

# *cosmiq* - COmbining Single Masses Into Quantities

David Fischer  
Christian Panse\*  
Endre Laczko

May 3, 2016

## Contents

---

---

\*[cp@fgcz.ethz.ch](mailto:cp@fgcz.ethz.ch)

## 1 Introduction

---

*cosmiq* is a tool for the preprocessing of liquid- or gas-chromatography mass spectrometry (LCMS/GCMS) data with a focus on metabolomics or lipidomics applications. *cosmiq* has been developed and has shown to be effective using liquid ultra performance capillary chromatography coupled with high accuracy mass data (*full width at half maximum* > 20000), e.g. using TOF or Q-TOF type mass spectrometer. The data we have used consists of one hundreds files having a size of approx. 500MBytes each (see also [?, to be published]).

Because those high resolution data are too huge for being included in the package we will demonstrate the usage of the *cosmiq* package using the smaller *faahKO* data set which is already available on Bioconductor.

The following code of the *cosmiq* wrapper function shows a typical usage:

```
> library(cosmiq)
> cdfpath <- file.path(find.package("faahKO"), "cdf")
> my.input.files <- dir(c(paste(cdfpath, "WT", sep='/'),
+      paste(cdfpath, "KO", sep='/')), full.names=TRUE)
> # run cosmiq wrapper function
> #
> x <- cosmiq(files=my.input.files, mzbin=0.25, SNR.Th=0, linear=TRUE)
> #
>
> # graph result
> image(t(x$eicmatrix), main='mz versus RT map')
> head(x$xs@peaks)
```

The *cosmiq* function is composed of the following steps:

- Combining spectra
- Detecting mz peaks on master spectrum
- Quantifying masses
- RT correction
- Computing the EIC matrix
- Detecting chromatographic peaks from EIC matrix
- Quantifying mz/RT features

*cosmiq* uses the *xcms* [?] object structure for handling the data. The following pages of this vignette are indented to demonstrate how all the steps can be run manually using the *faahKO* data set.

## 2 LCMS feature detection step by step using cosmiq

---

## 2.1 The Input

The faah knockout dataset [?] will be used as input.

```
> library(cosmiq)
> cdfpath <- file.path(find.package("faahKO"), "cdf")
> my.input.files <- dir(c(paste(cdfpath, "WT", sep='/'),
+       paste(cdfpath, "KO", sep='/')), full.names=TRUE)
> #
> # create xcmsSet object
> # todo
> xs <- new("xcmsSet")
> xs@filepaths <- my.input.files
```

Define the phenoData. This is usually done by the unexported method `xcms:::phenoDataFromPaths`.

```
> class <- as.data.frame(c(rep("KO",6),rep("WT", 6)))
> rownames(class) <- basename(my.input.files)
> xs@phenoData <- class
```

