# Differential Gene Expression Analysis Using DESeq2

@PeepTheCode

DESeq2 uses a statistical model based on the negative binomial distribution to identify genes that show significant differences in expression between experimental conditions (for example, Control vs. PulmoAD).

In this tutorial, we'll perform Differential Expression Analysis (DEA) using the DESeq2 package in R. The goal is to identify genes that are significantly upregulated or downregulated between the two conditions (*Control* and *PulmoAD*) using RNA-Seq count data generated in the previous preprocessing tutorial.

All analyses in this tutorial have been conducted in RStudio. Make sure you have both R and RStudio installed. You can download and install them by following the instructions available here: https://posit.co/download/rstudio-desktop/

## Step 1: Install Required Packages

Before any analysis, install all the necessary packages that will be used throughout the workflow. R's Bioconductor repository hosts specialized bioinformatics tools. DESeq2 and other packages are maintained there to ensure reproducibility and compatibility.

```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")

BiocManager::install(c("DESeq2", "EnhancedVolcano",
"org.Hs.eg.db", “pheatmap”))
```

Packages installed:

- DESeq2: Core package for normalization, dispersion estimation, and statistical testing.
- EnhancedVolcano: For visually intuitive volcano plots.
- org.Hs.eg.db: Human gene annotation database (Ensembl ↔ Gene Symbol mapping).
- pheatmap: For heat map visualization

**Note:** Run this only once on your system. Skip re-installation during future sessions.

## Step 2. Load Libraries

Loading the installed packages makes their functions available for use in your current R session. Without loading, R won't recognize the functions (e.g., DESeq() or EnhancedVolcano()).

```
library(DESeq2)
library(EnhancedVolcano)
library(org.Hs.eg.db)
library(pheatmap)
```

This loads all required R libraries into your session so their functions can be used. If you get an error like "package not found," rerun Step 1 to install it.

## Step 3: Set Working Directory

This tells R where to look for your files and where to save outputs. Setting this ensures that R reads and writes data files without needing to repeatedly specify full file paths. Replace the path with your own directory containing the count matrix file.

```
setwd("~/MyFiles/RNAseq_Tutorial")
# Change this to your folder containing count matrix
getwd()
```

**Note:** Always confirm your path with getwd(). If your count file is not found, double-check that it is located in this directory.

## Step 4: Load the Count Matrix

The count matrix contains raw read counts obtained from RNA-seq read quantification tools like featureCounts (in this tutorial), HTSeq, or Salmon. DESeq2 requires *raw counts* because it uses the negative binomial model that depends on integer count distributions.

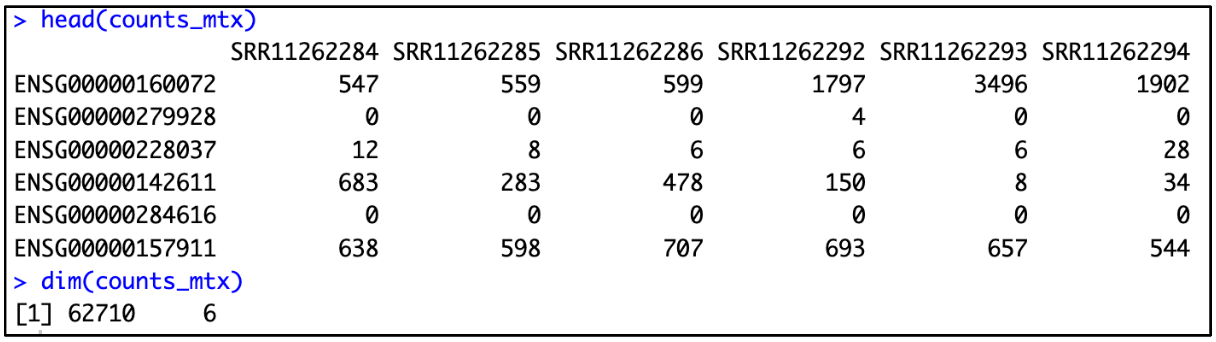
Expected format:

- Rows: Genes (usually Ensembl IDs)
- Columns: Samples
- Values: Raw integer counts (untransformed, unnormalized)

```
counts_mtx <- read.table("merged_counts_clean.txt",
                         header = TRUE,
                         row.names = 1,
                         sep = "\t",
                         check.names = FALSE)
```

Check your loaded count matrix:

```
head(counts_mtx)
dim(counts_mtx)
```



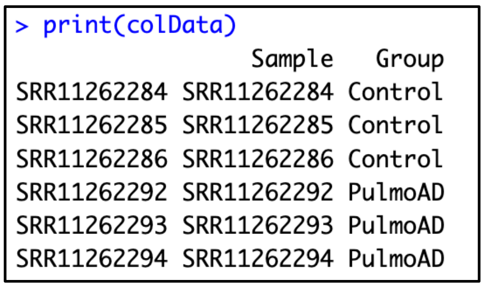
```
> head(counts_mtx)
                SRR11262284 SRR11262285 SRR11262286 SRR11262292 SRR11262293 SRR11262294
ENSG00000160072         547         559         599        1797        3496        1902
ENSG00000279928           0           0           0           4           0           0
ENSG00000228037          12           8           6           6           6          28
ENSG00000142611         683         283         478         150           8          34
ENSG00000284616           0           0           0           0           0           0
ENSG00000157911         638         598         707         693         657         544
> dim(counts_mtx)
[1] 62710     6
```

**Note:** If your count data are already log-transformed or normalized (e.g., TPM, FPKM), they cannot be used with DESeq2.

