HiTMaP

– An R package of High-resolution Informatics Toolbox for Maldi-imaging Proteomics

## Package installation

This is an tutorial for use of HiTMaP (An R package of High-resolution Informatics Toolbox for Maldi-imaging Proteomics). To access the software use the installation codes as below:

### Installation from Docker image

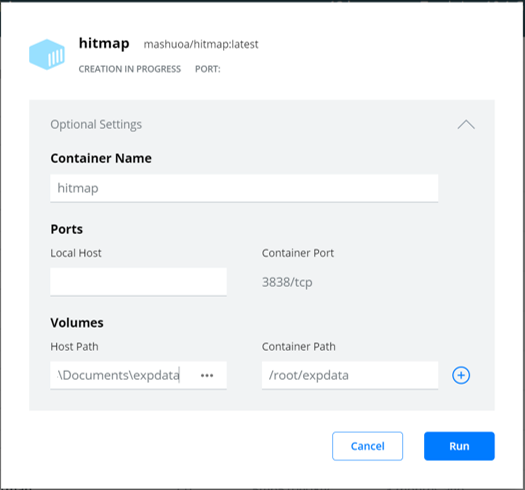
HiTMaP has been encapsulated as an docker image for each release. User can download the latest version by using the code as below.

docker login --username mashuoa  
0a9da0ae-8d7b-4e11-8587-be46e21ee937  
docker pull mashuoa/hitmap

Setup and run the container:

#for windows user, run the image with a local user\Documents\expdata folder mapped to the docker container:  
docker run --name hitmap -v %userprofile%\Documents\expdata:/root/expdata -a stdin -a stdout -i -t mashuoa/hitmap /bin/bash   
  
#for linux user, run the image with a local user/expdata folder mapped to the docker container:  
docker run --name hitmap -v ~/expdata:/root/expdata -a stdin -a stdout -i -t mashuoa/hitmap /bin/bash   
  
#Restart the shell   
docker container exec -it hitmap /bin/bash  
  
#Run R console  
R

If You are using docker GUI, pull the docker image using the codes above and follow the image as below to setup the container.



Docker GUI setting

### Installation code for R console installation

#install the git package  
install.packages("remotes")  
install.packages("devtools")  
#library(devtools)  
library(remotes)  
Sys.setenv("R\_REMOTES\_NO\_ERRORS\_FROM\_WARNINGS" = "true")  
remotes::install\_github("MASHUOA/HiTMaP",auth\_token ="cf6d877f8b6ada1865987b13f9e9996c1883014a",force=T)  
3  
no  
#Update all dependencies  
BiocManager::install(ask = F)  
yes  
library(HiTMaP)

For windows users, Rtools ([*https://cran.r-project.org/bin/windows/Rtools/*](https://cran.r-project.org/bin/windows/Rtools/)) is required.

### Codes for Linux OS building enviornment

Run the codes as below to enable the required components in Linux console.

sudo apt-get install tcl-dev tk-dev  
sudo apt-get install r-cran-ncdf4  
apt-get install libz-dev  
sudo apt install libxml2-dev  
sudo apt install libssl-dev  
sudo apt install libcurl4-openssl-dev  
sudo apt-get install libnss-winbind winbind  
sudo apt install dirmngr gnupg apt-transport-https ca-certificates software-properties-common  
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys  
sudo add-apt-repository 'deb https://cloud.r-project.org/bin/linux/ubuntu focal-cran40/'  
sudo apt-cache policy r-base  
apt-get purge r-base  
sudo apt-get install r-base-core="4.0.2-1.2004.0"  
sudo apt-get install libmagick++-dev  
apt-get install libfftw3-dev  
sudo apt-get install r-base-dev texlive-full  
sudo apt-get install libudunits2-dev  
sudo apt-get install libgdal-dev

### Codes for Mac OS building enviornment

You may need to update the Xcode. Go to your Mac OS terminal and input:

xcode-select --install

You’ll then receive: *xcode-select: note: install requested for command line developer tools* You will be prompted at this point in a window to update Xcode Command Line tools.

You may also need to install the X11.app and tcl/tk support for Mac system:

* X11.app: <https://www.xquartz.org/>
* Use the following link to download and install the correct tcltk package for your OS version. <https://cran.r-project.org/bin/macosx/tools/>

## Example data

The HitMaP comes with a series of Maildi imaging data sets acquired from either FT-ICR or TOF. By the following codes, you can download these raw data set into a local folder.

You can download the example file manually through this link: “<https://github.com/MASHUOA/HiTMaP/releases/download/1.0/Data.tar.gz>”

Or download the files in a R console:

if(!require(piggyback)) install.packages("piggyback")  
library(piggyback)  
  
Sys.setenv(GITHUB\_TOKEN="cf6d877f8b6ada1865987b13f9e9996c1883014a")  
  
#made sure that this folder has enough space  
wd="~/expdata/"  
dir.create(wd)  
setwd(wd)  
pb\_download("Data.tar.gz", repo = "MASHUOA/HiTMaP", dest = ".")  
untar('Data.tar.gz',exdir =".", tar="tar")  
  
#unlink('Data.tar.gz')  
list.dirs()

The example file contains three folder for three IMS dataset, configuration files, and the fasta database, respectively: *“./Bovinlens\_Trypsin\_FT” “./MouseBrain\_Trypsin\_FT” “./Peptide\_calibrants\_FT”*

## Proteomics identification on Maldi imaging data file

Now the HiTMaP is upon running. You could build the candidate list of your target proteome and perform image identification by using the function as below:

#creat candidate list  
library(HiTMaP)  
#set project folder that contains imzML, .ibd and fasta files  
#wd=paste0(file.path(path.package(package="HiTMaP")),"/data/")  
#set a series of imzML files to be processed  
datafile=c("Bovinlens\_Trypsin\_FT/Bovin\_lens.imzML")  
wd="~/expdata/"  
  
  
imaging\_identification(  
#==============Choose the imzml raw data file(s) to process make sure the fasta file in the same folder  
 datafile=paste0(wd,datafile),  
 threshold=0.005,   
 ppm=5,  
 FDR\_cutoff = 0.05,  
#==============specify the digestion enzyme specificity  
 Digestion\_site="trypsin",  
#==============specify the range of missed Cleavages  
 missedCleavages=0:1,  
#==============Set the target fasta file  
 Fastadatabase="uniprot-bovin.fasta",  
#==============Set the possible adducts and fixed modifications  
 adducts=c("M+H"),  
 Modifications=list(fixed=NULL,fixmod\_position=NULL,variable=NULL,varmod\_position=NULL),  
#==============The decoy mode: could be one of the "adducts", "elements" or "isotope"  
 Decoy\_mode = "isotope",  
 use\_previous\_candidates=F,  
 output\_candidatelist=T,  
#==============The pre-processing param  
 preprocess=list(force\_preprocess=TRUE,  
 use\_preprocessRDS=TRUE,  
 smoothSignal=list(method="gaussian"),  
 reduceBaseline=list(method="locmin"),  
 peakPick=list(method="adaptive"),  
 peakAlign=list(tolerance=ppm, units="ppm"),  
 normalize=list(method=c("rms","tic","reference")[1],mz=1)),  
#==============Set the parameters for image segmentation  
 spectra\_segments\_per\_file=4,  
 Segmentation="spatialKMeans",  
 Smooth\_range=1,  
 Virtual\_segmentation=FALSE,  
 Virtual\_segmentation\_rankfile=NULL,  
#==============Set the Score method for hi-resolution isotopic pattern matching  
 score\_method="SQRTP",  
 peptide\_ID\_filter=2,  
#==============Summarise the protein and peptide features across the project the result can be found at the summary folder  
 Protein\_feature\_summary=TRUE,  
 Peptide\_feature\_summary=TRUE,  
 Region\_feature\_summary=TRUE,  
#==============The parameters for Cluster imaging. Specify the annotations of interest, the program will perform a case-insensitive search on the result file, extract the protein(s) of interest and plot them in the cluster imaging mode  
 plot\_cluster\_image\_grid=FALSE,  
 ClusterID\_colname="Protein",  
 componentID\_colname="Peptide",  
 Protein\_desc\_of\_interest=c("Crystallin","Actin"),  
 Rotate\_IMG=NULL,  
 )

## Project folder and result structure

In the above function, You have performed proteomics analysis of the sample data file. It is a tryptic Bovin lens MALDI-imaging file which is acquired on an FT-ICR MS. The function will take the selected data files’ root directory as the project folder. In this example, the project folder will be:

library(HiTMaP)  
wd=paste0("D:\\GITHUB LFS\\HiTMaP-Data\\inst","/data/Bovinlens\_Trypsin\_FT/")  
#set a series of imzML files to be processed  
datafile=c("Bovin\_lens")  
wd

## [1] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT/"

After the whole identification process, we will get two types of sub-folders in the project folder:

list.dirs(wd, recursive=FALSE)

## [1] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT//Bovin\_lens ID"   
## [2] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT//Summary folder"

