Programming Exercises for R

Julia Hu(hslhu@outlook.com)

2025-08-24

```
# loading the library
library(dplyr)
library(tidyverse)
library(tibble)
library(ggplot2)
library(DESeq2)
library(limma)
library(pheatmap)
library(biomaRt)
library(reshape2)
library(conflicted)

conflicts_prefer(dplyr::select)
conflicts_prefer(dplyr::filter)
conflicts_prefer(Biobase::rowMedians)
conflicts_prefer(MatrixGenerics::colMedians)
conflicts_prefer(base::intersect)
```

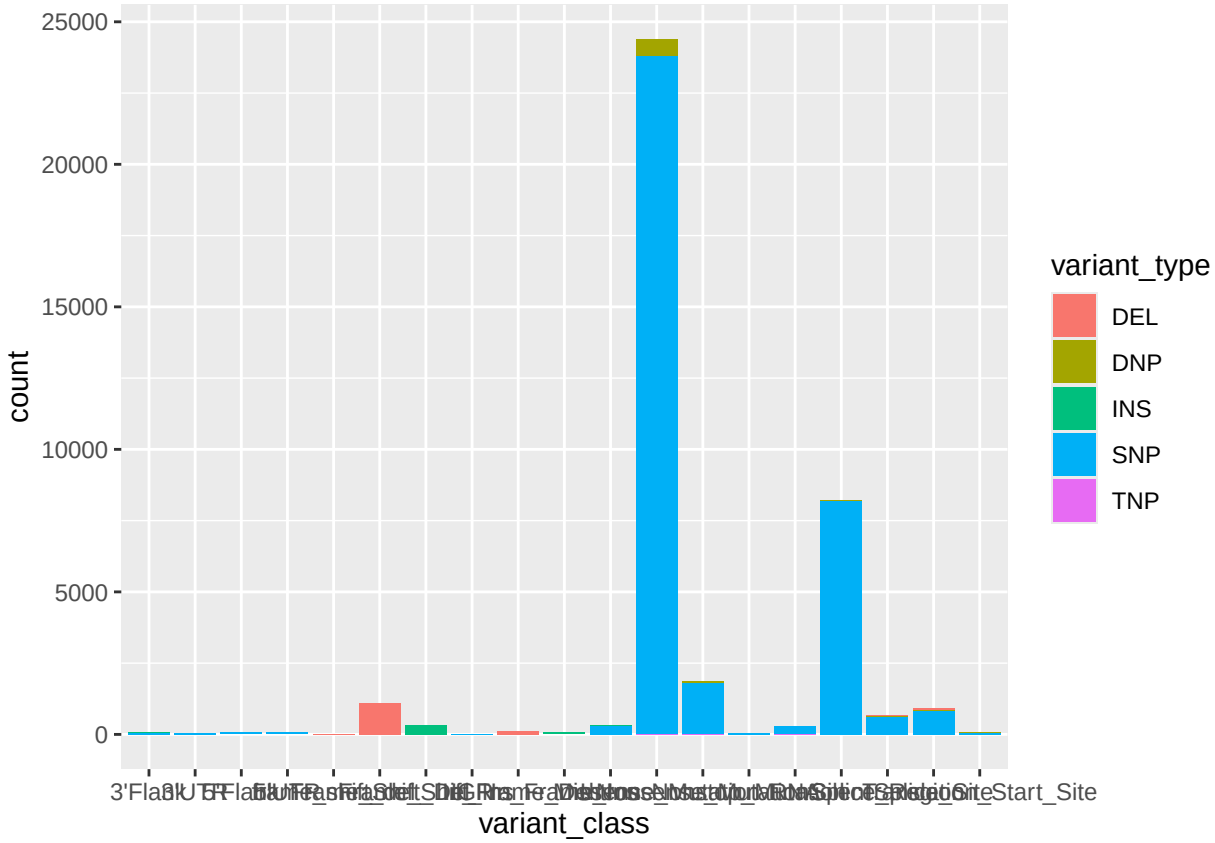
Mutation data analysis

1. Identify the top 10 most frequently mutated genes. Identify samples whose mutation count is in the 80 to 90 percentile.

```
# read rds data
setwd("exercise/") # set working directory
data <- readRDS("mutations_sclc_ucologne_2015.rds")
head(data,3)
```

```
##     gene chrom start_pos   end_pos strand     variant_class variant_type ref
## 1 MARCH1     4 164507073 164507073      + Missense_Mutation          SNP   T
## 2 MARCH1     1 220971347 220971347      +            Silent          SNP   C
## 3 MARCH1     4 164507055 164507055      + Missense_Mutation          SNP   T
##   alt dbsnp_rs                sample_id                      hgvsc   hgvsp
## 1   T       NA sclc_ucologne_2015_S00934 ENST00000274056.7:c.251A>G  p.H84R
## 2   C       NA sclc_ucologne_2015_S02249 ENST00000366910.5:c.744C>T p.V248=
## 3   T       NA sclc_ucologne_2015_S02285 ENST00000274056.7:c.269A>G  p.E90G
##     transcript_id protein_pos  codons
## 1 ENST00000274056          84 cAc/cGc
## 2 ENST00000366910         248 gtC/gtT
## 3 ENST00000274056          90 gAg/gGg
```

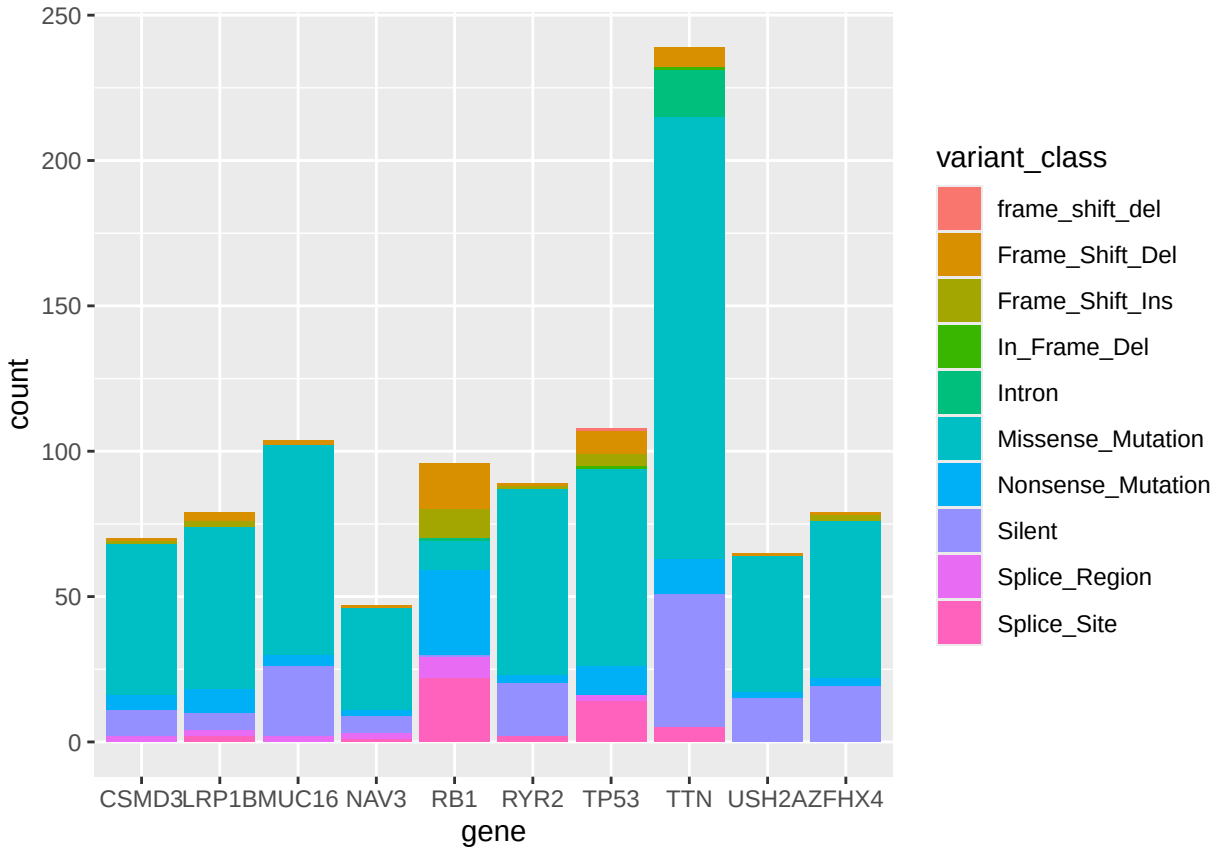
```r
# visualized the variant data
ggplot(data, aes(x = variant_class,fill = variant_type)) +
  geom_bar() # bar plot
```




```r
# count for top 10 genes
top_genes <- data %>%
  group_by(gene) %>%
  summarise(count = n()) %>%  # count variant for every gene
  arrange(desc(count)) %>%    # sort in descending order
  head(10)

print(top_genes)
```

```
## # A tibble: 10 x 2
##    gene  count
##    <chr> <int>
##  1 TTN     239
##  2 TP53    108
##  3 MUC16   104
##  4 RB1      96
##  5 RYR2     89
##  6 LRP1B    79
##  7 ZFHX4    79
##  8 CSMD3    70
##  9 USH2A    65
## 10 NAV3     47
```

```
# top 10 genes variant class
result <- data %>%
  filter(gene %in% top_genes$gene) %>%  # filter top genes in gene data
  select(gene, variant_class)   %>%    # select genes and variants
  group_by(gene,variant_class) %>%
  summarise(count = n())

ggplot(result,aes(x = gene, y = count, fill = variant_class)) +
  geom_bar(stat="identity")
```



