Tutorial

LC/MS preprocessing and analysis of the FAAH data

Background

Exercises

Start the program Rstudio (or R).

Question 1. LOADING AND VISUALIZING THE DATA

Note: the R commands required to answer the questions are given in the yellow boxes below.

a. Use xcms to determine how many samples are present in the current FAAH datasets. How many WT samples? How many KO samples?

b. How many mass spectra are in this data set? What was the duration of the measurement?

c. To have a better impression of the data you can use xcms (and, if you want, standard R functions) to visualize chromatograms and mass spectra (of the binned profile data in this example). Make of plot of the Total Ion Current (TIC). What does this plot show? Make a plot of the mass spectrum that was obtained in scan 601. What is the intensity of the highest peak in the spectrum? At what time (in seconds) was this scan measured? Plot a chromatogram based on m/z=496.2. How do you interpret this plot and how is this different from the TIC plot?

The R code to obtain the answer for question (a):

Note that the hash ('#') is used to denote comments in R. You can safely copy/paste comments into R: they will be neglected. The '>' sign denotes the 'R' prompt. Thus, don't type the first '>' in the commands below.

#set directory path to data files and list files

```
> cdfpath<-system.file("cdf",package="faahKO")
> list.files(cdfpath,recursive=TRUE)
```

The R code to obtain the answer for question (b):

#First initialize (=activate) all xcms functions required for the metabolomics data analysis

```
>library(xcms)
```

#determine all data files that need to be analysed and store them in a variable (= 'cdffiles')

```
➢  cdffiles <- list.files(cdfpath,recursive=TRUE, full.names=TRUE)
```

#Note that 'library' and 'list.files' are both R commands. To obtain more information about e.g., #list.files just type ?list.files at the R prompt.

#show what is contained in cdffiles
```
> cdffiles
> cdffiles[1]
```

#The next file reads in the first raw data file
#note: if profstep=0 then no profile matrix is generated
```
> rawdata <- xcmsRaw(cdffiles[1],profstep=0.1, profmethod="bin")
> rawdata
```

Note: if you want help about a xcms function (e.g., xcmsRaw) then type the following at the R prompt:
```
> ?xcmsRaw
Or
> help(xcmsRaw)
```

The R code to obtain the answer for question (c):
#note: the raw lc-ms data is internally stored in a variable that is called 'profile' which can be accessed through rawdata@env$profile. You can have a closer look at this data by using the commands below.

#show which data parts are stored in the object 'rawdata'. An object can be seen as a container that stores several other variables or objects.
```
> str(rawdata,max.level=2)
```

#one of the objects within the object rawdata is 'env'. 'env' itself is a compound variable that contains three sub-variables that contain the actual raw data
```
> ls(rawdata@env)
```

#to show part of the raw data (e.g., binned profile) use
```
> head(rawdata@env$profile)
```

#plot the total ion current and explain what you see.
```
> plotTIC(rawdata)
```

#visualize the mass trace that was measured in the 601th scan (time point) and explain what you see
```
> plotScan(rawdata,scan=601,mzrange=c(200,600))
```

#visualize the chromatogram based on m/z = 496.2 and explain what you see
```
> plotChrom(rawdata, fitgauss = FALSE, mzrange=c(496.2, 496.2))
```

Answer a:
6 WT, 6 KO

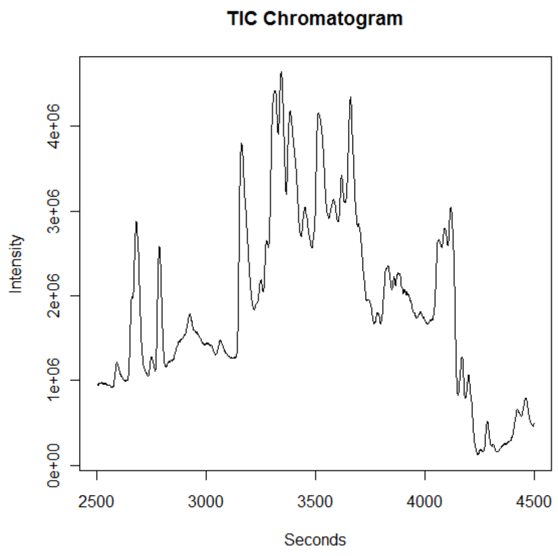
Answer b: 1278 mass spectra in maximal 75 minutes.

