LC/MS Preprocessing and Analysis with xcms

Colin A. Smith

April 14, 2009

Introduction

This document describes how to use *xcms* to preprocess LC/MS data for relative quantitation and statistical analysis. It gives examples of how visualization can be used throughout the process and to display final results. An overview of the preprocessing/analysis methodology, along with the function names associated with each step, is shown in Figure 1.

1 Raw Data File Preparation

The *xcms* package reads full-scan LC/MS data from AIA/ANDI format NetCDF, mzXML, and mzData files. All data to be analyzed by *xcms* must first be converted to one of those file formats. Software packages for many instruments are be able to export to NetCDF. For information about how to export to NetCDF, please consult the documentation that came with your instrument's software. The online help of most packages frequently use the terms "CDF" or "AIA" when referring to NetCDF format. In addition to NetCDF, mzXML exporters for a number of instruments are also available.[1]

After exporting all files to NetCDF/mzXML/mzData format, they should be put in a location that will remain the same throughout the analysis. That is because *xcms* records the location of the raw data files and refers back to them a number of times during preprocessing and analysis.

In most cases, LC/MS files that were acquired under different conditions should not be compared. For instance, positive and negative ionization mode files will have no ions in common and should thus be preprocessed separately. Similarly, data files acquired with different elution gradients should not be processed together.

Another important consideration is the directory structure in which the files are organized. *xcms* uses sample class information during preprocessing to help decide which groups of peaks are significant. If organized into subdirectories, samples will automatically be assigned to separate classes based on their location. Samples may be separated

[1] `http://sashimi.sourceforge.net/software_glossolalia.html`

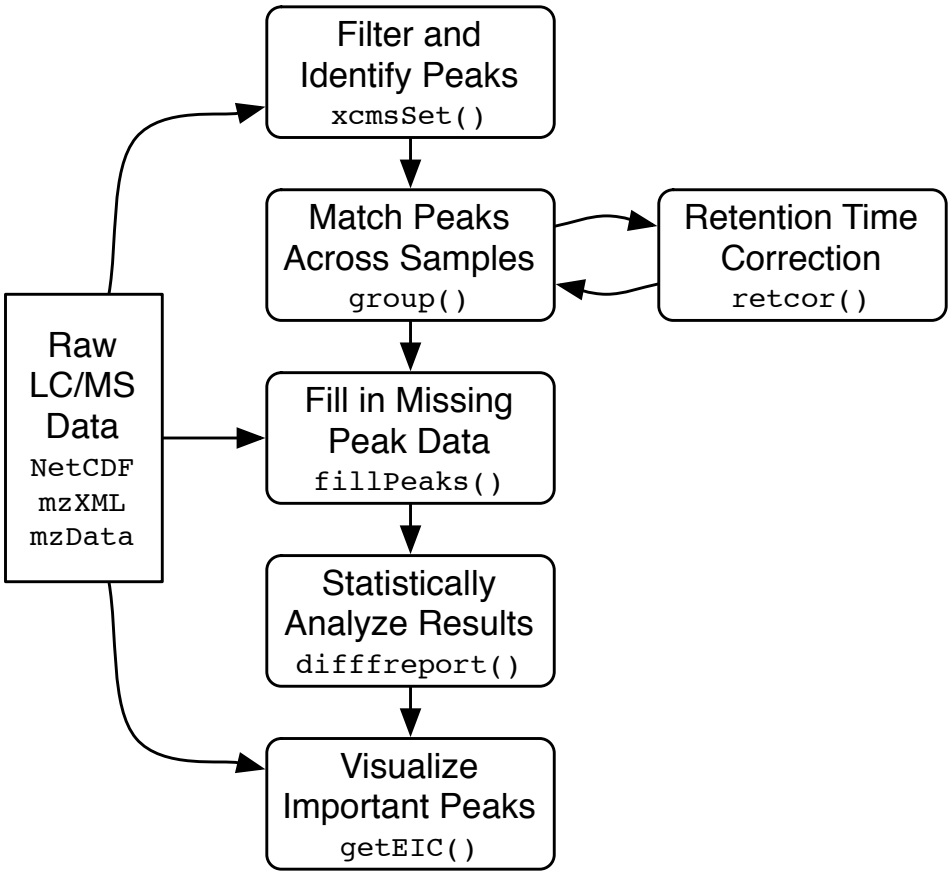

Figure 1: Flow chart showing a high-level overview of the preprocessing/analysis methodology employed by *xcms*. Function/method names corresponding to each step are also given.

into class based on tissue type, mutation, gender, disease, or time. For example, if you are analyzing the longitudinal effect of a drug in two patient groups, you may wish to put the groups into two directories "GroupA" and "GroupB". Within each of those directories, you could further separate the samples by the time they were taken, such as "Day1", "Day2", etc. In *xcms*, they will be automatically assigned class names "GroupA/Day1", "GroupA/Day2", etc.

