DIA_SILAC_workflow

1) Info

R version 4.0 was used

2) Setup

The following packages will be used for this document.

```
require(ggplot2)
require(ggforce)
require(tidyr)
require(data.table)
require(limma)
require(dplyr)
require(stringr)
require(cgwtools)
require(purrr)
require(broom)
```

3) Introduction

The following script will allow allow users to analyze their SILAC serial dilution DIA data. Input into this script will be searched data post Spectronaut (other search engines may be fine, please see section 4 for data requirements) and output will be a list of peptides and their linear concentration ranges after passing certain quantitaion criteria and an abundance threshold indicating high confidence quantitation. This data workflow will be split into four sections: data management and filtering, fitting linear models for peptides, data visualization, and creating an abundance threshold for high confidence quantitative peptides.

4) Data reformatting and filtering

This section achieves two goals: 1) handling raw MS data, converting peptide abundances into ratios, and transforming the peptide and ratio data into a format that will allow us to perform linear models for the next section and 2) filtering peptides based on user-decided quantitative criteria.

The raw MS data will need the following attributes to be compatible with this code. The data herein was analyzed and exported with Spectronaut, which has all of these outputs.

1) Long format, peptide output (ex: MS Stats output in Spectronaut)
2) Signal to noise readout labeled FG.SignalToNoise
3) Raw abundance readout of MS2 peak labeled FG.MS2RawQuantity
4) A condition column labeled R.Condition that reflects the order of the serial dilution curve (1 is the lowest concentration, 15 is the highest concentration)

5) A replicate column labeled R.Replicate that reflects the technical replicate of the serial dilution sample (1,2,3 *note if doing any other number of replicates XXXXXXX lines will need to be changed)

6) A peptide modification readout that contains the heavy label that says Arg and Lys named FG.Id

7) A column that contains the peptide stripped down to its indentified sequence named PEP.StrippedSequence

8) A column with the protein ID corresponding to each peptide labeled PG.ProteinAccessions

Note: All of these parameters are flexible and can be fit to purpose depending on the user's experiment. These have been marked throughout the document with ##. The filtering criteria is outlined as follows:

1) Signal to noise > 10

2) MS2 peak area >0

3) Heavy and light pair for every peptide

4) Peptide abundance appears in at least 2/3 technical replicates per dilution

5) Peptide ratio (ratio between heavy and light) CV is <20% across technical replicates per concentration

