Submission 2

Sabin Hart

2024-08-06

Library Imports

```
## -- 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   3.4.4     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 error
```

Import, clean, and combine data

```
gene_data_raw <- read_csv("./final-data/QBS103_GSE157103_genes.csv")
```

```
## New names:
## Rows: 100 Columns: 127
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): ...1 dbl (126): COVID_01_39y_male_NonICU, COVID_02_63y_male_NonICU,
## COVID_03_33y_...
## i Use ‘spec()‘ to retrieve the full column specification for this data. i
## Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
## * ‘‘ -> ‘...1‘
```

```
pheno_data_raw <- read_csv("./final-data/QBS103_GSE157103_series_matrix.csv")
```

```
## Rows: 126 Columns: 25
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (21): participant_id, geo_accession, status, !Sample_submission_date, la...
## dbl  (4): channel_count, charlson_score, ventilator-free_days, hospital-free...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```r
# flipping gene data
gene_data <- gene_data_raw %>%
  rename("gene_name" = 1) %>%
  column_to_rownames("gene_name") %>%
  t() %>%
  data.frame() %>%
  rownames_to_column("participant_id")

# combine data
cleaned_data <- gene_data %>%
  merge(pheno_data_raw, by='participant_id') %>%
  mutate(participant_id = as.numeric(sub(".*_(\\d+)_.*", "\\1", participant_id))) %>%
  mutate(participant_id = case_when(
    grepl('non', disease_status, ignore.case = TRUE) ~ participant_id + 200,
    TRUE ~ participant_id
  )) %>%
  arrange(participant_id)

rm(gene_data, gene_data_raw, pheno_data_raw)
```

Function 1: Histogram

```r
create_histo <- function(df, genes) {

  # create list for returning all plots
  histogram_list = list()

  # loop over provided genes
  for (gene in genes) {
    data <- df %>% select({{gene}})  # curly braces to unpack variable name

    # create plot
    histo <- ggplot(data = data, aes_string(x = gene)) +
    geom_histogram(fill = '#7f7da2', bins = 12) +
    geom_vline(xintercept = mean(data[[gene]]), linetype = 'dashed', color = 'darkslategrey') +
    annotate("text", x = mean(data[[gene]])+0.5*sd(data[[gene]]), y = Inf,
             color = 'darkslategrey', label = 'Mean expression', vjust = 1.5) +
    theme_minimal() +
    labs(title = "Gene Expression Histogram",
         x = paste0("Expression of ", gene),
         y = "Number of Samples") +
    theme(plot.title = element_text(hjust = 0.5))

    histogram_list <- list.append(histogram_list, histo)
  }

  return (histogram_list)
}
```

Function 2: Scatter Plot

```r
create_scatter <- function(df, genes, con_cov) {

  # create list for returning all plots
  scatter_list = list()

  # loop over provided genes
  for (gene in genes) {

    # prep data
    df$covariate = suppressWarnings(as.numeric(df[[con_cov]]))
    data <- df %>%
      filter(!is.na(covariate)) %>%
      select({{gene}}, covariate)


    # create plot
    scatter <- ggplot(data = data, aes_string(x=gene, y = 'covariate')) +
      geom_point() +
      geom_smooth(method='lm', formula= y~x, color = 'darkslategray') +
      theme_minimal() +
      labs(title = paste0(con_cov," versus Gene Expression"),
           x = paste0("Expression of ", gene),
           y = con_cov,
           caption = "Line of best fit and standard error") +
      theme(plot.title = element_text(hjust = 0.5))

    scatter_list <- list.append(scatter_list, scatter)
  }

  return (scatter_list)
}
```

Function 3: Box plot

```r
create_boxplot <- function(df, genes, cat_cov1, cat_cov2) {

  # create list for returning all plots
  boxplot_list = list()

  # loop over provided genes
  for (gene in genes) {

    # prep data
    data <- df %>%
      select({{gene}}, {{cat_cov1}}, {{cat_cov2}}) %>%
      na.omit()


    # create plot
    boxplot <- ggplot(data = data,
                      aes_string(y=gene, x = cat_cov1, color = cat_cov2)) +
      geom_boxplot() +
      theme_minimal() +
      labs(title = paste0("Gene Expression by ", cat_cov1, " and ", cat_cov2),
           y = paste0("Expression of ", gene)) +
      theme(plot.title = element_text(hjust = 0.5)) +
      scale_colour_brewer(gsub('_', ' ', cat_cov2), palette = "Set1") +
      scale_x_discrete(gsub('_', ' ', cat_cov1))

    boxplot_list <- list.append(boxplot_list, boxplot)
  }

  return (boxplot_list)
}
```

