EDA

Ruxue Sun

2024-07-20

EDA

```
#1. Read data
# Load necessary libraries
library(DESeq2)
```

```
## Warning: package 'DESeq2' was built under R version 4.3.3
## Loading required package: S4Vectors
## Loading required package: stats4
## Loading required package: BiocGenerics
##
## Attaching package: 'BiocGenerics'
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min
##
## Attaching package: 'S4Vectors'
## The following object is masked from 'package:utils':
##
##     findMatches
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
## Loading required package: IRanges
## Loading required package: GenomicRanges
## Loading required package: GenomeInfoDb
## Warning: package 'GenomeInfoDb' was built under R version 4.3.3
```

```
## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
library(readr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v purrr     1.0.2
## 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

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::%within%() masks IRanges::%within%()
## x dplyr::collapse()     masks IRanges::collapse()
## x dplyr::combine()      masks Biobase::combine(), BiocGenerics::combine()
## x dplyr::count()        masks matrixStats::count()
```

```
## x dplyr::desc()         masks IRanges::desc()
## x tidyr::expand()       masks S4Vectors::expand()
## x dplyr::filter()       masks stats::filter()
## x dplyr::first()        masks S4Vectors::first()
## x dplyr::lag()          masks stats::lag()
## x ggplot2::Position()   masks BiocGenerics::Position(), base::Position()
## x purrr::reduce()       masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()       masks S4Vectors::rename()
## x lubridate::second()   masks S4Vectors::second()
## x lubridate::second<-() masks S4Vectors::second<-()
## x dplyr::slice()        masks IRanges::slice()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
# Read data file
counts_all <- read_csv('counts_all.csv')
```

```
## Rows: 6795 Columns: 62
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): Geneid, Strand
## dbl (60): Chr, Start, End, Length, A_Y_0_1, A_R_30_1, A_R_120_1, A_R_240_1, ...
##
## 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.
```

```
samplesheet <- read_csv('Gat201_samplesheet.csv')
```

```
## Rows: 56 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): SampleID, Title, Group, Strain, GAT201, Media, Condition
## dbl (3): Timepoint, Time, BioRep
##
## 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.
```

```
# Convert tibble to data frame
counts_all <- as.data.frame(counts_all)
samplesheet <- as.data.frame(samplesheet)

# Set the line name
rownames(counts_all) <- counts_all$Geneid
# Remove rows containing 0. If a pedestrian data contains 0, it is considered an abnormal value and is
filtered_counts_all<-counts_all[rowSums(counts_all ==0) ==0,]
counts_all_without_names <- counts_all[,c(-1,-2,-3,-4,-5,-6)]

# Sample table settings Row name
rownames(samplesheet) <- samplesheet$Title
samplesheet_without_ids <- samplesheet[,c(-1,-2,-3,-6,-8)]
# Convert the necessary columns to factor types
samplesheet_without_ids$GAT201 <- as.factor(samplesheet_without_ids$GAT201)
samplesheet_without_ids$Condition <- as.factor(samplesheet_without_ids$Condition)
samplesheet_without_ids$BioRep <- as.factor(samplesheet_without_ids$BioRep)
#samplesheet_without_ids$Strain <- as.factor(samplesheet_without_ids$Strain)
#samplesheet_without_ids$Time <- as.factor(samplesheet_without_ids$Time)
```

```r
#3.exploratory data analysis

# Create DESeq2 object
dds <- DESeqDataSetFromMatrix(countData = counts_all_without_names,
                              colData = samplesheet_without_ids,
                              design = ~ GAT201 + Condition +  BioRep)
```

```
## converting counts to integer mode
```

```r
# Filter out low-expression genes
dds <- dds[rowSums(counts(dds)) > 10,]

# Run DESeq2 for standardisation and differential expression analysis
dds <- DESeq(dds)
```

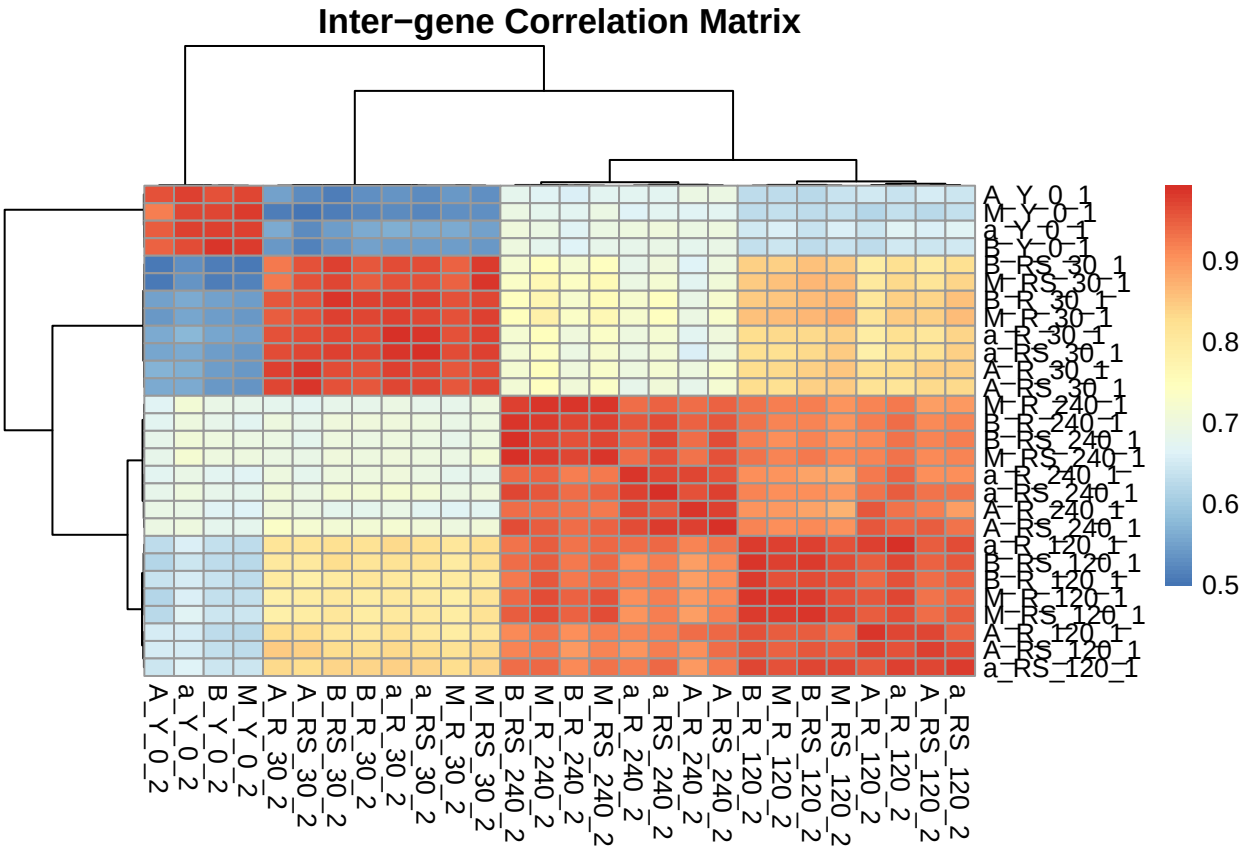
```
## estimating size factors
## estimating dispersions
## gene-wise dispersion estimates
## mean-dispersion relationship
## final dispersion estimates
## fitting model and testing
```

```r
# Perform variance stabilisation transformation
vsd <- vst(dds, blind = FALSE)

# Extract the duplicated sample twice
replicate1_samples <- rownames(samplesheet_without_ids[samplesheet_without_ids$BioRep == 1, ])
replicate2_samples <- rownames(samplesheet_without_ids[samplesheet_without_ids$BioRep == 2, ])
replicate1_data <- assay(vsd)[, replicate1_samples]
replicate2_data <- assay(vsd)[, replicate2_samples]
```

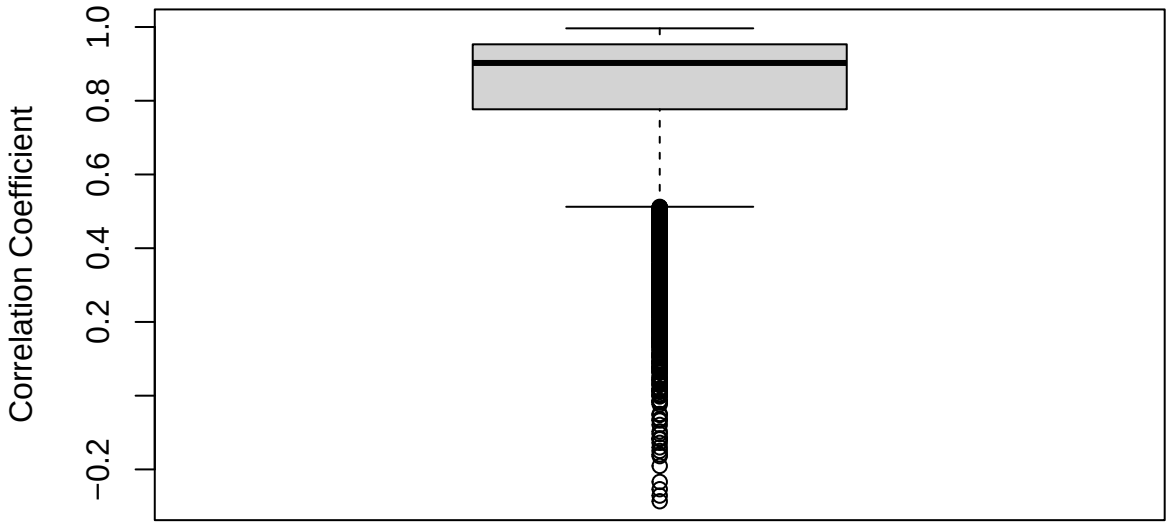
```r
#Level 1: Calculate the correlation coefficient between the two inter-gene replications
cor_matrix <- cor(assay(vsd)[, replicate1_samples], assay(vsd)[, replicate2_samples])

# Extract diagonal elements
diag_cor <- diag(cor_matrix)

# Average correlation coefficient
mean_cor <- mean(diag_cor)
print(mean_cor)
```

```
## [1] 0.9803225
```

```r
# Draw a heat map
library(pheatmap)
pheatmap(cor_matrix, clustering_distance_rows = "correlation", clustering_distance_cols = "correlation"
```




```r
#Level 2: Calculate the correlation between the two intra-gene replications
#That is, calculate the correlation coefficient between each gene
# Calculate the correlation coefficient of each gene
calculate_gene_correlation <- function(replicate1_data, replicate2_data, gene_names) {
  gene_correlations <- numeric(length(gene_names))

  for (i in 1:length(gene_names)) {
    gene <- gene_names[i]
    data1 <- replicate1_data[gene, ]
    data2 <- replicate2_data[gene, ]

    gene_correlations[i] <- cor(data1, data2)
  }

  names(gene_correlations) <- gene_names
  return(gene_correlations)
}

gene_correlations <- calculate_gene_correlation(replicate1_data, replicate2_data, rownames(replicate1_da