Identify samples whose mutation count is in the 80 to 90 percentile.

```
# Identify samples whose mutation count is in the 80 to 90 percentile.

sample_data <- data %>%
  group_by(sample_id) %>%  # group by samples id
  summarise(Count = n())   # count mutation

head(sample_data)
```

```
## # A tibble: 6 x 2
##   sample_id                Count
##   <chr>                    <int>
## 1 sclc_ucologne_2015_S00022   256
## 2 sclc_ucologne_2015_S00035   308
## 3 sclc_ucologne_2015_S00050   418
```

```
## 4 sclc_ucologne_2015_S00339   112
## 5 sclc_ucologne_2015_S00356   260
## 6 sclc_ucologne_2015_S00472   642
```

```
# find 80 to 90 percentile
sample_data$sample_id[sample_data$Count >= quantile(sample_data$Count, 0.8) & # select 80 percentile
                      sample_data$Count <= quantile(sample_data$Count, 0.9)]   # and 90 percentile
```

```
##  [1] "sclc_ucologne_2015_S00841" "sclc_ucologne_2015_S01020"
##  [3] "sclc_ucologne_2015_S01022" "sclc_ucologne_2015_S01023"
##  [5] "sclc_ucologne_2015_S01861" "sclc_ucologne_2015_S02242"
##  [7] "sclc_ucologne_2015_S02248" "sclc_ucologne_2015_S02285"
##  [9] "sclc_ucologne_2015_S02328" "sclc_ucologne_2015_S02344"
## [11] "sclc_ucologne_2015_S02376" "sclc_ucologne_2015_S02384"
```

2. Categorize variants based on their expected effects using the *data/mutation_effects.tsv* table. Generate a count matrix containing the numbers of loss-of-function and neutral mutations for each gene.

```
# read tsv data
df <- read.table("exercise/mutation_effects.tsv",header =T)
head(df)
```

```
##       variant_class           effect
## 1   Frame_Shift_Del loss_of_function
## 2   Frame_Shift_Ins loss_of_function
## 3      In_Frame_Del        uncertain
## 4      In_Frame_Ins        uncertain
## 5            Intron          neutral
## 6 Missense_Mutation        uncertain
```

```
# left join the data
data_effect <- data %>%
  left_join(df, by = "variant_class") %>%
  mutate(effect = coalesce(effect, "Missing")) # label missing value
head(data_effect,3)
```

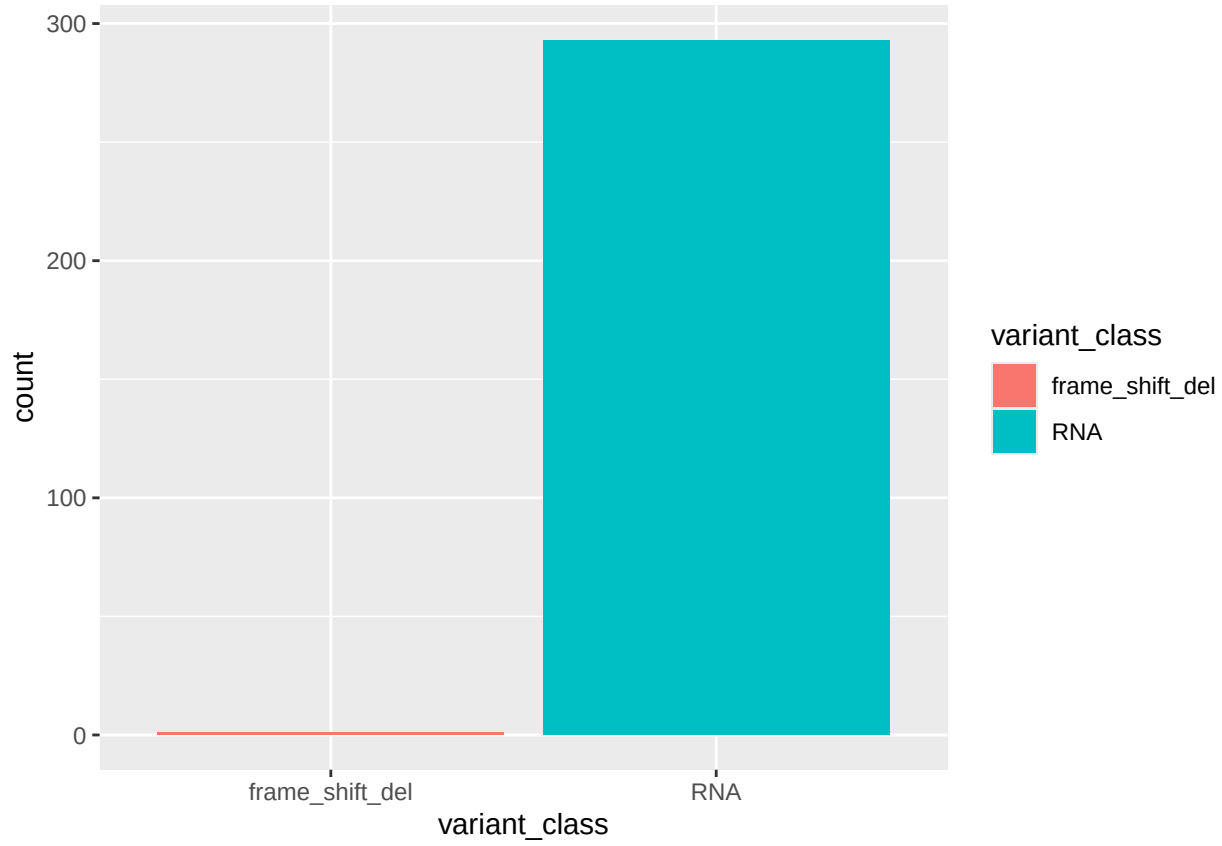
```
##     gene chrom start_pos   end_pos strand     variant_class variant_type ref
## 1 MARCH1     4 164507073 164507073      + Missense_Mutation          SNP   T
## 2 MARCH1     1 220971347 220971347      +            Silent          SNP   C
## 3 MARCH1     4 164507055 164507055      + Missense_Mutation          SNP   T
##   alt dbsnp_rs                 sample_id                      hgvsc   hgvsp
## 1   T       NA sclc_ucologne_2015_S00934 ENST00000274056.7:c.251A>G  p.H84R
## 2   C       NA sclc_ucologne_2015_S02249 ENST00000366910.5:c.744C>T p.V248=
## 3   T       NA sclc_ucologne_2015_S02285 ENST00000274056.7:c.269A>G  p.E90G
##     transcript_id protein_pos  codons    effect
## 1 ENST00000274056          84 cAc/cGc uncertain
## 2 ENST00000366910         248 gtC/gtT   neutral
## 3 ENST00000274056          90 gAg/gGg uncertain
```

Explore the missing data

```
# select missing_data
missing_data <- data_effect[data_effect$effect == "Missing",]
head(missing_data,3)
```

```
##        gene chrom start_pos   end_pos strand variant_class variant_type ref alt
## 333     ABO     9 136150594 136150594      +           RNA          SNP   C   C
## 1048 AGAP11    10  88769176  88769176      +           RNA          SNP   A   A
## 1049 AGAP11    10  88768734  88768734      +           RNA          SNP   C   C
##      dbsnp_rs                sample_id                      hgvsc hgvsp
## 333        NA sclc_ucologne_2015_S01578   ENST00000453660.2:n.24G>T   *8*
## 1048       NA sclc_ucologne_2015_S00022 ENST00000433214.2:n.2371A>G *791*
## 1049       NA sclc_ucologne_2015_S00827 ENST00000433214.2:n.1929C>A *643*
##        transcript_id protein_pos codons  effect
## 333  ENST00000453660          NA        Missing
## 1048 ENST00000433214          NA        Missing
## 1049 ENST00000433214          NA        Missing
```

```
# plot the missing data
ggplot(missing_data, aes(x = variant_class, fill = variant_class)) +
  geom_bar()
```




```
# check the variant class
data_effect[data_effect$variant_class == "frame_shift_del",] # in gene TP53
```

```
##       gene chrom start_pos end_pos strand   variant_class variant_type
## 34327 TP53    17   7579300 7579331      + frame_shift_del          DEL
##                                    ref                               alt
## 34327 AGGGCAACTGACCGTGCAAGTCACAGACTTGG AGGGCAACTGACCGTGCAAGTCACAGACTTGG
##       dbsnp_rs                sample_id hgvsc     hgvsp transcript_id
## 34327       NA sclc_ucologne_2015_S02385       p.A119fs.
##       protein_pos codons  effect
```

```
## 34327         119        Missing
```

```
# 'frame_shift_del' has misspelling
unique(data_effect$variant_class[data_effect$gene == "TP53"])
```

```
## [1] "Splice_Site"       "Missense_Mutation" "Frame_Shift_Ins"
## [4] "Nonsense_Mutation" "Splice_Region"     "Frame_Shift_Del"
## [7] "In_Frame_Del"      "frame_shift_del"
```

```
# correct the variant class
data_effect$variant_class[data_effect$variant_class == 'frame_shift_del'] <-
    "Frame_Shift_Del"

## correct the effect data
data_effect$effect[data_effect$variant_class == "Frame_Shift_Del"
                   & data_effect$effect == "Missing"] <- 'loss_of_function'
```