## Step 5: Create Sample Metadata

The metadata (or *colData*) describes each sample and its corresponding condition (e.g., Control vs. Disease). DESeq2 uses this data frame to understand which samples belong to which experimental group when fitting the model. The sample names in colData must match exactly the column names in the count matrix. Example **-** If your count matrix columns are named differently (e.g., “sample1”, “sample2”), update the sample_names vector accordingly.

```
sample_names <- c("SRR11262284", "SRR11262285", "SRR11262286",
                  "SRR11262292", "SRR11262293", "SRR11262294")

condition <- c("Control", "Control", "Control",
               "PulmoAD", "PulmoAD", "PulmoAD")

colData <- data.frame(
  Sample = sample_names,
  Group = condition,
  stringsAsFactors = FALSE
)

rownames(colData) <- sample_names
print(colData)
```



```
> print(colData)
                 Sample   Group
SRR11262284 SRR11262284 Control
SRR11262285 SRR11262285 Control
SRR11262286 SRR11262286 Control
SRR11262292 SRR11262292 PulmoAD
SRR11262293 SRR11262293 PulmoAD
SRR11262294 SRR11262294 PulmoAD
```

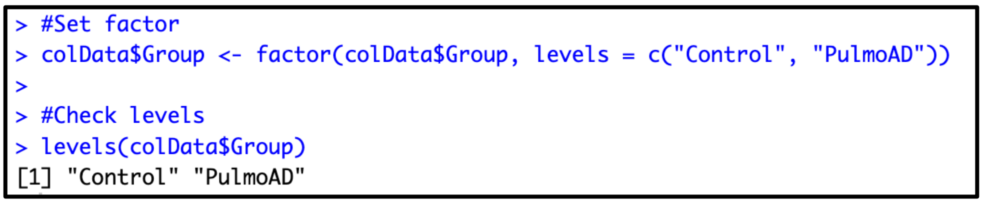
**Note:** Ensure that the sample names match exactly with column names in your count matrix.

### Step 6: Set a Factor Levels

In DESeq2, when you have a factor with two levels, it will compare - second level VS first level. By setting factor, you are telling DESeq2 that:

- *"Control" is the reference group*
- *"PulmoAD" will be compared against Control*

```
colData$Group <- factor(colData$Group, levels = c("Control",
"PulmoAD"))

#Check levels
levels(colData$Group)
```



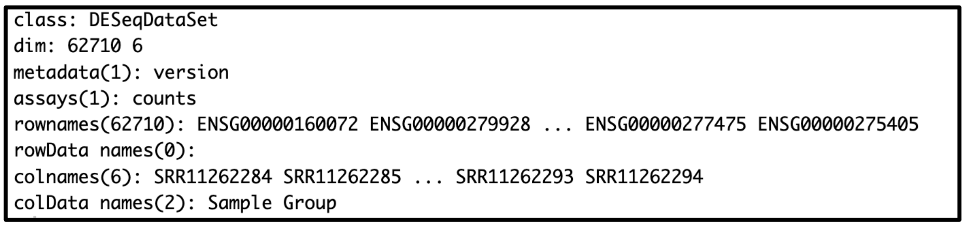
```
> #Set factor
> colData$Group <- factor(colData$Group, levels = c("Control", "PulmoAD"))
>
> #Check levels
> levels(colData$Group)
[1] "Control" "PulmoAD"
```

This ensures the results answer the question - "*How is PulmoAD (disease) different from Control*?" If you don't set the levels yourself, R might choose the reference automatically (alphabetical order), which may give you the comparison in the opposite direction.

### Step 7: Create and Preprocess DESeq2 Dataset

Convert the raw count matrix into a DESeq2 dataset (dds) which links count data and experimental design. DESeq2 uses this structured object to estimate normalization factors and fit its statistical model.

```
dds <- DESeqDataSetFromMatrix(countData = counts_mtx,
                              colData = colData,
                              design = ~ Group)

dds
```

