Preliminary RNAseq analysis

Jieun Jeong

2025-02-23

This example analyzes RNAseq data from GEO data series GSE263611. In this project, four cell lines of cardiomyocytes derived from patients' blood are cultured in two ways: control and treated with IFNg. Thus we observe expression variability stemming from initial genetic and epigenetic differences and from INFg treatment.

Data downloads

We start by collecting the data, and since we may analyze them more than once, testing different methods, we save the resulting tables. First, we make the table of sample metadata

```r
library(tinytex)
library(stringr)
library(tidyverse)
# we set the working directory in a manner that works in R markup
# pa() converts a local file name/path to "absolute path"
pa <- function(x) file.path('/Users/jieun/Work/Git_test/RNAseq_limma',x)
if (!file.exists(pa('GSE263611_key.RDS'))) { # avoid unnecessary downloads
  # load the meta data file, series matrix, to Line
  GEOpref = file.path('https://ftp.ncbi.nlm.nih.gov/geo/series',
        'GSE263nnn/GSE263611/matrix')
  Url = file.path(GEOpref,'GSE263611_series_matrix.txt.gz')
  temp_file <- pa('matrix.gz')
  download.file(Url, destfile = temp_file, mode = "wb")
  con <- gzfile(temp_file, "rt")
  Line <- readLines(con)
  close(con)
  file.remove(temp_file)
  Line <- gsub('"','',Line) # remove quote characters
  # after checking which lines have data we need, we make the table
  Accession <- character()
  Sample <- character()
  Treatment <- character()
  Assay <- character()
  CellLine <- character()
  E1 <- str_split_1(Line[30],'\t')
  E2 <- str_split_1(Line[39],'\t')
  E3 <- str_split_1(Line[47],'\t')
  for (i in 2:17) {
    Accession[i-1] <- E1[i]
    Sample[i-1] <- paste0("S",substr(E1[i],9,11)) # short sample names
    t <- sub('treatment: ','',E2[i]) # short treatment descriptions
    Treatment[i-1] <- sub('Untreated ','',t)
    B <- str_split_1(E3[i], ' ') # for extracting Assay and CellLine
    Assay[i-1] <- B[1]
    CellLine[i-1] <- B[3]
  }
  DF <- data.frame(Sample,Accession,Treatment,Assay,CellLine)
  column_to_rownames(DF, "Sample")
  saveRDS(DF,pa('GSE263611_key.RDS'))
}
```

Next, we collect count data. Like with metadata, GEO series may have different organization, so this approach may need to be adapted.

```
if (!file.exists(pa('GSE263611_counts.RDS'))) {
  library(GEOquery)
  library(readr)
  library(dplyr)
  Gse <- getGEO("GSE263611", GSEMatrix = FALSE)
  Sample <- GSMList(Gse)
  Url <- sapply(Sample, function(x) Meta(x)$supplementary_file)
  Url <- Url[grep("genes",Url)] # change if you want ATAC-seq
  n_samples <- length(Url)
  samples <- sapply(Url, function(x) sub("^.*GSM","GSM",x))
  coltypes = c(rep("c",2),rep("n",5))
  for (i in 1:n_samples) {
    temp_file = pa('sample.gz')
    download.file(Url[[i]], destfile = temp_file, mode = "wb")
    df <- read_tsv(temp_file, col_types = coltypes, progress = FALSE)
    file.remove(temp_file)
    df <- df %>% select(gene_id,expected_count) %>%
      mutate(gene_id = substr(gene_id,1,15)) %>% # removing suffixes
      distinct(gene_id,.keep_all = TRUE)
    if (i == 1) {
      Ensembl <- df$gene_id
      counts <- data.frame(df$expected_count)
    } else
      counts[[i]] <- df$expected_count
  }
# we will change identifiers to Symbol, so it is convenient to have format
# that makes it easy to change identifiers and then make named numeric matrix
  counts[[n_samples+1]] <- Ensembl
  colnames(counts) <- c(paste0("S",substr(samples,9,10)),"Ensembl")
  counts <- counts[apply(counts[,1:8], 1, sd) != 0] # remove genes with 0 std
  saveRDS(counts,pa('GSE263611_counts.RDS'))
}
```

There are many ways to change identifiers, I chose to use HGNC table because it additionally gives information about gene type, and this approach allows to modify it, e.g. `msigdb` uses Ensembl identifiers for some genes, so we can adapt to it by changing the table.

```
if (!(file.exists("HUGO.RDS"))) { # avoid unnecessary downloads
  library(readr)
  TT <- readLines(file.path('https://storage.googleapis.com',
                  'public-download-files/hgnc/tsv/tsv/hgnc_complete_set.txt'))
  Symbol <- character() # approved symbols
  Ensembl <- character() # Ensembl gene idT
  Type <- character() # gene type
  n = 0
  for (L in TT[2:length(TT)]) {
    n = n+1
    E <- strsplit(L,'\t')[[1]]
    Symbol[n] <- E[2]
    Type[n] <- E[5]
    Ensembl[n] <- E[20]
  }
  H <- data.frame(Symbol,Ensembl,Type)
  saveRDS(H,"HUGO.RDS")
}
```

Data exploration

Modify input tables

```
# making input tables
hugo <- readRDS(pa('HUGO.RDS')) # no modification needed
samples <- readRDS(pa('GSE263611_key.RDS')) %>%
  filter(Assay == "RNA-seq") %>%
  select(-Assay, -Accession)
print(samples)
```

```
##     Treatment CellLine
## S26   Control  UDID006
## S27      IFNg  UDID006
## S28   Control  UDID076
## S29      IFNg  UDID076
## S30   Control  UDID088
## S31      IFNg  UDID088
## S32   Control  UDID148
## S33      IFNg  UDID148
```