# Take the value of the correlation coefficient.
gene_correlations01 <- as.data.frame(gene_correlations)
# Draw a box plot
boxplot(gene_correlations, main = "Intra-gene Correlation Coefficients", ylab = "Correlation Coefficien
```




```r
# Identify low-correlation genes
threshold <- quantile(gene_correlations, 0.05)
low_correlation_genes <- names(gene_correlations[gene_correlations < threshold])
low_correlation_genes01 <- as.data.frame(low_correlation_genes)

# Extract low-correlation genes and their corresponding correlation coefficients
low_correlation_genes_df <- data.frame(
  Gene = names(gene_correlations[low_correlation_genes]),
  Correlation = gene_correlations[low_correlation_genes]
)

# View data frame contents
print(low_correlation_genes_df)
```

```
##                  Gene   Correlation
## CNAG_00173 CNAG_00173  3.844304e-01
## CNAG_00177 CNAG_00177  1.220487e-01
## CNAG_00248 CNAG_00248  7.698106e-02
## CNAG_00289 CNAG_00289  3.811664e-01
## CNAG_00310 CNAG_00310  3.971793e-01
## CNAG_00317 CNAG_00317  3.960089e-01
## CNAG_00323 CNAG_00323  3.914433e-01
## CNAG_00356 CNAG_00356  3.769811e-01
## CNAG_00375 CNAG_00375  3.466698e-01
## CNAG_00391 CNAG_00391  1.657230e-01
## CNAG_00419 CNAG_00419  3.714081e-01
## CNAG_00423 CNAG_00423  1.096653e-01
## CNAG_00428 CNAG_00428  1.801184e-01
## CNAG_00460 CNAG_00460  3.290409e-01
## CNAG_00475 CNAG_00475  3.678332e-01
## CNAG_00481 CNAG_00481 -1.498539e-01
## CNAG_00493 CNAG_00493  2.931876e-01
## CNAG_00525 CNAG_00525  2.882143e-02
## CNAG_00540 CNAG_00540  3.041858e-01
## CNAG_00602 CNAG_00602  2.836620e-01
```

```
## CNAG_00619 CNAG_00619  2.371251e-01
## CNAG_00751 CNAG_00751  7.804553e-02
## CNAG_00755 CNAG_00755  3.924702e-01
## CNAG_00789 CNAG_00789  2.041941e-03
## CNAG_00823 CNAG_00823  2.356694e-01
## CNAG_00828 CNAG_00828  3.703082e-01
## CNAG_00859 CNAG_00859 -2.863271e-01
## CNAG_00885 CNAG_00885  7.721389e-02
## CNAG_00900 CNAG_00900  3.754841e-01
## CNAG_00926 CNAG_00926  1.253925e-02
## CNAG_00939 CNAG_00939  6.774268e-02
## CNAG_00971 CNAG_00971  2.040375e-01
## CNAG_00974 CNAG_00974  1.144147e-01
## CNAG_01008 CNAG_01008  2.852381e-01
## CNAG_01050 CNAG_01050  1.501425e-01
## CNAG_01057 CNAG_01057  3.484597e-01
## CNAG_01058 CNAG_01058  3.371305e-01
## CNAG_01073 CNAG_01073  2.640654e-01
## CNAG_01077 CNAG_01077  1.933407e-01
## CNAG_01235 CNAG_01235  3.818670e-01
## CNAG_01299 CNAG_01299  5.332202e-02
## CNAG_01319 CNAG_01319  1.522091e-01
## CNAG_01363 CNAG_01363  1.871594e-01
## CNAG_01367 CNAG_01367  2.137167e-01
## CNAG_01372 CNAG_01372  3.503623e-01
## CNAG_01373 CNAG_01373  1.971130e-01
## CNAG_01389 CNAG_01389  3.710555e-01
## CNAG_01406 CNAG_01406  2.877424e-01
## CNAG_01438 CNAG_01438  2.577223e-01
## CNAG_01482 CNAG_01482  3.152114e-01
## CNAG_01497 CNAG_01497  2.478341e-01
## CNAG_01507 CNAG_01507  1.998116e-01
## CNAG_01521 CNAG_01521  1.796653e-01
## CNAG_01522 CNAG_01522  2.521510e-01
## CNAG_01531 CNAG_01531  2.216840e-01
## CNAG_01566 CNAG_01566  3.149404e-01
## CNAG_01606 CNAG_01606  2.899041e-01
## CNAG_01630 CNAG_01630  1.766316e-01
## CNAG_01640 CNAG_01640  2.448927e-01
## CNAG_01663 CNAG_01663  2.247145e-01
## CNAG_01666 CNAG_01666  1.432263e-01
## CNAG_01668 CNAG_01668  8.773511e-02
## CNAG_01669 CNAG_01669  2.281492e-01
## CNAG_01697 CNAG_01697  8.766991e-02
## CNAG_01733 CNAG_01733  2.014100e-01
## CNAG_01741 CNAG_01741  2.320493e-01
## CNAG_01754 CNAG_01754  2.028943e-01
## CNAG_01790 CNAG_01790  2.995110e-01
## CNAG_01806 CNAG_01806  3.435195e-01
## CNAG_01811 CNAG_01811  3.263283e-01
## CNAG_01826 CNAG_01826  2.959341e-01
## CNAG_01857 CNAG_01857  3.383021e-01
## CNAG_01873 CNAG_01873  1.365124e-01
## CNAG_01891 CNAG_01891  3.910026e-01
```

