QBS_sub_2

2024-08-07

```r
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
```

```r
library(dplyr)
library(ggplot2)
library(rlang)
```

```
##
## Attaching package: 'rlang'
##
## The following objects are masked from 'package:purrr':
##
##     %@%, flatten, flatten_chr, flatten_dbl, flatten_int, flatten_lgl,
##     flatten_raw, invoke, splice
```

```r
library(stringr)
```

```r
datafiles <- list.files(path = "/Users/amyw/Documents/QBS103/DSProject", pattern = ".csv")
print(datafiles)
```

```
## [1] "QBS103_GSE157103_genes.csv"         "QBS103_GSE157103_series_matrix.csv"
```

```r
setwd("/Users/amyw/Documents/QBS103/DSProject")

for (file in datafiles){
  the_file <- read.csv(file)}

gene_df <- read.csv('QBS103_GSE157103_genes.csv', header = TRUE,row.names = 1)
meta_df <- read.csv('QBS103_GSE157103_series_matrix.csv', header = TRUE, row.names = 1)

working_genes <- as.data.frame(t(gene_df)) #pivoting dataframe to select AAGAB as a column
```

```
#print(head(working_genes))

new_comp <- merge(working_genes, meta_df, by = 'row.names') #merging the two dataframes using the row n
final_comp <- data.frame(new_comp, row.names = 1) #renaming the row names as the participant ID
#print(head(final_comp))
```

