Project 5 - Proteomics - Analytical Methods

Veedhi Solanki

December 6, 2023

GitHub - https://github.com/Veedhi-Solanki/Assignment-5.git

Introduction:

Proteomics, the systematic study of proteins and their functions, is at the forefront of identifying the complex molecular landscapes that are present within biological systems (Nikolov et al., 2012). This project will explore quantitative proteomics, with a particular emphasis on determining the impact of various analytical methodologies. Understanding the quantitative proteome is critical for increasing understanding of many biological processes and has potential for applications in disease diagnosis, prognosis, and personalized therapy (Noble et al., 2012). Beyond the scientific arena, quantitative proteomics' societal importance arises as it contributes to healthcare and other advances, stressing its role in influencing the future of biological research and medicinal applications. The dataset under study comes from the PRIDE database (PXD000001), specifically the TMT 6-plex experiment. Exogenous proteins (ENO, BSA, PHO, CYT) are introduced into an *Erwinia carotovora* lysate in this experiment, providing a rich source for investigating the different aspects of quantitative protein expression.

While much is known about gene expression, one critical factor remains unknown: how statistical decisions affect protein studies (Noble et al., 2012). Considering this a new area in which the aim is to discover the best analytical methodologies and parameters that can be used in some of the commonly used visualizations and statistical analysis for examining protein behavior. Proteins, with their particular challenges, need a more complex approach than gene research. The aim here involves delving into exploring various methods for ex. filtering or choosing the method for calculating peptide intensities, determining the effect of analytical decisions on protein discoveries, and contributing to the developing understanding of protein research.

Inspiration for this project was drawn from the "Using R and Bioconductor for proteomics data analysis" paper (Gatto et al., 2014). This paper provides an overview of applications of R software in the analysis of MS-based quantitative proteomics data, including compliance with open proteomics data standards, input/output capabilities, quantitation pipelines, quality control, quantitative data analysis, and relevant annotation infrastructure. The R for proteomics vignette was explored fully first where I particularly got interested in quantitative proteomics. So, in this project the quantitative part of the vignette is explored. It has data files in two formats, first mztab file is analyzed through and some parameters here are explored such as for protein intensity plot, where the sum method was used to get protein intensity which was changed to mean method to see how results change. Furthermore, from the same file a heat map was also created where the normalization method was explored by changing sum and vsn(Variance Stabilizing Normalization) to median and quantile. Next, a normalized intensity spike plot was created using the same file where results were compared using the data from different sum vs median. Finally, the main dataset was also used to generate MAplot. So, this is how different analytical methods were changed for various types of analysis on data from mzTAB and mzxml files for proteins spiked in *Erwinia carotovora* lysate for TMT 6-plex experiment.

Description of Data Set:

The dataset used in the study is a TMT 6-plex experiment where four exogenous proteins were spiked into an equimolar *Erwinia carotovora* lysate with varying proportions in each channel of quantitation(Gatto et al., 2014). The proteins that were spiked in and their respective ratios in the experiment were yeast enolase (ENO) at 10:5:2.5:1:2.5:10, bovine serum albumin (BSA) at 1:2.5:5:10:5:1, rabbit glycogen phosphorylase (PHO) at 2:2:2:2:1:1 and bovine cytochrome C (CYT) at 1:1:1:1:1:2 (Gatto et al., 2014). The proteins were digested, differentially labeled with TMT reagents, fractionated by reverse phase nanoflow UPLC, and analyzed on an LTQ Orbitrap Velos mass spectrometer (Gatto et al., 2014). Regarding the specific formats, mzTab is a tab-delimited file format that represents peptide and protein identifications together with the evidence supporting these identifications. On the other hand, mzXML is an open, generic XML format for mass spectrometer output data. It is used to store MS data, including raw data and metadata. Both of these files were used in this project to analyze and visualize data as well as to explore various analytical methods.