```
## CNAG_01915 CNAG_01915  3.886114e-01
## CNAG_01921 CNAG_01921  3.885973e-01
## CNAG_01933 CNAG_01933  8.387052e-02
## CNAG_01941 CNAG_01941  3.100855e-01
## CNAG_01971 CNAG_01971 -1.905806e-01
## CNAG_02022 CNAG_02022  1.906943e-01
## CNAG_02028 CNAG_02028  3.468920e-01
## CNAG_02075 CNAG_02075  7.438553e-02
## CNAG_02151 CNAG_02151  2.929310e-02
## CNAG_02199 CNAG_02199  2.711910e-01
## CNAG_02207 CNAG_02207  3.721648e-01
## CNAG_02218 CNAG_02218  3.422357e-01
## CNAG_02236 CNAG_02236  3.672953e-01
## CNAG_02253 CNAG_02253  3.890491e-01
## CNAG_02281 CNAG_02281  3.953884e-01
## CNAG_02303 CNAG_02303  2.670593e-01
## CNAG_02311 CNAG_02311  4.327118e-02
## CNAG_02339 CNAG_02339  3.011186e-01
## CNAG_02341 CNAG_02341  2.256285e-01
## CNAG_02364 CNAG_02364  1.940558e-01
## CNAG_02389 CNAG_02389  3.490924e-01
## CNAG_02428 CNAG_02428  3.689056e-01
## CNAG_02483 CNAG_02483  1.402640e-01
## CNAG_02488 CNAG_02488  1.968328e-01
## CNAG_02491 CNAG_02491  3.524484e-01
## CNAG_02503 CNAG_02503  3.979319e-01
## CNAG_02517 CNAG_02517  2.809230e-01
## CNAG_02536 CNAG_02536  1.052849e-01
## CNAG_02538 CNAG_02538  2.607706e-01
## CNAG_02549 CNAG_02549  3.584517e-01
## CNAG_02555 CNAG_02555  3.370899e-01
## CNAG_02572 CNAG_02572  1.835347e-01
## CNAG_02582 CNAG_02582  1.211895e-02
## CNAG_02596 CNAG_02596 -2.710237e-01
## CNAG_02602 CNAG_02602  2.442632e-01
## CNAG_02671 CNAG_02671  3.675746e-01
## CNAG_02681 CNAG_02681  2.158324e-01
## CNAG_02706 CNAG_02706  3.101638e-01
## CNAG_02708 CNAG_02708  3.890723e-01
## CNAG_02711 CNAG_02711  2.472475e-01
## CNAG_02729 CNAG_02729  3.698493e-02
## CNAG_02741 CNAG_02741  3.144222e-01
## CNAG_02749 CNAG_02749  3.782110e-01
## CNAG_02781 CNAG_02781  4.198168e-02
## CNAG_02793 CNAG_02793  2.547088e-01
## CNAG_02806 CNAG_02806  3.691876e-01
## CNAG_02823 CNAG_02823  2.692441e-01
## CNAG_02849 CNAG_02849  9.225927e-02
## CNAG_02885 CNAG_02885  1.407529e-01
## CNAG_02900 CNAG_02900  2.459468e-01
## CNAG_02956 CNAG_02956  2.386243e-01
## CNAG_02977 CNAG_02977  1.716077e-01
## CNAG_02980 CNAG_02980  7.825500e-02
## CNAG_02987 CNAG_02987  2.633537e-01
```

```
## CNAG_03017 CNAG_03017  2.767971e-01
## CNAG_03091 CNAG_03091 -6.348255e-02
## CNAG_03169 CNAG_03169  3.774474e-01
## CNAG_03202 CNAG_03202  3.841128e-01
## CNAG_03292 CNAG_03292  4.543959e-02
## CNAG_03305 CNAG_03305 -1.039760e-01
## CNAG_03402 CNAG_03402  3.828460e-01
## CNAG_03406 CNAG_03406  1.735691e-01
## CNAG_03419 CNAG_03419  2.294355e-01
## CNAG_03422 CNAG_03422  3.509675e-01
## CNAG_03424 CNAG_03424 -1.166014e-01
## CNAG_03431 CNAG_03431  3.530677e-01
## CNAG_03443 CNAG_03443  1.896182e-01
## CNAG_03468 CNAG_03468  1.992477e-01
## CNAG_03509 CNAG_03509  2.812983e-01
## CNAG_03532 CNAG_03532 -2.537437e-01
## CNAG_03576 CNAG_03576  1.889824e-01
## CNAG_03617 CNAG_03617  1.573145e-01
## CNAG_03674 CNAG_03674  2.504451e-01
## CNAG_03726 CNAG_03726  3.132569e-01
## CNAG_03741 CNAG_03741  3.403265e-01
## CNAG_03744 CNAG_03744 -1.279045e-01
## CNAG_03811 CNAG_03811  2.674880e-01
## CNAG_03832 CNAG_03832  9.251836e-02
## CNAG_03833 CNAG_03833 -5.251663e-02
## CNAG_03905 CNAG_03905  6.194413e-02
## CNAG_03955 CNAG_03955  3.423809e-01
## CNAG_03971 CNAG_03971  1.888466e-01
## CNAG_03980 CNAG_03980  3.536349e-01
## CNAG_03997 CNAG_03997  1.890988e-01
## CNAG_04007 CNAG_04007  2.129770e-01
## CNAG_04031 CNAG_04031  1.778633e-01
## CNAG_04050 CNAG_04050  3.449066e-01
## CNAG_04081 CNAG_04081  