# Statistical Inference for RNA-seq

Keegan Korthauer

February 14, 2023

# Reminders/announcements

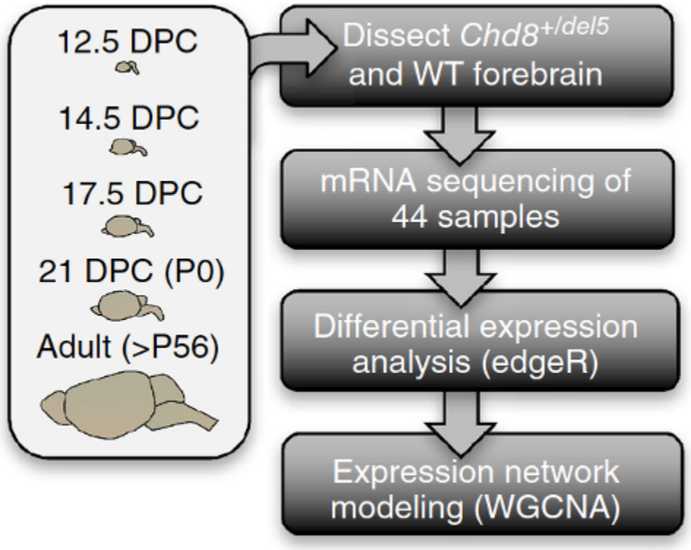
- Paper Critique due today
- Analysis Assignment released - due Feb 28
  - use limma + **voom** to analyze RNA-seq data from paper you critiqued

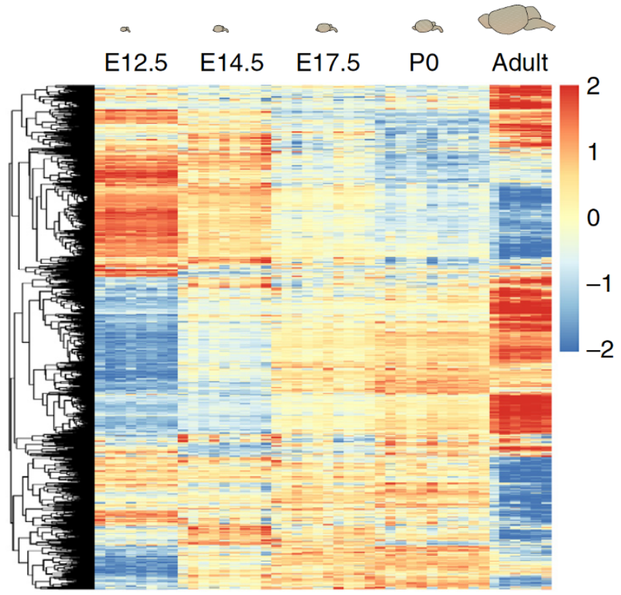
# Learning objectives

- Understand *why* and *when* between- and within-sample normalization are needed
- Apply common between- and within-sample normalization approaches to RNA-seq counts
- Understand why the *count nature* of RNA-seq data requires modification to the Differential Expression approaches applied to microarray data (e.g. `limma`)
- Apply models such as **limma-trend, limma-voom, `DESeq2` and `edgeR`** for inference of Differential Expression

# A CHD8 RNA-seq experiment

- Gompers et al. (Nature Neuroscience 2017) analyzed 26 Chd8 mutant and 18 WT mice
  - Tested for differential expression across ~12K genes accounting for sex, developmental stage and sequencing batch
- We'll use this dataset throughout this lecture to illustrate RNA-seq analysis




