# Differential Gene Expression Analysis with DESeq2

Zhiping Zhang

2025-04-20

## Introduction

This R Markdown document performs differential gene expression analysis using long-read RNA-seq data processed with featureCounts on Xanadu (alternatively `Rsubread::featureCounts`). The data consists of read counts from six samples (three wild-type day 0 (WT_D0) and three wild-type day 7 (WT_D7)) aligned to the human genome (hg38, chromosome 21). The analysis uses `DESeq2` to identify differentially expressed genes between WT_D0 and WT_D7 conditions, visualizes the results with an MA plot, and examines count data for genes of interest (CBS, ITSN1, PAXBP1, TRAPPC10).

## Notes

- **File Path**: The featureCounts file path (`./data/`) must be accessible. Upload it if you have the files in a local directory.
- **Gene IDs**: Ensure `goi` (CBS, ITSN1, PAXBP1, TRAPPC10) match the `Geneid` in `fc_file`. We use gene names instead.
- **Assignment**: Try to use Rsubread::featureCounts to process the sample .bam files in R and finish the differential gene expression analysis.
- **Further exploration**: featureCounts is fast but may lack precision for long-read RNA-seq data. For improved gene assignment, explore IsoQuant (https://github.com/ablab/IsoQuant) or Bambu (https://github.com/GoekeLab/bambu).

## Setup

Load required R packages: `data.table` for efficient data reading, and `DESeq2` for differential expression analysis.

```r
if (!require("BiocManager", quietly = TRUE))
     install.packages("BiocManager")
BiocManager::install("data.table",force = FALSE)
```

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

```r
BiocManager::install("DESeq2",force = FALSE)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   `force = TRUE` to re-install: 'DESeq2'
```

```r
library(data.table)
library(DESeq2)
```

## Load and Process Count Data

Read the featureCounts output file, extract the count matrix (columns 7 to 12), and set row names as gene IDs. Column names are cleaned to remove the directory path prefix.

```r
fc_file <- fread("./data/hg38_chr21_quant_name")
fc_counts <- as.matrix(fc_file[, 7:12])
rownames(fc_counts) <- as.factor(fc_file$Geneid)
colnames(fc_counts) <- sub(".*/minimap2_bam/", "", colnames(fc_counts))
```

## Build Sample Table

Create a sample table for DESeq2, specifying sample names, file names, and conditions (WT_D0 or WT_D7). Each condition has three replicates.

```r
sampleFiles <- colnames(fc_counts)
sampleCondition <- factor(rep(c("WT_D0", "WT_D7"), each = 3))
sampleTable <- data.frame(
  sampleName = sub("\\.chr21.bam$", "", sampleFiles),
  fileName = sampleFiles,
  condition = sampleCondition
)
```

## Create DESeq2 Dataset

Construct a DESeqDataSet from the count matrix, sample table, and experimental design (~ condition).

```r
dds <- DESeqDataSetFromMatrix(
  countData = fc_counts,
  colData = sampleTable,
  design = ~ condition
)
```

## Data Pre-filtering

Filter out genes with low counts to reduce noise. Keep genes with at least 10 counts in at least three samples (smallest group size).

```r
smallestGroupSize <- 3
keep <- rowSums(counts(dds) >= 10) >= smallestGroupSize
dds <- dds[keep,]
```

## Differential Expression Analysis

Run the DESeq2 pipeline to estimate size factors, dispersions, and perform differential expression analysis. Extract results comparing WT_D7 vs. WT_D0.