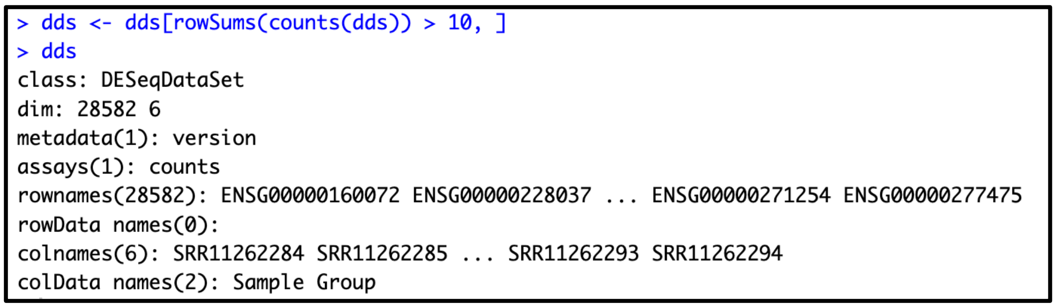


```
class: DESeqDataSet
dim: 62710 6
metadata(1): version
assays(1): counts
rownames(62710): ENSG00000160072 ENSG00000279928 ... ENSG00000277475 ENSG00000275405
rowData names(0):
colnames(6): SRR11262284 SRR11262285 ... SRR11262293 SRR11262294
colData names(2): Sample Group
```

**Note:** Your colData should be a small table where each row is a sample and each column describe something about that sample such as group (control/treatment), batch, sex, or cell line. The design should then use the column names from this table to tell DESeq2 what differences you want to test. For example, if your colData has a column called “Group,” you use design = ~ Group to compare those groups. If you also have a batch column and want to correct for it, you use design = ~ Batch + Group. In other words, colData lists the sample information, and the design chooses which of those variables DESeq2 should analyze. Here, the design formula ~ Group specifies that you want to test for differences in gene expression between the two groups defined by the "Group" variable.

**Step 8: Filter out low-count genes (optional but recommended)**

```
dds <- dds[rowSums(counts(dds)) > 10, ]
dds
```



```
> dds <- dds[rowSums(counts(dds)) > 10, ]
> dds
class: DESeqDataSet
dim: 28582 6
metadata(1): version
assays(1): counts
rownames(28582): ENSG00000160072 ENSG00000228037 ... ENSG00000271254 ENSG00000277475
rowData names(0):
colnames(6): SRR11262284 SRR11262285 ... SRR11262293 SRR11262294
colData names(2): Sample Group
```

Removes genes with very low read counts (sum ≤ 10 across all samples). These genes usually provide little information and can inflate false positives.

Estimate Size Factors:

```
dds <- estimateSizeFactors(dds)
```

DESeq2 estimates size factors to adjust for library size differences (sequencing depth differences across samples). After this step, samples with more reads are scaled down and samples with fewer reads are scaled up. This allows fair comparison of gene expression between samples.

**Step 9: Exploratory Analysis**

***Regularized Log Transformation***

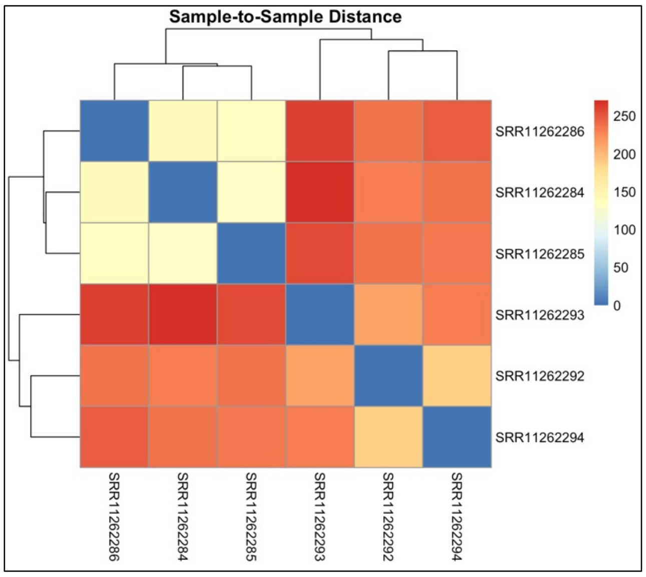
The rlog (regularized log) transformation is a method used to stabilize variance and makes count data more comparable between samples and suitable for clustering or PCA. It converts raw count data into a log-like scale while shrinking low-count variance. It behaves like a log2 transform for high counts but applies regularization (shrinkage) for low counts. In simple words, it makes samples more comparable. For large datasets (e.g., > 30 samples), DESeq2 recommends variance-stabilizing transformation (VST) instead because rlog is computationally heavy and VST produces similar variance stabilization much faster (dds_vst = vst(dds,blind = FALSE).

```
dds_rlog <- rlog(dds, blind = FALSE)
mat <- assay(dds_rlog)
```

### *Sample Distance Heatmap*

This visualizes the overall similarity between samples based on expression patterns. Samples that cluster together are more similar, often reflecting biological replicates. Samples within the same condition should cluster together; outliers may indicate technical artifacts. In other words, small distance values (close to 0) indicate that samples are very similar in their expression profiles whereas large distance values indicate that samples are dissimilar.