Generate a count matrix

```
# Generate a count matrix
count_effect <- data_effect %>%
  filter(effect == "loss_of_function"| effect =="neutral",) %>%
  select(gene,effect)
```

```
# create a count table for the each gene
mutation_count <- table(count_effect)
head(mutation_count)
```

```
##          effect
## gene      loss_of_function neutral
##   A1CF                    0       1
##   A2ML1                   0       2
##   A4GALT                  0       1
##   A4GNT                   1       0
##   AACS                    1       0
##   AADACL4                 1       0
```

3. Implement a statistical test that determines whether a gene has a significantly higher proportion of loss-of-function mutations (excluding mutations with uncertain effects), compared to other genes.

Why fisher test? **For the significant differences in proportions, I chose Fisher's exact test over the chi-squared test because most of the value in the contingency table had observed counts <5. This indicates expected frequencies would likely violate the chi-squared test's requirement (all 5). Fisher's test is robust to small sample sizes, making it more appropriate here.**

```
# convert to data frame
df_mutation_count <- melt(mutation_count)

head(df_mutation_count)
```

```
##      gene           effect value
## 1    A1CF loss_of_function     0
## 2   A2ML1 loss_of_function     0
## 3  A4GALT loss_of_function     0
```

```
## 4   A4GNT loss_of_function     1
## 5    AACS loss_of_function     1
## 6 AADACL4 loss_of_function     1
```

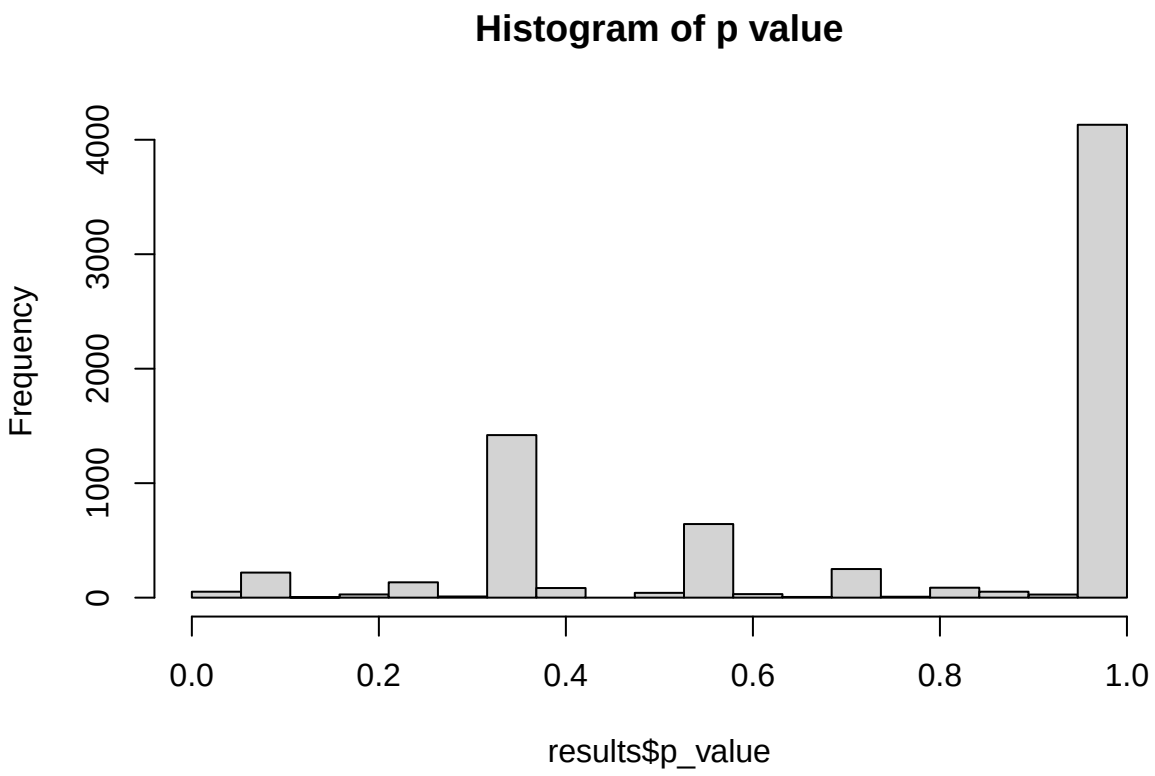
```
# convert to wide data for count
df_count <- pivot_wider(df_mutation_count,names_from = effect,values_from = value)
head(df_count)
```

```
## # A tibble: 6 x 3
##   gene    loss_of_function neutral
##   <fct>              <int>   <int>
## 1 A1CF                   0       1
## 2 A2ML1                  0       2
## 3 A4GALT                 0       1
## 4 A4GNT                  1       0
## 5 AACS                   1       0
## 6 AADACL4                1       0
```

```
# run for each gene vs others gene proportion
results <- data.frame(   # set a data frame for results
  gene = character(),
  p_value = numeric(),
  effect_size = numeric(),
  stringsAsFactors = FALSE
)

for (i in 1:nrow(df_count)) {

  target_gene <- df_count[i,] # each gene as target gene
  other_gene <- df_count[-i,] # excluded target gene as other gene

  df_matrix <- matrix(c(
      target_gene$loss_of_function, target_gene$neutral,  # target gene lof vs neutral
      sum(other_gene$loss_of_function) ,sum(other_gene$neutral)),  # other gene lof vs neutral
      nrow = 2
  )

  fisher_result <- fisher.test(df_matrix ,alternative = "greater") # single sided test

# collect results
  results <- rbind(results,data.frame(
    gene = target_gene$gene,            # gene name
    p_value = fisher_result$p.value,    # p-value
    effect_size = fisher_result$estimate # estimate of effect size
  ))
}

head(results)
```

```
##                gene   p_value effect_size
## odds ratio     A1CF 1.0000000           0
## odds ratio1   A2ML1 1.0000000           0
## odds ratio2  A4GALT 1.0000000           0
## odds ratio3   A4GNT 0.3239772         Inf
## odds ratio4    AACS 0.3239772         Inf
## odds ratio5 AADACL4 0.3239772         Inf
```

```r
# check p value dist
hist(results$p_value,breaks = seq(0,1,length= 20),
     main = "Histogram of p value")
```

