lucas_analysis

Lucas Childs

2025-11-03

1. What is the reason for log transforming protein levels in `biomarker-raw.csv`?

```r
set.seed(1234)
library(here)
```

```
## here() starts at /Users/lucaschilds/PSTAT197A/module-1-biomarker-data-table13
```

```r
rawdata <- read.csv(here("data", "biomarker-raw.csv"))

# random sample of 4 proteins to look at distributions of their levels
rand_indices <- sample(3:ncol(rawdata), 4)

prot1d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[1]])
prot1 <- prot1d[!is.na(prot1d)]

prot2d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[2]])
prot2 <- prot2d[!is.na(prot2d)]

prot3d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[3]])
prot3 <- prot3d[!is.na(prot3d)]

prot4d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[4]])
prot4 <- prot4d[!is.na(prot4d)]

summary(prot1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2568    4649    5169    5243    5668    7435
```

```r
summary(prot2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3317    8790   12179   17164   18800  122168
```

```r
summary(prot3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   283.1   378.7   414.2   457.8   496.9  1894.7
```

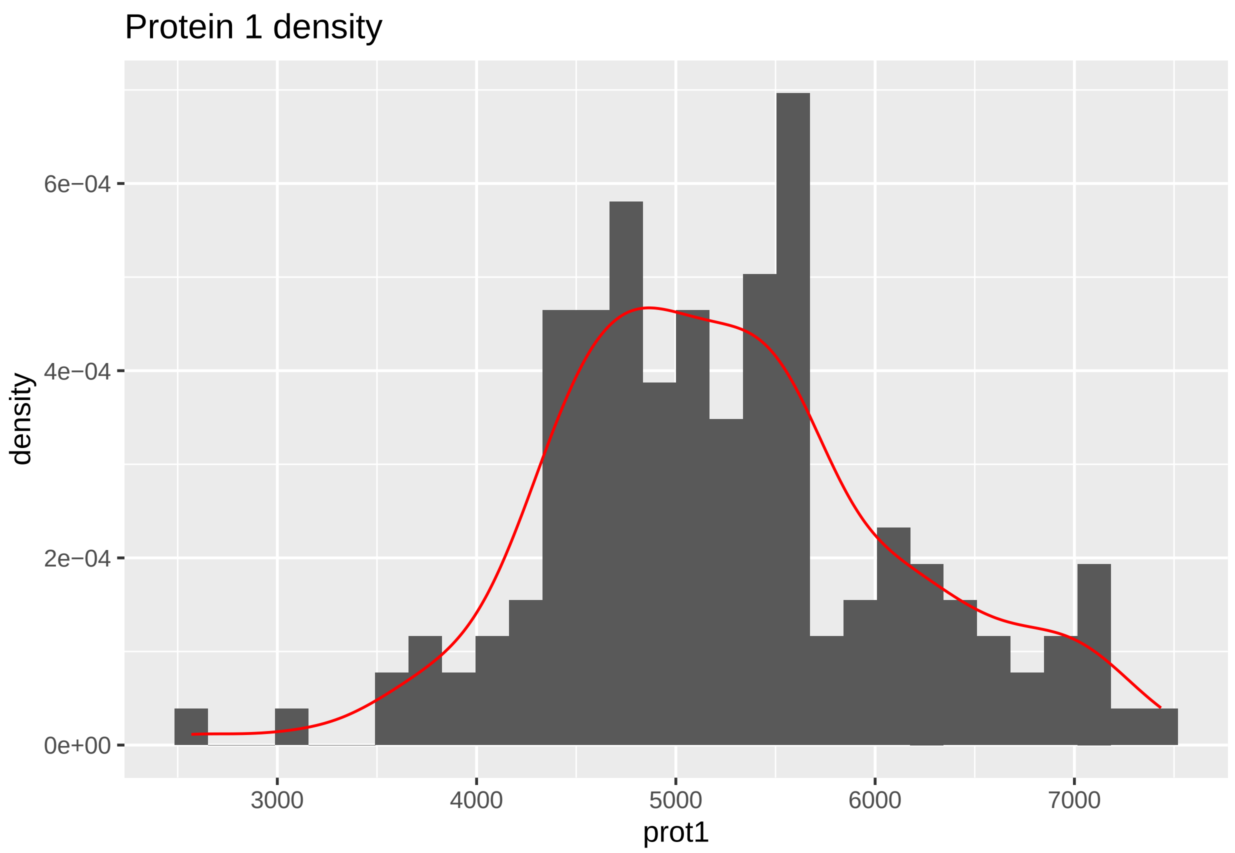
```r
summary(prot4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   954.9  5434.7  6539.2  6551.7  7719.1 10917.8
```

