

My_Graphs Part 2

2024-08-07

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.4      v readr      2.1.5
## v forcats    1.0.0      v stringr   1.5.1
## v ggplot2    3.5.1      v tibble    3.2.1
## v lubridate  1.9.3      v tidyr     1.3.1
## v purrr      1.0.2
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
#cannot call within function, must call outside
```

```
gene_express_df <- read.csv("C:/Users/casha/Downloads/QBS103_GSE157103_genes.csv", row.names = 1)
metadata_df <- read.csv("C:/Users/casha/Downloads/QBS103_GSE157103_series_matrix.csv")
```

```
#establish function per parameters
```

```
#https://www.dataquest.io/blog/write-functions-in-r/
```

```
gene_plots <- function(data_frame, name_genes, continuous_covariate, categorical_covariate_one, categorical_covariate_two)
```

```
# How to create a for loop in R-Studio:https://www.geeksforgeeks.org/for-loop-in-r/
```

```
for (gene in name_genes) {
  gene_expression <- data_frame %>%
    rownames_to_column("gene") %>% #rows to columns
    filter(gene == !!gene) %>% #ONLY INCLUDE what matches the gene we're asking for
}
```

```
#wide format to long format
```

```
pivot_longer(cols = -gene, names_to = "participant_id", values_to = "expression") %>%
  select(-gene)
```

```
# Merge expression data with metadata
```

```
merged_df <- gene_expression %>%
```

```
#https://www.datacamp.com/tutorial/merging-data-r: How to merge 2 datasets in R
```

```
merge(metadata_df, by = "participant_id")
```

```
# Histogram
```

```
histogram <- ggplot(merged_df, aes(x = expression)) + #define histogram
```

```
#https://www.datacamp.com/tutorial/make-histogram-basic-r
```

```
geom_histogram(binwidth = 0.1, fill = "pink", color = "black") + #binwidth is how wide we want each bin
labs(
```

```

    title = paste("Histogram of", gene, "Gene Expression"),
    x = "Gene Expression",
    y = "Count" #number which falls within THIS level of gene expression
  ) +
  theme(
    plot.background = element_rect(fill = "black"),
    panel.background = element_rect(fill = "black"),
    text = element_text(size = 12, color = "white"),
    plot.title = element_text(hjust = 0.5, size = 16, color = "white"),
    axis.text = element_text(color = "white") #changes the color of the numbers on gridlines
  )

  print(histogram)

