

Paramounter.R User Manual

(Version 2, 2022-02-02)

Jian Guo¹, Tao Huan^{1,*}

¹ Department of Chemistry, Faculty of Science, University of British Columbia, Vancouver Campus, 2036 Main Mall, Vancouver, V6T 1Z1, BC, Canada

* Author to whom correspondence should be addressed:

Dr. Tao Huan

Tel: (+1)-604-822-4891

E-mail: thuan@chem.ubc.ca

Internet: <https://huan.chem.ubc.ca/>

- Paramounter.R is an R script to directly measure parameters for processing LC-MS-based metabolomics data
- The program is written in the language ‘R’ and is publicly available at <https://github.com/HuanLab/Paramounter>
- Please follow the instructions for Paramounter.R

- 1) Create a folder to store the files used for parameter optimization (**Figure 1**). Use 3-4 sample or QC LC-MS data files from the beginning, middle, and end of the sample analysis. For using mzML files, please change line 20 in the Paramounter R code part 1 and line 26 in part 2 to “filename <- list.files(pattern = “.mzML”)”.

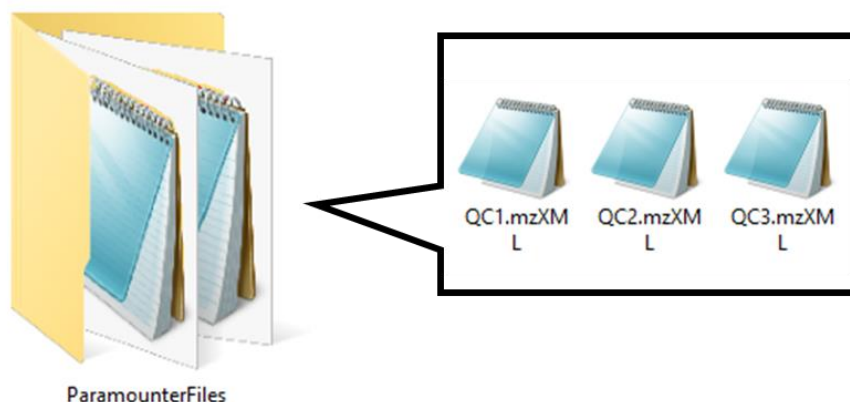


Figure 1. The file organization used for Paramounter.

- 2) Download “Paramounter_part1 (V2).R” and “Paramounter_part2 (V2).R” from GitHub (<https://github.com/HuanLab/Paramounter>) (**Figure 2**).

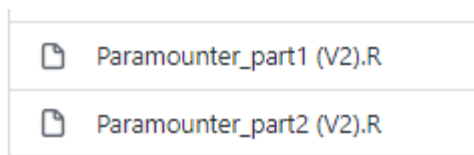


Figure 2. Paramounter part 1 and 2 on GitHub.

- 3) Use R Version 4.0 or newer. Install “xcms”, “MSnbase”, “dplyr”, “ggplot2”, and “gridExtra” libraries, if they were not installed previously. All packages should be updated to the newest available version.

- 4) Set the directory, massSDrange, smooth, and cutoff values in Paramounter part 1 (**Figure 3**).
 directory: The data path of the mzXML files.

massSDrange: The range of standard deviation of the mass differences in each ZOI. Default value is 2 (95% confidence). Users can adjust this value (e.g., 3 for 99.7% confidence).

smooth: Chromatographic smoothing level that is used in an embedded function to smooth out a chromatographic peak to better determine ZOIs. Default value is 0. For DDA data, use default value. For full-scan data with acquisition rate > 2 Hz, use a positive smooth value.

For example, smooth = 4 for spectral rate = 4 Hz.

cutoff: The ppm value for determining the cutoff line in the mass accuracy distribution. The default is 0.95 (95%). User can adjust this value based on their own discretion (e.g., 0.9 for less noise included in the parameter measurement).

```

1 #####
2 #This is the script to perform parameters estimation
3 #Jian Guo, Tao Huan 2021-07-28
4 #Copyright @ University of British Columbia
5 #####
6
7 library(xcms)
8 library(MSnbase)
9 library(dplyr)
10 library(ggplot2)
11 library(gridExtra)
12
13 # User input the directory and software to optimize parameters for (XCMS, MSIAL, or MZMINE2)
14 directory <- "F:/Jian_Guo/SoftwareComparison_20211103/singleDatafilecompare20220207/New10datasetResults20220204/BrukerUrineRPdiluted10DDA"
15 massSDrange <- 2
16 smooth <- 0
17 cutoff <- 0.95
  
```

Figure 3. Example of parameters set up in Paramounter part 1.