Code Section 1 – Data Acquisition, Exploration, Filtering, and Quality Control (Only briefe code here, more detailed can be found in RMarkdown file):

Code is drawn from link to vignette but analytical methodologies were explored: https://bioconductor.org/packages/release/data/experiment/vignettes/RforProteomics/inst/doc/RforProteomics.html#7_Annotation

In this project, data were acquired from pride using the code below:
Mztab data:
Getting the datafile from 11 files that were present:

```
library("rpx")
px1 <- PXDataset("PXD000001")
px1 # 11 files in total
mzxml <- pxget(px1, "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01.mzXML")
pxfiles(px1) # lists all the files
### Getting mztab data from px1.
mztab <- pxget(px1, "F063721.dat-mztab.txt")
# loading mztab peptide data
qnt <- readMzTabData(mztab, what = "PEP", version = "0.9")
sampleNames(qnt) <- reporterNames(TMT6)
```

After obtaining mztab data, main part was to make sure there were no missing values as a quality control,so, NA values were removed using the code below:

```
# removing missing values - Quality control
qnt <- filterNA(qnt)
processingData(qnt)
```

Moreover, after this, I wanted to check on before and after missing values so, bar plot was created from that using the sample code below which is for NA values before:

```
# Count missing values before removal
missing_before <- colSums(is.na(qnt))
```

```
# Bar plot for missing values before removal
barplot(missing_before, col = "red", main = "Missing Values Before Removal",
      xlab = "Variables", ylab = "Count")
```

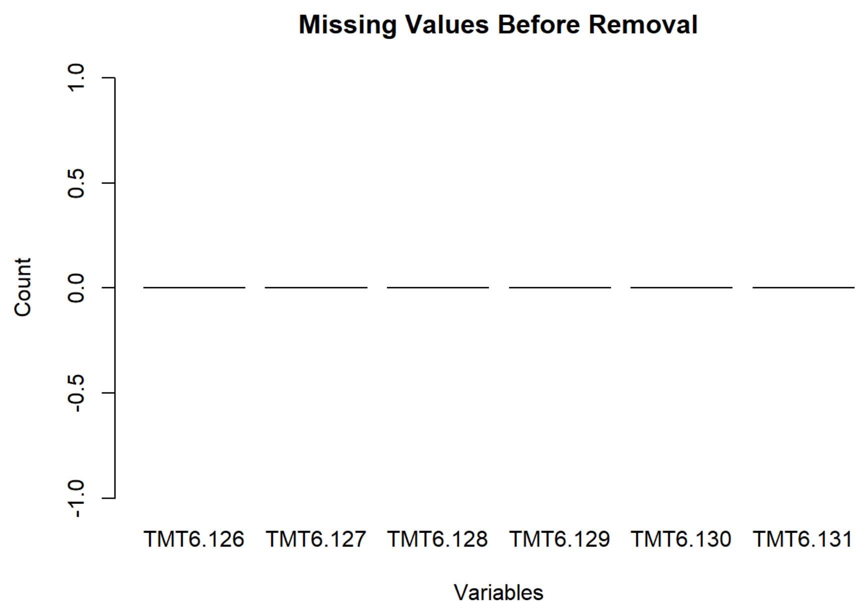


Figure 1: It shows there were no missing values in the dataset from the beginning even before filtering data by NA.

Mzxml data:

Getting the file of the 11 files on pride:

```
mzxml <- pxget(px1, "TMT_Erwinia_1uLSike_Top10HCD_isol2_45stepped_60min_01.mzXML")
rawms <- readMSData(mzxml, centroided = TRUE, verbose = FALSE)
```

Overall, both datasets were normalized, imputation and quantified as needed. Analytical method changes were also explored. Some sample code for some of these functions is as below:

```
qntS <- normalise(qnt, "sum")
qntms_combined <- quantify(rawms,
                    reporters = TMT7,
                    method = "Sum",
                    filterCriteria = 0.7,
                    imputation = "knn")
```

Moreover, accession numbers were used to combine peptides for protein intensities, below is a sample code for that in mztab:

```
protqnt <- combineFeatures(qnt,
                    groupBy = fData(qnt)$accession,
                    method = sum)
```

Analytic method changes and parameter changes in the quality control code were explored and it will be looked at in more detail in the main code section and discussed further in the results and discussion.