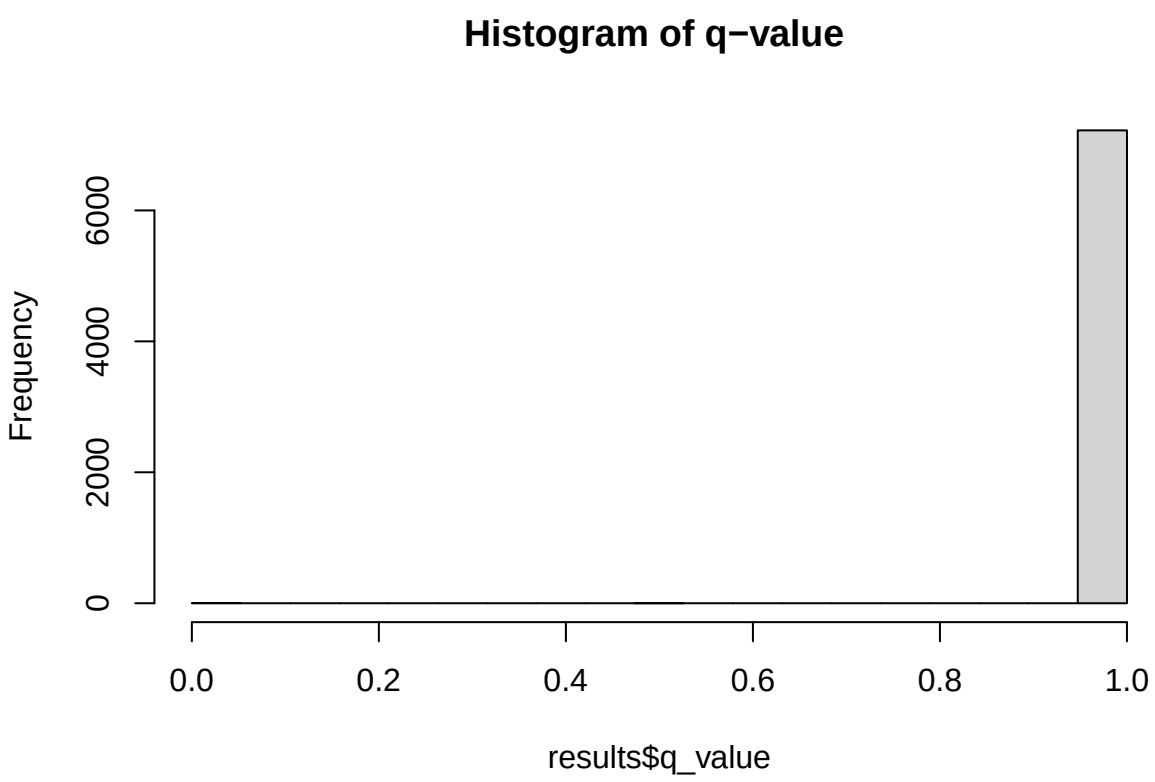


Identify candidate tumour suppressor genes using this statistical test, adjusting for multiple hypothesis testing. The output table should contain: gene symbol, an estimate of effect size,p value, q value

```r
# q value
results$q_value <- p.adjust(results$p_value, method = "BH") # fdr method
head(results)
```

```
##                gene   p_value effect_size q_value
## odds ratio     A1CF 1.0000000           0       1
## odds ratio1   A2ML1 1.0000000           0       1
## odds ratio2  A4GALT 1.0000000           0       1
## odds ratio3   A4GNT 0.3239772         Inf       1
## odds ratio4    AACS 0.3239772         Inf       1
## odds ratio5 AADACL4 0.3239772         Inf       1
```

```r
# plot the q value
hist(results$q_value,breaks = seq(0,1,length= 20),
     main = "Histogram of q-value ")
```




```
# select q value less than 0.05
candidate_TSG <- results[results$q_value < 0.05,]
# rank by q value
candidate_TSG[order(candidate_TSG$q_value), ]  # RB1 TP53 MLL2
```

```
##                gene      p_value effect_size      q_value
## odds ratio5228  RB1 1.832555e-35    81.67986 1.324388e-31
## odds ratio6464 TP53 6.957390e-19         Inf 2.514053e-15
## odds ratio3710 MLL2 4.727031e-06    11.16818 1.138742e-02
```

5. Perform a literature search and explain the function of each candidate gene in the context of cancer, as well as specifically in small cell lung cancer.

TP53: p53 is a major tumor suppressive transcription factor that responds to several cellular stresses, including DNA damage, oxidative stress, hypoxia, lack of nutrients, and hyperproliferative stimuli. The transcriptional activity of p53 upregulates the expression of numerous genes that participate in DNA repair mechanisms, cell cycle arrest, cell apoptosis, autophagy, and cellular senescence, all of which aim to maintain the stability of the cellular genome and suppress cancer development. Therefore, cells that harbor loss-of-function mutations in the gene TP53 are unable to repair DNA damage and acquire potentially oncogenic genetic alterations, cannot enter senescence and undergo apoptosis in response to oncogenic signals, and proliferate excessively and uncontrollably. The p53 gene is mutated in more than 50% of human cancers, and mutations in other genes that affect p53 function occur in many, if not all, tumors that retain a normal p53 gene. Loss of p53 pathway function gives cancer cells a survival advantage to bypass the resolution of oncogenic signals and DNA damage to continue abnormal proliferation.

RB1: Rb is another important transcriptional regulator with tumor suppressive functions that is involved in the regulation of the progression of the cell cycle. Given the central role of Rb in the negative regulation of the cell cycle, it becomes evident why loss-of-function mutations in the gene RB1 renders cells capable of transforming into cancer cells. While RB disruption is the sine qua non of retinoblastoma, RB loss occurs in many tumor types. 80% of Small cell lung cancer cause by RB loss.

MLL2: A study results revealed that frequent mutations in the lysine methyltransferase 2D gene (KMT2D) (also known as MLL2) is a key regulator of transcriptional enhancer function. KMT2D exhibited truncating nonsense/frameshift/splice site mutations in 8% of SCLC tumors and 17% of SCLC cell lines.KMT2D mutation in human SCLC cell lines was associated with reduced lysine methyltransferase 2D protein levels and reduced monomethylation of histone H3 lysine 4, a mark associated with transcriptional enhancers.

Conclusion: Small cell lung cancer (SCLC) is a highly aggressive human tumor with a more than 95% mortality rate.The TP53 and RB1 tumor suppressor genes, responsible for encoding the transcriptional regulators p53 and Rb, respectively, are almost universally inactivated in SCLC cells and are considered the genomic hallmarks of SCLC. Inactivating mutations in TP53 and RB1 have been shown to affect up to 90% and up to 65% of SCLC, respectively. Missense mutations in TP53 affected the functionally critical DNA binding domain, while RB1 was frequently altered by complex genomic translocations. KMT2D is also one of the major mutated genes in SCLC, and perturbation of transcriptional enhancer control as potentially contributing to SCLC.

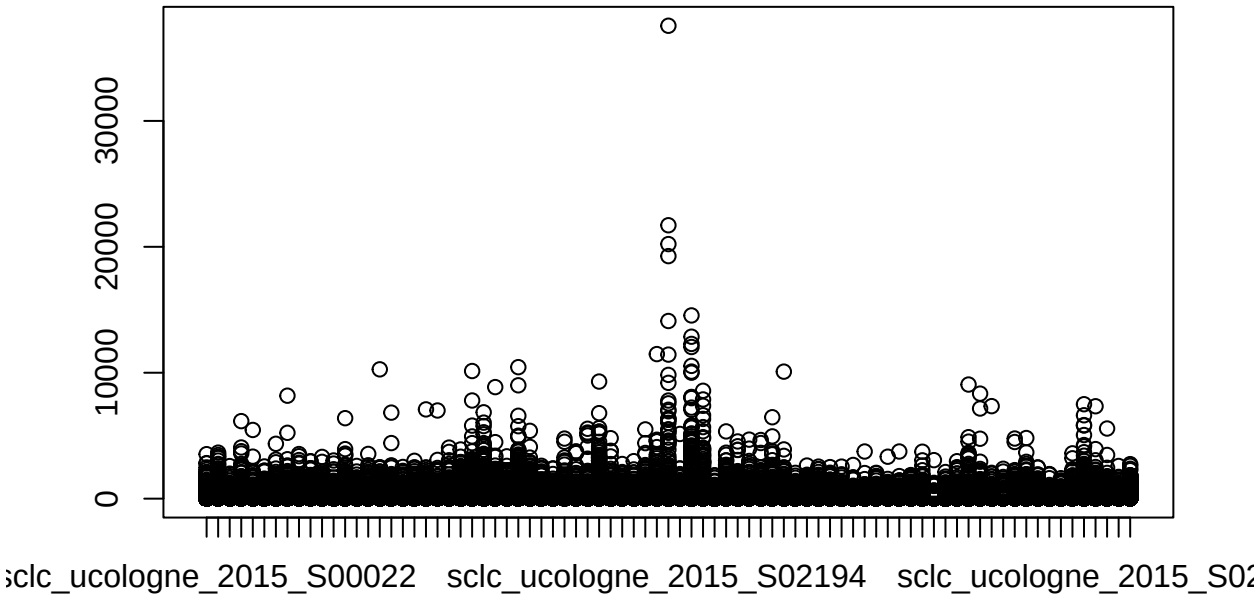
Key References:

- Papavassiliou KA, Sofianidi AA, Gogou VA, Anagnostopoulos N, Papavassiliou AG. P53 and Rb Aberrations in Small Cell Lung Cancer (SCLC): From Molecular Mechanisms to Therapeutic Modulation. Int J Mol Sci. 2024;25(5):2479.
- Zullo L, Dall'Olio FG, Rossi G, Dellepiane C, Barletta G, Bennicelli E, Ingaliso M, Tagliamento M, Genova C. Molecular and Genetic Advances in Small Cell Lung Cancer Landscape: From Homogeneity to Diversity. Int J Mol Sci. 2024;25:224.
- Peifer M, Fernández-Cuesta L, Sos ML, et al. Integrative genome analyses identify key somatic driver mutations of small cell lung cancer. Nat Genet.2012;44(10):1104-1110.
- Rudin CM, Durinck S, Stawiski EW, et al. Comprehensive genomic profiles of small cell lung cancer. Nature. 2015;524(7563):47-53.
- Sherr CJ, McCormick F. The RB and p53 pathways in cancer. Cancer Cell. 2002;2(2):103-112.
- Vousden KH, Lane DP. p53 in health and disease. Nat Rev Mol Cell Biol. 2007;8(4):275-283.
- Dyson NJ. RB1: a prototype tumor suppressor and an enigma. Genes Dev. 2016;30(13):1492-1502. doi:10.1101/gad.282145.116.
- Dick FA, Rubin SM. Molecular mechanisms underlying RB protein function. Nat Rev Mol Cell Biol. 2013;14(5):297-306.
- Knudsen KE, Witkiewicz AK. Therapeutic strategies for RB1-deficient cancers. Nat Rev Clin Oncol. 2017;14(3):157-171.
- Augert A, Zhang Q, Bates B, et al. Small Cell Lung Cancer Exhibits Frequent Inactivating Mutations in the Histone Methyltransferase KMT2D/MLL2: CALGB 151111 (Alliance). J Thorac Oncol. 2017;12(4):704-713.