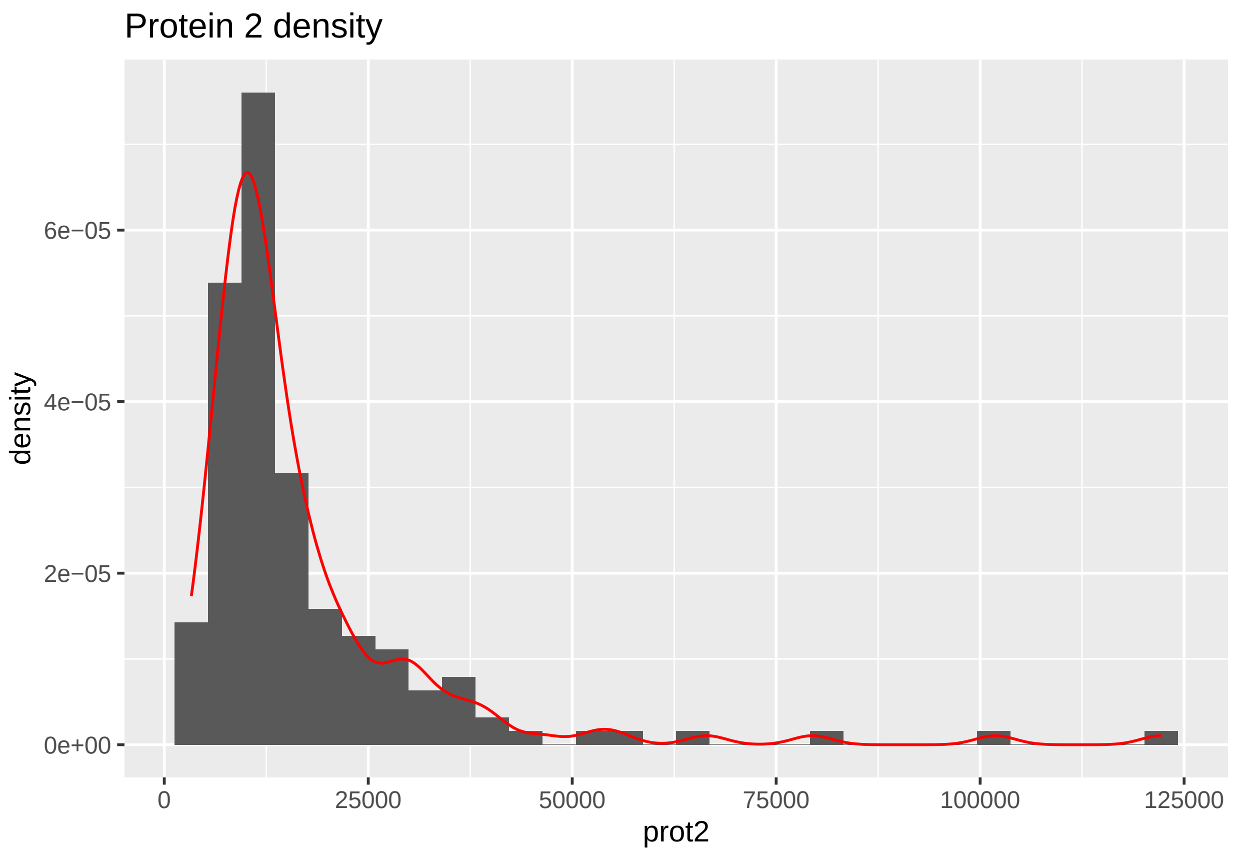
Looking at the summary statistics for 4 randomly selected proteins, the mean values differ significantly, with `prot1`'s mean being 17,164 and `prot3`'s mean being ≈ 458. `prot2`'s mean is roughly 5000 units greater than its median as well, indicating a right skew.

The log transformation of the protein levels helps compress the scale of protein levels since we have a wide range of positive values, where some are very large. Additionally, the logarithm helps with skewed data, and we have evidence that some of the data is skewed, since `prot2` has a mean much larger than its median.

