XCMS PART 1: UNTARGETED & FORMULA-TARGETED PEAK IDENTIFICATION AND VOLCANO ANALYSIS

EMMA SPADY

RHEE LAB

GROUP MEETING – MAY 23, 2023

R TERMINOLOGY

- Variable / object: a name assigned to a value using = or <-
- Class: The type of information in a variable. Could be a string (characters with " around them), numeric, a vector, a matrix, a dataframe...
- Function: function(variablename) performs an operation on the variable
- Directory: The folder or area you're working in, determined by setwd()
- Comment: Text following the # character isn't read into R

For more info, refer to the tutorial from Fernanda's course

- *Introduction to R* - "Introduction", "First Steps in R", "Reading and Writing Data"
- *Introduction to Programming* - "Possible R Workflows - Working with Directories"

Or search Google, StackOverflow, or the xcms package documentation for the relevant function name

XCMS PROCESS

- Read in data from .mzML files and your custom .csv files
- Prepare settings
- Find peaks in each sample with findChromPeaks()
- Merge overlapping peaks in each sample with refineChromPeaks()
- Adjust retention time of all samples with adjustRtime()
- Define features – sets of peaks with similar m/z and RT between samples - with groupChromPeaks()
- Fill peaks – integrate across the median m/z and RT for each feature in samples where no peak was found – using fillChromPeaks()

Returns a table with features, each with a m/z, a RT and a value for each sample, along with some other info.

ANALYSIS PROCESS – AS OF MAY 2023

- Save the output table
- Perform volcano analysis on all features in dataset
  - Calculate log2(foldchange)s and t-test p-values
  - Make a volcano plot for each comparison
  - Print chromatograms for 50 highest foldchange significant features
- Filter features for those with m/z that could correspond to the [M+H]+ or [M-H]- of your metabolite set.
- Perform volcano analysis on filtered features
  - Same as above. Fewer features make MHC less strict

FYI, presentation is not in the same order as the code. The code groups parts that you need to change together – see manual for operation.

RUNNING THE SCRIPT

- Leaving this for the instruction manual
- Requires some customization
  - Set own working directory
  - Settings must match instrument, method, and whether you have QC alignment samples
- You'll need to run it in chunks, rather than trying to push the 'run' button at the top

FILE SETUP

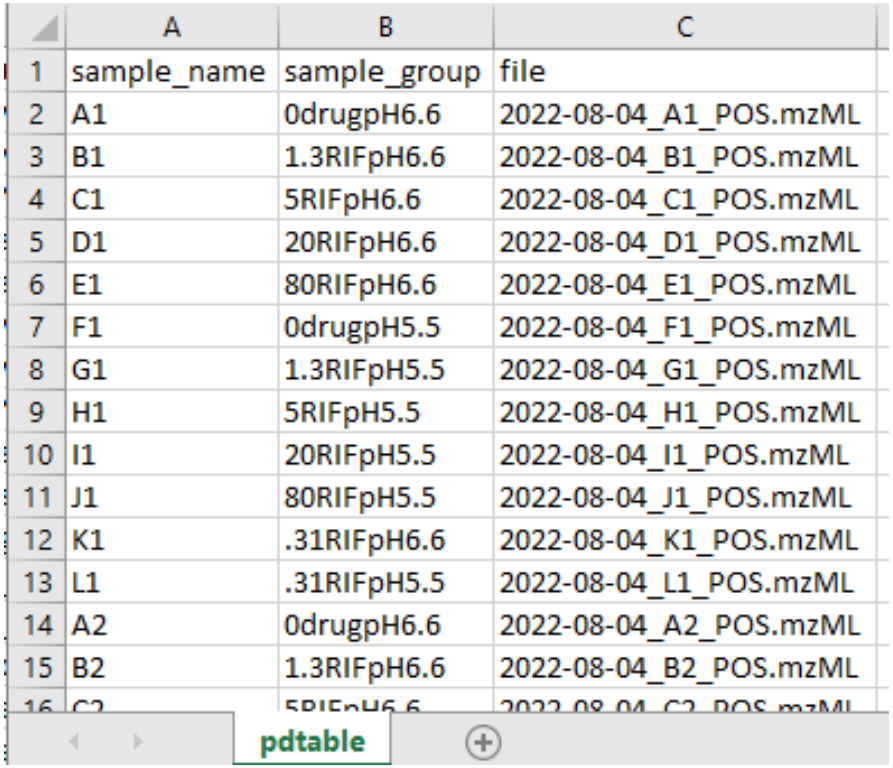
- Copy the entire template folder, then paste into the XCMS folder and rename for your project.
- Convert .d files to .mzML format using MSConvertGUI. Put them inside the rawdata folder.
- Edit the .csv files to refer to your experiments
  - conditionkey.csv assigns each sample group a short label, a color, and a group for plotting chromatograms.
  - pdtable.csv assigns each data file a label that matches a sample group. It also puts them in the order they were run.
  - comparekey.csv specifies which sample groups should be compared to each other for volcano plots

	A	B	C	D	E
1	label	color	plotgroup	pH	RIFdose
2	A	#8ba4b1	neu	pH 6.6	0
3	B	#7BCCC4	neu	pH 6.6	1.25
4	C	#4EB3D3	neu	pH 6.6	5
5	D	#2B8CBE	neu	pH 6.6	20
6	E	#08589E	neu	pH 6.6	80
7	F	#b1988b	acd	pH 5.5	0
8	G	#FD8D3C	acd	pH 5.5	1.25
9	H	#FC4E2A	acd	pH 5.5	5
10	I	#E31A1C	acd	pH 5.5	20
11	J	#B10026	acd	pH 5.5	80
12	K	#A8DDB5	neu	pH 6.6	0.3125
13	L	#FEB24C	acd	pH 5.5	0.3125
14					

conditionkey

FILE SETUP

- Copy the entire template folder, then paste into the XCMS folder and rename for your project.
- Convert .d files to .mzML format using MSConvertGUI. Put them inside the rawdata folder.
- Edit the .csv files to refer to your experiments
  - conditionkey.csv assigns each sample group a short label, a color, and a group for plotting chromatograms.
  - pdtable.csv assigns each data file a label that matches a sample group. It also puts them in the order they were run.
  - comparekey.csv specifies which sample groups should be compared to each other for volcano plots