1. The one which has an identical name to an input data file contains the identification result of that input:
   * the protein and peptides list of each segmentation region
   * the PMF matching plot of each segmentation
   * the image that indicates segmentations’ boundary (applies to either K-mean segmentation (powered by Cardinal) or manually defined segmentation)
   * folders of each region contains the detailed identification process, FDR plots and isotopic pattern matching plots
2. “Summary folder” contains:
   * the identification summary of protein and peptides across all the data
   * the candidate list of all possible proteins and peptides (if *use\_previous\_candidates* is set as **TRUE**)
   * the Cluster imaging files of the protein of interest

## Identification result visulasation and interpretation

Now we could visualize the result by the following functions:

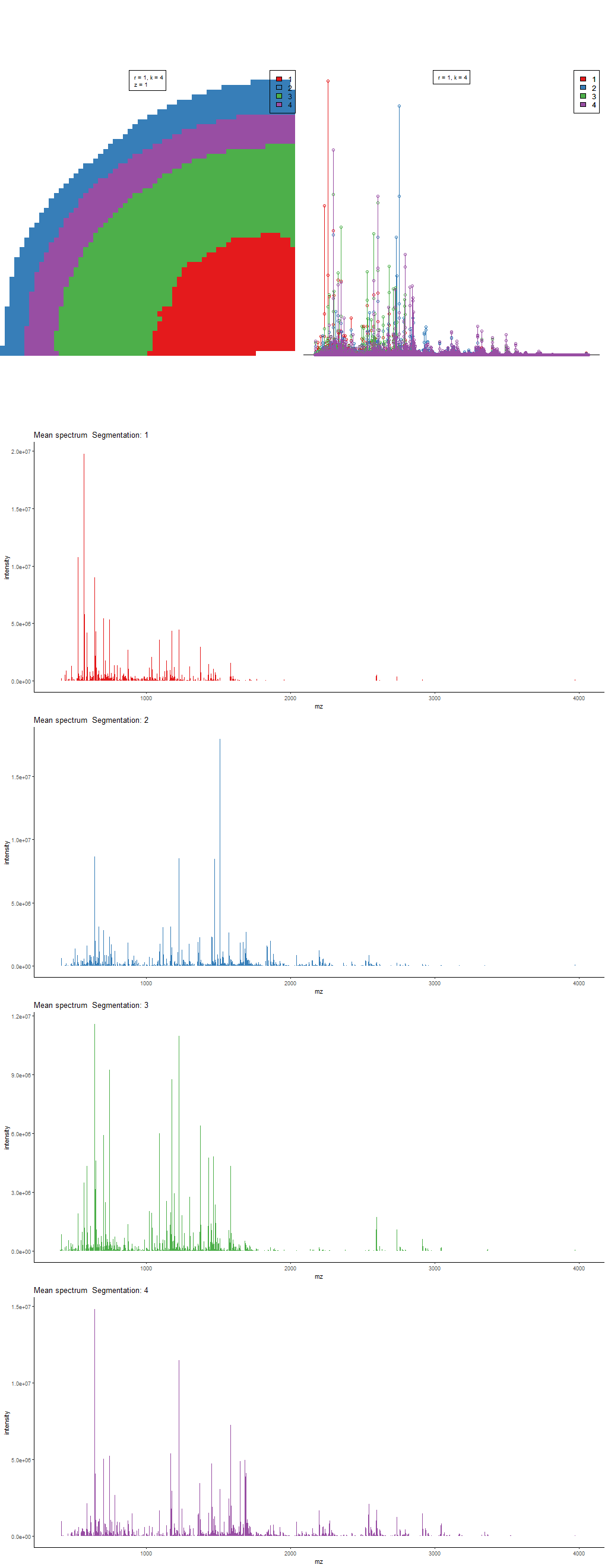
To check the segmentation result over the sample, you need got to each data file ID folder and find the “spatialKMeans\_image\_plot.png” (if you are using the spatial K-means method for segmentation.)

library(magick)

## Linking to ImageMagick 6.9.9.14  
## Enabled features: cairo, freetype, fftw, ghostscript, lcms, pango, rsvg, webp  
## Disabled features: fontconfig, x11

p<-image\_read(paste0(wd,datafile," ID/spatialKMeans\_image\_plot.png"))  
print(p)

## format width height colorspace matte filesize density  
## 1 PNG 1024 2640 sRGB FALSE 30726 72x72

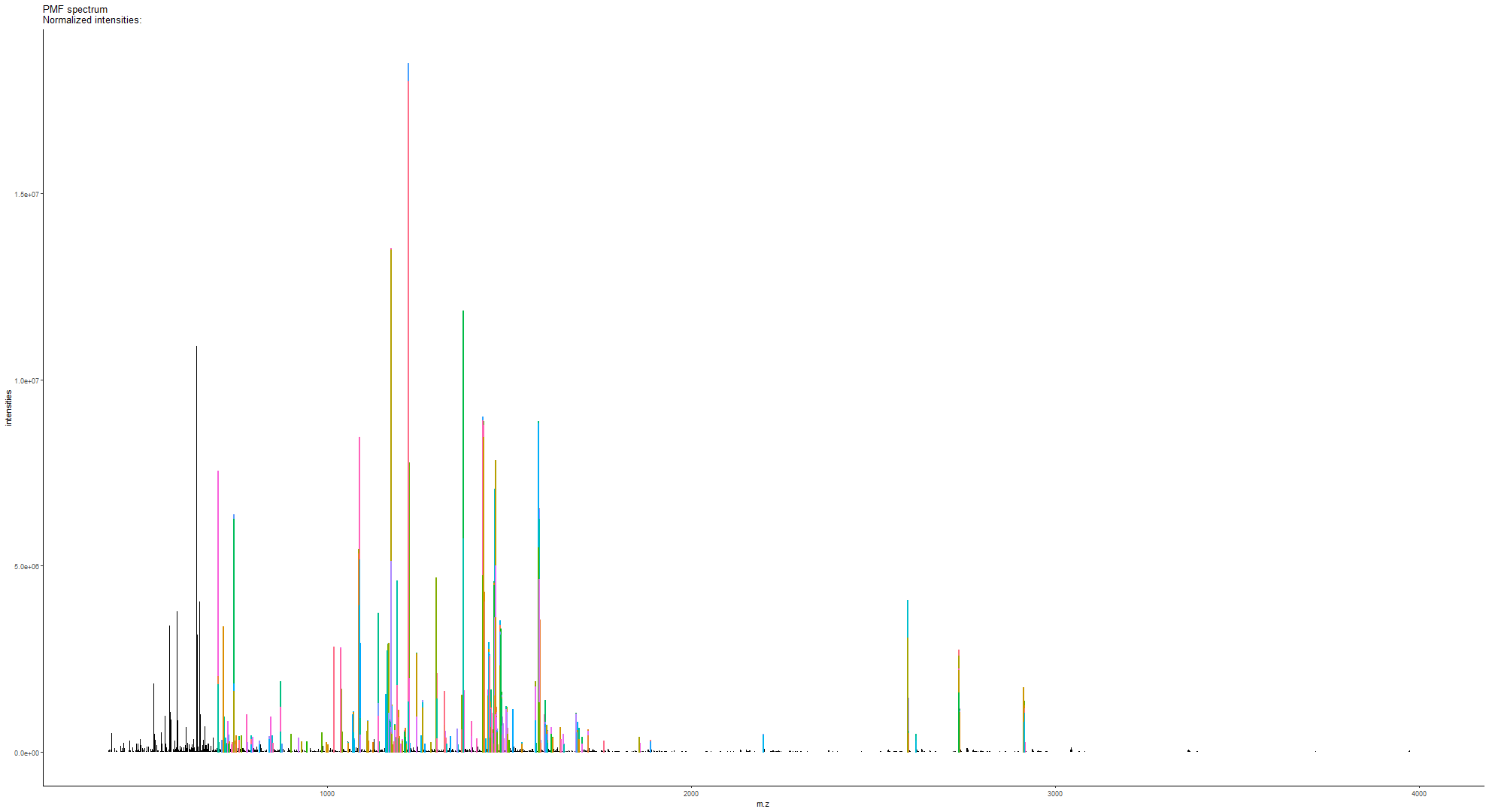


The pixels in image data now has been categorized into five regions according to the initial setting of segmentation (*spectra\_segments\_per\_file=5*). The rainbow shaped bovine lens segmentation image (on the left panel) shows a unique statistical classification based on the mz features of each region (on the right panel).