3.757630e-01
## CNAG_04086 CNAG_04086  3.033157e-01
## CNAG_04097 CNAG_04097  1.587068e-01
## CNAG_04109 CNAG_04109  2.483965e-01
## CNAG_04140 CNAG_04140  1.835663e-02
## CNAG_04165 CNAG_04165  3.404659e-01
## CNAG_04262 CNAG_04262  3.176666e-01
## CNAG_04271 CNAG_04271  3.970535e-01
## CNAG_04281 CNAG_04281  3.234786e-01
## CNAG_04350 CNAG_04350  3.985733e-01
## CNAG_04404 CNAG_04404  1.495851e-01
## CNAG_04460 CNAG_04460  3.435662e-01
## CNAG_04479 CNAG_04479  1.460490e-01
## CNAG_04506 CNAG_04506  2.776187e-01
## CNAG_04537 CNAG_04537  3.593920e-01
## CNAG_04539 CNAG_04539  1.994140e-01
## CNAG_04540 CNAG_04540  2.688088e-01
## CNAG_04557 CNAG_04557  3.223962e-01
## CNAG_04699 CNAG_04699  9.281431e-02
## CNAG_04702 CNAG_04702 -1.400617e-02
## CNAG_04716 CNAG_04716  3.264905e-01
```

```
## CNAG_04719 CNAG_04719  2.397836e-01
## CNAG_04759 CNAG_04759  1.886516e-02
## CNAG_04761 CNAG_04761  3.485490e-01
## CNAG_04766 CNAG_04766  3.373714e-01
## CNAG_04810 CNAG_04810  3.727263e-01
## CNAG_04840 CNAG_04840  2.207499e-01
## CNAG_04890 CNAG_04890 -4.950420e-02
## CNAG_04900 CNAG_04900  2.475128e-01
## CNAG_04947 CNAG_04947  3.055978e-01
## CNAG_04950 CNAG_04950  3.900010e-01
## CNAG_04962 CNAG_04962  2.999782e-01
## CNAG_04967 CNAG_04967  3.166091e-01
## CNAG_04986 CNAG_04986  1.558174e-01
## CNAG_04988 CNAG_04988  1.909232e-01
## CNAG_05009 CNAG_05009  2.481647e-01
## CNAG_05021 CNAG_05021  2.143390e-01
## CNAG_05037 CNAG_05037  1.035740e-01
## CNAG_05040 CNAG_05040  2.838106e-01
## CNAG_05072 CNAG_05072  1.788704e-01
## CNAG_05081 CNAG_05081  3.373705e-01
## CNAG_05099 CNAG_05099  1.317543e-01
## CNAG_05135 CNAG_05135  3.207744e-01
## CNAG_05164 CNAG_05164  1.691597e-01
## CNAG_05194 CNAG_05194  1.496516e-01
## CNAG_05230 CNAG_05230  2.331098e-01
## CNAG_05240 CNAG_05240  3.945466e-01
## CNAG_05311 CNAG_05311  2.742011e-01
## CNAG_05318 CNAG_05318  1.850736e-01
## CNAG_05325 CNAG_05325  1.258573e-01
## CNAG_05358 CNAG_05358  3.137750e-01
## CNAG_05369 CNAG_05369  3.743993e-01
## CNAG_05427 CNAG_05427 -5.133329e-05
## CNAG_05428 CNAG_05428  3.277794e-01
## CNAG_05454 CNAG_05454  2.868652e-01
## CNAG_05466 CNAG_05466  3.564288e-01
## CNAG_05467 CNAG_05467  3.398154e-01
## CNAG_05486 CNAG_05486  2.564425e-01
## CNAG_05505 CNAG_05505  2.563496e-01
## CNAG_05514 CNAG_05514  2.809052e-01
## CNAG_05552 CNAG_05552  2.289937e-01
## CNAG_05579 CNAG_05579  3.459324e-01
## CNAG_05613 CNAG_05613  3.299630e-01
## CNAG_05621 CNAG_05621  2.693430e-01
## CNAG_05658 CNAG_05658  3.476488e-01
## CNAG_05666 CNAG_05666  3.359156e-01
## CNAG_05703 CNAG_05703  1.176975e-01
## CNAG_05728 CNAG_05728  2.357573e-01
## CNAG_05748 CNAG_05748  3.796804e-01
## CNAG_05787 CNAG_05787 -1.386521e-02
## CNAG_05799 CNAG_05799  3.441200e-01
## CNAG_05848 CNAG_05848  1.067201e-01
## CNAG_05869 CNAG_05869  2.234686e-01
## CNAG_05912 CNAG_05912  1.942076e-01
## CNAG_05941 CNAG_05941  1.945189e-01
```

```
## CNAG_05987 CNAG_05987  1.772977e-01
## CNAG_05995 CNAG_05995  3.478868e-01
## CNAG_06051 CNAG_06051  3.279481e-01
## CNAG_06078 CNAG_06078  3.341729e-01
## CNAG_06131 CNAG_06131  3.728981e-01
## CNAG_06149 CNAG_06149  3.913443e-01
## CNAG_06159 CNAG_06159  3.968300e-01
## CNAG_06178 CNAG_06178  3.246186e-01
## CNAG_06181 CNAG_06181  2.478706e-01
## CNAG_06225 CNAG_06225 -1.402368e-01
## CNAG_06234 CNAG_06234 -2.201734e-03
## CNAG_06245 CNAG_06245  1.451586e-01
## CNAG_06312 CNAG_06312  3.638908e-01
## CNAG_06321 CNAG_06321  3.908873e-01
## CNAG_06325 CNAG_06325  3.147078e-01
## CNAG_06390 CNAG_06390  3.895181e-01
## CNAG_06394 CNAG_06394  2.213326e-01