The *xcms* object `xs` will be used as container to keep all the data.

```
> attributes(xs)
```

```
$peaks
<0 x 0 matrix>
```

```
$groups
<0 x 0 matrix>
```

```
$groupidx
list()
```

```
$filled
integer(0)
```

```
$phenoData
      c(rep("KO", 6), rep("WT", 6))
ko15.CDF      KO
ko16.CDF      KO
ko18.CDF      KO
ko19.CDF      KO
ko21.CDF      KO
ko22.CDF      KO
wt15.CDF      WT
wt16.CDF      WT
wt18.CDF      WT
wt19.CDF      WT
```

```
wt21.CDF          WT
wt22.CDF          WT

$rt
list()

$filepaths
[1] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/KO/ko15.CDF"
[2] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/KO/ko16.CDF"
[3] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/KO/ko18.CDF"
[4] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/KO/ko19.CDF"
[5] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/KO/ko21.CDF"
[6] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/KO/ko22.CDF"
[7] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/WT/wt15.CDF"
[8] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/WT/wt16.CDF"
[9] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/WT/wt18.CDF"
[10] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/WT/wt19.CDF"
[11] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/WT/wt21.CDF"
[12] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library/faahKO/cdf/WT/wt22.CDF"

$profinfo
list()

$dataCorrection
integer(0)

$polarity
character(0)

$progressInfo
list()

$mslevel
`\\001NULL\\001`

$scanrange
`\\001NULL\\001`

$progressCallback
function (progress)
NULL
<environment: namespace:xcms>

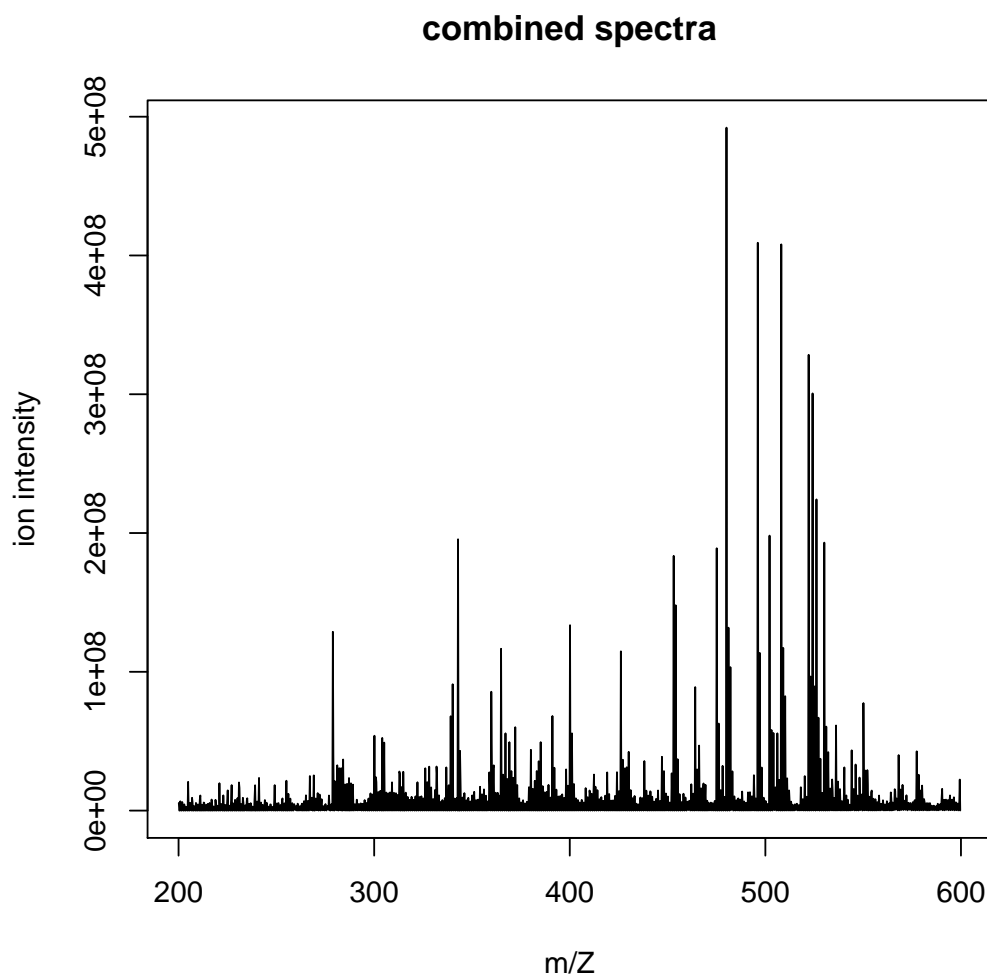
$class
[1] "xcmsSet"
```

```
attr(,"package")  
[1] "xcms"
```

## 2.2 Combination of mass spectra

The first two processing steps search for relevant mass bins in the dataset. In order to select for optimal bins, we first calculate a combined spectrum. This approach of overlaying and summing intensities of single scans together is usual for applications in flow injection mass spectrometry and aims to improve ion statistics. Not only are mass spectra from all scans from a single LCMS run combined but from all acquired datasets. As a result, signal to noise ratio increases for each additional LCMS run and a master list of observed mass is generated.

```
> x <- combine_spectra(xs=xs, mzbin=0.25,  
+                     linear=TRUE, continuum=FALSE)  
> plot(x$mz, x$intensity, type='l',  
+      main='combined spectra',  
+      xlab='m/Z', ylab='ion intensity')  
>
```