	A	B	C
1	sample_name	sample_group	file
2	A1	0drugpH6.6	2022-08-04_A1_POS.mzML
3	B1	1.3RIFpH6.6	2022-08-04_B1_POS.mzML
4	C1	5RIFpH6.6	2022-08-04_C1_POS.mzML
5	D1	20RIFpH6.6	2022-08-04_D1_POS.mzML
6	E1	80RIFpH6.6	2022-08-04_E1_POS.mzML
7	F1	0drugpH5.5	2022-08-04_F1_POS.mzML
8	G1	1.3RIFpH5.5	2022-08-04_G1_POS.mzML
9	H1	5RIFpH5.5	2022-08-04_H1_POS.mzML
10	I1	20RIFpH5.5	2022-08-04_I1_POS.mzML
11	J1	80RIFpH5.5	2022-08-04_J1_POS.mzML
12	K1	.31RIFpH6.6	2022-08-04_K1_POS.mzML
13	L1	.31RIFpH5.5	2022-08-04_L1_POS.mzML
14	A2	0drugpH6.6	2022-08-04_A2_POS.mzML
15	B2	1.3RIFpH6.6	2022-08-04_B2_POS.mzML
16	C2	5RIFpH6.6	2022-08-04_C2_POS.mzML

pdtable

FILE SETUP

- Copy the entire template folder, then paste into the XCMS folder and rename for your project.
- Convert .d files to .mzML format using MSConvertGUI. Put them inside the rawdata folder.
- Edit the .csv files to refer to your experiments
  - conditionkey.csv assigns each sample group a short label, a color, and a group for plotting chromatograms.
  - pdtable.csv assigns each data file a label that matches a sample group. It also puts them in the order they were run.
  - comparekey.csv specifies which sample groups should be compared to each other for volcano plots

	A	B	C	D
1	expcond	nullcond	compareID	category
2	K	A	RIF.31vsNoDrug_neu	RIFneu
3	B	A	RIF1.3vsNoDrug_neu	RIFneu
4	C	A	RIF5vsNoDrug_neu	RIFneu
5	D	A	RIF20vsNoDrug_neu	RIFneu
6	E	A	RIF80vsNoDrug_neu	RIFneu
7	L	F	RIF.31vsNoDrug_acd	RIFacd
8	G	F	RIF1.3vsNoDrug_acd	RIFacd
9	H	F	RIF5vsNoDrug_acd	RIFacd
10	I	F	RIF20vsNoDrug_acd	RIFacd
11	J	F	RIF80vsNoDrug_acd	RIFacd
12	F	A	NoDrug_acdvsneu	pHcompare1
13	L	K	RIF.31_acdvsneu	pHcompare2
14	G	B	RIF1.3_acdvsneu	pHcompare3
15	H	C	RIF5_acdvsneu	pHcompare4
16	I	D	RIF20_acdvsneu	pHcompare5
17	J	E	RIF80_acdvsneu	pHcompare6

comparekey

FILE SETUP

- Some other files:
- QCmols_ANP-NEG.csv and QCmols_ANP-POS.csv are a small set of molecules with known retention times that can be used to check your peak finding parameters. Feel free to change or add your own molecules
- PwaySet_ESpady01.txt has a set of 1,043 unique formulae from KEGG pathways in *Mycobacterium tuberculosis*. It will be used for targeted-unbiased feature filtering.
- customEICinfo.csv is a template to help you load in ranges of m/z and retention times for custom chromatograms
- [Scripts.R] contains my custom functions. It gets loaded into the main script automatically.

PEAK FINDING

- Dataset is intensities over m/z and time
- m/z spikes are tight and easy to find, so xcms starts by choosing m/z ranges
- Then uses algorithm to find where to start and end the time for each peak
- Settings defined with CentWaveParam()
  - Differs for TOFs versus QTOFs

```
cwp <- CentWaveParam(
  ppm = 25,                  #ppm error within each file within each peak. 25 is generous
  peakwidth = c(8, 30),     #min and max expected peak width, in seconds
  noise = 2000,              #remove all reads with intensity below this threshold
  snthresh = 10,             #signal-to-noise ratio cutoff. Not really sure what it does.
  prefilter = c(3, 3300))    #Require at least x reads with intensity >= y to look in a mass slice
```

- Run with findChromPeaks()

PEAK MERGING

- XCMS is a little overexcited about peaks, and is likely to break jagged peaks into separate but overlapping parts
- It merges the peaks after finding them to compensate
- Settings defined with MergeNeighboringPeaksParam()

```
#Peak merging parameters:
mpp <- MergeNeighboringPeaksParam(
  expandRt = 4,        #how close, in seconds, do two peaks need to be to count as overlapping?
  ppm = 10,            #how close, in m/z ppm, do the peaks need to be?
  minProp = 0.50)      #The lowest point between two peaks must be over this fraction of the lower peak's height
```

- Run with refineChromPeaks()

