Differential Gene Expression Analysis Using Salmon Quantification Data

Rashmi Dissasekara

2025-05-01

Overview

This R Markdown file performs differential gene expression (DGE) analysis using RNA-seq quantification data generated by Salmon, a lightweight and accurate tool for transcript-level quantification. The data used here corresponds to the experimental design and conditions reported in PMID: 40055794, which investigates transcriptional changes under different treatment conditions in breast cancer cell lines.

The analysis includes: - Importing quantification data with tximport - Differential gene expression analysis using DESeq2 - Visualization using PCA plots - Over-representation Analysis (OR) using GO gene datasets

Code: Package Installation

```
if (!require("readr")) {
  install.packages("readr")}

if (!require("ggplot2", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("ggplot2")
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'ggplot2'
```

```
if (!require("DESeq2", quietly = TRUE))
  BiocManager::install("DESeq2")

if (!require("ggrepel", quietly = TRUE))
  BiocManager::install("ggrepel")

if (!require("tximport", quietly = TRUE))
  BiocManager::install("tximport")

if(!require("tidyverse", quietly=TRUE))
  install.packages("tidyverse")

if (!require("ClusterProfiler", quietly=TRUE))
  BiocManager::install("clusterProfiler")
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'clusterProfiler'
```

```
library(tximport)
library(readr)
library(DESeq2)
library(ggrepel)
library("tidyverse")
library(clusterProfiler)
```

```
##

## clusterProfiler v4.14.6 Learn more at https://yulab-smu.top/contribution-knowledge-mining/
##
## Please cite:
##
## G Yu. Thirteen years of clusterProfiler. The Innovation. 2024,
## 5(6):100722

##
## Attaching package: 'clusterProfiler'

## The following object is masked from 'package:purrr':
##
##     simplify

## The following object is masked from 'package:IRanges':
##
##     slice

## The following object is masked from 'package:S4Vectors':
##
##     rename

## The following object is masked from 'package:stats':
##
##     filter
```

Load Sample Metadata

We load the metadata file `samples.csv`, which should contain columns like `Sample` and `Condition`. These columns are used to associate each sample with its experimental condition.

```
samples <- read.csv("samples.csv", header = TRUE)
samples$Condition
```

```
##  [1] "T47D-FulvR" "T47D-FulvR" "T47D-FulvR" "T47D-Par"   "T47D-Par"
##  [6] "T47D-Par"   "T47D-TamR"  "T47D-TamR"  "T47D-TamR"  "UCD4-EWD"
## [11] "UCD4-EWD"   "UCD4-EWD"   "UCD4-FulvR" "UCD4-FulvR" "UCD4-FulvR"
## [16] "UCD4-Par"   "UCD4-Par"   "UCD4-Par"   "UCD4-TamR"  "UCD4-TamR"
## [21] "UCD4-TamR"
```

```
samples$Sample
```

```
##  [1] "T47D_FulvR_1_quant"    "T47D_FulvR_2_quant"    "T47D_FulvR_3_quant"
##  [4] "T47D_parental_1_quant" "T47D_parental_2_quant" "T47D_parental_3_quant"
##  [7] "T47D_TamR_1_quant"     "T47D_TamR_2_quant"     "T47D_TamR_3_quant"
## [10] "UCD4_EWD_1_quant"      "UCD4_EWD_2_quant"      "UCD4_EWD_3_quant"
## [13] "UCD4_FulvR_1_quant"    "UCD4_FulvR_2_quant"    "UCD4_FulvR_3_quant"
## [16] "UCD4_parental_1_quant" "UCD4_parental_2_quant" "UCD4_parental_3_quant"
## [19] "UCD4_TamR_1_quant"     "UCD4_TamR_2_quant"     "UCD4_TamR_3_quant"
```

Define and Name Paths to Salmon Quantification Files

Here we construct the file paths to the `quant.sf` files generated by Salmon for each sample. It then prints the paths and checks if the files exist.Finally, it assigns sample names to each file path for easy identification.

```
files <- file.path(("salmon_quant"), samples$Sample, "quant.sf")
print(files)
file.exists(files)
```

```
names(files) <- samples$Sample
```

Load and Preview Transcript-to-Gene Mapping File

Now we load the `tx2gene.tsv` file, which maps transcript IDs to gene IDs, and previews the first few rows to ensure the file was loaded correctly.

```
tx2gene <- read.delim("tx2gene.tsv", header = FALSE, stringsAsFactors = FALSE)

head(tx2gene)
```

```
##                V1              V2
## 1 ENST00000832824 ENSG00000290825
## 2 ENST00000832825 ENSG00000290825
## 3 ENST00000832826 ENSG00000290825
## 4 ENST00000832827 ENSG00000290825
## 5 ENST00000832828 ENSG00000290825
## 6 ENST00000832829 ENSG00000290825
```

Filter and Load Data for T47D Cell Line

In this section, we filter the metadata to select samples from the T47D cell line. We then construct the file paths to their corresponding `quant.sf` files and load the Salmon data using `tximport`.

```
# Filter for T47D samples
T47D_samples <- samples[grep("T47D", samples$Sample), ]

#Construct the file paths to the quant.sf files
T47D_files <- file.path("salmon_quant", T47D_samples$Sample, "quant.sf")

#Use tximport to load the Salmon data
txi_T47D <- tximport(T47D_files, type = "salmon", tx2gene = tx2gene, ignoreTxVersion = TRUE)
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6 7 8 9
## transcripts missing from tx2gene: 13287
## summarizing abundance
## summarizing counts
## summarizing length
```

Create DESeq2 Dataset for T47D Cell Line

This step constructs a DESeq2 dataset (`dds_T47D_Txi`) from the `tximport` data for the T47D cell line. The design formula specifies the experimental condition as the variable of interest.

```
dds_T47D_Txi <- DESeqDataSetFromTximport(txi_T47D,
                                         colData = T47D_samples,
                                         design = ~ Condition)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]
```

```
## using counts and average transcript lengths from tximport
```

Pre-filtering and DESeq2 Analysis for T47D Cell Line

This section performs pre-filtering to retain genes with at least 50 counts in a minimum of 3 samples. Then, it runs DESeq2 for differential expression analysis and relevels the condition factor to set "T47D-Par" as the reference level.

```
smallestGroupSize <- 3
keep <- rowSums(counts(dds_T47D_Txi) >= 50) >= smallestGroupSize
ddsTxi <- dds_T47D_Txi[keep,]

dds <- ddsTxi

rm(ddsTxi)

# factor levels

dds$condition <- relevel(dds$Condition, ref = "T47D-Par")

dds <- DESeq(dds)
```

```
## estimating size factors
```

```
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## using 'avgTxLength' from assays(dds), correcting for library size

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## final dispersion estimates

## fitting model and testing
```

```
res <- results(dds)
res
```

```
## log2 fold change (MLE): Condition T47D.TamR vs T47D.FulvR
## Wald test p-value: Condition T47D.TamR vs T47D.FulvR
## DataFrame with 12420 rows and 6 columns
##                   baseMean log2FoldChange     lfcSE      stat      pvalue
##                  <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000000003   647.404      -0.570293 0.1396067  -4.08500 4.40776e-05
## ENSG00000000419  3024.467      -0.680883 0.0733402  -9.28390 1.63381e-20
## ENSG00000000457   515.064       0.484989 0.1022311   4.74405 2.09491e-06
## ENSG00000000460   782.358       0.816790 0.0839832   9.72563 2.34441e-22
## ENSG00000001036  1468.081      -0.515237 0.0739261  -6.96962 3.17802e-12
## ...                   ...            ...       ...       ...         ...
## ENSG00000293600  294.7907     -0.3185713 0.1385208 -2.299809  0.02145905
## ENSG00000293606   43.1315      1.4392062 0.5341378  2.694447  0.00705055
## ENSG00000293615   43.6573     -0.0407759 0.3326780 -0.122569  0.90244860
## ENSG00000293686  217.2422      0.3910352 0.1524938  2.564270  0.01033932
## ENSG00000310517 4126.6317     -0.1856936 0.0946807 -1.961262  0.04984845
##                       padj
##                  <numeric>
## ENSG00000000003 1.40774e-04
## ENSG00000000419 2.05443e-19
## ENSG00000000457 8.12244e-06
## ENSG00000000460 3.19742e-21
## ENSG00000001036 2.27991e-11
## ...                     ...
## ENSG00000293600   0.0397268
## ENSG00000293606   0.0147017
## ENSG00000293615   0.9284060
## ENSG00000293686   0.0206970
## ENSG00000310517   0.0836605
```