## CNAG_06395 CNAG_06395  3.657921e-01
## CNAG_06405 CNAG_06405  2.965948e-01
## CNAG_06418 CNAG_06418  1.677466e-01
## CNAG_06438 CNAG_06438  3.705356e-01
## CNAG_06445 CNAG_06445  2.877032e-01
## CNAG_06487 CNAG_06487  3.309921e-01
## CNAG_06503 CNAG_06503 -1.181366e-01
## CNAG_06556 CNAG_06556  6.455825e-02
## CNAG_06562 CNAG_06562 -1.618721e-02
## CNAG_06598 CNAG_06598  3.788810e-01
## CNAG_06613 CNAG_06613  1.569278e-01
## CNAG_06624 CNAG_06624  3.210992e-01
## CNAG_06636 CNAG_06636  3.153427e-01
## CNAG_06676 CNAG_06676  1.109538e-01
## CNAG_06677 CNAG_06677  2.286778e-01
## CNAG_06704 CNAG_06704  7.105312e-02
## CNAG_06707 CNAG_06707  1.127065e-01
## CNAG_06754 CNAG_06754  3.859480e-01
## CNAG_06784 CNAG_06784  2.919110e-01
## CNAG_06795 CNAG_06795  3.751910e-01
## CNAG_06819 CNAG_06819  3.033581e-01
## CNAG_06854 CNAG_06854  3.962596e-01
## CNAG_06877 CNAG_06877  3.559083e-01
## CNAG_06911 CNAG_06911  3.968827e-01
## CNAG_06918 CNAG_06918  3.395590e-02
## CNAG_07002 CNAG_07002  4.659781e-03
## CNAG_07276 CNAG_07276  2.396284e-01
## CNAG_07303 CNAG_07303  9.552834e-02
## CNAG_07304 CNAG_07304 -1.640711e-01
## CNAG_07349 CNAG_07349  2.939476e-01
## CNAG_07417 CNAG_07417  2.969828e-01
## CNAG_07422 CNAG_07422  3.076527e-01
## CNAG_07440 CNAG_07440  1.027769e-02
## CNAG_07447 CNAG_07447  1.027626e-01
## CNAG_07472 CNAG_07472  3.352994e-01
## CNAG_07496 CNAG_07496 -6.743938e-02
## CNAG_07530 CNAG_07530  6.469776e-02
```

```
## CNAG_07534 CNAG_07534  3.807738e-01
## CNAG_07565 CNAG_07565  3.362056e-01
## CNAG_07575 CNAG_07575  3.001980e-01
## CNAG_07584 CNAG_07584  2.054313e-01
## CNAG_07589 CNAG_07589  3.807679e-01
## CNAG_07592 CNAG_07592  3.197335e-01
## CNAG_07605 CNAG_07605  3.709795e-01
## CNAG_07610 CNAG_07610  1.336242e-01
## CNAG_07613 CNAG_07613 -2.357805e-02
## CNAG_07653 CNAG_07653 -7.928088e-02
## CNAG_07659 CNAG_07659  1.303829e-01
## CNAG_07663 CNAG_07663 -9.701127e-02
## CNAG_07666 CNAG_07666  1.798599e-01
## CNAG_07678 CNAG_07678  3.706454e-01
## CNAG_07679 CNAG_07679  1.847163e-01
## CNAG_07680 CNAG_07680  2.692214e-01
## CNAG_07698 CNAG_07698 -1.860608e-02
## CNAG_07706 CNAG_07706  3.050545e-01
## CNAG_07711 CNAG_07711  2.196265e-01
## CNAG_07725 CNAG_07725  3.013601e-01
## CNAG_07757 CNAG_07757  2.583875e-01
## CNAG_07759 CNAG_07759  1.635757e-01
## CNAG_07774 CNAG_07774  2.427041e-01
## CNAG_07793 CNAG_07793  2.131321e-01
## CNAG_07809 CNAG_07809  1.338681e-01
## CNAG_07816 CNAG_07816  1.442789e-01
## CNAG_07819 CNAG_07819  2.323956e-01
## CNAG_07829 CNAG_07829  3.264116e-01
## CNAG_07832 CNAG_07832  2.510471e-01
## CNAG_07842 CNAG_07842  2.473916e-01
## CNAG_07843 CNAG_07843  3.509210e-01
## CNAG_07850 CNAG_07850  3.463735e-01
## CNAG_07859 CNAG_07859  2.424328e-01
## CNAG_07876 CNAG_07876  2.976882e-01
## CNAG_07893 CNAG_07893  2.535931e-01
## CNAG_07900 CNAG_07900  2.017707e-01
## CNAG_07919 CNAG_07919 -1.158606e-01
## CNAG_07920 CNAG_07920  3.506807e-01
## CNAG_07934 CNAG_07934  4.962073e-02
## CNAG_07950 CNAG_07950  3.487926e-01
## CNAG_07952 CNAG_07952  2.497599e-01
## CNAG_07955 CNAG_07955 -2.339366e-01
## CNAG_07970 CNAG_07970  2.079559e-01
## CNAG_07983 CNAG_07983  2.942438e-01
## CNAG_07988 CNAG_07988  1.421027e-02
## CNAG_08000 CNAG_08000  1.599219e-01
## CNAG_08004 CNAG_08004  7.486088e-02
## CNAG_08017 CNAG_08017 -1.594278e-01
```

```r
# Export as CSV file
write.csv(low_correlation_genes_df, "low_correlation_genes.csv", row.names = FALSE)


print("The low-correlation genes and their correlation coefficients have been derived as low_correlatio
```

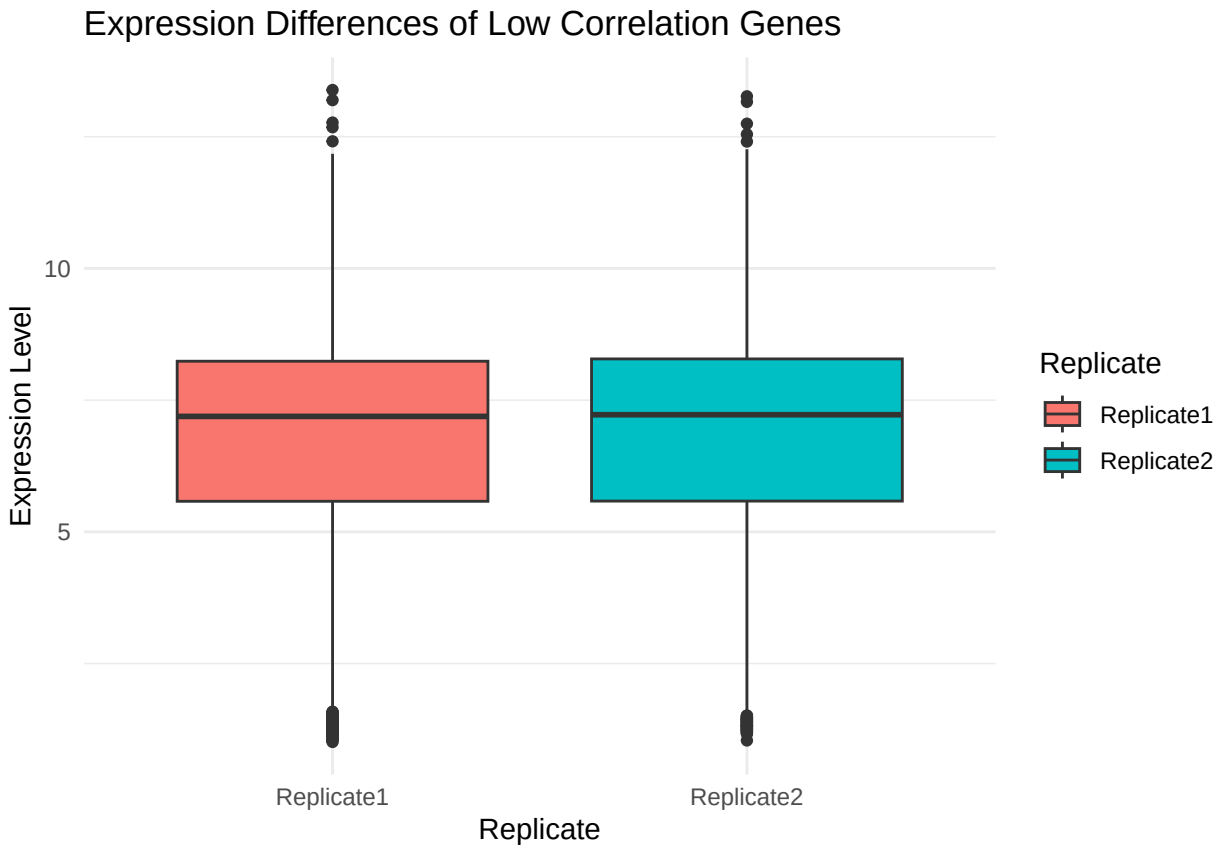