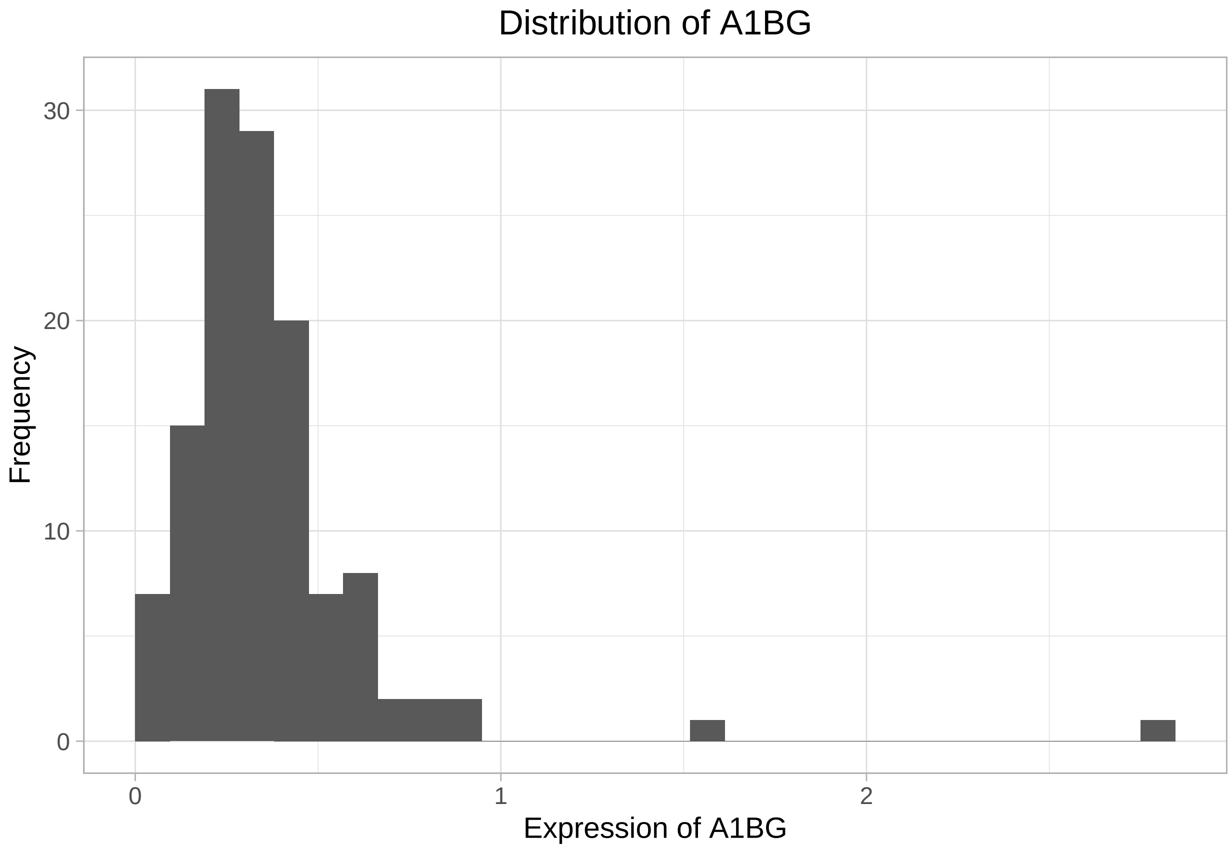
```
plotGenerator <- function(myDataFrame, geneNameList, contCovar, catCovar1, catCovar2) {
  myDataFrame <- as.data.frame(myDataFrame)
  final_comp$age<- as.numeric(final_comp$age) #converting the age column from characters to numerics
  #print(head(myDataFrame))

  #was having a lot of trouble referring to the input variables
  #was advised by my partner to refer to them as indices
  geneIndexList = list() #creating a list of indices in myDataFrame of the genes
  contVarIndex = 0
  catCovar1Index = 0
  catCovar2Index = 0

  for (i in 1:length(myDataFrame)) {
    #for loop to check if any column names match the inputted genes
    for (j in 1:length(geneNameList)) { #for loop to cycle through each gene
      if (colnames(myDataFrame)[i] == geneNameList[j]) {
        #if the column header in the dataframe matches the inputted gene
        geneIndexList <- append(geneIndexList, i) #add it to the list of gene indices
      }

      if (colnames(myDataFrame)[i] == contCovar) {
        contVarIndex = i
      }

      if (colnames(myDataFrame)[i] == catCovar1) {
        catCovar1Index = i
      }

      if (colnames(myDataFrame)[i] == catCovar2) {
        catCovar2Index = i
      }
    }
  }

  for (i in 1:length(geneNameList)) { #for loop for each inputted gene

    # ------------------------- HISTOGRAM ---------------------------------
    geneColumnName = colnames(myDataFrame)[geneIndexList[[i]]]
    #taking the index from earlier, finding it in the dataframe
    #renaming it as the new column name for ease of reference
    geneColumnData = myDataFrame[[geneIndexList[[i]]]]
    #taking the gene expression data of the gene
    histogramPlot <- ggplot(myDataFrame, aes(x=geneColumnData))+
                      geom_histogram(boundary = 0)+ #boundary at 0 to group data
                      #did not keep bins = 5 because other genes aren't represented properly
                      labs(title = paste('Distribution of', geneColumnName),
                           #paste function similar to f' string in python
```

```
                         x = paste('Expression of', geneColumnName),
                         y = 'Frequency') +
                    theme_light()+
                    theme(plot.title = element_text(hjust=0.5)) #centering title
  print(histogramPlot)

  # ------------------------- SCATTER PLOT ------------------------------
  contVarColumnName = colnames(myDataFrame)[contVarIndex]
  #same as earlier, taking the column name of the continuous variable
  #and referring to it as contVarColumnName
  scatterPlot <- ggplot(final_comp,
                        aes(x=as.numeric(as.character(myDataFrame[[contVarIndex]])),
                            y=geneColumnData))+
    #https://stackoverflow.com/questions/6386314/how-do-i-get-discrete-factor-levels-to-be-treated-as
    #needed to convert discrete variable to continuous
    geom_point()+
    labs(title = paste(str_to_title(contVarColumnName),
                       'compared to distribution of gene',
                       geneColumnName),
         x = str_to_title(contVarColumnName),
         y = paste('Expression of', str_to_title(contVarColumnName)))+
    #str_to_title to capitalize the first letter of the input
    scale_x_continuous()+
    theme_light()+
    theme(plot.title = element_text(hjust=0.5))

  print(scatterPlot)

  # ------------------------- BOX PLOT ----------------------------------
  boxFirst <- dplyr::select(myDataFrame, geneColumnName, catCovar1, catCovar2)
  #selecting data columns for easier viewing

  boxPlot <- ggplot(boxFirst,
                    aes (x = myDataFrame[[catCovar1Index]],
                         #x = the column of data indexed by catCovar1Index
                         y = geneColumnData,
                         #the column of expression data for the gene
                         fill = myDataFrame[[catCovar2Index]]))+
              geom_boxplot()+
              theme_light()+
              labs(title = paste('Distribution of', geneColumnName,
                                 'Based on', str_to_title(catCovar1),
                                 'Coloured by Mechanical Ventilation Status'),
                   # can also do 'Coloured by', str_to_title(catCovar2)),
                   x = paste(str_to_title(catCovar1)),
                   y = paste('Expression of', geneColumnName))+
              theme(plot.title = element_text(hjust=0))+
              scale_fill_discrete(name = 'Mechanical Ventilation Status')+
              # can also use name = paste(str_to_title(catCovar2), 'Status'
              scale_x_discrete(labels = c('Female', 'Male', 'Unknown'))

  print(boxPlot)
}
```

```
}

plotGenerator(final_comp, list("AAGAB", "A1BG", "A2M"), 'age', 'sex', 'mechanical_ventilation')
```

```
## Warning in plotGenerator(final_comp, list("AAGAB", "A1BG", "A2M"), "age", : NAs
## introduced by coercion
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



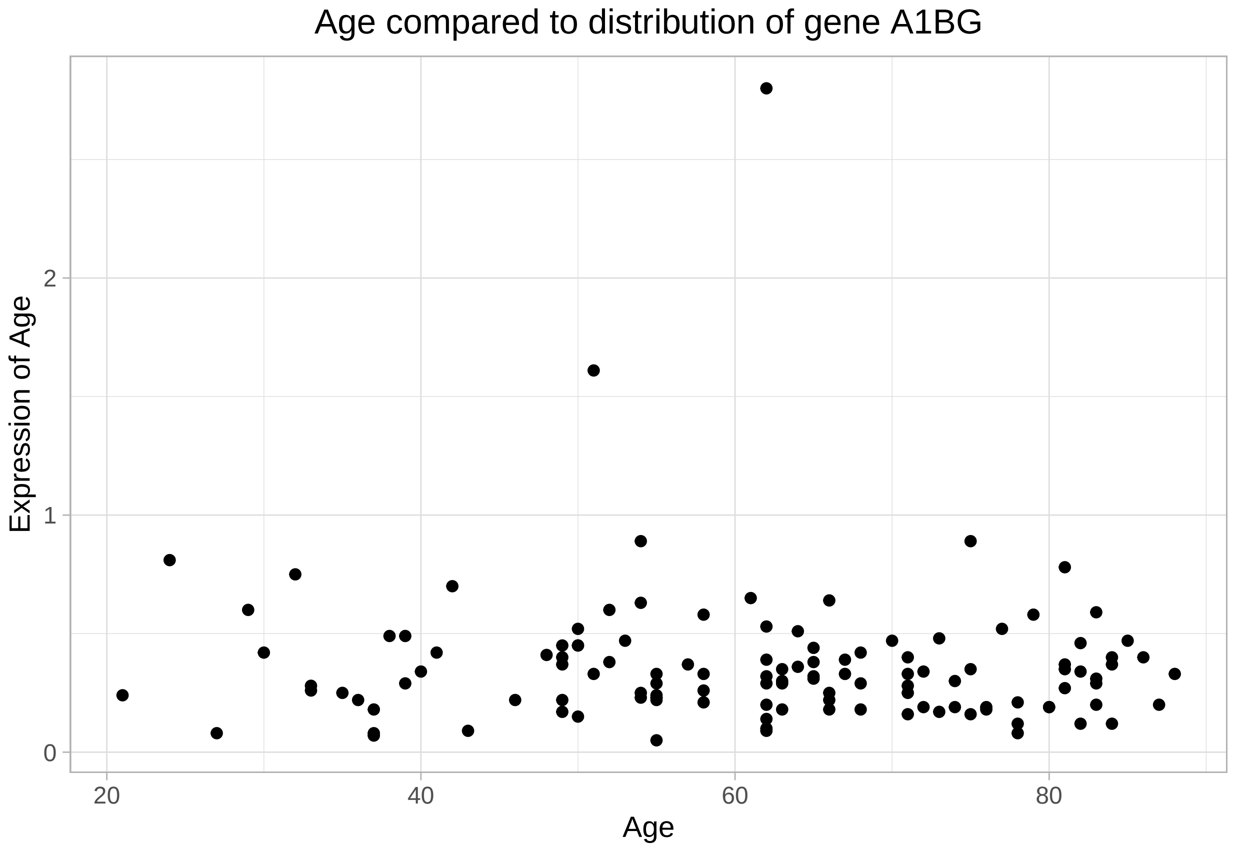
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

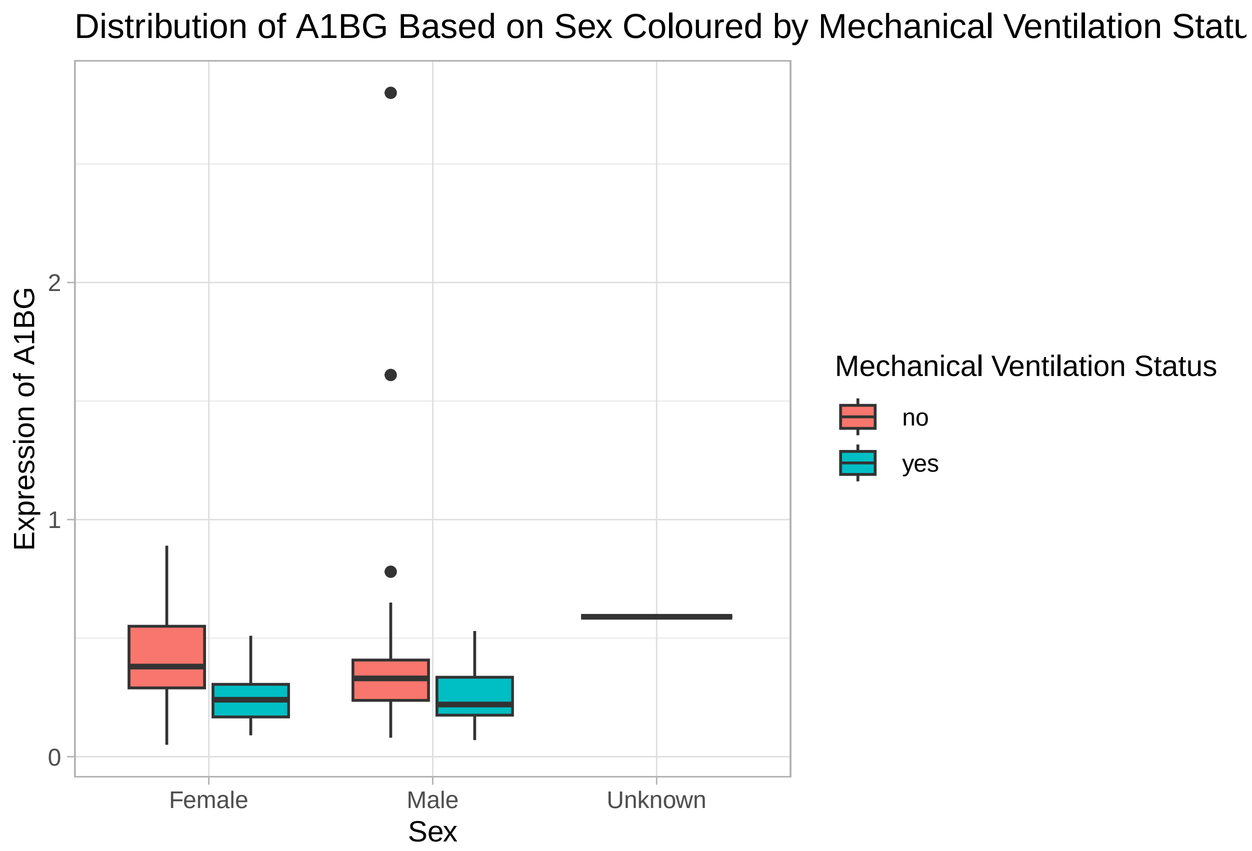
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (‘geom_point()‘).
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(geneColumnName)
##
##   # Now:
##   data %>% select(all_of(geneColumnName))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

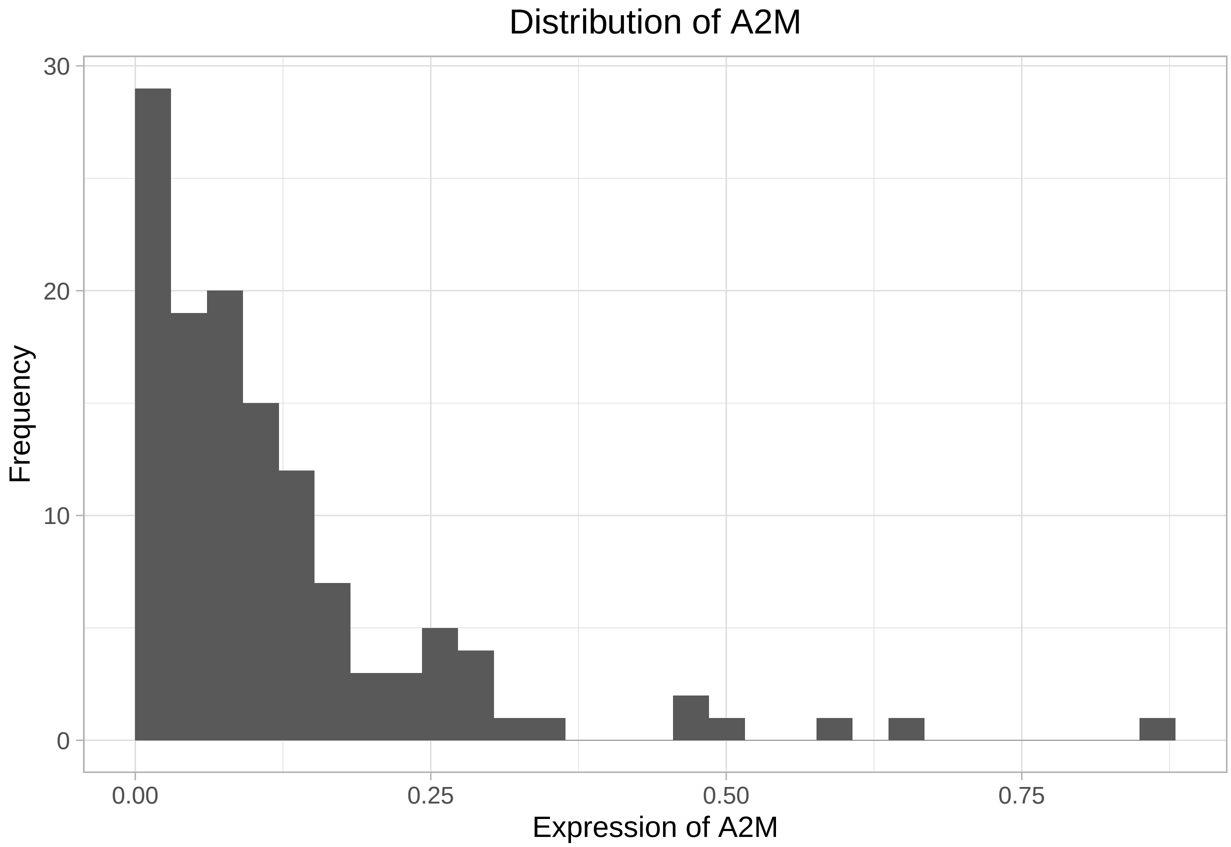
```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(catCovar1)
##
##   # Now:
##   data %>% select(all_of(catCovar1))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(catCovar2)
##
##   # Now:
##   data %>% select(all_of(catCovar2))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```