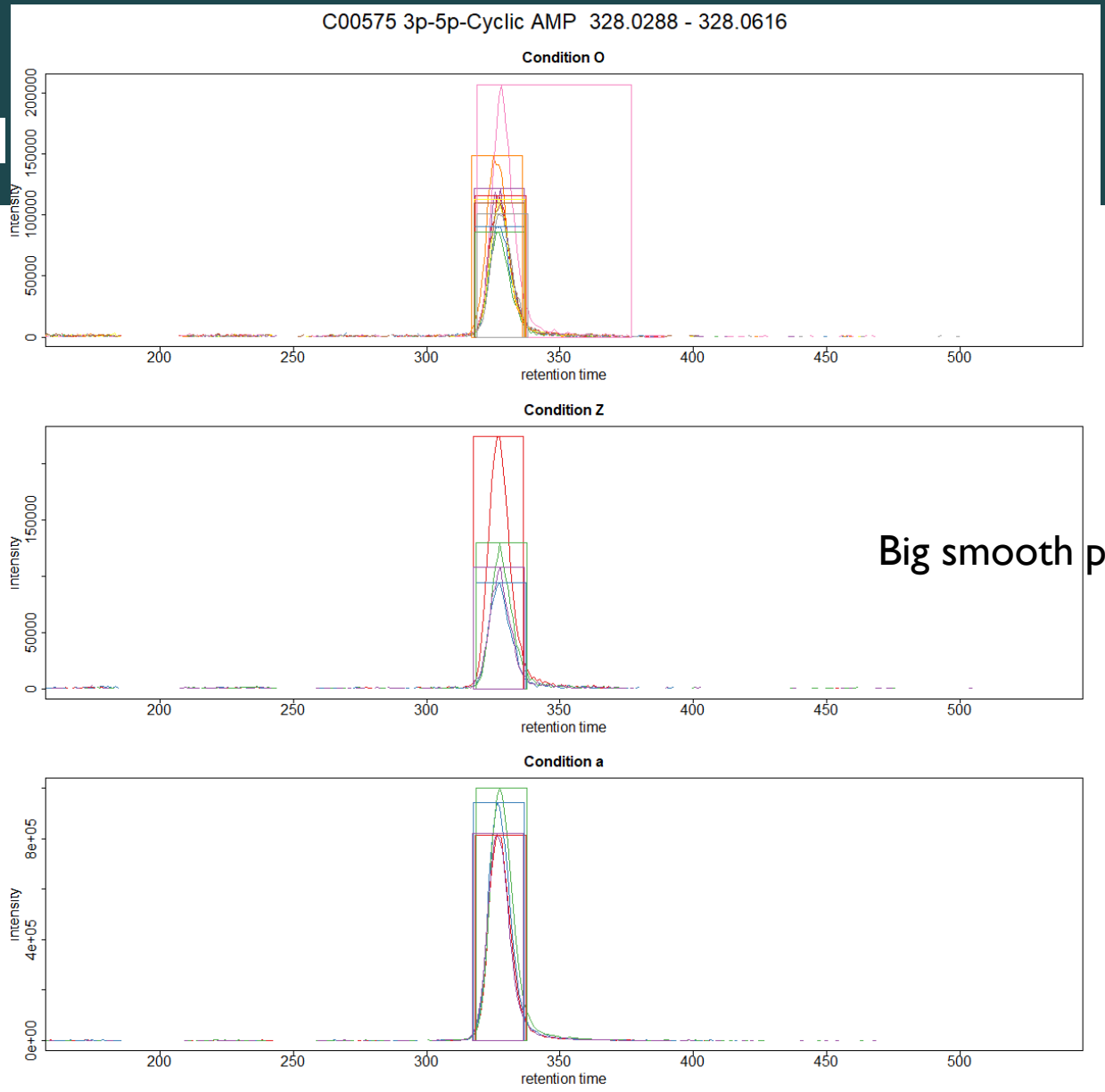
PEAK FINDING AND MERGING QC

- You can run the peak finding and merging functions on an extracted ion chromatogram to check your settings
- QC set has some molecules that should be in your samples at varying intensities / pattern complexities
- Select a few sample groups from a big experiment to check, using their label
- chromatogram() makes an EIC dataset that can be printed out, but it is VERY SLOW.
- ChromsByConds() is my custom function for plotting the EIC with the attempt at peak finding