```
## [1] "The low-correlation genes and their correlation coefficients have been derived as low_correlatio
```

```
threshold02 <- 0.75
low_correlation_genes02 <- names(gene_correlations[gene_correlations < threshold02])

# Extract low-correlation genes and their corresponding correlation coefficients
low_correlation_genes_df02 <- data.frame(
  Gene = names(gene_correlations[low_correlation_genes02]),
  Correlation = gene_correlations[low_correlation_genes02]
)

# Export as CSV file
write.csv(low_correlation_genes_df02, "low_correlation_genes02.csv", row.names = FALSE)
```

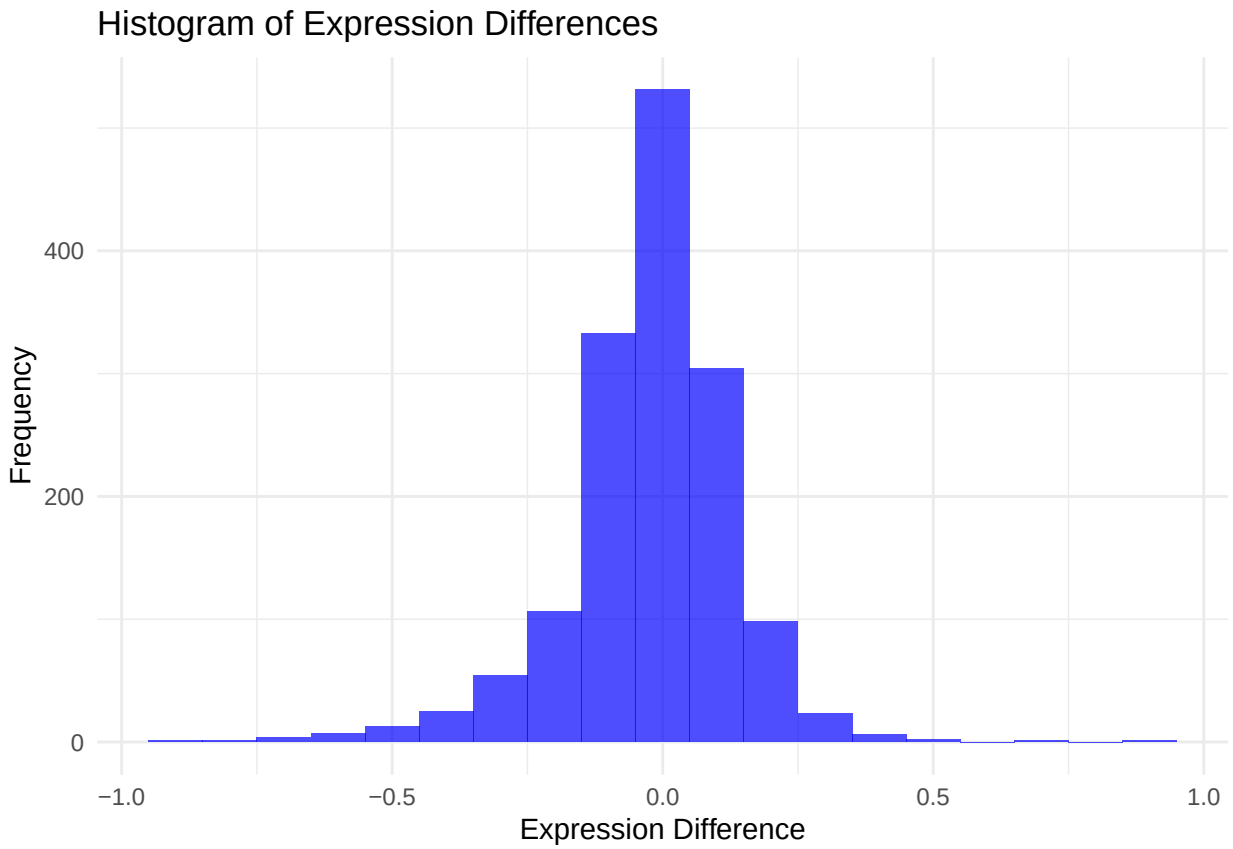
```
#Explore whether there is a systematic bias
#The number of genes with low correlation accounts for about 25% of the total, which is too much
#Extract the expression data of low-correlated genes in two replications
low_correlation_replicate1_data <- replicate1_data[low_correlation_genes02, ]
low_correlation_replicate2_data <- replicate2_data[low_correlation_genes02, ]

# Calculate expression differences
expression_diff <- rowMeans(low_correlation_replicate1_data) - rowMeans(low_correlation_replicate2_data)
expression_diff_df <- data.frame(
  Gene = low_correlation_genes02,
  ExpressionDiff = expression_diff
)

# View the first few lines of data
head(expression_diff_df)
```

```
##                  Gene ExpressionDiff
## CNAG_00013 CNAG_00013     0.06668942
## CNAG_00014 CNAG_00014    -0.04653233
## CNAG_00018 CNAG_00018    -0.07082300
## CNAG_00020 CNAG_00020    -0.01481215
## CNAG_00025 CNAG_00025    -0.06638373
## CNAG_00027 CNAG_00027    -0.04318767
```

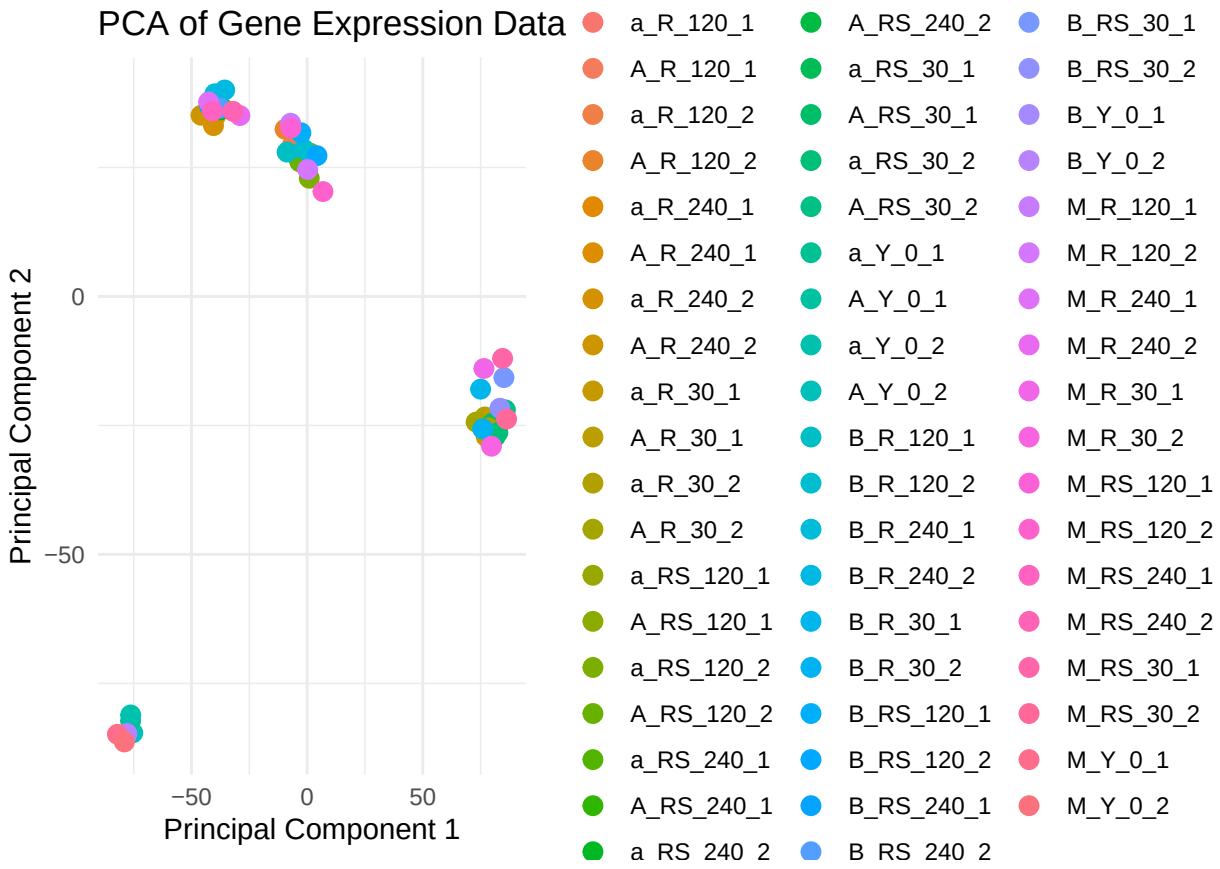
```
# Create long format data frame
expression_diff_long <- data.frame(
  Gene = rep(low_correlation_genes02, 2),
  Expression = c(rowMeans(low_correlation_replicate1_data), rowMeans(low_correlation_replicate2_data)),
  Replicate = rep(c("Replicate1", "Replicate2"), each = length(low_correlation_genes02))
)

# Draw a box plot
ggplot(expression_diff_long, aes(x = Replicate, y = Expression, fill = Replicate)) +
  geom_boxplot() +
  labs(title = "Expression Differences of Low Correlation Genes",
       x = "Replicate",
       y = "Expression Level") +
  theme_minimal()
```



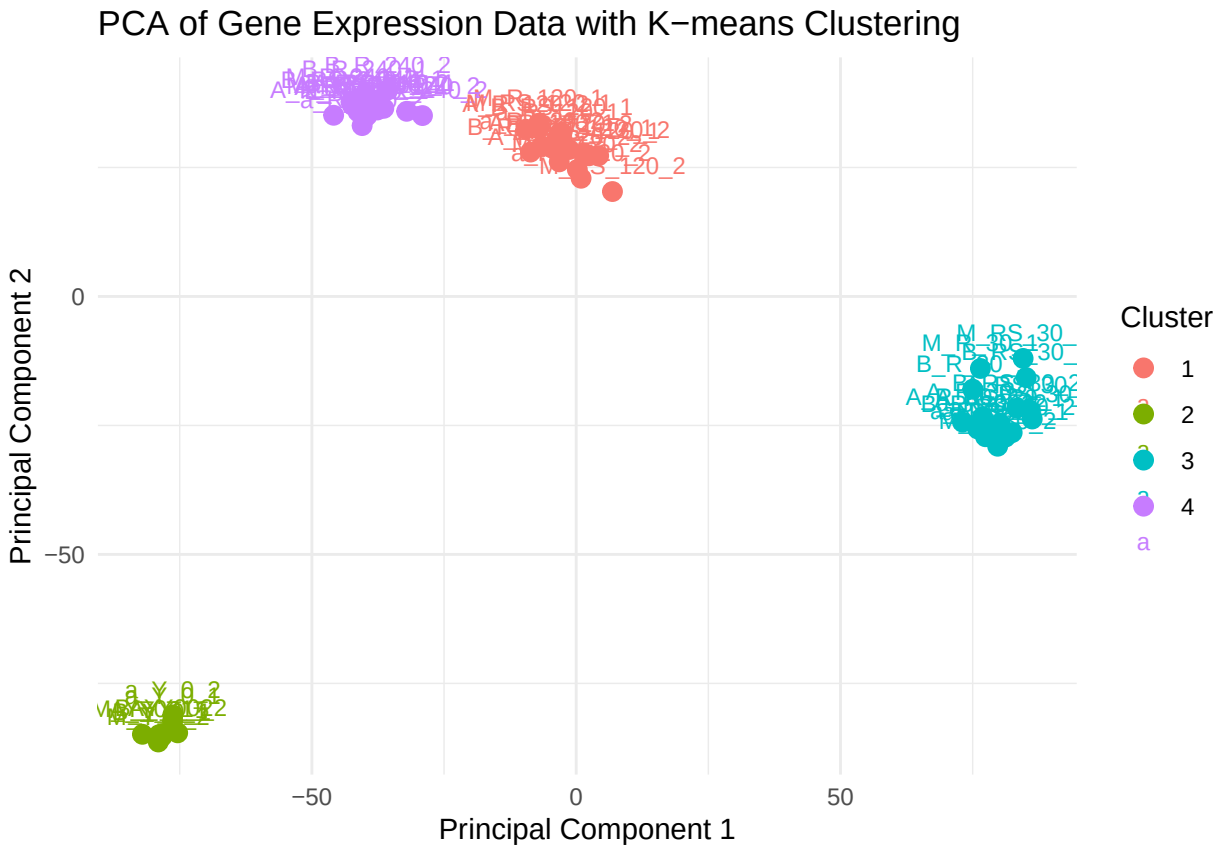

```
# Draw a difference histogram
ggplot(expression_diff_df, aes(x = ExpressionDiff)) +
  geom_histogram(binwidth = 0.1, fill = "blue", alpha = 0.7) +
  labs(title = "Histogram of Expression Differences",
       x = "Expression Difference",
       y = "Frequency") +
  theme_minimal()
```




```
# Extract the standardised data
normalized_counts <- assay(vsd)
# PCA analysis
pca_res <- prcomp(t(normalized_counts))

# Extract the first two principal components
pca_data <- as.data.frame(pca_res$x)
pca_data$Condition <- rownames(pca_data)

# Visualisation of PCA results
library(ggplot2)
ggplot(pca_data, aes(x = PC1, y = PC2, color = Condition)) +
  geom_point(size = 3) +
  labs(title = "PCA of Gene Expression Data",
       x = "Principal Component 1",
       y = "Principal Component 2") +
  theme_minimal()
```




```r
# Write out the different groups
# Set k to 4 because there appear to be four clusters in the graph
set.seed(123)  # Set a random seed to ensure reproducible results
k <- 4
kmeans_res <- kmeans(pca_data[, 1:2], centers = k)

# Add the clustering result to the data frame
pca_data$Cluster <- as.factor(kmeans_res$cluster)

# Visualise PCA results and label clusters
ggplot(pca_data, aes(x = PC1, y = PC2, color = Cluster, label = Condition)) +
  geom_point(size = 3) +
