DATA605 Final Project: Characterizing the gene expression patterns in immune cells from healthy subjects

Daniel Stratis

2024-02-13

```
library(dplyr)
library(tidyr)
library(tibble)
library(matrixStats)
library(ggplot2)
library(ggpubr)
library(reshape2)
library(biomaRt)
library(Rtsne)
library(pheatmap)
```

INTRODUCTION

Gene = sequence of DNA controlling the expression of traits.

Gene expression = process of translating the information encoded by a gene into a gene product or functional protein unit.

The immune system is a complex network of cells, tissues, and organs that protect the body from infections, foreign substances, and diseases. It consists of two main parts: (1) the non-specific *innate* system, which acts as the bodies first line of defense including physical barriers like skin, defense mechanisms like mucous secretion, and inflammatory responses; (2) the specific *adaptive* system, which uses antigens from pathogens to strategically mount an immune response specific to that pathogen.

A more thorough understanding of the immune system can be revealed by analyzing immune cell genes and their expression patterns, which underly the physiological processes that define the bodies immune strategies. For this we need to measure gene expression extracted from immune cells in people.

Dataset

The data obtained contains gene expression counts measured in 8 different immune cell types from 13 healthy Singaporean individuals (127 samples total). Cell types include B/T cells, Dendiritic Cells (DC) (from the *adaptive* system); and Natural Killer (NK) cells, granulocytes, monocytes, progenitor cells, and Peripheral Blood Mononuclear Cells (PBMCs) (from the *innate* system).

This dataset was obtained from the National Library of Medicine (NCBI) Gene Expression Omnibus (GEO) database (Accession number: GSE107011 (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE107011)) under an open-license with no copyright restrictions (https://www.ncbi.nlm.nih.gov/geo/info/disclaimer.html). The data was generated by Monaco et al., 2019 (https://www.cell.com/cell-reports/fulltext/S2211-1247(19)30059-2?_returnURL=https%3A%2F%2Flinkinghub.elsevier.com%2Fretrieve%2Fpii%2FS2211124719300592%3Fshowall%3Dtrue#secsectitle0135) for their paper entitled “RNA-Seq Signatures Normalized by mRNA Abundance Allow Absolute Deconvolution of Human Immune Cell Types”.

The objective of my analysis is to conduct exploratory analyses using various visuals to uncover the gene expression patterns in immune cells that ultimately protect us from disease.

Guiding questions

These are my 5 guiding questions:

1. What is the distribution of gene expression counts for all samples?
2. How do the proportions of gene biotypes differ between cell types?
3. How similar are the subjects gene expression patterns to each other?
4. Do different cell types show unique expression patterns?
5. How are genes being co-expressed with each other?

ANALYSIS

Load data and cleaning

Read count data and metadata

There are 3 data frames important for this analysis:

1. Gene expression matrix (rows represent genes and columns represent samples)
2. Metadata (data describing the characteristics of each sample)
3. Gene biotypes (describe the biological type of each gene)

```
#gene read count matrix
read_counts = read.csv("Data/TPM_gene_counts.csv", header = TRUE, check.names = FALSE, sep = ",", row.names = 1)

#sample metadata
metadata = read.csv("Data/Metadata.csv", header = TRUE, sep = ",", row.names = 1)
metadata = mutate_all(metadata, as.factor) #convert variables to factors
metadata = data.frame(sample = rownames(metadata), metadata)

#check for matching sample dimensions and names in both data frames
if(all(colnames(read_counts) %in% rownames(metadata)) & all(colnames(read_counts) == rownames(metadata)) == TRUE)
{
  print("Column dimensions and names match.")
} else{
  print("Column dimensions and names do not match.")
}
```

```
## [1] "Column dimensions and names match."
```

```
#check for null or NA values in the data
if(sum(is.na(read_counts)) == 0){
  print("There are no missing values in the data frame")
} else{
  print("The data frame contains missing values")
  read_counts = na.omit(read_counts)
  print("The data frame contains missing values")
}
```

```
## [1] "There are no missing values in the data frame"
```

```
#format gene names
sample_names = colnames(read_counts)
gene_names = sub("\\..*$", "", rownames(read_counts)) #remove version numbers from genes
read_counts = data.frame(gene_name = gene_names, read_counts)

#remove duplicate genes
duplicate_genes = duplicated(read_counts$gene_name)
read_counts = read_counts[!duplicate_genes, ]
rownames(read_counts) = sub("\\..*$", "", rownames(read_counts))
```