An "xcmsRaw" object with 1278 mass spectra

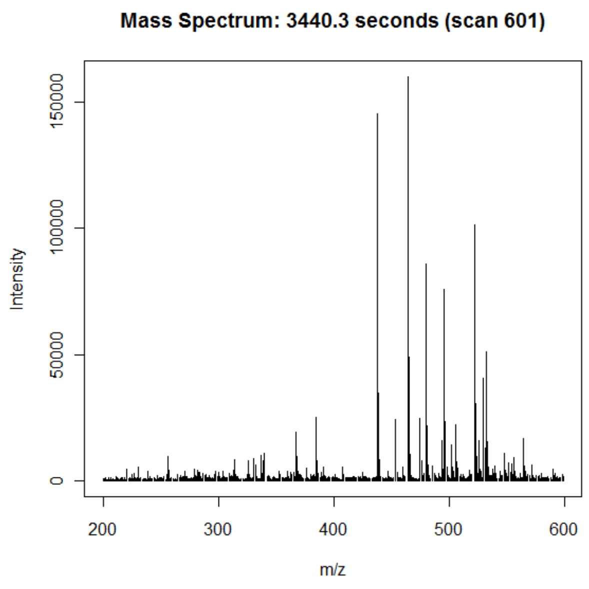
```
Time range: 2501.4-4499.8 seconds (41.7-75 minutes)
Mass range: 200-600 m/z
Intensity range: 70-1373180

MSn data on     0     mass(es)
               with     0     MSn spectra
Profile method: bin
Profile step: 0.1 m/z (4001 grid points from 200 to 600 m/z)
```

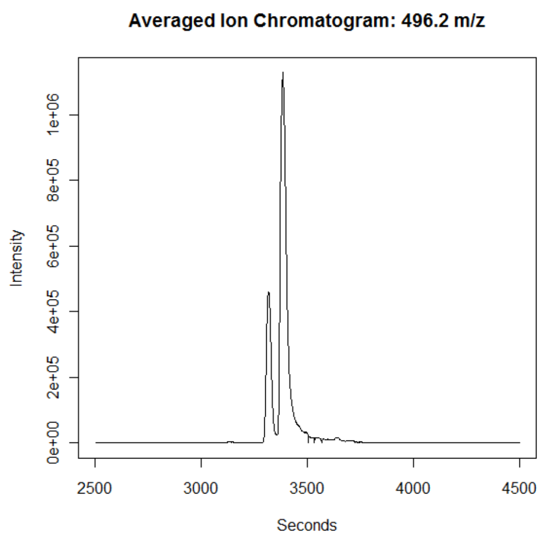
Answer c: to add



The TIC (total ion current chromatogram). This figure shows the chromatogram' (i.e., the signal in time). The y-axis shows the total 'ion current', which is a summation of all mass-peaks at every time point.



This figure shows one of the mass-spectra (m/z signals). This spectrum corresponds to measurement number 601, which is the measurement done at 3440.3 seconds.



This figure shows the averaged ion chromatogram but because a specific mass was chosen (and not range of masses), it shows how the signal of this specific mass (496.2) varies over time. In other words the compound with this mass elutes from the column after about 3300 seconds.

Peak identification

The class of objects used for pre-processing analyte data from multiple LC/MS files is xcmsSet. The class xcmsSet can be seen as a container that stores data and results, and provides the methods for dealing with them. It stores peak lists and provides methods for grouping and aligning those peaks. To create an xcmsSet object from a set of data files, use the `xcmsSet()` constructor function. It handles batch peak picking and generation of the xcmsSet object. There are a number of ways you can specify the files it should read. By default, it will recursively search through the current directory for FAAH data files. Alternatively, you can manually specify the files you are interested in, as shown below.

During peak identification, xcms uses a separate line for each sample to report on the status of processing. It outputs out pairs of numbers separated by a colon. **The first number is the m/z it is currently processing. The second number is the number of peaks that have been identified so far**. It is important to note that the number may be significantly larger than the final number of peaks as a 'vicinity elimination' post processing step removes duplicate peaks corresponding to the same ion.

The peak identification method that we will use is 'matched filtration'. This methods combines noise reduction and peak identification in one algorithm.

Question 2. PEAK IDENTIFICATION

a. Use matched filtration for peak detection. How many peaks are identified in each sample?

b: What are the mass and time ranges of the data set? How many peaks are identified in total? What do these peaks represent? Did the method detect the same number of peaks in every sample?

Note: you can neglect the Peak Groups, and profile settings for the time being

The R code to obtain the answer for question (a).
Note that xcmsSet is configured to use 'matched filtration' for peak detection. This function uses many parameters that influence the outcome of peak detection. Below are the default values that we will use in this exercise. Don't worry if you don't understand the meaning of all these parameters but you should be able to understand several of them. Try to give a brief description for each of the following parameters.

```
# sigma=fwhm/2.3548
> fwhm<- 30;
> sigma<- fwhm/2.3548

# maximum number of peaks per extracted ion-chromatogram
> max <- 5;

# signal to noise cutoff
> snthresh<- 10;

# step size to use for profile generation
> step <- 0.1

#number of steps to merge prior to matched filtration
> steps<- 2

# minimum difference in m/z for peaks with overlapping retention times
> mzdiff<- 0.8-steps*step
```

```
#"positive"    "negative"
> polarity <- NULL;
> profmethod   <- "bin"

#Now you can use all of these parameters to identify peaks in the LC-MS data:
> xset <- xcmsSet(cdffiles,
                  profmethod = profmethod,
                  polarity = polarity,
                  fwhm=fwhm,
                  sigma=sigma,
                  max=max,
                  snthresh=snthresh,
                  step=step,
                  steps=steps,
                  mzdiff=mzdiff,
                  sleep=0,
                  method="matchedFilter")
```