6) Peptides must have at least 4 consecutive dilutions

```
raw <- read.csv("Desktop/Serial_Dilutions_Workflow_Data.csv")
##can adjust signal to noise and MS2RawQuantity here
df <- raw %>%
  filter(FG.SignalToNoise > 10) %>%
  filter(FG.MS2RawQuantity > 0)
index <- c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)
##this is where you will put your light:heavy ratios
values <- c("0.000", "0.0024", "0.0048", "0.0098","0.019","0.039","0.078","0.15","0.31","0.625","1.25",
df$Conc <- values[match(df$R.Condition, index)]
df$Conc_Replicate <- paste(df$Conc, df$R.Replicate,sep="_")
df$Label = if_else(str_detect(df$FG.Id, "Arg|Lys"),"H","L")
df$Peptide_Label = paste(df$PEP.StrippedSequence, df$Label,sep="_")
wdf <- data.frame(Peptide = df$Peptide_Label,Run = df$Conc_Replicate, Value = df$FG.MS2RawQuantity)
test2 <- dcast(wdf, Peptide ~ Run, fun = sum)
RESULTS <- test2 %>%
  gather('conc_run', 'value', -Peptide) %>%
  filter(value > 0) %>%
  separate(Peptide, c("Peptide", "Label"), sep = "_" ) %>%
  mutate(Peptide_Run = paste(Peptide, conc_run, sep = "_"))
Heavy_Peptides <- filter(RESULTS, Label == "H")
Light_Peptides <- filter(RESULTS, Label == "L")
Heavy_Light_join <- inner_join(Heavy_Peptides, Light_Peptides, by = "Peptide_Run")
##change filter(is_na < 3) to 0 if okay with only 1 observation in the technical replicates
Present_in_2 <- Heavy_Light_join %>%
  mutate(ratio = (value.y/value.x)) %>%
  dplyr::select('conc_run.x','Peptide_Run','ratio',Peptide.x) %>%
  separate(Peptide_Run, c("Peptide_conc", "Run"), sep = "_(?=[^_]+$)" ) %>%
  dplyr::select(-conc_run.x) %>%
  spread(Run,ratio) %>%
  mutate(is_na = rowSums(is.na(.))) %>%
  filter(is_na < 2)
mean_intensities <- apply(Present_in_2[3:5], 1, mean, na.rm = T)
```

```
sd_intensities <- apply(Present_in_2[3:5], 1, sd, na.rm = T)
Present_in_2$CV <- sd_intensities/mean_intensities
##filter CV to desired thresholed, 0.2 = 20% CV
Low_CV <- Present_in_2 %>%
  filter(CV < 0.2) %>%
  dplyr::select(Peptide_conc, "1", "2", "3") %>%
  separate(Peptide_conc, c("Peptide", "conc"), sep = "_" )
consecutive_test <- Low_CV
sorted.out <- consecutive_test[order(as.numeric(as.character(consecutive_test$conc))), ]
consec2 <- sorted.out %>%
  group_by(Peptide) %>%
  arrange(.by_group = TRUE)
consec2$conc <- as.numeric(consec2$conc)
consec3 <- consec2 %>%
  group_by(Peptide) %>%
  mutate(Diff = as.integer(conc / lag(conc)))
consec4 <- consec3 %>%
  mutate(Diff=replace(Diff, Diff==1, 2))
consec <- consec4
consec$Diff <- replace_na(consec$Diff, 2)
consec$n <- ifelse(consec$Diff == 2, TRUE, FALSE)
##change filter(consecutive >= 4) if desiring different # of consecutive peptides
consecutive_only <- consec %>%
  group_by(Peptide) %>%
  mutate(count = ave(n, cumsum(!n), FUN = cumsum)) %>%
  group_by(Peptide, consecutive = rep(seqle(count)$length, seqle(count)$length)) %>%
  filter(consecutive >= 4) %>%
  ungroup() %>%
  dplyr::select(-consecutive)
df_trimmed_non_nest <- consecutive_only %>%
  gather(Replicate, Ratio, 3:5) %>%
  dplyr::select(Peptide, conc, Replicate, Ratio)
head(df_trimmed_non_nest)
```

```
## # A tibble: 6 x 4
##   Peptide            conc Replicate  Ratio
##   <chr>             <dbl> <chr>      <dbl>
## 1 AAAAAWEEPSSGNGTAR 0.039 1         0.0801
## 2 AAAAAWEEPSSGNGTAR 0.078 1         0.141
## 3 AAAAAWEEPSSGNGTAR 0.15  1         0.273
## 4 AAAAAWEEPSSGNGTAR 0.31  1         0.467
## 5 AAAAAWEEPSSGNGTAR 0.625 1         0.860
## 6 AAAAAWEEPSSGNGTAR 1.25  1         1.53
```

5) Linear modeling

Peptide abundance should behave linearly in accordance to the concentration loaded on the mass spectrometer. Here, a linear model is fit to each peptide that meets the above criteria. To determine peptides with linear behavior, peptides are subsequently filtered to include only those with an R^2 greater than 0.9. With dilution curves, non-linear behavior is observed at the saturation point (ULOQ) or when the peptide intensity is low and readouts are inaccurate (LLOQ). These points must be eliminated to fit the most accurate curve to the dilution schema. To achieve this, a linear curve was fit to every 3 dilutions in a peptide's dilution curve and the subsequent dilution factors were kept if the slope for this linear fit was >0.8 and <1.2. This

method is most sensitive to testing linearity at the bottom point of the curve. Three points at a time were chosen to assess slope to prevent the scenario where lower dilution factoris (higher intensities) are driving the observed linearity).