Data Normalization for T47D Cell Line

We normalize the data using three different methods: variance stabilizing transformation (VST), regularized log transformation (rlog), and a simple normalization transformation (`normTransform`). It then previews the results of the VST method.

```
vsd <- vst(dds, blind=FALSE)
rld <- rlog(dds, blind=FALSE)
# this gives log2(n + 1)
ntd <- normTransform(dds)

head(assay(vsd), 3)
```

```
##                         1        2        3         4        5         6
## ENSG00000000003 10.70257 10.47930 10.74800 10.050837 10.05537 10.149764
## ENSG00000000419 12.33835 12.26112 12.15331 11.648054 11.56625 11.648383
## ENSG00000000457 10.15266 10.14099 10.22352  9.965988 10.01131  9.967486
##                         7        8        9
## ENSG00000000003 10.32195 10.26436 10.39381
## ENSG00000000419 11.64710 11.71947 11.72158
## ENSG00000000457 10.46661 10.43138 10.36829
```

PCA Plot for T47D Cell Line

This section creates a PCA plot using the variance stabilizing transformation (VST) data, colored by the `Condition` variable, to visualize sample clustering based on gene expression.

```
## using ntop=500 top features by variance
```

Filter and Load Data for UCD4 Cell Line

This section filters the metadata to select samples from the UCD4 cell line. It then constructs the file paths to their corresponding `quant.sf` files and loads the Salmon data using `tximport`.

```
# Filter for UCD4 samples
UCD4_samples <- samples[grep("UCD4", samples$Sample), ]

#Construct the file paths to the quant.sf files
UCD4_files <- file.path("salmon_quant", UCD4_samples$Sample, "quant.sf")

#Use tximport to load the Salmon data
txi_UCD4 <- tximport(UCD4_files, type = "salmon", tx2gene = tx2gene, ignoreTxVersion = TRUE)
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12
## transcripts missing from tx2gene: 13287
## summarizing abundance
## summarizing counts
## summarizing length
```

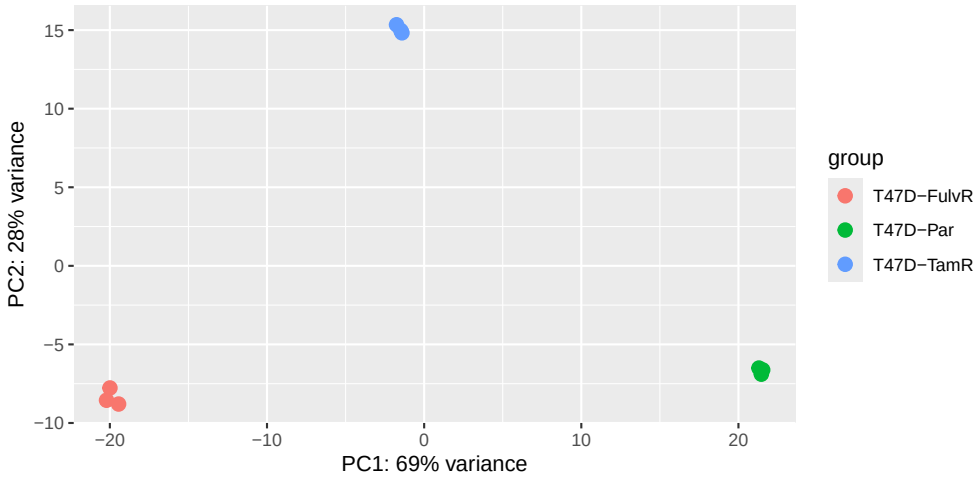


Figure 1: PCA plot of T47D samples colored by Condition

Create DESeq2 Dataset for UCD4 Cell Line

We construct a DESeq2 dataset (`dds_UCD4_Txi`) from the `tximport` data for the UCD4 cell line. The design formula specifies the experimental condition as the variable of interest.

```
dds_UCD4_Txi <- DESeqDataSetFromTximport(txi_UCD4,
                                         colData = UCD4_samples,
                                         design = ~ Condition)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]
```

```
## using counts and average transcript lengths from tximport
```

Pre-filtering and DESeq2 Analysis for UCD4 Cell Line

This section performs pre-filtering to retain genes with at least 50 counts in a minimum of 3 samples. Then, it runs DESeq2 for differential expression analysis and relevels the condition factor to set "UCD4-Par" as the reference level.

```r
smallestGroupSize <- 3
keep <- rowSums(counts(dds_UCD4_Txi) >= 50) >= smallestGroupSize
ddsTxiUCD4 <- dds_UCD4_Txi[keep,]

ddsUCD4 <- ddsTxiUCD4

rm(ddsTxiUCD4)

# factor levels

ddsUCD4$condition <- relevel(ddsUCD4$Condition, ref = "UCD4-Par")

ddsUCD4 <- DESeq(ddsUCD4)
```

```
## estimating size factors

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## using 'avgTxLength' from assays(dds), correcting for library size

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## final dispersion estimates

## fitting model and testing
```

```r
res <- results(ddsUCD4)
res
```

```
## log2 fold change (MLE): Condition UCD4.TamR vs UCD4.EWD
## Wald test p-value: Condition UCD4.TamR vs UCD4.EWD
## DataFrame with 13028 rows and 6 columns
##                   baseMean log2FoldChange     lfcSE      stat      pvalue
##                  <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000000003   300.387      0.2000529 0.1408299   1.42053 0.155453856
## ENSG00000000419  1434.322      0.0944114 0.0587372   1.60735 0.107977281
## ENSG00000000457   724.539      0.2765974 0.1036787   2.66783 0.007634261
## ENSG00000000460   291.709     -0.4404426 0.1203299  -3.66029 0.000251928
```

```
## ENSG00000001036  1263.523     -0.0962883 0.0673139  -1.43044 0.152591462
## ...                   ...            ...       ...       ...         ...
## ENSG00000293600  273.6336       0.323966 0.1346810  2.405430 1.61534e-02
## ENSG00000293606   38.1810       0.217690 1.3284985  0.163861 8.69840e-01
## ENSG00000293615   49.0853      -0.587284 0.2663586 -2.204863 2.74637e-02
## ENSG00000293686  354.2600       0.611213 0.1061073  5.760324 8.39529e-09
## ENSG00000310517 2539.6871       0.554784 0.0507177 10.938683 7.52853e-28
##                        padj
##                   <numeric>
## ENSG00000000003 0.199921889
## ENSG00000000419 0.143969708
## ENSG00000000457 0.012626788
## ENSG00000000460 0.000511955
## ENSG00000001036 0.196590228
## ...                     ...
## ENSG00000293600 2.53182e-02
## ENSG00000293606 8.94078e-01
## ENSG00000293615 4.13623e-02
## ENSG00000293686 2.63009e-08
## ENSG00000310517 6.43608e-27
```