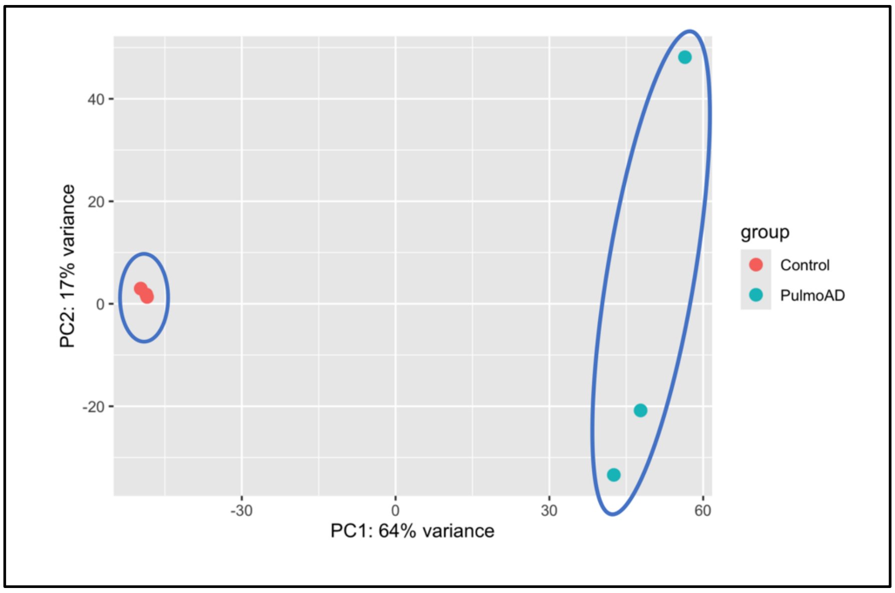
```
d <- dist(t(mat))

pheatmap(as.matrix(d),
         clustering_distance_rows = d,
         clustering_distance_cols = d,
         main = "Sample-to-Sample Distance")
```



### *PCA Plot*

PCA reduces thousands of gene expression values into principal components that capture the majority of variance. This helps visualize how distinct your experimental groups are. Samples should ideally separate along PC1 or PC2 according to their experimental condition.

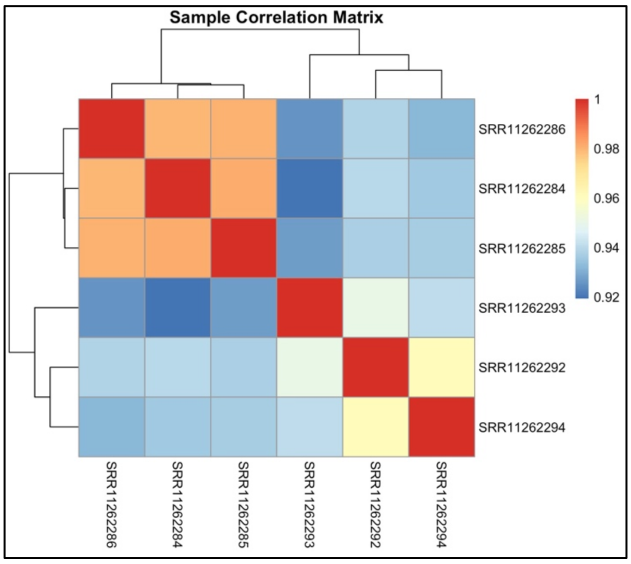
```
plotPCA(dds_rlog, intgroup = "Group")
```



***Correlation Heatmap***

Calculates Pearson correlations between samples. High correlation among replicates indicates good data quality.

```
cor.mat <- cor(mat)
pheatmap(cor.mat, main = "Sample Correlation Matrix")
```



**Step 10: Run DESeq2 Differential Expression Analysis**

This step executes the main DESeq2 pipeline, including:

- Estimating size factors (normalization)
- Estimating gene-wise dispersion

- Fitting the negative binomial model
- Performing hypothesis testing for differential expression

This is the core computational step where DESeq2 identifies genes whose expression differs significantly between conditions.

```
dds <- DESeq(dds)
```

### Step 11: Extract Results

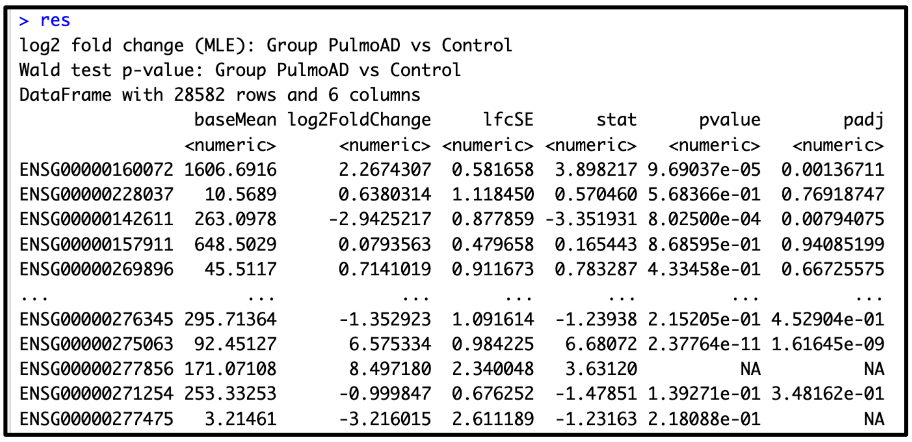
This step extracts statistical results for each gene, ordered by adjusted p-value (FDR). Then, maps Ensembl IDs to human-readable gene symbols for clarity.