Input is a dataset which was created from the last section. Output is a list of filtered peptides with their linear model results, their dilution data, and the number of peptides and proteins that are in this final list.

```
fit_model <- function(df) lm(conc ~ Ratio, data = df)
get_rsq <- function(mod) glance(mod)$r.squared
get_slope <- function(mod) tidy(mod)$estimate[2]
get_p_value <- function(mod) tidy(mod)$p.value[2]
df_trimmed_nest <- df_trimmed_non_nest %>%
  group_by(Peptide) %>%
  nest() %>%
  mutate(model = map(data, fit_model)) %>%
  mutate(r.squared = map_dbl(model, get_rsq)) %>%
  mutate(slope = map_dbl(model, get_slope)) %>%
  mutate(p_value = map_dbl(model, get_p_value))

##change 0.9 if desiring a different r^2 cutoff
df_rsquared3 <- df_trimmed_nest[ which(df_trimmed_nest$r.squared > 0.9), ]
high_rsquared_peptides3 <- subset(df_trimmed_non_nest, Peptide %in% df_rsquared3$Peptide)


##change filter(m>0.8 & m <= 1.2) if desiring a different slope criteria for the every 3 point ##check
ef <- df_rsquared3
output <-lapply(1:nrow(ef), function(x){
  pep <- ef$Peptide[x]
  dt <- ef$data[[x]]
  dt2 <- dt%>%
    group_by(conc)%>%
    transmute(Ratio_mean= mean(Ratio, na.rm = T))%>%
    unique()
  d <- do.call('rbind',lapply(1:(nrow(dt2)-2), function(i){
    fit <- lm(dt2[i:(i+2),]$conc~dt2[i:(i+2),]$Ratio_mean)
    s <- summary(fit)
    data.frame(segment = paste(i:(i+2), collapse = '_'),
               m = as.numeric(s$coefficients[2,1]),
               low_conc = min(dt2$conc[i:(i+2)]))
  }))
  #d$Peptide <- pep
  d%>%
    filter(m>0.8 & m <= 1.2)%>%
    select(low_conc)%>%unlist()%>%unname() ->vals
  dt%>%
    filter(conc >= min(vals))
})
ef$new_data <- output

final_peptide_list <- ef %>%
  select(Peptide, new_data) %>%
  unnest() %>%
  nest()
```

6) Visualizaton

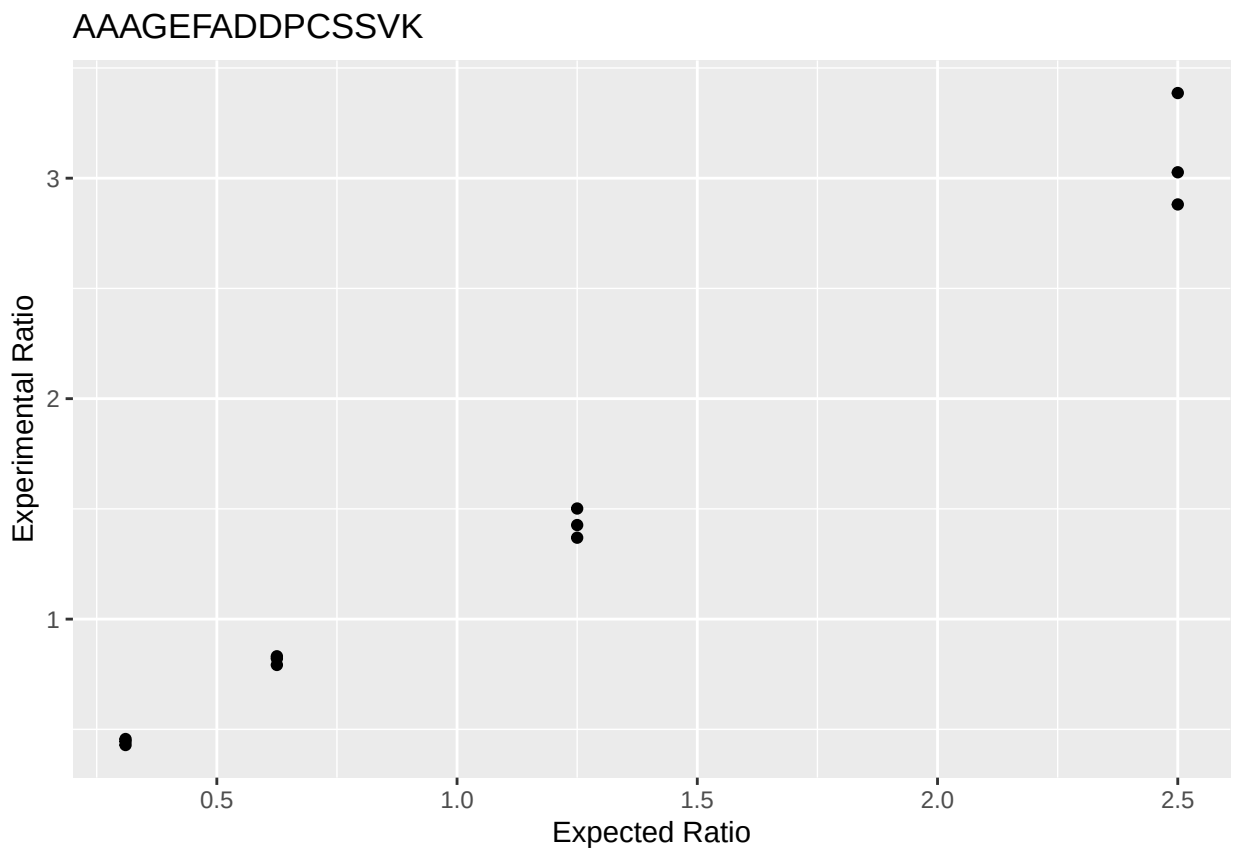
Next will be visualization of the acquired data.