- 5) Run Paramounter part 1 by clicking “Source” in the top right corner of the document window in RStudio. Paramounter part 1 generates a plot showing the distribution of mass accuracy in the dimension of m/z and the cutoff line (red line in **Figure 4**).

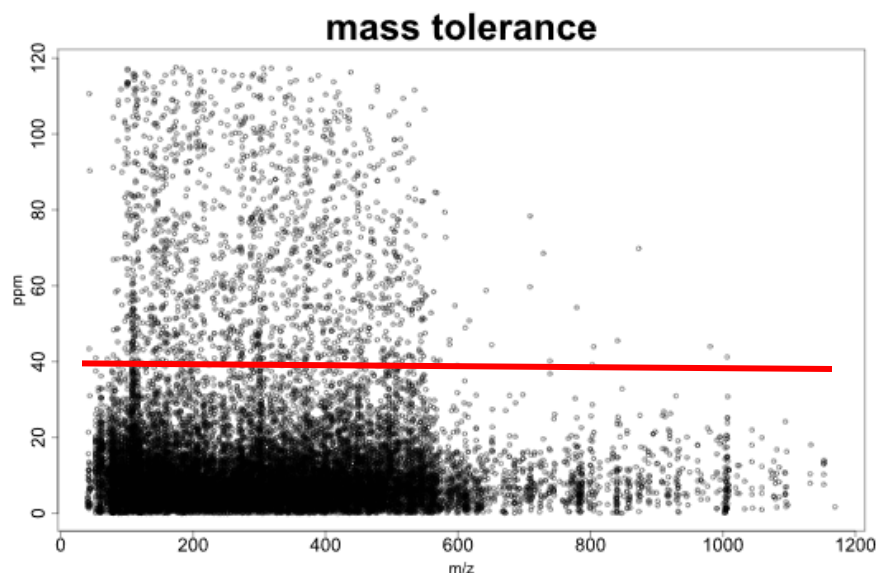


Figure 4. The mass accuracy distribution generated by Paramounter part 1.

- 6) Set the directory, Software, massSDrange, ppmCut, and smooth values in Paramounter part 2 (**Figure 5**).

directory: The same as part 1. The data files used for part 2 should be the same as part 1.

Software: The name of the software for parameter output. Select either XCMS, MSDIAL, MZMINE2, or ALL for any one of or all three software-specific parameters. The universal parameters are automatically generated together with the software-specific parameters.

massSDrange: The same as part 1.

ppmCut: the mass accuracy cutoff determined from Part 1.

Smooth: The same as part 1.

```

1 #####
2 #This is the script to perform parameters estimation
3 #Jian Guo, Tao Huan 2021-07-28
4 #Copyright @ University of British Columbia
5 #####
6
7 library(xcms)
8 library(MSnbase)
9 library(dplyr)
10 library(ggplot2)
11 library(gridExtra)
12
13 directory <- "F:/Jian_Guo/Paramounter_paper_20210421/Response_20211219/10datasetRED0withIPOAutoTuner_20220120/BrukerUrineHILICoriginalDOA/parameter"
14 # User input the directory and software to optimize parameters for (XCMS, MSDIAL, MZMINE2, or ALL)
15 # Available choices:
16 # "XCMS" for XCMS-based data processing
17 # "MSDIAL" for MSDIAL-based data processing
18 # "MZMINE2" for MZmine2-based data processing
19 # "ALL" for all three software data processing
20 Software <- "XCMS"
21 massSDrange <- 2
22 ppmCut <- 20
23 smooth <- 0
24 #####

```

Figure 5. Example of parameters set up in Paramounter part 2.

- 7) Run Paramounter part 2 by clicking on “Source” in the top right corner of the document window of RStudio. Paramounter part 2 generates two folders. Each folder contains one .png file and one .pdf file (**Figure 6**).