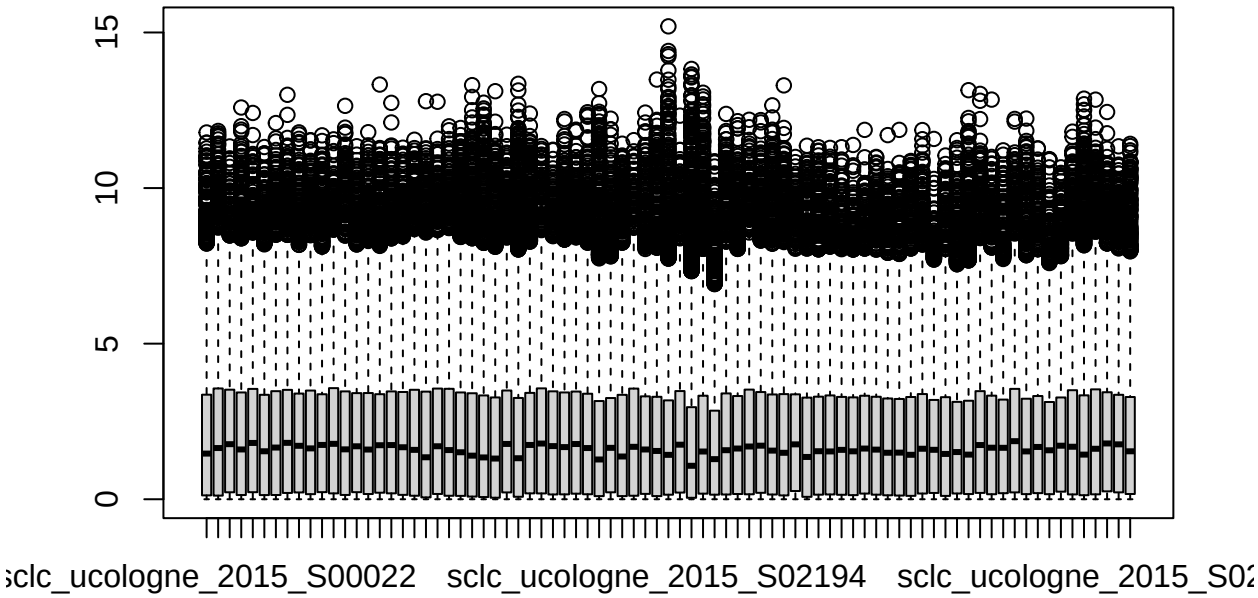
Transcriptomic data normalization

1.Perform an appropriate log transformation on the data.

```
# read rds data
trans <- readRDS('../Desktop/exercise/expr_sclc_ucologne_2015.rds')

## Check distributions
boxplot(trans)
```




```
# log2 counts
log_trans <- log2(trans + 1) # add 1 to create 'pseudo-count'

## Check distributions
boxplot(log_trans)
```



2. Implement a median polish algorithm from scartch.

```
# First iteration
row_effect <- log_trans - rowMedians(as.matrix(log_trans))
column_effect <- row_effect - colMedians(as.matrix(row_effect))

global_median <- median(as.matrix(column_effect))
residual <- column_effect - global_median

# Second iteration
row_eff <- residual - rowMedians(as.matrix(residual))
column_eff <- row_effect - colMedians(as.matrix(row_effect))
global_med <- median(as.matrix(column_eff))
residual_s <- column_eff - global_med # final residual
```

```
# row and column effect median are 0s,stop the iterating.
abs(median(unlist(row_eff))) # 0
```

```
## [1] 0
```

```
abs(median(unlist(column_eff))) # 0
```

```
## [1] 0
```

3. Compare the residuals of your algorithm and stats::medpolish.

```
# Using stats::medpolish
df.med <- medpolish(log_trans)
```

```
## 1: 650124.2
## Final: 649983.1
```

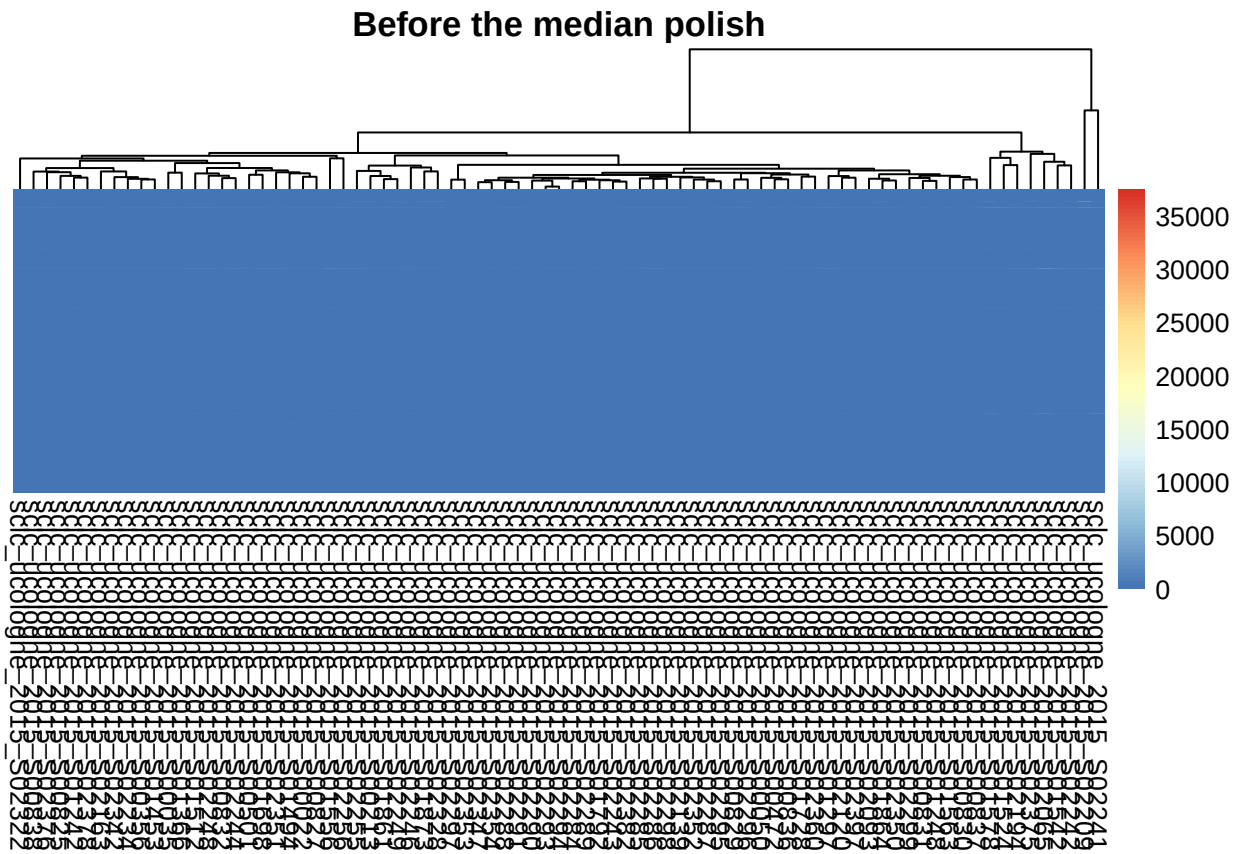
```
# stats::medpolish residual
stats_residual <- df.med$residuals
```

```
# compare two residuals by subtract each other
error_matrix <- abs(round(stats_residual,2) - round(residual_s,2))

# maximum error
max(error_matrix)  # the maximum error between two residuals is 0.28.
```