Data Normalization for UCD4 Cell Line

This section normalizes the data using three different methods: variance stabilizing transformation (VST), regularized log transformation (rlog), and a simple normalization transformation (`normTransform`). It then previews the results of the VST method.

```
vsdUCD4 <- vst(ddsUCD4, blind=FALSE)
rldUCD4 <- rlog(ddsUCD4, blind=FALSE)
# this gives log2(n + 1)
ntdUCD4 <- normTransform(ddsUCD4)

head(assay(vsdUCD4), 3)
```

```
##                         1        2        3        4        5        6        7
## ENSG00000000003 10.84923 10.86475 10.81037 10.86621 10.79169 10.82954 10.89949
## ENSG00000000419 11.62879 11.61952 11.66726 11.64849 11.66334 11.65186 11.76228
## ENSG00000000457 11.20721 11.24299 11.25255 11.12155 11.07352 11.18612 11.32888
##                         8        9       10       11       12
## ENSG00000000003 10.87874 10.90901 10.96424 10.86561 10.84491
## ENSG00000000419 11.73056 11.76984 11.65333 11.72998 11.66697
## ENSG00000000457 11.26734 11.35105 11.36476 11.28188 11.37138
```

PCA Plot for UCD4 Cell Line

This section creates a PCA plot using the variance stabilizing transformation (VST) data, colored by the `Condition` variable, to visualize sample clustering based on gene expression.

```
## using ntop=500 top features by variance
```

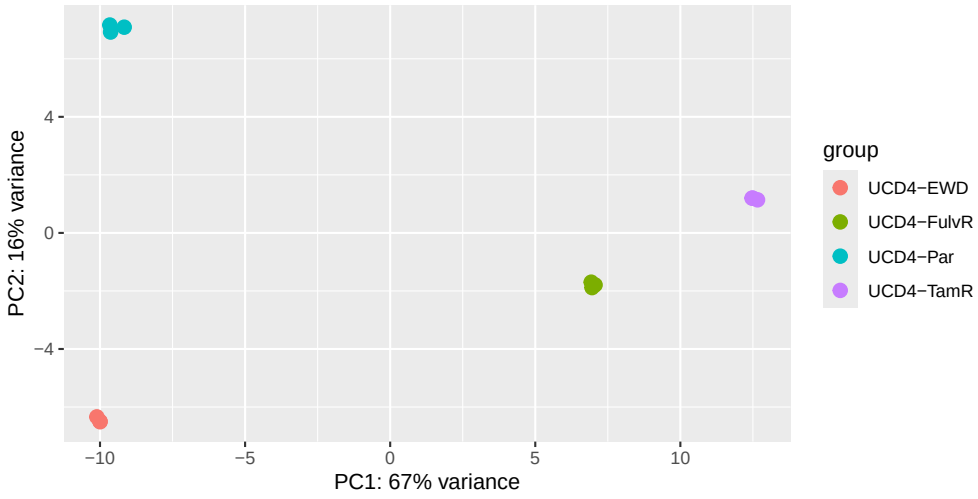


Figure 2: PCA plot of UCD4 samples colored by Condition

Extract Differential Expression Results for T47D and UCD4 Comparisons

Here we extract specific pairwise comparisons from the DESeq2 results for both T47D and UCD4 cell lines. These contrasts will help identify differentially expressed genes between resistant and parental conditions.

```
# T47D contrasts
res_T47D_FulvR_vs_Par <- results(dds, contrast = c("Condition", "T47D-FulvR", "T47D-Par"))
res_T47D_TamR_vs_Par <- results(dds, contrast = c("Condition", "T47D-TamR", "T47D-Par"))

# UCD4 contrasts
res_UCD4_FulvR_vs_Par <- results(ddsUCD4, contrast = c("Condition", "UCD4-FulvR", "UCD4-Par"))
res_UCD4_TamR_vs_Par <- results(ddsUCD4, contrast = c("Condition", "UCD4-TamR", "UCD4-Par"))
res_UCD4_EWD_vs_Par <- results(ddsUCD4, contrast = c("Condition", "UCD4-EWD", "UCD4-Par"))
```

Gene Annotation Using biomaRt

In this section, we annotate all Ensembl gene IDs found in our differential expression results using the biomaRt package. We retrieve gene descriptions, external gene names, and transcript lengths. To avoid redundancy, we retain only the longest transcript for each gene. This annotated table will be useful for downstream analyses like GO enrichment or gene-level summaries.

GO Term Annotation for DE Genes

We use biomaRt to retrieve Gene Ontology (GO) annotations for the differentially expressed genes across all comparisons. This includes GO IDs, term names, definitions, and categories such as biological_process, molecular_function, and cellular_component. The resulting annotations are saved to a file for use in downstream enrichment analysis or functional interpretation.

```
go_ann <- getBM(
  filter = "ensembl_gene_id",
  value = all_gene_ids,  # These are your DESeq2 result genes
  attributes = c(
    "ensembl_gene_id",      # The gene ID
    "description",          # Gene description (e.g., "transcription factor SOX2")
    "go_id",                # GO term ID (e.g., "GO:0003677")
    "name_1006",            # GO term name (e.g., "DNA binding")
    "definition_1006",      # GO term definition
    "namespace_1003"        # GO category: "biological_process", "molecular_function", or "cellular_com
  ),
  mart = ensembl
)


#let's look at the annotation:
filter(ann, description!="") %>% head()
```

```
##   ensembl_gene_id
## 1 ENSG00000000003
## 2 ENSG00000000419
## 3 ENSG00000000457
## 4 ENSG00000000460
## 5 ENSG00000001036
## 6 ENSG00000001084
##                                                                                 description
## 1                                                tetraspanin 6 [Source:HGNC Symbol;Acc:HGNC:11858]
## 2 dolichyl-phosphate mannosyltransferase subunit 1, catalytic [Source:HGNC Symbol;Acc:HGNC:3005]
## 3                                  SCY1 like pseudokinase 3 [Source:HGNC Symbol;Acc:HGNC:19285]
## 4  FIGNL1 interacting regulator of recombination and mitosis [Source:HGNC Symbol;Acc:HGNC:25565]
## 5                                       alpha-L-fucosidase 2 [Source:HGNC Symbol;Acc:HGNC:4008]
## 6                 glutamate-cysteine ligase catalytic subunit [Source:HGNC Symbol;Acc:HGNC:4311]
##   external_gene_name transcript_length
## 1             TSPAN6              3796
## 2               DPM1              3004
## 3              SCYL3              6308
## 4              FIRRM              4355
## 5              FUCA2              2385
## 6               GCLC              3785
```

```
head(ann)
```

```
##   ensembl_gene_id
## 1 ENSG00000000003
## 2 ENSG00000000419
## 3 ENSG00000000457
## 4 ENSG00000000460
## 5 ENSG00000001036
## 6 ENSG00000001084
##                                                                                 description
## 1                                                tetraspanin 6 [Source:HGNC Symbol;Acc:HGNC:11858]
## 2 dolichyl-phosphate mannosyltransferase subunit 1, catalytic [Source:HGNC Symbol;Acc:HGNC:3005]
## 3                                  SCY1 like pseudokinase 3 [Source:HGNC Symbol;Acc:HGNC:19285]
```

```
## 4  FIGNL1 interacting regulator of recombination and mitosis [Source:HGNC Symbol;Acc:HGNC:25565]
## 5                                       alpha-L-fucosidase 2 [Source:HGNC Symbol;Acc:HGNC:4008]
## 6                  glutamate-cysteine ligase catalytic subunit [Source:HGNC Symbol;Acc:HGNC:4311]
##   external_gene_name transcript_length
## 1             TSPAN6              3796
## 2               DPM1              3004
## 3              SCYL3              6308
## 4              FIRRM              4355
## 5              FUCA2              2385
## 6               GCLC              3785
```

```
# and Go term:

head(go_ann)
```

```
##   ensembl_gene_id
## 1 ENSG00000027001
## 2 ENSG00000027001
## 3 ENSG00000027001
## 4 ENSG00000027001
## 5 ENSG00000027001
## 6 ENSG00000027001
##                                                                 description
## 1 mitochondrial intermediate peptidase [Source:HGNC Symbol;Acc:HGNC:7104]
## 2 mitochondrial intermediate peptidase [Source:HGNC Symbol;Acc:HGNC:7104]
## 3 mitochondrial intermediate peptidase [Source:HGNC Symbol;Acc:HGNC:7104]
## 4 mitochondrial intermediate peptidase [Source:HGNC Symbol;Acc:HGNC:7104]
## 5 mitochondrial intermediate peptidase [Source:HGNC Symbol;Acc:HGNC:7104]
## 6 mitochondrial intermediate peptidase [Source:HGNC Symbol;Acc:HGNC:7104]
##       go_id                    name_1006
## 1
## 2 GO:0006508                  proteolysis
## 3 GO:0008237     metallopeptidase activity
## 4 GO:0004222 metalloendopeptidase activity
## 5 GO:0005737                    cytoplasm
## 6 GO:0046872            metal ion binding
##
## 1
## 2
## 3                                      Catalysis of the hydrolysis of peptide bonds by a mechanism
## 4 Catalysis of the hydrolysis of internal, alpha-peptide bonds in a polypeptide chain by a mechanism
## 5
## 6
##      namespace_1003
## 1
## 2 biological_process
## 3 molecular_function
## 4 molecular_function
## 5 cellular_component
## 6 molecular_function
```

```
# save go_genes object.
save(go_ann, file = "./go_ann.RData")
```

```
# you can later load this go_annotation object for future analyses
# go_ann <- load ("./go_ann.RData")

###
```