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

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

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

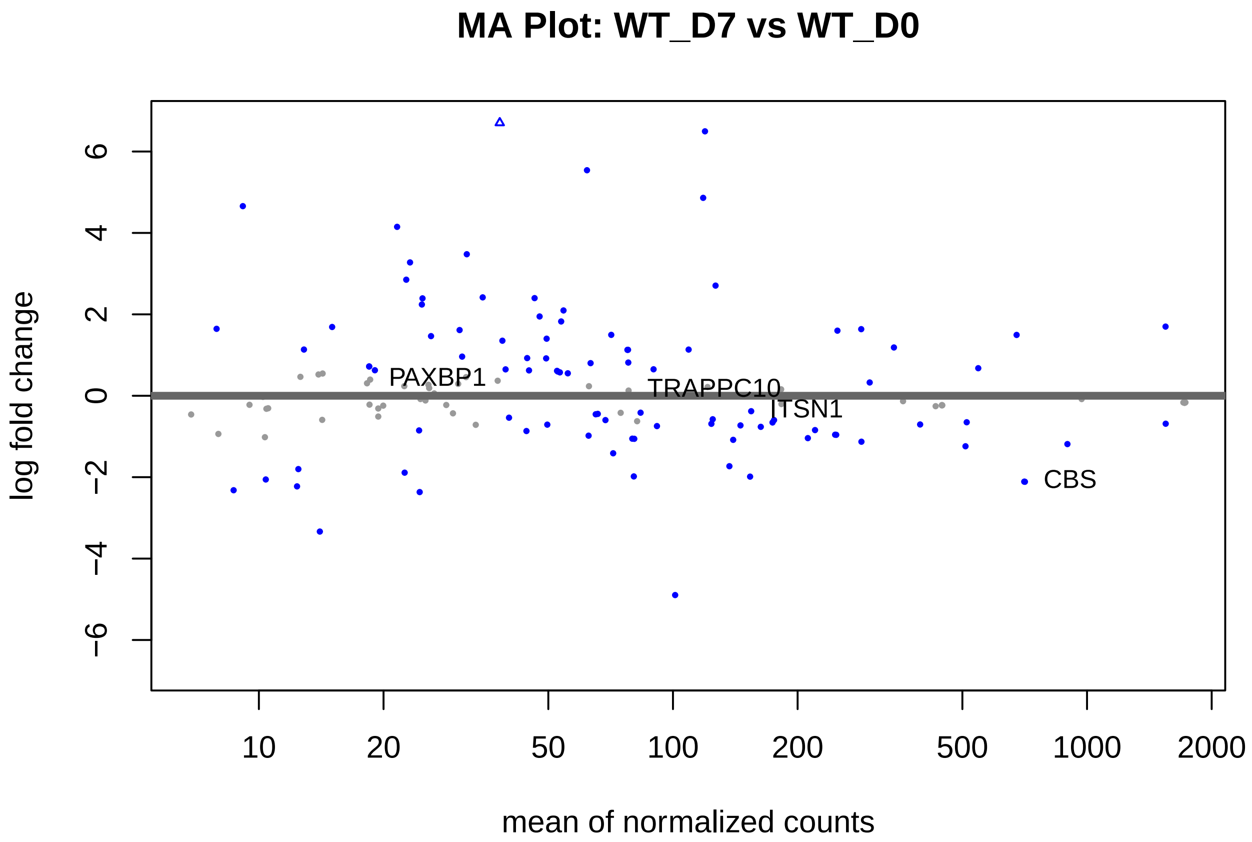
```
##
## out of 143 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 50, 35%
## LFC < 0 (down)     : 44, 31%
## outliers [1]       : 0, 0%
## low counts [2]     : 0, 0%
## (mean count < 