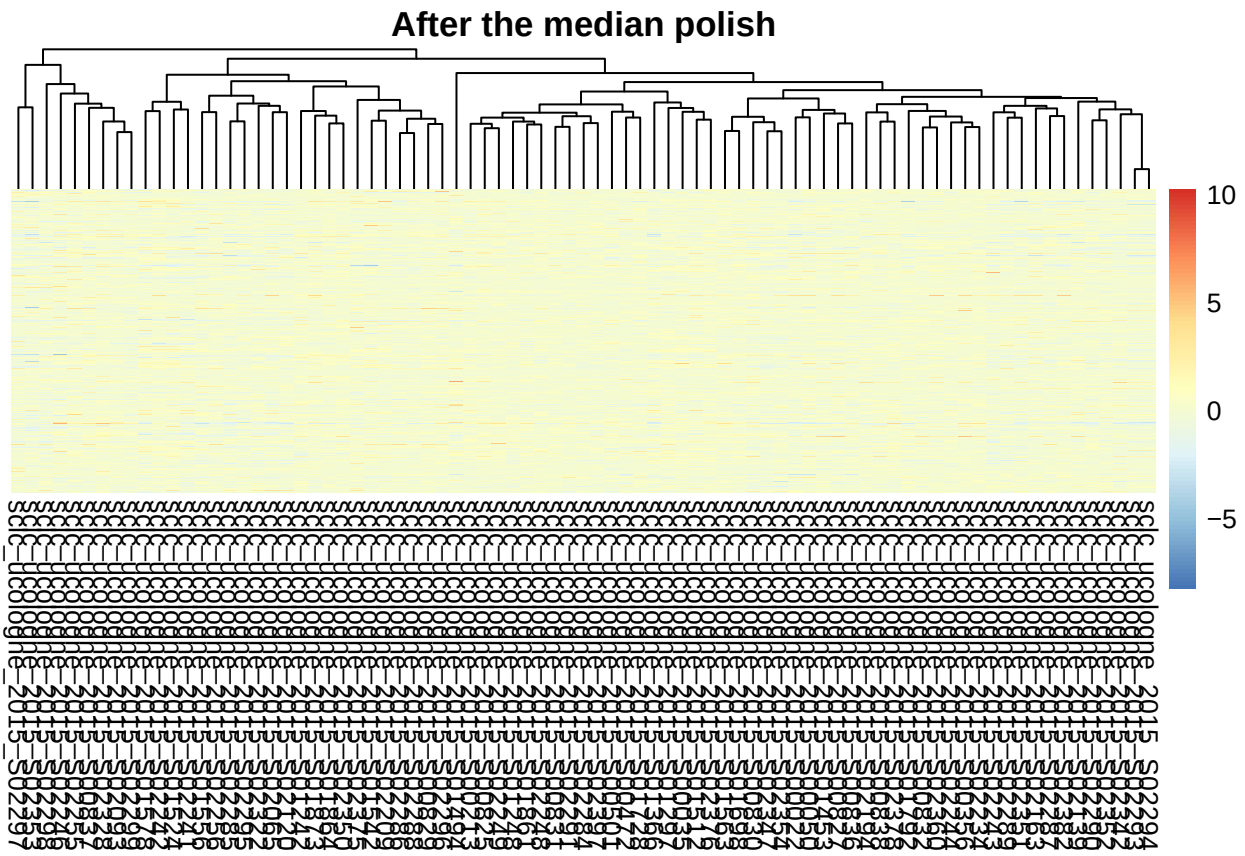
```
## [1] 0.28
```

4.Plot heatmaps of the results before and after median polish.

```
# Plot heatmap before median polish
conflicts_prefer(dplyr::desc)

# conver to matrix
matrix_med <- as.matrix(trans)  # use orignal data

#plot the heat map
pheatmap(matrix_med, main = "Before the median polish",
         show_rownames = FALSE,
         show_colnames = TRUE,
         cluster_rows = FALSE,
         cluster_cols = TRUE)
```




```
# Plot heatmaps after median polish
pheatmap(stats_residual,main = "After the median polish",
         show_rownames = FALSE,
         show_colnames = TRUE,
         cluster_rows = FALSE,
         cluster_cols = TRUE)
```



5.Output the median polished residual matrix as the normalized transcriptomic data.

```
# write the data as csv file
write.csv(residuals, "normalized_transcriptomic_data.csv")
```

Differential gene expression analysis

1. Define two groups of tumours as early stage (stages I-II) vs. advanced stage tumours (stages III-IV), while excluding samples missing stage information.

```
## load the data
  pheno_data <- read_tsv('exercise/pheno_sclc_ucologne_2015.tsv')
head(pheno_data) # 120  17
```

```
## # A tibble: 6 x 17
##   patient_id         age sex   ethnicity n_stage m_stage uicc_tumor_stage smoker
##   <chr>            <dbl> <chr> <chr>     <chr>   <chr>   <chr>            <chr>
## 1 sclc_ucologne_2~    47 Male  <NA>      2       0       IIIa             Curre~
## 2 sclc_ucologne_2~    65 Fema~ <NA>      1       1       IV               Former
## 3 sclc_ucologne_2~    47 Male  <NA>      0       0       Ia               Curre~
## 4 sclc_ucologne_2~    65 Male  <NA>      2       0       IIIa             <NA>
## 5 sclc_ucologne_2~    61 Male  <NA>      2       0       IIIa             <NA>
## 6 sclc_ucologne_2~    54 Fema~ <NA>      2       0       IIIa             Curre~
```

```
## # i 9 more variables: smoking_history <chr>, previous_treatment <chr>,
## #   method_of_sample_procurement <chr>, neoadj_chemo <chr>, chemotherapy <chr>,
## #   radiation_therapy <chr>, os_status <chr>, pfs_months <dbl>, os_months <dbl>
```

```
# check the missing data
pheno_data[is.na(pheno_data$uicc_tumor_stage),]
```

```
## # A tibble: 2 x 17
##   patient_id         age sex   ethnicity n_stage m_stage uicc_tumor_stage smoker
##   <chr>            <dbl> <chr> <chr>     <chr>   <chr>   <chr>            <chr>
## 1 sclc_ucologne_2~    71 Fema~ <NA>      <NA>    <NA>    <NA>             Former
## 2 sclc_ucologne_2~    53 Male  <NA>      <NA>    <NA>    <NA>             <NA>
## # i 9 more variables: smoking_history <chr>, previous_treatment <chr>,
## #   method_of_sample_procurement <chr>, neoadj_chemo <chr>, chemotherapy <chr>,
## #   radiation_therapy <chr>, os_status <chr>, pfs_months <dbl>, os_months <dbl>
```

```
## drop the missing data
pheno <- pheno_data[!is.na(pheno_data$uicc_tumor_stage),]
dim(pheno) # 118  17
```

```
## [1] 118  17
```

```
# pre-clean the data
unique(pheno$uicc_tumor_stage) # IB is misspelling
```

```
##  [1] "IIIa" "IV"   "Ia"   "IIb"  "IIa"  "IIIb" "I"    "III"  "Ib"   "II"
## [11] "IB"
```

```
# correct the stage
pheno$uicc_tumor_stage[pheno$uicc_tumor_stage == "IB"] <- "Ib"
unique(pheno$uicc_tumor_stage) # check the result
```

```
##  [1] "IIIa" "IV"   "Ia"   "IIb"  "IIa"  "IIIb" "I"    "III"  "Ib"   "II"
```

```
# define the early stage and advance stage
Early_stage <- c("I","Ia","Ib","II","IIa","IIb")
advance_Stage <- c("III","IIIa","IIIb","IV")
```

```
# create a new column to identify the tumor stage
pheno <- pheno %>%
  mutate(stage = case_when(
    uicc_tumor_stage %in% Early_stage ~ "Early stage",
    uicc_tumor_stage %in% advance_Stage ~ "Advanced stage"
  ))
head(pheno,3)
```

```
## # A tibble: 3 x 18
##   patient_id         age sex   ethnicity n_stage m_stage uicc_tumor_stage smoker
##   <chr>            <dbl> <chr> <chr>     <chr>   <chr>   <chr>            <chr>
## 1 sclc_ucologne_2~    47 Male  <NA>      2       0       IIIa             Curre~
## 2 sclc_ucologne_2~    65 Fema~ <NA>      1       1       IV               Former
## 3 sclc_ucologne_2~    47 Male  <NA>      0       0       Ia               Curre~
## # i 10 more variables: smoking_history <chr>, previous_treatment <chr>,
## #   method_of_sample_procurement <chr>, neoadj_chemo <chr>, chemotherapy <chr>,
## #   radiation_therapy <chr>, os_status <chr>, pfs_months <dbl>,
## #   os_months <dbl>, stage <chr>
```

2.Identify genes that differentially expressed in early vs. advanced stage tumours using an appropriate R package.

```
# read count data
countData <- readRDS('exercise/expr_sclc_ucologne_2015.rds')
countData <- as.matrix(countData,row.names = 1)
```

```
## read Count matrix
colData <- pheno %>%
  select(patient_id,stage) %>%
  column_to_rownames(var="patient_id") # set the column name
head(colData)
```

```
##                                   stage
## sclc_ucologne_2015_S00022 Advanced stage
## sclc_ucologne_2015_S00035 Advanced stage
## sclc_ucologne_2015_S00050    Early stage
## sclc_ucologne_2015_S00213 Advanced stage
## sclc_ucologne_2015_S00339 Advanced stage
## sclc_ucologne_2015_S00356 Advanced stage
```

```
# select countData patient id and colData patient id to find out the common id
count_samples <- colnames(countData)
colData_samples <- rownames(colData)
```

```
# match the common sample id
common_id <- intersect(count_samples, colData_samples)
length(common_id) ## 79 common id
```

```
## [1] 79
```

```
# subset common data in count data
countData <- countData[,common_id]
```

```
# subset common data in count matrix
colData <- colData %>%
  rownames_to_column(var = "patient_id") # convert row name to regular column

# select common data
colData <- colData[colData$patient_id %in% common_id, ]

# set col names again for deseq2 required
colData <- colData %>%
  remove_rownames %>%
  column_to_rownames(var="patient_id")
```

```
## check col data and count data in same order
all(rownames(colData) %in% colnames(countData))
```

