Transcriptome Analysis of Mycobacterium leprae

Isabella Fregoso

2025-05-02

Load required packages

```
library(ballgown)
library(RColorBrewer)
library(genefilter)
library(dplyr)
library(devtools)
```

This assigning the data to a frame or table in order to organize the results. This allows us to analyze the sample IDs and phenotypic data.

```
pheno_data<-data.frame(ids = c("old_01", "old_02", "young_01", "young_02"),
                       stage = c("old", "old", "young", "young"))
```

Create Ballgown object and check transcript number

```
samples.c <- paste('ballgown', pheno_data$ids, sep = '/')
bg <- ballgown(samples = samples.c, meas='all', pData = pheno_data)
bg
```

```
## ballgown instance with 3120 transcripts and 4 samples
```

This is creating a new filtered version of the ballgown object. This filtering reduces extra unnecessary information to be more accurate.

```
bg_filt = subset(bg,"rowVars(texpr(bg)) >1",genomesubset=TRUE)
bg_filt
```

```
## ballgown instance with 3008 transcripts and 4 samples
```

Table of transcripts

```
results_transcripts<- stattest(bg_filt, feature = "transcript", covariate = "stage",
getFC = TRUE, meas = "FPKM")
results_transcripts<-data.frame(geneNames=geneNames(bg_filt),
transcriptNames=transcriptNames(bg_filt), results_transcripts)
```

Examining transcript more closely.

```
results_transcripts[results_transcripts$transcriptNames == "rna-DIJ64_RS00045", ]
```

```
##   geneNames   transcriptNames    feature id        fc      pval      qval
## 9         . rna-DIJ64_RS00045 transcript  9 0.4893287 0.3644566 0.8897224
```

We are given the ID number, the fold change value, the p value, and the q value.

This code is searching for significant rows from the p value that was given, and shows how many significant columns and row there are.

```
sigdiff <- results_transcripts %>% filter(pval<0.05)
dim(sigdiff)
```

```
## [1] 173   7
```

The table below is organized by ascending p value.

```
o = order(sigdiff[,"pval"], -abs(sigdiff[,"fc"]), decreasing=FALSE)
output = sigdiff[o,c("geneNames","transcriptNames", "id","fc","pval","qval")]
write.table(output, file="SigDiff.txt", sep="\t", row.names=FALSE, quote=FALSE)
head(output)
```

```
##      geneNames    transcriptNames   id        fc         pval      qval
## 1840         . gene-DIJ64_RS18560 1840 0.6384140 0.0003046501 0.3702543
## 1343         . gene-DIJ64_RS21400 1343 1.4084948 0.0003818719 0.3702543
## 887          . gene-DIJ64_RS04330  887 0.9680325 0.0004785278 0.3702543
## 929          . gene-DIJ64_RS21230  929 0.9472872 0.0004923594 0.3702543
## 249          . gene-DIJ64_RS01285  249 0.6926288 0.0006491295 0.3905163
## 289          . gene-DIJ64_RS01500  289 1.1379513 0.0009113978 0.4000533
```

Load gene names

```
bg_table = texpr(bg_filt, 'all')
bg_gene_names = unique(bg_table[, 9:10])
```

Pull out gene expression data and visualize

```
gene_expression = as.data.frame(gexpr(bg_filt))
head(gene_expression)
```

```
##            FPKM.old_01 FPKM.old_02 FPKM.young_01 FPKM.young_02
## .              9.39228    4.162646      14.16595       0.00000
## MSTRG.1       99.48248  152.702316     124.94012     179.55749
## MSTRG.10     125.55261  122.107544     163.84876     159.45210
## MSTRG.100     59.18023   67.009201      90.70001      64.79395
## MSTRG.1000   356.89404  367.967255     425.27213     428.28647
## MSTRG.1002   151.47260  171.148743     217.57909     161.35928
```