Age compared to distribution of gene A1BG
Expression of Age
0
1
2
20
40
60
80
Age

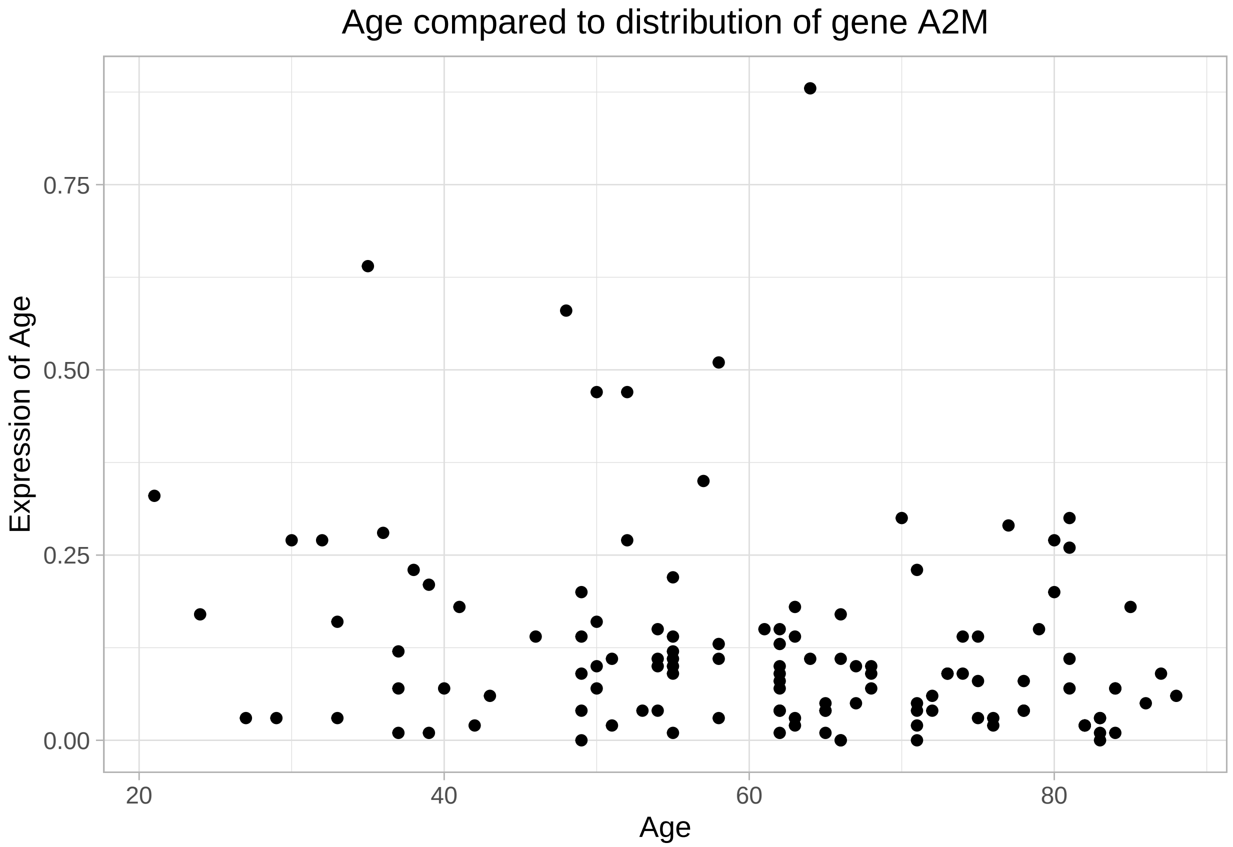


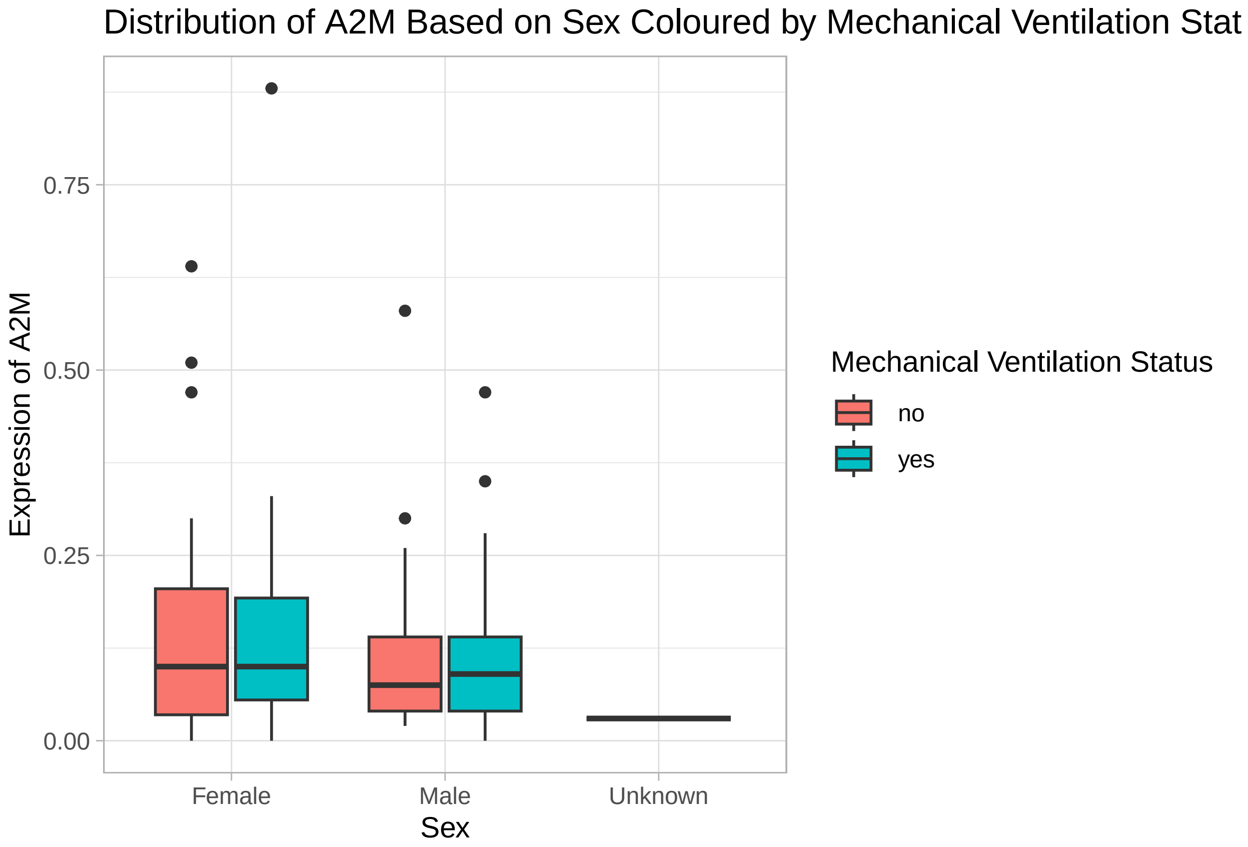
```
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
```



