# RNAseq Walkthrough part 3/3

C. Ryan Campbell

Duke University

*c.ryan.campbell@duke.edu*

9 Nov 2017

# Overview

# Today's Goals

- Discuss Methods and Results
- Format data for DESeq
- Run DESeq

# Methods

- Should be pretty dry reading
- Read like a lab notebook or recipe
- The reader should be able to replicate your work

# Methods

- If you used a software it should be in the Methods
- Include details of usage (flags, settings, or if you used defaults)
- Be sure to cite the software and list the version

# Methods

quenced on Lane 1, while six multiplexed amplified samples were sequenced on Lane 2. The data set supporting the results of this article are available in the NCBI Short Read Archive repository [accession number SRP049463; http://www.ncbi.nlm.nih.gov/sra/?term=SRP049463].

**Read mapping and transcriptome assembly**

RNA-Seq reads were filtered using the Trimmomatic program [56] and aligned to the *R. norvegicus* genome (Rnor_5.0; downloaded from Ensembl.org on 3/2014) using TopHat2 (v 2.0.5) [3,27,57]. The resulting alignment data from TopHat2 were then fed into Cufflinks (v 2.0.2) to assemble aligned RNA-Seq reads into transcripts for each rat individually. Annotated transcripts were obtained from Ensembl database (Rnor_5.0.75.gtf; downloaded from Ensembl.org on 3/2014). Transcript abundances estimates were measured in FPKM. Cuffdiff was used to determine differential gene expression profiles between amplified and unamplified matched samples for each biological replicate included in our study [58]. HTSeq was used under default parameters to produce gene counts for each of our twelve samples [32]. To estimate differential expression, we fed the matrix of read counts generated by HTSeq into DESeq [33]. Since we were working with biological, and not technical replicates, we reduced the read counts to two conditions for each sample (amplified and raw) and estimated dispersion by ignoring condition and

# Results

- Less dry reading, still a bit boring
- This is JUST for outcomes of the work
- No analysis or interpretation, only what the results were

# Results

- If a step was in your Methods, list the results
- This is obvious sometimes:
- "After running trimmomatic n% of the data was kept"
- And less so others:
- DESeq2 has a lot of output, summarize what is most important

# Results

## Results

We sequenced over 829 million paired-end reads, 2 ×100 base pairs in length (Lane 1 = 368,942,612, Lane 2 = 460,712,194 reads). Quality filtering resulted in the retention of approximately 90% and 86% of reads from Lanes 1 and 2, respectively (Table 1). Approximately 62% of all reads successfully mapped to the *R. norvegicus* genome (Table 1; Additional file 1: Table S1). Transcript assembly using Cufflinks resulted in 25,543 assembled transcripts.

We identified strong correlations in gene expression within each of the six pairwise comparisons of matched raw RNA and amplified RNA, and FPKM (fragments per kilobase of exon per million fragments mapped) values averaged slightly higher in libraries constructed from raw RNA samples (Figure 1). Transcripts with FPKM values greater than 0.05 were identified as being expressed (i.e. above the expected false discovery rate following Trapnell *et al.* 2010), and this threshold was used to determine the number of expressed genes for each sample. Libraries constructed from raw RNA and WTA samples averaged 15,298 and 15,253 expressed genes, respectively (Table 1; Additional file 1: Table S1). Approximately 93% of expressed genes were identical between each of the six matched raw RNA and WTA comparisons (Table 1, Figure 2) and gene density was similar across all samples (Figure 3). Using Cuffdiff, significant differentially expressed genes ($P < 0.05$) were

# SLURM Interactive Node

```
srun -mem-per-cpu=4000MB -pty bash -i
```

- When you're done with an interactive node type `exit`
- Also, check your SLURM queue `squeue -u <netID>`
- If you're still running a "bash" job, use `scancel <job ID number>` to cancel it