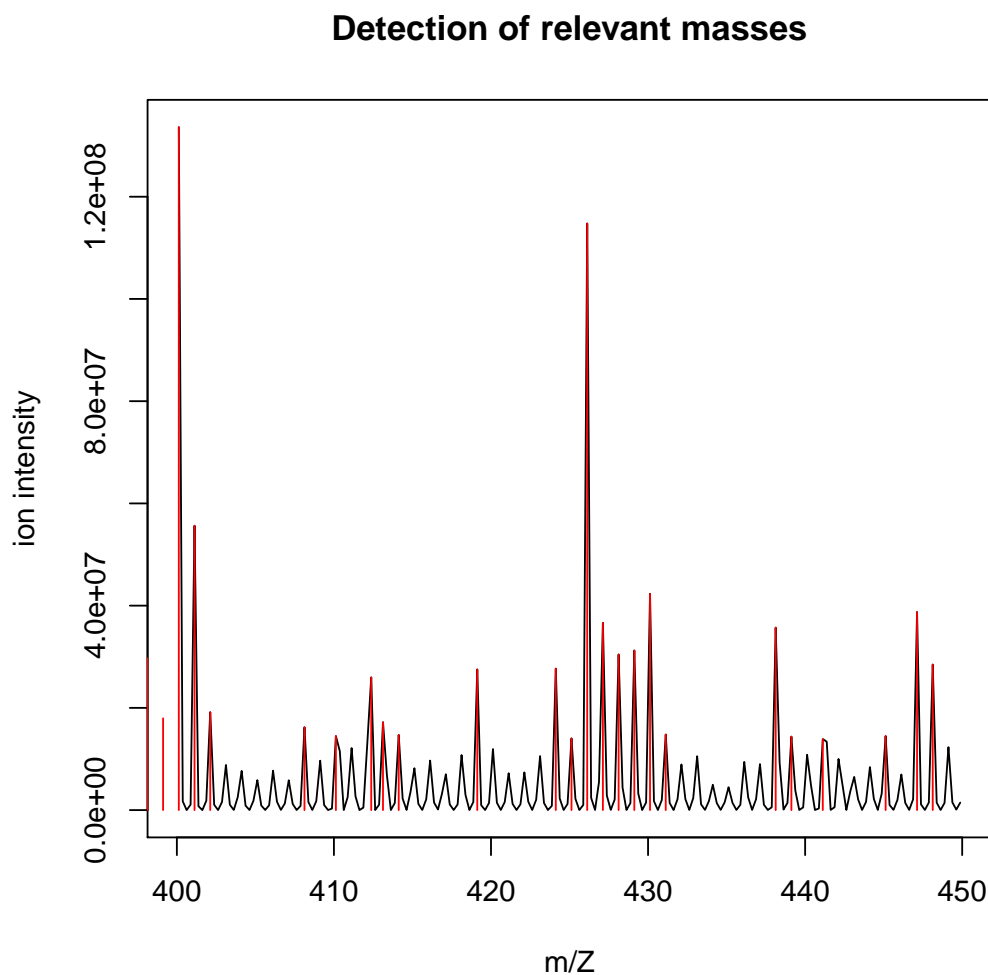


## 2.3 Detection of relevant masses

Based on this combined master mass spectrum we then determine location and boundaries of each observed mass. A modified peak detection algorithm based on continuous wavelet transformation (CWT) is used for this step [?]. Peak detection based on CWT has the advantage that a sliding scale of wavelets instead of a single filter function with fixed wavelength is used. This allows for a flexible and automatic approximation of the peak width. As a result it is possible to locate both narrow and broad peaks within a given dynamic range. The CWT algorithm was modified in order to consider overlapping peaks [?].

```
> xy <- peakdetection(x=x$mz, y=x$intensity,  
+   scales=1:10,  
+   SNR.Th=1.0,  
+   SNR.area=20, mintr=0.5)  
> id.peakcenter<-xy[,4]
```

```
> filter.mz <- 400 < x$mz & x$mz < 450
> plot(x$mz[filter.mz], x$intensity[filter.mz],
+      main='Detection of relevant masses',
+      type='l',
+      xlab='m/Z',
+      ylab='ion intensity')
> points(x$mz[id.peakcenter],
+        x$intensity[id.peakcenter],
+        col='red', type='h')
>
```



## 2.4 Generation and combination of extracted ion chromatograms

Until now only the *mz* information was considered. In the following processing steps the chromatographic information will be added. For the comparison of different LCMS datasets it is important to consider RT shifts. These shifts are typically caused by technical variations and need to be corrected before

chromatographic peaks between different LCMS runs are aligned. For this purpose cosmiq implements xcms retention time alignment using the obiwarp algorithm. For each detected mass in step 2.3 we calculate an extracted ion chromatogram (EIC). In order to determine the elution time for each detected mass, the EICs of every mass are combined between all acquired runs. Again, this combination approach aims for an improvement of the signal-to-noise ratio (SNR).

```
> # create dummy object
> xs@peaks <- matrix(c(rep(1, length(my.input.files) * 6),
+ 1:length(my.input.files)), ncol=7)
> colnames(xs@peaks) <- c("mz", "mzmin", "mzmax", "rt",
+ "rtmin", "rtmax", "sample")
> xs <- xcms::retcor(xs, method="obiwarp", profStep=1,
+ distFunc="cor", center=1)

center sample: ko15.CDF
Processing: ko16.CDF ko18.CDF ko19.CDF ko21.CDF ko22.CDF wt15.CDF wt16.CDF wt18.CDF
>
```