The plot below graphs the dilution curve for each peptide.

```
plots <-
  final_peptide_list %>%
  mutate(plot = map2(data, Peptide, ~ ggplot(data = .x, aes(x = conc, y = Ratio)) +
                       geom_point() + ggtitle(.y) + xlab("Expected Ratio") + ylab("Experimental Ratio"))

##select specific peptide to plot by its numerical order in the final_peptide_list. This selects ##the
print(plots$plot[3])
```

```
## [[1]]
```



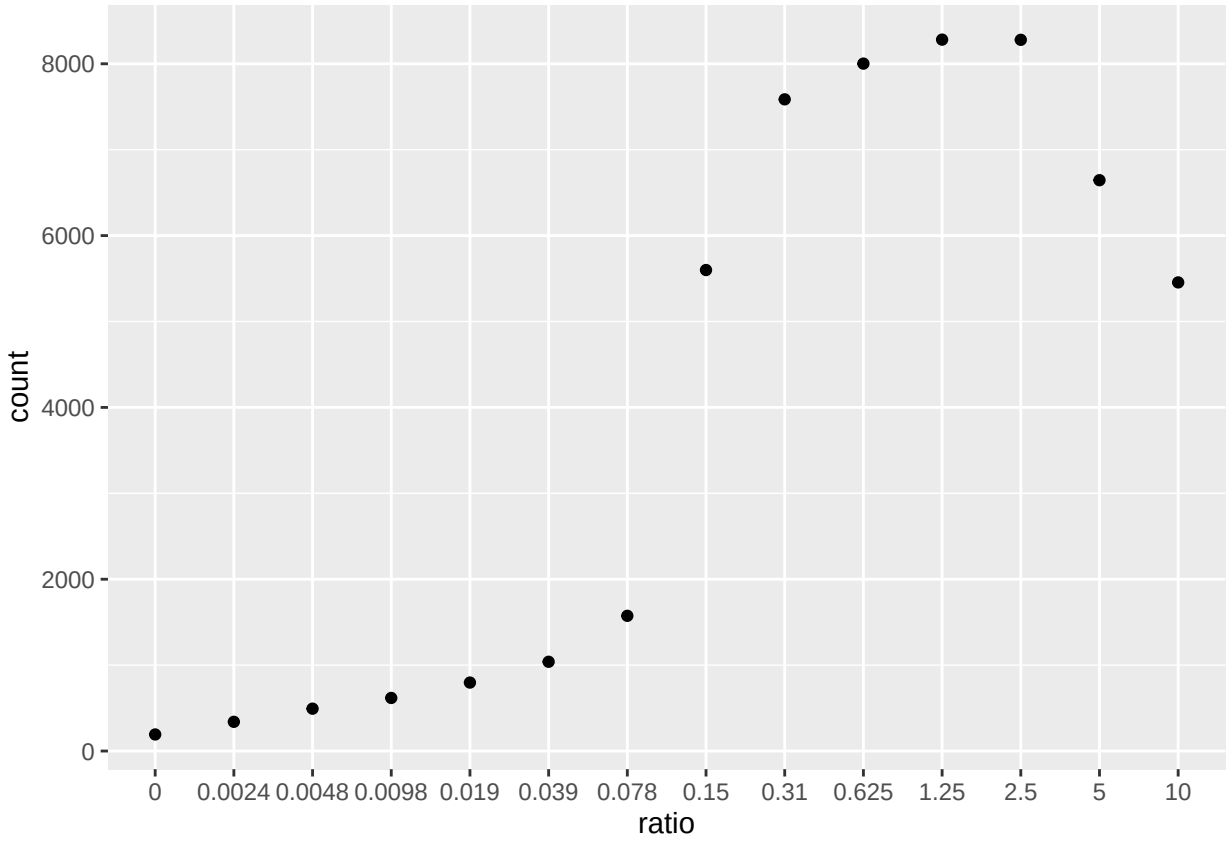

```
ef_ungrouped <- ef %>%
  select(Peptide, new_data) %>%
  unnest()

concentration_spread <- ef_ungrouped %>%
  spread(conc,Ratio)

concentration_gather <- concentration_spread %>%
  gather(concentration, Ratio, 3:dim(concentration_spread)[2])
```