The identification will take place on the **mean spectra** of each region. To check the peptide mass fingerprint (PMF) matching quality, you could locate the PMF spectrum matching plot of each individual region.

library(magick)  
p\_pmf<-image\_read(paste0(wd,datafile," ID/Bovin\_lens 3PMF spectrum match.png"))  
print(p\_pmf)

## # A tibble: 1 x 7  
## format width height colorspace matte filesize density  
## <chr> <int> <int> <chr> <lgl> <int> <chr>   
## 1 PNG 1980 1080 sRGB FALSE 17664 72x72



list of Peptides and proteins of each region has also been created so that you may check each individual region’s result.

peptide\_pmf\_result<-read.csv(paste0(wd,datafile," ID/Peptide\_segment\_PMF\_RESULT\_3.csv"))  
head(peptide\_pmf\_result)

## Protein mz Protein\_coverage isdecoy Peptide Modification pepmz  
## 1 48 1300.664 0.06875544 0 HLEQFATEGLR NA 1299.657  
## 2 48 1300.661 0.06875544 0 QYFLDLALSCK NA 1299.653  
## 3 48 1324.643 0.06875544 0 GSKCILYCFYK NA 1323.636  
## 4 53 1328.747 0.05542725 0 FKNINPFPVPR NA 1327.740  
## 5 53 1449.712 0.05542725 0 AVQNFTEYNVHK NA 1448.705  
## 6 53 1605.813 0.05542725 0 AVQNFTEYNVHKR NA 1604.806  
## formula adduct charge start end pro\_end mz\_align Score Rank  
## 1 C57H90N17O18 M+H 1 580 590 1149 1300.666 2.4633527 4  
## 2 C60H94N13O17S1 M+H 1 744 754 1149 1300.666 2.0216690 10  
## 3 C62H94N13O15S2 M+H 1 840 850 1149 1324.647 -0.2644896 32  
## 4 C64H98N17O14 M+H 1 207 217 433 1328.747 1.0865820 7  
## 5 C65H97N18O20 M+H 1 92 103 433 1449.714 0.7060553 10  
## 6 C71H109N22O21 M+H 1 92 104 433 1605.806 2.7178547 11  
## moleculeNames Region Delta\_ppm Intensity peptide\_count  
## 1 HLEQFATEGLR 3 0.9026772 4672324.6 3  
## 2 QYFLDLALSCK 3 1.4117311 4672324.6 3  
## 3 GSKCILYCFYK 3 1.5164261 145191.4 3  
## 4 FKNINPFPVPR 3 0.9094769 191636.4 3  
## 5 AVQNFTEYNVHK 3 2.8830137 1275214.1 3  
## 6 AVQNFTEYNVHKR 3 1.6464326 558610.4 3  
## desc.x  
## 1 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## 2 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## 3 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## 4 sp|Q3SX05|ECSIT\_BOVIN Evolutionarily conserved signaling intermediate in Toll pathway, mitochondrial OS=Bos taurus OX=9913 GN=ECSIT PE=2 SV=1  
## 5 sp|Q3SX05|ECSIT\_BOVIN Evolutionarily conserved signaling intermediate in Toll pathway, mitochondrial OS=Bos taurus OX=9913 GN=ECSIT PE=2 SV=1  
## 6 sp|Q3SX05|ECSIT\_BOVIN Evolutionarily conserved signaling intermediate in Toll pathway, mitochondrial OS=Bos taurus OX=9913 GN=ECSIT PE=2 SV=1  
## desc.y  
## 1 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## 2 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## 3 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## 4 sp|Q3SX05|ECSIT\_BOVIN Evolutionarily conserved signaling intermediate in Toll pathway, mitochondrial OS=Bos taurus OX=9913 GN=ECSIT PE=2 SV=1  
## 5 sp|Q3SX05|ECSIT\_BOVIN Evolutionarily conserved signaling intermediate in Toll pathway, mitochondrial OS=Bos taurus OX=9913 GN=ECSIT PE=2 SV=1  
## 6 sp|Q3SX05|ECSIT\_BOVIN Evolutionarily conserved signaling intermediate in Toll pathway, mitochondrial OS=Bos taurus OX=9913 GN=ECSIT PE=2 SV=1

protein\_pmf\_result<-read.csv(paste0(wd,datafile," ID/Protein\_segment\_PMF\_RESULT\_3.csv"))  
head(protein\_pmf\_result)

## Protein Proscore isdecoy Intensity Score peptide\_count Protein\_coverage  
## 1 10134 0.13943597 0 2873903.1 1.9269417 3 0.06715328  
## 2 10204 0.13654123 0 380571.3 0.7940642 3 0.18468468  
## 3 10370 0.20365140 0 1877250.1 2.0776861 4 0.09364548  
## 4 10659 0.11239668 0 327352.4 0.7448240 3 0.16400000  
## 5 10888 0.07975644 0 532832.0 1.2420183 3 0.06720978  
## 6 11270 0.10687770 0 2944154.2 1.3292158 3 0.07449857  
## Intensity\_norm  
## 1 1.0775539  
## 2 0.9310593  
## 3 1.0466962  
## 4 0.9201443  
## 5 0.9554442  
## 6 1.0793038  
## desc  
## 1 tr|G3N2M8|G3N2M8\_BOVIN Sterile alpha motif domain containing 15 OS=Bos taurus OX=9913 GN=SAMD15 PE=4 SV=2  
## 2 tr|A0A3Q1LYB6|A0A3Q1LYB6\_BOVIN Uncharacterized protein OS=Bos taurus OX=9913 PE=4 SV=1  
## 3 tr|E1B9U7|E1B9U7\_BOVIN Polypeptide N-acetylgalactosaminyltransferase OS=Bos taurus OX=9913 GN=GALNT17 PE=3 SV=3  
## 4 tr|A0A3Q1M1B1|A0A3Q1M1B1\_BOVIN Phosphatidylinositol transfer protein beta isoform OS=Bos taurus OX=9913 GN=PITPNB PE=4 SV=1  
## 5 tr|F1MMD4|F1MMD4\_BOVIN Matrix metallopeptidase 11 OS=Bos taurus OX=9913 GN=MMP11 PE=3 SV=2  
## 6 tr|F6RR01|F6RR01\_BOVIN Ribosome production factor 1 homolog OS=Bos taurus OX=9913 GN=RPF1 PE=4 SV=1

## Scoring system for protein and peptide

**Score** in peptide result table shows the isotopic pattern matching score of the peptide (Pepscore). In Protein result table, it shows the protein score (Proscore). The Pepscore consist of two parts: Intensity\_Score and Mass\_error\_Score:

* Intensity\_Score indicates how well a putative isotopic pattern can be matched to the observed spectrum.The default scoring method is SQRTP. It combines the Square root mean differences between observed and theoretical peaks and observed proportion of the isotopic peaks above a certain relative intensity threshold.
* Mass\_error\_Score indicates the summary of mass error (in *ppm*) for every detected isotopic peak. in order to integrate the Mass\_error\_Score in to scoring system. the mean ppm error has been normalized by ppm tolerance, and supplied to the probability normal distributions (*pnorm* function for R). The resulting value (Quantiles of the given Probability Density) is deducted by 0.5 and converted into absolute value.

**Proscore** in the protein result table shows the overall estimation of the protein identification Accuracy

A *Peptide\_region\_file.csv* has also been created to summarise all the IDs in this data file:

Identification\_summary\_table<-read.csv(paste0(wd,datafile," ID/Peptide\_region\_file.csv"))  
head(Identification\_summary\_table)

## Protein mz Protein\_coverage isdecoy Peptide Modification  
## 1 24 1143.5793 0.06119704 0 GFPGQDGLAGPK NA  
## 2 24 1684.8878 0.06119704 0 DGANGIPGPIGPPGPRGR NA  
## 3 24 742.3478 0.06119704 0 GDSGPPGR NA  
## 4 24 1693.8214 0.06119704 0 LLSTEGSQNITYHCK NA  
## 5 24 1881.9276 0.06119704 0 GQPGVMGFPGPKGANGEPGK NA  
## 6 48 1216.7008 0.03481288 0 ASTSVQNRLLK NA  
## pepmz formula adduct charge start end pro\_end mz\_align  
## 1 1142.5720 C51H79N14O16 M+H 1 516 527 1487 1143.5828  
## 2 1683.8805 C72H118N25O22 M+H 1 1175 1192 1487 1684.8830  
## 3 741.3406 C29H48N11O12 M+H 1 933 940 1487 742.3504  
## 4 1692.8141 C72H117N20O25S1 M+H 1 1380 1394 1487 1693.8197  
## 5 1880.9203 C82H129N24O25S1 M+H 1 597 616 1487 1881.9268  
## 6 1215.6935 C51H94N17O17 M+H 1 614 624 1149 1216.7047  
## Score Rank moleculeNames Region Delta\_ppm Intensity peptide\_count  
## 1 1.4443497 2 GFPGQDGLAGPK 2 1.3471596 250698.3 5  
## 2 1.9337304 2 DGANGIPGPIGPPGPRGR 2 1.5937657 2696717.3 5  
## 3 1.2698949 1 GDSGPPGR 2 0.1407633 190469.7 5  
## 4 1.3660521 3 LLSTEGSQNITYHCK 2 2.2329023 368927.9 5  
## 5 0.5868561 17 GQPGVMGFPGPKGANGEPGK 2 3.0817671 974427.3 5  
## 6 1.9039495 1 ASTSVQNRLLK 2 1.8837090 2036000.7 1  
## desc.x  
## 1 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 2 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 3 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 4 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 5 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 6 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2  
## desc.y  
## 1 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 2 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 3 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 4 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 5 sp|P02459|CO2A1\_BOVIN Collagen alpha-1(II) chain OS=Bos taurus OX=9913 GN=COL2A1 PE=1 SV=4  
## 6 sp|Q29449|AT8A1\_BOVIN Probable phospholipid-transporting ATPase IA OS=Bos taurus OX=9913 GN=ATP8A1 PE=1 SV=2