# Files to Use

- I have some example files to use for the DESeq tutorial
- They're human RNAseq files from a hypoxia experiment:

```
ls -lthr /work/cc216/490S/cc216/RNAseq_pt3
```

# Combining htseq Output

- There are four samples of htseq-count output:

```
s01.norm.counts, s02.norm.counts, s03.hypo.counts,
s04.hypo.counts
> head s01.norm.counts
3.8-1.4 0
3.8-1.5 0
5-HT3C2 0
A1BG 252
A1BG-AS1 47
```

# Combining htseq Output

- Use `paste` to combine these files:

```
> paste s01.norm.counts s02.norm.counts s03.hypo.counts
s04.hypo.counts
3.8-1.4 0 3.8-1.4 0 3.8-1.4 0 3.8-1.4 0
3.8-1.5 0 3.8-1.5 0 3.8-1.5 0 3.8-1.5 0
5-HT3C2 0 5-HT3C2 0 5-HT3C2 0 5-HT3C2 0
A1BG 252 A1BG 192 A1BG 175 A1BG 153
A1BG-AS1 47 A1BG-AS1 28 A1BG-AS1 31 A1BG-AS1 35
```

# Combining htseq Output

- And `cut` to eliminate redundant columns:

```
> paste s01.norm.counts s02.norm.counts s03.hypo.counts
s04.hypo.counts | cut -f1,2,4,6,8
3.8-1.4 0 0 0 0
3.8-1.5 0 0 0 0
5-HT3C2 0 0 0 0
A1BG 252 192 175 153
A1BG-AS1 47 28 31 35
```

Finally write to a file:

```
> paste s01.norm.counts s02.norm.counts s03.hypo.counts
s04.hypo.counts | cut -f1,2,4,6,8 > hypoxia_hsap.counts
```

# Move the count table down to your laptop

On your laptop's terminal (not slogin):

```
scp -v <yournetID>@dcc-slogin-02.oit.duke.edu:
/work/cc216/490S/cc216/RNAseq_pt3/*.counts
 /<your laptop location>/
```

(pay attention to the spaces! the spacing above isn't correct because the slide is too narrow)

# DESeq2

- See the website for installation instructions
- Needs two things:
  1. A matrix of counts = “cts”
  2. A matrix of sample conditions = “coldata”

# DESeq2

## In R:

```
> head(cts, n = 4)
s01norm s02norm s03hypox s04hypox
3.8-1.4 0 0 0 0
3.8-1.5 0 0 0 0
5-HT3C2 0 0 0 0
A1BG 252 192 175 153

> coldata
condition type
s01norm "untreated" "paired-end"
s02norm "untreated" "paired-end"
s03hypox "treated" "paired-end"
s04hypox "treated" "paired-end"
```

# DESeq2

- Order must match (01, 02, 03...)
- Name the columns and rows appropriately

```
> rownames(coldata)
[1] "s01norm" "s02norm" "s03hypox" "s04hypox"


> colnames(coldata)
[1] "condition" "type"
```

You use the same function to call (see above) define (see below) column and row names:

```
> colnames(coldata) <- c("condition","type")
```

# DESeq2

- Get your data in this format
- Keep track of your work in a .Rmd file!!
- Then (and only then) proceed to running DESeq (see further slides)

# DESeq2 Formatting Tips: Reading Data

- In R:

```
> countfile <- read.table("hypoxia_hsap.counts")
> head(countfile, n = 5)
V1 V2 V3 V4 V5
1 3.8-1.4 0 0 0 0
2 3.8-1.5 0 0 0 0
3 5-HT3C2 0 0 0 0
4 A1BG 252 192 175 153
```

Now we need to get the data into matrices with the correct row and column names

# DESeq2 Formatting Tips: Transforming Data

- `countfile` contains the count data
- We need to format it appropriately

```
> cts <- as.matrix(countfile[,2:5])
> colnames(cts) <- c("s01norm","s02norm","s03hypox","s04hypox")
> rownames(cts) <- countfile[,1]
> head(cts, n = 4)
s01norm s02norm s03hypox s04hypox
3.8-1.4 0 0 0 0
3.8-1.5 0 0 0 0
5-HT3C2 0 0 0 0
A1BG 252 192 175 153
```