The output provides for each gene:

- log2FoldChange - magnitude of expression change
- p-value - statistical significance
- padj (FDR) - false discovery rate–adjusted significance

A negative log2FoldChange means downregulation in PulmoAD; a positive one indicates upregulation.

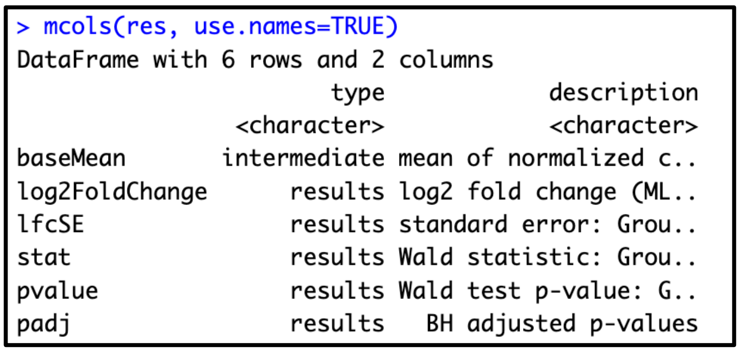
```
res <- results(dds, contrast = c("Group", "PulmoAD",
"Control"))

res
```



```
> res
log2 fold change (MLE): Group PulmoAD vs Control
Wald test p-value: Group PulmoAD vs Control
DataFrame with 28582 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat      pvalue        padj
                <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
ENSG00000160072 1606.6916      2.2674307  0.581658  3.898217 9.69037e-05  0.00136711
ENSG00000228037   10.5689      0.6380314  1.118450  0.570460 5.68366e-01  0.76918747
ENSG00000142611  263.0978     -2.9425217  0.877859 -3.351931 8.02500e-04  0.00794075
ENSG00000157911  648.5029      0.0793563  0.479658  0.165443 8.68595e-01  0.94085199
ENSG00000269896   45.5117      0.7141019  0.911673  0.783287 4.33458e-01  0.66725575
...                   ...            ...       ...       ...         ...         ...
ENSG00000276345 295.71364      -1.352923  1.091614  -1.23938 2.15205e-01 4.52904e-01
ENSG00000275063  92.45127       6.575334  0.984225   6.68072 2.37764e-11 1.61645e-09
ENSG00000277856 171.07108       8.497180  2.340048   3.63120          NA          NA
ENSG00000271254 253.33253      -0.999847  0.676252  -1.47851 1.39271e-01 3.48162e-01
ENSG00000277475   3.21461      -3.216015  2.611189  -1.23163 2.18088e-01          NA
```

Check the description of each results column:

```
mcols(res, use.names=TRUE)
```



```
> mcols(res, use.names=TRUE)
DataFrame with 6 rows and 2 columns
                       type            description
                <character>            <character>
baseMean       intermediate mean of normalized c..
log2FoldChange      results log2 fold change (ML..
lfcSE               results standard error: Grou..
stat                results Wald statistic: Grou..
pvalue              results Wald test p-value: G..
padj                results   BH adjusted p-values
```

Sort the results by p-adjusted values:

```
resOrdered <- res[order(res$padj), ]
```

Clean Ensembl IDs:

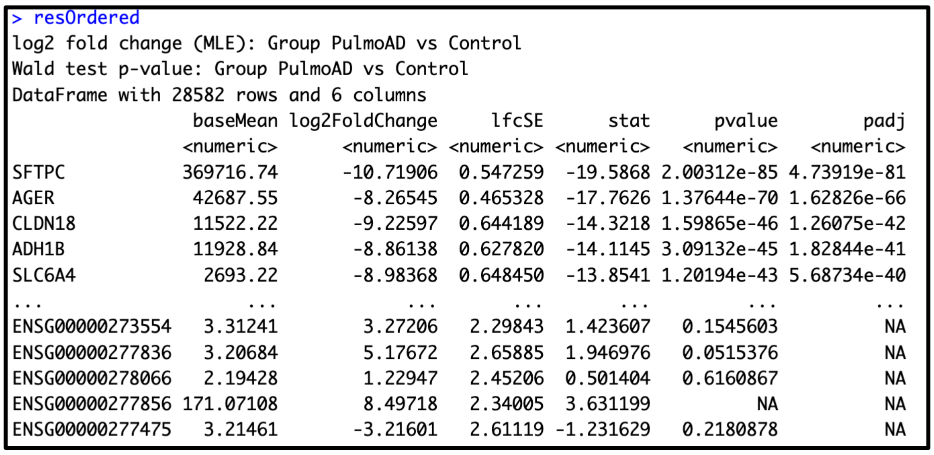
```
ensembl_ids <- gsub("\\..*", "", rownames(resOrdered))
```

