MVA Assignment - 19343511

Ross Jackson

2023-03-29

Loading necessary packages:

```
knitr::opts_chunk$set(tidy.opts=list(width.cutoff=60))
library(reshape2)
library(ggplot2)
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:reshape2':
##
##     smiths
```

```
library(reshape2)
library(cluster)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(pls)
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
library(missMDA)
```

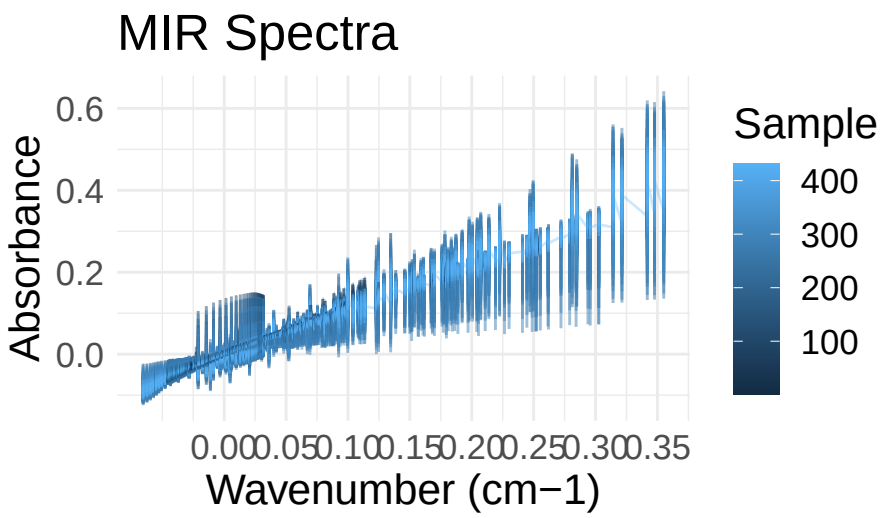
QUESTION 1:

```
milk = read.csv("/Users/rossjackson/Desktop/Multivariate Analysis/Milk_MIR_Traits_data_2023.csv")
set.seed(19343511)
rand_num <- sample(1:nrow(milk), 1) #Randomly generated no. between 1 and n and delete
data <- milk[-rand_num, ]
```

QUESTION 2:

```
#Removing NA / missing values
data_clean <- data[complete.cases(data$beta_lactoglobulin_b),]
#Firstly extract wavenumbers from 1st row of the spctra columns as numeric data
wavenumbers <- as.numeric(data_clean[1, 52:ncol(data_clean)])
# Remove the first row containing the wavenumbers from the spectra data
```

```
#(seperating the two)

spectra <- data_clean[-1, 52:ncol(data_clean)]

# Using melt function to reshape the spectra data from column by column for each
#wavenumber, to long format where each row contains a combo of sample,
#wavenumber and absorbance values. Had to manipulate the data in this way in
#order to visualise the data. The melt function also requires me to change the
#spectra data to a matrix using as.matrix

spectra_melt <- melt(as.matrix(spectra))
#Column renaming
colnames(spectra_melt) <- c("Sample", "Wavenumber_Index", "Absorbance")

# Add actual wavenumbers to the manipulated data from above. Vector contains the
# corresponding wavenumber values for each column in the spectra data.
spectra_melt$Wavenumber <- wavenumbers[spectra_melt$Wavenumber_Index]

ggplot(spectra_melt, aes(x = Wavenumber, y = Absorbance, color = Sample)) +
  geom_line(alpha = 0.3) +
  theme_minimal() +
  labs(title = "MIR Spectra",
       x = "Wavenumber (cm-1)",
       y = "Absorbance") +
  theme(
    text = element_text(size=16),
    plot.margin = unit(c(0.5, 0.5, 0.5, 0.5), "cm") #Adjusting so that it fits well in the RMD
  ) +
  scale_x_continuous(breaks = seq(0, max(spectra_melt$Wavenumber), by = 0.05))
```