First take in only the data (columns 2-5) as a matrix
Then name the columns the sample IDs

# DESeq2 Formatting: coldata matrix

- `coldata` needs to contain the appropriate sample info
- Each row is a sample, each column is information about the sample
- Experimental Treatment, Read Format, Cell Type can all be pertinent info
- First we will make a vector with the information
- Then we'll take the data and make it into a 2x4 matrix

# DESeq2 Formatting: coldata matrix

```
> sampleinfo <- c("untreated","untreated","treated","treated",
"paired-end","paired-end","paired-end","paired-end")
> sampleinfo
[1] "untreated" "untreated" "treated" "treated" "paired-end"
"paired-end"
[7] "paired-end" "paired-end"
> coldata <- matrix(sampleinfo, nrow = 4, ncol = 2, byrow = F)
> coldata
[,1] [,2]
[1,] "untreated" "paired-end"
[2,] "untreated" "paired-end"
[3,] "treated" "paired-end"
[4,] "treated" "paired-end"
```

# DESeq2 Formatting: coldata matrix

- `coldata` now needs correct row and column names
- What should they be? (hint: it is in this slide deck)
- Once you decide, use `rownames()` and `colnames()` to add them

# Today's (Remaining) Goals

1. Format data for DESeq
2. Successfully run DESeq

# DESeq2 guides

- Here are the DESeq guides that I have summarized in this walkthrough:
- Walkthrough Link
- Focus on “Quick Start” and more specifically:
- Setting the R objects `cts` and `coldata` correctly
- Using `paste` (a unix command) to format your data into `cts`

http://bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html

# DESeq2

- To run DESeq, first create a “DESeq dataset”

```
> dds <- DESeqDataSetFromMatrix(countData = cts,
colData = coldata,
design = ~ condition)

> dds
```

Where cts and coldata are the files described earlier

Output:

```
class:  DESeqDataSet
dim:  45381 4
metadata(1):  version
```

# DESeq2

- Let's throw out genes with low expression levels (here, below 10)

```
> keep <- rowSums(counts(dds)) >= 10
> dds <- dds[keep,]
> dds$condition <- relevel(dds$condition, ref = "untreated")
> dds
```

Output:

```
class:  DESeqDataSet
dim:  17380 4
metadata(1):  version
```

We've shrunk our gene list by roughly 60 percent

# DESeq2

- Now... Run DESeq!!!

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

Output:

```
log2 fold change (MAP): condition treated vs untreated
Wald test p-value:  condition treated vs untreated
DataFrame with 17380 rows and 6 columns
baseMean log2FoldChange lfcSE stat pvalue

A1BG 189.927175 -0.20541335 0.16773404 -1.22463727 0.2207119
A1BG-AS1 34.902293 0.02668120 0.26939459 0.09904134 0.9211054
```

# DESeq2

- A few final things to do and explore:
- Sort the results by p-value

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

- Summarize the results

```
> summary(res)
```

- How many genes are significant (at 0.10 and 0.05)?

```
> sum(res$padj < 0.1, na.rm=TRUE)
> sum(res$padj < 0.05, na.rm=TRUE)
```

# Plotting in DESeq2

- There are several ways to plot this data
- You can used the built-in DESeq functions:

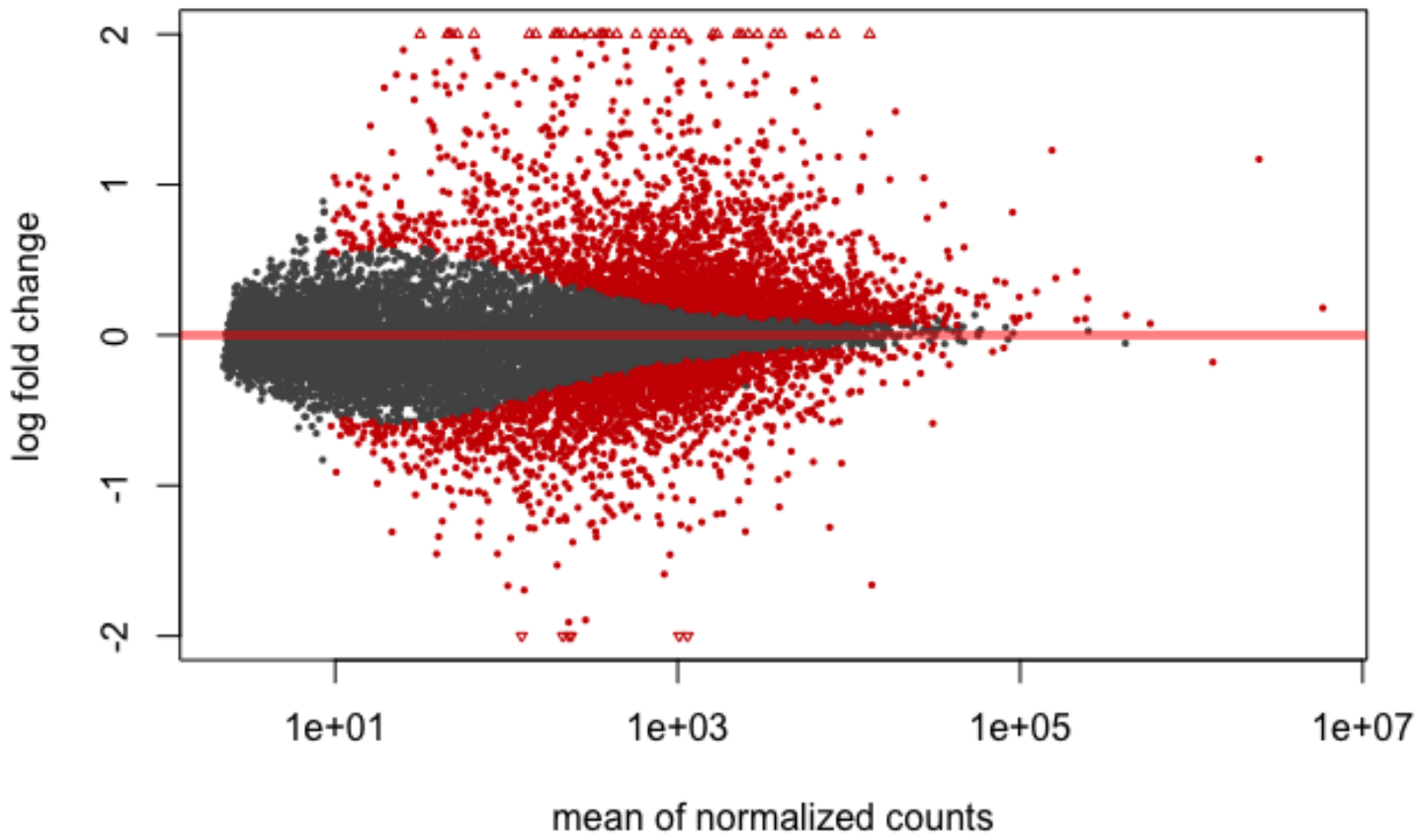
```
> plotMA(res, ylim=c(-2,2))


> plotCounts(dds, gene=which.min(res$padj),
intgroup="condition")
```