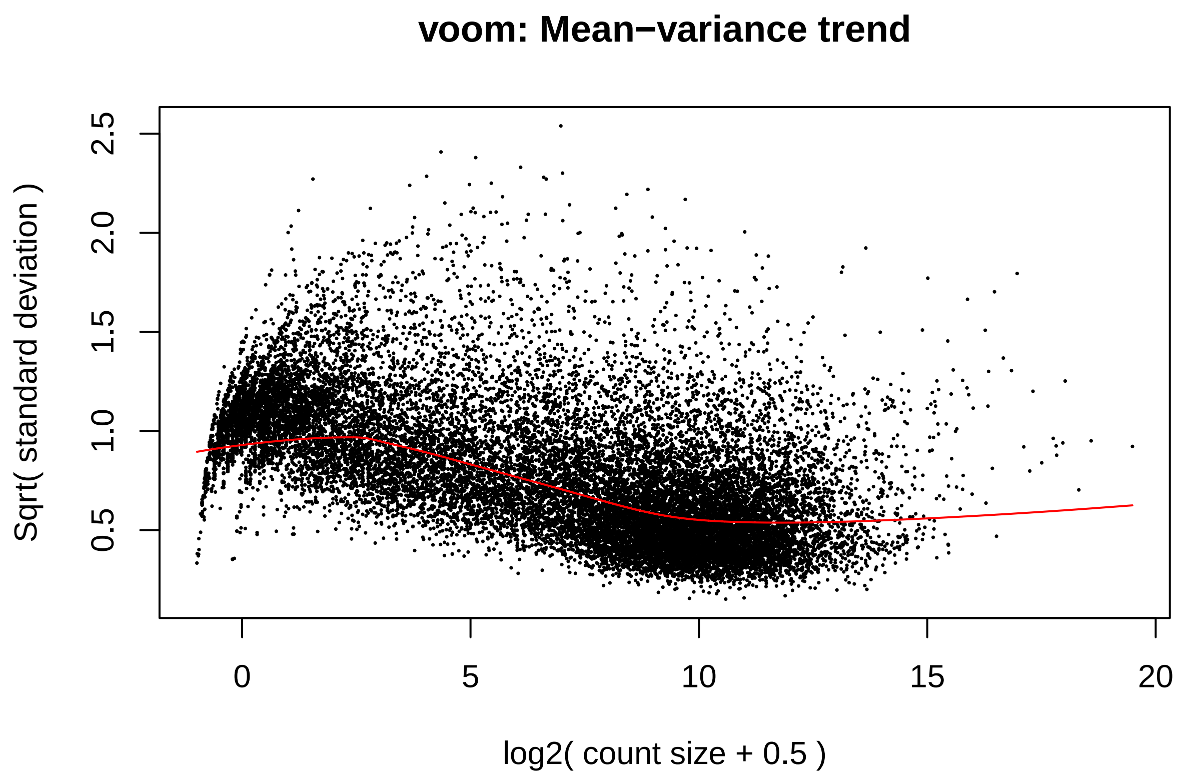
```
types_of_interest <- c("gene with protein product", "pseudogene",
                       "RNA, long non-coding")
counts <- readRDS(pa('GSE263611_counts.RDS')) %>%
  left_join(hugo, by="Ensembl") %>%
  filter(Type %in% types_of_interest) %>%
  select(-Type) %>%
  filter(Symbol != "") %>%
  distinct(Symbol, .keep_all = T) %>% # unnecessary in this case
  select(-Ensembl)
counts <- as.matrix(column_to_rownames(counts, "Symbol"))
```

Normalize count data and check the distribution of values. We will compare resuls for two designs

```r
library(edgeR)
### Definitions for limma
treatment <- as.factor(samples$Treatment)
cell_line <- as.factor(samples$CellLine)
design1 <- model.matrix(~ 0 + treatment) # minimal design
design2 <- model.matrix(~ 0 + treatment + cell_line) # a more complex design
# common normalizations
dge <- DGEList(counts = counts) # TMM normalization, reformmating
dge <- calcNormFactors(dge)
# separate fitting
limma_fit <- function(design) {
  contrast <- makeContrasts(treatmentIFNg - treatmentControl, levels = design)
  v <- voom(dge, design, plot = T) # log normalization and adjustment
  fit <- lmFit(v, design)
  fit <- contrasts.fit(fit, contrast)
  fit <- eBayes(fit)
  tT <- topTable(fit, adjust.method = "BH", coef = 1, number = Inf)
  return(list(E = v$E, preRank = tT, fit = fit))
}
```

Apply the limma fit to the 1st design:

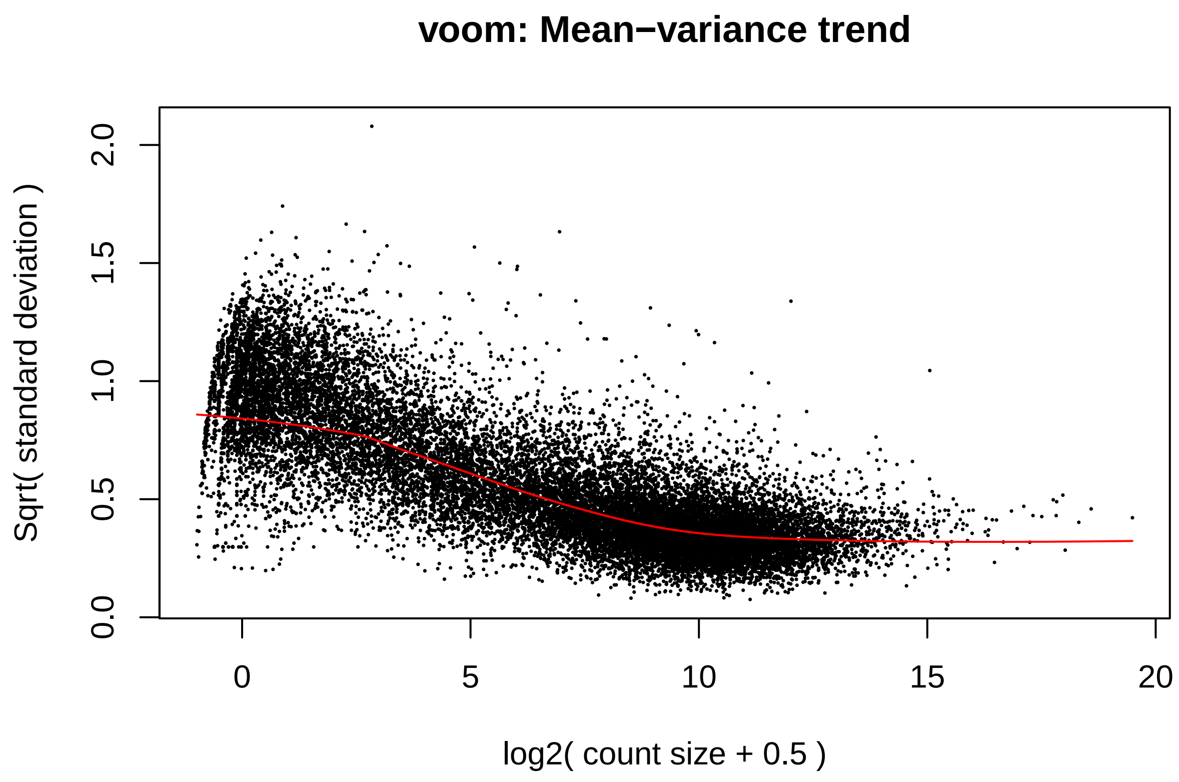
```r
results1 <- limma_fit(design1)
```



```r
saveRDS(results1$preRank,"Limma_1.RDS")
```

Apply the limma fit to the 2nd design:

```r
results2 <- limma_fit(design2)
```

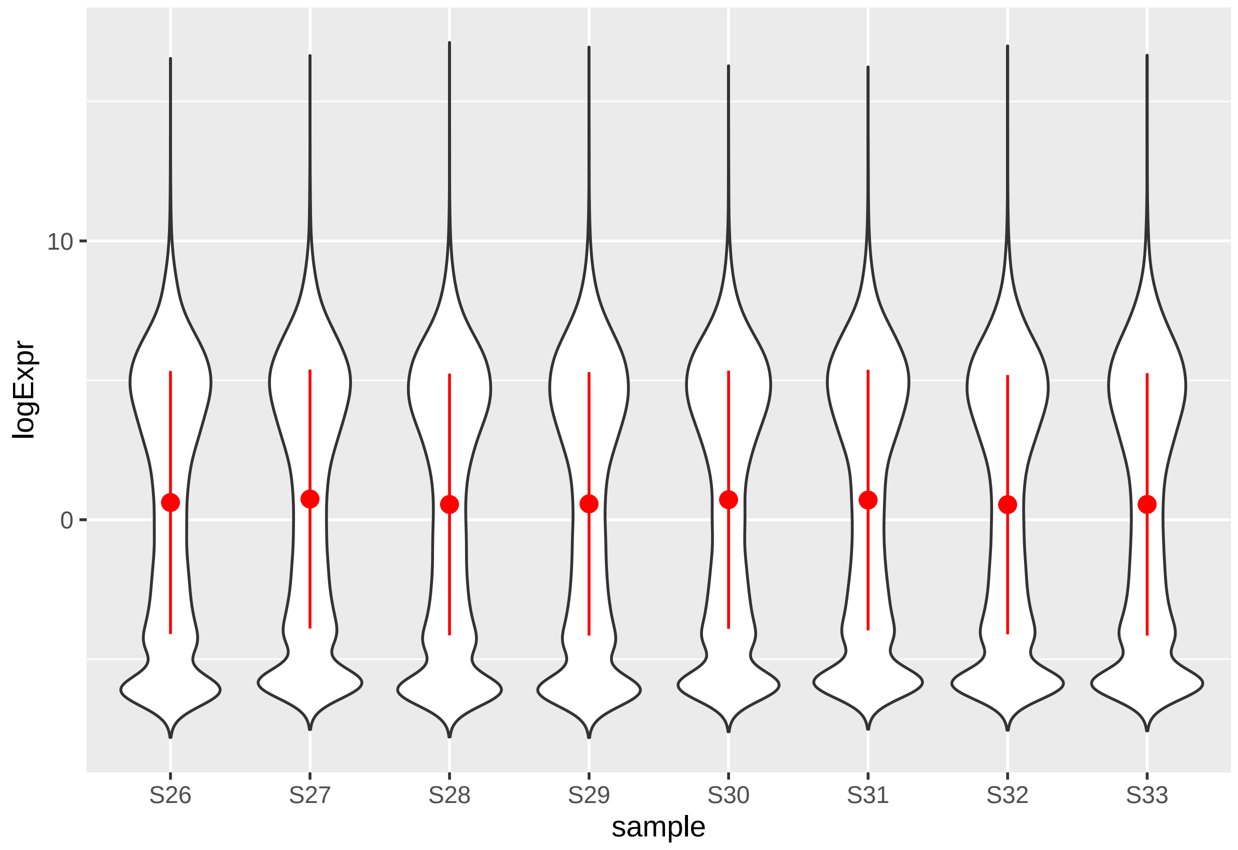


```r
saveRDS(results2$preRank,"Limma_2.RDS")
```

Observe the difference of two Mean-variance trends: the number of genes with variance above 1.5 decreased and the trend is not rising in the left half of the diagram. The reason is that some variance is explained/removed by the the variability of cell lines. We can also compare other findings when we apply results1 and results2.