This code is changing the name of the columns in a more organized fashion.

```
colnames(gene_expression) <- c("old_01", "old_02", "young_01", "young_02")
head(gene_expression)
```

```
##               old_01     old_02  young_01  young_02
## .            9.39228   4.162646  14.16595   0.00000
## MSTRG.1     99.48248 152.702316 124.94012 179.55749
## MSTRG.10   125.55261 122.107544 163.84876 159.45210
## MSTRG.100   59.18023  67.009201  90.70001  64.79395
## MSTRG.1000 356.89404 367.967255 425.27213 428.28647
## MSTRG.1002 151.47260 171.148743 217.57909 161.35928
```

```
dim(gene_expression)
```

```
## [1] 2693    4
```

Load the transcript to gene table and determine the number of transcripts and unique genes

```
transcript_gene_table = indexes(bg)$t2g
head(transcript_gene_table)
```

```
##   t_id    g_id
## 1    1 MSTRG.1
## 2    2 MSTRG.2
## 3    3 MSTRG.3
## 4    4 MSTRG.3
## 5    5 MSTRG.4
## 6    6 MSTRG.5
```

```
length(row.names(transcript_gene_table))
```

```
## [1] 3120
```

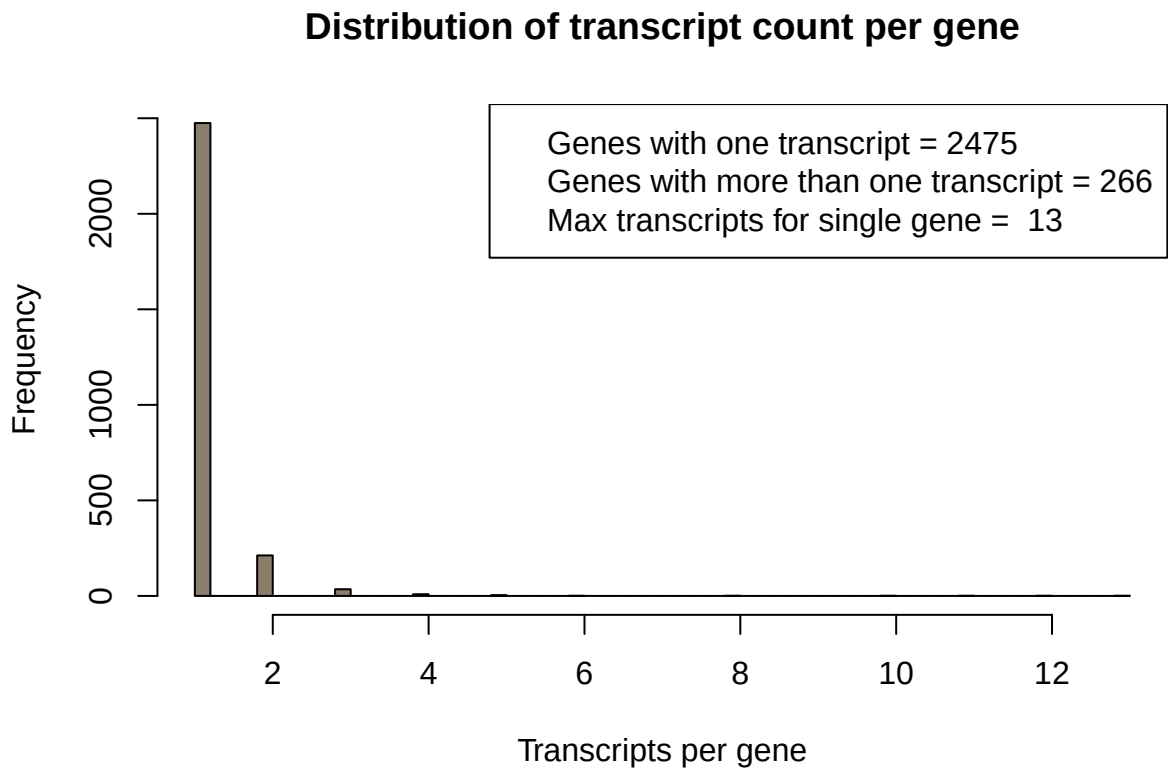
```
length(unique(transcript_gene_table[,"g_id"]))
```

```
## [1] 2741
```

