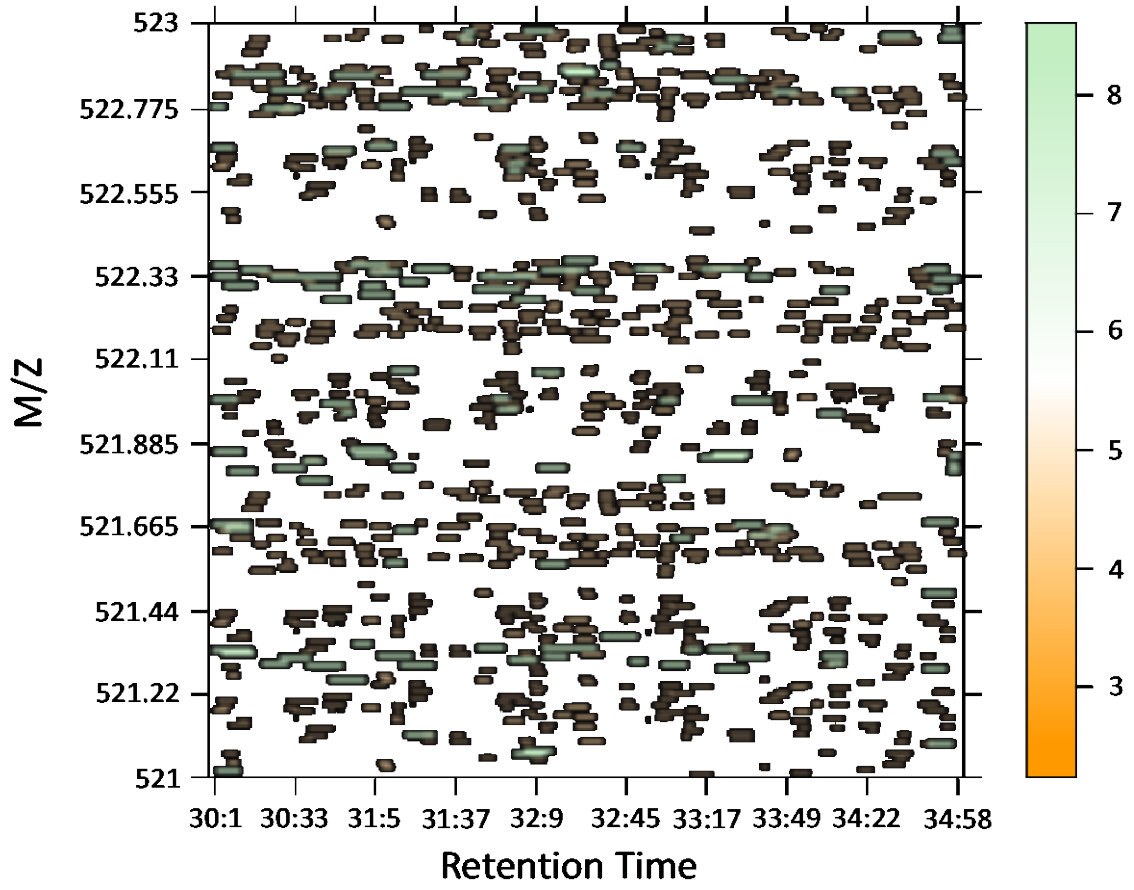
```
## a set of spectra of interest: MS1 spectra eluted
## between 30 and 35 minutes retention time
ms1 <- which(hd$msLevel == 1)
rtsel <- hd$retentionTime[ms1] / 60 > 30 & hd$retentionTime[ms1] / 60 < 35

## the map
M <- MSmap(ms, ms1[rtsel], 521, 523, .005, hd)
```