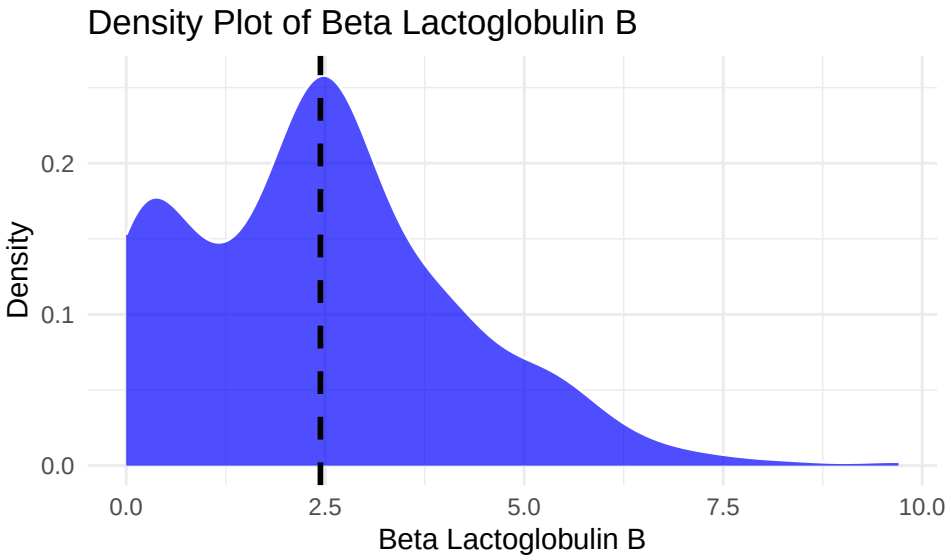


There are few notable points from the above plot of the Spectra from the data. In the Wavenumber region from roughly -0.05 to 0.03, there is a spike in absorbance values, suggesting the presence of certain chemicals/proteins. As wavelengths increase, the wavelength values become less dense, with a wider range of absorbance values.

```
ggplot(data_clean[-1,], aes(x = beta_lactoglobulin_b)) +
  geom_density(fill = "blue", alpha = 0.7, color = "white") +
  geom_vline(aes(xintercept = mean(beta_lactoglobulin_b)), color = "black", linetype = "dashed", size =
```

```
  labs(title = "Density Plot of Beta Lactoglobulin B",
       x = "Beta Lactoglobulin B",
       y = "Density") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
```



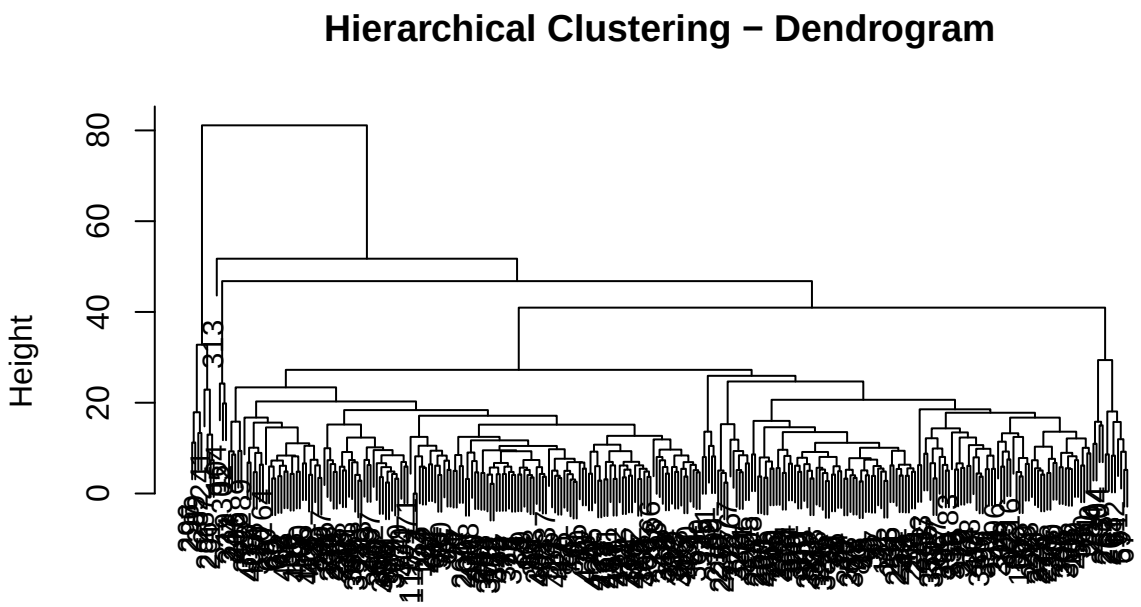
As we can see from the above graph, the Density plot for Beta Lactoglobulin B expresses a right-skewness present in the data. This may be due to the fact that the data points that exist furthest from the mean for such points with high levels of Beta Lactoglobulin B. The vertical dotted line represents the mean, which appears to be just shy of 2.5 (actual mean = 2.439735, obtained in Rough Work Analysis for validation)

```
#Calculate the mean and standard deviation of beta_lactoglobulin_b
mean_val <- mean(data_clean$beta_lactoglobulin_b)
sd_val <- sd(data_clean$beta_lactoglobulin_b)

#Calculate the lower and upper bounds (3 standard deviations from the mean)
lower_bound <- mean_val - 3 * sd_val
upper_bound <- mean_val + 3 * sd_val

data_filtered <- data_clean[data_clean$beta_lactoglobulin_b >= lower_bound & data_clean$beta_lactoglobul
```

QUESTION 3:

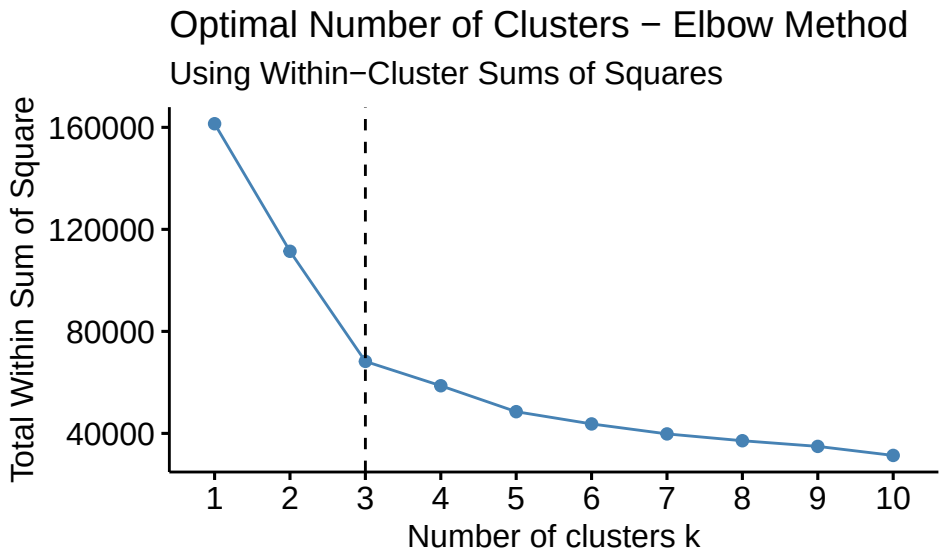
Hierarchical:

```
#HC clustering :
spectra_scaled <- scale(spectra)
hc <- hclust(dist(spectra_scaled), method = "average")

# Visualize the dendrogram
plot(hc, main = "Hierarchical Clustering - Dendrogram", xlab = "", sub = "", cex = 0.9)
```

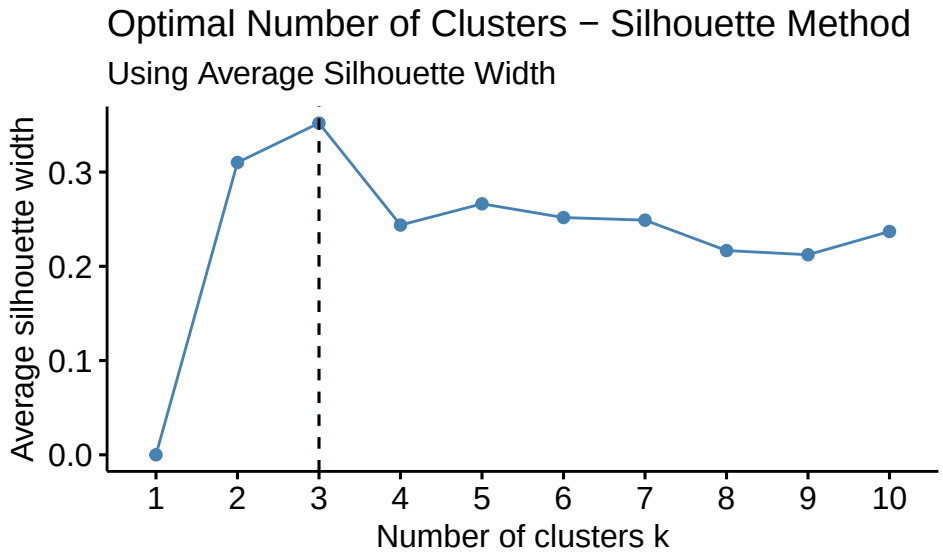


K-Means: Finding Optimal number of clusters

```
#Elbow method to determine optimal K
fviz_nbclust(spectra_scaled, kmeans, method = "wss") +
  geom_vline(xintercept = 3, linetype = 2) +
  labs(title = "Optimal Number of Clusters - Elbow Method",
       subtitle = "Using Within-Cluster Sums of Squares")
```

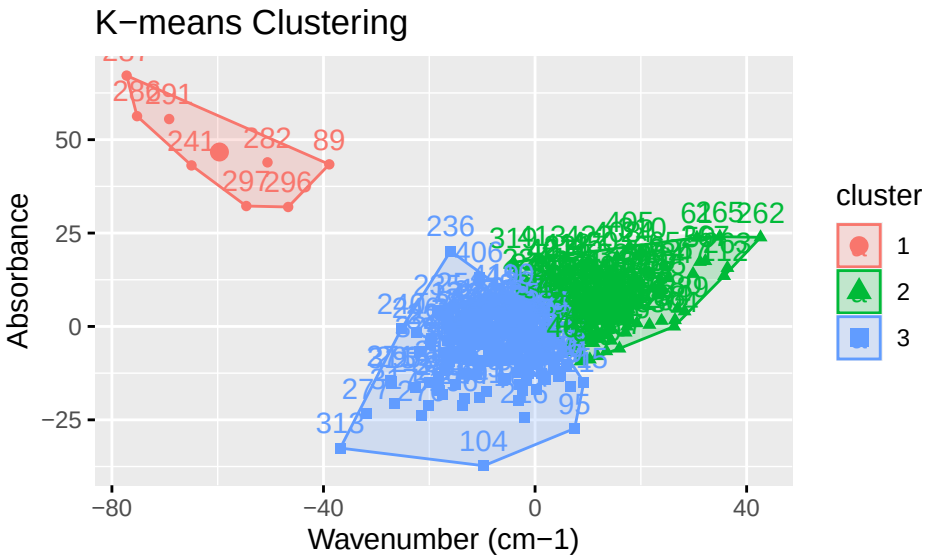



```
#Validating by seeing if Silhouette method gives same result
fviz_nbclust(spectra_scaled, kmeans, method = "silhouette") +
  geom_vline(xintercept = 3, linetype = 2) +
  labs(title = "Optimal Number of Clusters - Silhouette Method",
       subtitle = "Using Average Silhouette Width")
```