Then I also created a plot to look at overall data from the mzxml file data visualization from the code below:

```
d <- data.frame(Signal = rowSums(exprs(qntms)[, 1:6]),
            Incomplete = exprs(qntms)[, 7])
d <- log(d)
pch <- rep(1, nrow(qnt))
cls <- rep("#00000050", nrow(qnt))
cls[grep("P02769", fData(qnt)$accession)] <- "gold4" # BSA
cls[grep("P00924", fData(qnt)$accession)] <- "dodgerblue" # ENO
cls[grep("P62894", fData(qnt)$accession)] <- "springgreen4" # CYT
cls[grep("P00489", fData(qnt)$accession)] <- "darkorchid2" # PHO
pch[grep("P02769", fData(qnt)$accession)] <- 19
pch[grep("P00924", fData(qnt)$accession)] <- 19
pch[grep("P62894", fData(qnt)$accession)] <- 19
pch[grep("P00489", fData(qnt)$accession)] <- 19

# 3. Figure 7: MAplot on an MSnSet instance
MAplot(qnt[, c(4, 2)], cex = .9, col = cls, pch = pch, show.statistics = FALSE)
```

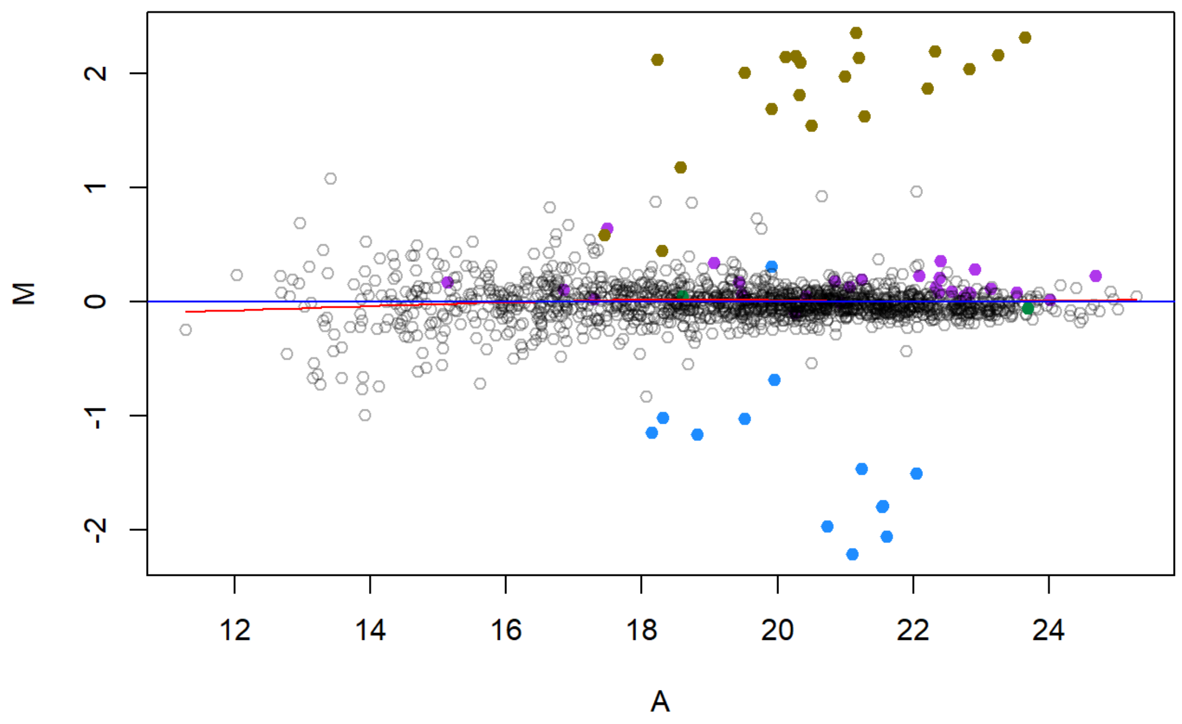


Figure 2: MAplot - the points are individual proteins where the colour scheme is gold4-BSA, dodgerblue-ENO, springgreen4-CUT and darkorchid2-PHO. BSA and ENO are differentially expressed whilePHO and CYT are not.