For the purposes of demonstration, we will use a subset of the data from Saghatelian et al. (2004) examining the metabolic consequences of knocking out the fatty acid amide hydrolase (FAAH) gene in mice. The raw data files are contained in the `cdf` directory of the *faahKO* data package. There are samples from the spinal cords of 6 knockout mice and 6 wild-type mice placed in two subdirectories. Each file contains centroided data acquired in positive ion mode from 200-600 m/z and 2500-4500 seconds. To access the NetCDF files, we first locate the *cdf* directory in the *faahKO* package.

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

```
 [1] "KO/ko15.CDF" "KO/ko16.CDF" "KO/ko18.CDF" "KO/ko19.CDF" "KO/ko21.CDF"
 [6] "KO/ko22.CDF" "WT/wt15.CDF" "WT/wt16.CDF" "WT/wt18.CDF" "WT/wt19.CDF"
[11] "WT/wt21.CDF" "WT/wt22.CDF"
```

2 Filtration and Peak Identification

The class of objects used for preprocessing analyte data from multiple LC/MS files is *xcmsSet*. It stores peak lists and provides methods for grouping and aligning those peaks. To create an *xcmsSet* object from a set of NetCDF 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 NetCDF/mzXML/mzData files. Alternatively, you can manually specify the files you are interested in, as shown below.

During peak identification, *xcms* uses a sepearate 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 imporant to note that the number may be significantly larger than the final number of peaks as a vicinity elimination posprocessing step removes duplicate peaks corresponding to the same ion.

```
> library(xcms)
> cdffiles <- list.files(cdfpath, recursive = TRUE, full.names = TRUE)
> xset <- xcmsSet(cdffiles)
```

```
ko15: 250:38 300:103 350:226 400:338 450:431 500:529 550:674 600:847
ko16: 250:43 300:128 350:275 400:394 450:500 500:637 550:835 600:1027
```

```
ko18: 250:25 300:93 350:227 400:337 450:411 500:498 550:640 600:758
ko19: 250:19 300:67 350:169 400:258 450:301 500:373 550:488 600:580
ko21: 250:24 300:60 350:166 400:254 450:315 500:391 550:501 600:582
ko22: 250:31 300:71 350:183 400:280 450:338 500:422 550:532 600:604
wt15: 250:41 300:105 350:212 400:319 450:416 500:533 550:684 600:838
wt16: 250:27 300:107 350:232 400:347 450:440 500:549 550:712 600:905
wt18: 250:24 300:87 350:200 400:293 450:351 500:426 550:548 600:661
wt19: 250:22 300:65 350:161 400:243 450:293 500:358 550:483 600:561
wt21: 250:28 300:69 350:157 400:229 450:282 500:364 550:493 600:592
wt22: 250:30 300:81 350:188 400:280 450:356 500:473 550:618 600:765

> xset

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

Memory usage: 0.708 MB
```

The default arguments for `xcmsSet` should work acceptably in most cases. However, there are a number of parameters that may need to be optimized for a particular instrument or group of samples. The full set of parameters can be seen by viewing the documentation for the `xcmsSet` function and *findPeaks* method.

The method *findPeaks* can make use of different algorithms for peak detection. Currently *findPeaks.matchedFilter* and *findPeaks.centWave* are available, *findPeaks.matchedFilter* is used by default. First, several of the most important parameters of *findPeaks.matchedFilter* will be discussed.

findPeaks.matchedFilter

One parameter to consider is the Gaussian model peak width used for matched filtration, an integral part of the peak detection algorithm. For a discussion of how model peak width affects the signal to noise ratio, see Danielsson et al. (2002). It can be specified as either the standard deviation (*sigma*) or full width at half maximum (*fwhm*). By default, a FWHM of 30 seconds is used. Depending on the type of chromatography, the correct model peak width can be quite different. One means of determining the peak width is to fit the Gaussian function to one or more peaks in representative samples

produced with your experimental protocol. Functionality for doing so is provided in the *plotChrom* method with the *fitgauss* argument set to TRUE.