Gene names and biotypes

```
#load gene names and biotypes
gene_biotypes = read.csv("Data/Gene_biotypes.csv", header = TRUE, sep = ",")
```

```
#merge gene biotypes
read_counts = right_join(read_counts, gene_biotypes, by = 'gene_name')

#convert to long table format
read_counts_long = pivot_longer(read_counts, cols = colnames(read_counts)[-c(1,129,130)], names_to = 'sample', va
lues_to = 'TPM_read_counts')

#merge metadata
read_counts_long = left_join(read_counts_long, metadata, by = 'sample')

#clean read count matrix
rownames(read_counts) = read_counts$gene_name
read_counts = read_counts[, !names(read_counts) %in% c('gene_name','hgnc','biotype')]
colnames(read_counts) = sample_names
```

Question 1: Distribution of gene expression counts

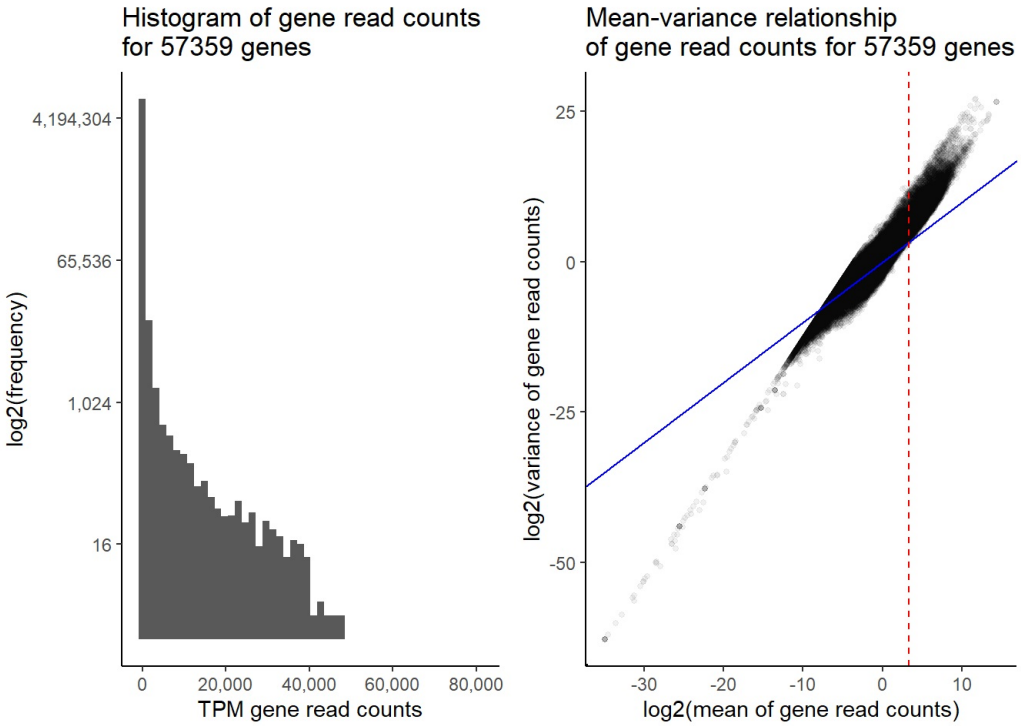
```
#plot the histogram of gene read counts
n_genes = nrow(read_counts)
read_count_hist = ggplot(data = read_counts_long, aes(x = TPM_read_counts)) +
    geom_histogram(bins = 50) +
    labs(x = "TPM gene read counts",
         y = "log2(frequency)",
         title = paste0("Histogram of gene read counts\nfor ", n_genes, " genes")) +
  scale_x_continuous(labels = scales::comma) +
  scale_y_continuous(labels = scales::comma, trans = 'log2') +
    theme_classic()
```

```
#calculate the mean and variance for each gene
gene_means = rowMeans(read_counts)
gene_vars = rowVars(as.matrix(read_counts), useNames = TRUE)

gene_mean_var_df = data.frame(means = gene_means, vars = gene_vars)
```

```
#gene read count mean-variance scatterplot
count_cutoff = 10
mean_var_plot = ggplot(data = gene_mean_var_df, aes(x = log2(means), y = log2(vars))) +
    geom_point(size = 1, alpha = 0.05) +
    geom_abline(intercept = 0, slope = 1, color = "blue", linetype = 'solid') +
  geom_vline(xintercept = log2(count_cutoff), color = "red", linetype = 'dashed') +
    labs(x = "log2(mean of gene read counts)", y = "log2(variance of gene read counts)",
         title = paste0("Mean-variance relationship\nof gene read counts for ", n_genes, " genes")) +
    theme_classic()
```

```
#display the plots together
ggarrange(nrow = 1, ncol = 2, read_count_hist, mean_var_plot)
```




```
#remove genes below the read count cutoff
read_counts = read_counts[rowMeans(read_counts[]) > count_cutoff,]
print(paste0("There are ", nrow(read_counts), " genes above the read count cutoff"))
```

```
## [1] "There are 10205 genes above the read count cutoff"
```