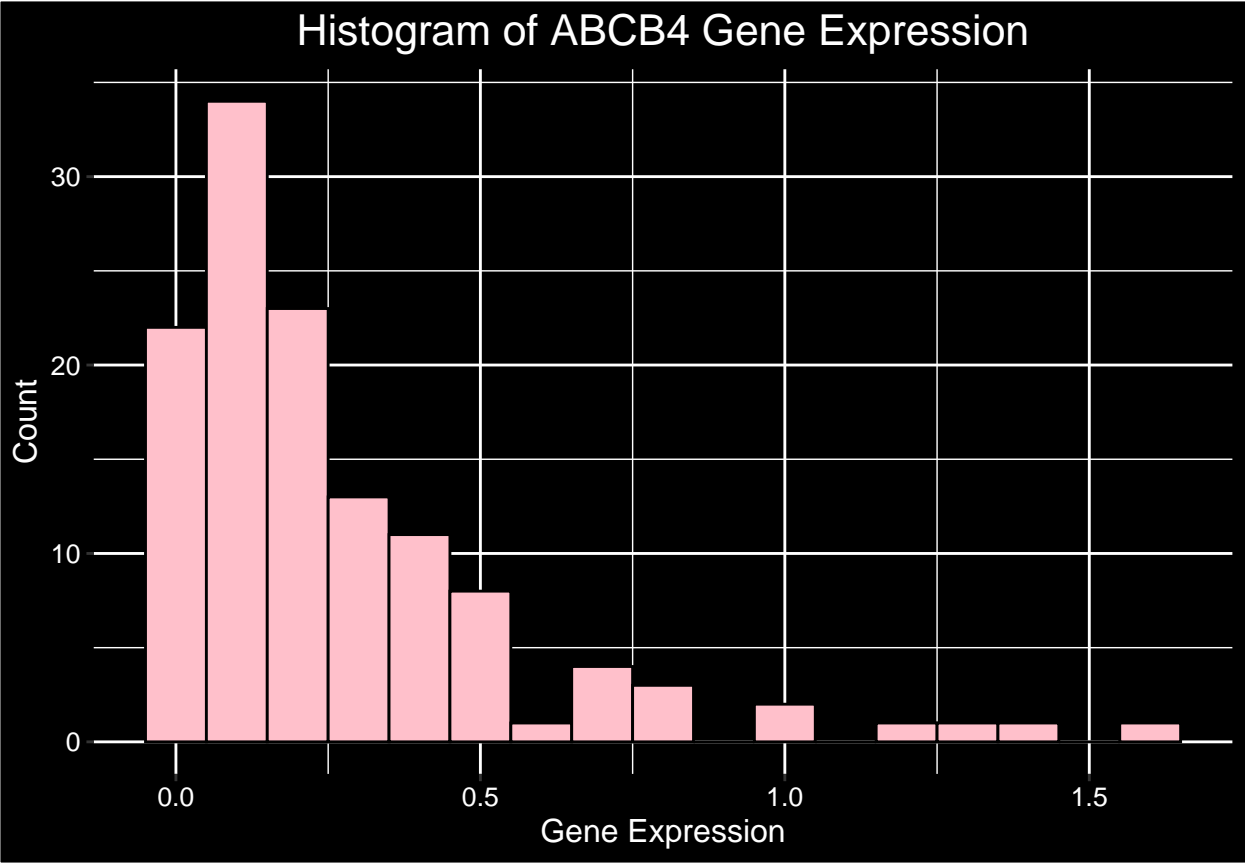
# Boxplot
  boxplot <- ggplot(merged_df, aes(x = .data[[categorical_covariate_one]], y = expression, fill = .data[[categorical_covariate_one]])) +
    geom_boxplot() +
#Define colors: Have to use three colors because sex is female, male, and unknown
    scale_fill_manual(values = c('darkgreen', 'grey', 'yellow')) +
    labs(
      title = paste("Boxplot of", gene, "Gene Expression by", categorical_covariate_one, "and", categorical_covariate_two),