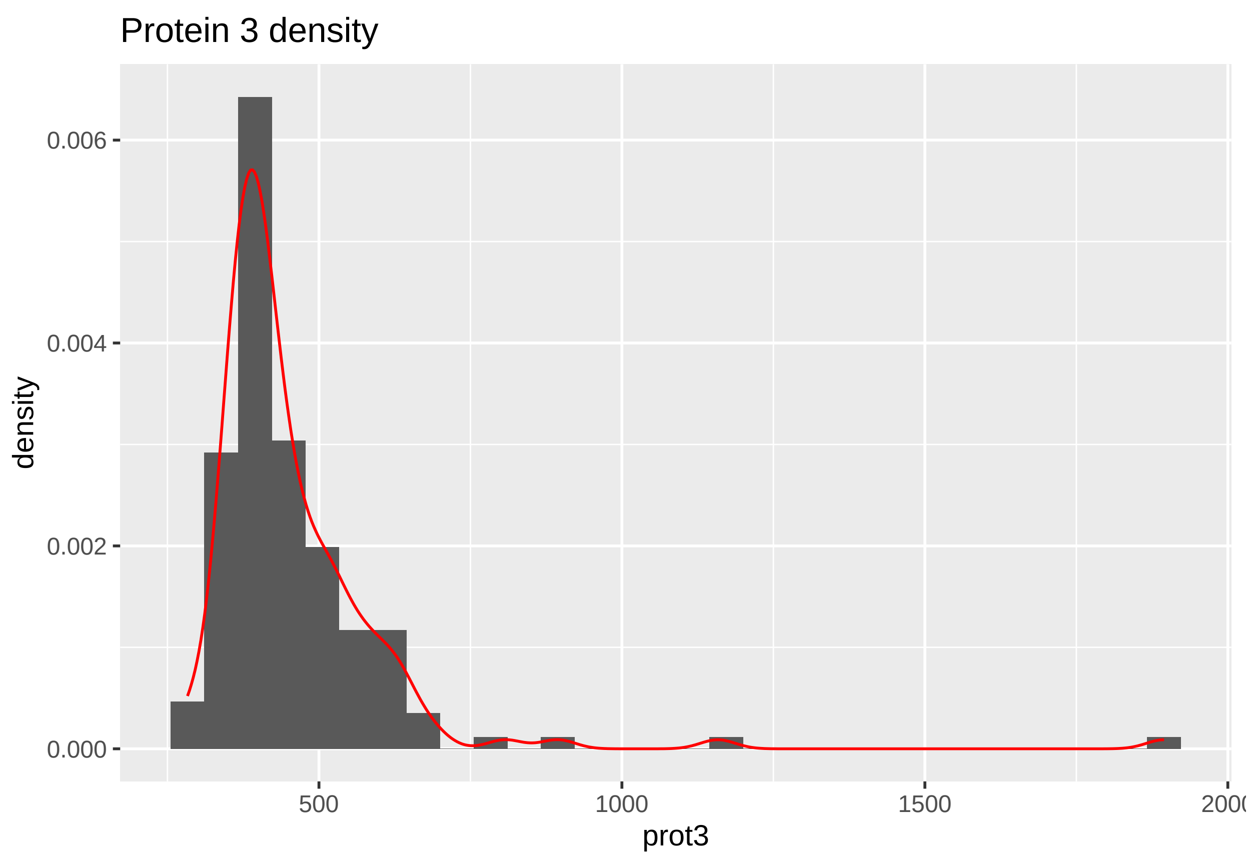
```r
library(ggplot2)
ggplot(as.data.frame(prot1), aes(x = prot1)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 1 density')
```