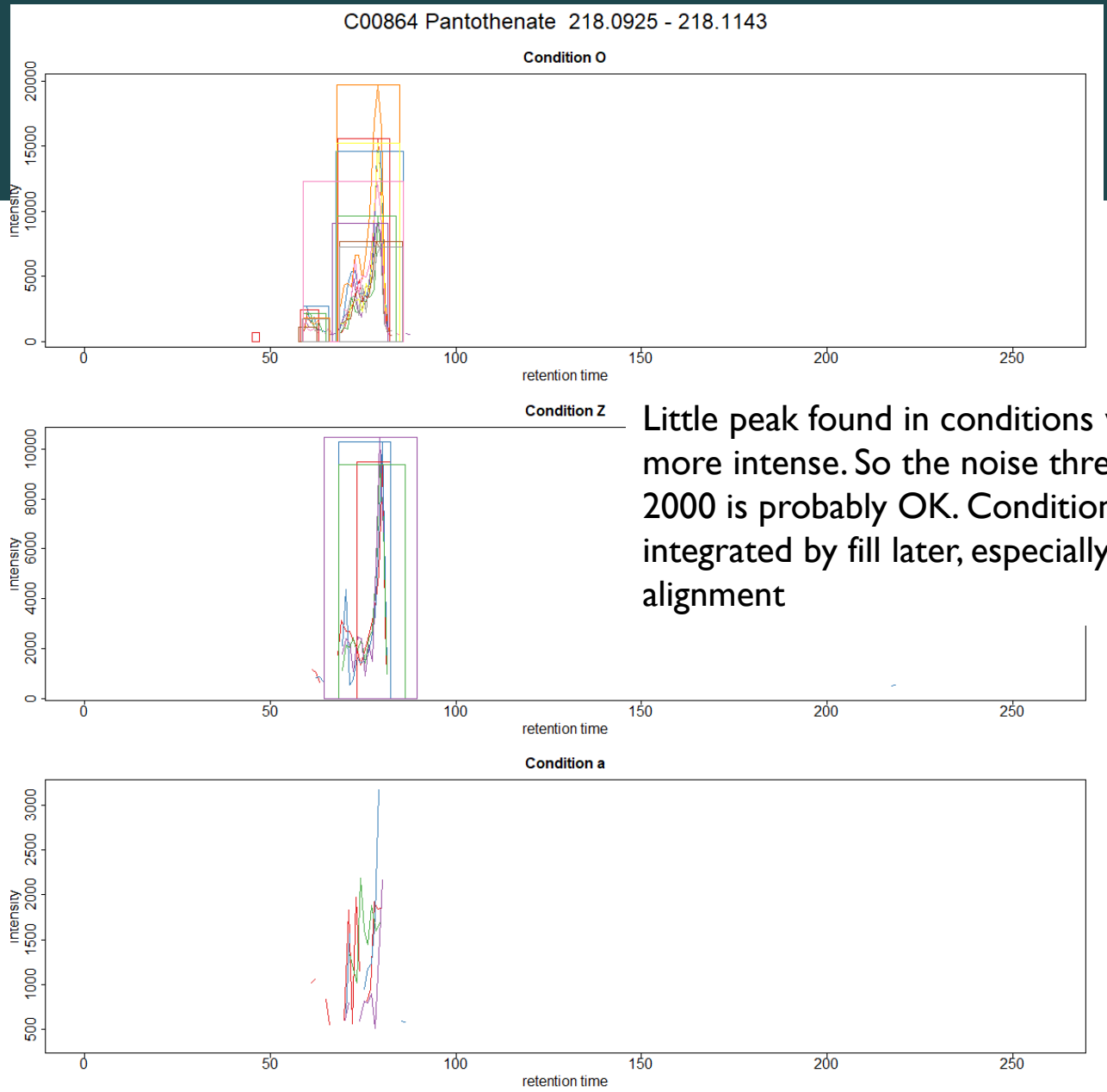
READING QC EICS

- .png files made by ChromsByConds()
- Replicates are different colors, panel shows each condition
- Box represents a found peak
- Confirm peaks are in boxes, not split oddly down the middle, not lumping too many peaks together
- OK if some small peaks are not in boxes – they'll be fixed in peak fill
- Note that Y axis scale is NOT matched between conditions