Check the normalized distribution of gene expression

```r
# checking the normalized distribution of gene expression
A <- as.data.frame(results1$E)
A <- rownames_to_column(A,var='gene')
AL <- pivot_longer(A,-gene,names_to = "sample", values_to = "logExpr")
AL$sample <- as.factor(AL$sample)
p <- ggplot(AL,aes(x=sample,y=logExpr)) + geom_violin(trim=F,width=0.8)
p + stat_summary(fun.data=mean_sdl, geom="pointrange", color="red",fun.args=list(mult=1))
```

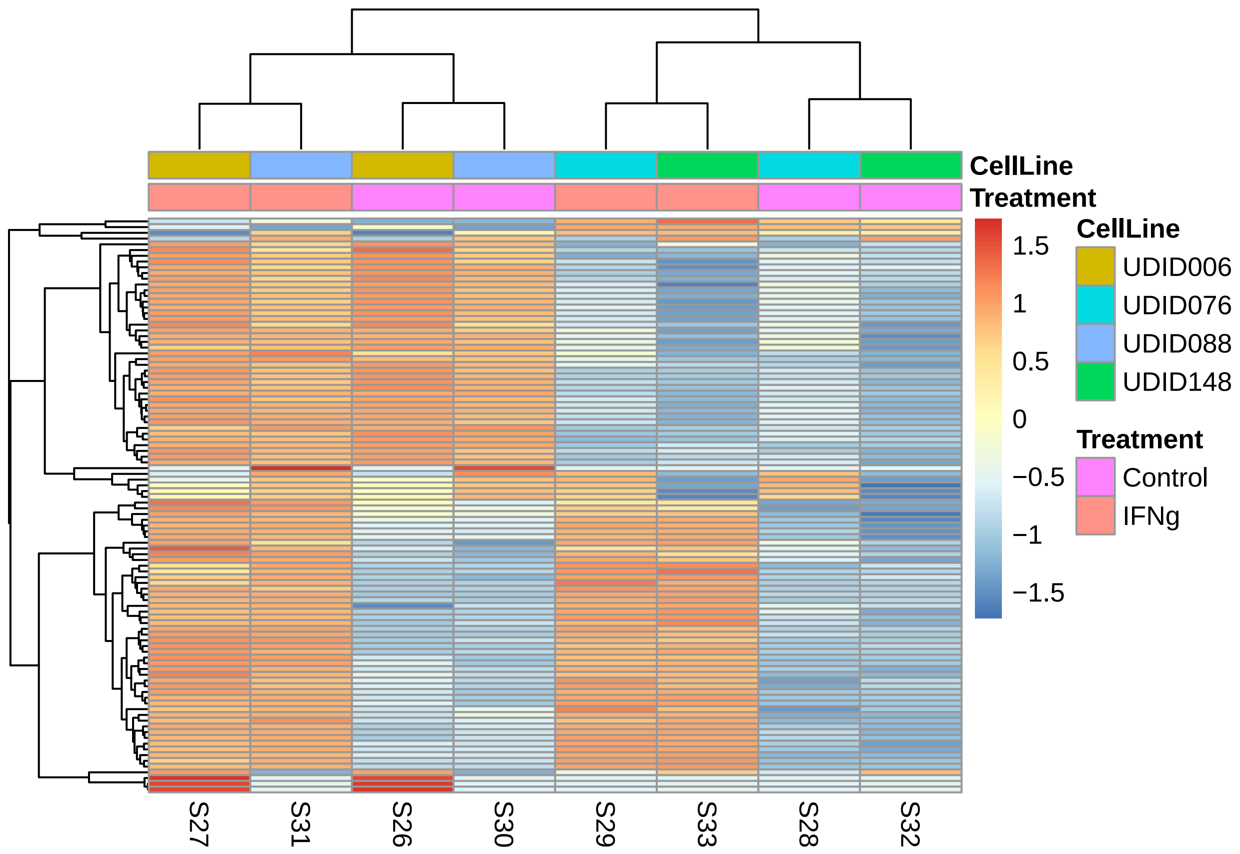


Other useful visualization give highlights of the effect of the treatment, i.e. INFg. First we compute a new table

```
### Clustering on results1$E
library(SummarizedExperiment)
se <- SummarizedExperiment(assays = list(counts = results1$E), colData = samples)
colData = as.data.frame(colData(se))
```

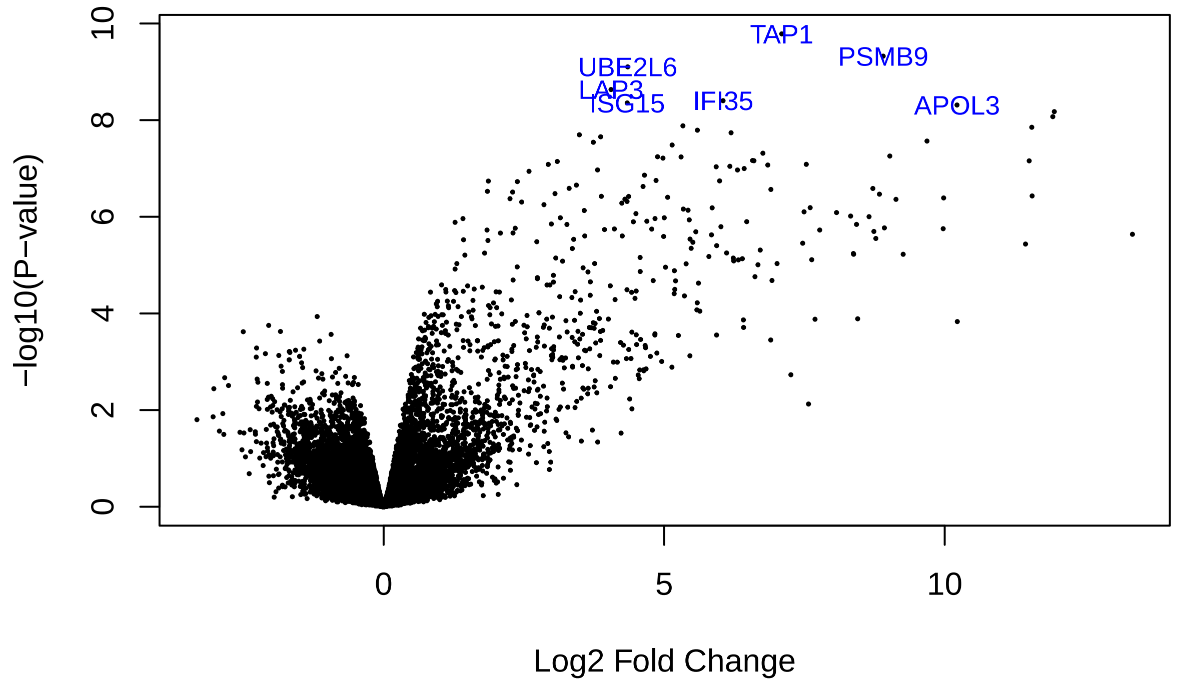
then we see the heatmap

```
library(pheatmap)
# Select the top 100 most variable genes among the samples
VG <- apply(results1$E, 1, var)
selectedGenes <- names(VG[order(VG, decreasing = T)])[1:100]

pheatmap(results1$E[selectedGenes, ],
         scale = 'row',
         show_rownames = FALSE,
         annotation_col = colData)
```