Plot the number of transcripts per gene

```
counts=table(transcript_gene_table[,"g_id"])
c_one = length(which(counts == 1))
```

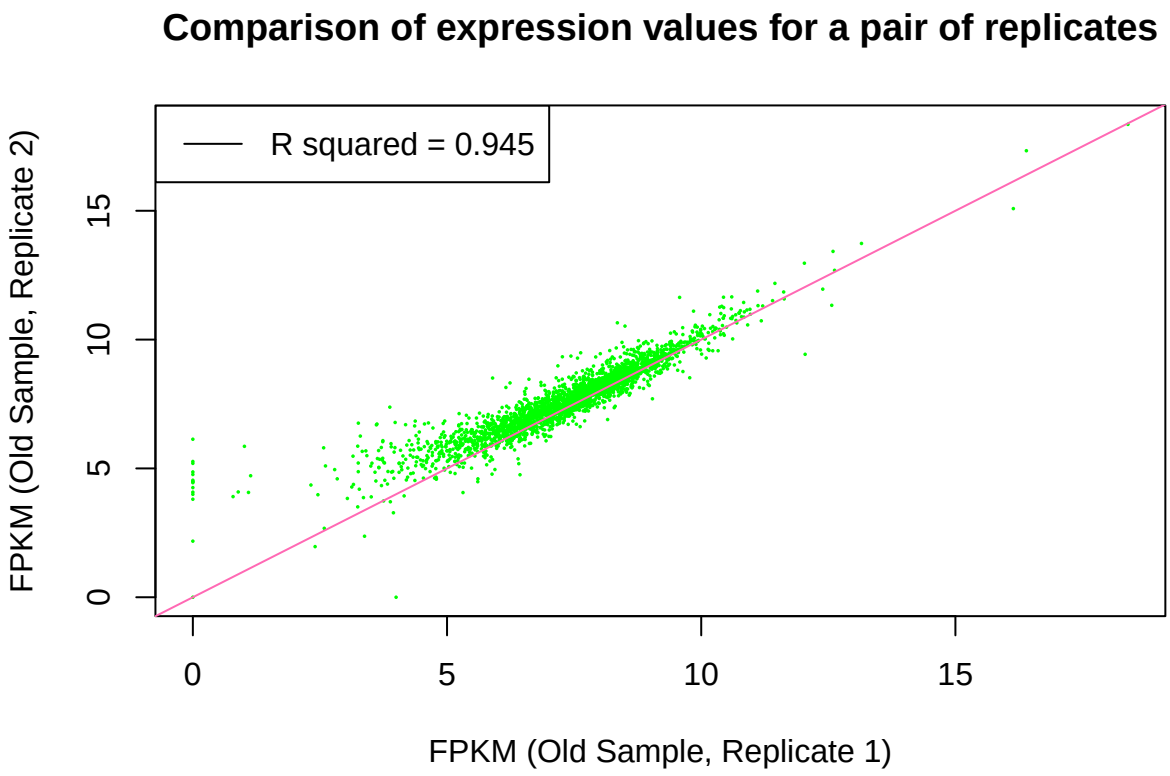
```
c_more_than_one = length(which(counts > 1))
c_max = max(counts)
hist(counts, breaks=50, col="bisque4", xlab="Transcripts per gene",
main="Distribution of transcript count per gene")
legend_text = c(paste("Genes with one transcript =", c_one),
paste("Genes with more than one transcript =", c_more_than_one),
paste("Max transcripts for single gene = ", c_max))
legend("topright", legend_text, lty=NULL)
```