Several parameters depend on the resolution your mass spectrometer. Prior to matched filtration, the peak detection algorithm creates extracted ion base peak chromatograms (EIBPC) on a fixed step size defined by the *step* argument (default 0.1 m/z). To take into account uncertainties in scan to scan mass accuracy, the peak identification algorithm combines a given number of EIBPCs prior to filtration and peak detection, as defined by the *steps* argument. The default value, 2, combines EIBPCs 1-2, 2-3, 3-4, etc. If the peak width is significantly greater than the step size, you may wish to turn off combination using a value of 1. If your scan to scan accuracy is worse, you may wish to increase the number of scans combined. For example, a value of 3 would combine EIBPCs 1-3, 2-4, 3-5, etc.

Another factor to consider is the algorithm by which EIBPCs are produced. One way of thinking about that process is as a transformation of the data from being separate lists of mass/intensity pairs (one list for each scan) to a matrix with rows representing equally spaced masses and a column for each scan. Data transformed into such a matrix is usually referred to as being in profile mode. To do so, each scan of unequally spaced masses must be mapped onto a column of the final matrix. The algorithm used to do so is selected using the *profmethod* argument and can be either "bin", "binlin", "binlinbase", or "intlin".

The simplest algorithm, "bin", simply bins the intensity into the matrix cell closest to it in mass. If more than one intensity value is assigned to the same cell, then the greatest intensity is used. All other matrix cells are left at zero. It is the default and is especially useful for centroided data. "binlin" does the same thing except that it uses linear interpolation to fill in cells that otherwise would have been left at zero. It works well for sparsely populated continuum data.

Some mass spectrometry software allows the user to set an intensity threshold below which no mass/intensity values are recorded in continuum mode. When the mass spectral signal falls below that threshold, simple linear interpolation will not provide a good approximation of the original signal, instead creating artificially high background. To address that, the "binlinbase" method uses linear interpolation between data points within 0.15 m/z of each other, and otherwise inserts a basal intensity value set to half of the minimum intensity. Those specific parameters can be changed using the *profparam* argument. See documentation for the function `profBinLinBase` for more details.

The last method, "intlin", uses integration and linear interpolation between mass/intensity pairs to determine the equally spaced intensity values. This has the advantage of being numerically stable regardless of the mass step size. However, it is more useful for visualization than peak identification and is generally not recommended as such.

findPeaks.centWave

The method *findPeaks.centWave* follows a different approach. This algorithm is most suitable for high resolution LC/{TOF,OrbiTrap,FTICR}-MS data in centroid mode. Due to the fact that peak centroids are used, a binning step is not necessary.

In the first phase of the method mass traces (characterised as regions with less than *ppm* m/z deviation in consecutive scans) in the LC/MS map are located. In the second phase these mass traces are further analysed. Continuous wavelet transform (CWT) is used to locate chromatographic peaks on different scales. Accordingly, two parameters have to be adjusted. The *ppm* parameter has to be set according to the machine accuracy, e.g. *ppm=25*. The peak width range (*peakwidth=c(min,max)*) has to be set according to the chromatographic peak width range, e.g. *peakwidth=c(20,50)* seconds for HPLC and *peakwidth=c(5,12)* seconds for UPLC chromatography.

The method is capable of detecting close-by-peaks and also overlapping peaks. Some efforts are made to detect the exact peak boundaries to get precise peak integrals. The peak attributes `sn` (Signal/Noise Ratio) and `egauss` (root-mean-square-error of the gaussian fit) can be used to assess the peak quality.

3 Matching Peaks Across Samples

After peak identification, peaks representing the same analyte across samples must be placed into groups. That 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 `xset` object 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.

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

```
262 325 387 450 512 575
```

There are several grouping parameters to consider optimizing for your chromatography and mass spectrometer. Please consult the *group* documentation for more details. To see what the algorithm is doing while running, use the *sleep* argument to specify a time (in seconds) to pause and plot each iteration. That can be quite useful for visualizing parameter effects.