The code to obtain the answer for question (b)

```
> xset
```

Answer to question (a):

To determine the number of peaks per sample you have to write a few lines of R code:

```
#Number of peaks per sample
for(i in c(1:12)) {
   p=subset(peaks(xset), peaks(xset)[,"sample"] == i)
   cat("sample ",i,": ",dim(p)[1],"\n")
}
```

Answer to question (b):

```
An "xcmsSet" object with 12 samples

Time range: 2506.1-4147.7 seconds (41.8-69.1 minutes)
Mass range: 200.1-599.3338 m/z
Peaks: 4721 (about 393 per sample)
Peak Groups: 0
Sample classes: KO, WT

Profile settings: method = bin
                  step = 0.1
```

.

Matching peaks across samples

After peak identification, peaks representing the same analyte across samples must be placed into groups to allow comparison (e.g., statistical analysis) of the compounds). This is accomplished with the group method, which returns a new xcmsSet object with the additional group information. The grouping process is non-destructive and does not affect the other data stored in the xcmsSet object. Therefore, we can safely replace the xsetobject with the grouped version. The grouping algorithm processes the peak lists in order of increasing mass and will regularly output the mass it is currently working on.

Question 3. MATCHING PEAKS ACROSS SAMPLES

a. Use the group method to identify peaks representing the same analyte across samples. How many peak groups can be obtained for this data? What does this mean?

b. Have a closer look at the peak groups that are generated. How is the first peak group defined in terms of m/z range, rt-range, and number of samples in which this peak group occurs.

The R code to obtain the answer for question (a)

```
> xset<-group(xset)
> xset
```

Note: if you provide the following additional argument to the function group then you get graphical output showing the wild type and KO peaks (black and red dots) that were grouped together

```
> xset<-group(xset, sleep=1)
```

The R code to obtain the answer for question (b)

#have a look at the first 5 peaks that were identified (or use peaks(xset) to show them all)

```
> head(peaks(xset),5)
```

#have a look at the first 5 groups that were identified (or use groups(xset) to show them all)

```
> head(groups(xset),5)
```

Answer to question (a): 403

An "xcmsSet" object with 12 samples

Time range: 2506.1-4147.7 seconds (41.8-69.1 minutes)
Mass range: 200.1-599.3338 m/z
Peaks: 4721 (about 393 per sample)
Peak Groups: 403
Sample classes: KO, WT

Peak groups are groups of peaks in the Rt, m/z domain that occur in all or most of the samples. Thus, if a metabolite occurs in all samples then you would expect to see the chromatographic peak caused by these masses in every sample, and you will see the corresponding masses in every sample (thus a cluster in 2-dimensions Rt, m/z).

Answer to question (b);
> head(peaks(xset))

```
           mz  mzmin  mzmax        rt     rtmin     rtmax        into       intf
maxo
[1,] 200.1000 200.1  200.1  2928.610 2912.961 2942.695  147887.53   290506.9
9687
[2,] 201.0638 201.0  201.1  2531.112 2515.463 2549.892  204572.42   280386.0
7726
```

```
[3,] 205.0000 205.0 205.0 2784.635 2770.550 2800.284 1778568.94 3610059.7
84280
[4,] 205.9819 205.9 206.0 2786.200 2772.115 2800.284  237993.62  448580.0
10681
[5,] 207.0821 207.0 207.1 2712.647 2698.562 2726.731  380873.05  730980.9
18800
[6,] 208.0671 208.0 208.1 2640.659 2625.009 2656.308   96070.72  150033.4
4112
           maxf i        sn sample
[1,]  15899.054 1 13.40257      1
[2,]  13300.725 1 12.13748      1
[3,] 195026.431 1 70.14981      1
[4,]  23860.099 1 31.89939      1
[5,]  40065.736 1 23.89175      1
[6,]   7560.078 1 12.67485      1
```

mz weighted (by intensity) mean of peak m/z across scans

mzmin m/z of minimum step
mzmax m/z of maximum step
rt retention time of peak midpoint
rtmin leading edge of peak retention time
rtmax trailing edge of peak retention time
into integrated area of original (raw) peak
intf integrated area of filtered peak
maxo maximum intensity of original (raw) peak
maxf maximum intensity of filtered peak
i rank of peak identified in merged EIC (<= max)
sn signal to noise ratio of the peak

```
> head(groups(xset))
        mzmed    mzmin    mzmax    rtmed    rtmin    rtmax npeaks KO WT
[1,] 200.1000 200.1000 200.1000 2925.480 2876.967 2931.740      9  4  5
[2,] 205.0000 205.0000 205.0000 2790.894 2784.635 2795.591     12  6  6
[3,] 205.9927 205.9786 206.0023 2790.112 2784.635 2795.591     12  6  6
[4,] 207.0850 207.0440 207.1000 2718.906 2712.647 2726.731     12  6  6
[5,] 219.0848 219.0488 219.1000 2524.852 2518.592 2529.547      9  4  5
[6,] 231.0236 231.0000 231.0812 2517.029 2509.202 2535.807      6  3  3
```

The first peak group occurs in 4 KO samples and 5 WT samples. It only contains one mass peak (200.1) within the retention time range of 2876 – 2931 seconds.