The histogram of gene expression counts for **57,359 genes** are not normally distributed and show evidence for a poisson or binomial distribution. However, given the scatter plot where each point represents a gene, there is a heteroscedastic or overdispersed relationship in the data (ie. the variance becomes greater with an increasing mean). Therefore, gene expression counts can be modeled by a negative binomial (NB) distribution.

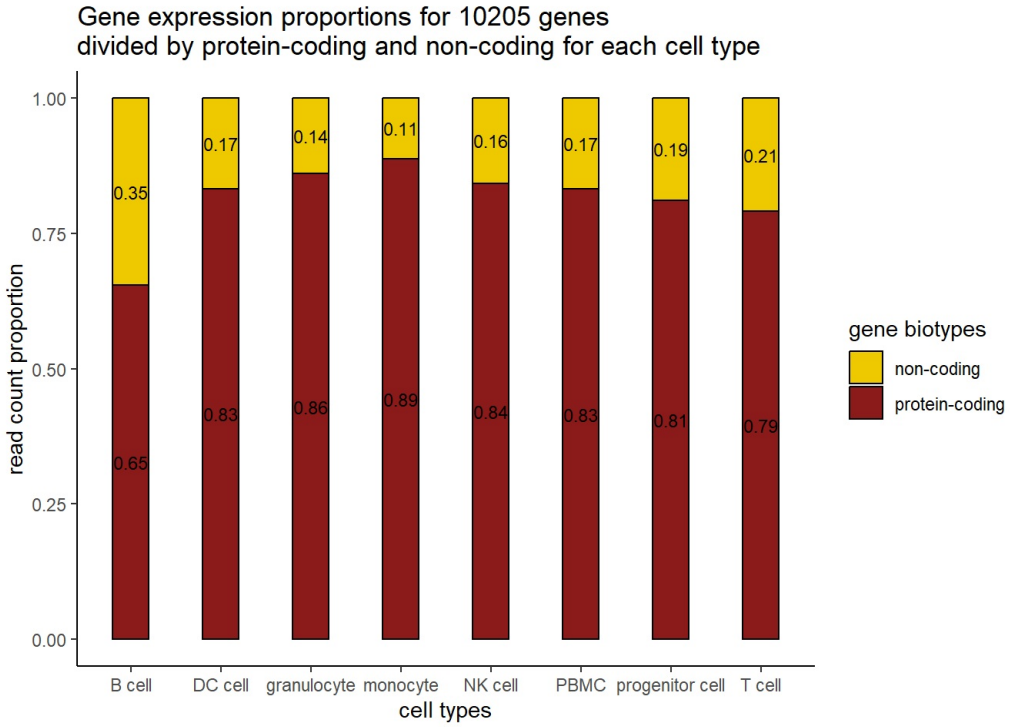
Additionally, because lower gene expression counts show greater dispersion (low signal to noise ratios), these genes must be removed. Any gene with an average gene expression count across all samples less than 10 were removed (indicated by the dashed red line in the scatter plot). This resulted in **10,205 genes** to be further analyzed.