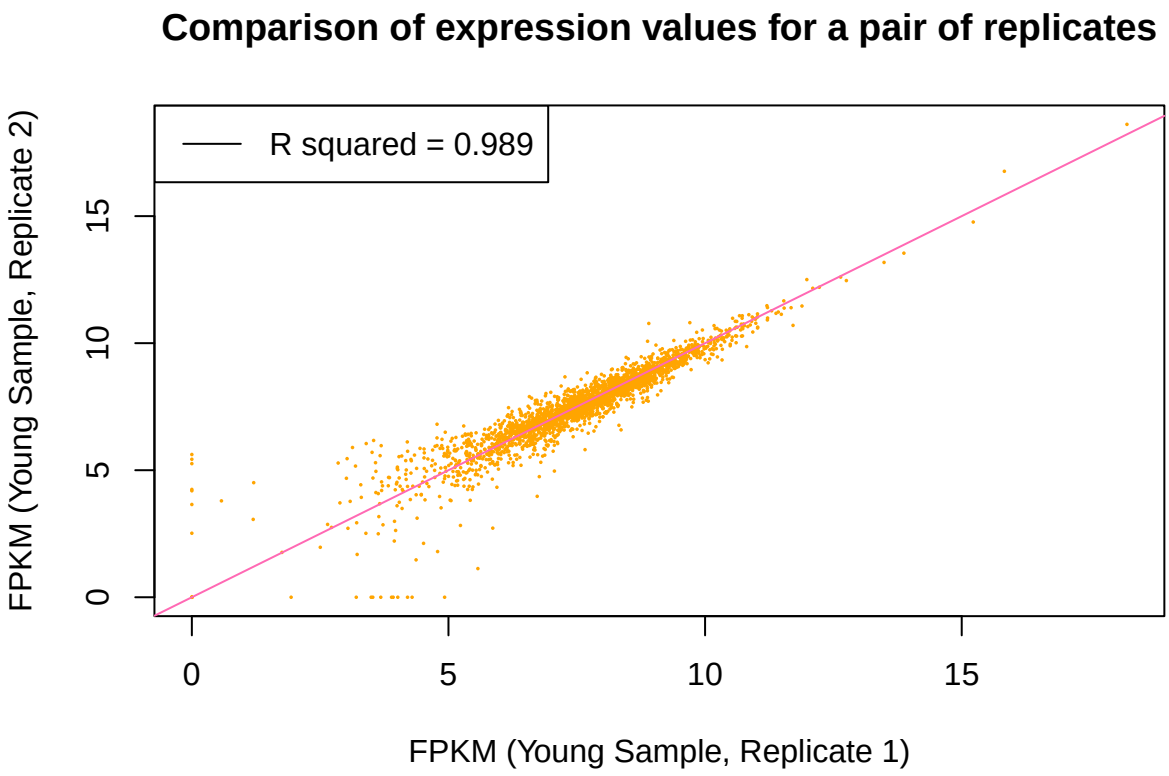
The graph above shows 2475 genes with one transcript, 266 genes with more than one, and 13 as the maximum transcripts for a single gene. There is a high frequency peak at almost 0 transcripts per gene.

Create a plot of how similar the two replicates are for one another

```
x = gene_expression[,"old_01"]
y = gene_expression[,"old_02"]
min_nonzero=1
plot(x=log2(x+min_nonzero), y=log2(y+min_nonzero), pch=16, col="green", cex=0.25,
xlab="FPKM (Old Sample, Replicate 1)", ylab="FPKM (Old Sample, Replicate 2)",
main="Comparison of expression values for a pair of replicates")
abline(a=0,b=1, col = "hotpink")
rs=cor(x,y)^2
legend("topleft", paste("R squared = ", round(rs, digits=3), sep=""), lwd=1, col="black")
```