As seen from the plot produced from the above, the optimal number of clusters using the elbow method is 3, which has been validated through both the elbow and silhouette methods of finding the optimal number of clusters. In the ELbow method graph, we know 3 is the optimal number as it is the point in the curve where the WSS value begins to decrease at a lower rate than before. For the silhoutte method, we know 3 is the optimal number as the optimal number of clusters k is given when the average silhouette width is maximized.

```
#As seen from the plot produced from the above, the optimal number of clusters using the elbow method i
# Perform k-means clustering with k = 3
k <- 3
kmeans_result <- kmeans(spectra_scaled, centers = k)

# Visualize the k-means clustering results
fviz_cluster(kmeans_result, data = spectra_scaled) +
  labs(title = "K-means Clustering",
       x = "Wavenumber (cm-1)",
       y = "Absorbance")
```

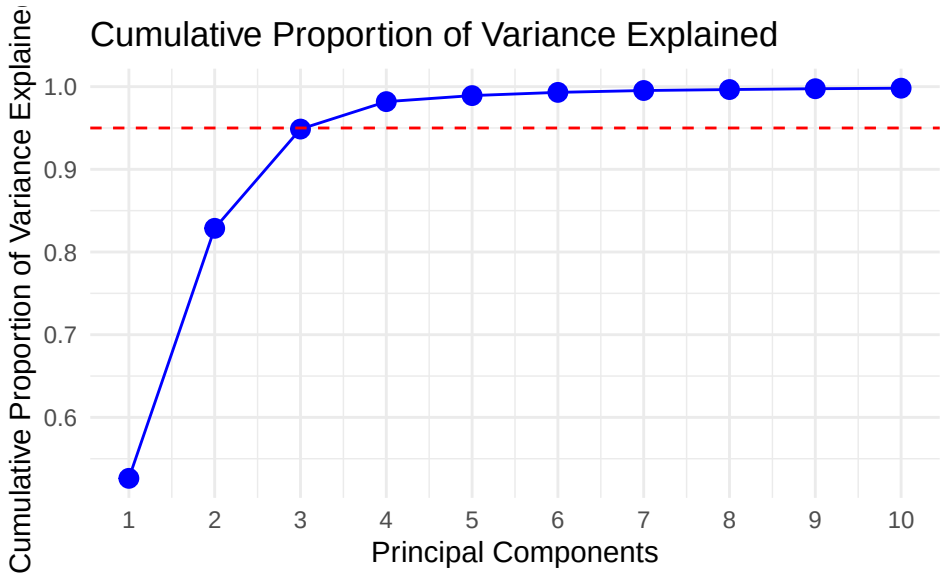


As we can see from the K-Means clustering plot we have obtained before, this method identifies 3 similar clusters from the MIR spectra data. It is also clear that we have chosen the correct number of K clusters, as

there is little overlap in the clusters (validated through trial and error that this is in fact the optimal number for k). This allows us to conclude that there are 3 groups of significantly different clusters present in the MIR spectra data, bar a slight overlap between cluster 1 and 3. Cluster 2 seems to account for a lesser number of observations than the other 2, suggesting that it could be an outlier groups of spectral data. Clusters 1 and 3 are likely to have similarities and share some characteristics.

QUESTION 4:

```
#PCA on the scaled spectral data
pca <- prcomp(spectra_scaled, center = TRUE, scale. = TRUE)

#record  proportion of variance explained by each principal component
pve <- pca$sdev^2 / sum(pca$sdev^2)

#Calculate the cumulative proportion of variance explained (CPVE) for the first 10 principal components
cpve <- cumsum(pve)[1:10]

#data frame for plotting
cpve_df <- data.frame(
  Principal_Component = 1:10,
  Cumulative_Proportion_of_Variance_Explained = cpve
)

#Plotting CPVE  for first 10 principal components
ggplot(cpve_df, aes(x = Principal_Component, y = Cumulative_Proportion_of_Variance_Explained)) +
  geom_point(size = 3, color = "blue") +
  geom_line(color = "blue") +
  geom_hline(yintercept = 0.95, linetype = "dashed", color = "red") + # Add a horizontal line at 95%
  labs(title = "Cumulative Proportion of Variance Explained",
       x = "Principal Components",
       y = "Cumulative Proportion of Variance Explained") +
  theme_minimal() +
  scale_x_continuous(breaks = 1:10)
```