The details of protein/peptide identification process has been save to the folder named by the segmentation:

list.dirs(paste0(wd,datafile," ID/"), recursive=FALSE)

## [1] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT/Bovin\_lens ID//1"  
## [2] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT/Bovin\_lens ID//2"  
## [3] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT/Bovin\_lens ID//3"  
## [4] "D:\\GITHUB LFS\\HiTMaP-Data\\inst/data/Bovinlens\_Trypsin\_FT/Bovin\_lens ID//4"

In the identification details folder, you will find a series of FDR file and plots to demonstrate the FDR model and score cutoff threshold:

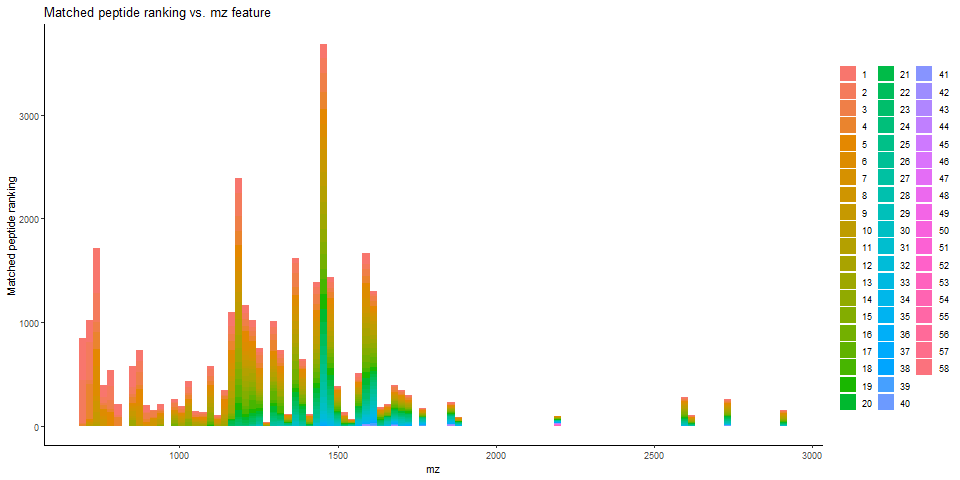
dir(paste0(wd,datafile," ID/1/"), recursive=FALSE)

## [1] "FDR.CSV"   
## [2] "FDR.png"   
## [3] "Matching\_Score\_vs\_mz\_target-decoy.png"   
## [4] "Peptide\_1st\_ID.csv"   
## [5] "Peptide\_1st\_ID\_score\_rank\_SQRTP.csv"   
## [6] "Peptide\_2nd\_ID\_score\_rankSQRTP\_Rank\_above\_3.csv"  
## [7] "Peptide\_Score\_histogram\_target-decoy.png"   
## [8] "ppm"   
## [9] "PROTEIN\_FDR.CSV"   
## [10] "Protein\_FDR.png"   
## [11] "Protein\_ID\_score\_rank\_SQRTP.csv"   
## [12] "PROTEIN\_Score\_histogram.png"   
## [13] "Spectrum.csv"   
## [14] "unique\_peptide\_ranking\_vs\_mz\_feature.png"

In this folder, you will find the FDR plots for protein and peptide. The software will take the proscore and its FDR model to trim the final identification result. The *unique\_peptide\_ranking\_vs\_mz\_feature.png* is a plot that could tell you the number of peptide candidates have been matched to the mz features in the first round run.You can also access the peptide spectrum match (first MS dimension) data via the “/ppm” subfolder.

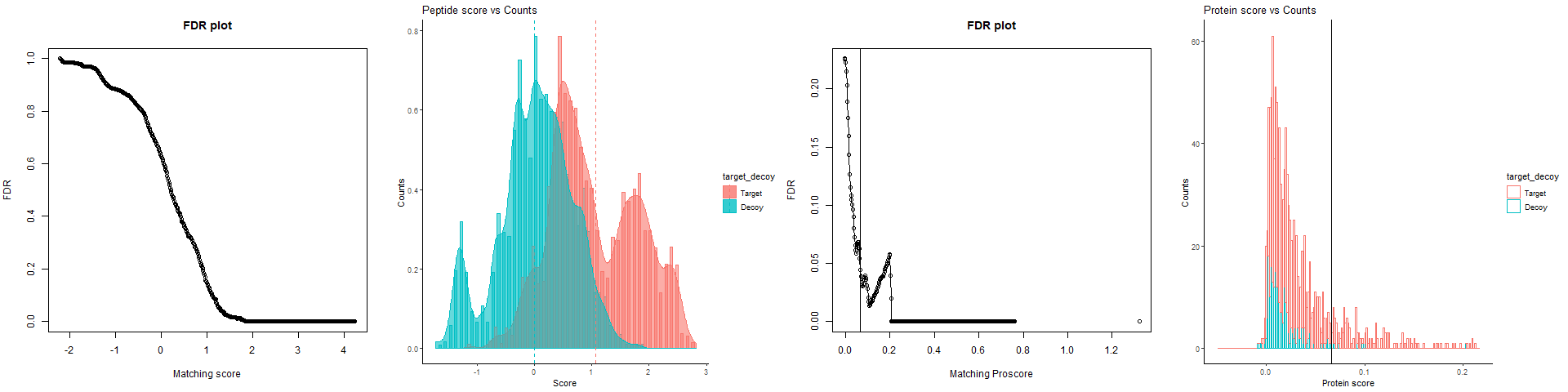
library(magick)  
p\_peptide\_vs\_mz\_feature<-image\_read(paste0(wd,datafile," ID/3/unique\_peptide\_ranking\_vs\_mz\_feature.png"))  
print(p\_peptide\_vs\_mz\_feature)

## format width height colorspace matte filesize density  
## 1 PNG 960 480 sRGB FALSE 11196 72x72



p\_FDR\_peptide<-image\_read(paste0(wd,datafile," ID/3/FDR.png"))  
p\_FDR\_protein<-image\_read(paste0(wd,datafile," ID/3/protein\_FDR.png"))  
p\_FDR\_peptide\_his<-image\_read(paste0(wd,datafile," ID/3/Peptide\_Score\_histogram\_target-decoy.png"))  
p\_FDR\_protein\_his<-image\_read(paste0(wd,datafile," ID/3/PROTEIN\_Score\_histogram.png"))  
p\_combined<-image\_append(c(p\_FDR\_peptide,p\_FDR\_peptide\_his,p\_FDR\_protein,p\_FDR\_protein\_his))  
print(p\_combined)

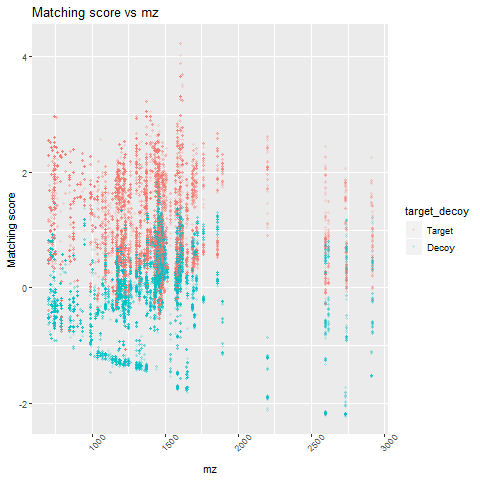
## format width height colorspace matte filesize density  
## 1 PNG 1920 480 sRGB FALSE 0 72x72



you will also find a *Matching\_Score\_vs\_mz* plots for further investigation on peptide matching quality.

library(magick)  
#plot Matching\_Score\_vs\_mz  
p\_Matching\_Score\_vs\_mz<-image\_read(paste0(wd,datafile," ID/3/Matching\_Score\_vs\_mz\_target-decoy.png"))  
print(p\_Matching\_Score\_vs\_mz)

## format width height colorspace matte filesize density  
## 1 PNG 480 480 sRGB FALSE 47438 72x72



## Identification summary and cluster imaging

In the project summary folder, you will find four files and a sub-folder:

wd\_sum=paste(wd,"/Summary folder", sep="")  
dir(wd\_sum)