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

```
all(colnames(countData) == rownames(colData))
```

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

limma-trend approach

```
# log2 transform the values and add 1 to create pseudo count
log_data <- log2(countData + 1)

# create design matrix
design <- model.matrix(~ stage, data = colData) # early vs.advanced stage

# fit the model
fit <- lmFit(log_data, design)

# Apply empirical Bayes
fit <- eBayes(fit, trend=TRUE,robust=TRUE) # robust reduce the zero variances
```

Integrative analysis

1.For each gene involved in a structural variant (SV), determine the expression level of the gene in the sample that harbours the SV.

```
# load the data
sv<-read_tsv("exercise/sv_sclc_ucologne_2015.tsv")
head(sv)
```

```
## # A tibble: 6 x 12
##   sample_id       sv_status site1_hugo_symbol site1_chromosome site2_hugo_symbol
##   <chr>           <chr>     <chr>                        <dbl> <chr>
## 1 sclc_ucologne_~ SOMATIC   KDM2A                           11 GLB1L2
## 2 sclc_ucologne_~ SOMATIC   ELMOD1                           3 HPS3
## 3 sclc_ucologne_~ SOMATIC   MTCH2                            11 ODZ4
## 4 sclc_ucologne_~ SOMATIC   TM7SF2                           11 PRRG4
## 5 sclc_ucologne_~ SOMATIC   PPME1                            11 FYCO1
## 6 sclc_ucologne_~ SOMATIC   TAF1D                            3 PIK3CA
## # i 7 more variables: site2_chromosome <dbl>, site2_effect_on_frame <chr>,
## #   tumor_paired_end_read_count <dbl>, tumor_split_read_count <dbl>,
## #   event_info <chr>, ncbi_build <chr>, center <chr>
```

```
# load the data
sv<-read_tsv("exercise/sv_sclc_ucologne_2015.tsv")

# select all gene according sv sample
sv_sample <- trans[, c("sclc_ucologne_2015_S02297", "sclc_ucologne_2015_S02353")]

# find sv gene level
sv_sample <- as.data.frame(sv_sample) %>%
  rownames_to_column(var = "gene_id") # set the row name
```

```
# sample sclc_ucologne_2015_S02297
sample_s <- sv[sv$sample_id == 'sclc_ucologne_2015_S02297',] # select sample data

# get all the site 1 and site 2 genes
site_gene <- sample_s$site1_hugo_symbol # site 1 genes
site_genes <- sample_s$site2_hugo_symbol # site 2 genes
all_gene_samples <- unique(c(site_gene, site_genes)) # all genes
```

```r
# rank the genes in sample
sv_sample$rank <- rank(sv_sample$sclc_ucologne_2015_S02297)
sv_sample$percentile  <-
  rank(sv_sample$sclc_ucologne_2015_S02297)/length(sv_sample$sclc_ucologne_2015_S02297)*100

# get sv rank data
sv_rank <- sv_sample %>%
  filter(gene_id %in% all_gene_samples) %>%
  select(gene_id, sclc_ucologne_2015_S02297,rank, percentile)

# set descending order of the data
order_rank <- sv_rank[order(sv_rank$rank,decreasing = T),]

# get rank sv genes in the first sample
head(order_rank)
```

```
##    gene_id sclc_ucologne_2015_S02297  rank percentile
## 25  PODXL2                   55.3133 18350   97.57005
## 31   SETD5                   44.1639 18179   96.66082
## 35   TAF1D                   33.9192 17864   94.98591
## 4    CADM2                   30.2330 17722   94.23087
## 26   PPME1                   25.2323 17429   92.67294
## 9     EI24                   23.2915 17274   91.84878
```

```r
# sample sclc_ucologne_2015_S02353
sample_S <- sv[sv$sample_id == 'sclc_ucologne_2015_S02353',] # select sample data

# get all the site 1 and site 2 genes
site_gene_S <- sample_S$site1_hugo_symbol # site 1 genes
site_genes_S <- sample_S$site2_hugo_symbol # site 2 genes
all_gene_samples_S <- unique(c(site_gene_S, site_genes_S)) # all genes


# rank the genes in sample
sv_sample$rank_S <- rank(sv_sample$sclc_ucologne_2015_S02353)
sv_sample$percentile_S  <-
  rank(sv_sample$sclc_ucologne_2015_S02353)/length(sv_sample$sclc_ucologne_2015_S02353)*100

# get sv rank data
sv_rank_S <- sv_sample %>%
  filter(gene_id %in% all_gene_samples_S) %>%
  select(gene_id, sclc_ucologne_2015_S02353,rank_S, percentile_S)

# set descending order of the data
ordered_rankS2 <- sv_rank_S[order(sv_rank_S$rank_S,decreasing = T),]

# get rank sv genes in the second sample
head(ordered_rankS2)
```

```
##    gene_id sclc_ucologne_2015_S02353 rank_S percentile_S
## 5     ATP5L                   31.1334  17840     94.85830
## 12     DVL3                   20.6609  17135     91.10969
## 27   NUP210                   19.8358  17050     90.65773
## 39    SAMD5                   13.6785  16075     85.47349
## 29    PACS1                   13.4166  16020     85.18105
```

```
## 8    CCKBR                   12.7256  15887     84.47387
```

```
# load the data
sv<-read_tsv("exercise/sv_sclc_ucologne_2015.tsv")

# select all gene according sv sample
sv_sample <- trans[, c("sclc_ucologne_2015_S02297", "sclc_ucologne_2015_S02353")]
```

```
# connect to the human gene database
ensembl = useEnsembl(biomart="ensembl", dataset="hsapiens_gene_ensembl")

# get gene length according to the gene symbol, "hgnc_symbol"
genelength =  getBM(attributes=c('hgnc_symbol', 'ensembl_transcript_id',
                                 'transcript_length'),
                    filters ='hgnc_symbol',   # human gene database
                    values = rownames(sv_sample),
                    mart = ensembl,
                    useCache = FALSE)

# get the canonical version gene length from multiple versions
gene_canonical_transcript =  getBM(attributes=c('hgnc_symbol',
                                                'ensembl_transcript_id',
                                                'transcript_is_canonical'),
                                   filters =  'hgnc_symbol',
                                   values = rownames(sv_sample),
                                   mart = ensembl,
                                   useCache = FALSE)

# get rid of NA value
gene_canonical_transcript_subset =
  gene_canonical_transcript[!is.na(gene_canonical_transcript$transcript_is_canonical),]

# combine unique canonical gene length with gene name
genelength = merge(gene_canonical_transcript_subset,
                   genelength,
                   by = c('hgnc_symbol','ensembl_transcript_id'))

head(genelength)
```

```
##   hgnc_symbol ensembl_transcript_id transcript_is_canonical transcript_length
## 1        A1BG       ENST00000263100                       1              3382
## 2        A1CF       ENST00000373997                       1              9221
## 3         A2M       ENST00000318602                       1              4610
## 4       A2ML1       ENST00000299698                       1              5127
## 5      A4GALT       ENST00000642412                       1              2092
## 6       A4GNT       ENST00000236709                       1              1779
```

2. Identify SVs that satisfy the following critiera:

- The involved pair of genes both have elevated expression levels in samples with the SV compared to samples without the SV.

- The second gene in the pair is in frame.