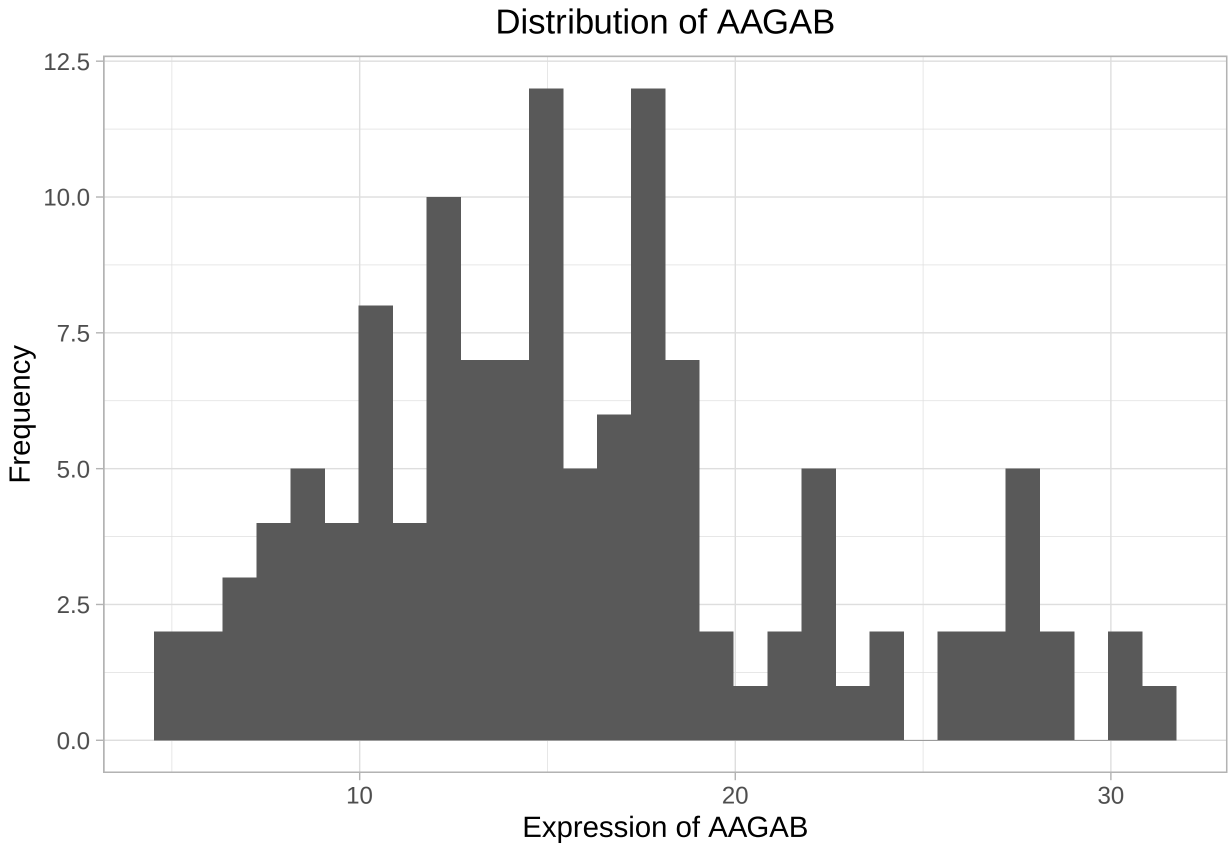
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (‘geom_point()‘).
```

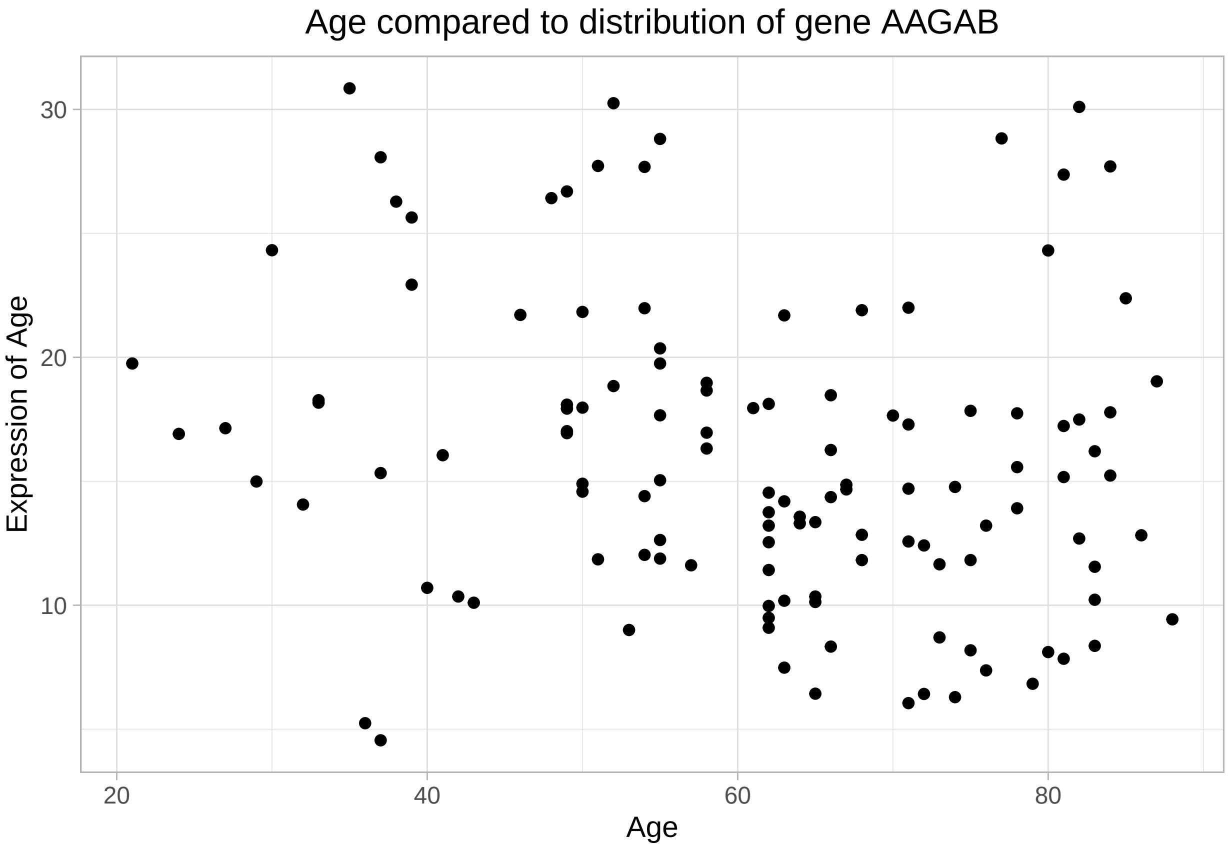

Age compared to distribution of gene A2M
Expression of Age
0.75
0.50
0.25
0.00
20
40
60
80
Age

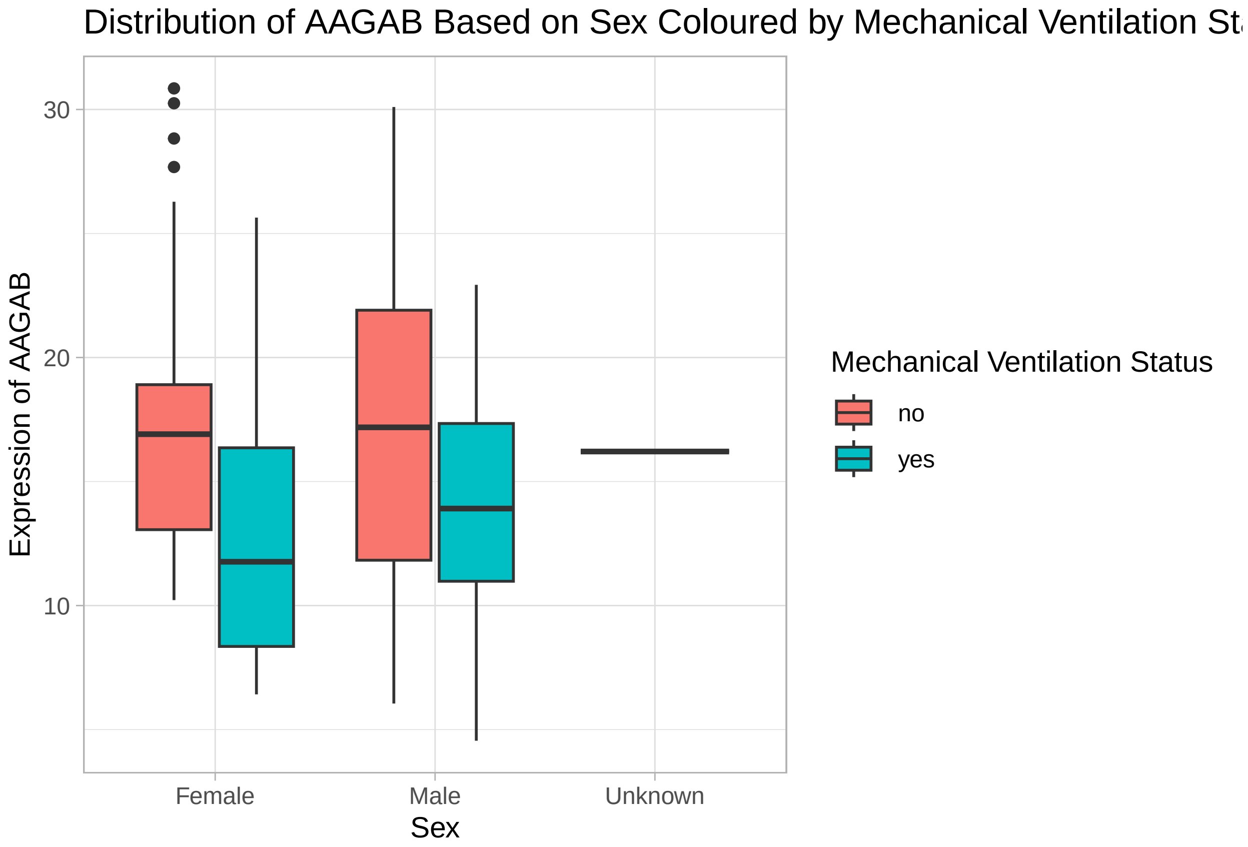