Annotating and Exporting DESeq2 Results

In this step, we merge the differential expression results from DESeq2 with gene annotation data (gene name, description, transcript length). This creates a biologically meaningful result table for each condition comparison. The merged tables are saved as individual .csv files for downstream analysis and reporting.

```
# List your DESeq2 result objects and corresponding labels
result_list <- list(
  T47D_TamR = res_T47D_TamR_vs_Par,
  T47D_FulvR = res_T47D_FulvR_vs_Par,
  UCD4_TamR = res_UCD4_TamR_vs_Par,
  UCD4_FulvR = res_UCD4_FulvR_vs_Par,
  UCD4_EWD = res_UCD4_EWD_vs_Par
)

# Initialize an empty list to store results
merged_results <- list()

# Loop through each DESeq2 result
for (label in names(result_list)) {

  res <- result_list[[label]]

  # Put results and annotation in the same table
  res_data <- res %>%
    as.data.frame() %>%
    rownames_to_column(var = "ensembl_gene_id")

  # Merge with annotation
  res_ann <- merge(x=res_data, y=ann, by.x="ensembl_gene_id", by.y="ensembl_gene_id", all.x=TRUE)

  # Add the result to the list
  merged_results[[label]] <- res_ann

  # Optionally, write the merged result to CSV
  write.csv(as.data.frame(res_ann),
            file = paste0("res_ann_", label, ".csv"))
}

# Check the first merged result (for example, T47D_TamR)
head(merged_results[["T47D_TamR"]])
```

```
##   ensembl_gene_id  baseMean log2FoldChange      lfcSE      stat       pvalue
## 1 ENSG00000000003  647.4037     0.49066912 0.14183025  3.459552 5.410749e-04
## 2 ENSG00000000419 3024.4665     0.09621473 0.07409275  1.298571 1.940911e-01
## 3 ENSG00000000457  515.0640     0.90327388 0.10349812  8.727442 2.604966e-18
```

```
## 4 ENSG00000000460  782.3583     0.16563411 0.08143754  2.033879 4.196378e-02
## 5 ENSG00000001036 1468.0812    -0.12358156 0.07458981 -1.656816 9.755677e-02
## 6 ENSG00000001084 1039.7515    -0.33808254 0.11674130 -2.895998 3.779552e-03
##           padj
## 1 1.257072e-03
## 2 2.587589e-01
## 3 2.337689e-17
## 4 6.729293e-02
## 5 1.419933e-01
## 6 7.529378e-03
##                                                                                 description
## 1                                                 tetraspanin 6 [Source:HGNC Symbol;Acc:HGNC:11858]
## 2 dolichyl-phosphate mannosyltransferase subunit 1, catalytic [Source:HGNC Symbol;Acc:HGNC:3005]
## 3                                  SCY1 like pseudokinase 3 [Source:HGNC Symbol;Acc:HGNC:19285]
## 4  FIGNL1 interacting regulator of recombination and mitosis [Source:HGNC Symbol;Acc:HGNC:25565]
## 5                                       alpha-L-fucosidase 2 [Source:HGNC Symbol;Acc:HGNC:4008]
## 6                 glutamate-cysteine ligase catalytic subunit [Source:HGNC Symbol;Acc:HGNC:4311]
##   external_gene_name transcript_length
## 1             TSPAN6              3796
## 2               DPM1              3004
## 3              SCYL3              6308
## 4              FIRRM              4355
## 5              FUCA2              2385
## 6               GCLC              3785
```

Over-representation Analysis (OR)

This section performs ORA using clusterProfiler on differentially expressed genes (padj < 0.1) from each DESeq2 comparison. Enrichment is conducted against Gene Ontology (GO) terms using pre-fetched annotations. Both the DE gene set and the background universe are defined per contrast. The enriched GO terms are saved as .csv files for each condition.

```
library(clusterProfiler)


# List of DESeq2 results with their labels
result_list <- list(
  T47D_TamR = res_T47D_TamR_vs_Par,
  T47D_FulvR = res_T47D_FulvR_vs_Par,
  UCD4_TamR = res_UCD4_TamR_vs_Par,
  UCD4_FulvR = res_UCD4_FulvR_vs_Par,
  UCD4_EWD = res_UCD4_EWD_vs_Par
)

# List to store results of ORA
ora_results <- list()

# Loop through each DESeq2 result for ORA
for (label in names(result_list)) {
  res <- result_list[[label]]

  # Get the DE genes (padj < 0.1)
  genes <- rownames(res[which(res$padj < 0.1),])
```

```
  # Get the universe (all genes with non-NA padj)
  univ <- rownames(res[!is.na(res$padj),])

  # Perform Over-Representation Analysis (ORA)
  res_enrich <- enricher(
    gene = genes,
    universe = univ,
    TERM2GENE = go_ann[, c(3, 1)], # Ensure this is the correct mapping
    TERM2NAME = unique(go_ann[, c(3, 4)])
  )

  # Store ORA results in the list
  ora_results[[label]] <- res_enrich

  # Optionally save the ORA results to CSV
  write.csv(as.data.frame(res_enrich),
            file = paste0("ORA_results_", label, ".csv"))
}

# check ORA results for a specific comparison, e.g.:
head(ora_results[["T47D_TamR"]])
```

Filtering for Lipid-Related GO Enrichments

To focus on biologically relevant pathways, this section filters GO enrichment results for terms containing the keyword "lipid." The filter_lipid_enrichments function searches the enrichment description column and retains only lipid-associated terms across all contrasts. Filtered results are stored in a named list for downstream analysis.