Retention time correction

After matching peaks into groups, xcms can use those groups to identify and correct drifts in retention time from run to run. The aligned peaks can then be used for a second pass of peak grouping which will be more accurate than the first. The whole process can be repeated in an iterative fashion, although we will only demonstrate a single pass of retention time alignment here.

Not all peak groups will be helpful for identifying retention time drifts. Some groups may be missing peaks from a large fraction of samples and thus provide an incomplete picture of the drift at that time point. Still others may contain multiple peaks from the same sample, which is a sign of improper grouping. xcms ignores those groups by only considering **"well-behaved" peak groups** which are missing at most one sample and have at most one extra peak. (Those values can be changed with the missing and extra arguments). For each of those well-behaved groups, the algorithm calculates a median retention time and, for every sample, a deviation from that median.

Within a sample, the observed deviation generally changes over time in a nonlinear fashion. Those changes are approximated using a local polynomial regression technique. By default, the curve fitting is done using least-squares on all data points. However, it is possible to enable outlier detection and removal by setting the family argument to "symmetric", as shown here.

Retention time correction is performed by the retcor method, which returns an xcmsSet object with corrected retention times. Because it changes the retention times of all peaks, it is important to store the new object under a new variable name. That will allow you to backtrack and repeat retention time correction if necessary.

After retention time correction, the initial peak grouping becomes invalid and is discarded. Therefore, the resulting object needs to be regrouped. Here, we decrease the inclusiveness of the grouping using the bw argument (default 30 seconds).

Question 4. RETENTION TIME CORRECTION

a. Perform a retention time correction and new peak groups. How many retention time correction groups were used? What does this mean?
b. The retention time correction produces a graph. Explain this figure.
c. Perform a second iteration of grouping. How many peaks groups are identified (after retention time correction)? Have a look at the time alignment figure. What do you notice? Why do we have fewer groups compared to the grouping prior to retention time correction?

Note: in the remaining exercises we will continue with xset2 (only one round of time alignment) and not with xset3 (obtained after two rounds of alignment).

The code to obtain this answer:

```
> xset2 <- retcor(xset, missing=1, extra=1, smooth="loess", family=
"symmetric", plottype = "mdevden")
```

#If you inspect the content of xset2 then you will see the raw and corrected retention times

```
> str(xset2)
```

#note: next a regroup is necessary. Note that the inclusiveness is decreased by using the bw argument.

```
> xset2 <- group(xset2,bw=10)

#to do a next iteration of alignment:
> xset3 <- retcor(xset2, missing=1, extra=1, smooth="loess", family=
"symmetric", plottype = "mdevden")
> xset3 <- group(xset3,bw=10) # and re-group again.
```

Answer to question (a): 133

These are the 'well behaved' groups that could be used for the correction. A well-behaved group is defined by the parameters 'missing' and 'extra'. In this example, a well behaved group is a group of peaks that is missing in no more than 1 of the samples. Moreover, the 'extra' parameter indicates that it is allowed to use one extra peak in the retention time correction and which is present in the EIC. For example for peak group 7:

head(groups(xset),7)

```
        mzmed    mzmin    mzmax    rtmed    rtmin    rtmax npeaks KO WT
[1,] 200.1000 200.1000 200.1000 2925.480 2876.967 2931.740      9  4  5
[2,] 205.0000 205.0000 205.0000 2790.894 2784.635 2795.591     12  6  6
[3,] 205.9927 205.9786 206.0023 2790.112 2784.635 2795.591     12  6  6
[4,] 207.0850 207.0440 207.1000 2718.906 2712.647 2726.731     12  6  6
[5,] 219.0848 219.0488 219.1000 2524.852 2518.592 2529.547      9  4  5
[6,] 231.0236 231.0000 231.0812 2517.029 2509.202 2535.807      6  3  3
[7,] 233.0371 233.0128 233.0652 3022.507 3016.247 3077.281     13  6  6
```