      x = categorical_covariate_one,
      y = "Gene Expression"
    ) +
    theme(
      text = element_text(size = 12),
      plot.title = element_text(hjust = 0.5, face = "bold", size = 14),
      axis.title = element_text(face = "bold")
    )

  print(boxplot)
}
#MAKE SURE TO CLOSE BRACKET OR FUNCTION WILL NOT RUN

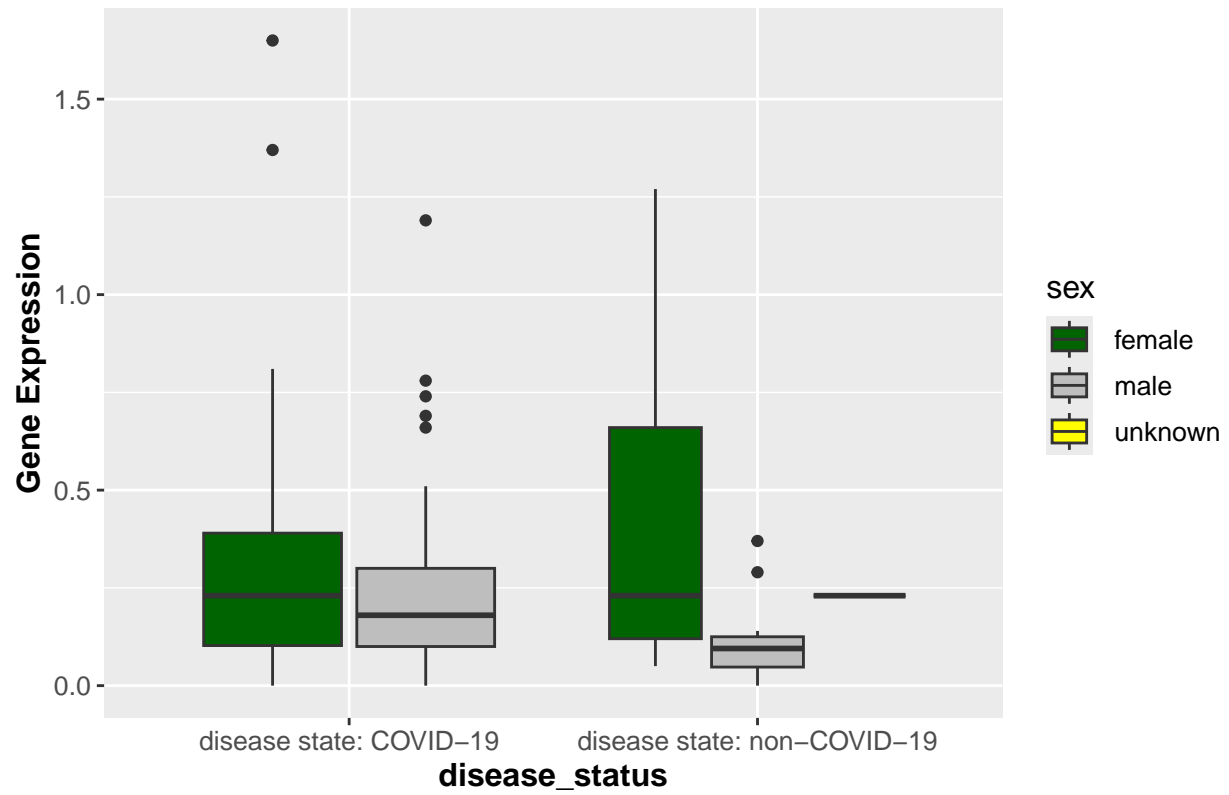
# Define the genes to plot
genes_want <- c("ABCB4", "AAR2", "ABHD1")