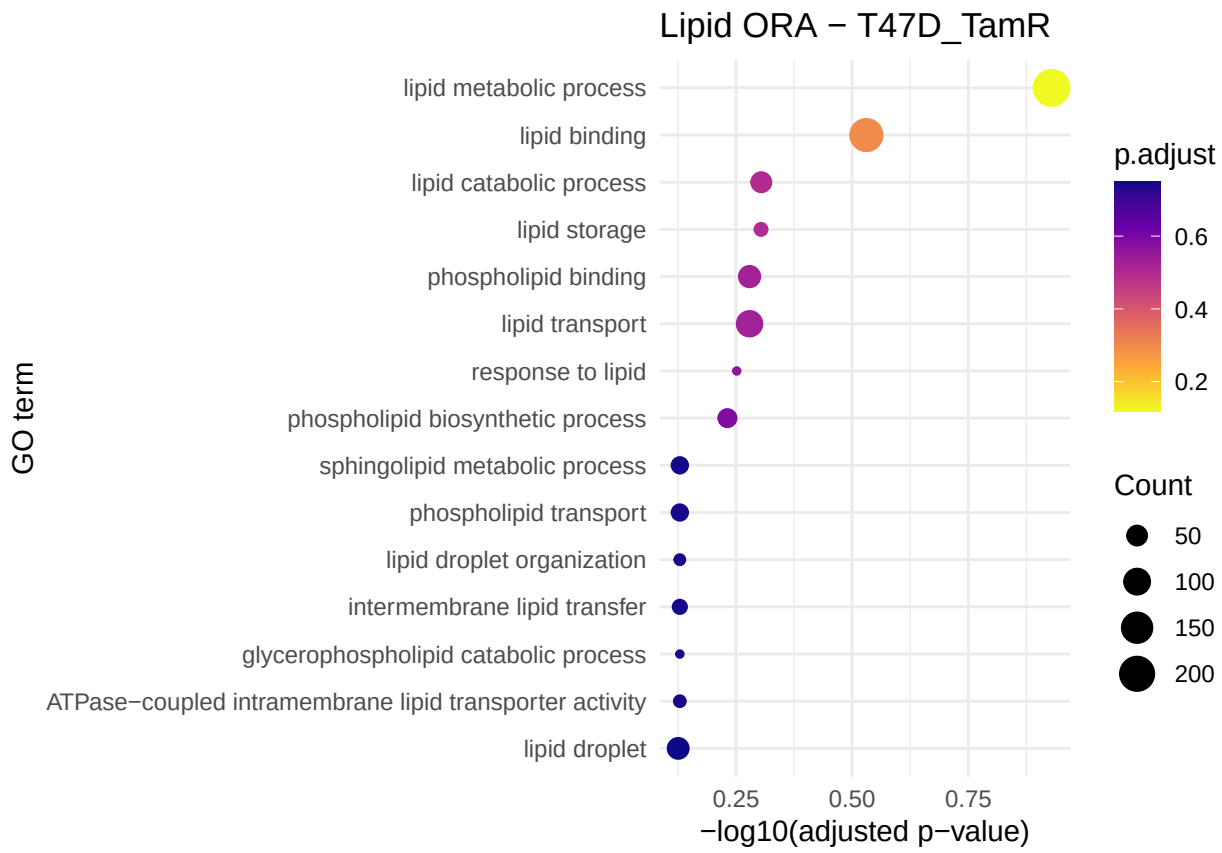
```
filter_lipid_enrichments <- function(enrich_obj) {
  if (is.null(enrich_obj)) return(NULL)
  if (!inherits(enrich_obj, "enrichResult")) return(NULL)

  df <- enrich_obj@result
  if (is.null(df) || !"Description" %in% colnames(df)) return(NULL)

  df %>% dplyr::filter(grepl("lipid", tolower(Description)))
}

lipid_ora_results <- lapply(ora_results, filter_lipid_enrichments)
head(lipid_ora_results)
```

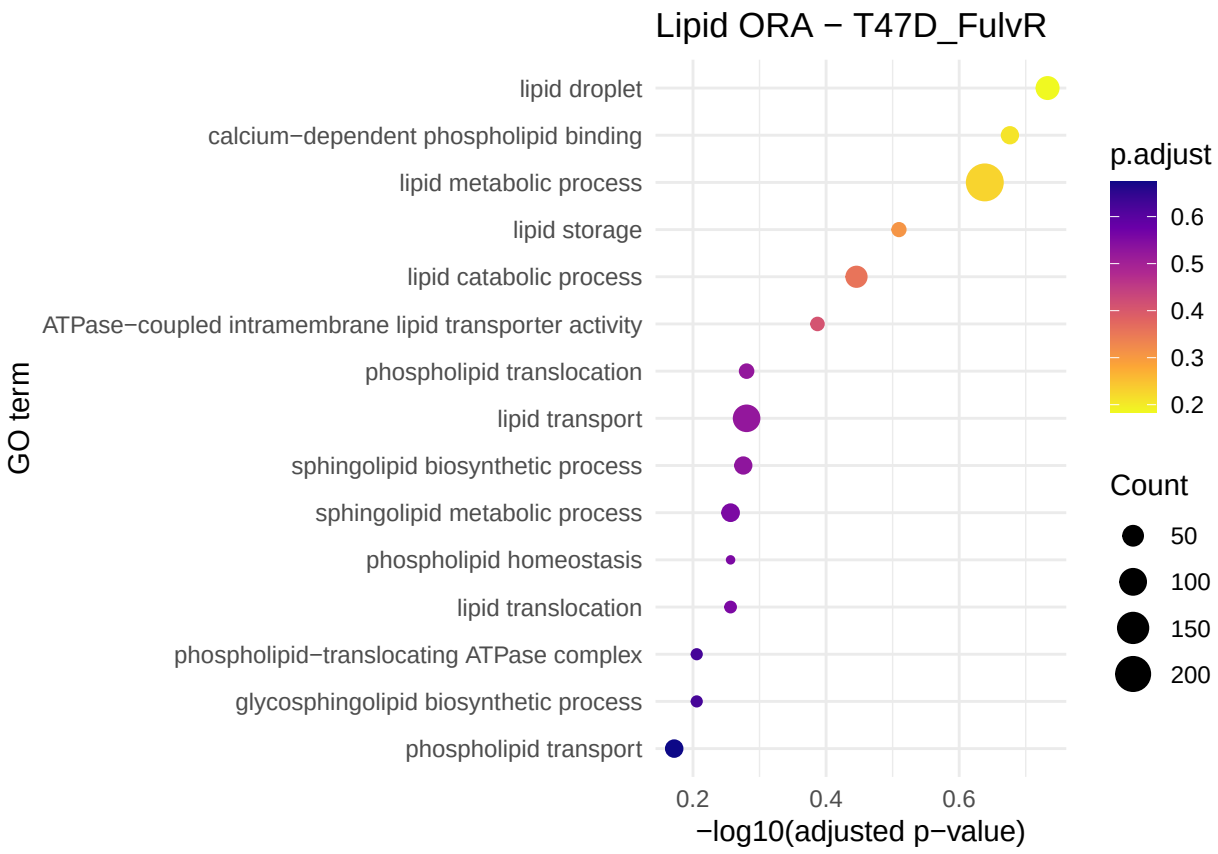
Visualization of Lipid-Associated GO Enrichments