We see that 13 peaks have been identified in 6 KO and 6 WT samples. To identify these peaks we use

p<-as.vector(unlist(xset@groupidx[7]))
peaks(xset)[p,]

```
            mz mzmin mzmax       rt    rtmin    rtmax     into     intf  maxo     maxf i       sn
 [1,] 233.0390   233 233.1 3019.378 3005.293 3033.462 399145.3 749332.4 19752 41554.24 1 27.47936
 [2,] 233.0335   233 233.1 3024.072 3009.988 3038.157 356951.3 648640.1 16157 35014.64 1 24.31308
 [3,] 233.0156   233 233.1 3077.281 3066.326 3089.800 178932.9 231151.1  9235 15004.80 2 10.41887
 [4,] 233.0383   233 233.1 3020.944 3006.859 3035.028 410550.7 803872.5 21352 44454.67 1 32.58706
 [5,] 233.0155   233 233.1 3016.247 3000.598 3030.332 198416.5 313822.4  8706 16799.48 1 16.50511
 [6,] 233.0642   233 233.1 3028.767 3014.683 3042.852 363381.7 624562.4 16320 32749.65 1 25.37145
 [7,] 233.0371   233 233.1 3020.943 3006.858 3035.027 317805.8 591878.9 15184 32006.61 1 27.35407
 [8,] 233.0455   233 233.1 3019.378 3005.293 3033.462 397107.8 716883.3 18256 39091.41 1 23.43451
 [9,] 233.0128   233 233.1 3025.637 3011.552 3039.722 271252.1 466054.7 12202 25639.14 1 19.66060
[10,] 233.0371   233 233.1 3025.638 3011.553 3039.722 334459.9 619613.8 15924 33644.34 1 27.49708
[11,] 233.0512   233 233.1 3017.813 3003.728 3031.897 181901.3 321217.3  8871 17608.98 1 17.84666
[12,] 233.0652   233 233.1 3028.769 3013.120 3042.854 456900.5 813150.5 19992 42694.23 1 30.30085
[13,] 233.0149   233 233.1 3022.507 3008.423 3036.592 294270.6 543480.0 14369 29351.68 1 25.96288
      sample
 [1,]      1
 [2,]      2
 [3,]      2
 [4,]      3
 [5,]      4
 [6,]      5
 [7,]      6
 [8,]      7
 [9,]      8
[10,]      9
[11,]     10
[12,]     11
[13,]     12
```

From this output we see that in sample 2, two peaks have been found in the same EIC. Ideally, this should not happen. However, both peaks can be used in the retention time correction or you can use the first peak only. Note the lower s/n ratio for the second peak.