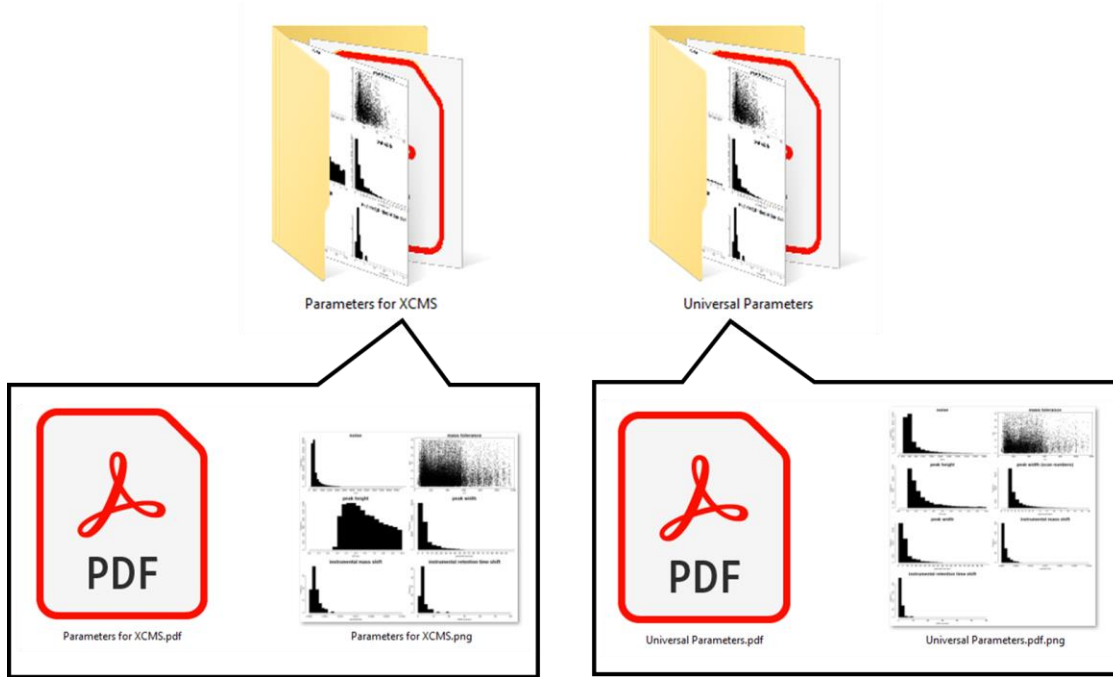


Figure 6. The output of Paramounter part 2.

- 8) Open the .pdf file and export the recommended values into the corresponding data processing software (e.g., XCMS.R in **Figure 7**). Open the complementary .png file to visualize the distribution of each parameter for the raw data.

	Parameters	Value	
1	ppm	8.000	
2	minimum peakwidth	4.000	
3	maximum peakwidth	11.000	
4	signal/noise threshold	1.071	
5	mzdiff	-0.010	
6	Integration method	2.000	
7	prefilter peaks	3.000	
8	prefilter intensity	3551.000	
9	noise filter	3551.000	
10	bw	5.000	
11	minfrac	0.500	
12	mzwid	0.005	
13	minsamp	1.000	
14	max	100.000	

XCMS.R

```
#####
directory <- "F:/Jian_Guo/OptimusDOE_20210715/test/cycle3"
#####
library(xcms)
library(MSNbase)
library(dplyr)
setwd(directory)
rawfile <- list.files(pattern = ".mzXML")
data <- readMSData(rawfile, mode = "onDisk")
cwp <- CentWaveParam(ppm=8,
                     peakwidth=c(4,11),
                     snthresh = 1.071,
                     mzdiff = -0.01,
                     integrate = 2,
                     prefilter = c(3,3551),
                     noise = 3551)
data_filtered <- filterMSLevel(data, param = cwp,
                               dataFilterOnPeaks = FALSE, msLevel = 1L)
xset <- as(data_filtered, 'xcmsSet')

#ALIGNMENT
xset@peaks <- xset@peaks[order(xset@peaks[,11]),]
table <- as.data.frame(xset@peaks)
for(n in 1:length(rawfile)){
  sampleoutput <- table[table$sample == n, ]
  sampleoutput <- sampleoutput[order(sampleoutput[,1]),]
  colnames(sampleoutput)[9] <- "intMax"
  row.names(sampleoutput) <- 1:nrow(sampleoutput)
  write.csv(sampleoutput, file = paste(n,"featuretable.csv",sep = "_"))
}

xset <- group(xset, bw = 5, minfrac = 0.5, mzwid = 0.005, minsamp = 1, max = 100)
xset <- retcor(xset, method = "obiwarp", profStep = 1)
xset <- group(xset, bw = 5, minfrac = 0.5, mzwid = 0.005, minsamp = 1, max = 100)
xset <- callPeaks(xset)
xcmt <- data.frame(xset@groups)
xcmt <- groupval(xset, value = "maxo")
featureTable <- cbind(XCMTsmzmed, XCMTsrtmed, XCMTsrtmin, XCMTsrtmax, xcmt)
colnames(featureTable)[1:4] <- c("mz", "rt", "rtmin", "rtmax")
featureTable <- featureTable[order(featureTable[,1]),]
featureTable <- as.data.frame(featureTable)
#Output
write.csv(featureTable, file = "1.csv", row.names = FALSE)
```

Figure 7. Where to copy the recommended parameters into XCMS.R.