Figures from Gompers et al. (2017) paper

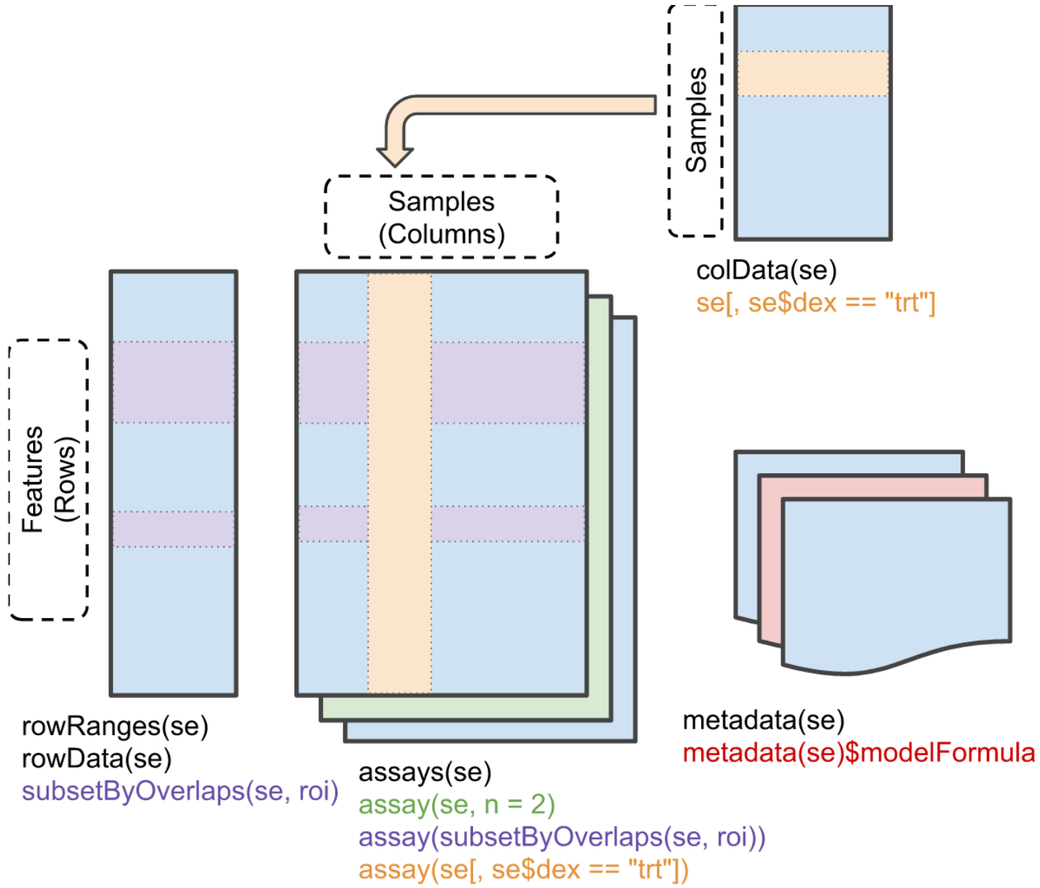
# `SummarizedExperiment` object

`SummarizedExperiment`: A special object format that is designed to contain data & metadata

▶ Code

```
1 sumexp
```

```
class: SummarizedExperiment
dim: 20962 44
metadata(0):
assays(1): counts
rownames(20962): 0610005C13Rik 0610007P14Rik ... Zzef1
Zzz3
rowData names(0):
colnames(44): Sample_ANAN001A Sample_ANAN001B ...
Chd8.adult.S29
  Chd8.adult.S31
colData names(7): DPC Sex ... FeatureCounts Sample
```