```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
## Warning in FUN(X[[i]], ...): Removed 2 rows containing missing values or values outside the scale ran
## (‘geom_point()‘).
```

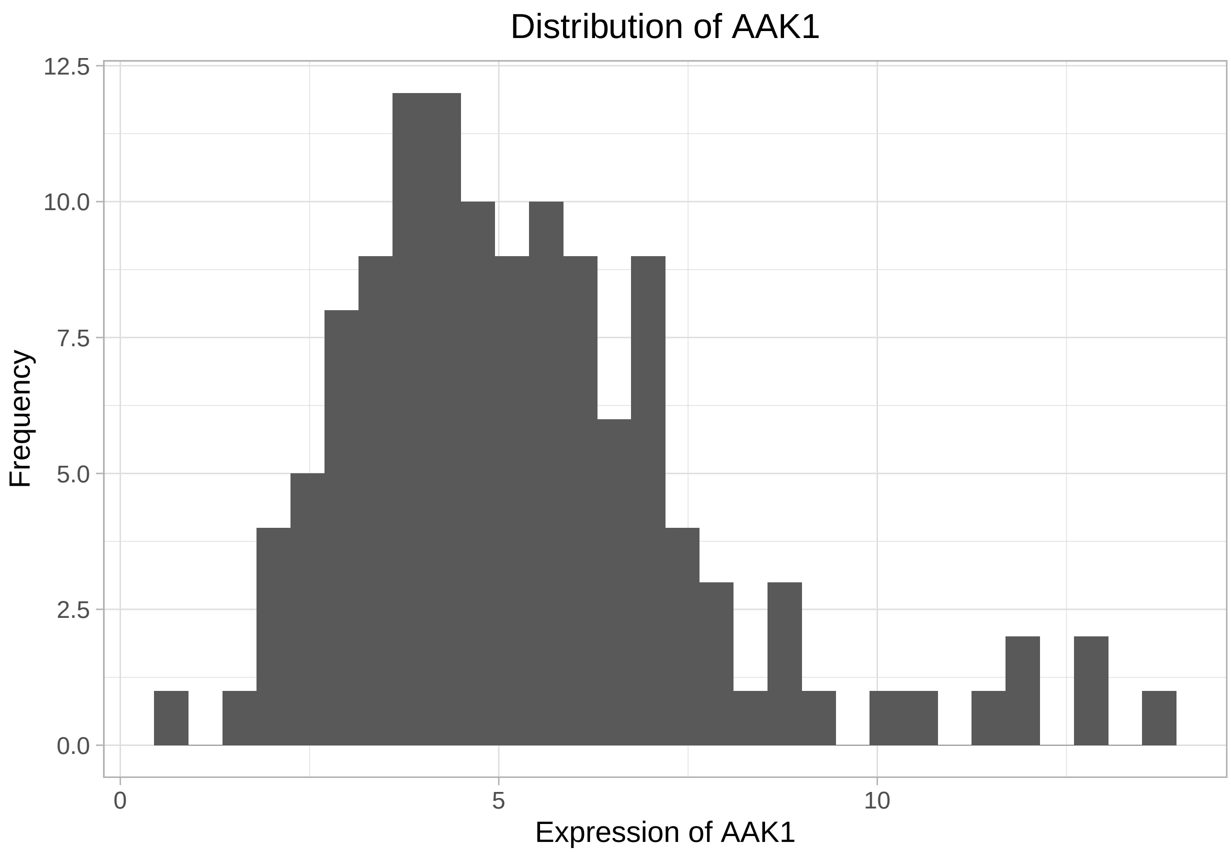

Age compared to distribution of gene AAGAB
Expression of Age
30
20
10
20
40
60
80
Age



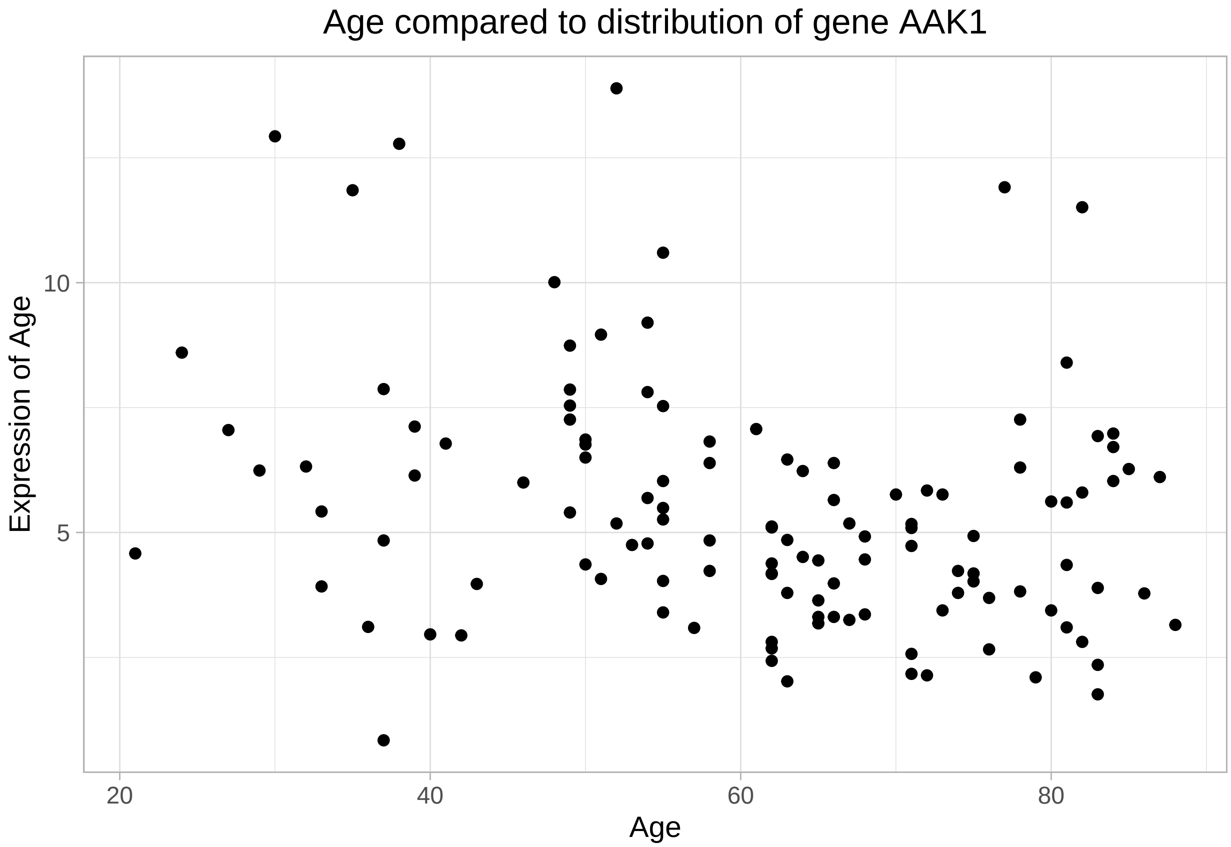
```
#the code works fine when inputting a list of 3 genes

