```
install.packages("tidyverse")
library(tidyverse)

#Read file as matrix
setwd("~/RStudioSessions")
proteingroups <- as.matrix(
  read.delim2("proteinGroups[5860].txt",head=TRUE,sep="\t"))

grep("HUMAN",proteingroups[,8]) %>% length()
proteingroups[,8] %>% length()

proteingroups_df <- as.data.frame(proteingroups)
metadata <- select(proteingroups_df,1,2,6,7,8,9,12,214,626,629,630)

#df <- as.data.frame(proteingroups)
#df1 <- df[!df$Potential.contaminant == "+"]
#Select LFQ intensities
proteingroups_LFQ <- proteingroups[,492:558]

#Making data numerical
proteingroups_LFQ_num <- matrix(
  as.numeric(proteingroups_LFQ),ncol = ncol(proteingroups_LFQ))
colnames(proteingroups_LFQ_num)<- colnames(proteingroups_LFQ)
rownames(proteingroups_LFQ_num) <- proteingroups[,1]


#make a list of normalization ratio using an imported function
QC_normalise <- function(x) {
  x <- 1+(x/10)
  y <- ((0.008*x^2)-0.0427*x+1.0101)
  return(1/y)
}

#Proteingroup file but just patient intensities (removing QC columns)
proteingroups_LFQ_num_patients <- proteingroups_LFQ_num[,-(2:8)]

#normalising every patient/column
norm_patients_matrix <-
as.matrix(proteingroups_LFQ_num_patients[,1]*QC_normalise(1))
for (i in seq(2,60)){
  norm_patients_matrix <- cbind(norm_patients_matrix,
                                proteingroups_LFQ_num_patients[,i]
                                *QC_normalise(i))
}

#Modifying colnames to be just EXT numbers
colnames(norm_patients_matrix) <- sub("LFQ.intensity.20210917_TTP_P_","",
                                      colnames(proteingroups_LFQ_num_patients))
colnames(norm_patients_matrix) <- sub("_..._1_.....","",
                                      colnames(norm_patients_matrix))

#write.table(norm_patients_matrix,"R_QC_Normalised_patients", sep= '\t')

#_____________________________________________________Normalization on SPIKE ADH
```

```
#Defining the row as ADH
SpikeRow <- c("P00326;P07327;P00325")
#Defining CVD column
CVD <- c("before","before","after","after","control","control",
         "before","before","after","after","control","control",
         "before","before","after","after","control","control",
         "before","before","after","after","control","control",

         "before","before","after","after","control","control",
         "before","before","after","after","control","control",

         "before","before","after","after","control","control",
         "before","before","after","after","control","control",

         "before","before","after","after","control","control",

         "before","before","after","after","control","control")

#getting ratio of deviation form average
SpikeRowMean <- mean(norm_patients_matrix[SpikeRow,])
SpikeRatio <- SpikeRowMean/norm_patients_matrix[SpikeRow,]

#Multiplying column by ratio
sp_norm_patients_matrix <- as.matrix(norm_patients_matrix[,1]*SpikeRatio[[1]])
for (i in seq(2,60)) {
  sp_norm_patients_matrix <-
    cbind(sp_norm_patients_matrix,norm_patients_matrix[,i]*SpikeRatio[[i]])
  print(SpikeRatio[[i]])
}

colnames(sp_norm_patients_matrix) <- colnames(norm_patients_matrix)

PerseusProteins <- cbind(sp_norm_patients_matrix,metadata)
write.table(PerseusProteins,"PerseusProteins.tsv", sep= '\t', row.names=F)

#transposing for XQboost
transposed_sp_norm_patients_matrix <- t(sp_norm_patients_matrix)

CVD_transposed_patients_matrix <- cbind(transposed_sp_norm_patients_matrix,CVD)

#making a Only QC normalized matrix
T_norm_patients_matrix <- t(norm_patients_matrix)
T_norm_patients_matrix_CVD <- cbind(T_norm_patients_matrix,CVD)

################################################################################

df <- CVD_transposed_patients_matrix[,c(1:429)]
df <- apply(df, 2, as.numeric)
#Replacing all values +1 for log transformation
df <- df+1

clrdat <- as.data.frame(compositions::clr((t(df))))
clrdat1 <- otu_table((clrdat),taxa_are_rows= T)
```

```
dist <- philentropy::distance(t(clrdat), method = "euclidean")
#check of de distance matrix gebaseerd is op je subjecten; niet je variabelen

CVD_transposed_patients_matrix <- as.data.frame(CVD_transposed_patients_matrix)
class(CVD_transposed_patients_matrix)
adonis2(dist ~ CVD, data = CVD_transposed_patients_matrix,
        permutations = 9999,
        method = "bray",
        na.rm = T)
```