Anatomy of a SummarizedExperiment object

# A look inside our `SummarizedExperiment` object

Counts | Metadata

```
1 assays(sumexp)$counts %>% head()
```

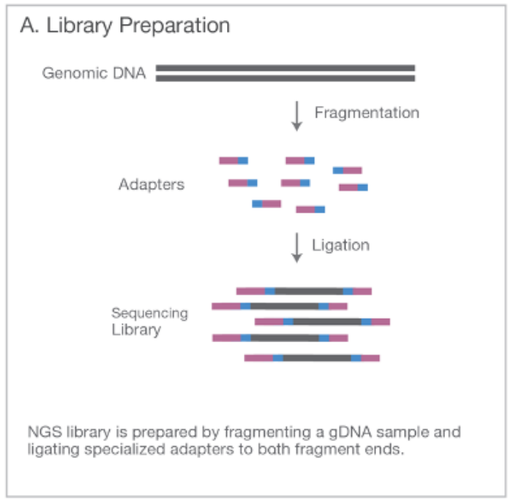
| | Sample_ANAN001A | Sample_ANAN001B | Sample_ANAN001C | Sample_ANAN001D |
|---|---|---|---|---|
| 0610005C13Rik | 20 | 24 | 20 | 8 |
| 0610007P14Rik | 1714 | 1796 | 1970 | 1996 |
| 0610009B22Rik | 578 | 866 | 790 | 858 |
| 0610009L18Rik | 50 | 82 | 38 | 64 |
| 0610009O20Rik | 2580 | 2964 | 2942 | 3084 |
| 0610010B08Rik | 0 | 10 | 0 | 2 |
| | Sample_ANAN001E | Sample_ANAN001F | Sample_ANAN001G | Sample_ANAN001H |
| 0610005C13Rik | 6 | 18 | 17 | 20 |
| 0610007P14Rik | 1864 | 1626 | 2103 | 1422 |
| 0610009B22Rik | 786 | 662 | 710 | 502 |
| 0610009L18Rik | 70 | 28 | 51 | 44 |
| 0610009O20Rik | 2848 | 2640 | 3210 | 2160 |
| 0610010B08Rik | 0 | 0 | 3 | 0 |
| | Chd8.e14.S12 | Chd8.e14.S13 | Chd8.e14.S14 | Chd8.e14.S16 | Chd8.e14.S17 |

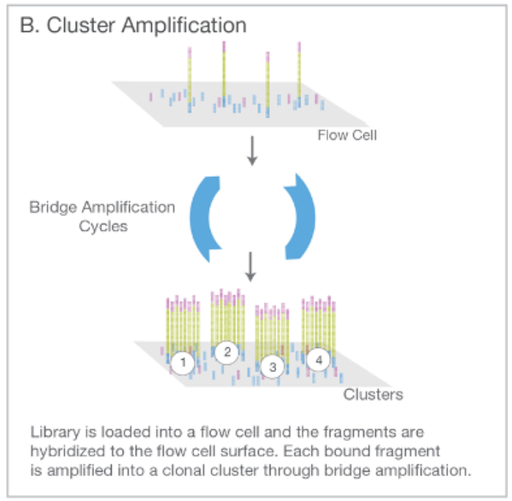
# Now we have count data[1]

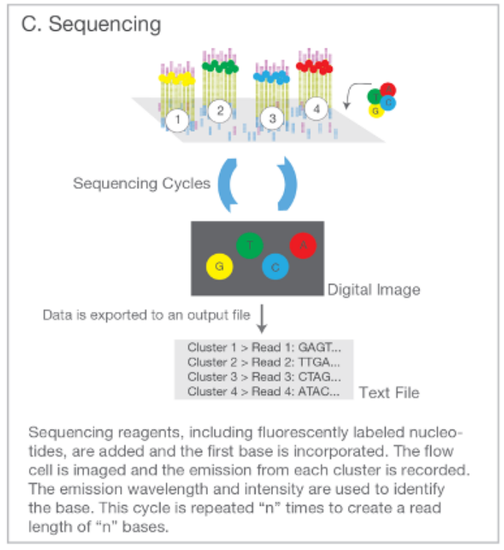
- In the Nrl microarray experiment we worked with **continuous** microarray values
- Now we will work with the raw RNA-seq **counts** (discrete)
- These counts represent the number of reads mapping to each feature (gene or transcript) - here we have gene counts
- Seminar 6 explored how to obtain read counts from alignment (BAM or SAM) files