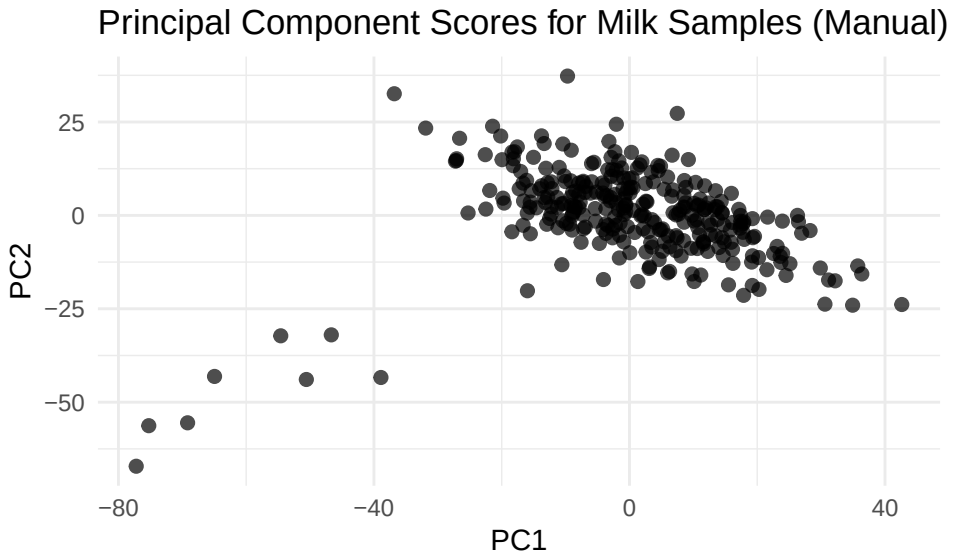


From this graph, we can see that the number of Principal components required to represent the data here is 3. This is the point where the curve levels off, and this is due to the fact that most of the variation in the

spectral data is explained by these first 3 components.

QUESTION 5:

```
#Centre+Scale data
spectra_centered_scaled <- scale(spectra)

#Calc the covariance matrix of the scaled data
cov_matrix <- cov(spectra_centered_scaled)

#eigenvalue decomposition on the covariance matrix
eig <- eigen(cov_matrix)
loadings <- eig$vectors

#Get PC scores manually for the first 2 principal components
pc_scores_manual <- spectra_centered_scaled %*% loadings[, 1:2]

#Convert the scores to a data frame
pc_scores_df_manual <- as.data.frame(pc_scores_manual)
colnames(pc_scores_df_manual) <- c("PC1", "PC2")

#Add row numbers as an identifier
pc_scores_df_manual$SampleID <- 1:nrow(pc_scores_df_manual)

#Plot the PC scores for the milk samples
ggplot(pc_scores_df_manual, aes(x = PC1, y = PC2)) +
  geom_point(size = 2, alpha = 0.7) +
  labs(title = "Principal Component Scores for Milk Samples (Manual)",
       x = "PC1",
       y = "PC2") +
  theme_minimal()
```



From the graph we can conclude that there is a clear cluster just right of the centre of the plot. This could represent a group of similar observations which have similar traits and spectral values. There is a slight tendancy in the data down towards the negative quadrant suggesting another potential clustering in the data (which would intuitively make sense given the K-Means graph we observed above).

QUESTION 6:

i). The main purpose of the PCR model is to confront a common challenge in the world of Data Analysis, that of data sets having a large number of potentially correlated/collinear predictor variables, and relatively few observations or samples. This increases the interpretability of datasets with a large number of predictor variables ("The main reason is that they have been designed to confront the situation that there are many, possibly correlated, predictor variables, and relatively few samples—a situation that is common, especially in chemistry where developments in spectroscopy since the seventies have revolutionised chemical analysis." - The pls Package: Principal Component and Partial Least Squares Regression in R by Mevik and Wehrens (2007).) The models purpose and utility on such datasets has made it a prolific model of choice in the fields of Chemistry and Natural sciences, as well as finance and more recently it has been used in image processing systems.

ii). The method works by firstly constructing the first M Principal Components, denoted by Z1,...,ZM. These are the principal components derived in the Principal Component Analysis (PCA) phase. These components which usually explain a high percentage of the variability in the data, are then used as the predictor variables in a linear regression model. This model is fitted using the least squares method, and this process shows how the PCR method serves its purpose explained above. This is due to the fact that the variation in datasets can usually be explained by a relatively small number of principal components, allowing the PCR method to also explain the relationship between the predictors and the response variable. It may now be obvious that there is an assumption here, in that if the PCR holds, fitting a least squares model to components Z1,...,ZM will lead to more robust results than fitting the same model to X1,...,Xp variables. This is made obvious when you reconsider the earlier point that most of the information in the data is explained by the Z1,...,ZM components, which results in less noise and/or uninformative variables being considered.

iii). The most common choice that needs to be made when implanting the PCR method is that of standardization. In general, standardization is recommended in order to ensure that all of the variables in question are on the same scale. This step eliminates high-variance variables ability to impact the Principal Components that will be obtained from the method. It ensures that these high variance variables don't have an unnecessarily large impact on the resulting PCR model obtained. The choice here exists in whether or not the variables are measured in the same units eg. Kg's or inches, and in such a case one may choose not to standardize their data. One must also choose the number of Principal Components they wish to retain for the final PCR. Retaining a number of PC's that is too small may result in underfitting, and vice versa for too many PC's. After one decides on the number of PC's, they must then use their information to decide on an optimal regression model for the given number of PC's.

iv). Advantages: The PCR method offers a number of advantages; mainly dimensionality reduction, Solving issues with collinearity/multicollinearity and being a very efficient method to conduct such analysis. One of the main things PCR does is dimensionality reduction, making the data far more interpretable.