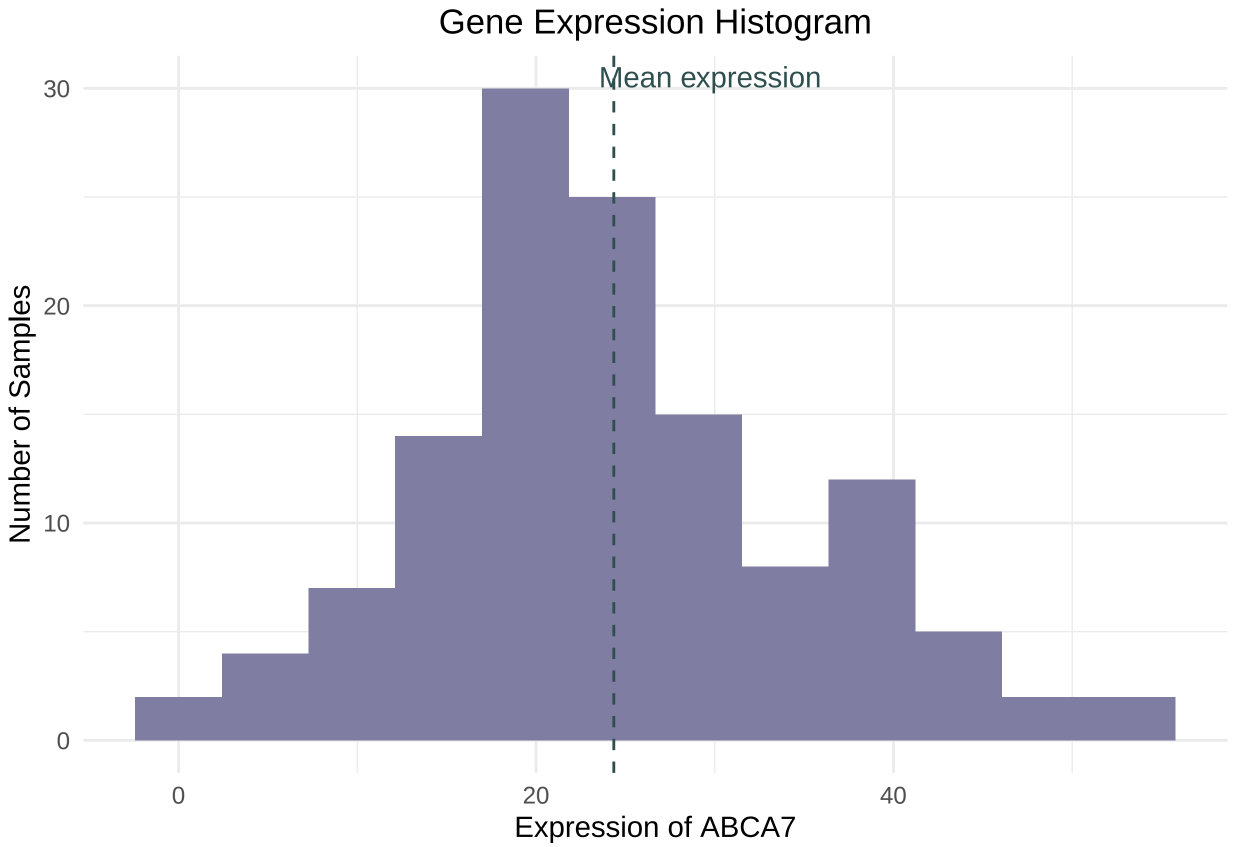
Implementation

```
# list of three genes, first is my original (ABCA7)
gene_list <- list("ABCA7", "AASS", "ABAT")


# histograms
histograms <- create_histo(cleaned_data, gene_list)
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

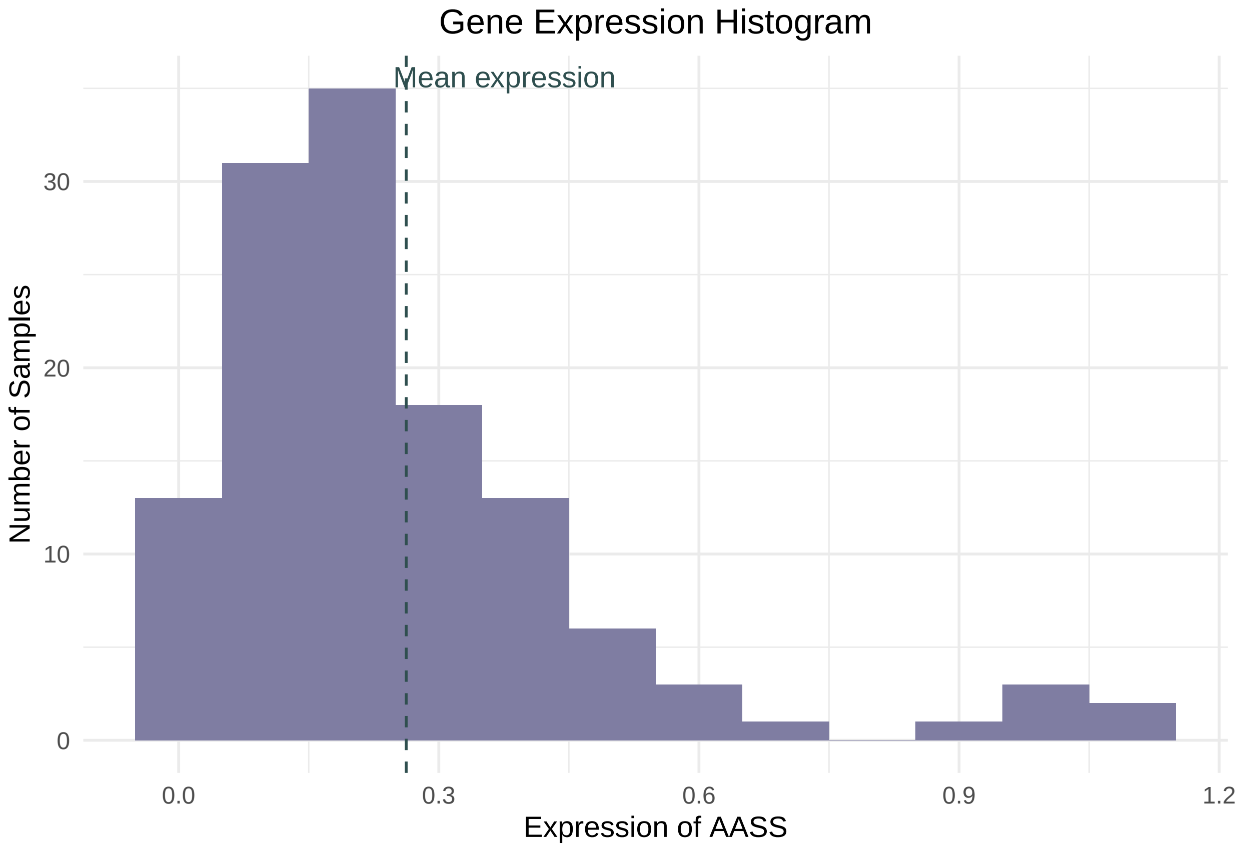
```
histograms[1]
```