Main Software Tools Description:
I chose to highlight the core packages MSnbase and RforProteomics as they are fundamental to the infrastructure and reproducibility of quantitative proteomics analysis within the R environment, as outlined in the paper by (Gatto et al., 2014).

- **MSnbase:** This R/Bioconductor package is specifically tailored for the analysis of mass spectrometry-based quantitative proteomics data. It provides a comprehensive infrastructure for handling and processing MS data, with a strong emphasis on quantitative analysis. MSnbase offers functionalities for importing raw data, preprocessing, normalization, and quality control. Additionally, it facilitates visualization of quantitative proteomics data, enabling researchers to gain insights into the abundance and dynamics of proteins across different experimental conditions. The package's capabilities make it an essential tool for researchers engaged in quantitative proteomics studies, providing a robust framework for data analysis and interpretation.

- **RforProteomics:** The RforProteomics package serves as a companion to the review, offering a curated selection of tools for the analysis of MS-based quantitative proteomics data. It is designed to provide researchers with the necessary code to install and utilize relevant software tools, ensuring reproducibility and adaptability of the examples presented in the review. By offering a collection of tools within the R environment, RforProteomics aims to promote sound and reproducible data analysis in the field of quantitative proteomics. The package's focus on facilitating reproducibility underscores its importance in ensuring the reliability and transparency of quantitative proteomics research.

While these packages are tailored for quantitative proteomics analysis, they have limitations in addressing specialized or niche requirements within the field. I built upon existing vignettes provided by the RforProteomics package, adapting and extending them to address the specific analytical decisions and methodologies under investigation. There are different types of methods available such as sum, max, quantile, median and many more, so, I chose to explore them in terms of protein intensity, imputation, normalization and filtering through different values.

Code Section 2 (Only main code is discussed here but more detailed is in the RMarkdown) – Main Analysis:

Code is drawn from link to vignette but analytical methodologies were explored: https://bioconductor.org/packages/release/data/experiment/vignettes/RforProteomics/inst/doc/RforProteomics.html#7_Annotation

In this section all the main code and analytical methods that were changed will be explored with the figure comparisons.

1. Analytical Method exploration 1: Sum vs Mean for calculating protein intensity

Here, combining the peptides into proteins by grouping through the accession number in the metadata. This code below also calculates protein intensities by summing the peptide intensity data. As it can be noticed, the method of calculating protein intensity here is sum.

```
protqnt <- combineFeatures(qnt,
                    groupBy = fData(qnt)$accession,
                    method = sum)
```

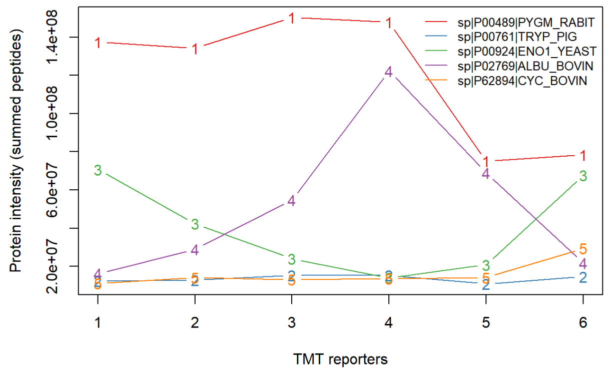
I was interested in changing the way protein intensity from peptides were calculated so the method was changed to mean which means that the mean of the same peptides will be used to get the protein intensities.

```
protqnt_m <- combineFeatures(qnt,
                    groupBy = fData(qnt)$accession,
                    method = mean)
```