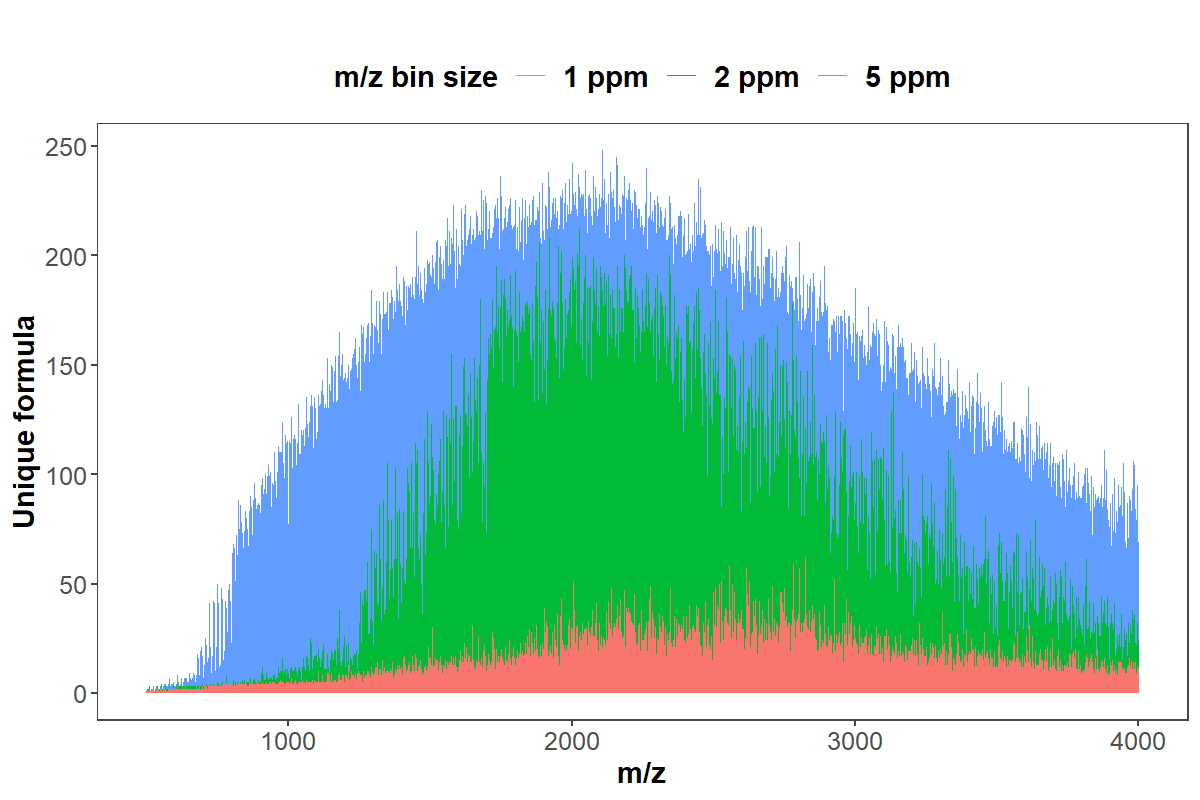
## [1] "candidatelist.csv" "cluster Ion images"   
## [3] "Peptide\_Summary.csv" "Protein\_feature\_list\_trimmed.csv"  
## [5] "protein\_index.csv" "Protein\_Summary.csv"

“candidatelist.csv” and “protein\_index.csv” contains the candidates used for this project. They are output after the candidate processing while *output\_candidatelist* set as TRUE, and can be used repeatedly while *use\_previous\_candidates* set as TRUE.

we have now implemented a functionality to perform additional statistical analyses around the number of tryptic enzymatically generated peptides generated derived from a given proteome (‘Database\_stats’). If the user sets the variable ‘Database\_stats’ to TRUE in the main workflow, then the function will be called.

Briefly, the function will list all of the m/z’s of a unique formulae from the peptide candidate pool within a given m/z range. The m/z’s will then be binned using three resolution m/z windows: 1ppm, 2ppm and 5ppm. A plot showing the number of unique formulae vs. binned m/z windows will be generated and exported to the summary folder (DB\_stats\_mz\_bin).

The example figure as below shows the m/z bin(s) analysis result of a mouse brain proteome without modification(s) and with up to 1 missed cleavage(D). This represents a ‘worst case scenario’ since there is an assumption that all competitive candidates in each bin have equal ionisability, which in practice is not the case. The reviewer is therefore correct that a significant number of competitive candidates can be found within the m/z range of a common proteomics/peptidomics investigation. In practical terms, the number of competitive candidates is likely to be far fewer, due to the previously mentioned unequal ionisation of predicted peptides and the known bias for MALDI MSI to detect higher abundance peptides and proteins.



Proteome database stats

However, one of the main applications of HIT-MAP is the annotation of proteins and their spatial distribution. Since only unique peptides are retained, and all peptides are rank-scored, and protein maps are only produced when 2+ unique peptides are matched, we feel that the likelihood of misidentifying proteins significantly decreases with increasing peptide number.

“Peptide\_Summary.csv” and “Protein\_Summary.csv” contains the table of the project identification summary. You could set the *plot\_cluster\_image\_grid* as TRUE to enable the cluster imaging function. Please be noted that you could indicate *Rotate\_IMG* with a CSV file path that indicates the rotation degree of image files.

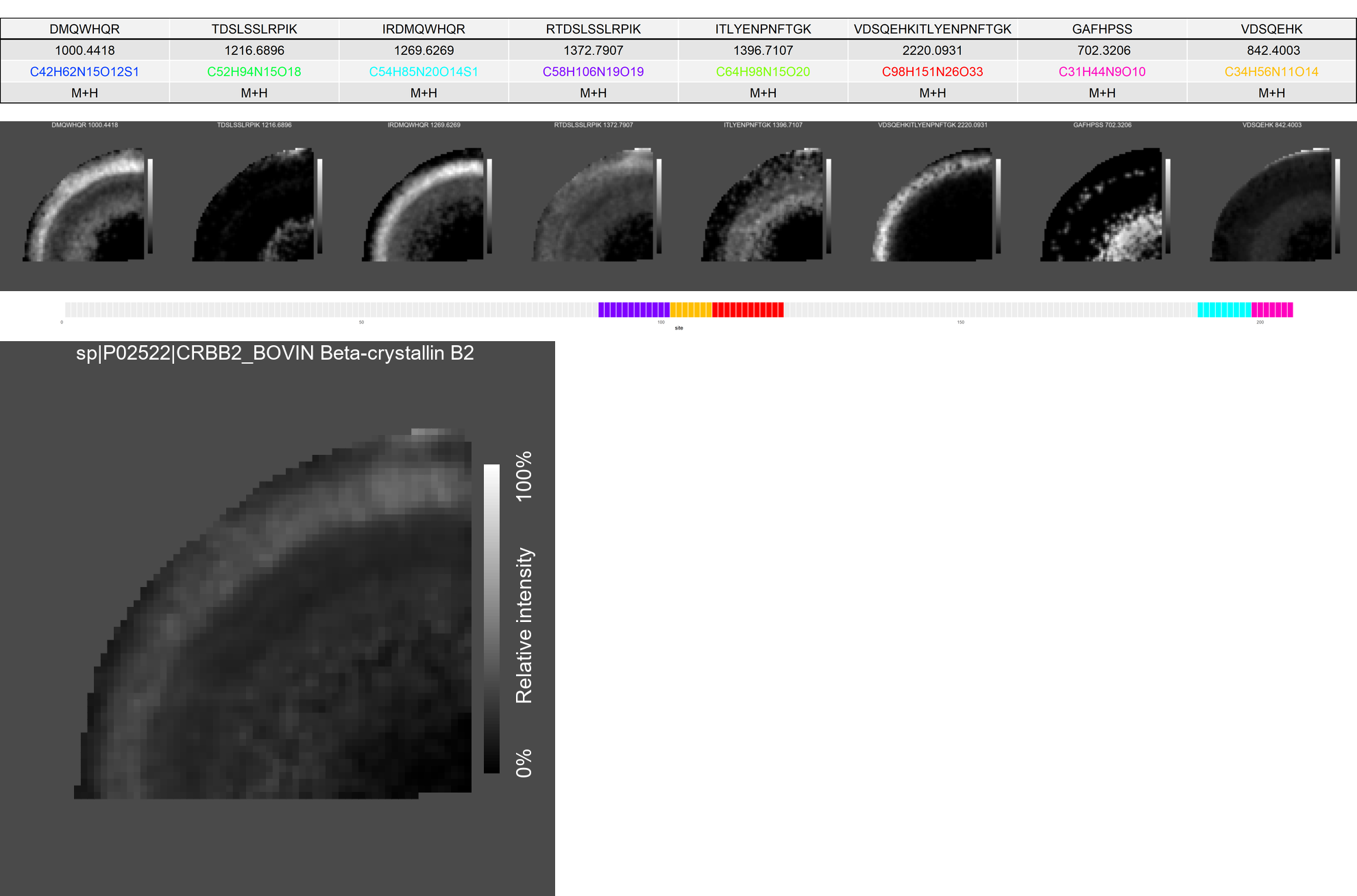
**Note**: 90, 180 and 270 are recommended for image rotation. You may find an example CSV file in the library/HiTMaP/data folder.

library(dplyr)  
Protein\_desc\_of\_interest<-c("Crystallin","Actin")  
Protein\_Summary\_tb<-read.csv(paste(wd,"/Summary folder","/Protein\_Summary.csv", sep=""),stringsAsFactors = F)

Now you could visualized the interest proteins and their associated peptides’ distribution via cluster imaging function.