# Call the function using original form of: "data_frame, name_genes, continuous_covariate, categorical_covariate, gene_plots"
gene_plots(gene_express_df, genes_want, "age", "disease_status", "sex")

```

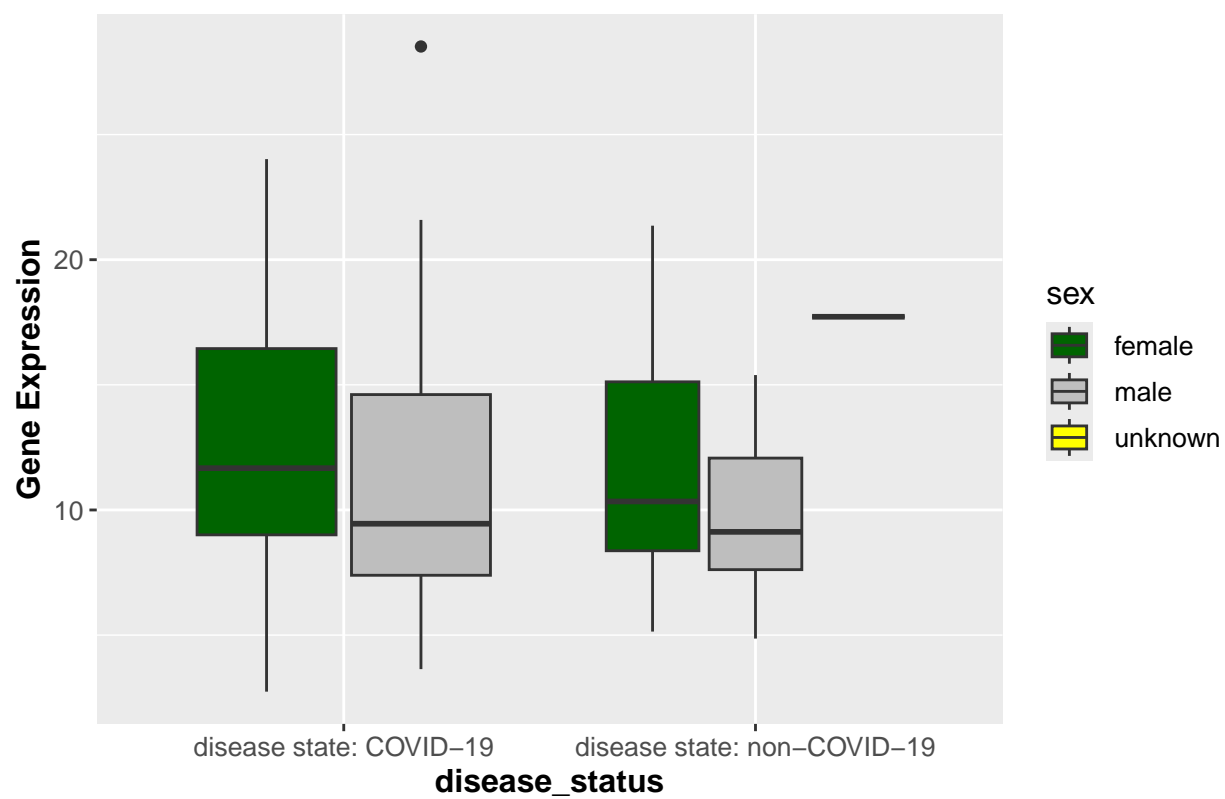


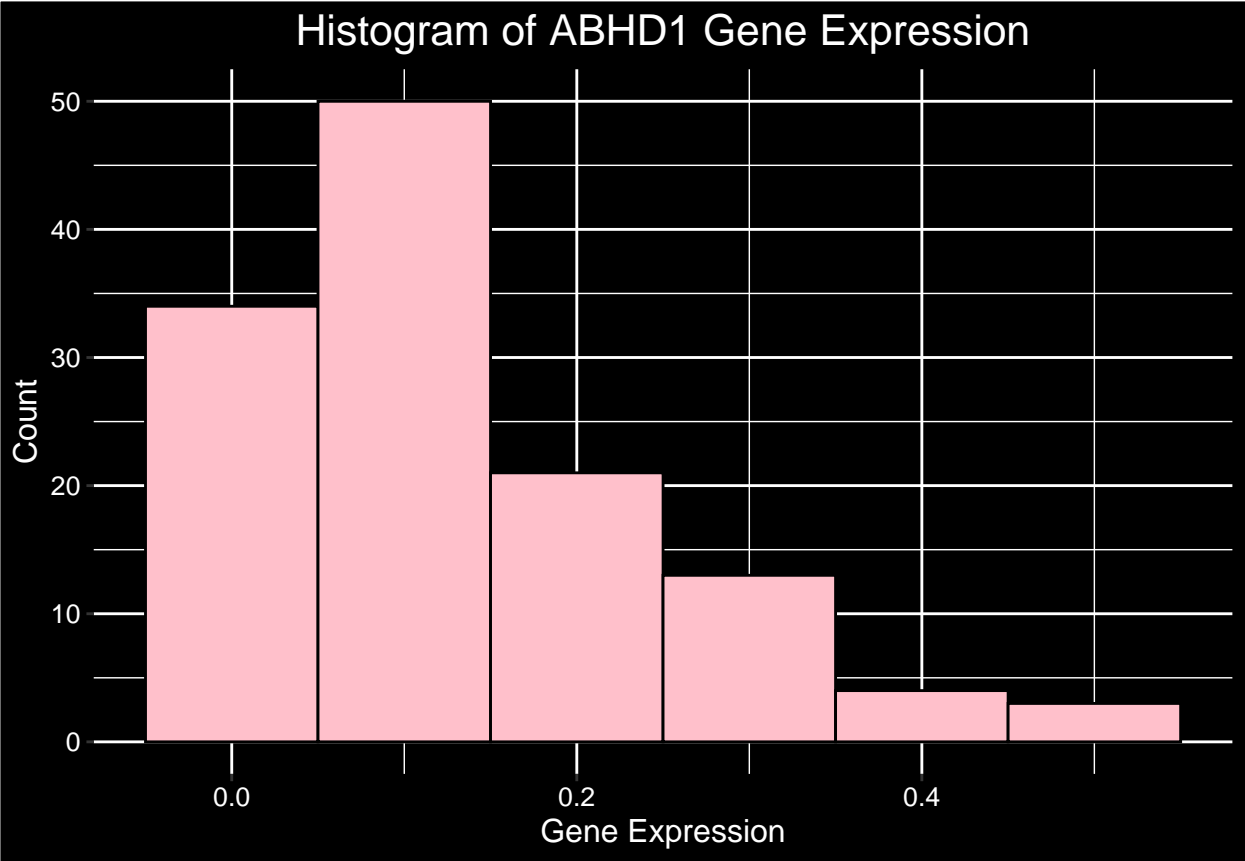
Boxplot of ABCB4 Gene Expression by disease_status and sex





Boxplot of AAR2 Gene Expression by disease_status and sex





Boxplot of ABHD1 Gene Expression by disease_status and sex