Below, the most common ratios that were included in the data are visualized. This is likely the dilution range that peptides are most quantitatively accurate.

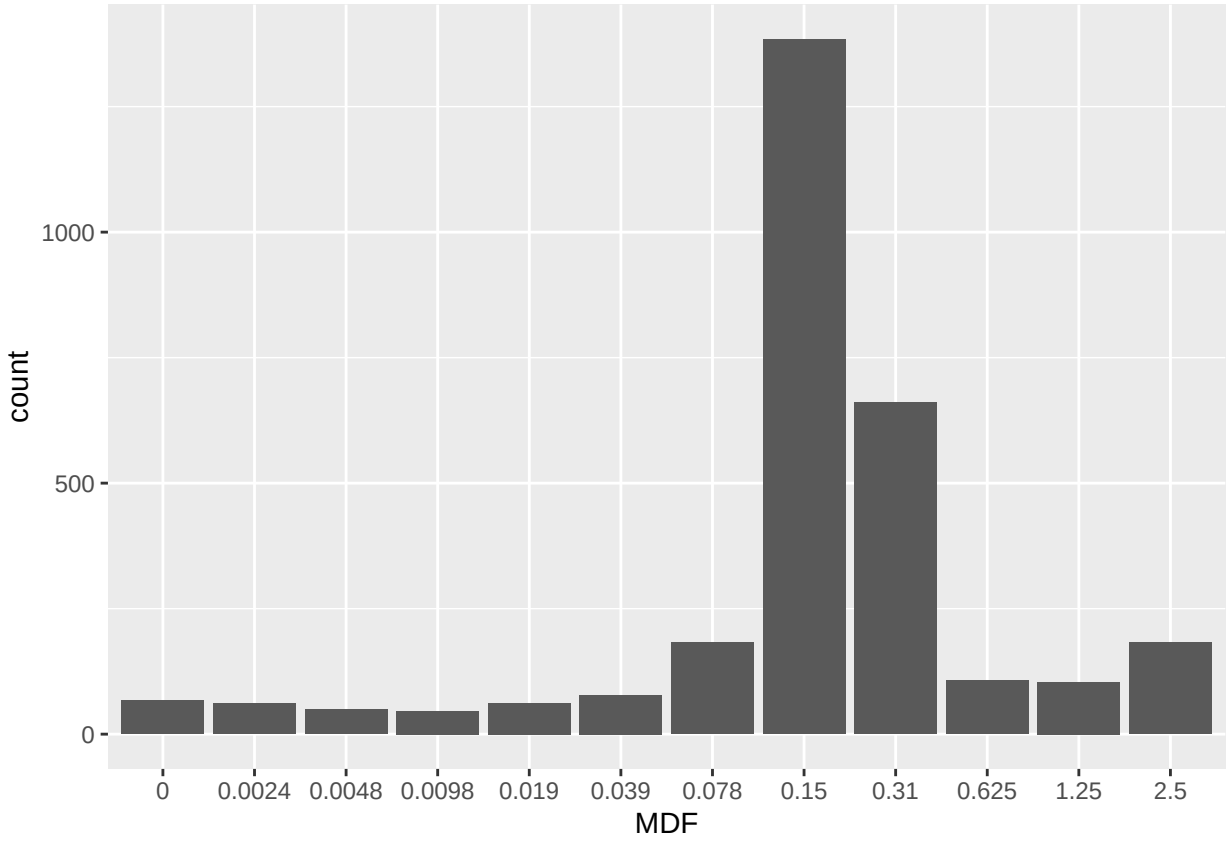
```
yes_concentration_value <- colSums(!is.na(concentration_spread[,3:dim(concentration_spread)[2]]))
melt_conc_value <- reshape2::melt(yes_concentration_value)
melt_conc_value$conc <- rownames(melt_conc_value)
concentration_presence_yes <- ggplot(melt_conc_value, aes(x = reorder(conc, sort(as.numeric(conc))), y=
concentration_presence_yes
```