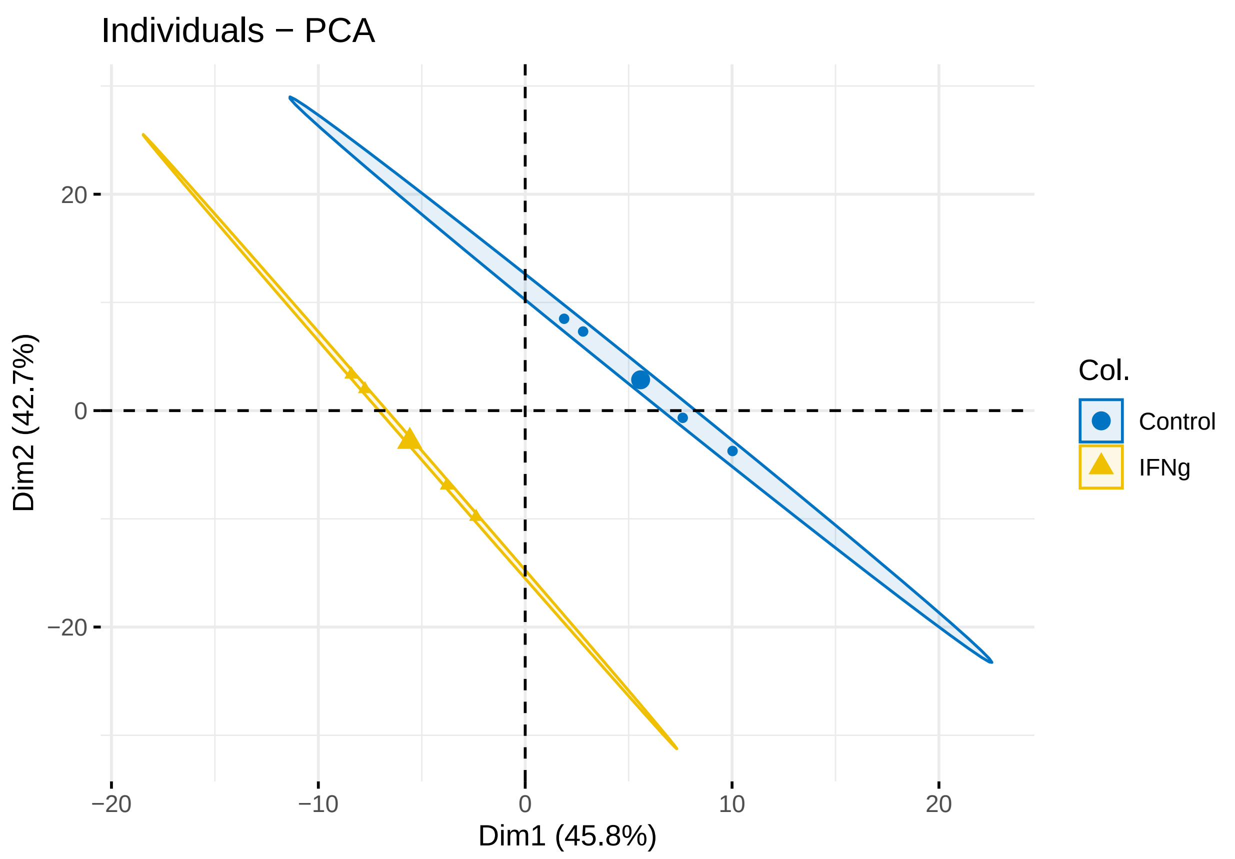
We can also use volcano plots. The first example uses a function of `limma`

```
# Volcano plot using volcanoplot() in limma
volcanoplot(results1$fit)

results1$preRank$genes <- rownames(counts)
volcanoplot(results1$fit, highlight = 7, names = names(results1$fit$Amean))
```