data rt range mz range header

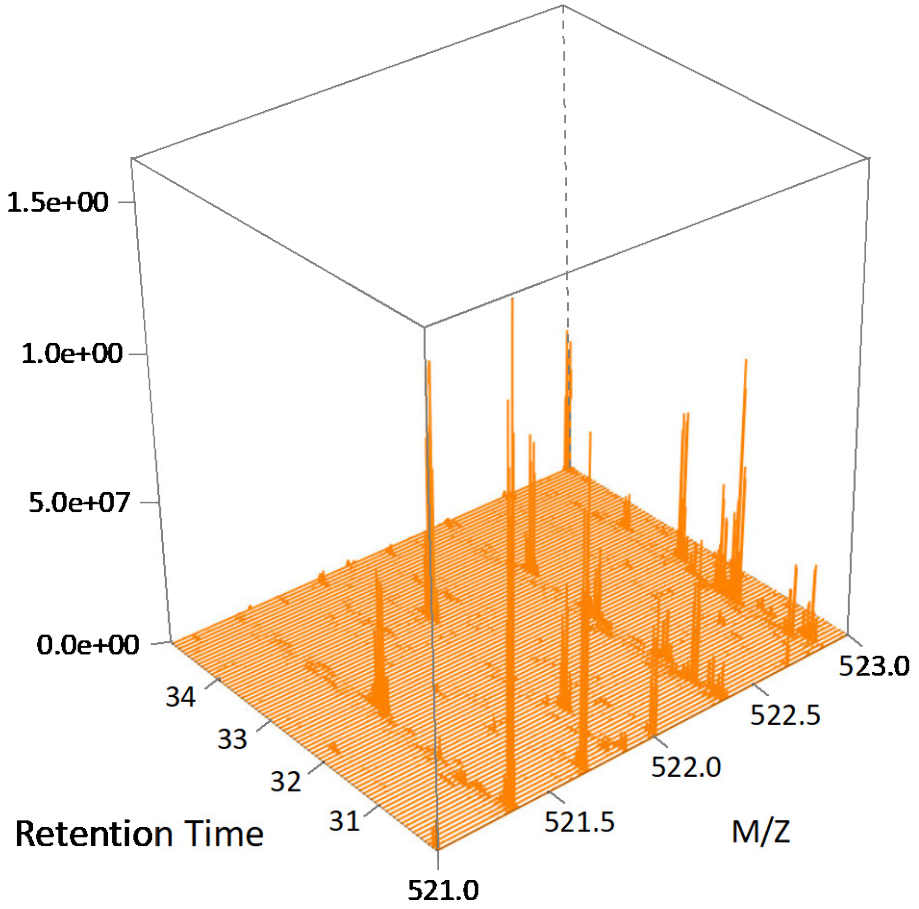
Extracting and Plotting a Slice of MS Data

```
plot(M, aspect = 1, allTicks = FALSE)
```

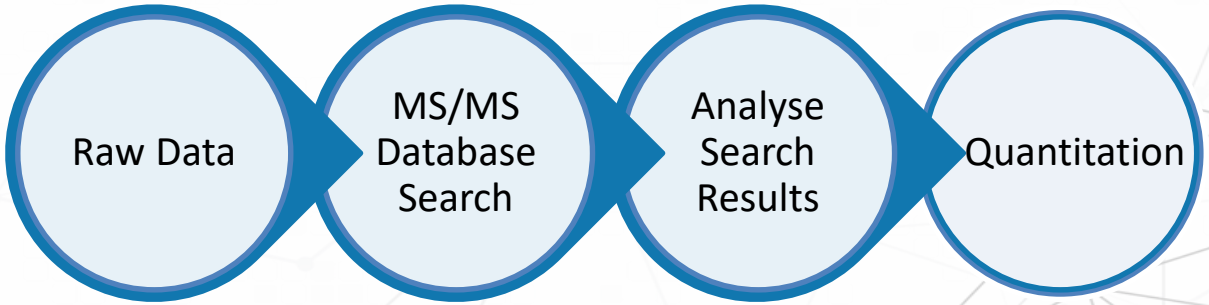


Extracting and Plotting a Slice of MS Data

plot3D(M)



Processing Raw Data



Processing data is a lot of work and requires very specific knowledge on formats, appropriate algorithms and reasonable parameters.

Even if we do know what needs to be done at each step/layer, knowing how to use the software packages isn't necessarily so straightforward.

Bioconductor workflows helps to ease some of the knowledge-practice gap.

Proteomics Data Layers

Raw

- Data and metadata generated by mass spectrometers.
- The data may be the original profile mode scans or may already have had some basic processing.
- **Binary output**

Standardised MS Data Formats

- Represent processed peak lists, as well as raw data. In addition to the mass spectra, they contain detailed metadata that gives context to the information.
- **mzML**

Processed Peak Lists

- Heavily processed.
- These files are formatted in plain text, with typical formats like **dta, pkl, ms2 or mgf**.

Search Engine Output

- Engines used for performing the identification and quantification of peptides and proteins.
- **mzIdentML** provides a common format for the export of identification results from any search engine.
- **mzTab** represents both identification and basic quantification results.

Proteomics Data Layers

Peptide Identifications

- Proteomics mass spectra can be matched to peptides or proteins, resulting in identifications for those spectra.
- Typically a spectrum is considered to have been identified if the score attributed to a peptide or protein match qualifies against an *a priori* or *a posteriori* defined threshold.

Protein Identifications

- The protein assembly step can be a discernible process with its own input and output files, or it can be implicit in the overall identification software.

Protein/Peptide Quantification

- Protein/peptide expression values can also be obtained from an MS-based proteomics experiment and then this data and metadata is used for performing the quantification analysis of peptides and proteins.

Meta-data

- Data that provides additional information about a particular data set. This information can include how, when and where the data set was generated and what standards were used.