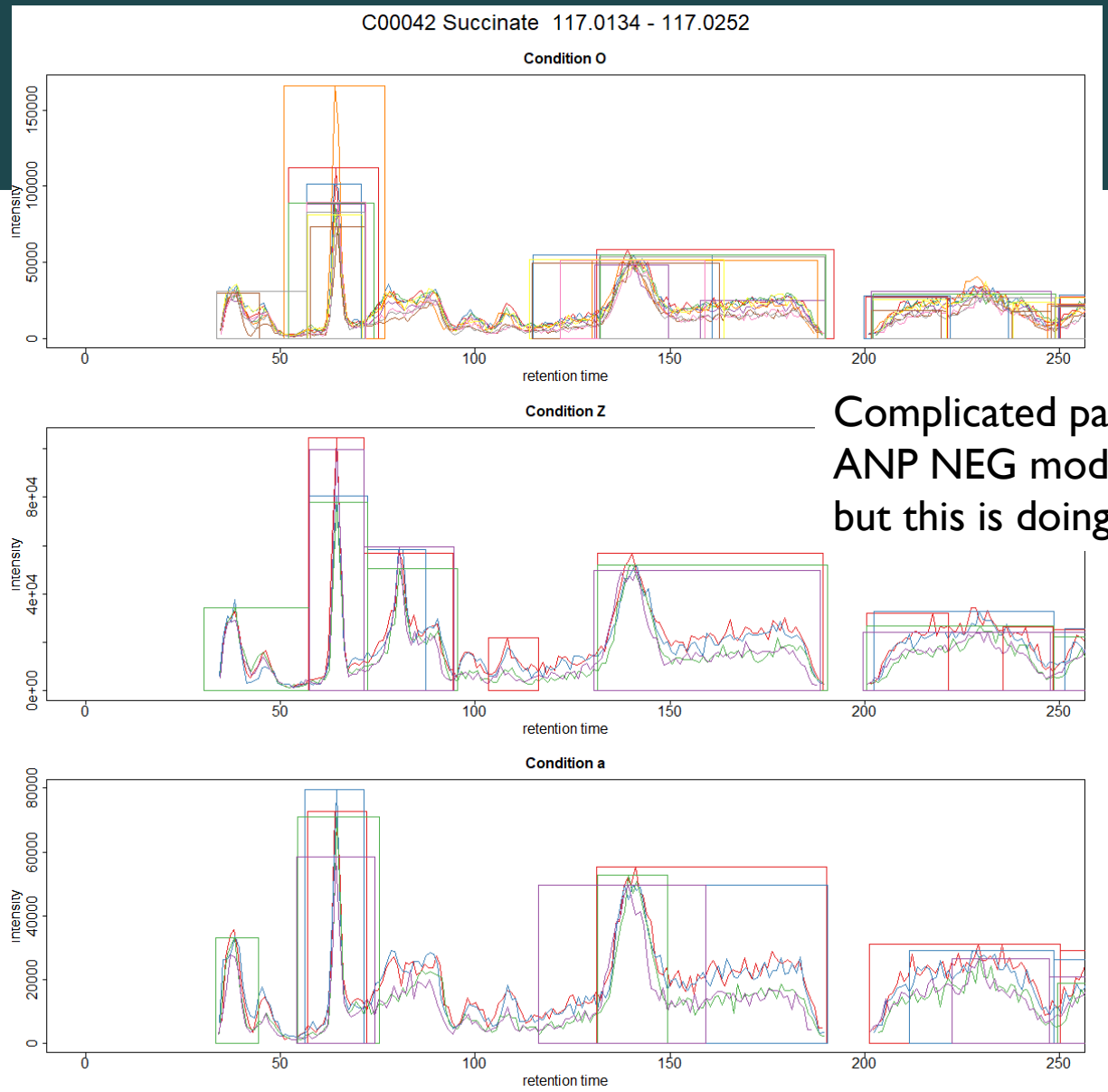
READI



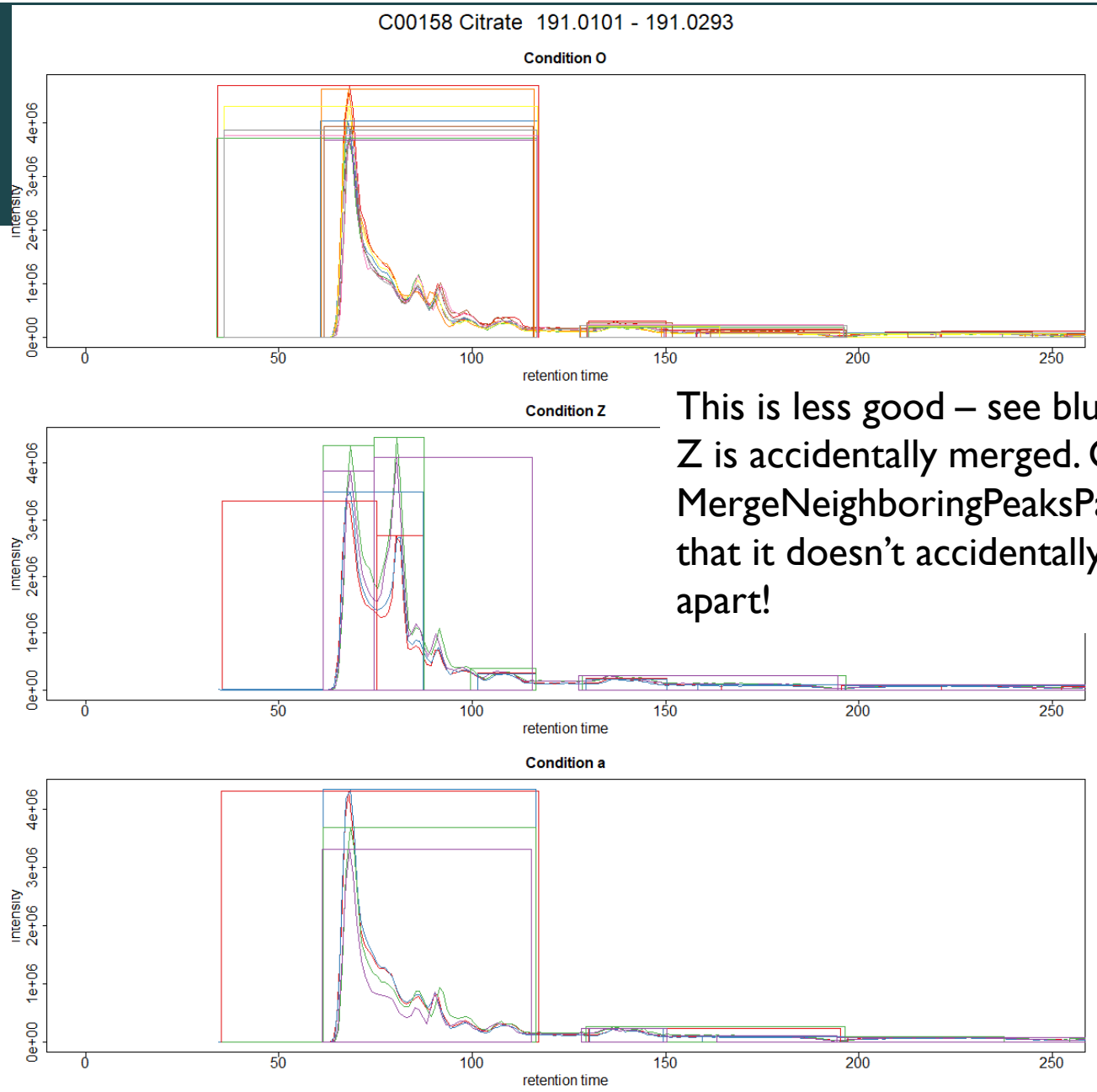
Big smooth peak found easily



Little peak found in conditions where it is more intense. So the noise threshold of 2000 is probably OK. Condition a will get integrated by fill later, especially with RT alignment



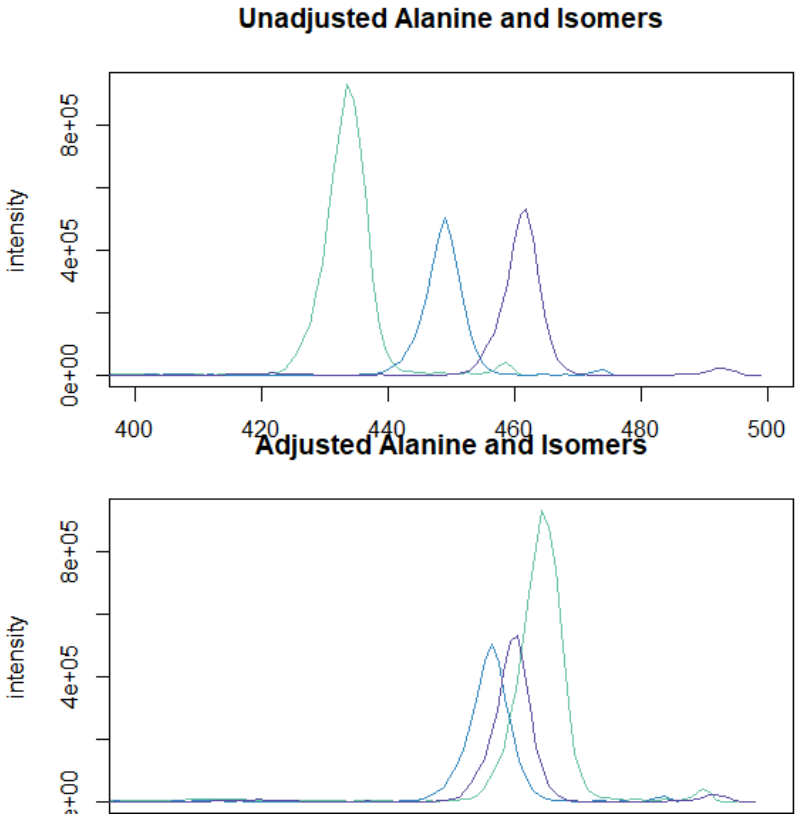
Complicated patterns more common in ANP NEG mode than ANP POS mode, but this is doing OK



This is less good – see blue peak in Condition Z is accidentally merged. Consider changing MergeNeighboringPeaksParam(), but be careful that it doesn't accidentally split whole peaks apart!

RETENTION TIME ALIGNMENT

- Uses the identified peaks to align all the samples
- Facilitates peak grouping
- Can use QC samples to align very precisely between sample runs
  - Still best if you run samples within 3 weeks
- Settings defined with ObiwarpParam()
- Run with adjustRtime()