Map Ensembl to Gene Symbols:

```
gene_symbols <- mapIds(org.Hs.eg.db,
                       keys = ensembl_ids,
                       column = "SYMBOL",
                       keytype = "ENSEMBL",
                       multiVals = "first")
```

Replace IDs:

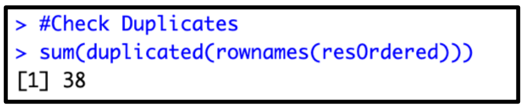
```
rownames(resOrdered) <- ifelse(!is.na(gene_symbols) &
gene_symbols != "", gene_symbols, ensembl_ids)

resOrdered
```



```
> resOrdered
log2 fold change (MLE): Group PulmoAD vs Control
Wald test p-value: Group PulmoAD vs Control
DataFrame with 28582 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat      pvalue        padj
                <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
SFTPC           369716.74      -10.71906  0.547259  -19.5868 2.00312e-85 4.73919e-81
AGER             42687.55       -8.26545  0.465328  -17.7626 1.37644e-70 1.62826e-66
CLDN18           11522.22       -9.22597  0.644189  -14.3218 1.59865e-46 1.26075e-42
ADH1B            11928.84       -8.86138  0.627820  -14.1145 3.09132e-45 1.82844e-41
SLC6A4            2693.22       -8.98368  0.648450  -13.8541 1.20194e-43 5.68734e-40
...                   ...            ...       ...       ...         ...         ...
ENSG00000273554   3.31241        3.27206   2.29843  1.423607   0.1545603          NA
ENSG00000277836   3.20684        5.17672   2.65885  1.946976   0.0515376          NA
ENSG00000278066   2.19428        1.22947   2.45206  0.501404   0.6160867          NA
ENSG00000277856 171.07108        8.49718   2.34005  3.631199          NA          NA
ENSG00000277475   3.21461       -3.21601   2.61119 -1.231629   0.2180878          NA
```

Check Duplicate Gene Names (if any):

```
sum(duplicated(rownames(resOrdered)))
```



```
> #Check Duplicates
> sum(duplicated(rownames(resOrdered)))
[1] 38
```

Remove duplicates (or make unique – be careful):

```
resOrdered <- resOrdered[!duplicated(rownames(resOrdered)), ]
```

Save results (unfiltered):

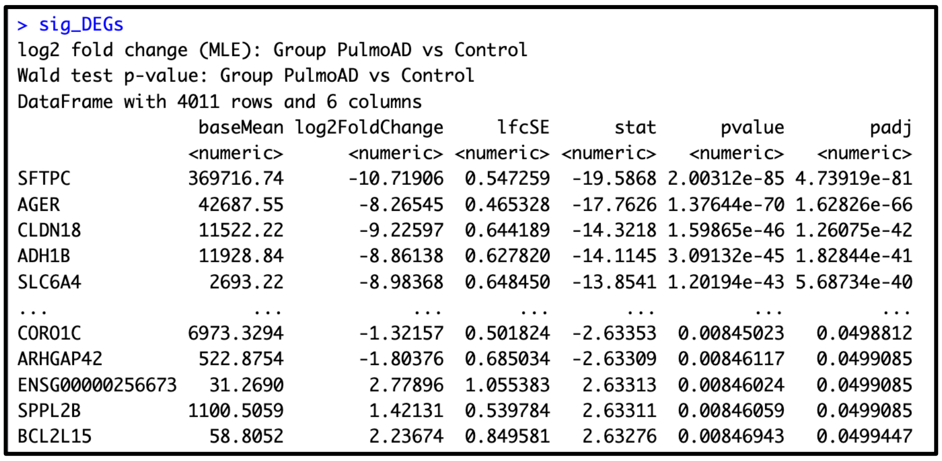
```
write.csv(as.data.frame(resOrdered), "DESeq2_results_all.csv")
```

**Step 12: Identify Significant DEGs**

Selects significantly differentially expressed genes using statistical and biological cutoffs:

- Adjusted p-value < 0.05 (statistical confidence)
- |log2FoldChange| > 1 (≥2-fold change)