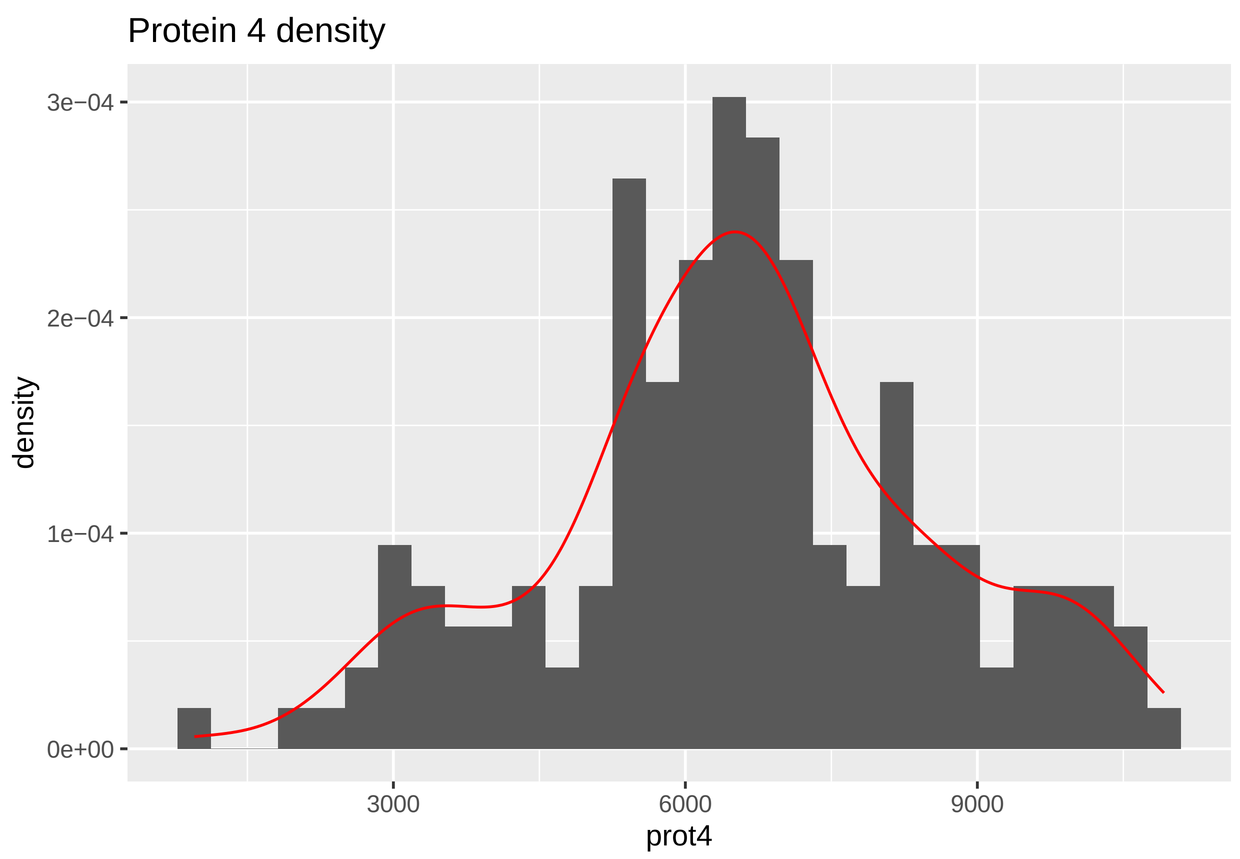
```r
ggplot(as.data.frame(prot2), aes(x = prot2)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 2 density')
```



```
ggplot(as.data.frame(prot3), aes(x = prot3)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 3 density')
```