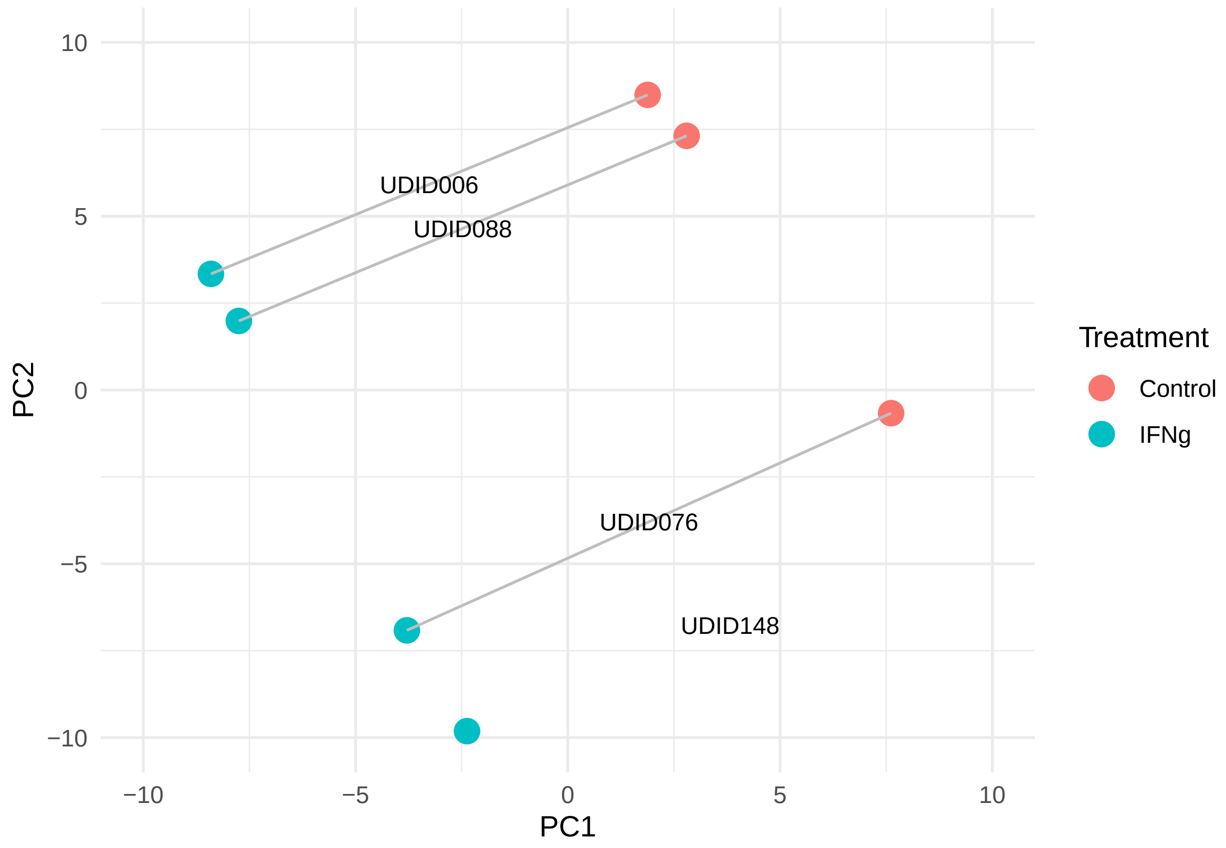
and finally, PCA results

```
library(factoextra) # for PCA
df_t <- t(results1$E)
# Using only most variable genes
pr1 <- prcomp(df_t[,selectedGenes], scale. = TRUE)

fviz_pca_ind(pr1,
             geom.ind = "point",
             col.ind = colData$Treatment,
             palette = "jco",
             addEllipses = TRUE,
             legene.title = "Treatment")
```



The heatmap shows that some genes have high fold change explained by Treatment and some my the difference between cell lines UDID006 and UDID088 (S26, S27, S30, S31) and the other two. PCA plot groups by Treatment, but we can also check how cell line look in two principal PCA components

```
pca_coords <- as.data.frame(pr1$x)
pca_coords$Sample <- rownames(pca_coords)
pca_coords$CellLine <- colData$CellLine
pca_coords$Treatment <- colData$Treatment  # Add treatment groups
pca_mid <- pca_coords %>%
  group_by(CellLine) %>%
  summarise(
    x_mid = mean(PC1),
    y_mid = mean(PC2)
  )
p <- ggplot(pca_coords, aes(x = PC1, y = PC2, color = Treatment)) +
  geom_point(size = 4) +
  # stat_ellipse(type = "norm", level = 0.95) +  # 95% confidence ellipses
  geom_line(aes(group = CellLine), color = "gray") +
  geom_text(data = pca_mid, aes(x=x_mid, y=y_mid, label=CellLine), color="black", size=3) +

  xlim(-10, 10) + ylim(-10, 10) +
  theme_minimal()
p
```



# Pathway analysis

We will use positive ranking function because it is easier to interpet with `plotEnrichment()`. Later I will add more interpretative functions. We start from assigning adjusted probabilities of set significance and identifying sets that pass 'padj < 0.05' conditions.

```
library(fgsea)
goBP <- gmtPathways(pa('c5.go.bp.v2024.1.Hs.symbols.gmt'))
Ranking <- function(x) {
  df <- x$preRank
  r <- (-log10(df$adj.P.Val)) # *sign(df$logFC)
  names(r) <- rownames(df)
  r <- sort(r, decreasing = TRUE)
  return(r)
}
Ranks <- function(x,y) {
  r <- Ranking(x)
  res <- fgsea(pathways = y,
                stats = r,
                scoreType = 'pos',
                minSize = 10,
                maxSize = 500,
                nproc = 1) %>%
    filter(padj < 0.05) %>% arrange(padj) # keep only more significant results
  res$common <- sapply(res[, 8], length)
  res <- res %>%
    rowwise() %>%
    mutate(common = length(cur_data()[[8]][[1]])) %>%
    ungroup()
  res$short <- substr(res$pathway,6,45)
  return(res)
}
r1 <- Ranks(results1,goBP)
```

```
##   |                                                                              |
```

```
r2 <- Ranks(results2,goBP)
```

```
##   |                                                                              |
```

```
sprintf("Design 1 identified %d biological processes, top 10:", dim(r1)[1])
```

```
## [1] "Design 1 identified 307 biological processes, top 10:"
```

```
print(r1[1:20,] %>% select(short,padj,common))
```

```
## # A tibble: 20 x 3
##    short                                         padj common
##    <chr>                                        <dbl>  <int>
##  1 REGULATION_OF_RESPONSE_TO_BIOTIC_STIMULU 8.45e-15     95
##  2 CYTOKINE_MEDIATED_SIGNALING_PATHWAY      5.24e-14     74
##  3 RESPONSE_TO_VIRUS                        1.44e-13     99
##  4 REGULATION_OF_INNATE_IMMUNE_RESPONSE     3.26e-12     76
##  5 VIRAL_PROCESS                            8.83e-12     66
##  6 POSITIVE_REGULATION_OF_DEFENSE_RESPONSE  2.20e-11     69
##  7 DEFENSE_RESPONSE_TO_VIRUS                4.06e-11     82
##  8 ACTIVATION_OF_IMMUNE_RESPONSE            2.30e-10     84
##  9 NEGATIVE_REGULATION_OF_IMMUNE_SYSTEM_PRO 1.78e- 9     74
## 10 IMMUNE_RESPONSE_REGULATING_SIGNALING_PAT 1.84e- 9     77
## 11 POSITIVE_REGULATION_OF_RESPONSE_TO_BIOTI 1.84e- 9     58
## 12 POSITIVE_REGULATION_OF_CYTOKINE_PRODUCTI 3.85e- 9     64
## 13 VIRAL_LIFE_CYCLE                         6.30e- 9     57
## 14 ADAPTIVE_IMMUNE_RESPONSE                 2.65e- 8     75
## 15 REGULATION_OF_LYMPHOCYTE_ACTIVATION      8.64e- 8     70
## 16 REGULATION_OF_IMMUNE_EFFECTOR_PROCESS    1.48e- 7     61
## 17 NEGATIVE_REGULATION_OF_RESPONSE_TO_EXTER 1.63e- 7     52
## 18 PROTEIN_POLYUBIQUITINATION               4.51e- 7     38
## 19 REGULATION_OF_RESPONSE_TO_CYTOKINE_STIMU 4.51e- 7     37
## 20 ACTIVATION_OF_INNATE_IMMUNE_RESPONSE     5.18e- 7     45
```

```
print(sprintf("Design 2 identified %d biological processes", dim(r2)[1]))
```

```
## [1] "Design 2 identified 1186 biological processes"
```

```
r2[1:20,] %>% select(short,padj,common)
```