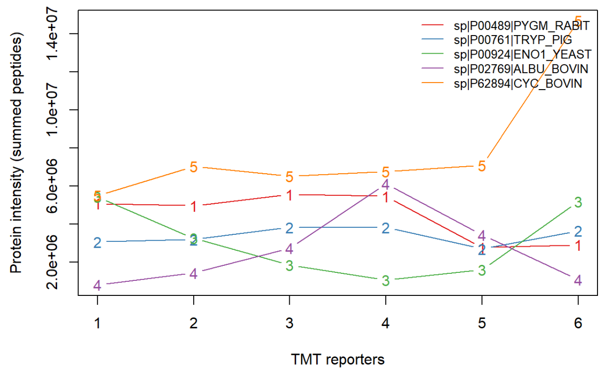
Now, visualizing the intensities of 5 proteins for both datasets protqnt where method was sum and protqnt_m where method was mean. This will produce a plot on the protein quantitation data.

```
#protqnt_SUM
cls <- brewer.pal(5, "Set1")
matplot(t(tail(exprs(protqnt), n = 5)), type = "b",
        lty = 1, col = cls,
        ylab = "Protein intensity (summed peptides)",
        xlab = "TMT reporters")
legend("topright", tail(featureNames(protqnt), n=5),
       lty = 1, bty = "n", cex = .8, col = cls)

#protqnt_MEAN
cls <- brewer.pal(5, "Set1")
matplot(t(tail(exprs(protqnt_m), n = 5)), type = "b",
        lty = 1, col = cls,
        ylab = "Protein intensity (summed peptides)",
        xlab = "TMT reporters")
legend("topright", tail(featureNames(protqnt_m), n=5),
       lty = 1, bty = "n", cex = .8, col = cls)
```



A)



B)

Figure 3: This figure shows the protein intensities of the five proteins that were spiked in the lysate. The method for calculating protein intensity from peptide intensities for **A)** was Sum and

B) was Mean. The differences are significant. Intensity of CYC_BOVIN and PYGM_RABIT has been significantly affected by changing the protein intensity method from sum to mean.

1. Analytical Method exploration 2: Normalisation (sum and vsn) vs (median and quintiles) for Heat Map

First, Normalize the quantitation data by sum then by vsn and plot heat map 1.

```
qntS <- normalise(qnt, "sum")
# Further normalize the data using vsn (variance stabilizing normalization)
qntV <- normalise(qntS, "vsn")
qntV2 <- normalise(qnt, "vsn")
# Define a list of protein accessions
acc <- c("P00489", "P00924",
      "P02769", "P62894",
      "ECA")
# Find indices of proteins in the dataset based on the accessions
idx <- sapply(acc, grep, fData(qnt)$accession)
idx2 <- sapply(idx, head, 3)
# Extract a subset of data (small) containing the first 3 peptides of each protein
small <- qntS[unlist(idx2), ]
# Further subset the data (medium) by taking the first 10 peptides of each protein
idx3 <- sapply(idx, head, 10)
medium <- qntV[unlist(idx3), ]
# Extract expression values from the medium dataset
m <- exprs(medium)
# Define column and row names
colnames(m) <- c("126", "127", "128",
           "129", "130", "131")
rownames(m) <- fData(medium)$accession
rownames(m)[grep("CYC", rownames(m))] <- "CYT"
rownames(m)[grep("ENO", rownames(m))] <- "ENO"
rownames(m)[grep("ALB", rownames(m))] <- "BSA"
rownames(m)[grep("PYGM", rownames(m))] <- "PHO"
rownames(m)[grep("ECA", rownames(m))] <- "Background"
# Define colors for the heatmap
cls <- c(brewer.pal(length(unique(rownames(m)))-1, "Set1"),
      "grey")
names(cls) <- unique(rownames(m))
# Define a color ramp for the heatmap
wbcol <- colorRampPalette(c("white", "darkblue"))(256)
# Create a heatmap with specified colors and row side colors
heatmap(m, col = wbcol, RowSideColors=cls[rownames(m)])
```