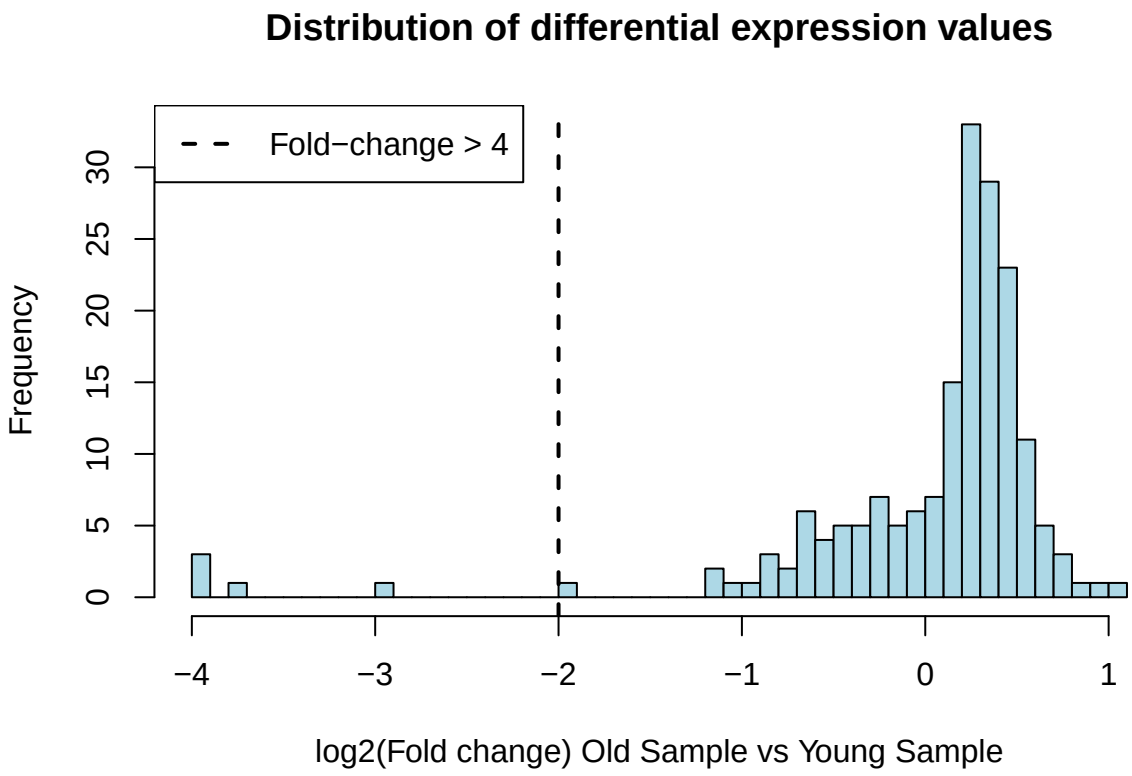

```
x = gene_expression[,"young_01"]
y = gene_expression[,"young_02"]
min_nonzero=1
plot(x=log2(x+min_nonzero), y=log2(y+min_nonzero), pch=16, col="orange", cex=0.25,
xlab="FPKM (Young Sample, Replicate 1)", ylab="FPKM (Young Sample, Replicate 2)",
main="Comparison of expression values for a pair of replicates")
abline(a=0,b=1, col = "hotpink")
rs=cor(x,y)^2
legend("topleft", paste("R squared = ", round(rs, digits=3), sep=""), lwd=1, col="black")
```



If the data sets are similar, that means the genome has conserved sequences that are shown in the alignment.

Create plot of differential gene expression between the conditions

```
results_genes = stattest(bg_filt, feature="gene", covariate="stage", getFC=TRUE, meas="FPKM")
results_genes = merge(results_genes,bg_gene_names,by.x=c("id"),by.y=c("gene_id"))
sig=which(results_genes$pval<0.05)
results_genes[,"de"] = log2(results_genes[,"fc"])
hist(results_genes[sig,"de"], breaks=50, col="lightblue",
xlab="log2(Fold change) Old Sample vs Young Sample",
main="Distribution of differential expression values")
abline(v=-2, col="black", lwd=2, lty=2)
abline(v=2, col="black", lwd=2, lty=2)
legend("topleft", "Fold-change > 4", lwd=2, lty=2)
```



The graph above is showing the frequency compared to the log2 of the old vs young samples. We can see a fold change of less than 4 on the x axis of about 0.5 There is slight left distribution here.