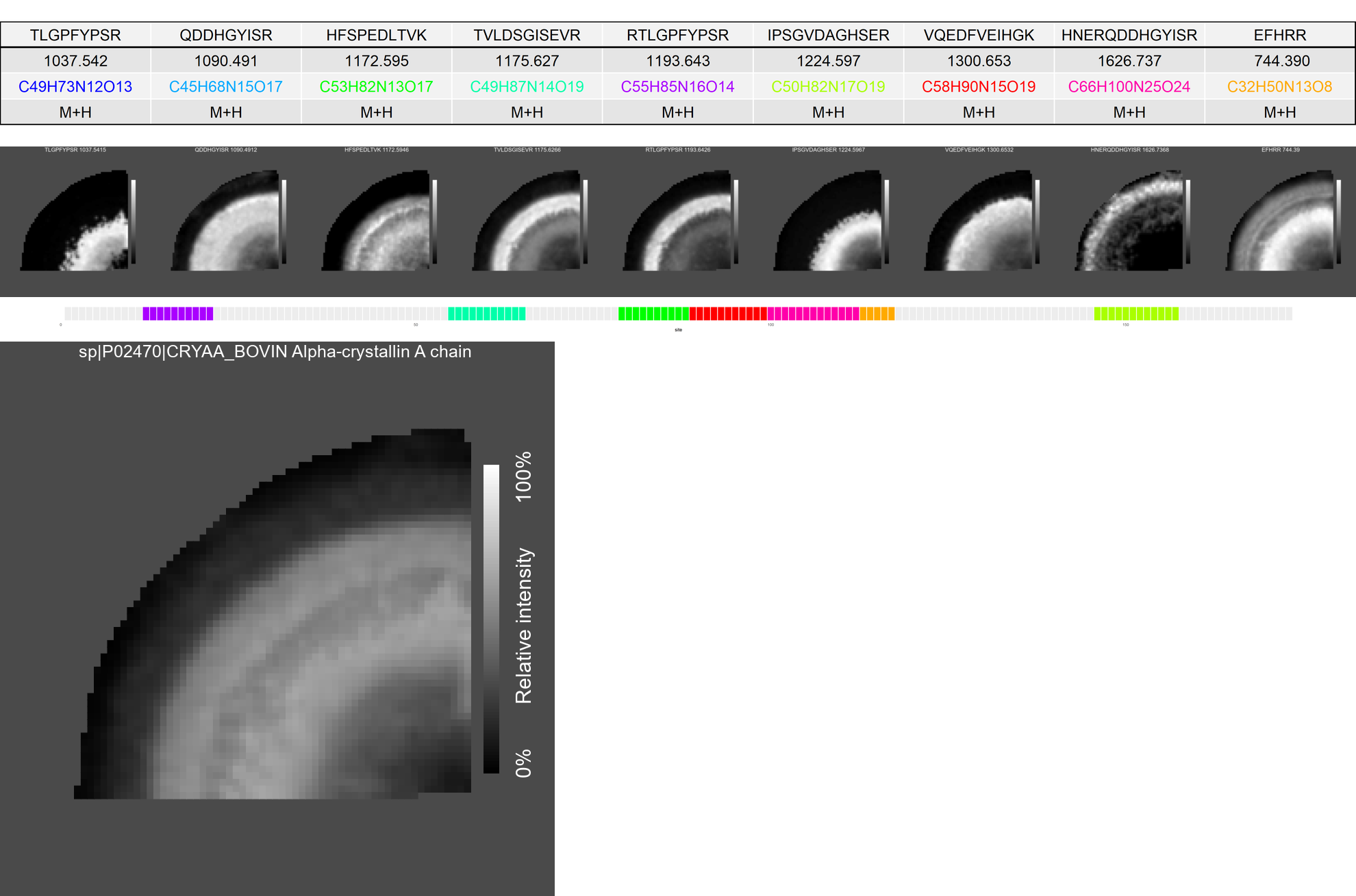
library(magick)  
p\_cluster1<-image\_read(paste0(wd,"/Summary folder/cluster Ion images/791\_cluster\_imaging.png"))  
print(p\_cluster1)

## # A tibble: 1 x 7  
## format width height colorspace matte filesize density  
## <chr> <int> <int> <chr> <lgl> <int> <chr>   
## 1 PNG 1980 1308 sRGB TRUE 302087 118x118



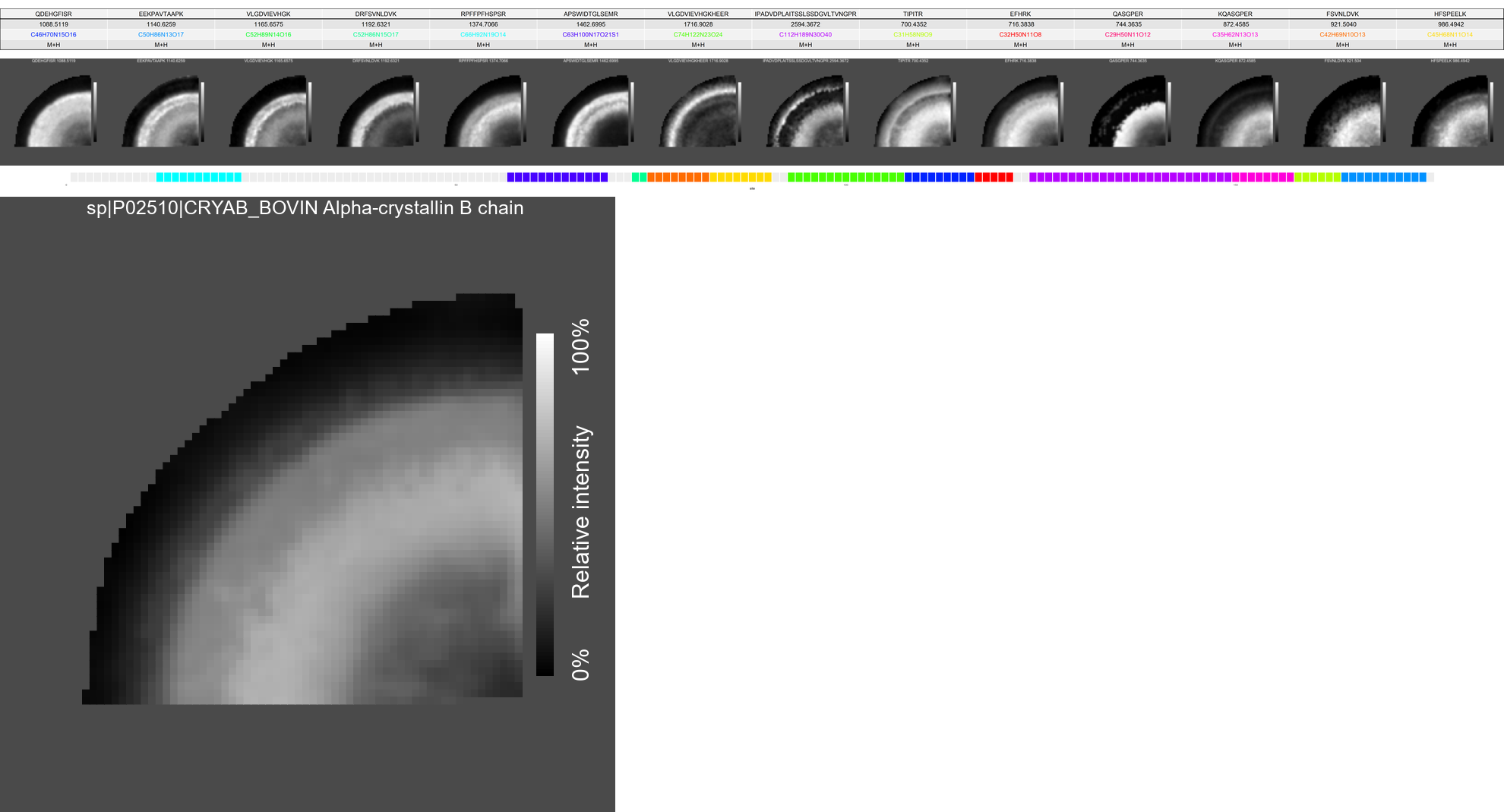
p\_cluster2<-image\_read(paste0(wd,"/Summary folder/cluster Ion images/5027\_cluster\_imaging.png"))  
print(p\_cluster2)

## # A tibble: 1 x 7  
## format width height colorspace matte filesize density  
## <chr> <int> <int> <chr> <lgl> <int> <chr>   
## 1 PNG 1980 1309 sRGB TRUE 348111 118x118



p\_cluster3<-image\_read(paste0(wd,"/Summary folder/cluster Ion images/5479\_cluster\_imaging.png"))  
print(p\_cluster3)

## # A tibble: 1 x 7  
## format width height colorspace matte filesize density  
## <chr> <int> <int> <chr> <lgl> <int> <chr>   
## 1 PNG 1980 1069 sRGB TRUE 237191 118x118



## Details of parameter setting

### Modification

you can choose one or a list of modifications from the unimod modification list. *Peptide\_modification* function is used to load/rebuild the modification database into the global enviornment of R. It will be called automatically in the identification work flow. you can use the *code\_name* or *record\_id* to refer the modification (see example data “peptide calibrants” to find more details). The pipeline will select the *non-hidden* modifications.