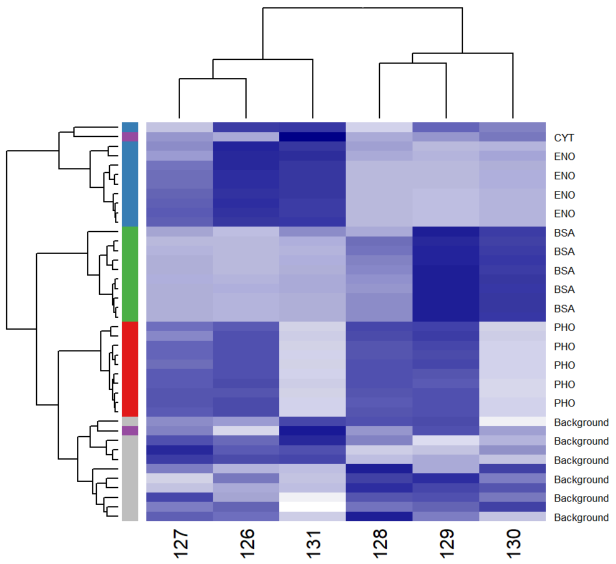
Next, new normalization methods to quintiles and median:

```
qntQ <- normalise(qnt, "quantiles")
```

```
# Normalize the quantitation data using median (replacing "sum" with "median")
qntM <- normalise(qntQ, "center.median")
qntM <- normalise(qnt, "center.median")
qntQ <- normalise(qntM, "quantiles")
small_ <- qntM[unlist(idx2), ]
medium_ <- qntQ[unlist(idx3), ]
m_ <- exprs(medium_)
```

The code for the heatmap 2 is the same as previously shown for the same data that were normalized differently.

A)



B)

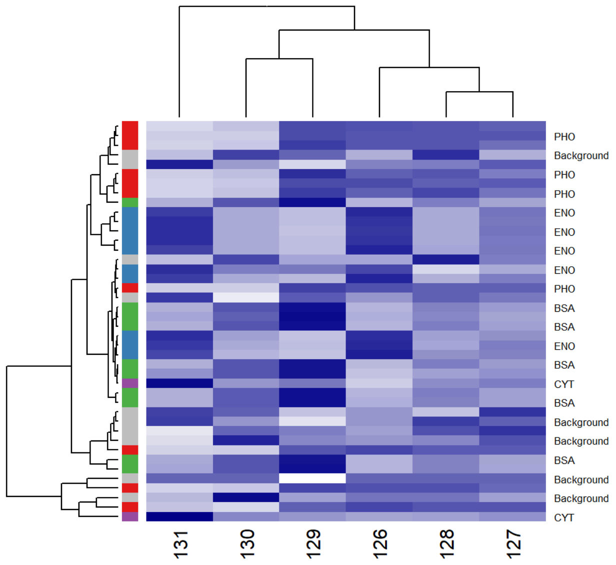


Figure 4: Heat maps generated using different normalization approach were the **A)** uses sum and vsn while **B)** uses median and quantile. There are significant differences between both heat maps which is very obvious by noticing color patterns for particular proteins. Here different proteins are measured including BSA(Bovine Serum Albumin), PHO(Phosphorylase), ENO(Enolase), CYT(Cytochrome c) and the background protein in cell for TMT tags 127 - 131. Moreover, different clustering patterns can be noticed.