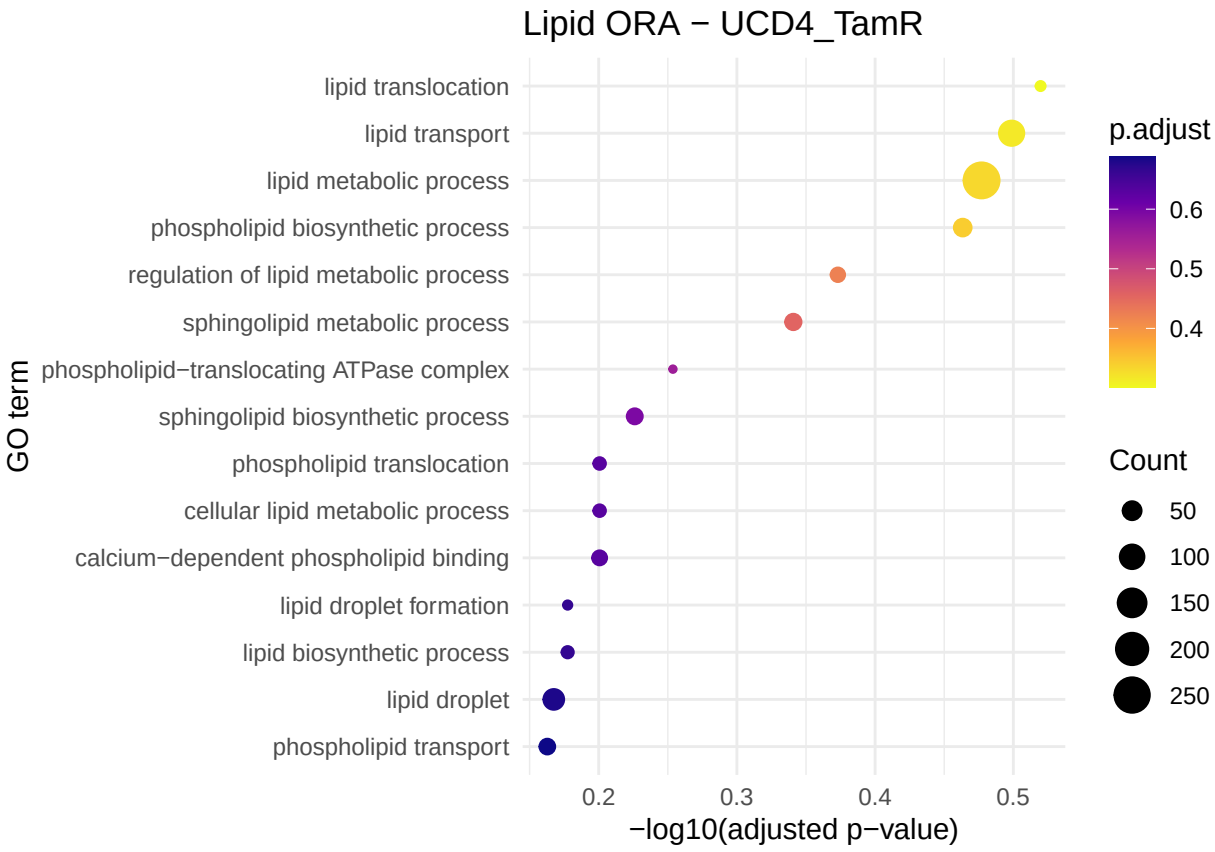
This section visualizes the top lipid-related GO terms (based on adjusted p-value) for each condition using dot plots. The plots display GO terms on the y-axis and their significance (-log10(p.adjust)) on the x-axis, with point size representing gene count and color indicating p-value significance. Only the top 15 most significant terms are shown per contrast.

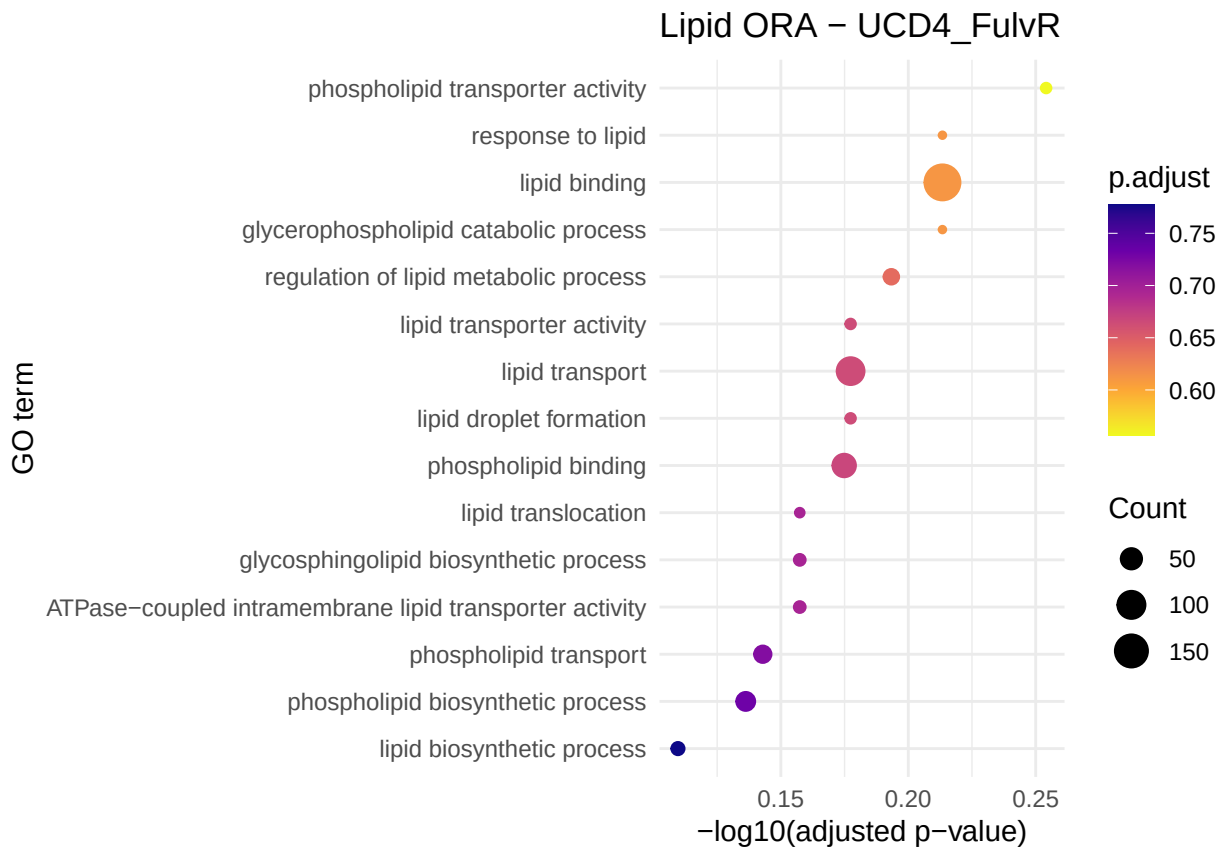
```
for (label in names(lipid_ora_results)) {
  df <- lipid_ora_results[[label]]
```