```
## [[1]]
```



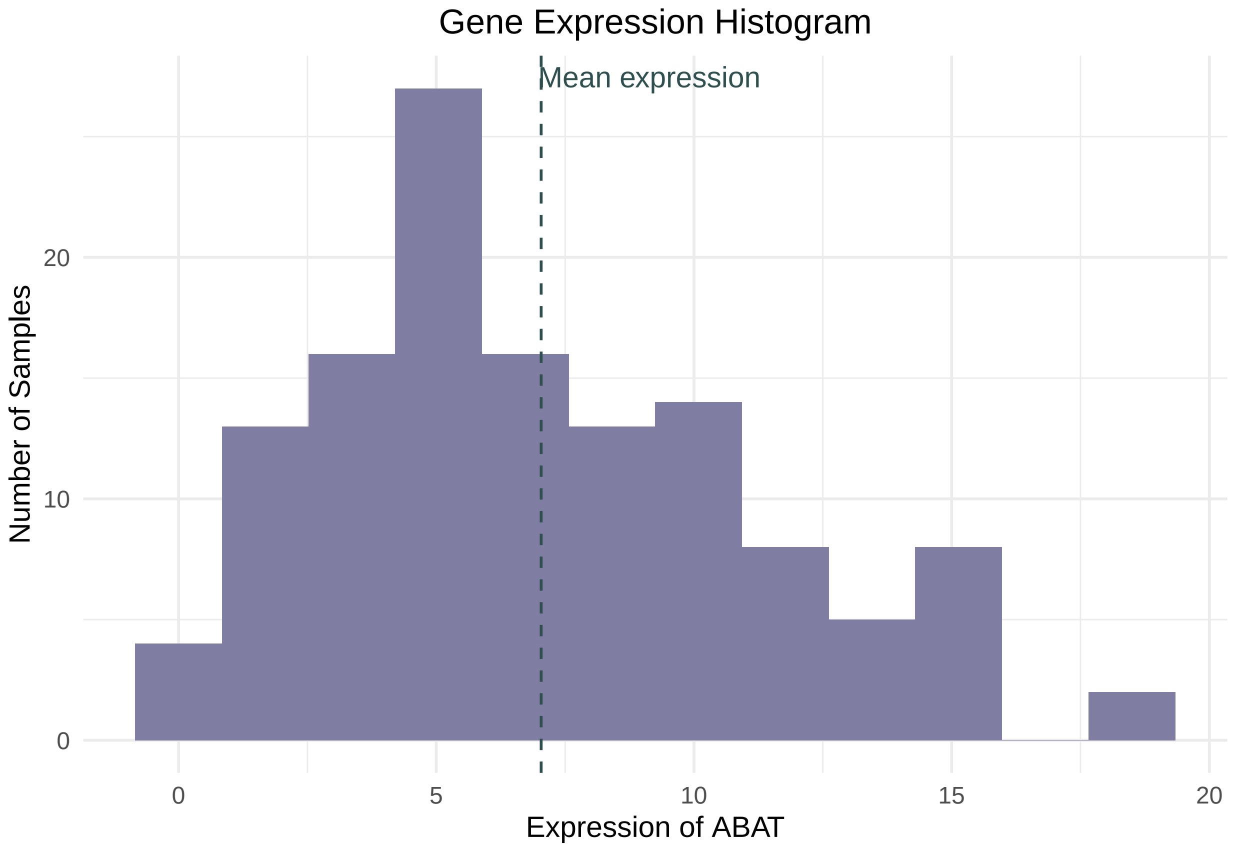
```
histograms[2]
```