1. Note that only RPKM values were provided in GEO; raw counts obtained directly from authors

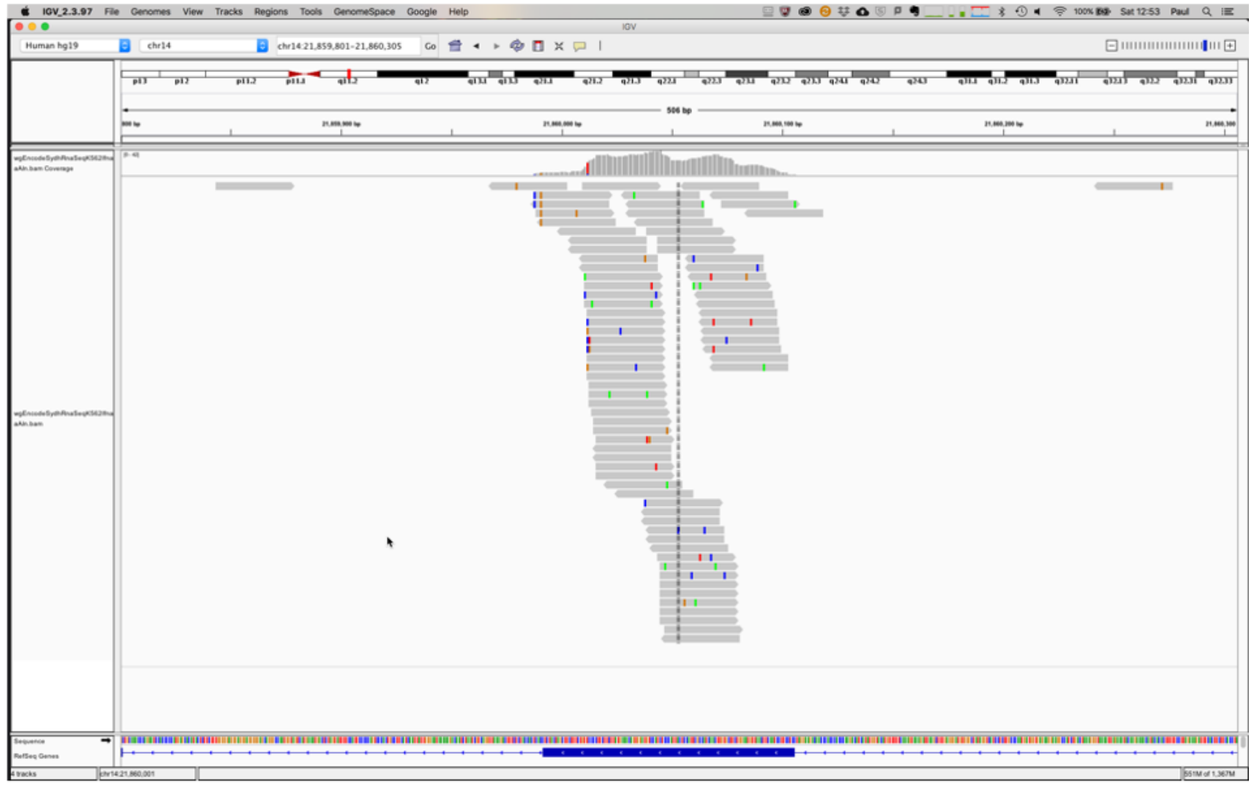
# Recall where these counts came from





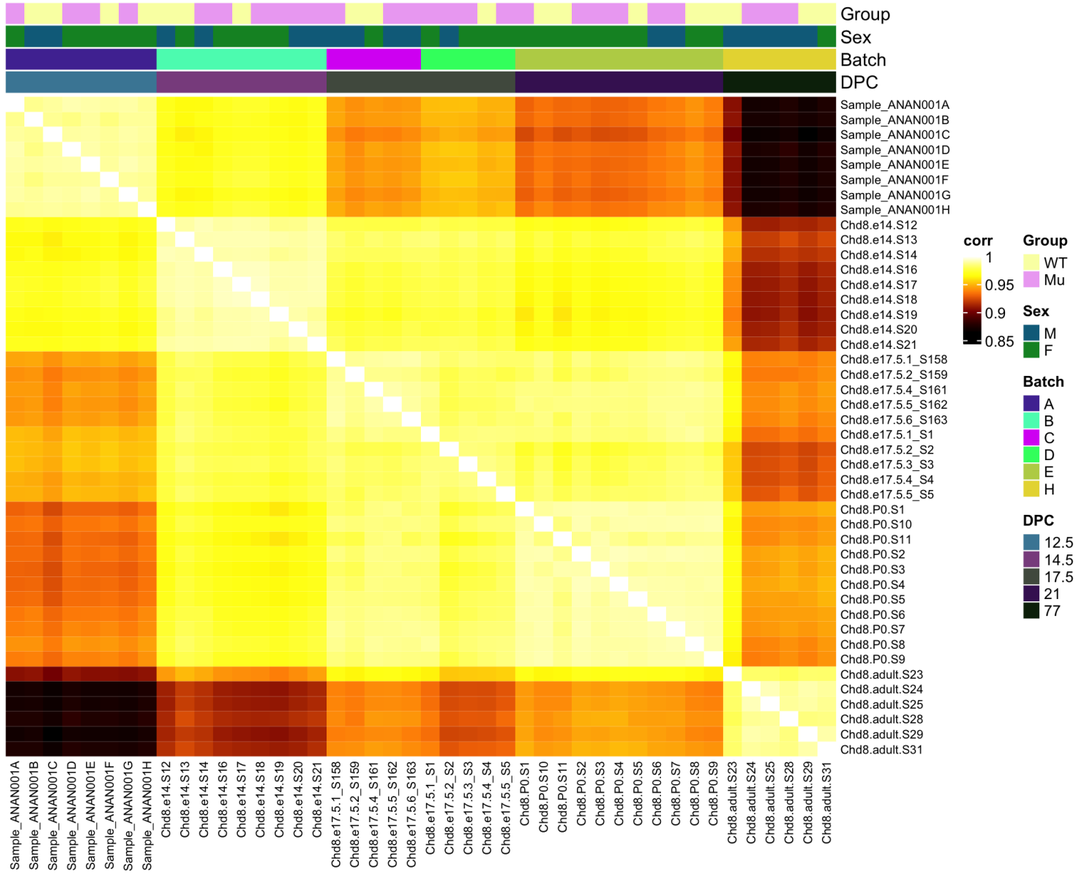


Millions of short (~100bp) reads, each assigned to a gene

# EDA summary

- Experimental design variables:
  - **Group** (Genotype: Chd8 mutant vs WT)
  - **Sex** (M vs F, 2 level factor)[1]
  - **DPC** (days post conception, 5 level factor)
  - **Batch** (sequencing run)
- Notable findings:
  - **Batch** and **DPC** are major sources of variation
  - **Batch** and **DPC** are confounded
  - One sample is a potential minor outlier

▶ Code




1. Note that sex was mislabeled for some samples in the GEO entry (we are using a corrected version obtained from the authors)

# EDA: confounding

## **Batch** (sequencing run) and **DPC** (days post conception) are confounded

```
table(sumexp$SeqRun,
      sumexp$DPC)
```

```
  12.5 14.5 17.5 21 77
A    8    0    0  0  0
B    0    9    0  0  0
C    0    0    5  0  0
D    0    0    5  0  0
E    0    0    0 11  0
H    0    0    0  0  6
```

# Differential expression analysis on Chd8 data

- Main variable of interest: **Group** (Genotype: Chd8 mutant vs WT)