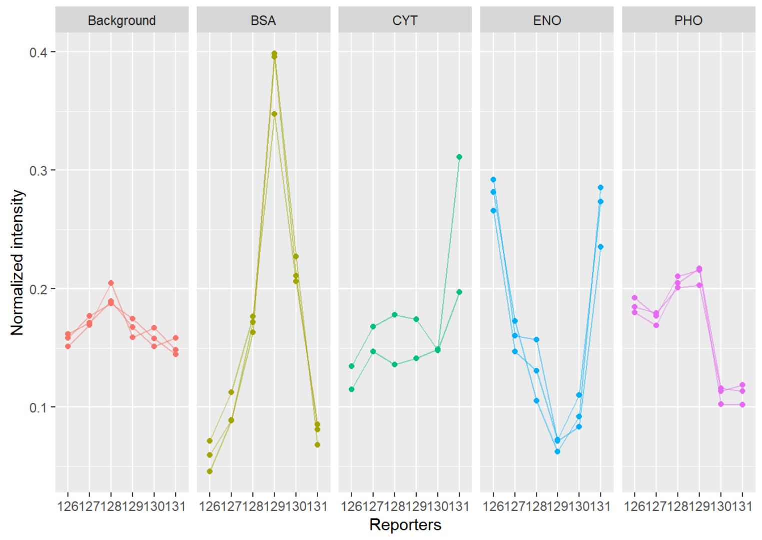
3) Analytical Method exploration 3: Previous modification of normalize from Sum to quantile to be checked on the spikes plot

For method sum for normalization:

```
dfr <- data.frame(exprs(small),
            Protein = as.character(fData(small)$accession),
            Feature = featureNames(small),
            stringsAsFactors = FALSE)
colnames(dfr) <- c("126", "127", "128", "129", "130", "131",
             "Protein", "Feature")
#Replace the protein names in the data frame with more readable names (e.g., ENO, CYT, BSA, PHO) for better visualization
dfr$Protein[dfr$Protein == "sp|P00924|ENO1_YEAST"] <- "ENO"
dfr$Protein[dfr$Protein == "sp|P62894|CYC_BOVIN"]  <- "CYT"
dfr$Protein[dfr$Protein == "sp|P02769|ALBU_BOVIN"] <- "BSA"
dfr$Protein[dfr$Protein == "sp|P00489|PYGM_RABIT"] <- "PHO"
dfr$Protein[grep("ECA", dfr$Protein)] <- "Background"
dfr2 <- melt(dfr)
# Using Protein, Feature as id variables
ggplot(aes(x = variable, y = value, colour = Protein),
    data = dfr2) +
geom_point() +
geom_line(aes(group=as.factor(Feature)), alpha = 0.5) +
facet_grid(. ~ Protein) + theme(legend.position="none") +
labs(x = "Reporters", y = "Normalized intensity")
```

This was simple code for the first method; the code for method quantile is the same, just replacing the dataset that was produced from the quantile normalization method .

A)



B)

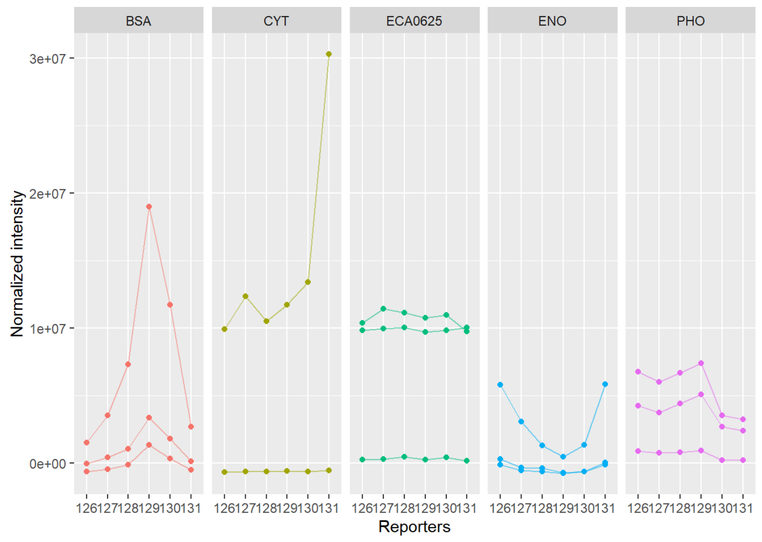


Figure 5: Spike plot for ENO, CYT, BSA, PHO and ECA0625(Background) that are labeled with different TMT labels and intensity of three peptides are shown for each. A) where the

normalization is through sum and B) where normalization through the quantile method produces significantly different results for the normalized intensity.