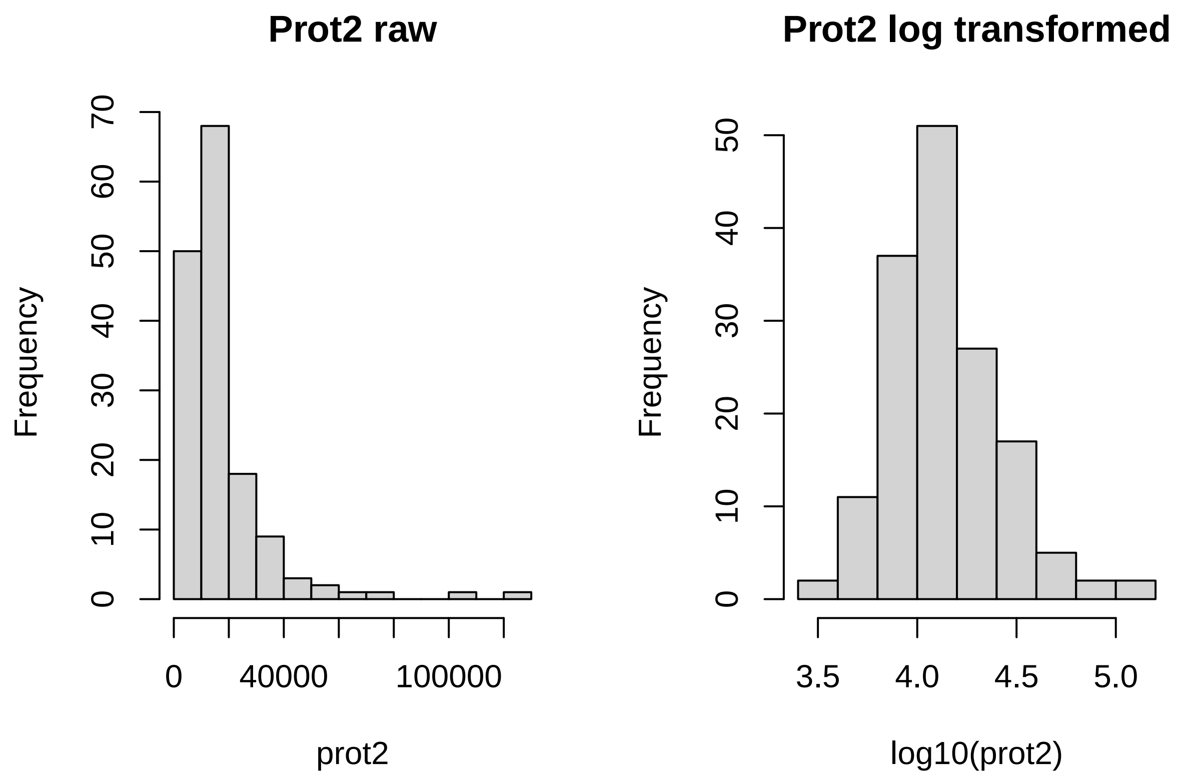
```r
ggplot(as.data.frame(prot4), aes(x = prot4)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 4 density')
```



From the density plots, `prot2` looks the most skewed (strongly right-skewed) `prot3` looks slightly right-skewed as well. Both proteins contain large outliers, however the scale of `prot2`'s protein level is much higher, so the same follows for its outliers.

Comparison of `prot2` histogram to its log transformed counterpart:

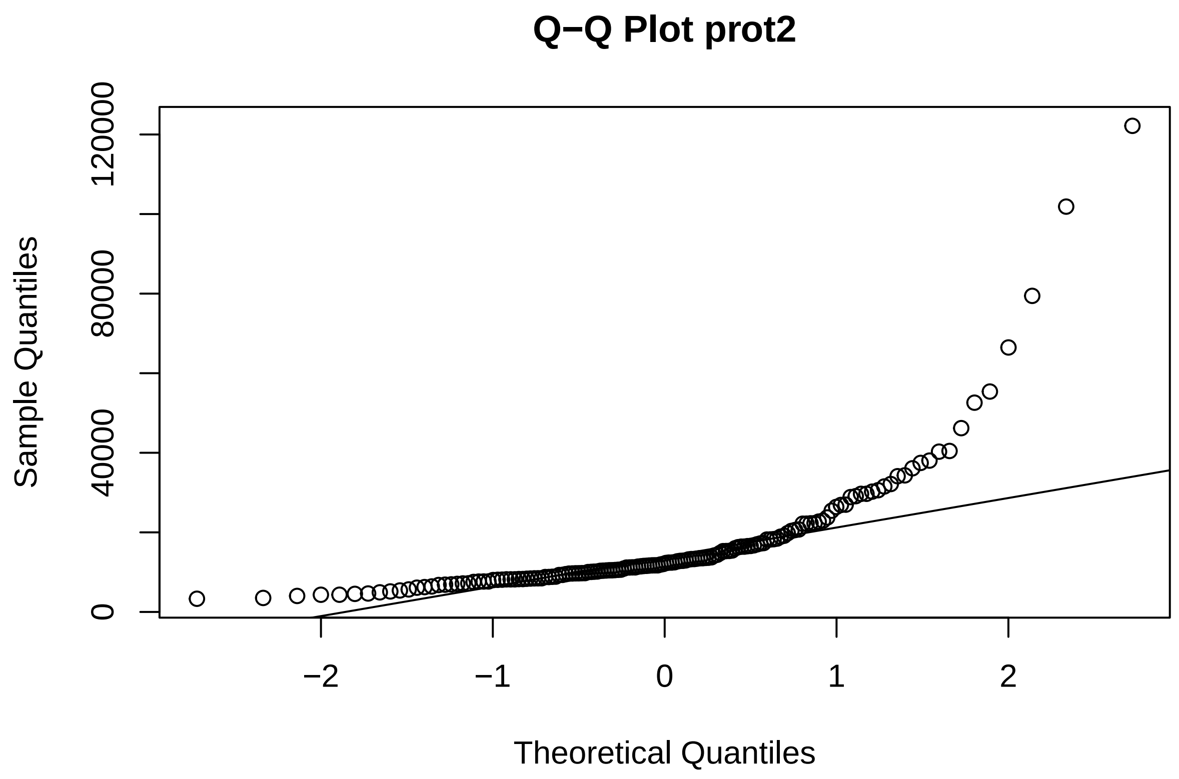
```r
par(mfrow=c(1,2))
hist(prot2, main="Prot2 raw")
hist(log10(prot2), main="Prot2 log transformed")
```