```
## [[1]]
```



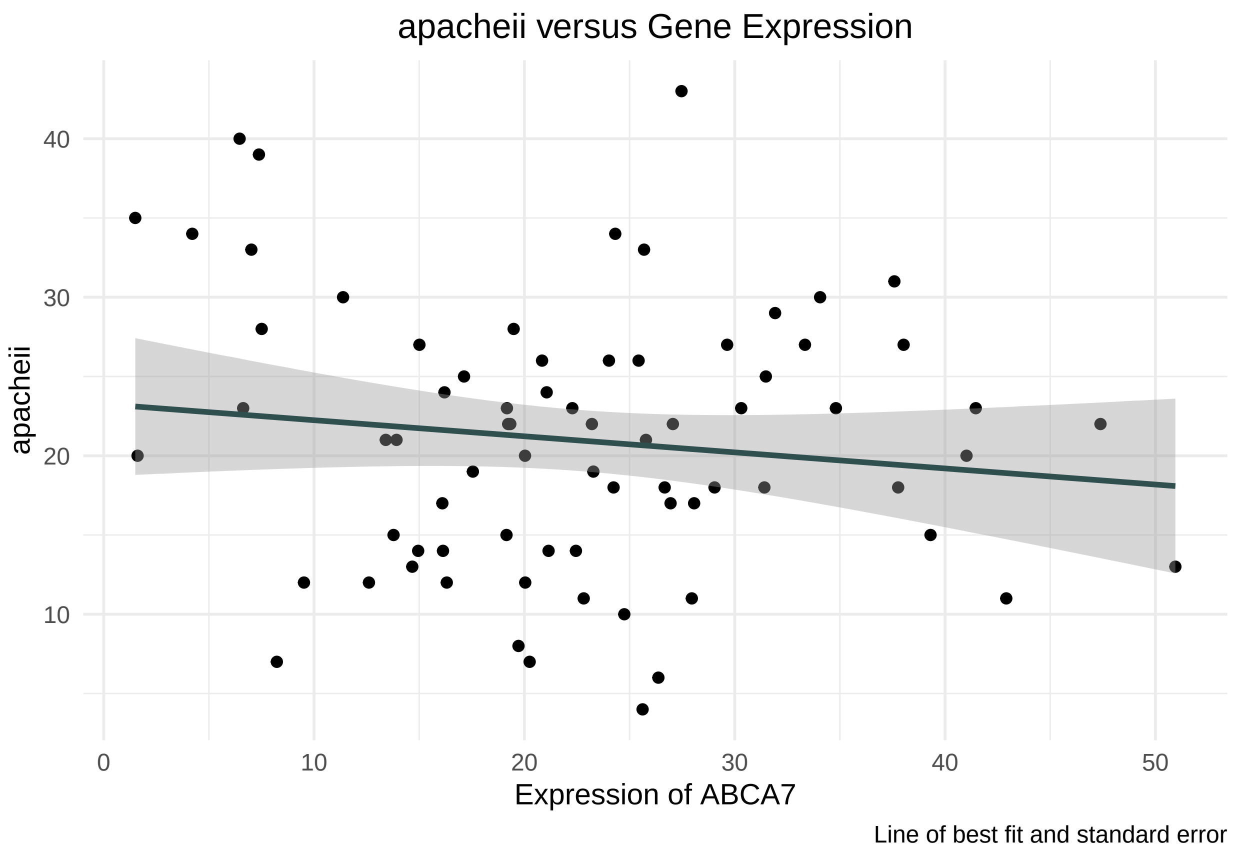
```
histograms[3]
```