Results and Discussion:

Going back to the impact of analytical methodologies on protein studies, the results revealed noticeable shifts in protein expression profiles based on methodological choices. Mainly sum vs mean for protein intensity (**Figure 3**) and normalization by vsn and sum or median and quantile (**Figure 4 and 5**) are the methodologies that were explored. **Figure 3**, comparing protein intensities calculated by sum and mean methods, underscored the sensitivity of certain proteins, such as BSA and ENO, to the calculation approach. The variation observed suggests that the choice of calculation method plays a pivotal role in delineating the differential expression of proteins. The sum of peptide intensity or the mean value has a huge difference, meaning that one is very specific while the other is average. This aligns with findings by (Gatto et al., 2014), emphasizing the importance of robust computational methodologies in proteomics data analysis. Furthermore, **Figure 4,** which depicts heat maps generated through different normalization approaches (sum and vsn vs. median and quantile), highlights distinctive clustering patterns. Notably, proteins like BSA, PHO, ENO, and CYT exhibited diverse responses to normalization methods, reinforcing the need for a thoughtful selection of normalization techniques. This was also expected because each normalization method has different value such that VSN (Variance Stabilizing Normalization) addresses heteroscedasticity, sum normalization ensures consistent overall abundance, quantile normalization aligns intensity distributions, and median normalization scales data based on the median intensity, each serving specific purposes in normalizing proteomics data. This resonates with studies by (Nikolov et al. 2012), which emphasize the significance of normalization in quantitative mass spectrometry-based proteomics.

Despite the valuable insights gained, there are certain limitations that influence the generalizability of conclusions. The explorations conducted in this study were based on a specific dataset (PXD000001), and the findings can be influenced by the inherent characteristics of this dataset. The limitations extend to the potential impact of sample size, as addressed by (Noble et al., 2012), who discuss the challenges in computational and statistical analysis of protein mass spectrometry data. These factors introduce a degree of caution in the interpretation, suggesting the need for validation across diverse datasets to establish more robust conclusions.

Looking forward, this study provides a foundation for future investigations. The observed variations in protein expression patterns under different analytical methodologies open avenues for in-depth mechanistic exploration. **Figure 5**, the spike plot contrasting normalization through sum and quantile methods, particularly for ENO, CYT, BSA, PHO, and Background (ECA0625), highlights the need for a more detailed understanding of how normalization impacts specific proteins. This paves the way for targeted studies investigating the molecular underpinnings of these variations, aligning with the recommendations of (Gatto et al., 2014) to delve deeper into specific aspects of proteomic analysis. Overall, the sophisticated proteomics data analysis heavily depends on the choice of analytical methodology.

Reflection:

Through this project and the course, I've developed practical skills in R programming, data analysis, and interpreting proteomics data. Navigating tools like MSnbase and RforProteomics, I gained insights into diverse analytical methods. Managing time effectively and adapting analytical strategies to specific questions were key takeaways. In future coursework and my career, I aim to enhance my statistical skills, explore advanced techniques, and focus on clear communication of complex findings. This experience has emphasized the importance of adaptability and continuous improvement in research and beyond.

References:

Gatto, L., & Christoforou, A. (2014). Using R and Bioconductor for proteomics data analysis. Biochimica et Biophysica Acta (BBA)-Proteins and Proteomics, 1844(1), 42-51.

Nikolov, M., Schmidt, C., & Urlaub, H. (2012). Quantitative mass spectrometry-based proteomics: an overview. Quantitative methods in proteomics, 85-100.

Noble, W. S., & MacCoss, M. J. (2012). Computational and statistical analysis of protein mass spectrometry data. PLoS computational biology, 8(1), e1002296.

https://bioconductor.org/packages/release/data/experiment/vignettes/RforProteomics/inst/doc/RforProteomics.html#7_Annotation

https://www.ebi.ac.uk/pride/archive/projects/PXD000001