  geom_text(vjust = -1, size = 3) +
  labs(title = "PCA of Gene Expression Data with K-means Clustering",
       x = "Principal Component 1",
       y = "Principal Component 2") +
  theme_minimal()
```

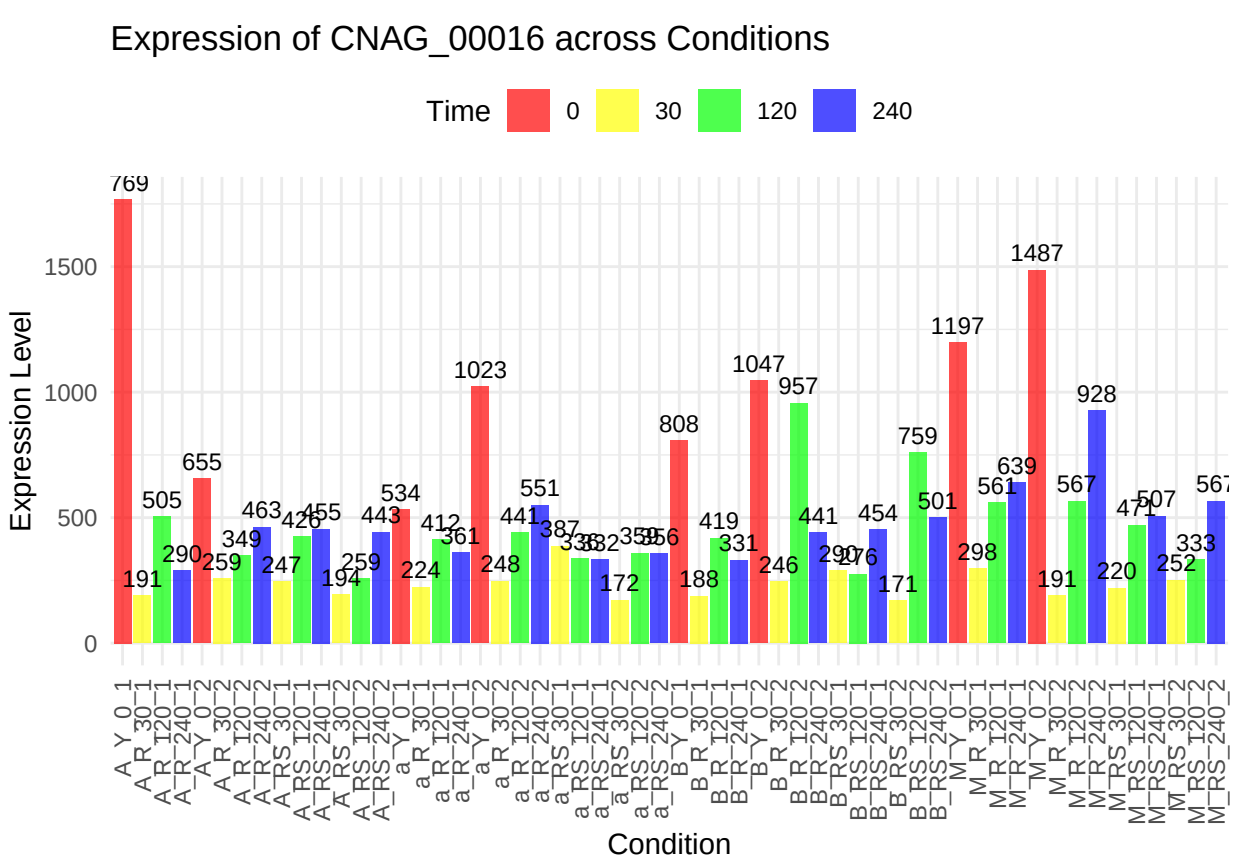



```
# List the condition names for each cluster
cluster_conditions <- split(pca_data$Condition, pca_data$Cluster)

# Print the condition name for each cluster
for (i in 1:k) {
  cat("Cluster", i, "conditions:\n")
  print(cluster_conditions[[i]])
  cat("\n")
}
```

```
## Cluster 1 conditions:
##  [1] "A_R_120_1"  "A_R_120_2"  "A_RS_120_1" "A_RS_120_2" "a_R_120_1"
##  [6] "a_R_120_2"  "a_RS_120_1" "a_RS_120_2" "B_R_120_1"  "B_R_120_2"
## [11] "B_RS_120_1" "B_RS_120_2" "M_R_120_1"  "M_R_120_2"  "M_RS_120_1"
## [16] "M_RS_120_2"
##
## Cluster 2 conditions:
## [1] "A_Y_0_1" "A_Y_0_2" "a_Y_0_1" "a_Y_0_2" "B_Y_0_1" "B_Y_0_2" "M_Y_0_1"
## [8] "M_Y_0_2"
##
## Cluster 3 conditions:
##  [1] "A_R_30_1"  "A_R_30_2"  "A_RS_30_1" "A_RS_30_2" "a_R_30_1"  "a_R_30_2"
##  [7] "a_RS_30_1" "a_RS_30_2" "B_R_30_1"  "B_R_30_2"  "B_RS_30_1" "B_RS_30_2"
## [13] "M_R_30_1"  "M_R_30_2"  "M_RS_30_1" "M_RS_30_2"
##
## Cluster 4 conditions:
##  [1] "A_R_240_1"  "A_R_240_2"  "A_RS_240_1" "A_RS_240_2" "a_R_240_1"
```

```
##  [6] "a_R_240_2"  "a_RS_240_1" "a_RS_240_2" "B_R_240_1"  "B_R_240_2"
## [11] "B_RS_240_1" "B_RS_240_2" "M_R_240_1"  "M_R_240_2"  "M_RS_240_1"
## [16] "M_RS_240_2"
```

```r
#1. Analyse the behaviour of individual genes
# Define a function to extract and merge gene expression data
get_gene_data <- function(gene_name, dds) {
  # Extract gene expression data
  gene_expression <- counts(dds)[gene_name, ]

  # Combine expression data and conditional information
  gene_data <- data.frame(
    Expression = gene_expression,
    Condition = colnames(counts(dds))
  )

  return(gene_data)
}

gene_data <- get_gene_data("CNAG_00016",dds)

# Define a function to draw a histogram
plot_gene_expression <- function(gene_name, dds) {
  # Obtain genetic data
  gene_data <- get_gene_data(gene_name, dds)
  # Make sure that the Condition column is of the factor type and sorted in the original data order.
  gene_data$Condition <- factor(gene_data$Condition, levels = unique(gene_data$Condition))

  # Extract the second-to-last digit of the Condition
  gene_data$Time <- as.numeric(sub(".*_(\\d+)_.*", "\\1", gene_data$Condition))

  color_palette <- c("0" = "red", "30" = "yellow", "120" = "green", "240" = "blue")

  # Draw a bar chart and optimise it
  p <- ggplot(gene_data, aes(x = Condition, y = Expression, fill = as.factor(Time))) +
    geom_bar(stat = "identity", alpha = 0.7) +
    geom_text(aes(label = round(Expression, 2)), vjust = -0.5, size = 3) +
    scale_fill_manual(values = color_palette, name = "Time") +
    labs(title = paste("Expression of", gene_name, "across Conditions"),
         x = "Condition",
         y = "Expression Level") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1),
          legend.position = "top")

  return(p)
}
plot_gene_expression("CNAG_00016",dds)
```



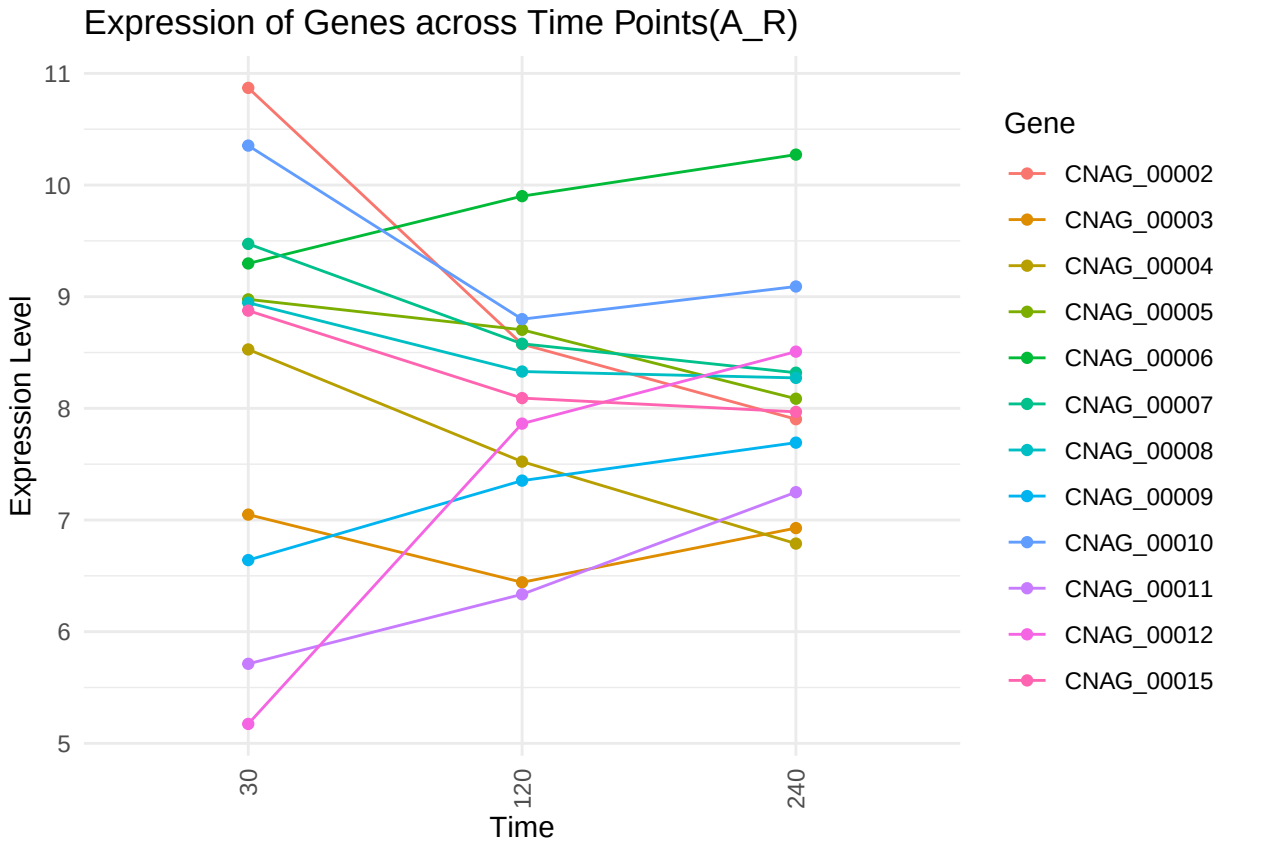

```
###################
#Explore the expression behaviour of different genes under the same conditions
#First, remove the gene data with a correlation coefficient of less than 0.75 between the first and sec
#because this part is flawed in the experiment and is not conducive to exploration
# Remove low correlation genes
filtered_counts <- counts(dds)[!rownames(counts(dds)) %in% low_correlation_genes02, ]
filtered_dds <- dds[!rownames(dds) %in% low_correlation_genes02, ]