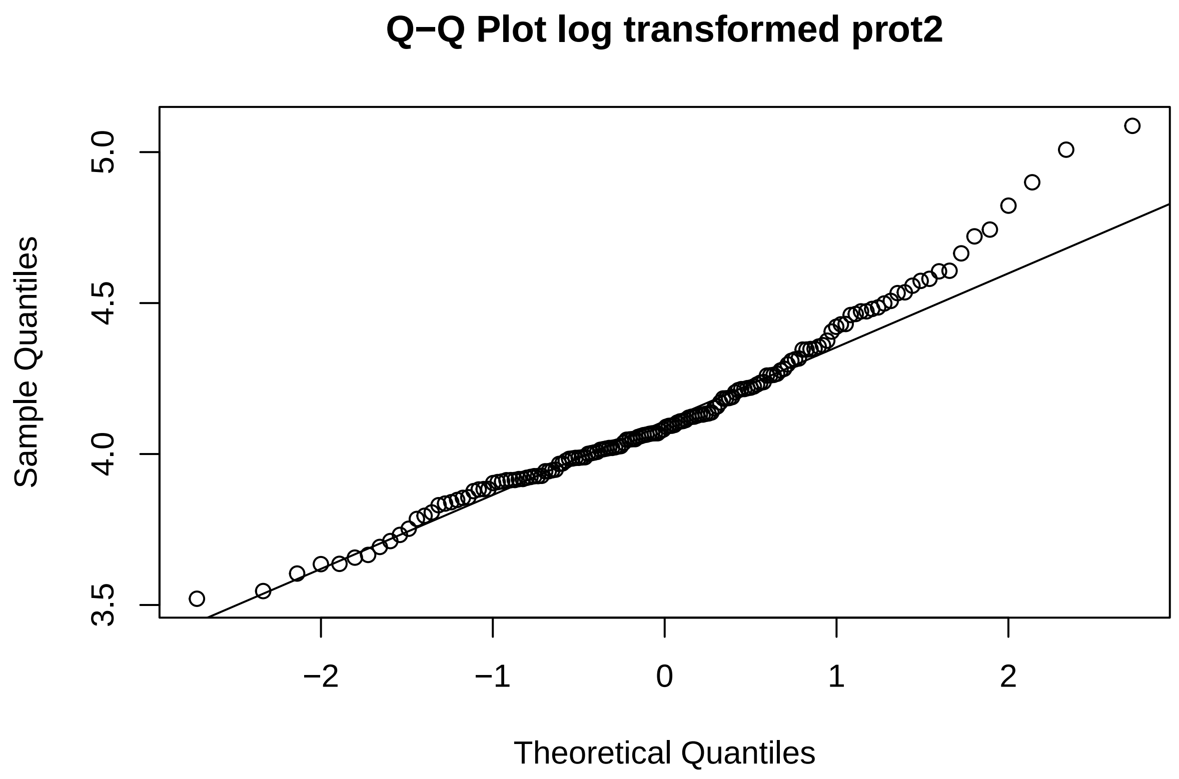
As we can see, after log transforming, the right-skewed `prot2` now appears more symmetric and of a much smaller and more readable scale.

We can check more rigorously to see how normality differs between the raw and log transformed `prot2` with a QQ-Plot.

```r
# normality check for raw data
qqnorm(prot2, main = 'Q-Q Plot prot2')
qqline(prot2)
```



```r
# normality check for log transformed data
qqnorm(log10(prot2), main = 'Q-Q Plot log transformed prot2')
qqline(log10(prot2))
```



After transforming `prot2`, the protein values appear slightly more normal, reducing the influential outliers. Overall, log transforming the protein levels acted as a way to reduce the large scale of the values, and make the data more symmetric.