Disadvantages: On the flip side of this, the PC's obtained in the PCA process provides us with linear combinations of the predictor variables in the data, and although this may now be more interpretable, it can also be less meaningful, making it difficult to interpret the underlying relationship between the predictors and the response variable. It should also be noted that PCR must not be mistaken for a feature selection method ("We note that even though PCR provides a simple way to perform regression using $M < p$ predictors, it is not a feature selection method. This is because each of the M principal components used in the regression is a linear combination of all p of the original features" - An Introduction to Statistical Learning with Applications in R by James et al. (2021))

QUESTION 7:Use the function pcr in the pls R package to use PCR to predict the Lactoglobulin B levels from the spectra for a test set, where the test set is one third of the data. Motivate any decisions you make.

```
# Create idx for the training set
train_idx <- sample(seq_len(nrow(data_filtered)), size = floor(2 * nrow(data_filtered) / 3))
```

```r
#Split  data into training and test sets
train_data <- data_filtered[train_idx, ]
test_data <- data_filtered[-train_idx, ]

#Extract spectral data and response variable for the training set
train_spectra <- train_data[, 52:582]
train_beta_lactoglobulin_b <- train_data$beta_lactoglobulin_b
train_data_combined <- cbind(train_spectra, beta_lactoglobulin_b = train_beta_lactoglobulin_b)

#Obtain spectral data and response variable  for the training set + combine
train_beta_lactoglobulin_b_df <- data.frame(beta_lactoglobulin_b = train_beta_lactoglobulin_b)
train_data_combined <- cbind(train_spectra, beta_lactoglobulin_b = train_beta_lactoglobulin_b)

#PCR function
pcr_model <- plsr(beta_lactoglobulin_b ~ ., data = train_data_combined, scale = TRUE, validation = "CV"
pcr_model
```

```
## Partial least squares regression , fitted with the kernel algorithm.
## Cross-validated using 10 random segments.
## Call:
## plsr(formula = beta_lactoglobulin_b ~ ., data = train_data_combined,     scale = TRUE, validation =
```

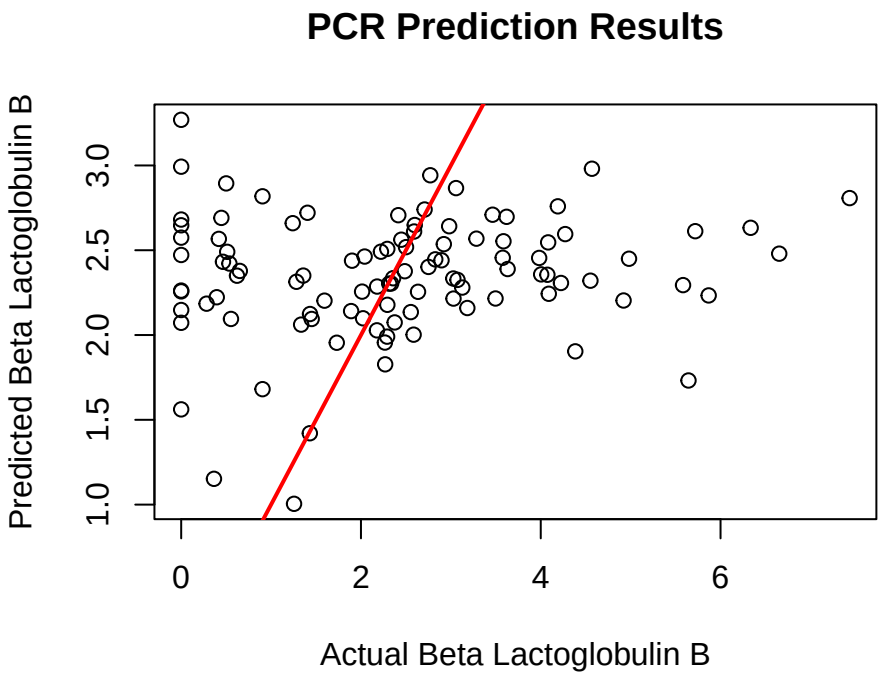
```r
#Calc the RMSEP for each number of components
rmsep_values <- RMSEP(pcr_model, estimate = "CV")$val[, , 1]

#Find optimal no. of components
optimal_components <- which.min(rmsep_values)
cat("Optimal number of components:", optimal_components)
```

```
## Optimal number of components: 1
```

```r
#Fit PCR model with the optimal no. of components
pcr_optimal_model <- plsr(beta_lactoglobulin_b ~ ., data = train_data_combined, ncomp = optimal_componer

#Extract the test set spectra and response variable
test_spectra <- test_data[, 52:582]
test_beta_lactoglobulin_b <- test_data$beta_lactoglobulin_b

#Predict Beta Lactoglobulin B levels for the test set
predicted_beta_lactoglobulin_b <- predict(pcr_optimal_model, newdata = test_spectra, ncomp = optimal_co

#Calc RMSE on the test set
rmse <- sqrt(mean((predicted_beta_lactoglobulin_b - test_beta_lactoglobulin_b)^2))
cat("RMSE on the test set:", rmse, "\n")
```

```
## RMSE on the test set: 1.6796
```

```r
#plot of the predicted vs. actual Beta Lactoglobulin B levels
plot(test_beta_lactoglobulin_b, predicted_beta_lactoglobulin_b, xlab = "Actual Beta Lactoglobulin B", y
abline(a = 0, b = 1, col = "red", lwd = 2)
```