HiTMaP:::Peptide\_modification(retrive\_ID=NULL,update\_unimod=F)  
modification\_list<-merge(unimod.df$modifications,unimod.df$specificity,by.x=c("record\_id"),by.y=c("mod\_key"),all.x=T)  
head(modification\_list['&'(modification\_list$code\_name=="Phospho",modification\_list$hidden!=1),c("code\_name","record\_id","composition","mono\_mass","position\_key","one\_letter")])

## code\_name record\_id composition mono\_mass position\_key one\_letter  
## 1615 Phospho 21 H O(3) P 79.966331 2 Y  
## 1618 Phospho 21 H O(3) P 79.966331 2 T  
## 1620 Phospho 21 H O(3) P 79.966331 2 S

head(modification\_list['&'(modification\_list$code\_name=="Amide",modification\_list$hidden!=1),c("code\_name","record\_id","composition","mono\_mass","position\_key","one\_letter")])

## code\_name record\_id composition mono\_mass position\_key one\_letter  
## 1552 Amide 2 H N O(-1) -0.984016 6 C-term  
## 1553 Amide 2 H N O(-1) -0.984016 4 C-term

head(modification\_list['&'(stringr::str\_detect(modification\_list$code\_name,"Ca"),modification\_list$hidden!=1),c("code\_name","record\_id","composition","mono\_mass","position\_key","one\_letter")])

## code\_name record\_id composition mono\_mass position\_key one\_letter  
## 1946 Carbamidomethyl 4 H(3) C(2) N O 57.021464 2 C  
## 1949 Carbamidomethyl 4 H(3) C(2) N O 57.021464 3 N-term  
## 2119 Carbamyl 5 H C N O 43.005814 3 N-term  
## 2121 Carbamyl 5 H C N O 43.005814 2 K  
## 2271 Carboxymethyl 6 H(2) C(2) O(2) 58.005479 2 C

If a modification occurs on different types of site , you will also need to specify the position of a modifications.

* *Anywhere*, side chain of possible amino acids
* *Any N-term*, any N-term of enzymatic peptide
* *Protein N-term*, any N-term of protein

unimod.df[["positions"]]

## position record\_id  
## 1 - 1  
## 2 Anywhere 2  
## 3 Any N-term 3  
## 4 Any C-term 4  
## 5 Protein N-term 5  
## 6 Protein C-term 6

### Amino acid substitution

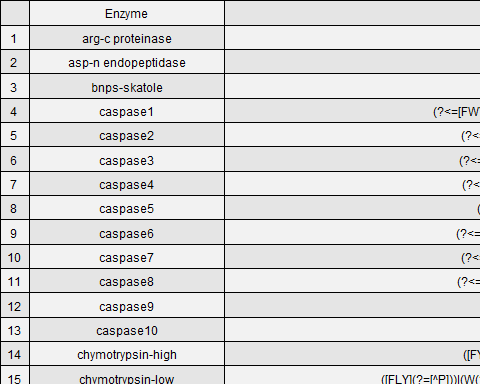
You can set the *Substitute\_AA* to make the uncommon amino acid available to the workflow: *Substitute\_AA=list(AA=c(“X”),AA\_new\_formula=c(“C5H5NO2”),Formula\_with\_water=c(FALSE))*

* AA: the single letter amino acid to be replaced
* AA\_new\_formula: the new formula for the amino acid
* Formula\_with\_water: Set *TRUE* to indicate the formula represents the intact amino acid, *FALSE* to indicate that the formula already lost one H2O molecule and can be considered as AA backbone.

### Digestion site and enzyme

The *Digestion\_site* allows you to specify a list of pre-defined enzyme and customized digestion rules in regular expression format. You can either use the enzyme name, customized cleavage rule or combination of them to get the enzymatics peptides list.

Cleavage\_rules<-Cleavage\_rules\_fun()  
Cleavage\_df<-data.frame(Enzyme=names(Cleavage\_rules),Cleavage\_rules=unname(Cleavage\_rules),stringsAsFactors = F)  
library(gridExtra)  
grid.ftable(Cleavage\_df, gp = gpar(fontsize=9,fill = rep(c("grey90", "grey95"))))



## Example workflow command

Below is a list of commands including the parameters for the example data sets.

### Peptide calibrant

#peptide calibrant  
library(HiTMaP)  
datafile=c("Peptide\_calibrants\_FT/trypsin\_non-decell\_w.calibrant\_FTICR")  
wd="~/expdata/"  
  
# Calibrants dataset analysis with modification  
imaging\_identification(datafile=paste0(wd,datafile),  
 Digestion\_site="trypsin",  
 Fastadatabase="uniprot\_cali.fasta",  
 output\_candidatelist=T,  
 plot\_matching\_score=T,  
 spectra\_segments\_per\_file=1,  
 use\_previous\_candidates=F,  
 peptide\_ID\_filter=1,ppm=5,missedCleavages=0:5,  
 Modifications=list(fixed=NULL,fixmod\_position=NULL,variable=c("Amide"),varmod\_position=c(6)),  
 FDR\_cutoff=0.1,  
 Substitute\_AA=list(AA=c("X"),AA\_new\_formula=c("C5H5NO2"),Formula\_with\_water=c(FALSE)))  
  
# Calibrants dataset analysis with no modification  
imaging\_identification(datafile=paste0(wd,datafile),  
 Digestion\_site="trypsin",  
 Fastadatabase="uniprot\_cali.fasta",  
 output\_candidatelist=T,  
 plot\_matching\_score=T,  
 spectra\_segments\_per\_file=1,  
 use\_previous\_candidates=T,  
 peptide\_ID\_filter=1,ppm=5,missedCleavages=0:5,  
 FDR\_cutoff=0.1)  
  
# Calibrants dataset analysis with modification   
imaging\_identification(datafile=paste0(wd,datafile),  
 Digestion\_site="trypsin",  
 Fastadatabase="calibrants.fasta",  
 output\_candidatelist=T,  
 plot\_matching\_score=T,  
 spectra\_segments\_per\_file=1,  
 use\_previous\_candidates=T,  
 peptide\_ID\_filter=1,ppm=5,missedCleavages=0:5,  
 Modifications=list(fixed=NULL,fixmod\_position=NULL,variable=c("Amide"),varmod\_position=c(6)),  
 FDR\_cutoff=100,  
 Substitute\_AA=list(AA=c("X"),AA\_new\_formula=c("C5H5NO2"),Formula\_with\_water=c(FALSE)),Thread = 1)

### Bovin lens

library(HiTMaP)  
datafile=c("Bovinlens\_Trypsin\_FT/Bovin\_lens.imzML")  
wd="~/expdata/"  
  
# Data pre-processing and proteomics annotation  
library(HiTMaP)  
imaging\_identification(datafile=paste0(wd,datafile),Digestion\_site="trypsin",  
 Fastadatabase="uniprot-bovin.fasta",output\_candidatelist=T,  
 preprocess=list(force\_preprocess=TRUE,  
 use\_preprocessRDS=TRUE,  
 smoothSignal=list(method="gaussian"),  
 reduceBaseline=list(method="locmin"),  
 peakPick=list(method="adaptive"),  
 peakAlign=list(tolerance=ppm, units="ppm"),  
 normalize=list(method=c("rms","tic","reference")[1],mz=1)),  
 spectra\_segments\_per\_file=9,use\_previous\_candidates=F,ppm=5,FDR\_cutoff = 0.5,IMS\_analysis=T,  
 Rotate\_IMG="file\_rotationbk.csv",  
 mzrange = c(500,4000),plot\_cluster\_image\_grid=F)  
  
# Re-analysis and cluster image rendering  
  
library(HiTMaP)  
datafile=c("Bovinlens\_Trypsin\_FT fdr01f/Bovin\_lens.imzML")  
wd="~/expdata/"  
imaging\_identification(datafile=paste0(wd,datafile),Digestion\_site="trypsin",  
 Fastadatabase="uniprot-bovin.fasta",  
 use\_previous\_candidates=T,ppm=5,IMS\_analysis=F,  
 plot\_cluster\_image\_grid=T,  
 export\_Header\_table=T,   
 img\_brightness=250,   
 plot\_cluster\_image\_overwrite=T,  
 cluster\_rds\_path = "/Bovin\_lens ID/preprocessed\_imdata.RDS",pixel\_size\_um = 150,  
 Plot\_score\_abs\_cutoff=-0.1,  
 remove\_score\_outlier=T,  
 Protein\_desc\_of\_interest=c("Crystallin","Phakinin","Filensin","Actin","Vimentin","Cortactin","Visinin","Arpin","Tropomyosin","Myosin Light Chain 3","Kinesin Family Member 14","Dynein Regulatory Complex","Ankyrin Repeat Domain 45"))