Next, the most common minimal dilution factors (MDF) in the data are visualized. A peptide MDF is the lowest dilution factor for which a peptide behaves linearly on the dilution curve. While the exact concentration of the peptide is unknown, their respective dilution factors are known. In this dataset, a ratio of heavy to light of .15 (0.015 light on the serial dilution curve, sample #8) tends to be one of the most often observed MDF within a peptide's linear dilution range.

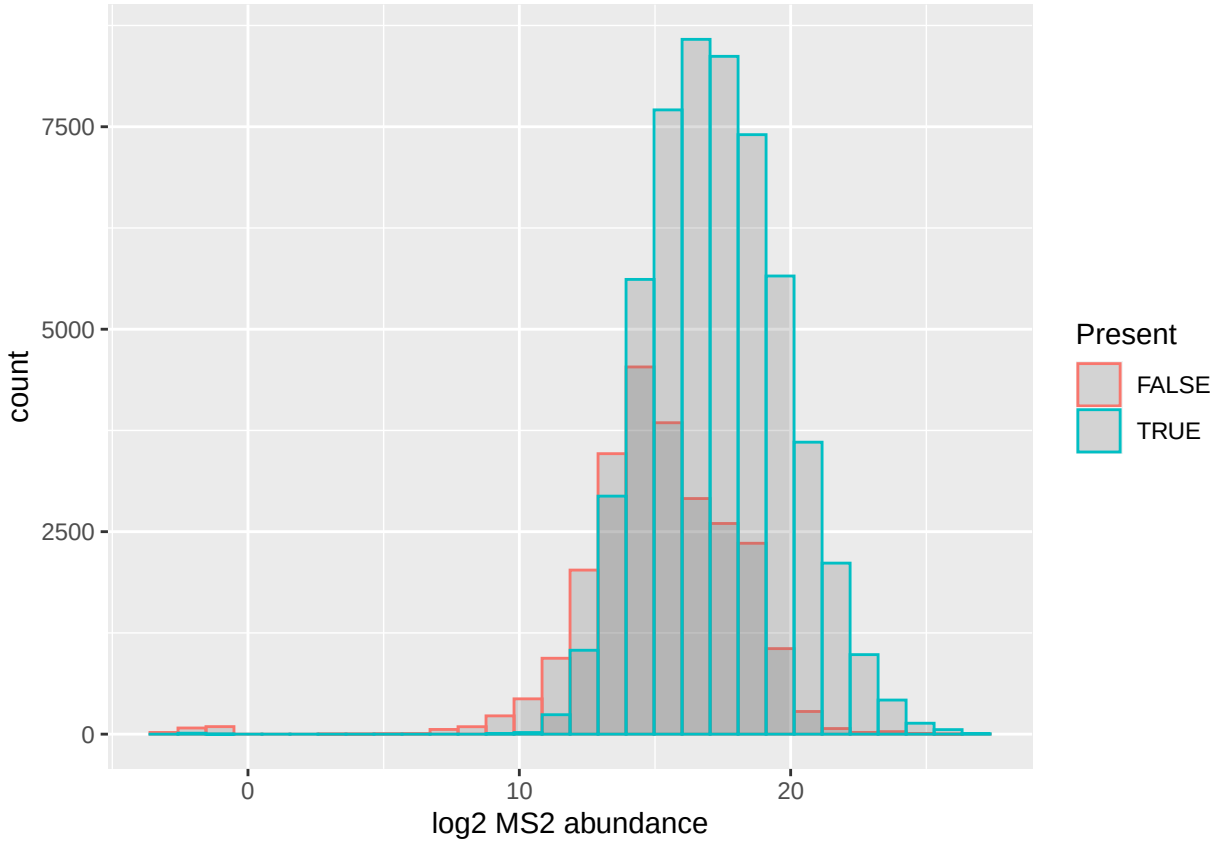
```
MDF <- ef_ungrouped %>%
  group_by(Peptide) %>%
  summarise(conc_min = min(conc))

peptide_MDF_bar <- ggplot(MDF, aes(x=as.character(conc_min))) + geom_bar() + labs(x="MDF")
peptide_MDF_bar
```