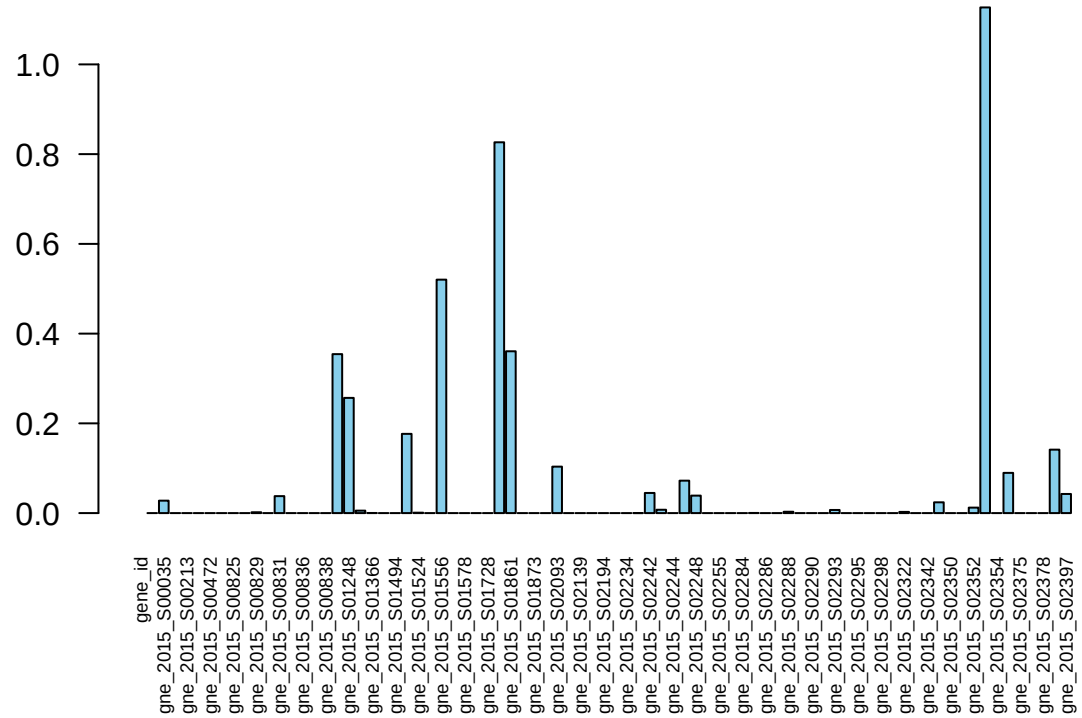
```
# read rds data
expr_data <- readRDS('exercise/expr_sclc_ucologne_2015.rds')

expr_data <- as.data.frame(expr_data) %>%
  rownames_to_column(var = "gene_id") # set row name
```

```
# the genes has high expression level (top 10%) in the first sv sample
high_rank1 <- order_rank[order_rank$percentile > 90,]

# the genes in the samples without the sv sample
expr_data1 = subset(expr_data, select = -c(sclc_ucologne_2015_S02297) )

# select high expression genes (Top 10%) in the sample
high_exp1 <- expr_data1[expr_data1$gene_id %in%
      c("PODXL2","SETD5","TAF1D","CADM2","PPME1","EI24" ,"ST14" ,"PSMC3","PRRG4"),]


high_exp1[1,1:5]
```

```
##      gene_id sclc_ucologne_2015_S00022 sclc_ucologne_2015_S00035
## 2518   CADM2                     1e-06                  0.027683
##      sclc_ucologne_2015_S00050 sclc_ucologne_2015_S00213
## 2518                         0                         0
```

show the first gene's expression level in samples

```
# subset the first row
gene_data <- as.numeric(high_exp1[1, ])

# set the sample name
sample_names <- colnames(high_exp1)

# plot bar chart
barplot(
  gene_data,
  names.arg = sample_names,  # set names
  col = "skyblue",      # change the color
  main = paste("Gene", high_exp1[1,1], "Expression level in samples"),  # set title
  las = 2,         #  display all names
  cex.names = 0.6   # set label size
)
```

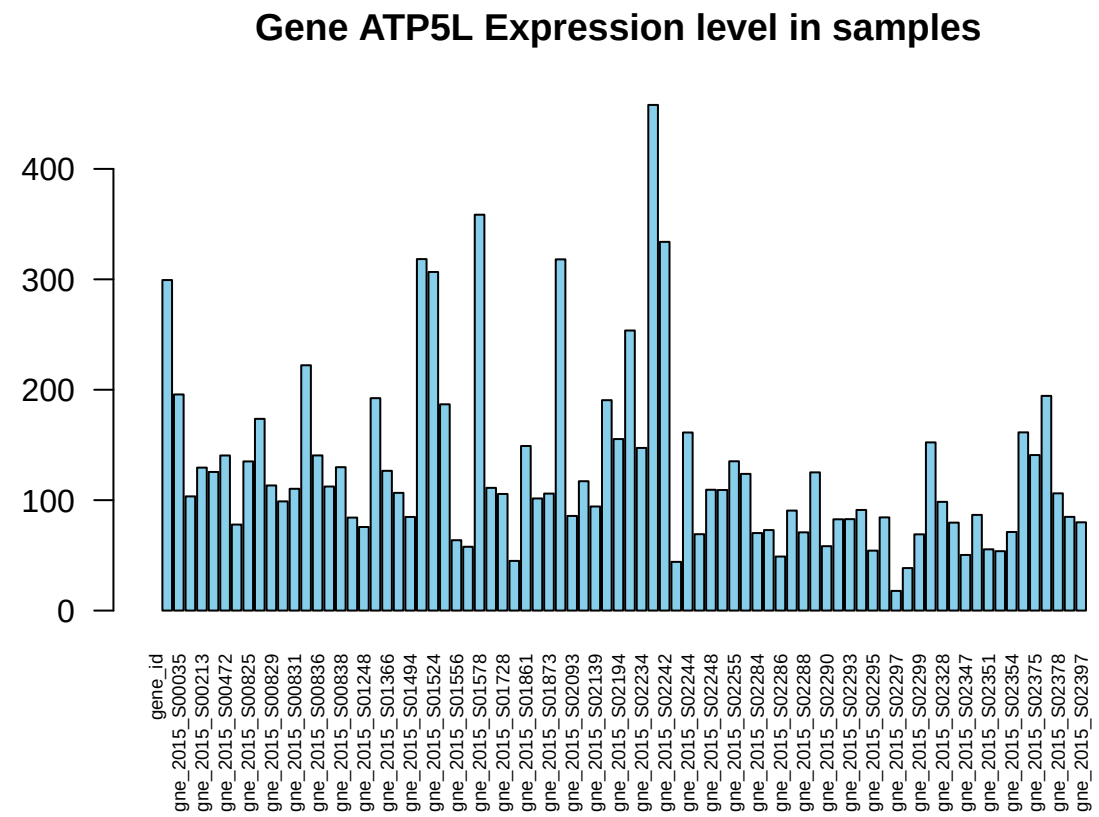



```
# the genes has high expression level in the second sv sample
high_rank2 <- ordered_rankS2[ordered_rankS2$percentile > 90,]

# the genes in the samples without the sv
expr_data2 = subset(expr_data, select = -c(sclc_ucologne_2015_S02353) )

# select high expression genes (Top 10%) in the sample 2
high_exp2 <- expr_data2[expr_data2$gene_id %in%
      c("ATP5L","DVL3" ,"NUP210"),]

high_exp2[1,1:10]
```

```
##      gene_id sclc_ucologne_2015_S00022 sclc_ucologne_2015_S00035
## 1188   ATP5L                   299.324                   195.736
##      sclc_ucologne_2015_S00050 sclc_ucologne_2015_S00213
## 1188                   103.407                   129.462
##      sclc_ucologne_2015_S00356 sclc_ucologne_2015_S00472
## 1188                   125.535                   140.455
##      sclc_ucologne_2015_S00501 sclc_ucologne_2015_S00825
## 1188                   77.8735                   135.101
##      sclc_ucologne_2015_S00827
## 1188                   173.643
```

show the first gene's expression level in samples

```
# subset the first row
gene_data2 <- as.numeric(high_exp2[1, ])

# set the sample name
sample_names2 <- colnames(high_exp2)

# plot bar chart
barplot(
  gene_data2,
  names.arg = sample_names2,  # set bar names
  col = "skyblue",            # change the color
  main = paste("Gene", high_exp2[1,1], "Expression level in samples"),  # set title
  las = 2,                    # display all names
  cex.names = 0.6             # set label size
)
```



2. The second gene in the pair is in frame.

```
# subset in frame data
pair_in_frame <- sv[sv$site2_effect_on_frame == 'in-frame',]

head(pair_in_frame)
```

```
## # A tibble: 6 x 12
```

```
##   sample_id       sv_status site1_hugo_symbol site1_chromosome site2_hugo_symbol
##   <chr>           <chr>     <chr>                         <dbl> <chr>
## 1 sclc_ucologne_~ SOMATIC   KDM2A                            11 GLB1L2
## 2 sclc_ucologne_~ SOMATIC   MTCH2                            11 ODZ4
## 3 sclc_ucologne_~ SOMATIC   PPME1                            11 FYCO1
## 4 sclc_ucologne_~ SOMATIC   TAF1D                             3 PIK3CA
## 5 sclc_ucologne_~ SOMATIC   MYO7A                            11 PSMC3
## 6 sclc_ucologne_~ SOMATIC   TXNRD3                            3 PODXL2
## # i 7 more variables: site2_chromosome <dbl>, site2_effect_on_frame <chr>,
## #   tumor_paired_end_read_count <dbl>, tumor_split_read_count <dbl>,
## #   event_info <chr>, ncbi_build <chr>, center <chr>
```

```
#head(sv)
colnames(sv)
```

```
##  [1] "sample_id"                 "sv_status"
##  [3] "site1_hugo_symbol"         "site1_chromosome"
##  [5] "site2_hugo_symbol"         "site2_chromosome"
##  [7] "site2_effect_on_frame"     "tumor_paired_end_read_count"
##  [9] "tumor_split_read_count"    "event_info"
## [11] "ncbi_build"                "center"
```