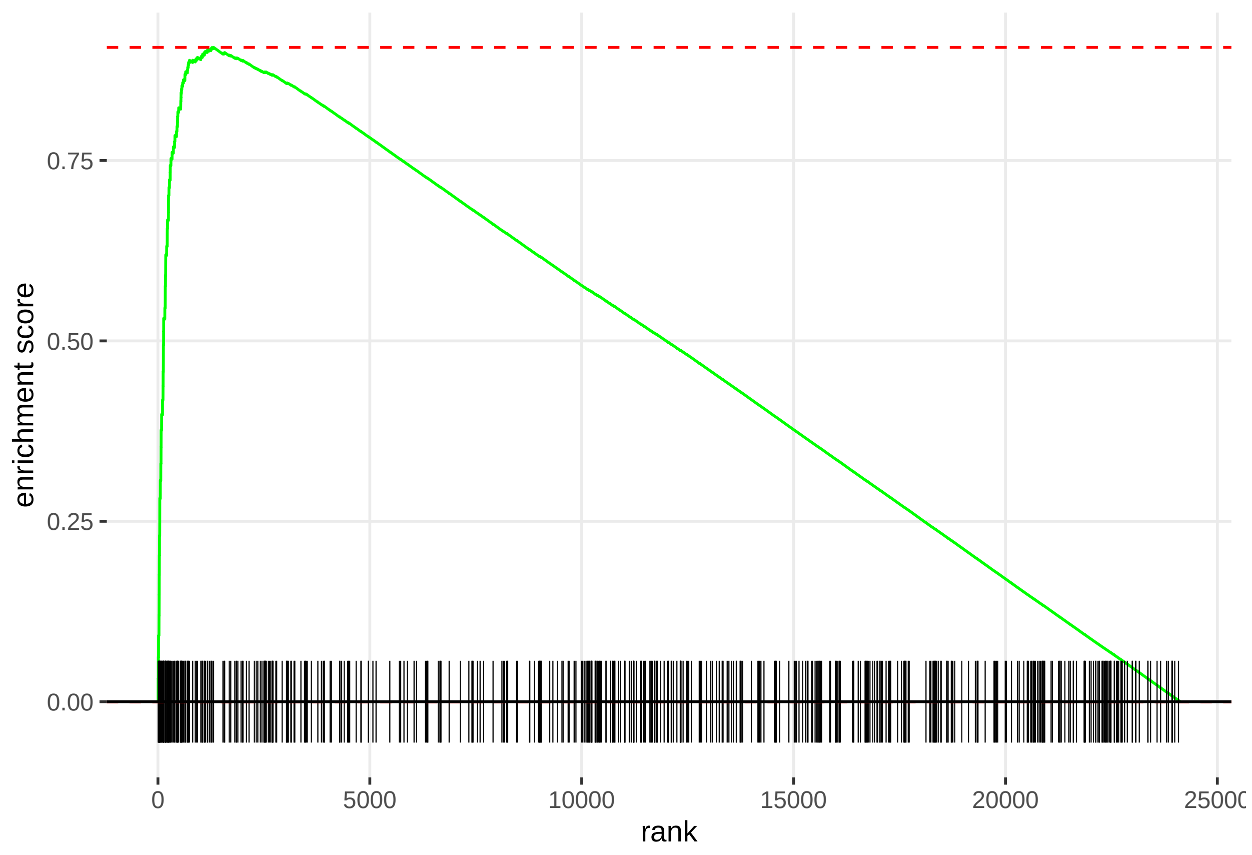
```
## # A tibble: 20 x 3
##    short                                     padj common
##    <chr>                                    <dbl>  <int>
##  1 RESPONSE_TO_VIRUS                     3.69e-43    119
##  2 DEFENSE_RESPONSE_TO_VIRUS             2.45e-40     94
##  3 ADAPTIVE_IMMUNE_RESPONSE              7.65e-26    129
##  4 REGULATION_OF_RESPONSE_TO_BIOTIC_STIMULU 1.90e-25    122
##  5 ACTIVATION_OF_IMMUNE_RESPONSE         2.16e-22    161
##  6 ADAPTIVE_IMMUNE_RESPONSE_BASED_ON_SOMATI 1.93e-20     97
##  7 CYTOKINE_MEDIATED_SIGNALING_PATHWAY   2.65e-20    146
##  8 NEGATIVE_REGULATION_OF_IMMUNE_SYSTEM_PRO 6.11e-20    115
##  9 POSITIVE_REGULATION_OF_CYTOKINE_PRODUCTI 6.59e-20    149
## 10 REGULATION_OF_INNATE_IMMUNE_RESPONSE  7.35e-20    139
## 11 REGULATION_OF_VIRAL_PROCESS           1.00e-19     56
## 12 LYMPHOCYTE_MEDIATED_IMMUNITY          1.81e-18     86
## 13 POSITIVE_REGULATION_OF_DEFENSE_RESPONSE  7.42e-18    155
## 14 IMMUNE_RESPONSE_REGULATING_SIGNALING_PAT 9.87e-18    158
## 15 NEGATIVE_REGULATION_OF_VIRAL_PROCESS  9.87e-18     41
## 16 LEUKOCYTE_MEDIATED_IMMUNITY           3.44e-17    113
## 17 INTERFERON_MEDIATED_SIGNALING_PATHWAY 1.12e-16     32
## 18 RESPONSE_TO_TYPE_II_INTERFERON        4.77e-16     46
## 19 DEFENSE_RESPONSE_TO_BACTERIUM         9.77e-16     55
## 20 NEGATIVE_REGULATION_OF_RESPONSE_TO_BIOTI 4.87e-15     35
```