```
## [[1]]
```



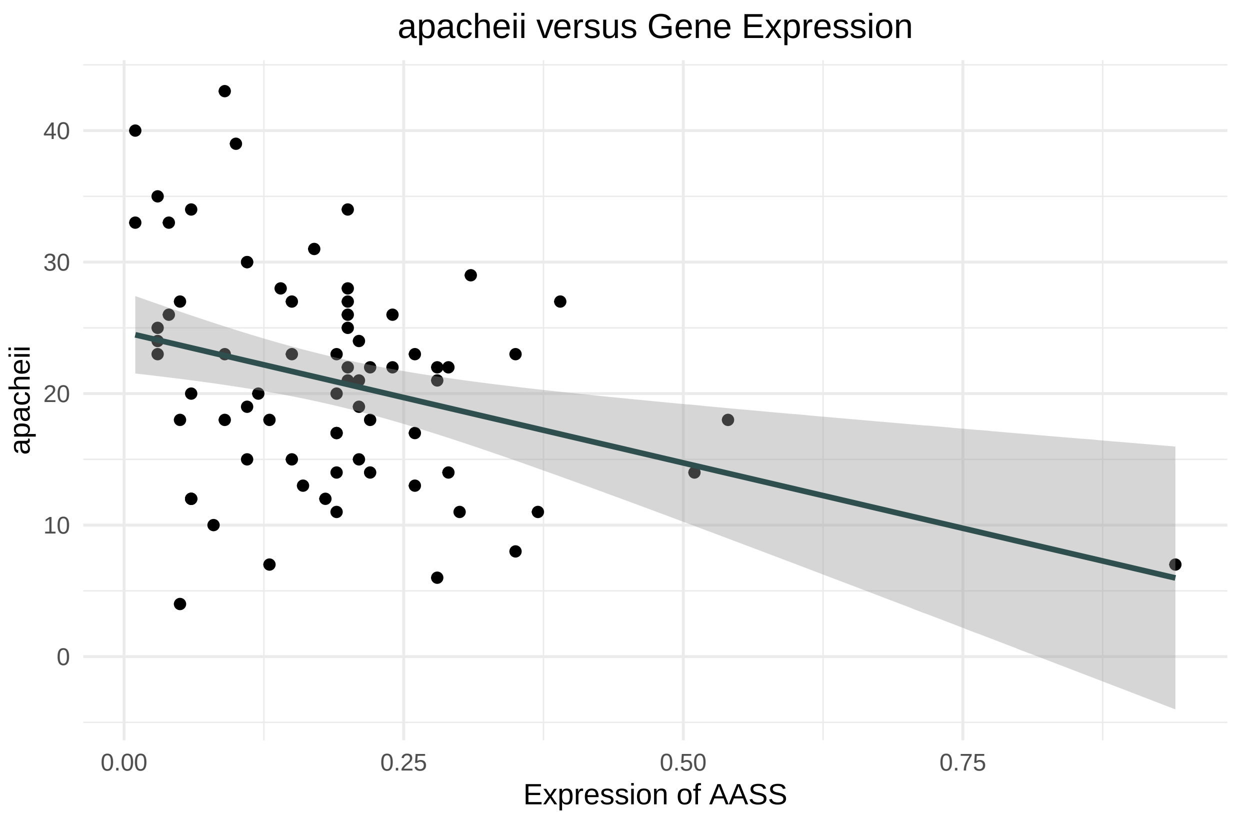
```
# scatter plots
scatters <- create_scatter(cleaned_data, gene_list, 'apacheii')

scatters[1]
```

```
## [[1]]
```


apacheii versus Gene Expression
apacheii
40
30
20
10
0
10
20
30
40
50
Expression of ABCA7
Line of best fit and standard error