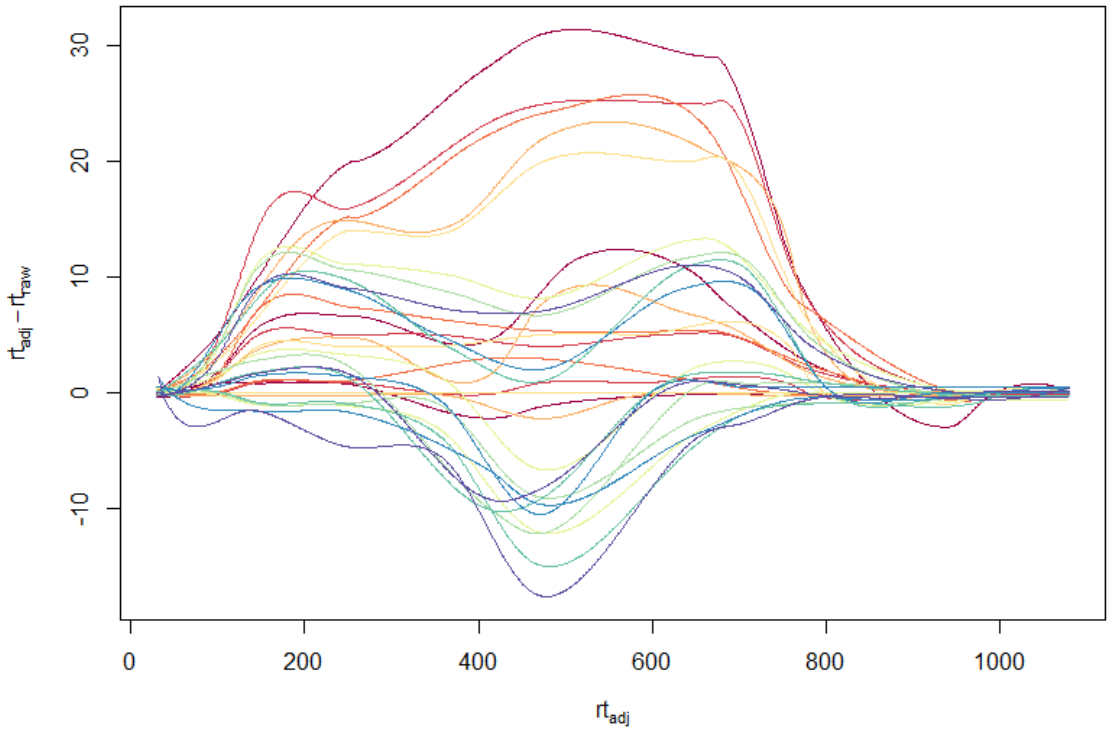


It works better than this for recent data because of improved LC maintenance

RETENTION TIME ALIGNMENT CHECK

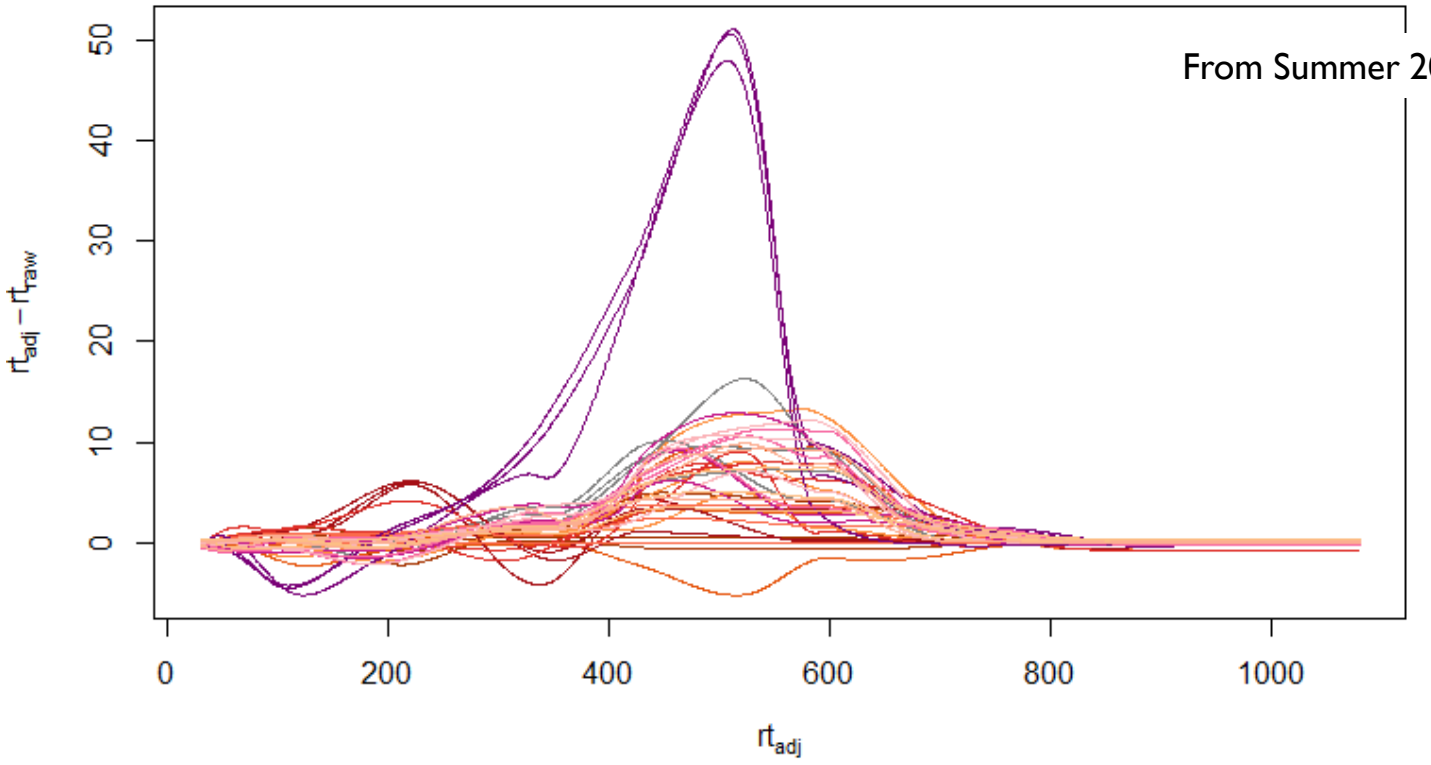
plotAdjustedRtime(xdata_mrg, col = pdextra[,"sample_color"])