Plot total gene expression highlighting differentially expressed genes

```
gene_expression[,"old"]=apply(gene_expression[,c(1:2)], 1, mean)
gene_expression[,"young"]=apply(gene_expression[,c(3:4)], 1, mean)
x=log2(gene_expression[,"old"]+min_nonzero)
y=log2(gene_expression[,"young"]+min_nonzero)
plot(x=x, y=y, pch=16, cex=0.25, xlab="Old FPKM (log2)", ylab="Young FPKM (log2)",
main="Old vs Young FPKMs")
abline(a=0, b=1)
xsig=x[sig]
ysig=y[sig]
points(x=xsig, y=ysig, col="magenta", pch=16, cex=0.5)
legend("topleft", "Significant", col="magenta", pch=16)
```

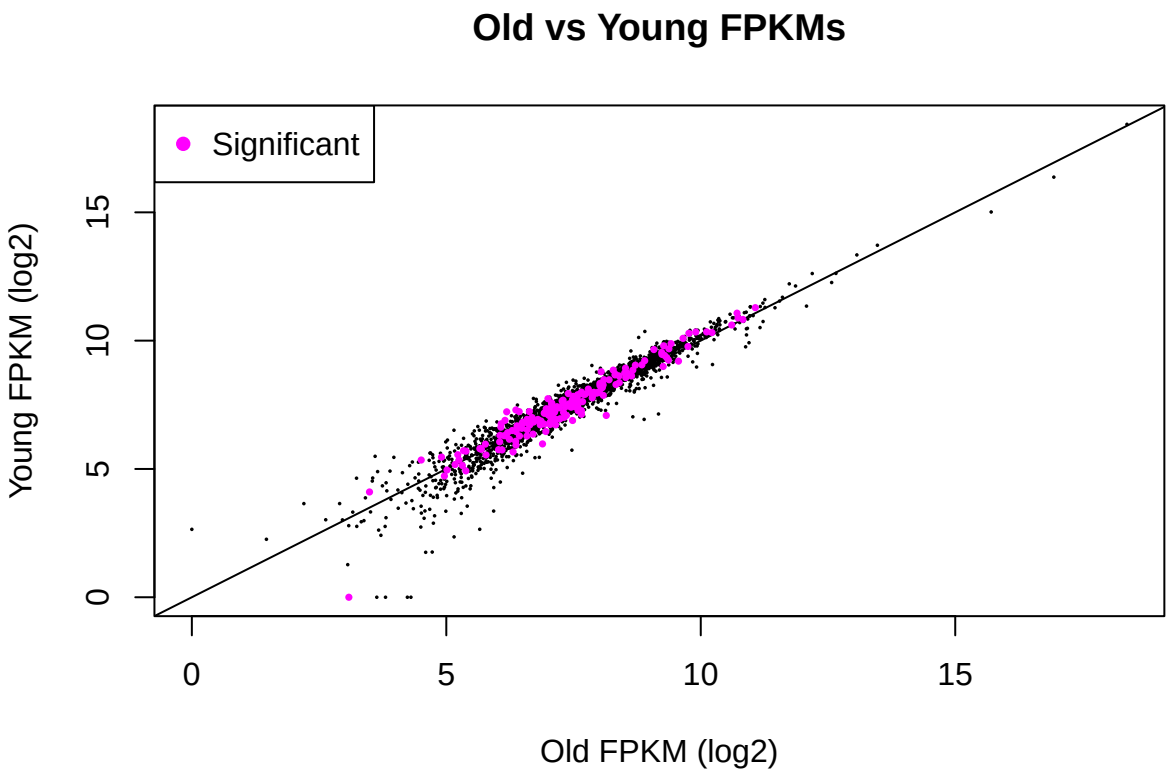


Table of FPKM values

```
fpkm = texpr(bg_filt,meas="FPKM")
```

Choose a gene to determine individual expression

```
ballgown::transcriptNames(bg_filt)[3]
```

```
##                     3
## "gene-DIJ64_RS00015"
```

```
ballgown::geneNames(bg_filt)[3]
```

```
##     3
## "recF"
```