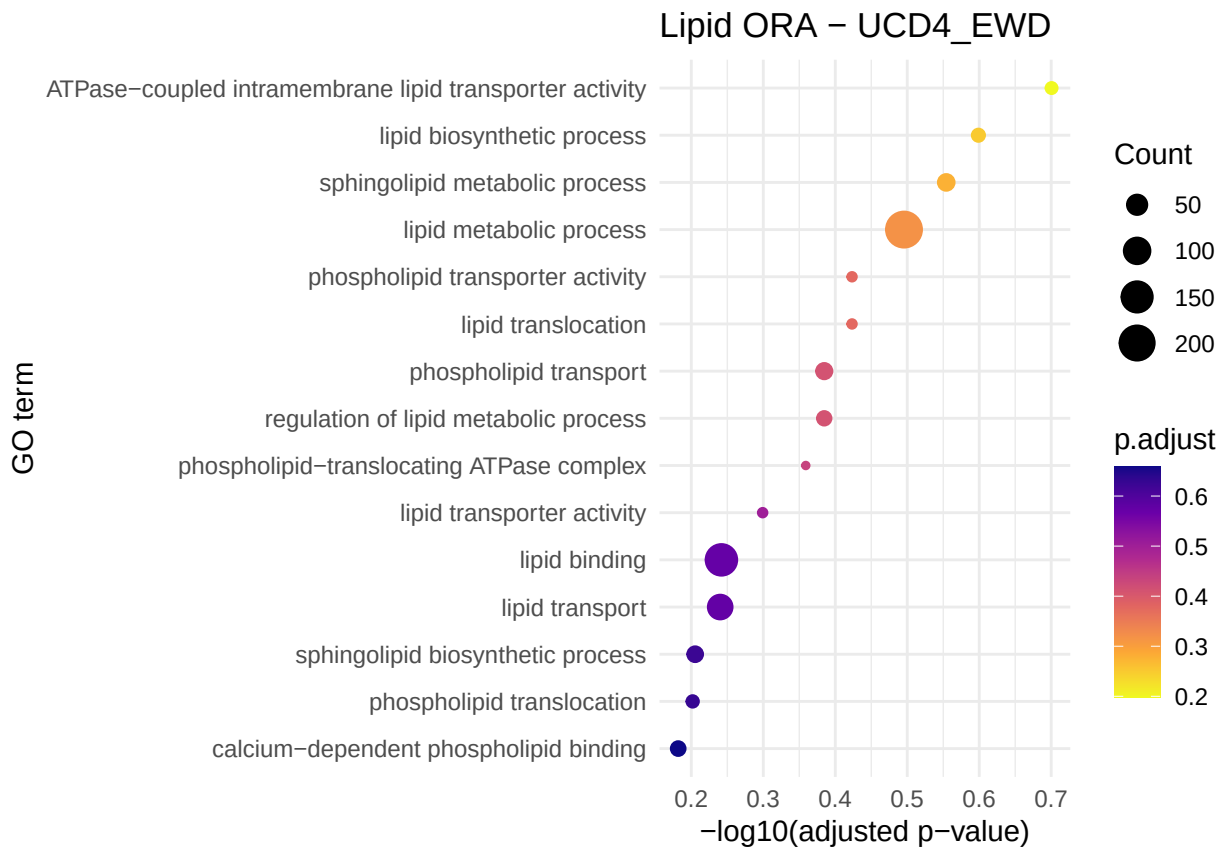
```r
  if (!is.null(df) && nrow(df) > 0) {
    df <- df[order(df$p.adjust), ]
    df_top <- head(df, 15)

    p <- ggplot(df_top, aes(x = -log10(p.adjust),
                            y = reorder(Description, -p.adjust),
                            size = Count,
                            color = p.adjust)) +
      geom_point() +
      scale_color_viridis_c(option = "C", direction = -1) +
      labs(title = paste("Lipid ORA -", label),
           x = "-log10(adjusted p-value)", y = "GO term") +
      theme_minimal()

    print(p)

    # Save as PDF
    ggsave(filename = paste0("Lipid_ORA_", label, ".pdf"), plot = p, width = 7, height = 5)
  }
}
```




Lipid ORA − T47D_FulvR
lipid droplet
calcium−dependent phospholipid binding
lipid metabolic process
lipid storage
lipid catabolic process
ATPase−coupled intramembrane lipid transporter activity
phospholipid translocation
lipid transport
sphingolipid biosynthetic process
sphingolipid metabolic process
phospholipid homeostasis
lipid translocation
phospholipid−translocating ATPase complex
glycosphingolipid biosynthetic process
phospholipid transport
GO term
0.2
0.4
0.6
−log10(adjusted p−value)
p.adjust
0.6
0.5
0.4
0.3
0.2
Count
50
100
150
200


Lipid ORA − UCD4_TamR
lipid translocation
lipid transport
lipid metabolic process
phospholipid biosynthetic process
regulation of lipid metabolic process
sphingolipid metabolic process
phospholipid−translocating ATPase complex
sphingolipid biosynthetic process
phospholipid translocation
cellular lipid metabolic process
calcium−dependent phospholipid binding
lipid droplet formation
lipid biosynthetic process
lipid droplet
phospholipid transport
GO term
0.2
0.3
0.4
0.5
−log10(adjusted p−value)
p.adjust
0.6
0.5
0.4
Count
50
100
150
200
250


Lipid ORA − UCD4_FulvR
phospholipid transporter activity
response to lipid
lipid binding
glycerophospholipid catabolic process
regulation of lipid metabolic process
lipid transporter activity
lipid transport
lipid droplet formation
phospholipid binding
lipid translocation
glycosphingolipid biosynthetic process
ATPase−coupled intramembrane lipid transporter activity
phospholipid transport
phospholipid biosynthetic process
lipid biosynthetic process
GO term
0.15
0.20
0.25
−log10(adjusted p−value)
p.adjust
0.75
0.70
0.65
0.60
Count
50
100
150



Identifying Upregulated and Downregulated Genes for T47D and UCD4 Here we identify the upregulated and downregulated genes across different conditions for the T47D and UCD4 cell lines. A gene is considered upregulated if its log2 fold change (LFC) is greater than or equal to 2 and its adjusted p-value (padj) is below 0.05. Conversely, a gene is considered downregulated if its LFC is less than or equal to -2 and its padj is below 0.05. The function get_upregulated_genes() and get_downregulated_genes() are used to filter the genes based on these criteria.

```
get_upregulated_genes <- function(res, lfc = 2, padj = 0.05) {
  up_genes <- rownames(res[which(res$padj < padj & res$log2FoldChange >= lfc), ])
  return(up_genes)
}

get_downregulated_genes <- function(res, lfc = 2, padj = 0.05) {
  down_genes <- rownames(res[which(res$padj < padj & res$log2FoldChange <= -lfc), ])
  return(down_genes)
}

# T47D upregulated and downregulated
up_t47d <- unique(c(
  get_upregulated_genes(res_T47D_TamR_vs_Par),
  get_upregulated_genes(res_T47D_FulvR_vs_Par)
))

down_t47d <- unique(c(
  get_downregulated_genes(res_T47D_TamR_vs_Par),
  get_downregulated_genes(res_T47D_FulvR_vs_Par)
))
```

```r
# UCD4 upregulated and downregulated
up_ucd4 <- unique(c(
  get_upregulated_genes(res_UCD4_TamR_vs_Par),
  get_upregulated_genes(res_UCD4_FulvR_vs_Par),
  get_upregulated_genes(res_UCD4_EWD_vs_Par)
))

down_ucd4 <- unique(c(
  get_downregulated_genes(res_UCD4_TamR_vs_Par),
  get_downregulated_genes(res_UCD4_FulvR_vs_Par),
  get_downregulated_genes(res_UCD4_EWD_vs_Par)
))
```