Question 2: Gene expression expression variability between gene biotypes

```
#sum the read counts for each gene biotype in each cell type
biotype_counts = aggregate(TPM_read_counts ~ biotype + cell_group, data = read_counts_long, FUN = sum)
biotype_counts$biotype_group = ifelse(biotype_counts$biotype == "protein_coding", yes = "protein_coding", no = "n
on_coding")
biotype_counts = aggregate(TPM_read_counts ~ biotype_group + cell_group, data = biotype_counts, FUN = sum)
```

```
#calculate the biotype proportion for each cell type
biotype_counts <- biotype_counts %>%
    group_by(cell_group) %>%
    mutate(biotype_perc = TPM_read_counts/sum(TPM_read_counts))
```

```
ggplot(data = biotype_counts, aes(x = cell_group, y = biotype_perc,
                                  fill = biotype_group, label = round(biotype_perc, 2))) +
  geom_col(position = position_stack(), width = 0.4, color = 'black') +
  geom_text(size = 3, position = position_stack(vjust = 0.5)) +
  labs(x= 'cell types', y = 'read count proportion',
       title = paste0("Gene expression proportions for ", nrow(read_counts), " genes\ndivided by protein-coding a
nd non-coding for each cell type")) +
  scale_fill_manual(name = 'gene biotypes',
                    labels = c('non-coding','protein-coding'),
                    values = c('gold2','firebrick4')) +
  theme_classic()
```



Not all genes encode a protein product, in fact most genes in the human genome encode for a molecule similar to DNA called RNA, which play major roles in regulatory processes in the cell. However, the study from which this data was generated used an extraction method that's bias toward measuring protein-coding genes. This is why the majority of gene expression counts are coming from protein-coding genes for all cell types. Interestingly, B cells seem to contain a significantly smaller proportion of gene expression counts from protein-coding genes compared to non-coding genes. This would indicate that this cell type is characterized by more regulatory processes rather than protein products.

Because of the biased method of gene expression measures, the data was further subsetted to only include protein-coding genes. This resulted in **8,484 protein-coding** genes left for further analysis.

Question 3: Gene expression variability between subjects

```
#subset to protein-coding genes
tmp = rownames_to_column(read_counts, var = 'gene_name')
tmp = left_join(tmp, gene_biotypes, by = 'gene_name')
tmp = filter(tmp, biotype == 'protein_coding')
tmp = column_to_rownames(tmp, var = 'gene_name')
read_counts = tmp[, !names(tmp) %in% c('hgnc','biotype')]
```

```
#subset to PBMC samples only
PBMC_read_counts = read_counts[ ,grepl('PBMC', colnames(read_counts))]
```

```
#calculate the correlation matrix
cor_mat = cor(PBMC_read_counts,
    use = "complete.obs",
    method = "spearman")
```