- We'd like to fit a model for each gene so we can test for Group effect, and adjust for:
  - **Sex** (M vs F, 2 level factor)
  - **DPC** (days post conception, 5 level factor)

- Using what we learned in previous lectures, we can formulate this model as

$$Y_i = \theta + \tau_{Mut} x_{i,Mut} + \tau_F x_{i,F} + \tau_{D14.5} x_{i,D14.5} + \tau_{D17.5} x_{i,D17.5} + \tau_{D21} x_{i,D21} + \tau_{D77} x_{i,D77} + \epsilon_i$$

$$x_{i,Mut} = \begin{cases} 1 \text{ if } i \text{ is Mutant} \\ 0 \text{ otherwise} \end{cases}, \quad x_{i,F} = \begin{cases} 1 \text{ if } i \text{ is Female} \\ 0 \text{ otherwise} \end{cases}, \quad x_{i,D\#} = \begin{cases} 1 \text{ if } i \text{ is DPC\#} \\ 0 \text{ otherwise} \end{cases}$$

where $D\# \in \{D14.5, D17.5, D21, D77\}$

# Differential expression analysis on Chd8 data

- Our model has no interaction term (though we could add one if we wish)
- $p = 7$ parameters to estimate in our model: $\theta, \tau_{Mut}, \tau_F, \tau_{D14.5}, \tau_{D17.5}, \tau_{D21},$ and $\tau_{D77}$
- $n = 44$ samples total, so our model has $n - p = 44 - 7 = 37$ degrees of freedom
- What is the null hypothesis for the test of differential expression between Chd8 Mut and WT using our model?