- Or the R plotting function:

```
> plot(res$log2FoldChange,-log(res$padj))
```

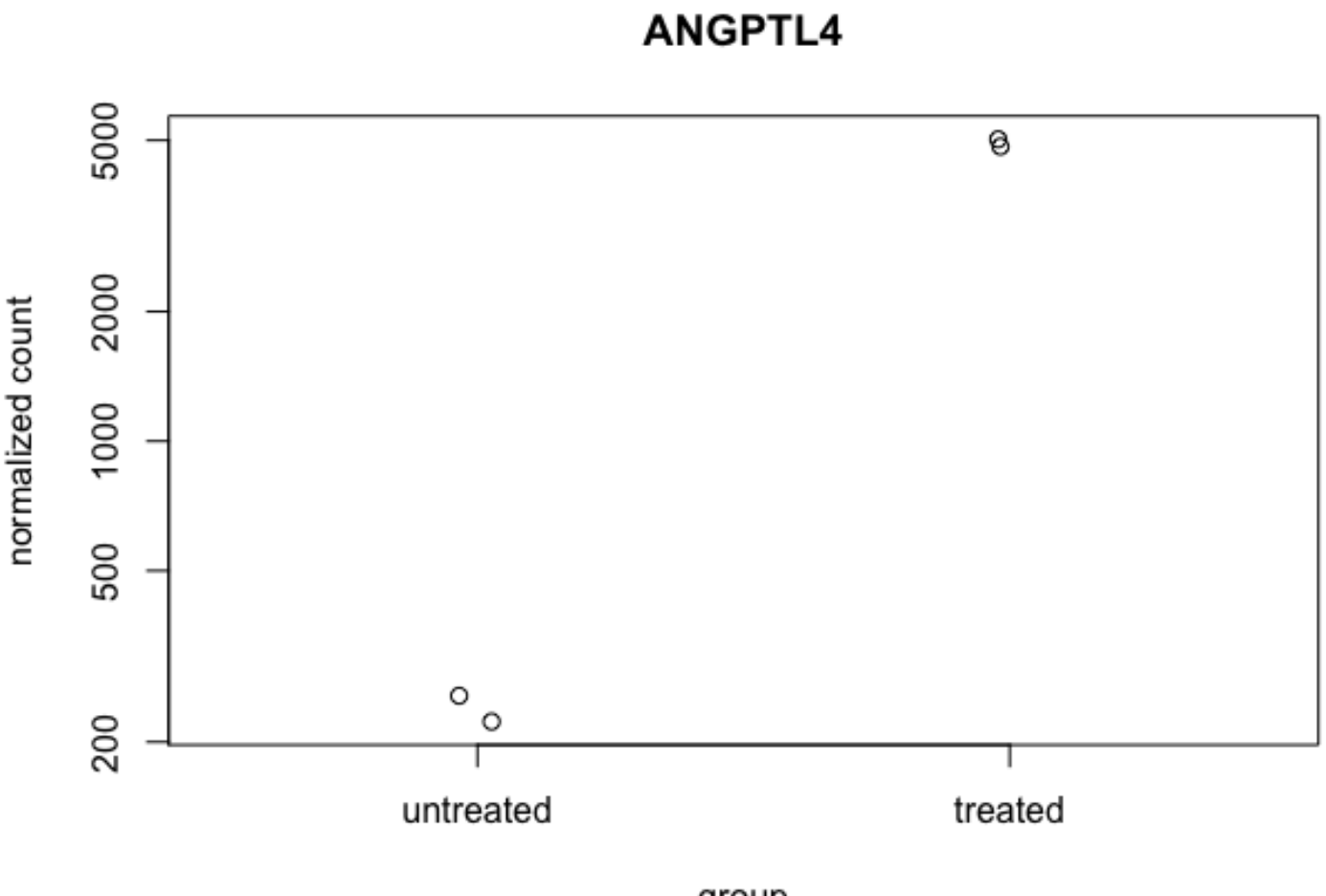
# Plotting in DESeq2

```
> plotMA(res, ylim=c(-2,2))
```