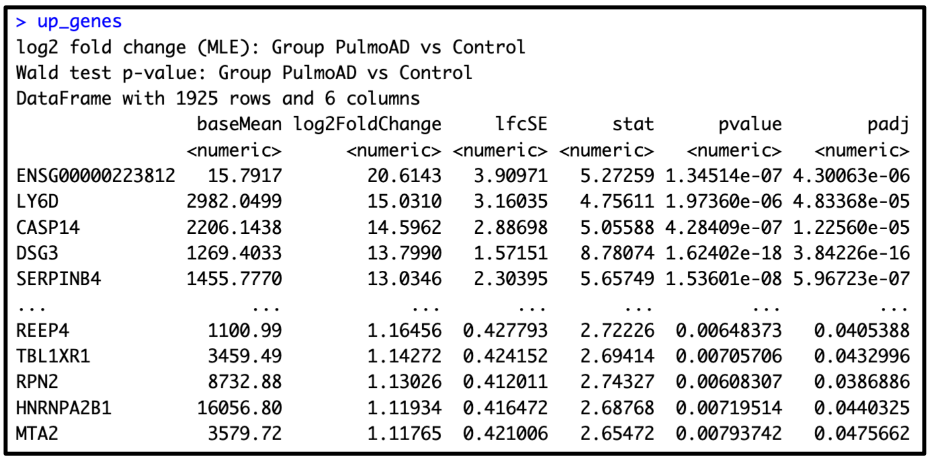
```
sig_DEGs <- resOrdered[
  !is.na(resOrdered$padj) &
    resOrdered$padj < 0.05 &
    abs(resOrdered$log2FoldChange) > 1, ]
```



```
> sig_DEGs
log2 fold change (MLE): Group PulmoAD vs Control
Wald test p-value: Group PulmoAD vs Control
DataFrame with 4011 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat      pvalue        padj
                <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
SFTPC           369716.74      -10.71906  0.547259  -19.5868 2.00312e-85 4.73919e-81
AGER             42687.55       -8.26545  0.465328  -17.7626 1.37644e-70 1.62826e-66
CLDN18           11522.22       -9.22597  0.644189  -14.3218 1.59865e-46 1.26075e-42
ADH1B            11928.84       -8.86138  0.627820  -14.1145 3.09132e-45 1.82844e-41
SLC6A4            2693.22       -8.98368  0.648450  -13.8541 1.20194e-43 5.68734e-40
...                   ...            ...       ...       ...         ...         ...
CORO1C          6973.3294       -1.32157  0.501824  -2.63353  0.00845023   0.0498812
ARHGAP42         522.8754       -1.80376  0.685034  -2.63309  0.00846117   0.0499085
ENSG00000256673   31.2690        2.77896  1.055383   2.63313  0.00846024   0.0499085
SPPL2B          1100.5059        1.42131  0.539784   2.63311  0.00846059   0.0499085
BCL2L15           58.8052        2.23674  0.849581   2.63276  0.00846943   0.0499447
```

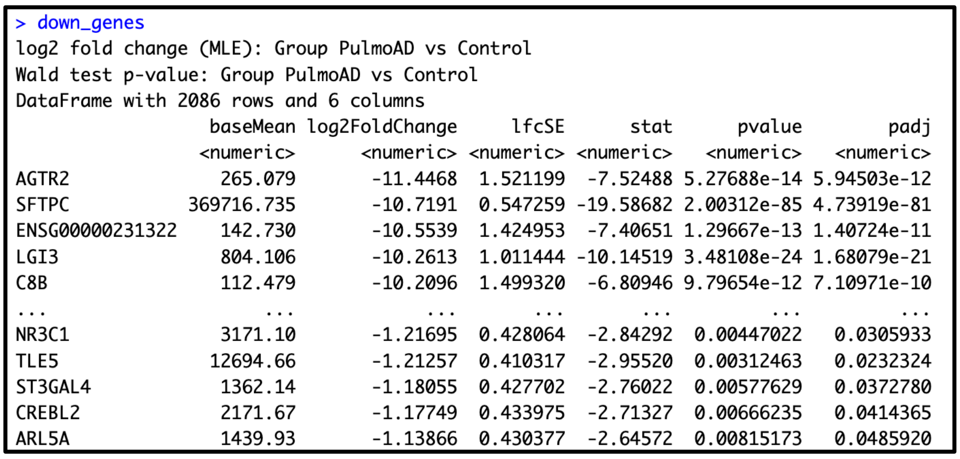
Extract Significantly Upregulated and Downregulated genes:

```
up_genes <- sig_DEGs[sig_DEGs$log2FoldChange > 1, ]
up_genes <- up_genes[order(up_genes$log2FoldChange, decreasing
= TRUE), ]
up_genes
```



```
> up_genes
log2 fold change (MLE): Group PulmoAD vs Control
Wald test p-value: Group PulmoAD vs Control
DataFrame with 1925 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat      pvalue        padj
                <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
ENSG00000223812   15.7917        20.6143   3.90971   5.27259 1.34514e-07 4.30063e-06
LY6D            2982.0499        15.0310   3.16035   4.75611 1.97360e-06 4.83368e-05
CASP14          2206.1438        14.5962   2.88698   5.05588 4.28409e-07 1.22560e-05
DSG3            1269.4033        13.7990   1.57151   8.78074 1.62402e-18 3.84226e-16
SERPINB4        1455.7770        13.0346   2.30395   5.65749 1.53601e-08 5.96723e-07
...                   ...            ...       ...       ...         ...         ...
REEP4             1100.99        1.16456  0.427793   2.72226  0.00648373   0.0405388
TBL1XR1           3459.49        1.14272  0.424152   2.69414  0.00705706   0.0432996
RPN2              8732.88        1.13026  0.412011   2.74327  0.00608307   0.0386886
HNRNPA2B1        16056.80        1.11934  0.416472   2.68768  0.00719514   0.0440325
MTA2              3579.72        1.11765  0.421006   2.65472  0.00793742   0.0475662
```

```
down_genes <- sig_DEGs[sig_DEGs$log2FoldChange < -1, ]
down_genes <- down_genes[order(down_genes$log2FoldChange,
decreasing = FALSE), ]
down_genes
```



```
> down_genes
log2 fold change (MLE): Group PulmoAD vs Control
Wald test p-value: Group PulmoAD vs Control
DataFrame with 2086 rows and 6 columns
                  baseMean log2FoldChange     lfcSE      stat      pvalue        padj
                 <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
AGTR2              265.079       -11.4468  1.521199  -7.52488 5.27688e-14 5.94503e-12
SFTPC           369716.735       -10.7191  0.547259 -19.58682 2.00312e-85 4.73919e-81
ENSG00000231322    142.730       -10.5539  1.424953  -7.40651 1.29667e-13 1.40724e-11
LGI3               804.106       -10.2613  1.011444 -10.14519 3.48108e-24 1.68079e-21
C8B                112.479       -10.2096  1.499320  -6.80946 9.79654e-12 7.10971e-10
...                    ...            ...       ...       ...         ...         ...
NR3C1              3171.10       -1.21695  0.428064  -2.84292  0.00447022   0.0305933
TLE5              12694.66       -1.21257  0.410317  -2.95520  0.00312463   0.0232324
ST3GAL4            1362.14       -1.18055  0.427702  -2.76022  0.00577629   0.0372780
CREBL2             2171.67       -1.17749  0.433975  -2.71327  0.00666235   0.0414365
ARL5A              1439.93       -1.13866  0.430377  -2.64572  0.00815173   0.0485920
```