#------------------------------FOR LOOP----------------------------------
geneList <- list('AAK1','AAAS','AACS')
for (i in 1:length(geneList)){
  plotGenerator(final_comp, geneList[[i]], 'age', 'sex', 'mechanical_ventilation')
}
```

```
## Warning in plotGenerator(final_comp, geneList[[i]], "age", "sex",
## "mechanical_ventilation"): NAs introduced by coercion
```

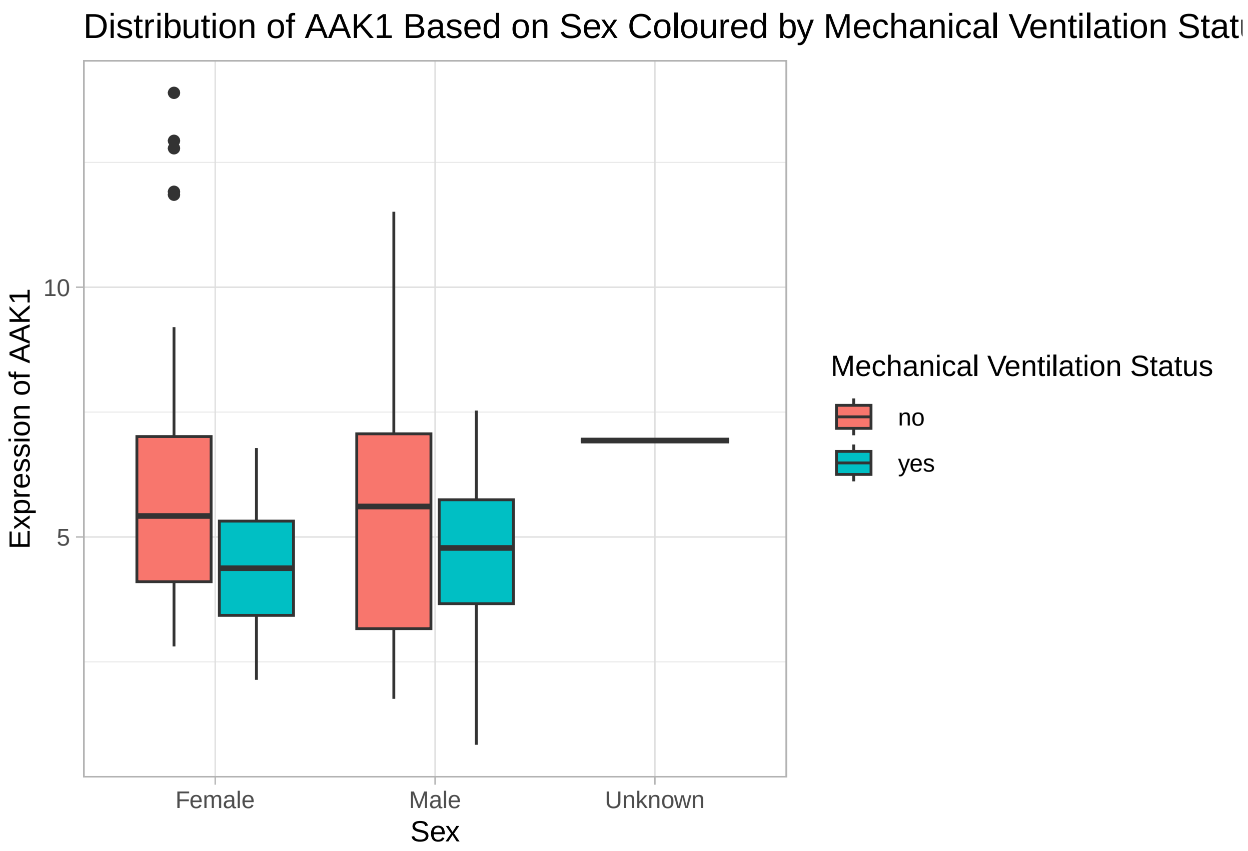
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



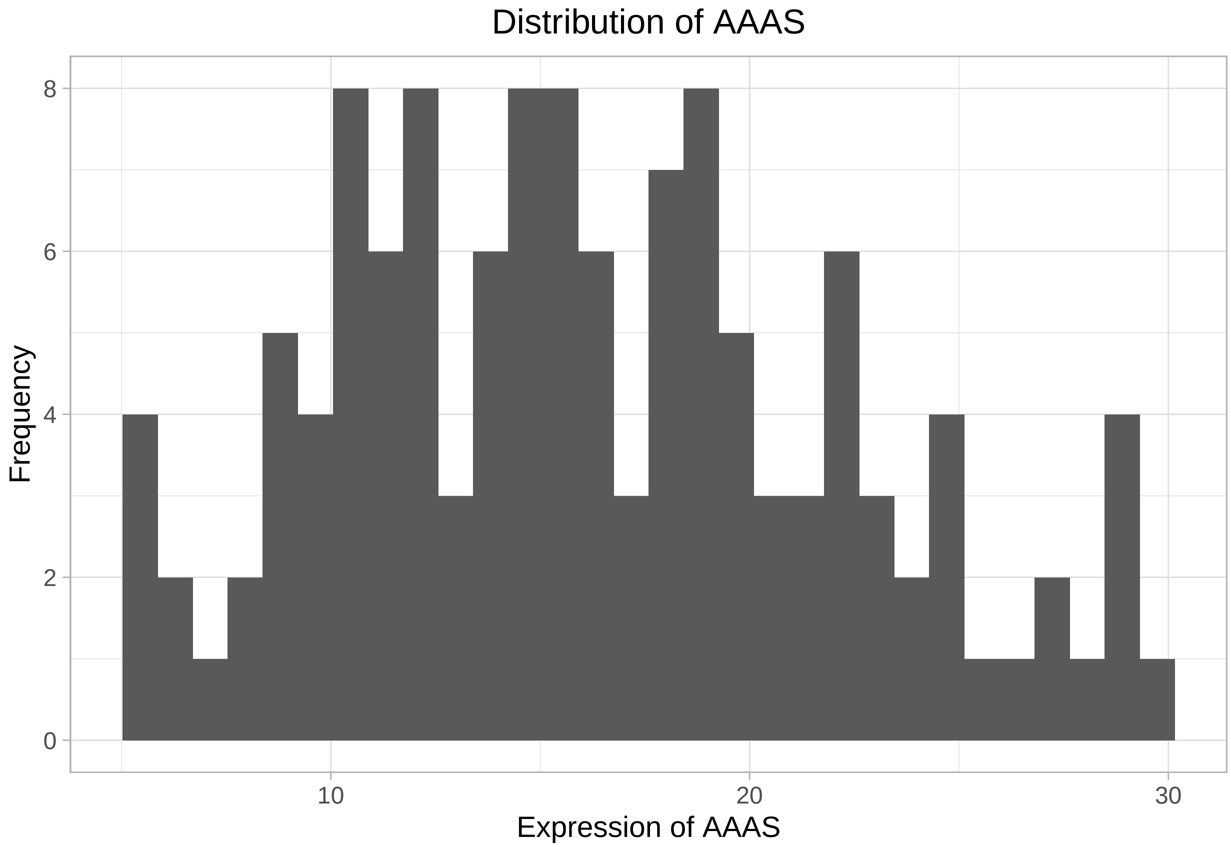
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
## Warning in FUN(X[[i]], ...): Removed 2 rows containing missing values or values outside the scale ran
## (‘geom_point()‘).
```