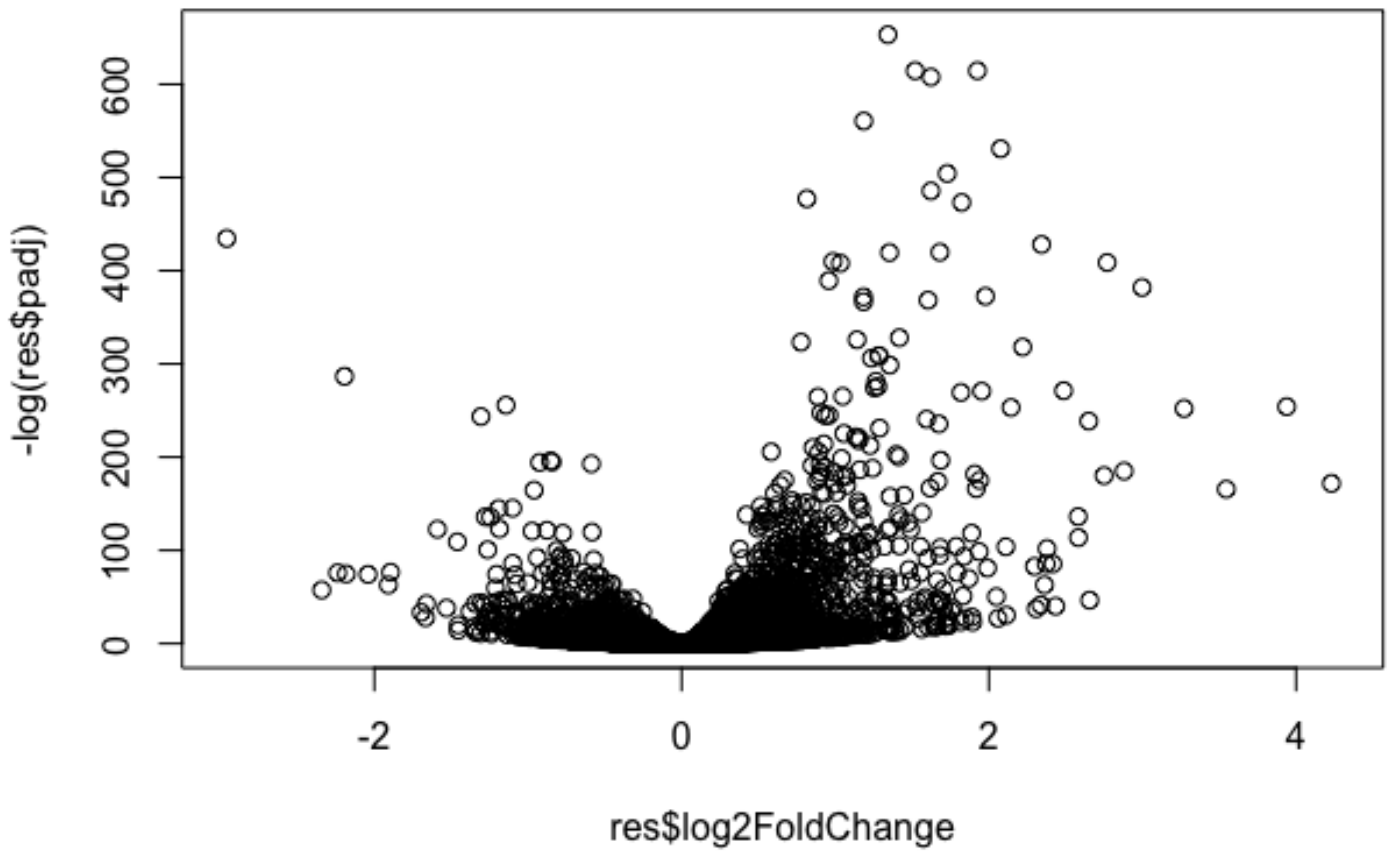
# Plotting in DESeq2

```
> plotCounts(dds, gene=which.min(res$padj),
intgroup="condition")
```



# Plotting in DESeq2

```
> plot(res$log2FoldChange,-log(res$padj))
```



# The End