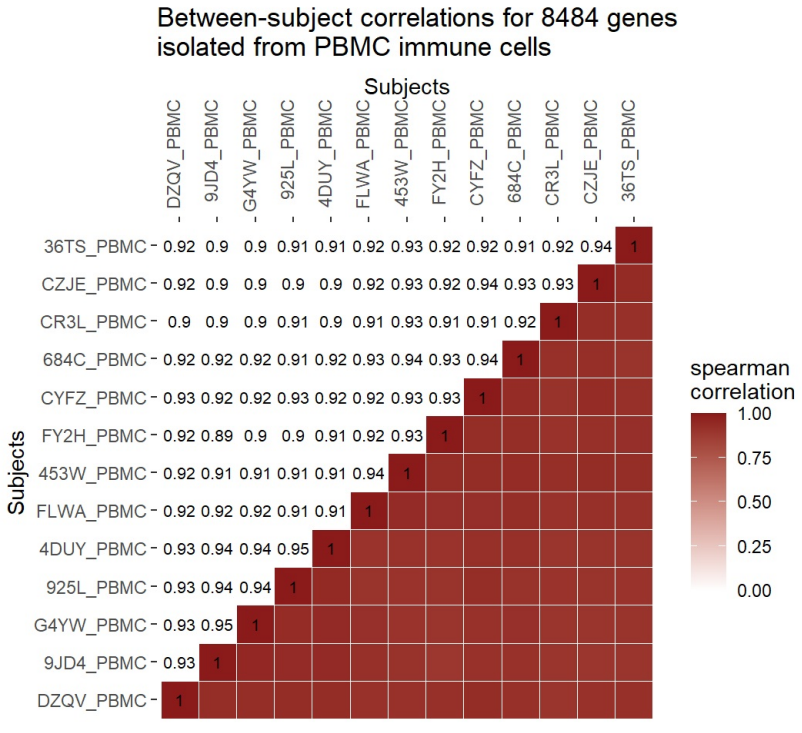
```
#define the functions to plot a symmetrical heatmap
get_upper_tri = function(cor_mat){ #function for upper triangle of matrix
    cor_mat[upper.tri(cor_mat)] = NA
    return(cor_mat)
}

get_lower_tri = function(cor_mat){ #create function for lower triangle of matrix
    cor_mat[lower.tri(cor_mat)] = NA
    return(cor_mat)
}

reorder = function(cor_mat){ #create function to hierarchically cluster the matrix
  dd = as.dist((1 - cor_mat) / 2)
  hc = hclust(dd)
  cor_mat = cor_mat[hc$order, hc$order]
}

cor_mat = reorder(cor_mat) #cluster matrix
upper_tri = get_upper_tri(cor_mat) #define upper matrix triangle
lower_tri = get_lower_tri(cor_mat) #define lower matrix triangle
meltNum = melt(upper_tri, na.rm = TRUE) #reshape upper matrix triangle
meltColor = melt(lower_tri, na.rm = TRUE) #reshape lower matrix triangle
```

```
#plot the heatmap
ggplot() +
    labs(x = "Subjects", y = "Subjects",
         title = paste0("Between-subject correlations for ", nrow(read_counts),
                        " genes\nisolated from PBMC immune cells")) +
    geom_tile(data = meltColor, color = "white",
              mapping = aes(Var2, Var1, fill = value)) +
    geom_text(data = meltNum, size = 2.75,
              mapping = aes(Var2, Var1, label = round(value, digit = 2))) +
    scale_x_discrete(position = "top") +
    scale_fill_gradient(low = "white", high = "firebrick4",
                        limit = c(0,1), name = "spearman\ncorrelation") +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.background = element_blank(),
          axis.text.x = element_text(angle = 90, hjust = 1)) +
    coord_fixed()
```



```
print(paste0("The average correlation coefficient is ", round(mean(filter(meltColor, value != 1)$value), 2)))
```

```
## [1] "The average correlation coefficient is 0.92"
```

For this question, the data was subsetted to include only samples coming from Peripheral Blood Mononuclear Cell (PBMC) types. This is because PBMCs are any white blood cell containing a round nucleus and describe a broad range of different cell types including T, B, NK, DC cells, and monocytes. Therefore, they provide a strong representation for the gene expression counts for all cell types in this data.

Given the average correlation coefficient of gene expression counts between subjects is 0.92, this demonstrated similarity or homogeneity between healthy subjects immune systems at the gene expression level.

Question 4: Gene expression variability between cell types

Principle Component Analysis (PCA)

```
#calculate the PC scores for each sample while scaling the data
pca = prcomp(t(read_counts), scale = TRUE)
var = pca$sdev^2 #obtain the variability explained by each PC
per = round(var / sum(var) * 100, 1) #convert to percentages
pca_df = data.frame(metadata, PC1 = pca$x[,1], PC2 = pca$x[,2])
```

```
#create the PCA plot
pca_plot = ggplot(data = pca_df, aes(x = PC1, y = PC2, col = cell_group)) +
  geom_point(size = 2, show.legend = FALSE) +
    labs(color = 'cell types',
        x = paste('PC1 - ', per[1], '%', sep = ''),
        y = paste('PC2 - ', per[2], '%', sep = '')) +
    theme_classic()
```