7)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

## Visualize Results

### MA Plot

Create an MA plot to visualize log2 fold changes against mean expression levels. Highlight genes of interest (CBS, ITSN1, PAXBP1, TRAPPC10) with labels.

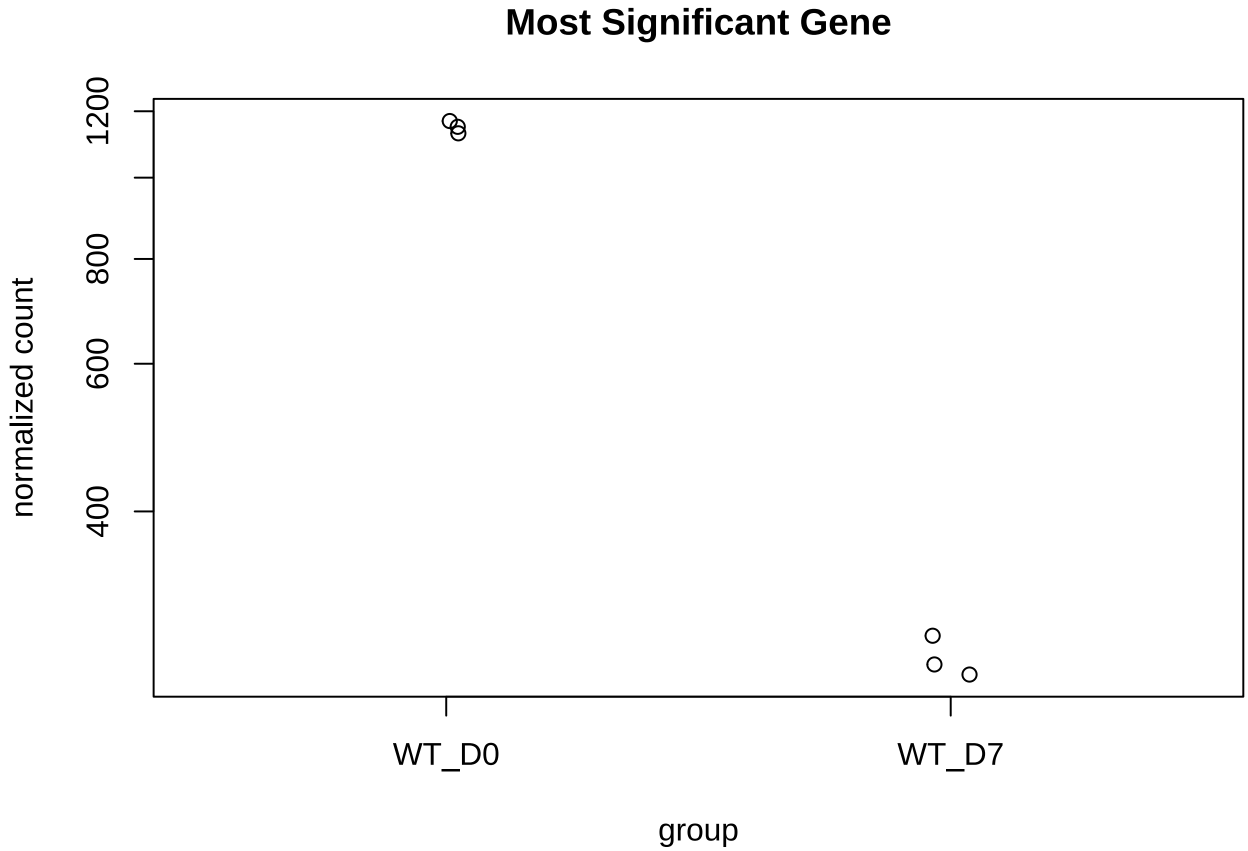
```
plotMA(res, main = "MA Plot: WT_D7 vs WT_D0")
goi <- c("CBS", "ITSN1", "PAXBP1", "TRAPPC10")
table <- as.data.frame(res)[goi, ]
text(table$baseMean, table$log2FoldChange,