- Recall that since this is an additive model, the parameters represent **main effects** (not conditional)

# Design matrix in R

```
modm <- model.matrix(~ Sex + Group + DPC, data = colData(sumexp))
modm
```

| | (Intercept) | SexF | GroupMu | DPC14.5 | DPC17.5 | DPC21 | DPC77 |
|---|---|---|---|---|---|---|---|
| Sample_ANAN001A | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Sample_ANAN001B | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sample_ANAN001C | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sample_ANAN001D | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Sample_ANAN001E | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Sample_ANAN001F | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Sample_ANAN001G | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Sample_ANAN001H | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Chd8.e14.S12 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| Chd8.e14.S13 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| Chd8.e14.S14 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| Chd8.e14.S16 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| Chd8.e14.S17 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| Chd8.e14.S18 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

# Are we ready to fit the model?

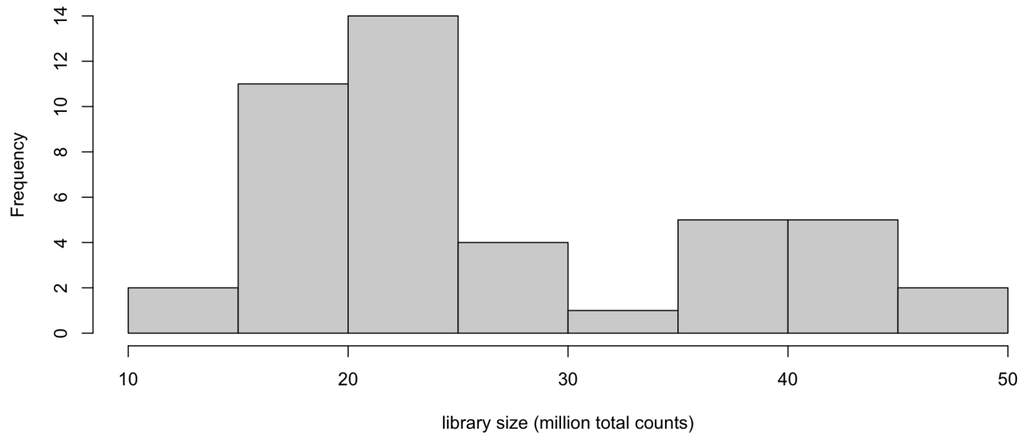
Might start with the `limma` approach on the raw counts, but…

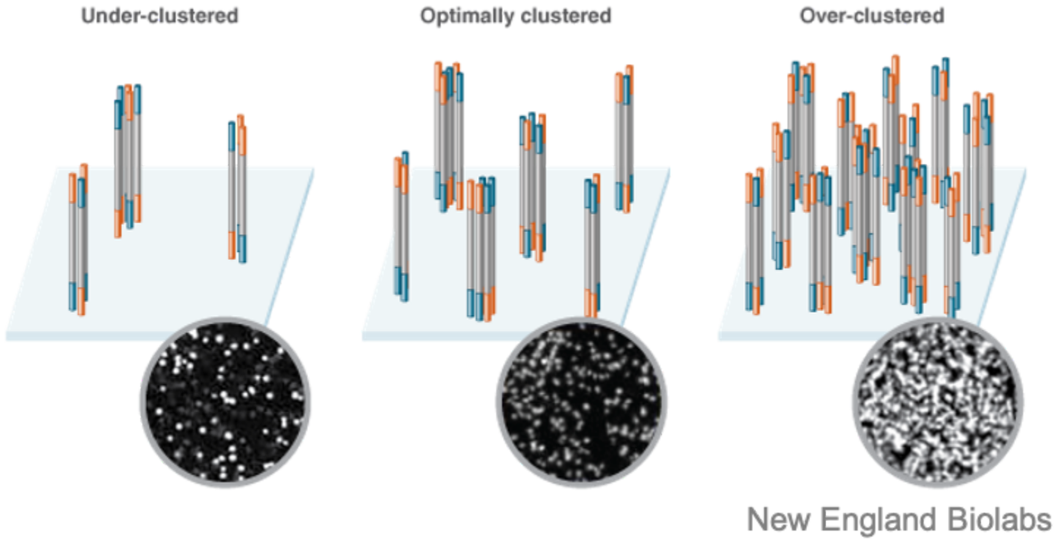
**Not so fast** - we have to consider additional sources of variation!