From Summer 2022 I think



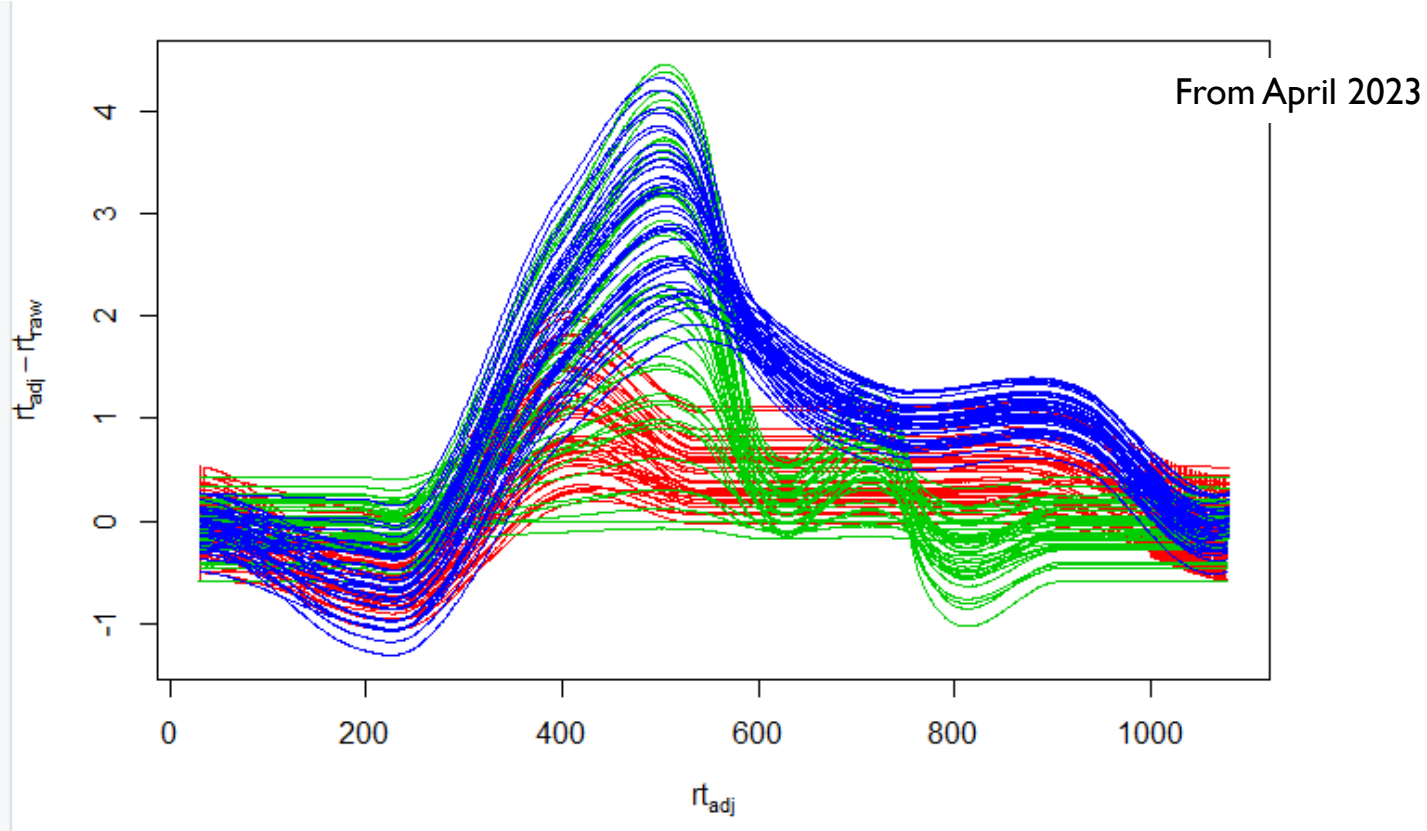
RETENTION TIME ALIGNMENT CHECK

plotAdjustedRtime(xdata_mrg, col = pdextra[,"sample_color"])



RETENTION TIME ALIGNMENT CHECK

- You can color by batch set if you can code a little



FEATURE GROUPING

- Groups peaks with similar m/z and RT across samples into a single ‘feature’
- Output is measured by feature, each with unique ID
- Settings defined with PeakDensityParam()

```
#Feature grouping parameters:
pdp <- PeakDensityParam(
  sampleGroups = pdtable[,"sample_group"],
  minFraction = 0.8,    #Minimum fraction of samples in at least one sample group that have a peak
  bw = 8,               #Adjusts the allowed retention time difference between peaks to group them in the same feature.
  binSize = 0.01)       #Adjusts the m/z difference allowed between peaks to get them grouped in the same feature. I'm
```

- Run with groupChromPeaks()

PEAK FILL

- If a feature group lacks a peak in a sample, xcms can integrate any intensity between the median minimum RT and median maximum RT at the relevant mass
- Filled peaks do not get a box around them when you look at feature chromatograms, but they are there
- Settings defined with ChromPeakParam()
  - I use the defaults
- Run with fillChromPeaks()

DATA OUTPUT

- Processed data is originally in xdata_ftg
- Extract information using functions
  - featureDefinitions(xdata_ftg) shows m/z and RT info for each feature
  - featureValues(xdata_ftg) shows the peaks areas for each sample. Can show peak heights using further arguments
- This info will be saved into the file *_AllFeatures.txt
  - * is your experiment abbreviation, which is assigned to exptkey in the settings
  - Prefix helps with analyzing multiple experiments

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	RZ01_NEG_FeatureID	mzmed	mzmin	mzmax	rtmed	rtmin	rtmax	npeaks	RZ01_NEG_qc_1	RZ01_NEG_O_1	RZ01_NEG_A_1	RZ01_NEG_B_1	RZ01_NE
2	FT0001	51.02404	51.0239	51.02415	139.515	137.8854	141.5154	119	60686.16046	59237.66094	71606.62112	71375.5437	66027.64
3	FT0002	53.46454	53.46343	53.46563	195.4555	192.0347	197.534	93	29637.58979	32922.98815	23440.82541	36148.25479	27844.34
4	FT0003	53.51446	53.51349	53.51529	195.4427	192.6921	198.5328	116	49541.35126	56135.54692	48868.46409	57550.06393	59898.67
5	FT0004	53.56429	53.56336	53.56889	195.5327	191.8728	200.5321	124	66499.67243	61285.28234	66021.884	72221.19505	75242.88
6	FT0005	53.61405	53.61328	53.61825	195.533	192.6921	198.5283	119	73232.54783	87215.76812	75616.66862	87353.13085	91049.27
7	FT0006	53.66372	53.66277	53.66825	195.5316	190.6727	198.5335	120	77769.4123	89394.72448	75638.02843	87886.95716	91779.51