Transform to log2

```
transformed_fpkm <- log2(fpkm[3, ] + 1)
```

Make sure values are properly coded as numbers

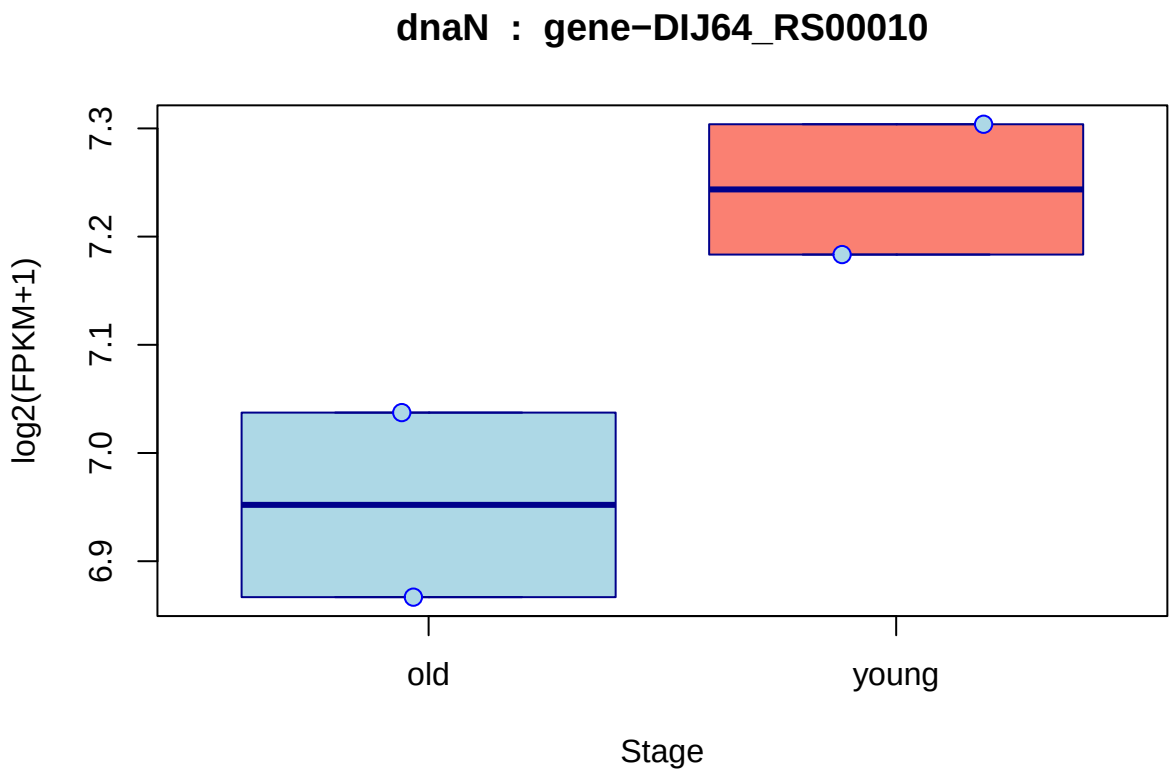
```
numeric_stages <- as.numeric(factor(pheno_data$stage))

jittered_stages <- jitter(numeric_stages)
```

Plot expression of individual gene

```
boxplot(transformed_fpkm ~ pheno_data$stage,
        main=paste(ballgown::geneNames(bg_filt)[2], ' : ', ballgown::transcriptNames(bg_filt)[2]),
        xlab="Stage",
        ylab="log2(FPKM+1)",
        col=c("lightblue", "salmon"),
        border="darkblue")

points(transformed_fpkm ~ jittered_stages,
       pch=21, col="blue", bg="lightblue", cex=1.2)
```



The figure above shows an increase log2 value for the old sample compared to the young sample value. The young sample boxplot has smaller quartiles for the minimum and maximum values.