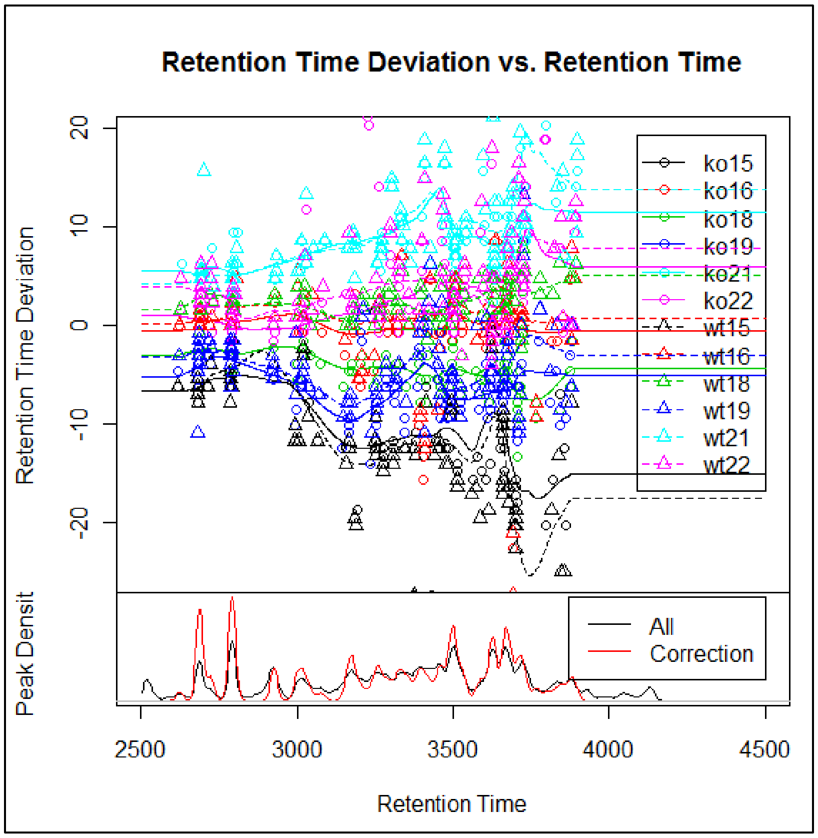


Figure: Retention time deviation profiles used for aligning the samples. The data points used for generating each profile are also shown. For each sample 132 peak groups are plotted. All times are in seconds. A negative number indicates a sample was eluting before most of the others, and vice versa. Samples that were acquired on the same day are colored similarly and have correlated deviation profiles, as expected (thus, on every day a WT and KO were done). Below, kernel density estimation is used to show the distribution of all peaks and those peaks used as standards for retention time correction.

```
# to get some idea of how the Rt vs Rt deviation plot was generation consider the following
# commands

#show peak groups
groups(xset)

#select well-behaved peaks groups according to 'missing' and 'extra' criteria
groups(xset)[groups(xset)[,"npeaks"]<14 &(groups(xset)[,"KO"]+groups(xset)[,"WT"])>10,]

#or just get the indices of these well-behaved peak groups
idx<-which(groups(xset)[,"npeaks"]<14 & (groups(xset)[,"KO"]+groups(xset)[,"WT"])>10)

#get individual rt values for each sample for these well-behaved peak groups
rt<-groupval(xset, "maxint", "rt")[idx,]
#calculate deviation from median for each sample
rtdev <- rt - apply(rt, 1, median, na.rm = TRUE)
```

```
#now in retcor, the lowess is performed:
#for every peak group the rt and rt deviation
#pts <- na.omit(data.frame(rt = rt[,i], rtdev = rtdev[,i]))
#pts is input to lowess:
#lo <- suppressWarnings(loess(rtdev ~ rt, pts, span = span, degree = 1, family = family))

#plot first sample
pts <- na.omit(data.frame(rt = rt[,1], rtdev = rtdev[,1]))
plot(x=pts[,"rt"],y=pts[,"rtdev"],ylim=c(-25,20),xlim=c(2500,4500),pch='x')

#plot other samples
for (i in 2:12) {
  points(data.frame(rt = rt[,i], rtdev = rtdev[,i]), col = i, pch = 'x', type="p")
}
```

Answer to question (c): 400

```
An "xcmsSet" object with 12 samples
Time range: 2505-4147.1 seconds (41.7-69.1 minutes)
Mass range: 200.1-599.3338 m/z
Peaks: 4721 (about 393 per sample)
Peak Groups: 400
Sample classes: KO, WT
```

Thus, we have 400 peak groups instead of 403. These 3 peak groups have probably disappeared because they have been 'merged' by the retention time correction to other groups.

Filling in missing peak data

After the second pass of peak grouping, there will still be peak groups which are missing peaks from some of the samples. That can occur because peaks were missed during peak identification or because an analyte was not present in a sample. In any case, those missing data points can be filled in by rereading the raw data files and integrating them in the regions of the missing peaks. That is performed using the fillPeaks method, which returns a xcmsSet object with the filled in peak data. While running, it outputs the name of the sample it is currently processing.

Question 5. Filling in missing peak data

a. Fill in the missing peaks.
b. How many peaks were missing?
c. Find the missing peaks in the peak table. What do you notice?

The code to obtain this answer:

```
> xset3 <- fillPeaks(xset2)
> xset3
```

You can inspect the first few lines of the peak table:

```
> head(peaks(xset3))
```

You can also specifically identify the peaks that were added to the peak table

```
> head(peaks(xset3)[which(is.na(peaks(xset3)[,'sn'])),])
```

Answer to question (b): 1319

An "xcmsSet" object with 12 samples

Time range: 2501.2-4148 seconds (41.7-69.1 minutes)
Mass range: 200.1-599.3338 m/z
Peaks: 6040 (about 503 per sample)
Peak Groups: 400
Sample classes: KO, WT

We now have 6040 peaks compared to 4721 peaks prior to peak filling, thus 1319 additional peaks have been added to the 12 samples (on average 110 per sample).

Answer to question (c):

> head(peaks(xset3)[which(is.na(peaks(xset3)[,'sn'])),])

```
           mz  mzmin  mzmax       rt    rtmin    rtmax       into intf maxo
maxf
[1,] 231.0544 231.0 231.1 2522.179 2503.896 2530.500      0.000   NA    0
NA
[2,] 254.1000 254.1 254.2 3230.552 3216.743 3248.509  83320.920   NA 3987
NA
[3,] 264.1272 264.1 264.2 3162.771 3140.345 3170.147  25037.660   NA 1906
NA
[4,] 268.2000 268.2 268.2 3876.785 3864.512 3900.137   5455.199   NA  872
NA
[5,] 273.2000 273.2 273.2 3677.008 3657.227 3689.323  79091.653   NA 3586
NA
[6,] 291.2000 291.2 291.2 3666.507 3653.600 3687.651 112716.676   NA 5514
NA
      i sn sample
[1,] NA NA      1
[2,] NA NA      1
[3,] NA NA      1
[4,] NA NA      1
[5,] NA NA      1
[6,] NA NA      1
```

For the 'filled peaks' you notice that no information about signal-to-noise ratio (sn) is available. No information about the filtered data (maxf, intf) because no matched filtration was applied to these peaks.

Analysing and visualizing results

We have now identified all peaks for all samples, aligned the chromatograms to allow comparison of the samples, and have added missing peaks. Now we are ready to compare the WT and KO mice to find the most statistically significant differences in analyte intensities. This is done with the diffreport method.

It will automatically generate extracted ion chromatograms for a given number of them (in this case 10).

Question 6. Identify metabolite differences between WT and KO mice

a. Determine which metabolites have different concentrations between WT and KO mice: open the file report.tsv in Excel. Give a description of the information that is in this file.

b. Have a look at the EICs and boxplots that are produced. What is an EIC? What is a boxplot? What do you notice from the EIC's? What do the boxplots represent (hint: do help(boxplot) in R.

c. Can you confirm that the peak of the first EIC corresponds to a N-acyl ethanolamine. What do you conclude from this?