4 Retention Time Correction

After matching peaks into groups, *xcms* can use those groups to identify and correct correlated 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 implemented in the `loess` function. 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.

```
> xset2 <- retcor(xset, family = "symmetric", plottype = "mdevden")

Retention Time Correction Groups: 133
```

The above command uses the *plottype* argument to produce a plot, shown in Figure 2, which is useful for supervising the algorithm. It includes the data points used for loess regression and the resulting deviation profiles. It additionally shows the distribution of peak groups across retention time.

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).

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

262 325 387 450 512 575
```

5 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.

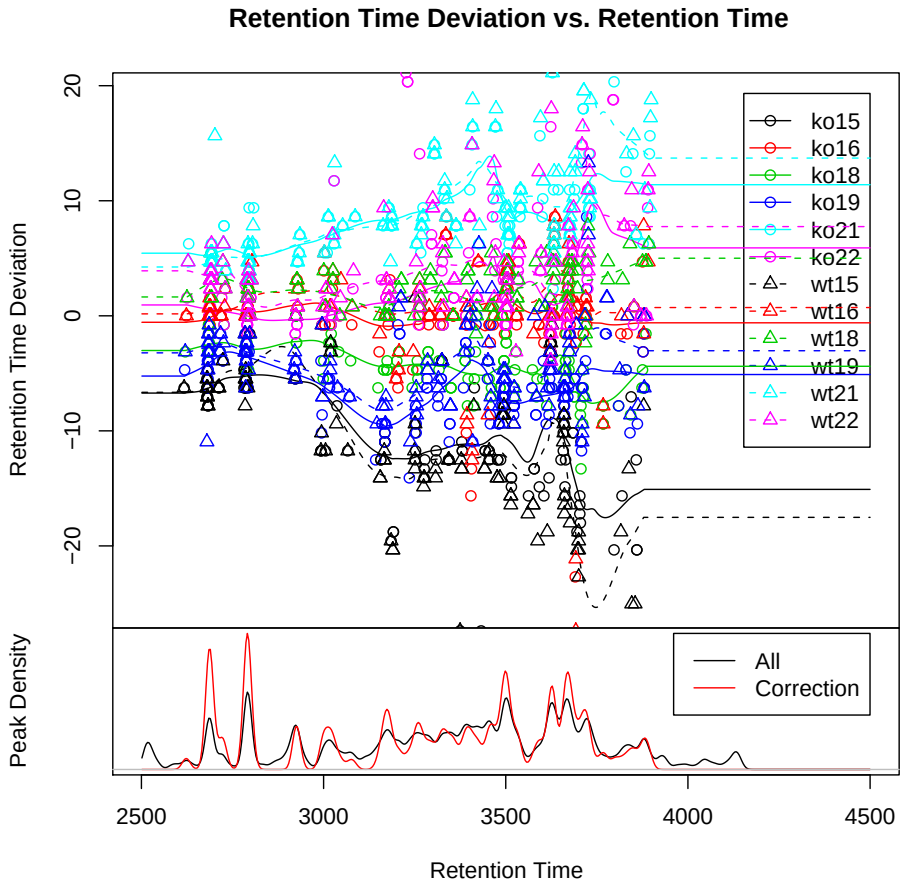

Figure 2: Retention time deviation profiles used for aligning the samples. The data points used for generating each profile are also shown. All times are in seconds. A negative number indicates a sample was eluting before most of the others, and vice versa. Samples that were acquired on the same day are colored similarly and have correlated deviation profiles, as expected. Below, kernel density estimation is used to show the distribution of all peaks and those peaks used as standards for retention time correction. Examples of two peaks before and after alignment are shown in Figure 4.

```
> xset3 <- fillPeaks(xset2)

ko15 ko16 ko18 ko19 ko21 ko22 wt15 wt16 wt18 wt19 wt21 wt22

> xset3

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

Profile settings: method = bin
                  step = 0.1

Memory usage: 0.896 MB
```

6 Analyzing and Visualizing Results

A report showing the most statistically significant differences in analyte intensities can be generated with the *diffreport* method. It will automatically generate extracted ion chromatograms for a given number of them, in this case 10. Several of those chromatograms are shown in Figure 3.