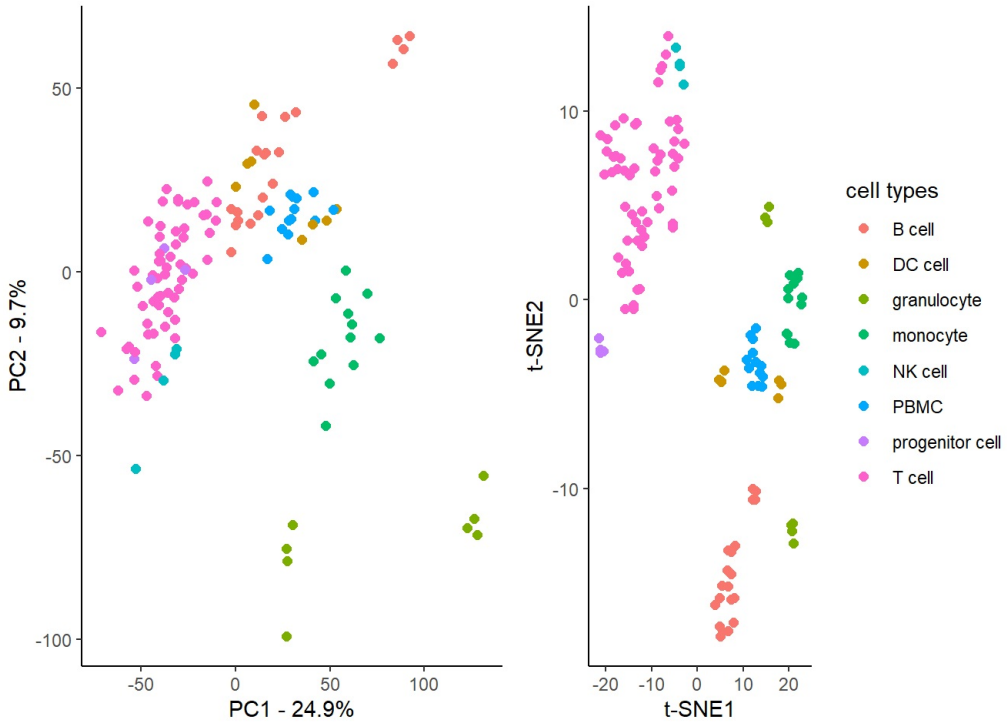
t-distributed Stochastic Neighbor Embedding (t-SNE)

```
scaled_read_counts = t(scale(t(read_counts)))
```

```
set.seed(123) #set the seed to obtain the same results (t-SNE is stochastic)
tsne = Rtsne(as.matrix(t(scaled_read_counts)), dims = 2, perplexity = 12) #use the t-SNE algorithm on samples
tsne_df = data.frame(metadata, tSNE1 = tsne$Y[ ,1], tSNE2 = tsne$Y[ ,2])
```

```
#create the t-SNE plot
tsne_plot = ggplot(data = tsne_df, aes(x = tSNE1, y = tSNE2, col = cell_group)) +
  geom_point(size = 2) +
    labs(color = 'cell types',
        x = "t-SNE1",
        y = "t-SNE2") +
    theme_classic()
```

```
#display the plots together
ggarrange(nrow = 1, ncol = 2, pca_plot, tsne_plot)
```



Both dimensionality reduction methods of PCA and t-SNE show distinct clustering of samples based on cell type. This suggests that the immune cell type is a significant explanatory variable for the gene expression counts and that different cell types show different gene expression patterns.

Question 5: Gene co-expression of T and B cells

For a more focused and simplified analysis of gene co-expression, the data was subsetted to only samples from T and B cell types.

```
#log2 transform the gene expression counts
log2_read_counts = log2(read_counts + 1)

#subset to T and B cell types
annot_col = filter(metadata, cell_group %in% c('T cell','B cell'))[,c('sample','cell_group')]
sample_names = rownames(annot_col)
log2_read_counts = log2_read_counts[,sample_names]
```