My exptkey is “RZ01_NEG” so my file is RZ01_NEG_AllFeatures.txt

FORMULA-TARGETED FEATURE IDENTIFICATION

- By focusing only on features with m/z that could correspond to a known formula, we make our feature list smaller
  - Makes multiple hypothesis correction less strict
- Assign a table of unique formulae to mymols
  - Currently uses PwaySet_ESpady01.txt, which is 1,043 unique formulae from about 40 central Mtb KEGG pathways
- Match features against predicted $[M+H]^+$ or $[M-H]^-$ mass using FeatMatching()

```
featfilter = FeatMatching(featdefs = featdefs,    #The feature definitions from xdata_ftg
                          molset = mymols,        #Formulae with masses to match
                          mode = setmode,     #Mode, either "POS" or "NEG"
                          mzerr = seterr,     #Error for m/z match, Default 50 ppm
                          rterr = 120)        #Error for retention time. Not used yet
```

- Result is a set of features that are within seterr (default 50 ppm) of at least one formula.
  - <5% of the features will be in the featfilter set

VOLCANO ANALYSIS

- Run using MultiVolcano(), which references your comparekey.csv
  - Takes p-values and log2 foldchanges for each feature and each comparison
  - Makes an interactive volcano plot for each feature and each comparison

```
volctest_PwaySet = MultiVolcano(dataset = xdata_ftg,              #Results as XCMSnExp class
                                ftsubset = rownames(featfilter),  #The features you matched
                                molmatchinfo = featfilter,        #Table with feature names and match candidates
                                comparekey = tTestKey,            #The comparison set from your csv
                                pthresh = 0.05,                   #p-value significance threshold
                                l2fcthresh = 1)                   #log2 fold change significance threshold
```

- Code titled #Save the feature information... saves all features that were significant to a table *_SigPwaySet.txt
- Code titled #Print chromatograms... makes chromatograms for the top 50 significant, ranked by foldchange features within each 'category'.
  - If you want to see top 50 for all comparisons, make each 'category' unique

VOLCANO PLOTS

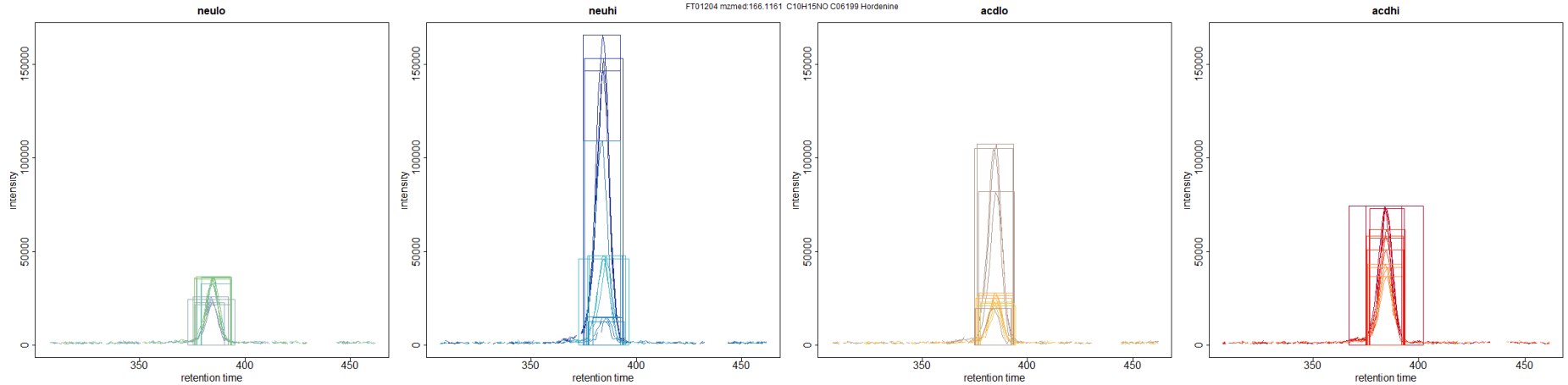
- See .html files

PRINTING CHROMATOGRAMS

- First, make chromatogram object from dataset
  - For chromatograms of features, use featureChromatograms()
  - For an EIC, use chromatogram(). But I can't print this yet.
- This function is slow because it references the .mzML files, so it's best to batch features or (m/z,RT) ranges together at this step
- Each feature prints into its own .png file

PRINTING CHROMATOGRAMS

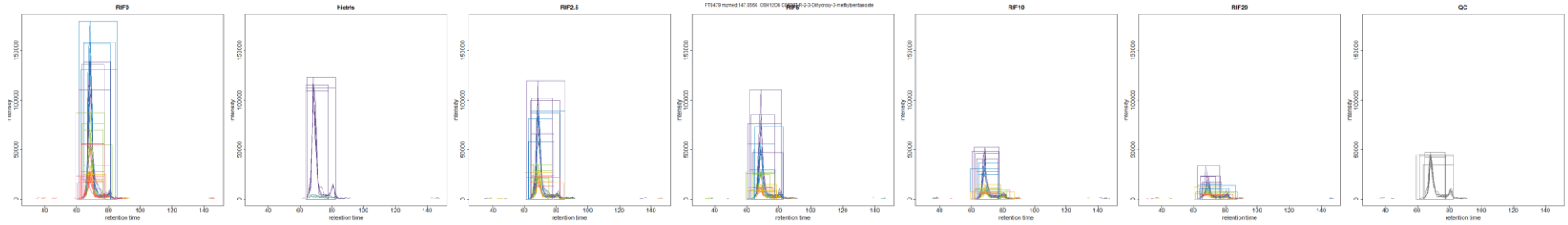
- Here, green is lower doses of RIF, blue is higher doses, at neutral pH. Orange is lower doses of RIF, red is higher doses. Gray is the no-drug control at each pH
- You'll have to remember your color key, because sadly I don't have it print automatically on these chromatograms. It could accidentally overlap important content.



PRINTING CHROMATOGRAMS

From RIF-PZA checkerboard, spring 2023

Always comes out as a long strip, but the image is larger so you have the same resolution



From RIF-PZA checkerboard, spring 2023

Titled: FT0479 C06007 R-2-3-Dihydroxy-3-methylpentanoate