QUESTION 8:

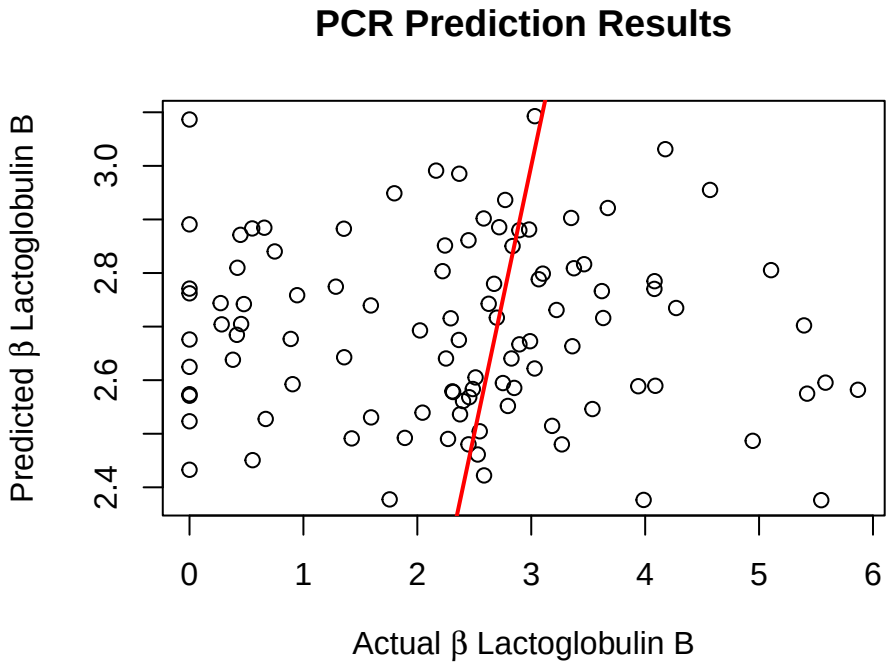
QUESTION 9:

(a).

```
#Filter the data
train_data_nonzero <- data_filtered[data_filtered$beta_lactoglobulin_b > 0, ]

#Split the data into training and test sets
train_idx <- sample(seq_len(nrow(data_filtered)), size = floor(2 * nrow(data_filtered) / 3))
test_data <- data_filtered[-train_idx, ]

#Perform PCR
train_spectra <- train_data_nonzero[, 52:582]
train_beta_lactoglobulin_b <- train_data_nonzero$beta_lactoglobulin_b
train_data_combined <- cbind(train_spectra, beta_lactoglobulin_b = train_beta_lactoglobulin_b)

pcr_model <- pcr(beta_lactoglobulin_b ~ ., data = train_data_combined, scale = TRUE, validation = "CV")
optimal_components <- which.min(RMSEP(pcr_model, estimate = "CV")$val[, , 1])
pcr_optimal_model <- pcr(beta_lactoglobulin_b ~ ., data = train_data_combined, ncomp = optimal_componen

#Predict the Beta Lactoglobulin B values for the test set
test_spectra <- test_data[, 52:582]
test_beta_lactoglobulin_b <- test_data$beta_lactoglobulin_b
predicted_beta_lactoglobulin_b <- predict(pcr_optimal_model, newdata = test_spectra, ncomp = optimal_co

#Calc RMSE on the test set
rmse <- sqrt(mean((predicted_beta_lactoglobulin_b - test_beta_lactoglobulin_b)^2))
cat("RMSE on the test set:", rmse, "\n")
```

```
## RMSE on the test set: 1.541682
```

```r
#plot of the predicted vs. actual Beta Lactoglobulin B levels
plot(test_beta_lactoglobulin_b, predicted_beta_lactoglobulin_b, xlab = expression(paste("Actual ", beta
abline(a = 0, b = 1, col = "red", lwd = 2) # Add a 45-degree line to represent perfect predictions
```



As we can see from the plot, filtering the data for only non-zero values has not improved the observed relationship between actual and predicted Beta Lactoglobulin B values. The RMSE is also slightly higher than in the previous PCR model obtained.