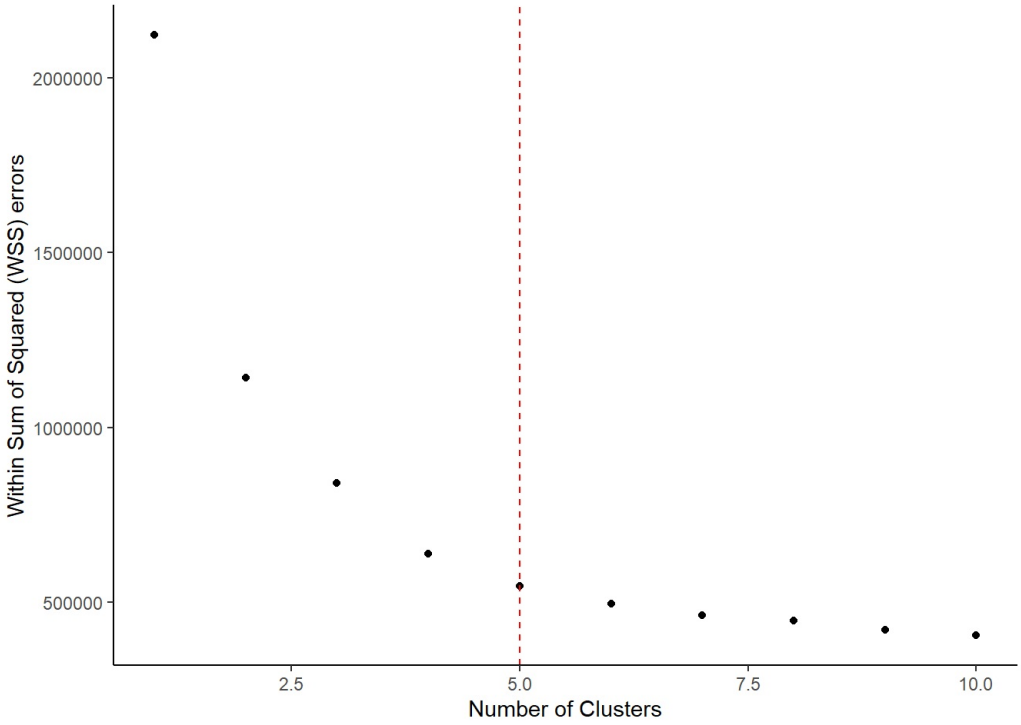
K-means clustering of genes

```
#initial within sum-squared error (WSS) for one cluster
wss = (nrow(log2_read_counts)) * sum(apply(log2_read_counts, 2, var))

#calculate WSS as a function of number of clusters
for(i in 2:10){
  wss[i] = sum(kmeans(log2_read_counts, centers = i)$withinss)
}
cluster <- 1:10
wss_df <- data.frame(cluster, wss)
```

```
#define the number of clusters
k = 5

#create the eblow plot
ggplot(data = wss_df, #create elbow plot
    aes(x = cluster, y = wss)) +
    geom_point() +
    labs(y = "Within Sum of Squared (WSS) errors",
         x = "Number of Clusters") +
    geom_vline(xintercept = k, colour = "red", linetype = 'dashed') +
    theme_classic()
```



Based on the elbow plot generated from the k-means clustering algorithm, the optimal number of gene clusters is 5.