```
scatters[2]
```

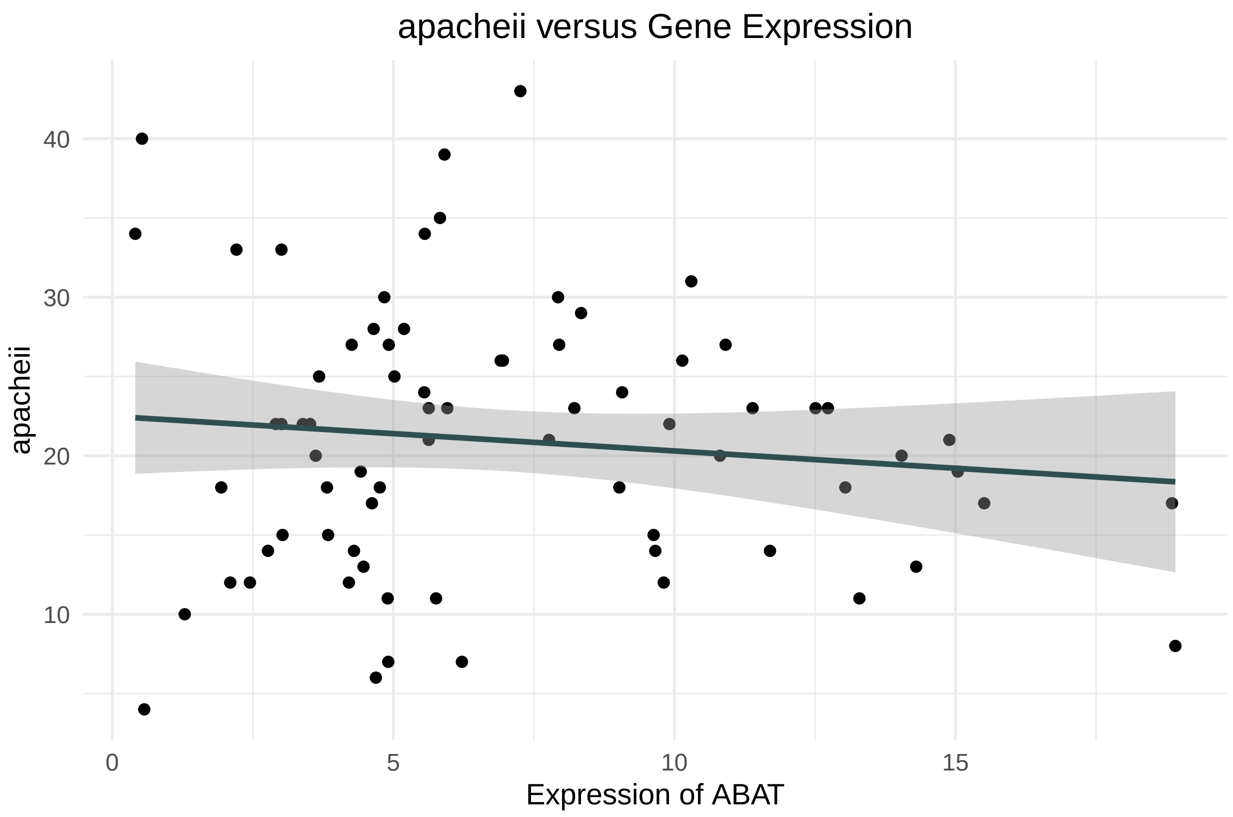
```
## [[1]]
```



Line of best fit and standard error