     labels = goi,
     pos = 4,
     cex = 0.8)
```



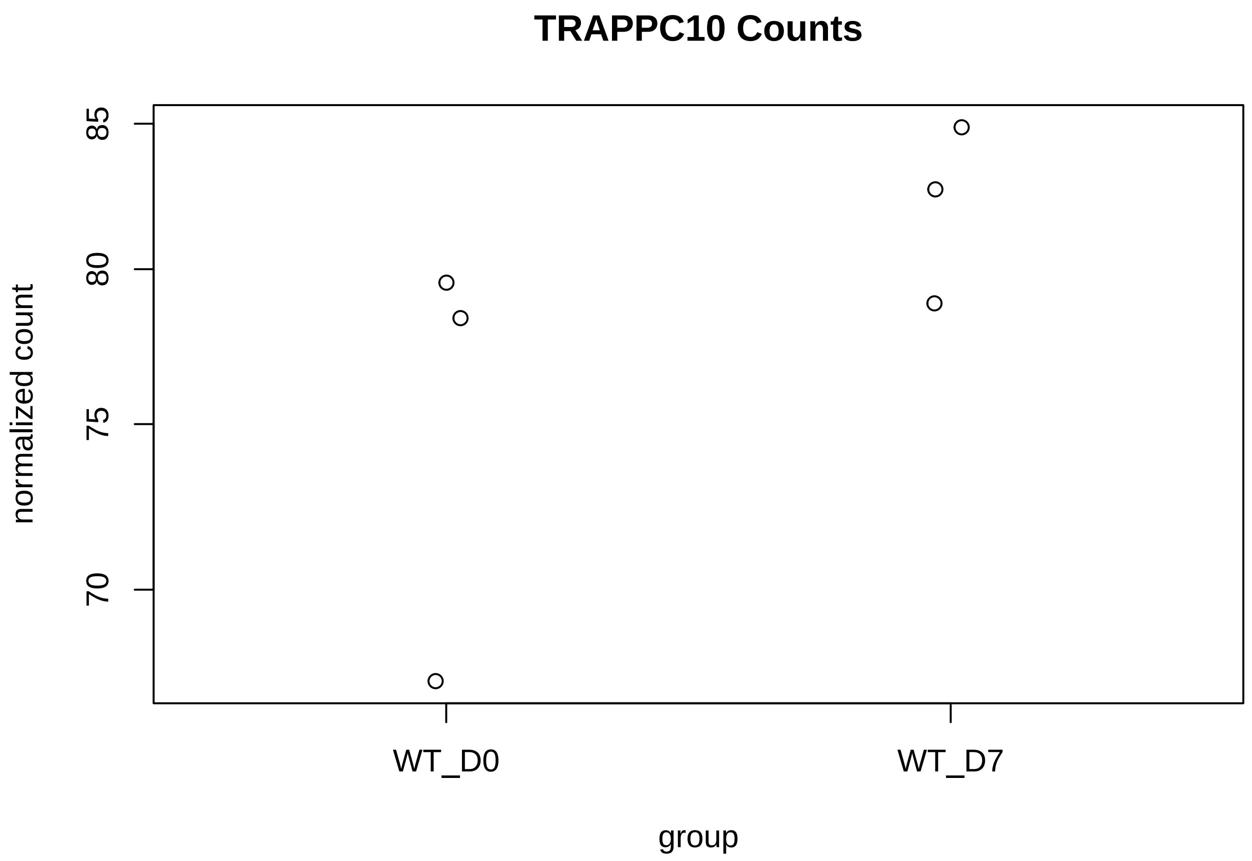
### Count Plots for Genes of Interest

Plot normalized counts for the gene with the smallest adjusted p-value and the specified genes of interest (TRAPPC10, ITSN1, PAXBP1) across conditions.

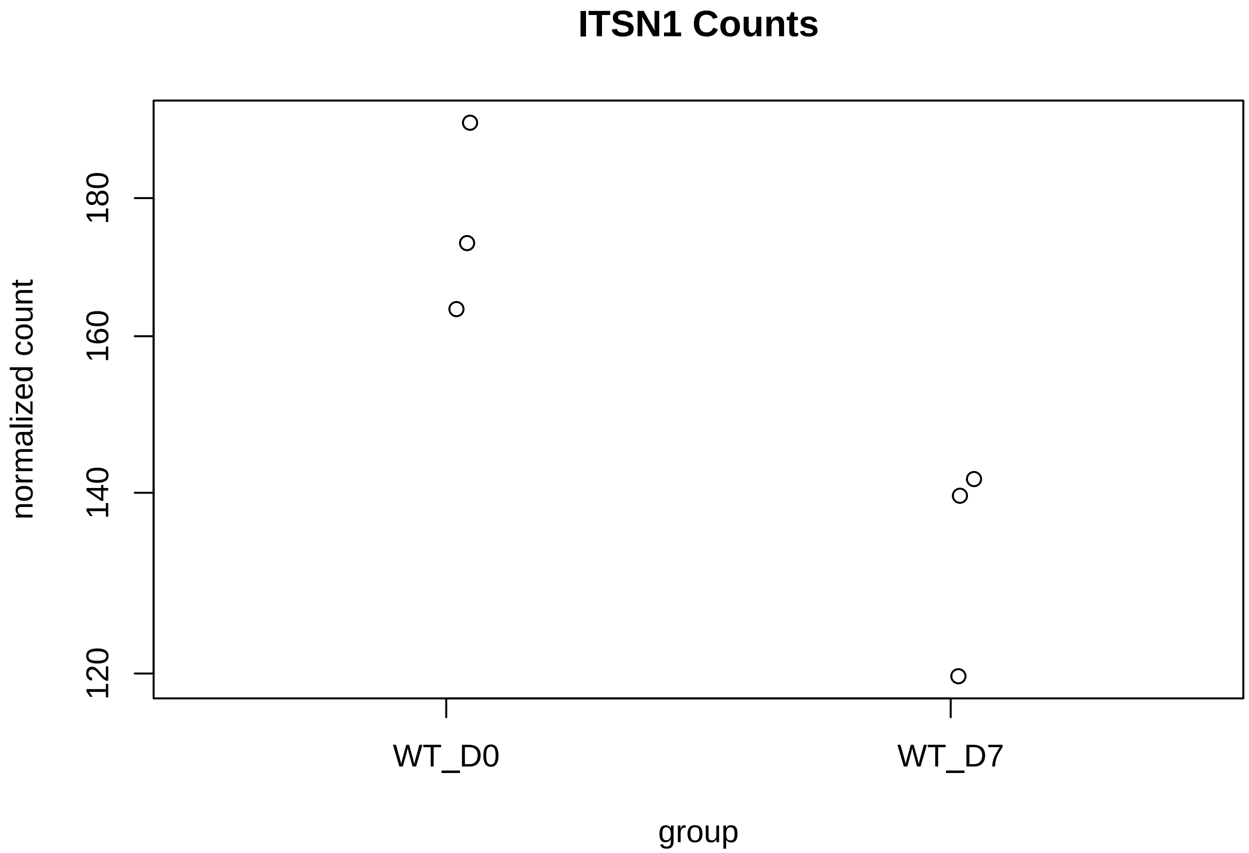
```r
plotCounts(dds, gene = which.min(res$padj), intgroup = "condition", main = "Most Significant Gene")