The R code to obtain the answer to this question:

#Note: **replace the directory for fb with your own directory** where you want to store the results (e.g., C:\\Users\\AvK\\Desktop).

```
>fb="C:\\Users\\AvK\\Desktop"

>setwd(fb) #set working directory

>reporttab<-diffreport(xset3, filebase=paste(fb, "\\report",sep=""),
"WT", "KO", 10, metlin=0.15)

> write.table(reporttab, file="plist.txt", na="", quote=FALSE,
row.names=FALSE, dec=".",sep = "\t")
```

The 'diffreport' function produces a tab-delimited output file (report.tsv) that can be opened in Excel. In this example, it also produces also 10 EIC's and 10 boxplots of the most significant metabolites. These are automatically placed in two subdirectories: report_box and report_eic.

The Excel file contains the links to the Metlin database for all identified peaks. If the Metlin argument (metlin=0.15 in this example) is set to a numeric value, the report will include links to the Metlin Metabolite Database (http://metlin.scripps.edu/) showing potential metabolite identities. A positive value indicates the data was acquired in positive ion mode and the neutral mass is calculated assuming all ions are M+H. A negative value does the opposite. The value itself indicates the uncertainty in mass accuracy.

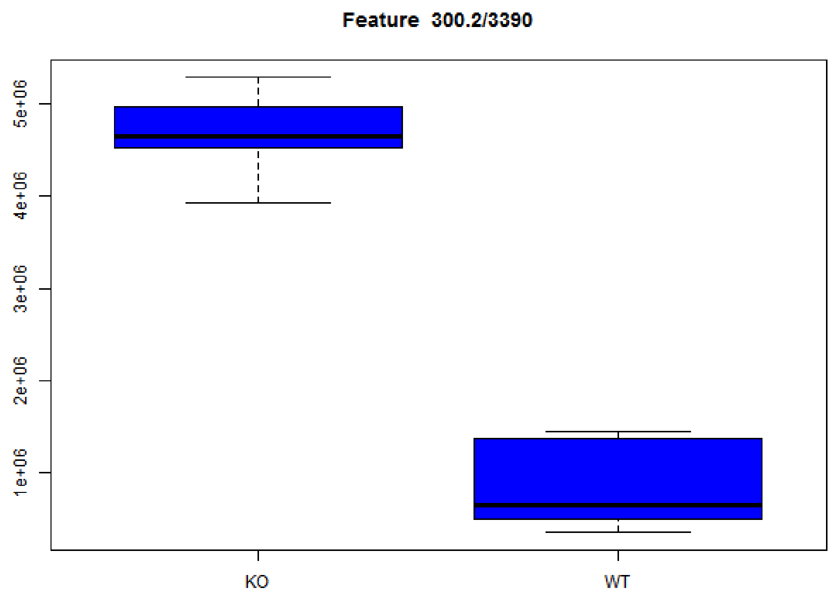


Figure. One of the 10 box plots generated by diffreport. It shows the difference in intensity between KO and WT mice for the compound measured at m/z=300.2 and t=3390 seconds. From the excel sheet you can see that the associated p-value is $5.02*10^{-8}$. Use help(boxplot) in R to find out what these boxes represent.

The corresponding EIC is shown in the figure below.

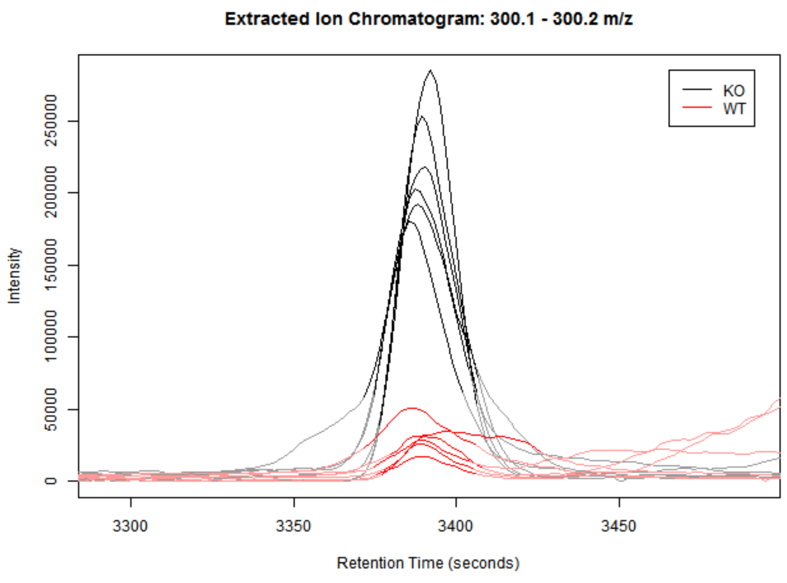


Figure. The extracted ion chromatogram for the mass-slice 300.1-300.2. You see the intensity values for 6 WT and 6 KO samples. The intensities between these two groups are clearly different. Notice that the alignment of the samples was not perfect. Also notice that one of the WT samples does not have a very well defined peak. The reason for this is unclear at this stage but could be investigated.