Normalized Expression Data for Upregulated and Downregulated Genes

Now we normalize the gene expression data using Variance Stabilizing Transformation (VST) for both T47D and UCD4 cell lines. After transforming the data, the expression values for the upregulated and downregulated genes identified previously are extracted for each cell line. The resulting matrices (up_t47d_mat, down_t47d_mat, up_ucd4_mat, down_ucd4_mat) contain the normalized expression data for these specific gene sets.

```r
# Transform the data (vst or rlog)
vsdT47D <- vst(dds, blind = TRUE)
vsdUCD4 <- vst(ddsUCD4, blind = TRUE)

# Subset the normalized expression data
up_t47d_mat <- assay(vsdT47D)[up_t47d, ]
down_t47d_mat <- assay(vsdT47D)[down_t47d, ]
up_ucd4_mat <- assay(vsdUCD4)[up_ucd4, ]
down_ucd4_mat <- assay(vsdUCD4)[down_ucd4, ]
```

Create Column Annotations for Conditions

The annotations are extracted from the colData of the DESeq2 datasets for each cell line, ensuring that each sample is labeled with its corresponding condition (e.g., T47D-Par, T47D-FulvR, etc.). The resulting data frames, annotation_t47d and annotation_ucd4, will be used for visualizing the data and adding relevant metadata to the heatmap.

```r
# Create column annotations for treatments
annotation_t47d <- as.data.frame(colData(dds)[, "Condition", drop=FALSE])
annotation_ucd4 <- as.data.frame(colData(ddsUCD4)[, "Condition", drop=FALSE])
```

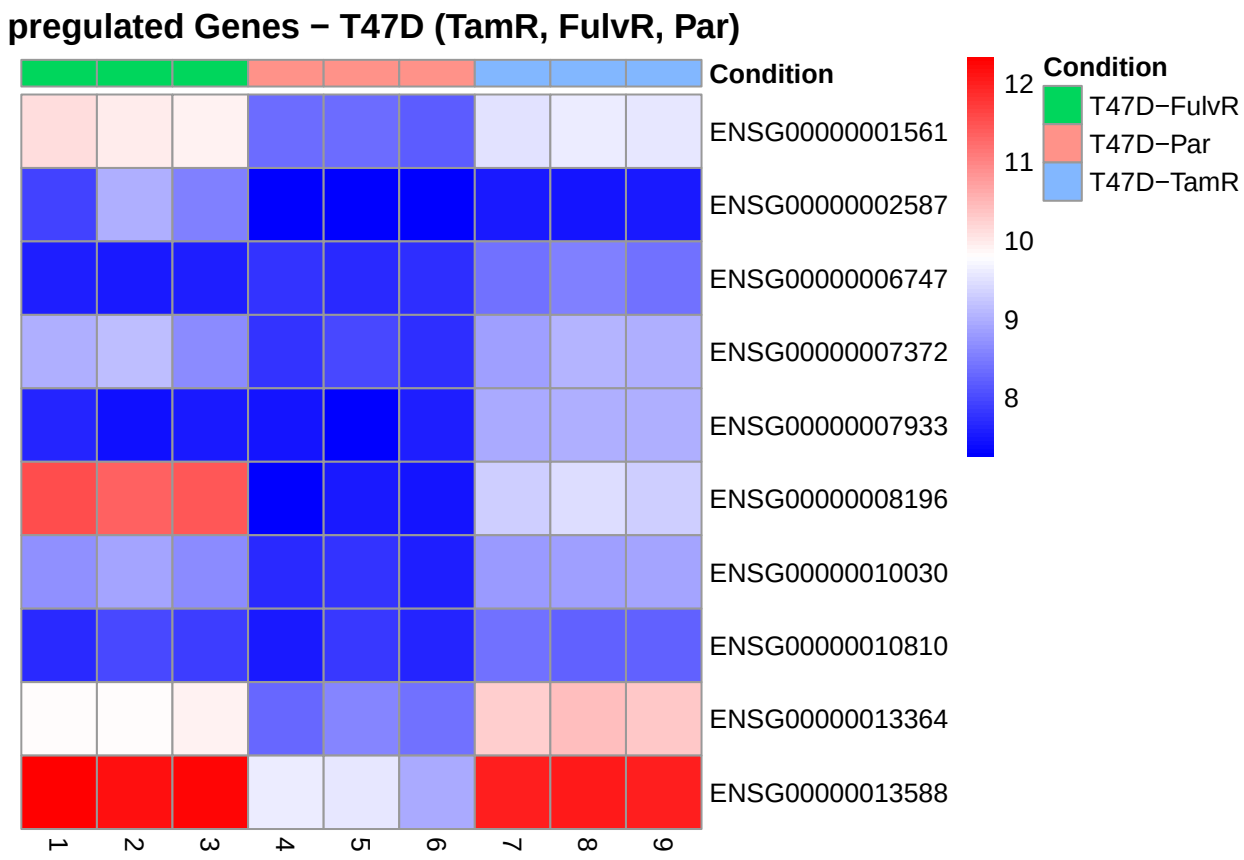
Generating the heatmaps

```r
install.packages("pheatmap")
```

```
## The following package(s) will be installed:
## - pheatmap [1.0.12]
## These packages will be installed into "~/GenomicsFinalProject/renv/library/windows/R-4.4/x86_64-w64-m
```

```
##
## # Installing packages --------------------------------------------------------
## - Installing pheatmap ...                       OK [linked from cache]
## Successfully installed 1 package in 26 milliseconds.
```

```
library(pheatmap)

# Heatmap for Upregulated T47D
pdf("Upregulated_Genes_T47D.pdf")
pheatmap(head(up_t47d_mat,10),
         annotation_col = annotation_t47d,
         show_rownames = TRUE,
         cluster_rows = FALSE,
         cluster_cols = FALSE,
         main = "Upregulated Genes - T47D (TamR, FulvR, Par)",
         color = colorRampPalette(c("blue", "white", "red"))(100))
```




```
dev.off()
```

```
## pdf
##   3
```

```
# Heatmap for Downregulated T47D
pdf("Downregulated_Genes_T47D.pdf")
```

```r
pheatmap(head(down_t47d_mat,10),
         annotation_col = annotation_t47d,
         show_rownames = TRUE,
         cluster_rows = FALSE,
         cluster_cols = FALSE,
         main = "Downregulated Genes - T47D (TamR, FulvR, Par)",
         color = colorRampPalette(c("blue", "white", "red"))(100))
dev.off()
```

```
## pdf
##   3
```

```r
# Heatmap for Upregulated UCD4
pdf("Upregulated_Genes_UCD4.pdf")
pheatmap(head(up_ucd4_mat,10),
         annotation_col = annotation_ucd4,
         show_rownames = TRUE,
         cluster_rows = FALSE,
         cluster_cols = FALSE,
         main = "Upregulated Genes - UCD4 (TamR, FulvR, EWD, Par)",
         color = colorRampPalette(c("blue", "white", "red"))(100))
dev.off()
```

```
## pdf
##   3
```

```r
# Heatmap for Downregulated UCD4
pdf("Downregulated_Genes_UCD4.pdf")
pheatmap(head(down_ucd4_mat,10),
         annotation_col = annotation_ucd4,
         show_rownames = TRUE,
         cluster_rows = FALSE,
         cluster_cols = FALSE,
         main = "Downregulated Genes - UCD4 (TamR, FulvR, EWD, Par)",
         color = colorRampPalette(c("blue", "white", "red"))(100))
dev.off()
```

```
## pdf
##   3
```