```
scatters[3]
```

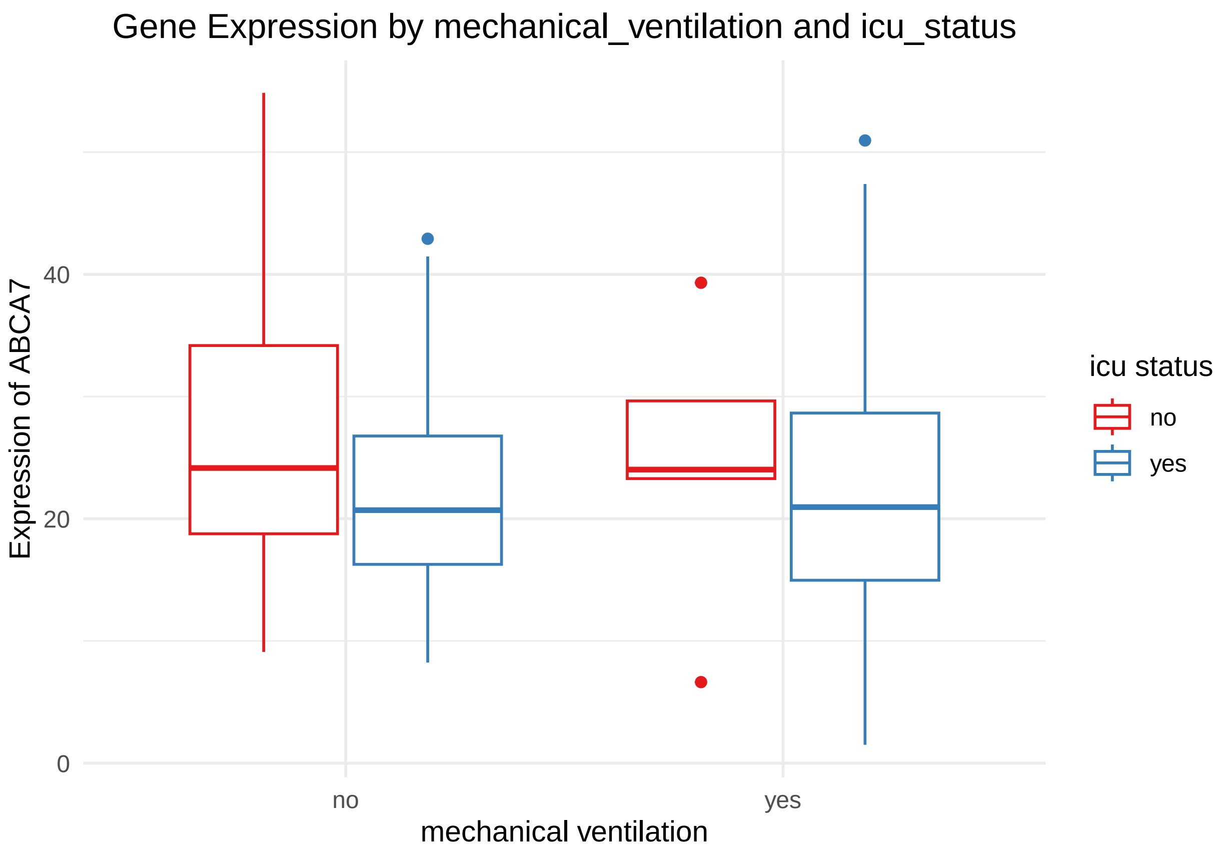
```
## [[1]]
```



Line of best fit and standard error

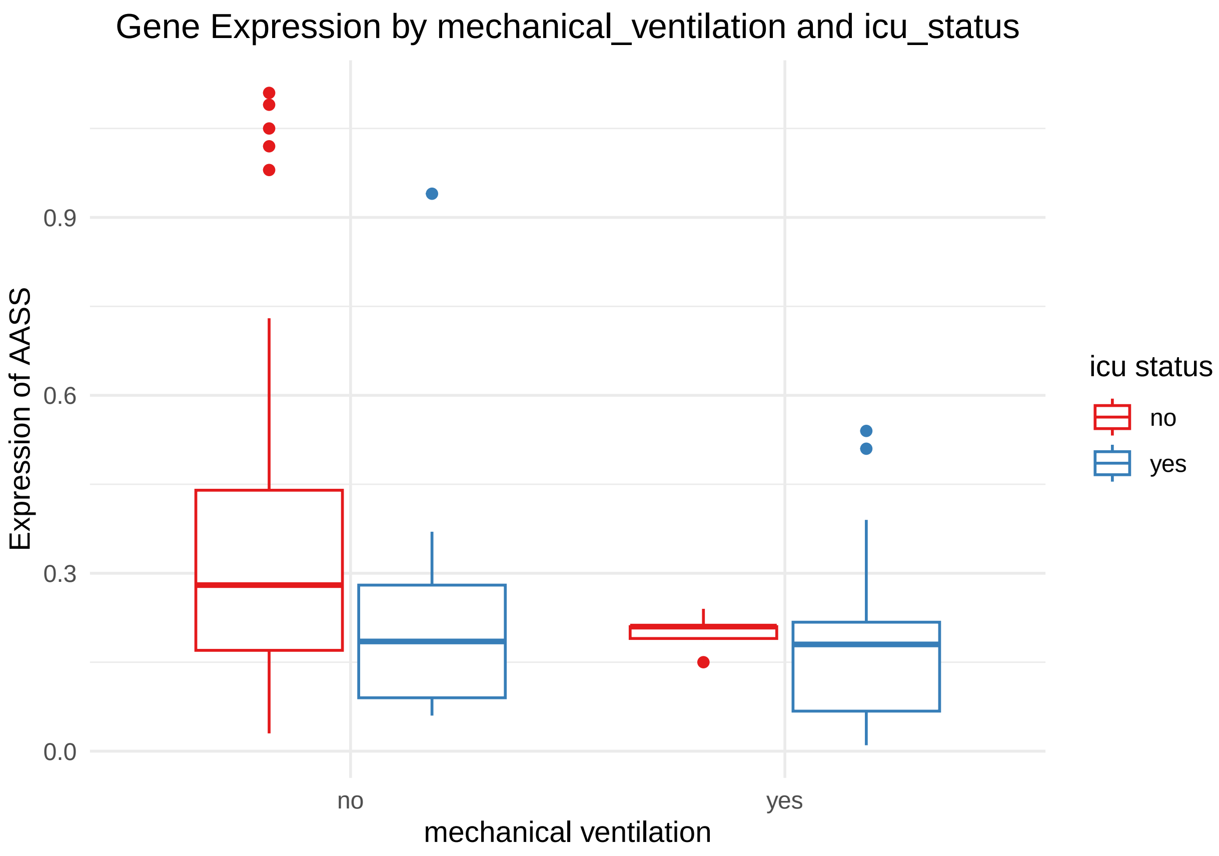
```
# box plots
boxplots <- create_boxplot(cleaned_data, gene_list, 'mechanical_ventilation', 'icu_status')