### Mouse brain

library(HiTMaP)  
datafile=c("MouseBrain\_Trypsin\_FT/Mouse\_brain.imzML")  
wd="~/expdata/"  
  
# Data pre-processing and proteomics annotation  
library(HiTMaP)  
imaging\_identification(datafile=paste0(wd,datafile),Digestion\_site="trypsin",  
 Fastadatabase="uniprot\_mouse\_20210107.fasta",output\_candidatelist=T,  
 preprocess=list(force\_preprocess=TRUE,  
 use\_preprocessRDS=TRUE,  
 smoothSignal=list(method="gaussian"),  
 reduceBaseline=list(method="locmin"),  
 peakPick=list(method="adaptive"),  
 peakAlign=list(tolerance=ppm, units="ppm"),  
 normalize=list(method=c("rms","tic","reference")[1],mz=1)),  
 spectra\_segments\_per\_file=9,use\_previous\_candidates=F,ppm=5,FDR\_cutoff = 0.5,IMS\_analysis=T,  
 Rotate\_IMG="file\_rotationbk.csv",  
 mzrange = c(500,4000),plot\_cluster\_image\_grid=F)  
  
# Re-analysis and cluster image rendering  
  
imaging\_identification(datafile=paste0(wd,datafile),Digestion\_site="trypsin",  
 Fastadatabase="uniprot\_mouse\_20210107.fasta",  
 preprocess=list(force\_preprocess=FALSE),  
 spectra\_segments\_per\_file=9,use\_previous\_candidates=T,ppm=5,FDR\_cutoff = 0.5,IMS\_analysis=F,  
 mzrange = c(500,4000),plot\_cluster\_image\_grid=T,  
 img\_brightness=250, plot\_cluster\_image\_overwrite=T,  
 cluster\_rds\_path = "/Mouse\_brain ID/preprocessed\_imdata.RDS",  
 pixel\_size\_um = 150,  
 Plot\_score\_abs\_cutoff=-0.1,  
 remove\_score\_outlier=T,  
 Protein\_desc\_of\_interest=c("Crystallin","Phakinin","Filensin","Actin","Vimetin","Cortactin","Visinin","Arpin","Tropomyosin","Myosin Light Chain 3","Kinesin Family Member 14","Dyenin Regulatory Complex","Ankyrin Repeat Domain 45"))

## Session information

toLatex(sessionInfo())

## \begin{itemize}\raggedright  
## \item R version 4.0.2 (2020-06-22), \verb|x86\_64-w64-mingw32|  
## \item Locale: \verb|LC\_COLLATE=English\_Australia.1252|, \verb|LC\_CTYPE=English\_Australia.1252|, \verb|LC\_MONETARY=English\_Australia.1252|, \verb|LC\_NUMERIC=C|, \verb|LC\_TIME=English\_Australia.1252|  
## \item Running under: \verb|Windows 10 x64 (build 19042)|  
## \item Matrix products: default  
## \item Base packages: base, datasets, graphics, grDevices, grid,  
## methods, stats, utils  
## \item Other packages: data.table~1.13.6, dplyr~1.0.2, gridExtra~2.3,  
## HiTMaP~1.6.0, lattice~0.20-41, magick~2.5.2, pls~2.7-3,  
## protViz~0.6.8, XML~3.99-0.5  
## \item Loaded via a namespace (and not attached): assertthat~0.2.1,  
## Biobase~2.48.0, BiocGenerics~0.34.0, BiocManager~1.30.10,  
## BiocParallel~1.22.0, cli~2.3.0, codetools~0.2-18, compiler~4.0.2,  
## crayon~1.4.0, digest~0.6.27, ellipsis~0.3.1, evaluate~0.14,  
## fansi~0.4.2, fastmap~1.1.0, generics~0.1.0, glue~1.4.2,  
## gtable~0.3.0, htmltools~0.5.1.1, httpuv~1.5.5, knitr~1.30,  
## later~1.1.0.1, lifecycle~0.2.0, magrittr~2.0.1, MASS~7.3-53,  
## Matrix~1.3-2, mime~0.9, multtest~2.44.0, pacman~0.5.1,  
## parallel~4.0.2, pillar~1.4.7, pkgconfig~2.0.3, png~0.1-7,  
## promises~1.1.1, purrr~0.3.4, R6~2.5.0, Rcpp~1.0.6, rlang~0.4.10,  
## rmarkdown~2.6, S4Vectors~0.26.1, shiny~1.6.0, splines~4.0.2,  
## stats4~4.0.2, stringi~1.5.3, stringr~1.4.0, survival~3.2-7,  
## tibble~3.0.6, tidyselect~1.1.0, tools~4.0.2, utf8~1.1.4,  
## vctrs~0.3.6, xfun~0.20, xtable~1.8-4, yaml~2.2.1  
## \end{itemize}

End of the tutorial, Enjoy~

## References

R Packages used in this project:

* viridisLite(Garnier 2018)
* rcdklibs(Guha 2017)
* rJava(Urbanek 2019)
* data.table(Dowle and Srinivasan 2019)
* RColorBrewer(Neuwirth 2014)
* magick(Ooms 2019)
* ggplot2(Wickham 2016)
* dplyr(Wickham et al. 2019)
* stringr(Wickham 2019)
* protViz(Panse and Grossmann 2019)
* cleaver(Gibb 2019)
* Biostrings(Pag�s et al. 2019)
* IRanges(Lawrence et al. 2013)
* Cardinal(Bemis et al. 2015)
* tcltk(R Core Team 2019)
* BiocParallel(Morgan et al. 2019)
* spdep(Bivand and Wong 2018)
* FTICRMS(Barkauskas 2012)
* UniProt.ws(Carlson 2019)

Barkauskas, Don. 2012. *FTICRMS: Programs for Analyzing Fourier Transform-Ion Cyclotron Resonance Mass Spectrometry Data*. <https://CRAN.R-project.org/package=FTICRMS>.

Bemis, Kyle D., April Harry, Livia S. Eberlin, Christina Ferreira, Stephanie M. van de Ven, Parag Mallick, Mark Stolowitz, and Olga Vitek. 2015. “Cardinal: An r Package for Statistical Analysis of Mass Spectrometry-Based Imaging Experiments.” *Bioinformatics*. <https://doi.org/10.1093/bioinformatics/btv146>.

Bivand, Roger, and David W. S. Wong. 2018. “Comparing Implementations of Global and Local Indicators of Spatial Association.” *TEST* 27 (3): 716–48. <https://doi.org/10.1007/s11749-018-0599-x>.

Carlson, Marc. 2019. *UniProt.ws: R Interface to UniProt Web Services*.

Dowle, Matt, and Arun Srinivasan. 2019. *Data.table: Extension of ‘Data.frame‘*. <https://CRAN.R-project.org/package=data.table>.

Garnier, Simon. 2018. *viridisLite: Default Color Maps from ’Matplotlib’ (Lite Version)*. <https://CRAN.R-project.org/package=viridisLite>.

Gibb, Sebastian. 2019. *Cleaver: Cleavage of Polypeptide Sequences*. <https://github.com/sgibb/cleaver/>.

Guha, Rajarshi. 2017. *Rcdklibs: The CDK Libraries Packaged for r*. <https://CRAN.R-project.org/package=rcdklibs>.

Lawrence, Michael, Wolfgang Huber, Hervé Pagès, Patrick Aboyoun, Marc Carlson, Robert Gentleman, Martin Morgan, and Vincent Carey. 2013. “Software for Computing and Annotating Genomic Ranges.” *PLoS Computational Biology* 9. <https://doi.org/10.1371/journal.pcbi.1003118>.

Morgan, Martin, Valerie Obenchain, Michel Lang, Ryan Thompson, and Nitesh Turaga. 2019. *BiocParallel: Bioconductor Facilities for Parallel Evaluation*. <https://github.com/Bioconductor/BiocParallel>.

Neuwirth, Erich. 2014. *RColorBrewer: ColorBrewer Palettes*. <https://CRAN.R-project.org/package=RColorBrewer>.

Ooms, Jeroen. 2019. *Magick: Advanced Graphics and Image-Processing in r*. <https://CRAN.R-project.org/package=magick>.

Pag�s, H., P. Aboyoun, R. Gentleman, and S. DebRoy. 2019. *Biostrings: Efficient Manipulation of Biological Strings*.

Panse, Christian, and Jonas Grossmann. 2019. *protViz: Visualizing and Analyzing Mass Spectrometry Related Data in Proteomics*.

R Core Team. 2019. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

Urbanek, Simon. 2019. *rJava: Low-Level r to Java Interface*. <https://CRAN.R-project.org/package=rJava>.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. <https://ggplot2.tidyverse.org>.

———. 2019. *Stringr: Simple, Consistent Wrappers for Common String Operations*. <https://CRAN.R-project.org/package=stringr>.

Wickham, Hadley, Romain Fran�ois, Lionel Henry, and Kirill Muller. 2019. *Dplyr: A Grammar of Data Manipulation*. <https://CRAN.R-project.org/package=dplyr>.