## 2.5 Detection of chromatographic peaks

Based on the combined EICs there is another peak detection step to be performed. The algorithm as described for the peak picking of m/z signals in Step 2.3 is used also for peak picking in the retention time domain. The final result is a peak table with location and boundaries of each mz/RT feature. This information will be further used to locate the relevant position in every single LCMS dataset in order to quantify sample specific feature intensities. Because the mz/RT features were detected on the combined mass spectra or EICs of all samples it is not necessary to align features between different LCMS runs as for a typical raw data processing workflow. Instead, a data matrix with intensity values for every mz/RT feature and every sample can be immediately calculated.

```
> eicmat <- eicmatrix(xs=xs, xy=xy, center=1)
> #
> # process a reduced mz range for a better package build performance
> (eicmat.mz.range <- range(which(475 < xy[,1] & xy[,1] < 485)))

[1] 136 143

> eicmat.filter <- eicmat[eicmat.mz.range[1]:eicmat.mz.range[2],]
> xy.filter <- xy[eicmat.mz.range[1]:eicmat.mz.range[2],]
> #
> # determine the new range and plot the mz versus RT map
> (rt.range <- range(as.double(colnames(eicmat.filter))))

[1] 2501.378 4499.824

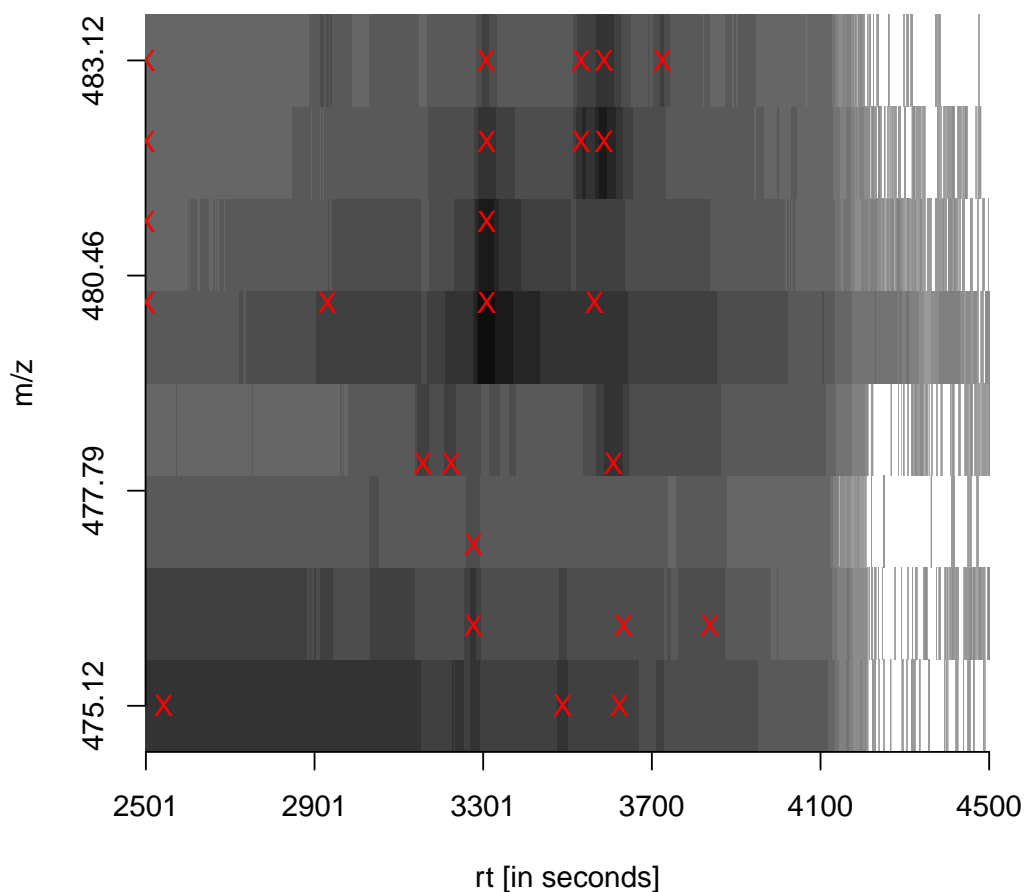
> (mz.range<-range(as.double(row.names(eicmat.filter))))

[1] 475.125 483.125
```



```
> image(log(t(eicmat.filter))/log(2),
+       main='overlay of 12 samples using faahK0',
+       col=rev(gray(1:20/20)),
+       xlab='rt [in seconds]',
+       ylab='m/z', axes=FALSE)
> axis(1, seq(0,1, length=6),
+      round(seq(rt.range[1], rt.range[2], length=6)))
> axis(2, seq(0,1, length=4),
+      round(seq(mz.range[1], mz.range[2], length=4), 2))
> #
> # determine the chromatographic peaks
> rxy <- retention_time(xs=xs,
+   RTscales=c(1:10, seq(12,32, by=2)),
+   xy=xy.filter,
+   eicmatrix=eicmat.filter,
+   RTSNR.Th=120, RTSNR.area=20)
> rxy.rt <- (rxy[,4] - rt.range[1]) / diff(rt.range)
> rxy.mz <- (rxy[,1] - mz.range[1]) / diff(mz.range)
> points(rxy.rt, rxy.mz, pch="X", lwd=2, col="red")
>
```