boxplots[1]
```

```
## [[1]]
```


Gene Expression by mechanical_ventilation and icu_status
Expression of ABCA7
0
20
40
no
yes
mechanical ventilation
icu status
no
yes

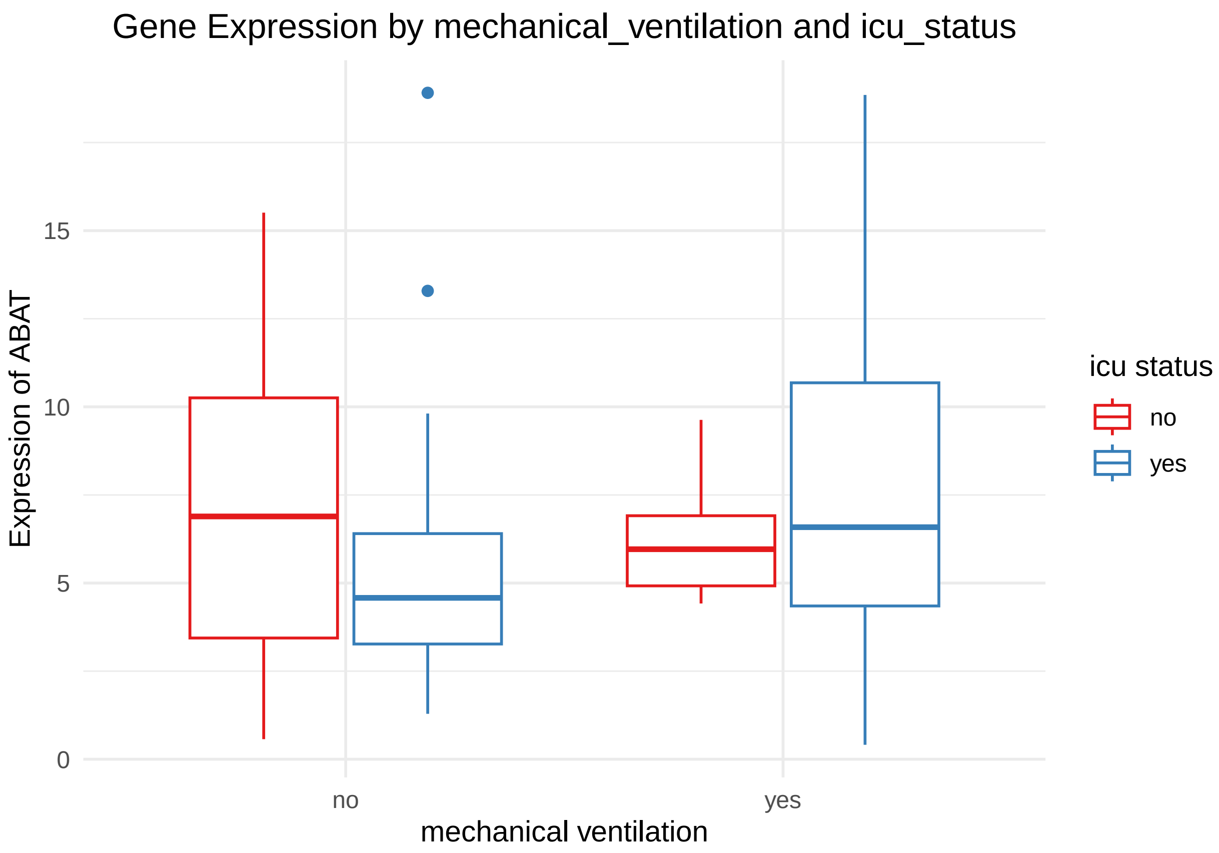
```
boxplots[2]
```

```
## [[1]]
```


Gene Expression by mechanical_ventilation and icu_status
Expression of AASS
0.0
0.3
0.6
0.9
no
yes
mechanical ventilation
icu status
no
yes

```
boxplots[3]
```

```
## [[1]]
```


Gene Expression by mechanical_ventilation and icu_status
Expression of ABAT
15
10
5
0
no
yes
mechanical ventilation
icu status
no
yes