Finally, all the peptides from the final list and all of their abundances at every dilution are considered. Dilutions that demonstrated linear behavior and were included after all filtering criteria are labeled quantitative or "TRUE" abundances and concentrations that were excluded due to not meeting some of the filtering criteria are labeled as qualitative or "FALSE" abundances. Note- MS2 abundance is now being plotted, no longer the ratio.

```
trim_Light <- Light_Peptides %>%
  select(Peptide_Run, value)
concentration_present <- concentration_gather %>%
  mutate(Present = !is.na(Ratio))
  ##mutate(Present=replace(Present,Present == TRUE, "Quantitative")) %>%
  ##mutate(Present=replace(Present,Present == FALSE, "Qualitative"))
options(scipen=999)
concentration_present$concentration <- as.character(concentration_present$concentration)
concentration_present <- concentration_present %>%
  mutate(concentration=replace(concentration, concentration==5, "5.0")) %>%
  mutate(concentration=replace(concentration, concentration==0, "0.000"))
abundance_MDF_table <- concentration_present %>%
  unite(Peptide_Run, c(Peptide, concentration, Replicate), sep="_" )
abundance_MDF_table2 <- left_join(abundance_MDF_table, trim_Light, by = "Peptide_Run")
peptide_abundance_plot_histogram <- ggplot(abundance_MDF_table2, aes(x=log2(value), color = Present)) +
peptide_abundance_plot_histogram
```



7) Abundance threshold for % peptides that behave linearly

Lastly, the point on the above graph in which at least X% of peptides are quantitative over qualitative can be established. This cutoff point can be useful for future experiments in the same matrix, as the user can use the thresholding cutoff without the need to run the above filtering criteria.

```
df2 <- abundance_MDF_table2

df2 <- as.data.table(df2)
df2 <- df2[!is.na(value)]
df2 <- df2[order(-value)]
##cutoff can be changes, 0.8 = 80% quantitative, 0.9 = 90% quantitative
cutoff = 0.8
n = 0
f = 0
for(i in seq_len(nrow(df2))){
  p_val <- df2[i][,Present]
  n = n+1
  if(!p_val){
    f = f+1
    q = (n-f)/n
    if(q <= cutoff){
      message("Val: ", paste(format(q*100,4),"%"))
      message("Cutoff reached, Abundance value: ", df2[i][, value],"\n")
      message('Row number: ',i," || ",paste(df2[i], collapse = " , "))
```

```
      break
    }
  }
}
```

```
## Val: 80 %
```

```
## Cutoff reached, Abundance value: 65611.25
```

```
## Row number: 46595 || IPGGIIEDSCVLR_0.078_1 , NA , FALSE , 65611.25
```