```
> reporttab <- diffreport(xset3, "WT", "KO", "example", 10, metlin = 0.15,
+     h = 480, w = 640)

ko15 ko16 ko18 ko19 ko21 ko22 wt15 wt16 wt18 wt19 wt21 wt22

> reporttab[1:4, ]

       name      fold    tstat       pvalue    mzmed    mzmin    mzmax    rtmed
1 M300T3390  5.693594 14.44368 5.026336e-08 300.1898 300.1706 300.2000 3390.324
2 M301T3390  5.876588 15.57570 6.705719e-08 301.1879 301.1659 301.1949 3389.627
3 M298T3187  3.870918 11.93891 3.310025e-07 298.1508 298.1054 298.1592 3186.803
4 M491T3397 12.499877 15.45496 1.922359e-06 491.2000 491.1877 491.2063 3397.160
     rtmin    rtmax npeaks KO WT
1 3386.765 3396.335     12  6  6
2 3386.765 3392.101      7  6  1
3 3184.124 3191.312      4  4  0
4 3367.123 3424.681      6  6  0
```

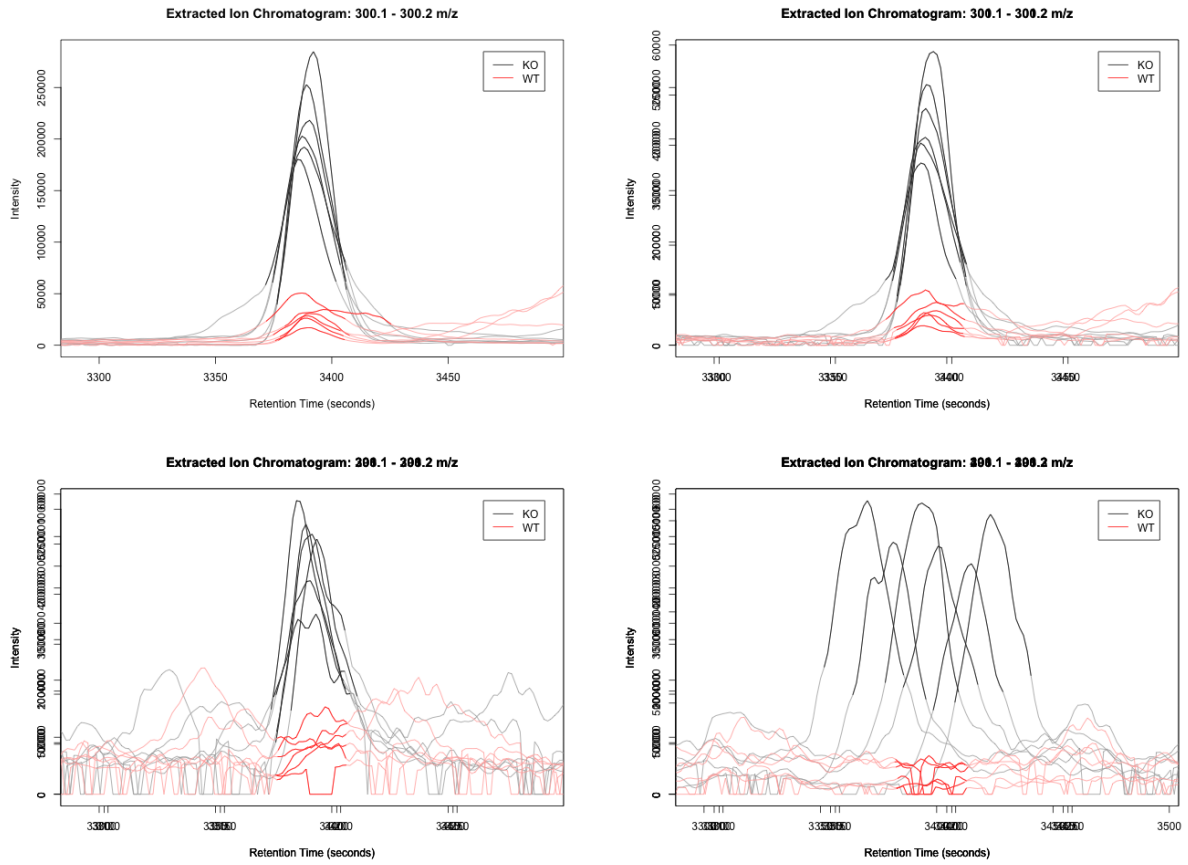

Figure 3: Auto-generated extracted ion chromatograms for the top four differentially regulated ions. Darkened lines indicate where the peaks were integrated for quantitation. The top two plots show the primary and secondary isotopic peaks of an N-acyl ethanolamine (NAE) with a 16 carbon acyl chain. The lower left plot shows the primary isotopic peak of an NAE with a 16 carbon, monounsaturated acyl chain. The lower right plot shows another potential FAAH substrate of unknown identity. Its peaks are not aligned because it is showing a different elution profile than the majority of the other metabolites. Compare it with peaks in the top two plots, which are also eluting at the same time but are correctly aligned.