# The original data contains data for 6795 genes. Why are there only 6756 genes after VSD?
# Possible reasons
# Filtering of lowly expressed genes:
#
# When using DESeq2 for analysis, lowly expressed genes are often filtered out. For example, the code d
# Missing data handling:
#
# During some preprocessing steps, genes may be filtered out if they are not detected in all samples (i
# Variance stabilisation (VSD):
#
# During variance stabilisation, DESeq2 may further filter out some genes, especially those that are no

# Performing variance stabilisation
filtered_vsd <- vst(filtered_dds, blind = FALSE)
# Performing a Variance Stabilising Transformation (VST) can help us better deal with some of the chara
#
# Reasons and effects
# Standardising data: RNA-seq data often has a high degree of variability and a wide dynamic range. VST
#
```

```r
# Reduce noise: In RNA-seq data, low-expression genes often have high noise levels. VST can reduce this
#
# Suitable for downstream analysis: VST data is more suitable for downstream statistical analysis and v
colnames(assay(filtered_vsd))
```

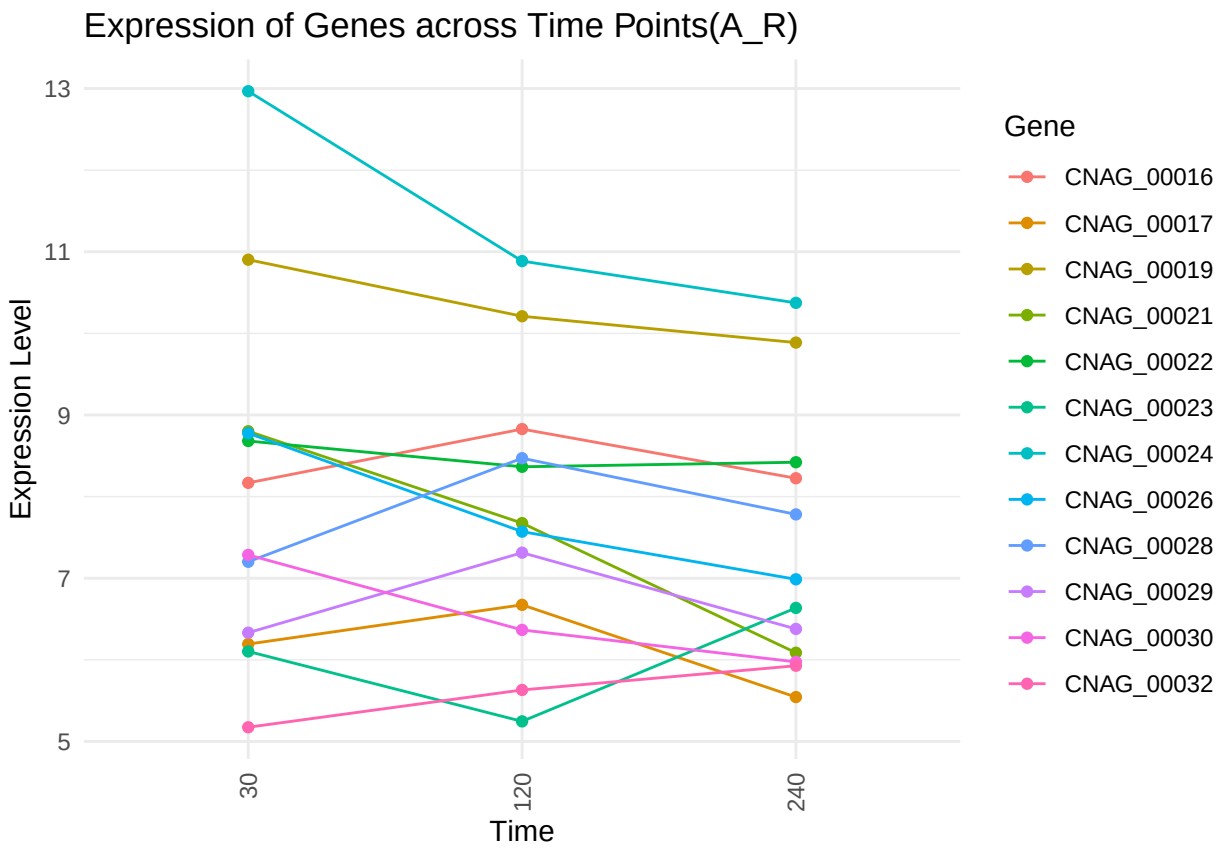
```
##  [1] "A_Y_0_1"    "A_R_30_1"   "A_R_120_1"  "A_R_240_1"  "A_Y_0_2"
##  [6] "A_R_30_2"   "A_R_120_2"  "A_R_240_2"  "A_RS_30_1"  "A_RS_120_1"
## [11] "A_RS_240_1" "A_RS_30_2"  "A_RS_120_2" "A_RS_240_2" "a_Y_0_1"
## [16] "a_R_30_1"   "a_R_120_1"  "a_R_240_1"  "a_Y_0_2"    "a_R_30_2"
## [21] "a_R_120_2"  "a_R_240_2"  "a_RS_30_1"  "a_RS_120_1" "a_RS_240_1"
## [26] "a_RS_30_2"  "a_RS_120_2" "a_RS_240_2" "B_Y_0_1"    "B_R_30_1"
## [31] "B_R_120_1"  "B_R_240_1"  "B_Y_0_2"    "B_R_30_2"   "B_R_120_2"
## [36] "B_R_240_2"  "B_RS_30_1"  "B_RS_120_1" "B_RS_240_1" "B_RS_30_2"
## [41] "B_RS_120_2" "B_RS_240_2" "M_Y_0_1"    "M_R_30_1"   "M_R_120_1"
## [46] "M_R_240_1"  "M_Y_0_2"    "M_R_30_2"   "M_R_120_2"  "M_R_240_2"
## [51] "M_RS_30_1"  "M_RS_120_1" "M_RS_240_1" "M_RS_30_2"  "M_RS_120_2"
## [56] "M_RS_240_2"
```

```r
#View column names to prevent errors
test01 <- as.data.frame(assay(filtered_vsd)[,c("A_R_30_1","A_R_120_1","A_R_240_1")])
Gene <- rownames(test01)
test01 <- cbind(Gene,test01)


library(ggplot2)
library(readr)
library(tidyr)
library(dplyr)

# # Convert data to long format
test01_long <- test01 %>%
  pivot_longer(cols = -Gene, names_to = c("Condition1", "Condition2", "Time", "Replicate"),
               names_pattern = "(.*)_([^_]+)_(\\d+)_(\\d+)$", values_to = "Expression") %>%
  unite("Condition", Condition1, Condition2, sep = "_")

# Make sure that Time is a numeric type
test01_long$Time <- factor(test01_long$Time, levels = c(30, 120, 240))
# Plot the chart
#plot_batch(test01_long)
# The data volume is too large to be plotted, and R is stuck


# For debugging
# # View the converted data structure
# str(test01_long)
# head(test01_long)

# Function to plot a line chart
plot_expression_data <- function(data) {
  p <- ggplot(data, aes(x = Time, y = Expression, color = Gene, group = Gene)) +
    geom_line() +
    geom_point() +
    labs(title = "Expression of Genes across Time Points(A_R)",
         x = "Time",
```

```
        y = "Expression Level") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))

  return(p)
}

plot_expression_data(test01_long[1:36,])
```




```
plot_expression_data(test01_long[37:72,])
```


Expression of Genes across Time Points(A_R)
Expression Level
13
11
9
7
5
30
120
240
Time
Gene
CNAG_00016
CNAG_00017
CNAG_00019
CNAG_00021
CNAG_00022
CNAG_00023
CNAG_00024
CNAG_00026
CNAG_00028
CNAG_00029
CNAG_00030
CNAG_00032