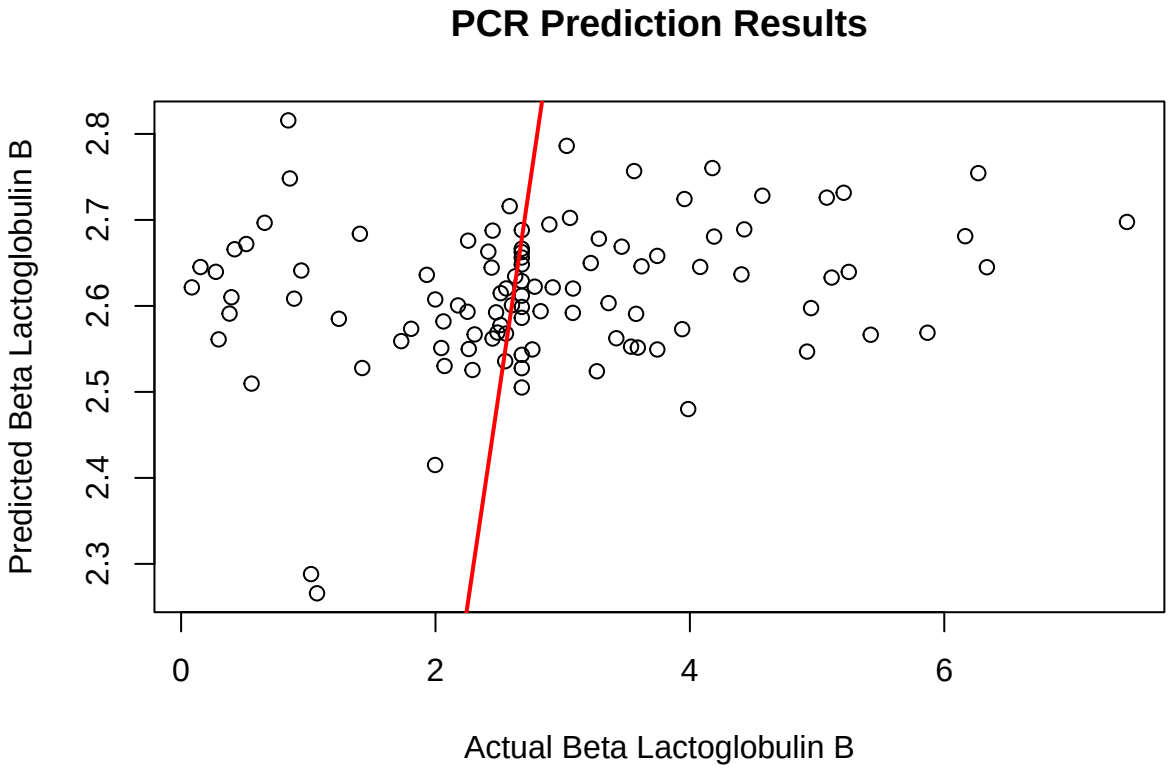
(b).

```r
#Impute 0 values of Beta Lactoglobulin B with the observed mean
observed_mean <- mean(data_filtered$beta_lactoglobulin_b[data_filtered$beta_lactoglobulin_b > 0])
data_imputed <- data_filtered
data_imputed$beta_lactoglobulin_b[data_imputed$beta_lactoglobulin_b == 0] <- observed_mean

#Split the data
train_idx <- sample(seq_len(nrow(data_imputed)), size = floor(2 * nrow(data_imputed) / 3))
train_data <- data_imputed[train_idx, ]
test_data <- data_imputed[-train_idx, ]

#Perform PCR on the training set + obtain model
train_spectra <- train_data[, 52:582]
train_beta_lactoglobulin_b <- train_data$beta_lactoglobulin_b
train_data_combined <- cbind(train_spectra, beta_lactoglobulin_b = train_beta_lactoglobulin_b)

pcr_model <- pcr(beta_lactoglobulin_b ~ ., data = train_data_combined, scale = TRUE, validation = "CV")
optimal_components <- which.min(RMSEP(pcr_model, estimate = "CV")$val[, , 1])
pcr_optimal_model <- pcr(beta_lactoglobulin_b ~ ., data = train_data_combined, ncomp = optimal_componen

#Predict Beta Lactoglobulin B values for the test set
test_spectra <- test_data[, 52:582]
test_beta_lactoglobulin_b <- test_data$beta_lactoglobulin_b
predicted_beta_lactoglobulin_b <- predict(pcr_optimal_model, newdata = test_spectra, ncomp = optimal_co
```

```r
#Calc RMSE
rmse <- sqrt(mean((predicted_beta_lactoglobulin_b - test_beta_lactoglobulin_b)^2))
cat("RMSE on the test set:", rmse, "\n")
```

```
## RMSE on the test set: 1.482513
```

```r
#plot of the predicted vs. actual Beta Lactoglobulin B levels
plot(test_beta_lactoglobulin_b, predicted_beta_lactoglobulin_b, xlab = "Actual Beta Lactoglobulin B", y
abline(a = 0, b = 1, col = "red", lwd = 2)
```



Imputing the observed mean here has definitly had an impact on the relationship between the predicted and actual values. Although far from perfect, the predicted values seem to perform better under these conditions, with a lower RMSE of 1.536409. Also note that there is a cluster centered around the red line, which may be a direct result to imputing the mean.

(c).