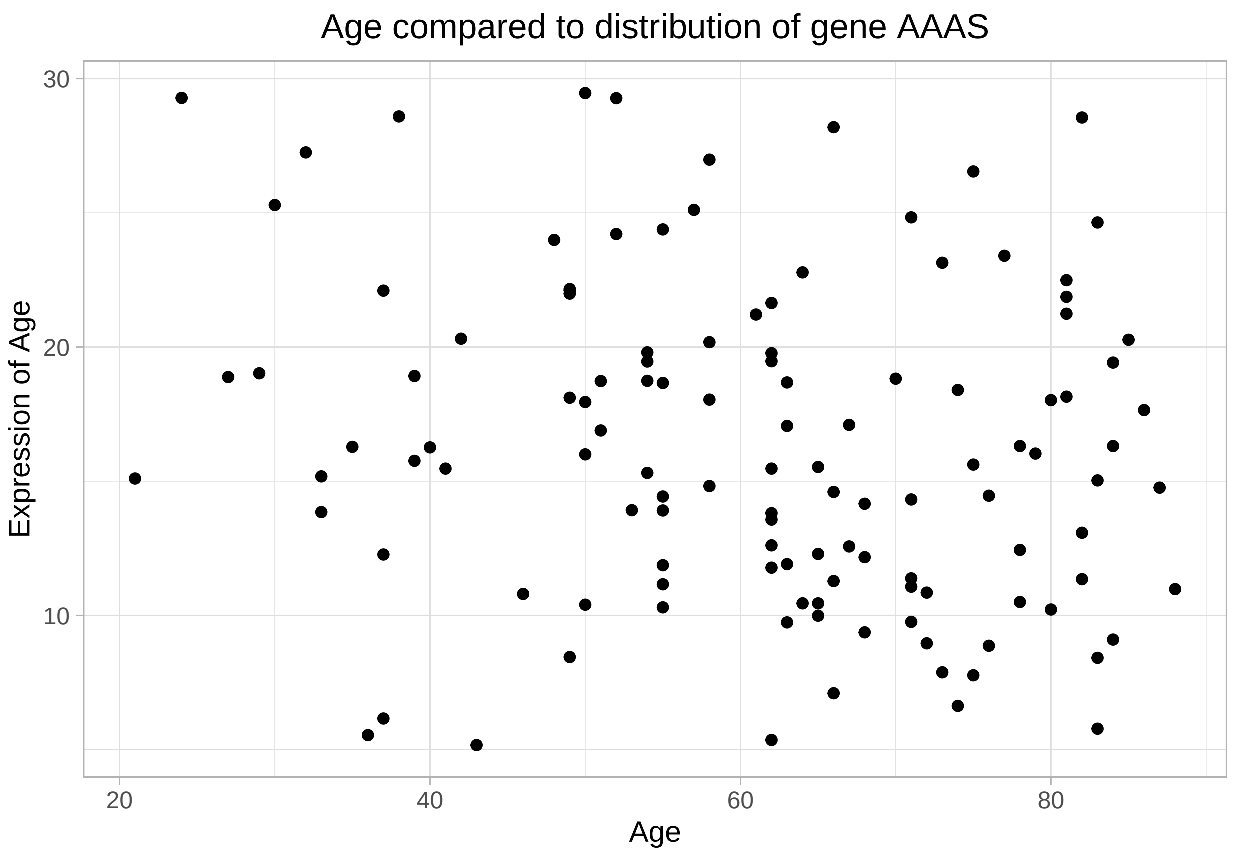
```
## Warning in plotGenerator(final_comp, geneList[[i]], "age", "sex",
## "mechanical_ventilation"): NAs introduced by coercion
```



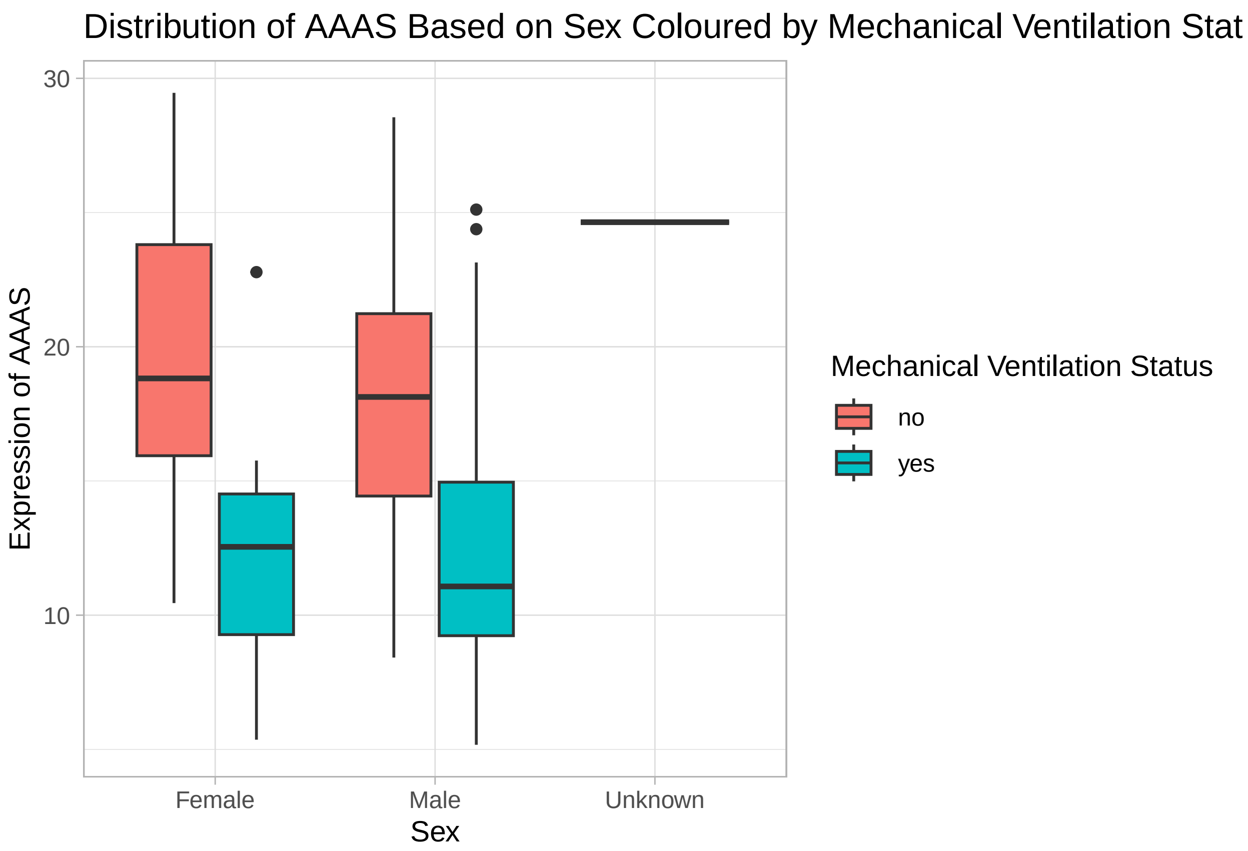
```
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
```