4. Find an alternative panel that achieves improved classification accuracy

```r
library(tidyverse)
library(infer)
library(randomForest)
library(tidymodels)
library(modelr)
library(yardstick)
library(glmnet)
library(here)

load(here("data", "biomarker-clean.RData")) # loads data as biomarker_clean
```

```r
## MULTIPLE TESTING
####################
# function to compute tests
test_fn <- function(.df){
  t_test(.df,
         formula = level ~ group,
         order = c('ASD', 'TD'),
         alternative = 'two-sided',
         var.equal = F)
}

ttests_out <- biomarker_clean %>%
  # drop ADOS score
  select(-ados) %>%
  # arrange in long format
  pivot_longer(-group,
               names_to = 'protein',
               values_to = 'level') %>%
  # nest by protein
  nest(data = c(level, group)) %>%
  # compute t tests
  mutate(ttest = map(data, test_fn)) %>%
  unnest(ttest) %>%
  # sort by p-value
  arrange(p_value) %>%
  # multiple testing correction
  mutate(m = n(),
         hm = log(m) + 1/(2*m) - digamma(1),
         rank = row_number(),
         p.adj = m*hm*p_value/rank)

# select significant proteins
proteins_s1 <- ttests_out %>%
  slice_min(p.adj, n = 10) %>%
  pull(protein)

## RANDOM FOREST
##################
# store predictors and response separately
predictors <- biomarker_clean %>%
  select(-c(group, ados))

response <- biomarker_clean %>% pull(group) %>% factor()

# fit RF
set.seed(101422)
rf_out <- randomForest(x = predictors,
                       y = response,
                       ntree = 1000,
                       importance = T)

# compute importance scores
proteins_s2 <- rf_out$importance %>%
  as_tibble() %>%
  mutate(protein = rownames(rf_out$importance)) %>%
  slice_max(MeanDecreaseGini, n = 10) %>%
  pull(protein)

## LASSO with guided penalty
###########################################
```

```r
# Get the intersection of important proteins
proteins_sstar <- intersect(proteins_s1, proteins_s2)
cat("Selected proteins from intersection:", proteins_sstar, "\n")
```

```
## Selected proteins from intersection: DERM RELT MRC2 IgD Cadherin-5
```

```r
cat("Number of selected proteins:", length(proteins_sstar), "\n")
```

```
## Number of selected proteins: 5
```

```r
# LASSO with all proteins, choosing different penalties for sstar vs x_train

# full dataset (all proteins except ADOS)
biomarker_full <- biomarker_clean %>%
  select(-ados) %>%
  mutate(class = (group == 'ASD')) %>%
  select(-group)

# training and test set
set.seed(101422)
biomarker_split <- biomarker_full %>%
  initial_split(prop = 0.8)

# matrices for glmnet
x_train <- training(biomarker_split) %>%
  select(-class) %>%
  as.matrix()

y_train <- training(biomarker_split) %>%
  pull(class) %>%
  as.numeric()

x_test <- testing(biomarker_split) %>%
  select(-class) %>%
  as.matrix()

y_test <- testing(biomarker_split) %>%
  pull(class) %>%
  as.numeric()

# penalty factors: lower penalty for pre-selected proteins
penalty_factors <- rep(1, ncol(x_train))  # Start with penalty = 1 for all
names(penalty_factors) <- colnames(x_train)

# no penalty for sstar proteins (ones selected by t-test and RF)
penalty_factors[proteins_sstar] <- 0

cat("\nPenalty factors:\n")
```

```
##
## Penalty factors:
```

```r
print(table(penalty_factors))
```

```
## penalty_factors
##    0    1
##    5 1312
```

```r
# fit LASSO with guided penalties
set.seed(101422)
lasso_guided <- cv.glmnet(
  x_train, y_train,
  alpha = 1,
  family = "binomial",
  penalty.factor = penalty_factors,
  type.measure = "auc",
  nfolds = 10
)

# get predictions
pred_prob_guided <- predict(lasso_guided, newx = x_test,
                           s = "lambda.min", type = "response")[,1]

# metrics
test_results_guided <- data.frame(
  pred = pred_prob_guided,
  pred_class = factor(pred_prob_guided > 0.5,
                      levels = c(FALSE, TRUE),
                      labels = c("TD", "ASD")),
  class_factor = factor(y_test,
                       levels = c(0, 1),
                       labels = c("TD", "ASD"))
)

# class-based metrics
class_metrics <- metric_set(sensitivity, specificity, accuracy)
class_output_guided <- test_results_guided %>%
  class_metrics(estimate = pred_class,
                truth = class_factor,
                event_level = 'second')

# AUC
prob_output_guided <- test_results_guided %>%
  roc_auc(truth = class_factor,
          pred,
          event_level = 'second')

# Combine all metrics
all_metrics_guided <- bind_rows(class_output_guided, prob_output_guided)
print("\nGuided LASSO Results:")
```

```
## [1] "\nGuided LASSO Results:"
```

```r
print(all_metrics_guided)
```

```
## # A tibble: 4 x 3
##   .metric     .estimator .estimate
##   <chr>       <chr>          <dbl>
## 1 sensitivity binary         0.75
## 2 specificity binary         0.933
## 3 accuracy    binary         0.839
## 4 roc_auc     binary         0.858
```

```r
# which coefficients are non-zero
coef_guided <- coef(lasso_guided, s = "lambda.min")
selected_features <- rownames(coef_guided)[which(coef_guided != 0)]
selected_features <- selected_features[selected_features != "(Intercept)"]

cat("\nSelected features by guided LASSO:\n")
```

```
##
## Selected features by guided LASSO:
```

```r
print(selected_features)
```

```
##  [1] "CD59"                         "4-1BB"
##  [3] "Dtk"                          "Cadherin-5"
##  [5] "HAI-1"                        "Kallikrein 11"
##  [7] "PAI-1"                        "Growth hormone receptor"
##  [9] "IGFBP-4"                      "MRC2"
## [11] "CRDL1"                        "IL-17 RD"
## [13] "TPSG1"                        "MP2K2"
## [15] "ENPP7"                        "MFGM"
## [17] "PCSK7"                        "ITI heavy chain H4"
## [19] "IgD"                          "DBNL"
## [21] "DERM"                         "Elafin"
## [23] "RELT"                         "PPID"
## [25] "Semaphorin 3E"                "CD27"
## [27] "CNDP1"                        "IL-17 RC"
## [29] "SRCN1"                        "Epo"
## [31] "GDNF"                         "14-3-3 protein zeta/delta"
## [33] "a-Synuclein"                  "CSRP3"
## [35] "MIG"
```

```r
cat("\nFeatures from proteins_sstar that were kept:\n")
```

```
##
## Features from proteins_sstar that were kept:
```

```r
print(intersect(selected_features, proteins_sstar))
```

```
## [1] "Cadherin-5" "MRC2"       "IgD"        "DERM"       "RELT"
```