```



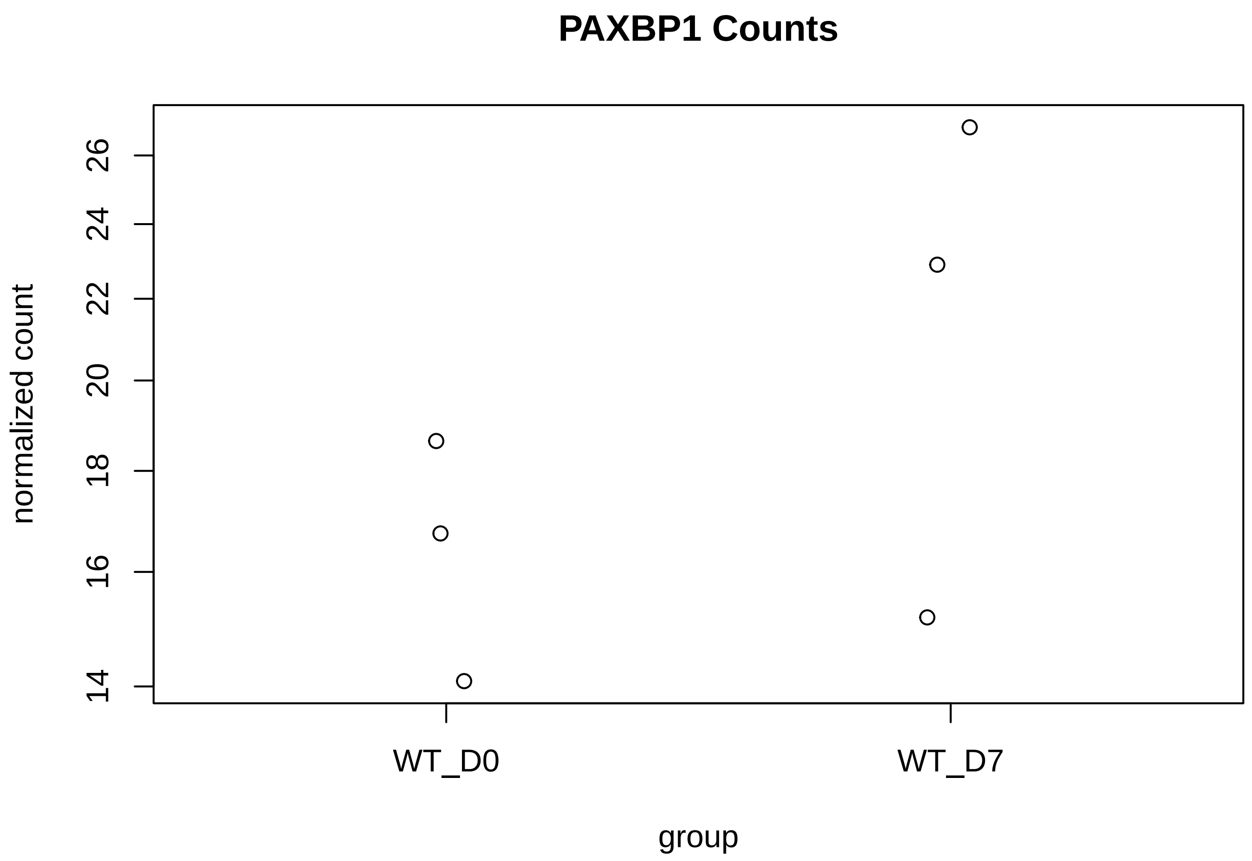
```r
plotCounts(dds, gene = "TRAPPC10", intgroup = "condition", main = "TRAPPC10 Counts")
```



```
plotCounts(dds, gene = "ITSN1", intgroup = "condition", main = "ITSN1 Counts")
```



```r
plotCounts(dds, gene = "PAXBP1", intgroup = "condition", main = "PAXBP1 Counts")
```


PAXBP1 Counts
normalized count
26
24
22
20
18
16
14
WT_D0
WT_D7
group