```
                                                                        metlin
1 http://metlin.scripps.edu/metabo_list.php?mass_min=299.04&mass_max=299.34
2 http://metlin.scripps.edu/metabo_list.php?mass_min=300.04&mass_max=300.34
3     http://metlin.scripps.edu/metabo_list.php?mass_min=297&mass_max=297.3
4 http://metlin.scripps.edu/metabo_list.php?mass_min=490.05&mass_max=490.35
              ko15             ko16             ko18             ko19
1 4534353.62273683 4980914.48421051 5290739.13866664 4564262.89684209
2 962353.429578945 1047934.14136842 1109303.04472222 946943.392842103
3 180780.817277777 203926.952354905 191015.910842105 190626.849523810
4 432037.001363632     332159.07255     386966.75145 334951.452952381
              ko21             ko22             wt15             wt16
1 4733236.07999997      3931592.586  349660.88536842 491793.181333331
2 984787.204999993      806171.4729 86450.4116463556 120096.519533855
3 156869.080488805      220288.6218 16269.0960107969 43677.7839771580
4   294816.2356500 373577.607619048  43466.894054651  13063.634954536
              wt18             wt19             wt21             wt22
1 645526.704947367 634108.848947367 1438254.44559999 1364627.84400000
2 143007.948300124 137319.686021111 218483.142591830 291392.971409092
3 54739.1288875568 76318.0076842747 54726.1153827847 49679.9424723104
4 47098.4179948027 49333.6890058031 12069.7623392174 7329.94597858784
```

If the *metlin* argument 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. For instance, the first and third metabolites in the report produce the following URLs:

- `http://metlin.scripps.edu/metabo_list.php?mass_min=299.04&mass_max=299.34`
- `http://metlin.scripps.edu/metabo_list.php?mass_min=297&mass_max=297.3`

7 Selecting and Visualizing Peaks

It is also possible to generate extracted ion chromatograms for arbitrary peak groups selected using various criteria. Here we generate EICs for two analytes eluting at different times. They are shown using both unaligned and aligned retention times. The resulting plots are shown in Figure 4.

```
> gt <- groups(xset3)
> colnames(gt)
```

```
[1] "mzmed"  "mzmin"  "mzmax"  "rtmed"  "rtmin"  "rtmax"  "npeaks" "KO"
[9] "WT"
```

```
> groupidx1 <- which(gt[, "rtmed"] > 2600 & gt[, "rtmed"] < 2700 &
+     gt[, "npeaks"] == 12)[1]
> groupidx2 <- which(gt[, "rtmed"] > 3600 & gt[, "rtmed"] < 3700 &
+     gt[, "npeaks"] == 12)[1]
> eiccor <- getEIC(xset3, groupidx = c(groupidx1, groupidx2))

ko15 ko16 ko18 ko19 ko21 ko22 wt15 wt16 wt18 wt19 wt21 wt22

> eicraw <- getEIC(xset3, groupidx = c(groupidx1, groupidx2), rt = "raw")

ko15 ko16 ko18 ko19 ko21 ko22 wt15 wt16 wt18 wt19 wt21 wt22

> plot(eicraw, xset3, groupidx = 1)

> plot(eicraw, xset3, groupidx = 2)

> plot(eiccor, xset3, groupidx = 1)

> plot(eiccor, xset3, groupidx = 2)

> cat("These are the warning")

These are the warning

> warnings()