Evaluation metrics: In Hewitson et al. 2021, a logistic regression model was fit using their final panel of proteins, and they evaluated its performance using AUC, sensitivity, and specificity. To benchmark our method, we'll compare our results to the `inclass-analysis.R` results, where the metrics were

Their results were:

$$\text{roc auc} = 0.883 \text{ ; accuracy} = 0.774 \text{ ; sensitivity} = 0.812 \text{ ; specificity} = 0.733$$

Adding more benchmark results to `bahaar_analysis`:

```
rmarkdown::render("bahaar_analysis.rmd")
```

```
##   |                                                                      |

##   |                                                                      |................................

## /Applications/RStudio.app/Contents/Resources/app/quarto/bin/tools/aarch64/pandoc +RTS -K512m -RTS ba
```

```
# Get confusion matrix object
cm <- confusionMatrix(pred_class_lasso, y_test, positive = "ASD")

# Print all metrics
cat("AUC ROC (LASSO classifier) on Test Set: ", auroc_final, "\n")
```

```
## AUC ROC (LASSO classifier) on Test Set:  0.8577778
```

```
cat("Test Set Accuracy (LASSO classifier): ", cm$overall['Accuracy'], "\n")
```

```
## Test Set Accuracy (LASSO classifier):  0.7666667
```

```
cat("Sensitivity (LASSO classifier): ", cm$byClass['Sensitivity'], "\n")
```

```
## Sensitivity (LASSO classifier):  0.8
```

```
cat("Specificity (LASSO classifier): ", cm$byClass['Specificity'], "\n")
```

```
## Specificity (LASSO classifier):  0.7333333
```

Thus, accuracy slightly down, same for auc, sensitivity, but specificity remains the same.