The second design finds different genes in the "leading edge, so we can compare ACTIVATION_OF_IMMUNE_RESPONSE in our two designs.

```
print("ACTIVATION_OF_IMMUNE_RESPONSE in the 1st design")
```

```
## [1] "ACTIVATION_OF_IMMUNE_RESPONSE in the 1st design"
```

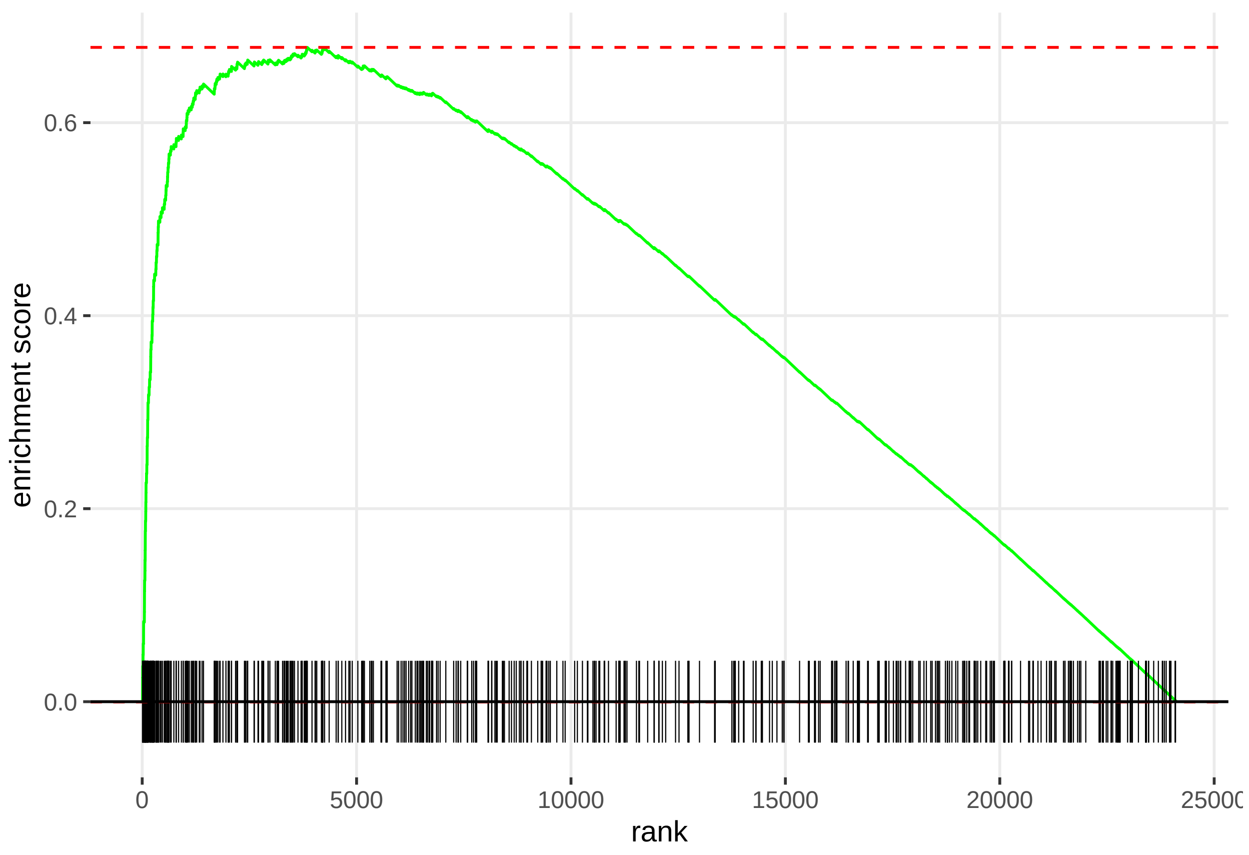
```
plotEnrichment(goBP[["GOBP_ACTIVATION_OF_IMMUNE_RESPONSE"]],Ranking(results1))
```



```
print("ACTIVATION_OF_IMMUNE_RESPONSE in the 2st design")
```

```
## [1] "ACTIVATION_OF_IMMUNE_RESPONSE in the 2st design"
```

```
plotEnrichment(goBP[["GOBP_ACTIVATION_OF_IMMUNE_RESPONSE"]],Ranking(results2))
```


enrichment score
0.6
0.4
0.2
0.0
0
5000
10000
15000
20000
25000
rank