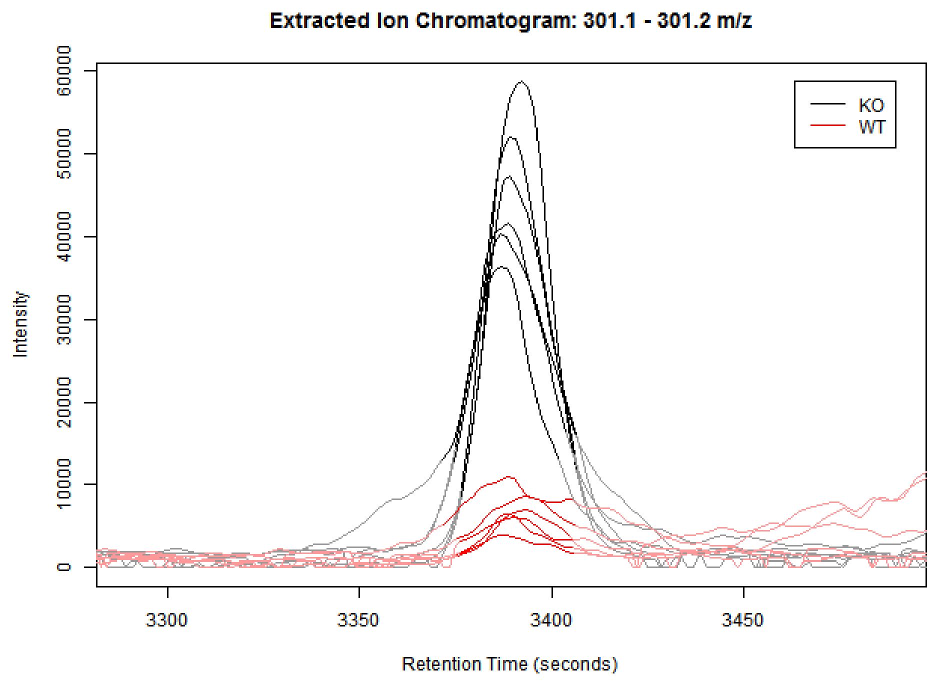
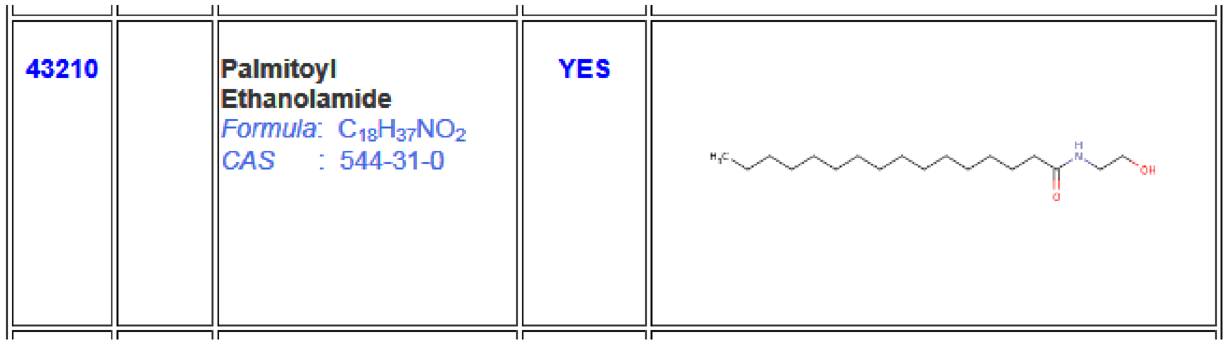


Figure. The extracted ion chromatogram for the mass-slice 301.1-301.2. This is the compound with the second largest significant difference. Notice that the difference in mass with the previous figure is exactly 1. The peaks shown in this and the previous figure represent different isotopes from the same compound. While all isotopes of a given element share the same number of protons, each isotope differs from the others in its number of neutrons. The isotope peak in this figure is cause by an additional neutron in one of the carbon atoms of this compound.

In the excel file we find the following links to the metlin database for the first four compounds

- http://metlin.scripps.edu/metabo_list.php?mass_min=299.04&mass_max=299.34
- http://metlin.scripps.edu/metabo_list.php?mass_min=300.04&mass_max=300.34
- http://metlin.scripps.edu/metabo_list.php?mass_min=297&mass_max=297.3
- http://metlin.scripps.edu/metabo_list.php?mass_min=490.05&mass_max=490.35

The first two EICs thus show the primary and secondary isotopic peaks of an N-acyl ethanolamine (NAE) with a 16-carbon acyl chain. This is, indeed, confirmed by the first link of Metlin.
Thus this compound is no longer degraded in the KO mouse.



43210		**Palmitoyl Ethanolamide** *Formula*: $C_{18}H_{37}NO_2$ *CAS* : 544-31-0	**YES**