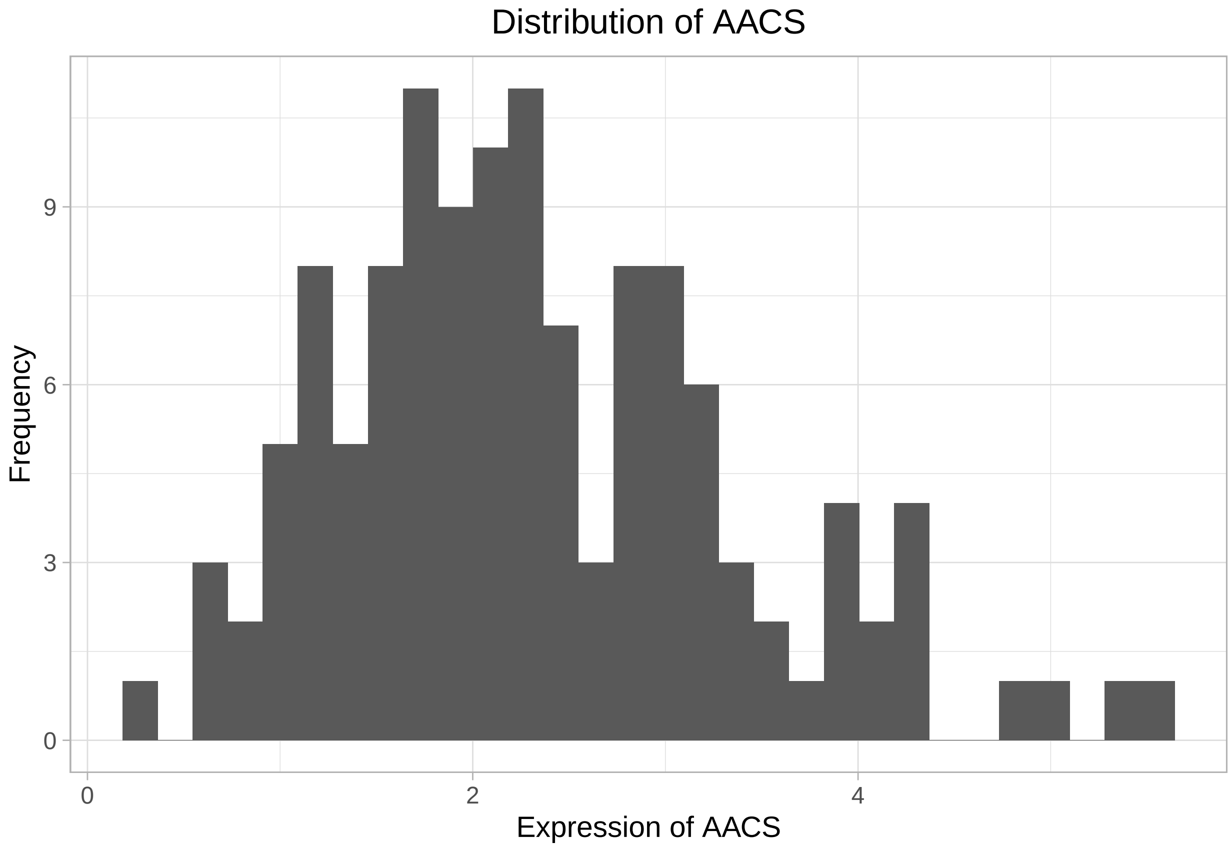
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
## Warning in FUN(X[[i]], ...): Removed 2 rows containing missing values or values outside the scale ran
## (‘geom_point()‘).
```



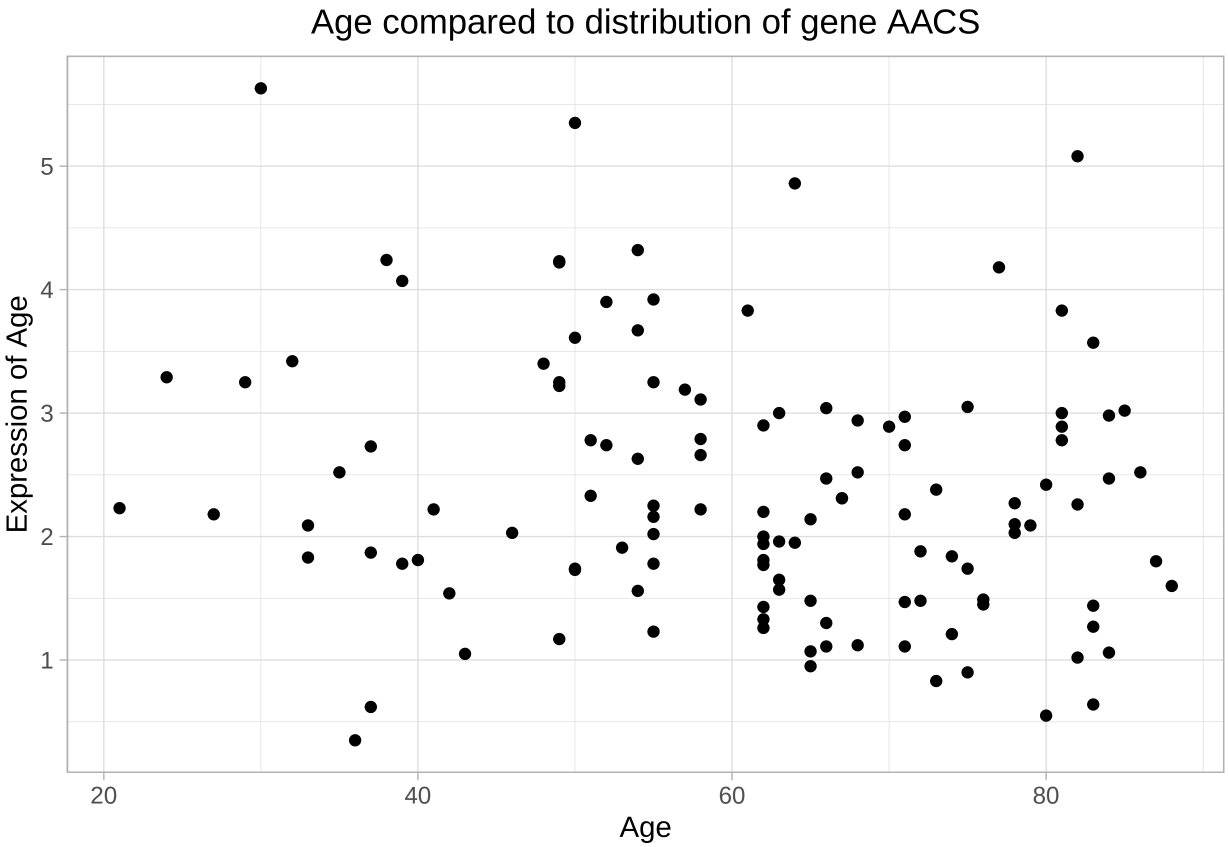
```
## Warning in plotGenerator(final_comp, geneList[[i]], "age", "sex",
## "mechanical_ventilation"): NAs introduced by coercion
```

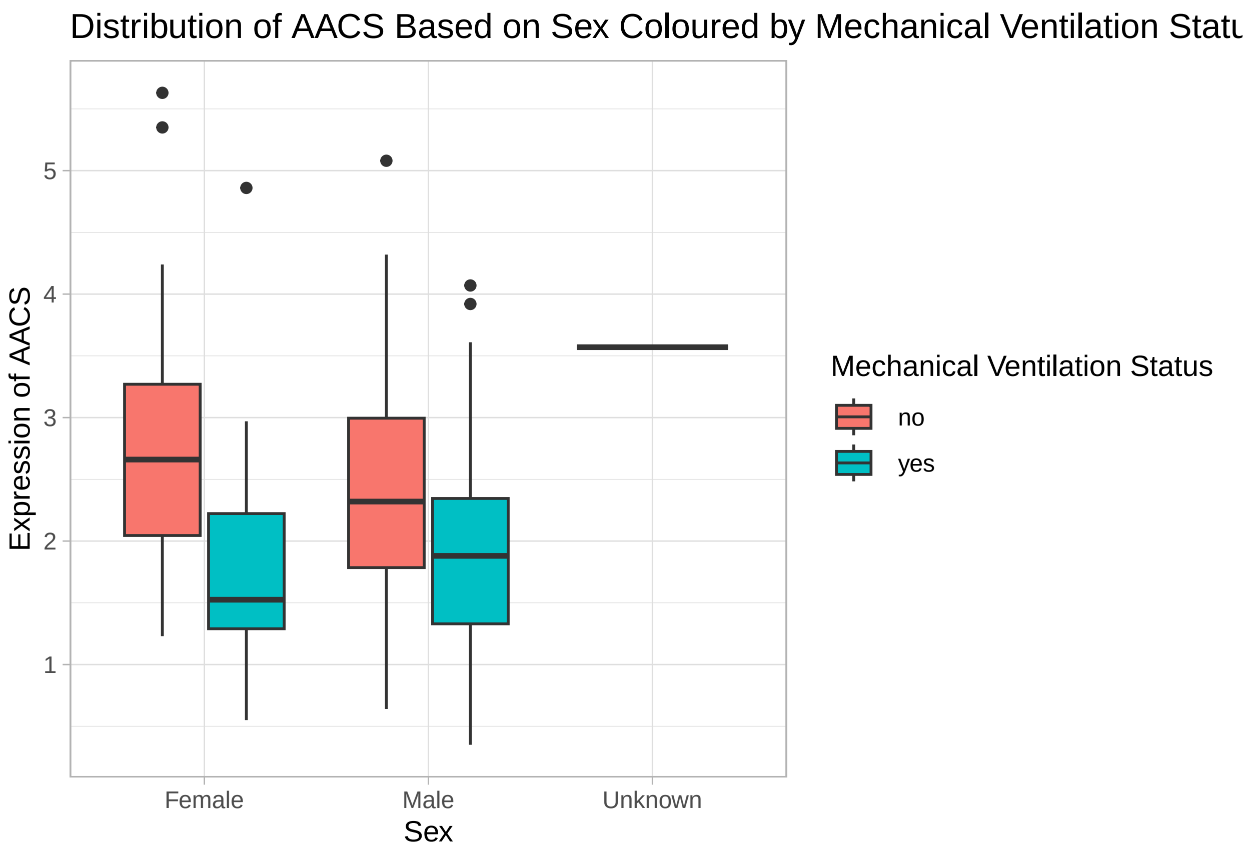


```
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
```



```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
## Warning in FUN(X[[i]], ...): Removed 2 rows containing missing values or values outside the scale ran
## (‘geom_point()‘).
```


Age compared to distribution of gene AACS
Expression of Age
Age
1
2
3
4
5
20
40
60
80



```
#unnecessary, since the original code includes cycling through the list of genes
```