Importing Third-party Quantitation Data

In most cases, especially as biologists, you usually won't work directly with raw data. The PSI mzTab file format is aimed at providing a simpler (than XML formats) and more accessible file format to the wider community.

```
mztf <- pxget(px, "F063721.dat-mztab.txt")
(mzt <- readMzTabData(mztf, what = "PEP", version = "0.9"))
```

```
## MSnSet (storageMode: lockedEnvironment)
## assayData: 1528 features, 6 samples
## element names: exprs
## protocolData: none
## phenoData
## sampleNames: sub[1] sub[2] ... sub[6] (6 total)
## varLabels: abundance
## varMetadata: labelDescription ## featureData ## featureNames: 1 2 ... 1528 (1528 total)
…
```

Importing Third-party Quantitation Data

Sections in an mzTab file:

Metadata

- Key-value pairs
- Information about experimental methods and sample

Protein Section

- Table-based
- Basic information about protein identifications

Peptide Section

- Table-based
- Aggregates quantitative information on peptide level
- Only recommended in “Quantitation” files

PSM Section

- Table-based
- Basic information about peptide identifications
- Can reference external spectra

Small Molecule Section

- Table-based
- Basic information about small molecule identifications
- Can reference external spectra

The mzTab format is not intended to store the complete experimental evidence but provides mechanisms to report results at different levels of detail. These range from a simple summary of the final results to a representation of the results including the experimental design.

Source: https://www.ncbi.nlm.nih.gov/pubmed/24980485

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Summary

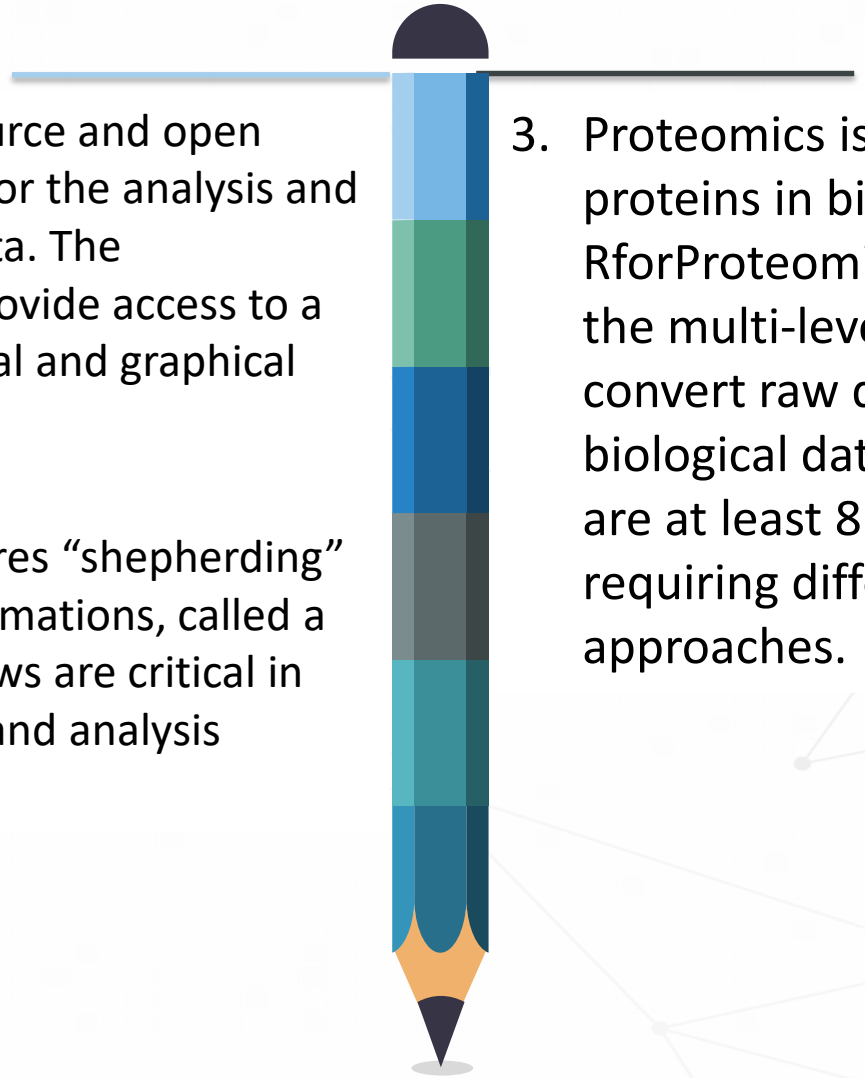
BS3033 Data Science for Biologists

Dr Wilson Goh
School of Biological Sciences

Key Takeaways from this Topic

1. Bioconductor is a free, open source and open development software project for the analysis and comprehension of biological data. The Bioconductor project aims to provide access to a wide range of powerful statistical and graphical methods.

2. Analysis of biological data requires “shepherding” files through a series of transformations, called a pipeline or a workflow. Workflows are critical in biological data analysis as data and analysis objectives are both complex.



3. Proteomics is the high-throughput analysis of proteins in biological sample. Using the RforProteomics suite, you should appreciate the multi-leveled complexity of trying to convert raw data into meaningful analysable biological data. In proteomics alone, there are at least 8 different data levels, each requiring different processing and analysis approaches.