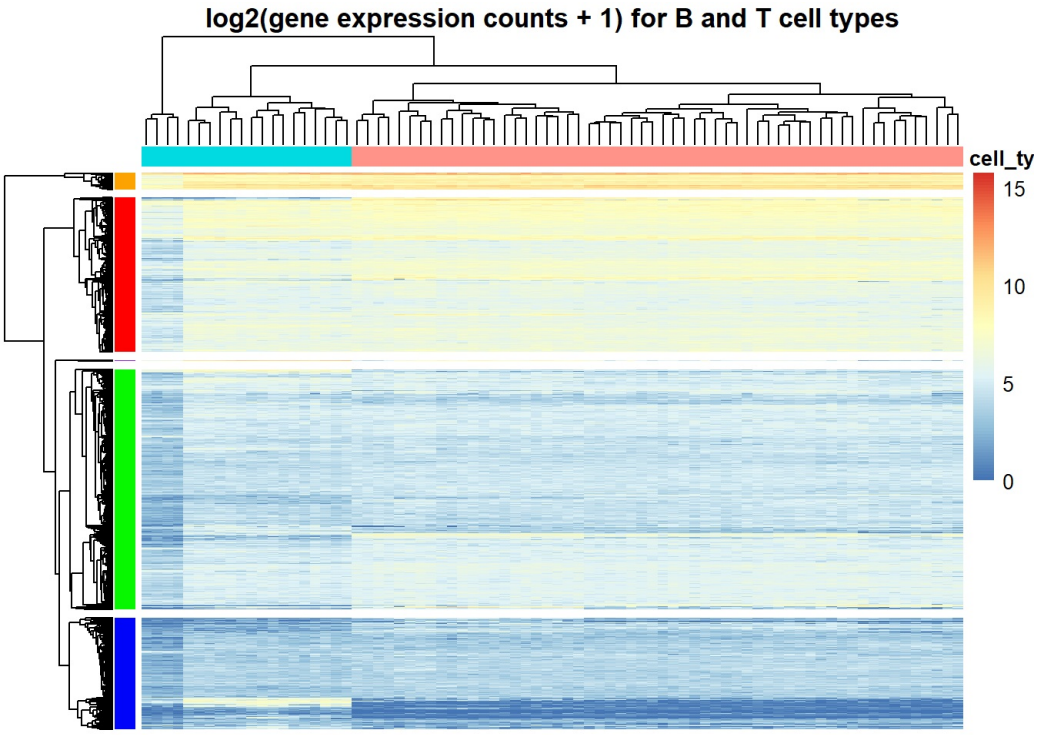
Hierarchical clustering of gene expression counts

```
#define annotations for clustered genes
hc = hclust(dist(log2_read_counts))
clusters = cutree(hc, k)
annot_row = data.frame(cluster = clusters)

#define the cluster colors
annot_colors = list(cluster = c('red','green','blue','orange','purple'))
```

```
#define annotations for samples
annot_col = data.frame(cell_type = annot_col[,'cell_group'])
rownames(annot_col) = sample_names
```

```
#create heatmap
pheatmap(log2_read_counts,
         scale = 'none',
         cluster_cols = TRUE,
         cluster_rows = TRUE,
         clustering_distance_cols = 'euclidean',
         clustering_distance_rows = 'euclidean',
         clustering_method = 'complete',
         cutree_rows = k,
         annotation_col = annot_col,
         annotation_row = annot_row,
         annotation_colors = annot_colors,
         annotation_legend = FALSE,
         annotation_names_row = FALSE,
         show_colnames = FALSE,
         show_rownames = FALSE,
         border_color = 'NA',
         main = "log2(gene expression counts + 1) for B and T cell types")
```




```
#gene cluster sizes
table(annot_row)
```

```
## cluster
##    1    2    3    4    5
## 2503 3893 1810  264   14
```

Hierarchical clustering of samples has separated B (light blue) and T (light red) cell types, a consistent result with PCA and t-SNE methods.

Hierarchical clustering of genes resulted in 5 gene clusters showing strong co-expression for each cluster. Each gene cluster shows a unique profile of expression patterns across all samples with some exceptions between cell types. For instance, a small subset in the top of the red gene cluster show lower expression in B cells compared to T cells. Conversely, genes in the purple cluster show higher expression in B cells compared to T cells.

Further analyses would be required for a closer look at the individual genes including a gene differential expression analysis to identify genes with significant differences in expression between cell types. Additionally, functional enrichment analyses would reveal broad Gene Ontology terms which could describe the biological functions of each gene cluster. More analysis is required but is beyond the scope of this project.

CONCLUSION

1. Gene expression counts can be modeled using a negative binomial distribution.
2. The majority of gene expression counts were measured in protein-coding genes. B cells showed a higher proportion of non-coding genes.
3. The high correlations of gene expression counts between subjects reveal homogeneity of the immune systems among healthy subjects at the gene expression level.
4. Immune cell types are a significant explanatory variable for gene expression counts.
5. The 5 gene clusters reveal co-expression patterns between genes with some differences between B and T cell types that require further investigation.

REFERENCES

Monaco et al., 2019. *RNA-seq signatures normalized by mrna abundance allow absolute deconvolution of human immune cell types*. Cell Reports, 26(6). https://doi.org/10.1016/j.celrep.2019.01.041 (https://doi.org/10.1016/j.celrep.2019.01.041)