Save Results:

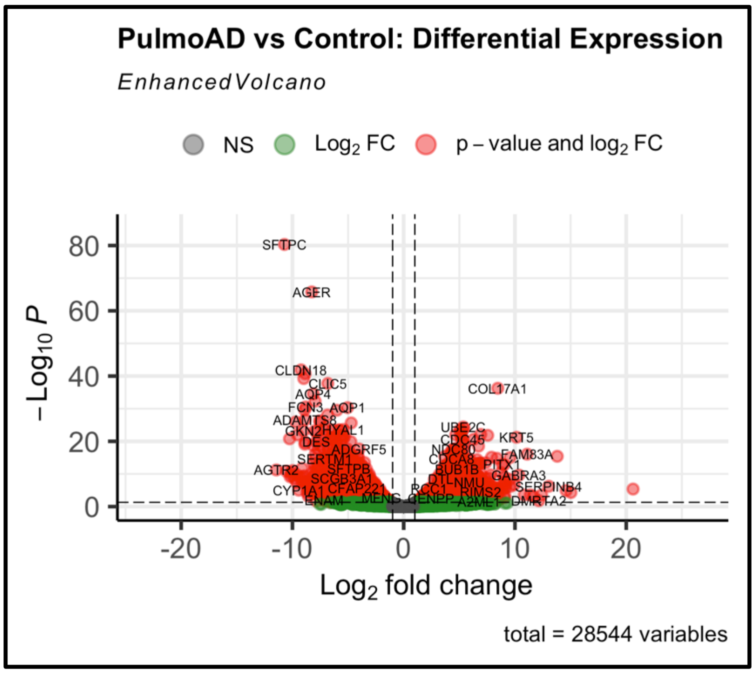
```
write.csv(as.data.frame(up_genes), "PulmoAD_Sig_DEGs_up.csv")
write.csv(as.data.frame(down_genes),
"PulmoAD_Sig_DEGs_down.csv")
```

**Step 13: Visualize with Volcano Plot**

A volcano plot combines fold-change and statistical significance in one visual summary.

- X-axis - log2 fold change (magnitude of expression difference)
- Y-axis - -log10(adjusted p-value) (significance)
- Red points - significant up/down-regulated genes
- Grey and Green - non-significant genes

```
EnhancedVolcano(resOrdered,
                lab = rownames(resOrdered),
                x = 'log2FoldChange',
                y = 'padj',
                title = 'PulmoAD vs Control: Differential
Expression',
                pCutoff = 0.05,
                FCcutoff = 1,
                pointSize = 2.5,
                labSize = 3)
```



**Note**: You can customize the volcano plot using ggplot and other visualization packages.

**Important Points:**

- The metadata should be changed according to your study design. And the design and extraction should be according to your study.
- The number of biological replicates required for DESeq2 is at least three per condition. If fewer than three replicates are available, DESeq2 can still run, but the statistical power will be very limited, and dispersion estimates will be unreliable, leading to a high false positive or false negative rate. If only two biological replicates are available per condition, DESeq2 can still be used, but the analysis will have low statistical power and reduced reliability in detecting true differential expression.
- Recommendations: Use DESeq2 cautiously. Interpret significant genes as *putative* rather than *definitive*. If possible, increase to ≥3 biological replicates per condition for statistically robust results. Alternatively, you can use

independent filtering or shrinkage methods (e.g., lfcShrink()) to improve stability of results when replicates are limited. Validate findings with methods like qPCR.