NULL
```

References

Rolf Danielsson, Dan Bylund, and Karin E. Markides. Matched filtering with background suppression for improved quality of base peak chromatograms and mass spectra in liquid chromatography-mass spectrometry. *Analytica Chimica Acta*, 454:167–184, 2002. URL http://dx.doi.org/10.1016/S0003-2670(01)01574-4.

A. Saghatelian, S. A. Trauger, E. J. Want, E. G. Hawkins, G. Siuzdak, and B. F. Cravatt. Assignment of endogenous substrates to enzymes by global metabolite profiling. *Biochemistry*, 43:14332–9, 2004. URL http://dx.doi.org/10.1021/bi0480335.

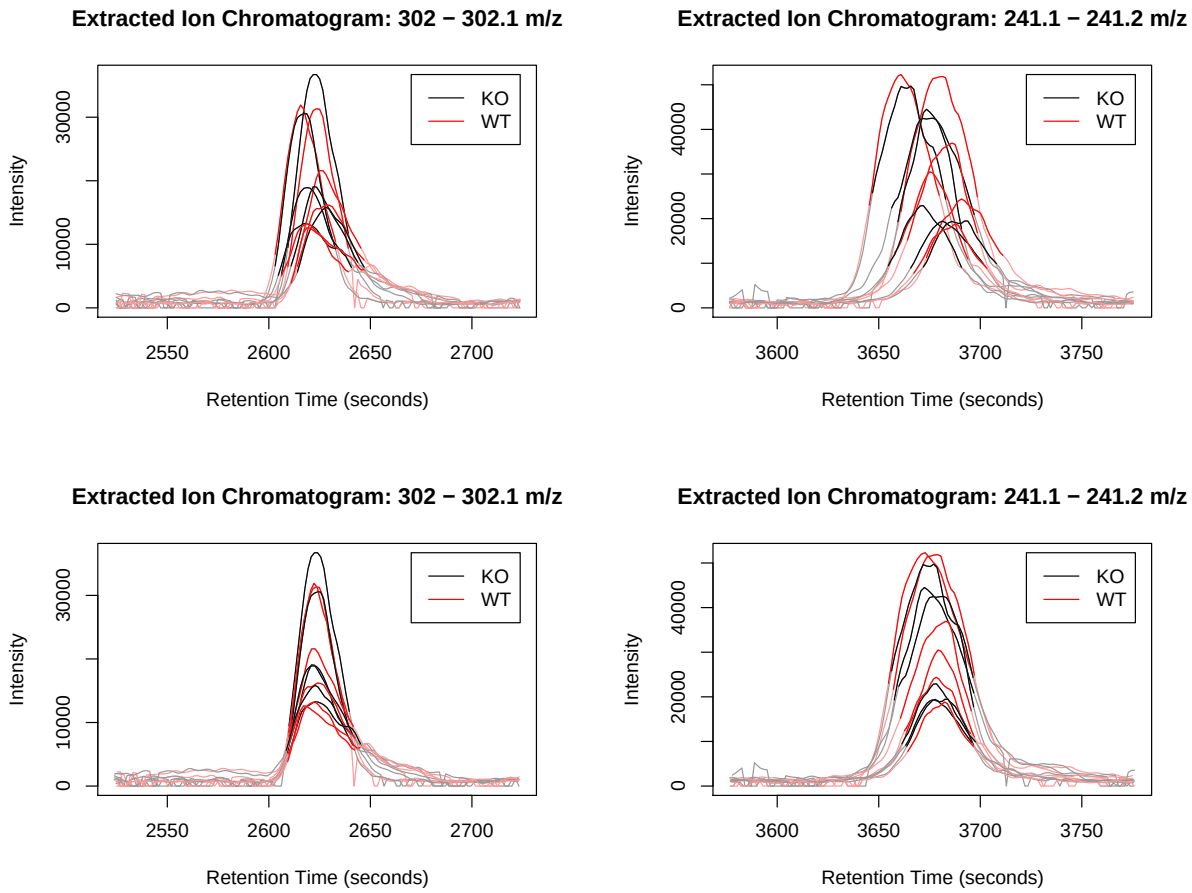

Figure 4: Unaligned (top) and aligned (bottom) extracted ion chromatograms from two analytes eluting at 2624 and 3678 seconds. Darkened lines indicate where the peaks were integrated for quantitation. A plot illustrating the retention time correction is shown in Figure 2.