### overlay of 12 samples using faahKO



## 2.6 Localisation and quantification of detected peaks

With the information about their position in the combined datasets, each individual  $m/z$ /RT feature is then located in the raw data. Due to the retention time correction, each feature is expected at the same RT position as in the combined EIC. However small shifts in retention time still occur for most of the peaks. In order to locate the correct position of each feature, the EIC of the selected mass is calculated for the whole retention time. This EIC is filtered with CWT using only the scale where the feature was optimally located on the combined EIC in step 3. Local maxima are calculated on this transformed data and the maximum with the closest position to the expected retention time is chosen.

```
> xs <- create_datamatrix(xs=xs, rxy=rxy)
```

## 2.7 The Output

The output is a `xcmsSet` object including all necessary information (peak location and peak area), for further data analysis (statistics, metabolite database information).

```
> peaktable <- xcms::peakTable(xs)
> idx <- order(rowSums(peaktable[,8:19]), decreasing=TRUE)
> head(peaktable[idx,])
```

	mz	mzmin	mzmax	rt	rtmin	rtmax	npeaks	ko15.CDF
13	480.125	479.625	480.625	3308.894	3269.770	3346.453	12	50056575
16	481.125	480.625	481.625	3308.894	3269.770	3346.453	12	13099239
20	482.125	481.625	482.625	3587.456	3556.157	3625.015	12	9236049
1	475.125	474.625	475.625	2543.632	2501.378	2620.315	12	5186949
14	480.125	479.625	480.625	3563.981	3517.033	3599.975	12	3793218
18	482.125	481.625	482.625	3308.894	3269.770	3346.453	12	2436222
	ko16.CDF	ko18.CDF	ko19.CDF	ko21.CDF	ko22.CDF	wt15.CDF	wt16.CDF	
13	49188674.0	42604200	32851699	32167083	28232603.9	50843962	53491143.1	
16	12892410.4	11239241	8722476	8416403	7386527.2	13329889	13761044.0	
20	9808221.2	9208821	5944353	6643533	4673246.6	6213538	9478365.3	
1	487408.5	5335581	6752928	1041640	325511.9	5199330	390509.4	
14	3609194.9	3262888	2278957	2281626	1919592.2	3233217	3710765.2	
18	2411749.0	2178509	1850595	1760885	1513305.6	2430864	2558690.9	
	wt18.CDF	wt19.CDF	wt21.CDF	wt22.CDF				
13	44170198	31030988	33940727	26817322.5				
16	11389325	8178813	8871975	7162358.7				
20	8806739	7065545	6784720	5304007.7				
1	4955608	7390373	1091445	332135.6				
14	3778748	2791484	2661820	1700321.2				
18	2246852	1680231	1852049	1488797.2				

```
>
```

## 3 Session information

---

An overview of the package versions used to produce this document are shown below.

- R version 3.3.0 RC (2016-04-26 r70550), x86\_64-apple-darwin13.4.0
- Locale: C/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: Biobase 2.32.0, BiocGenerics 0.18.0, ProtGenerics 1.4.0, Rcpp 0.12.4.5, cosmiq 1.6.0, mzR 2.6.0, xcms 1.48.0
- Loaded via a namespace (and not attached): BiocStyle 2.0.0, MassSpecWavelet 1.38.0, RColorBrewer 1.1-2, codetools 0.2-14, faahKO 1.11.0, grid 3.3.0, lattice 0.20-33, pracma 1.8.8, tools 3.3.0