# Library size (sequencing depth)

- **Library size**: Total number of read counts per sample
- Ideally this would be the same for all samples, but it isn't
- Number of reads per sample depends on factors like how many samples were multiplexed and how evenly, cluster density, RNA quality, etc.

▶ Code

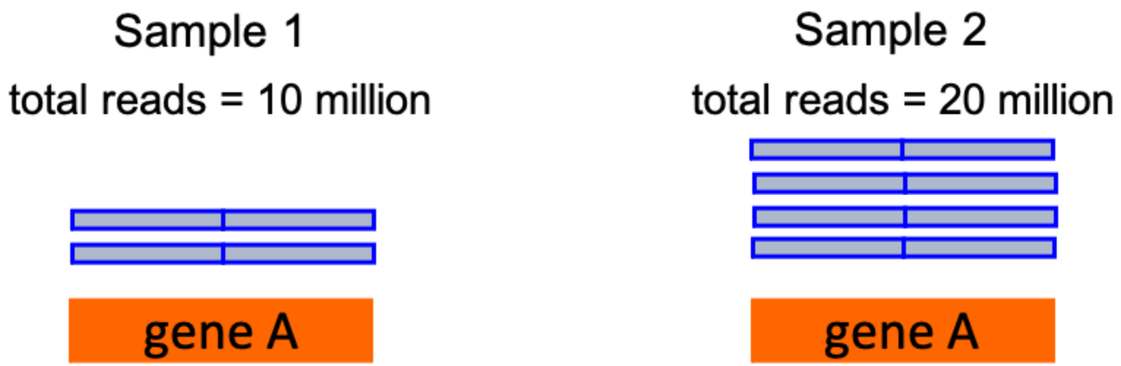


# Why does library size matter?

**Read depth variation is a potential source of confounding!**

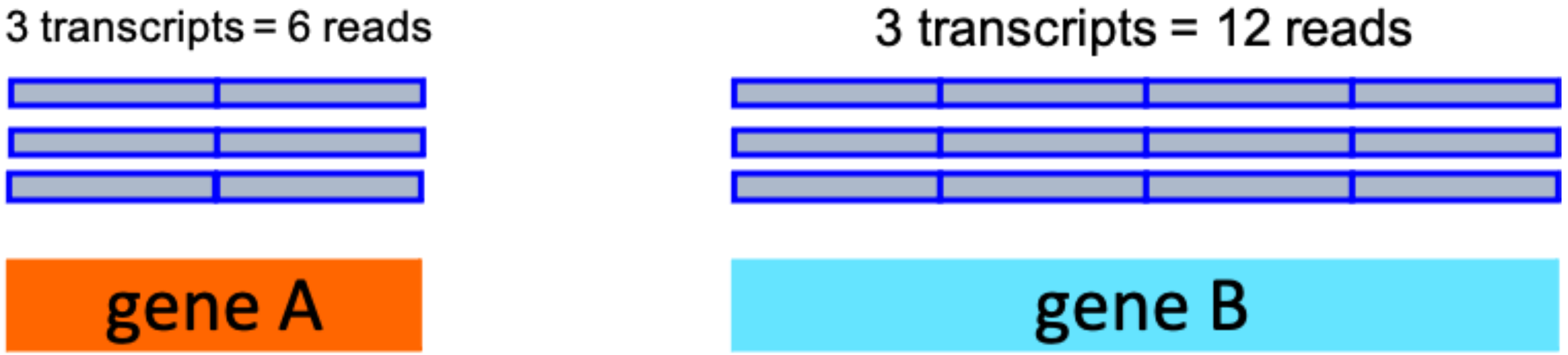
- We typically want to compare gene counts **between** samples
- **Intuition**: if we sequence one group of samples 2X as much, gene counts in that sample look ~2X as large even if there's no DE!




- You may come across (older) literature where data was down-sampled to make library sizes the same (**not recommended**)

# Within-sample comparisons

- Other factors of variation come into play if we also want to compare counts between genes within sample (less common)
- At the same expression level, longer genes/transcripts have more read counts

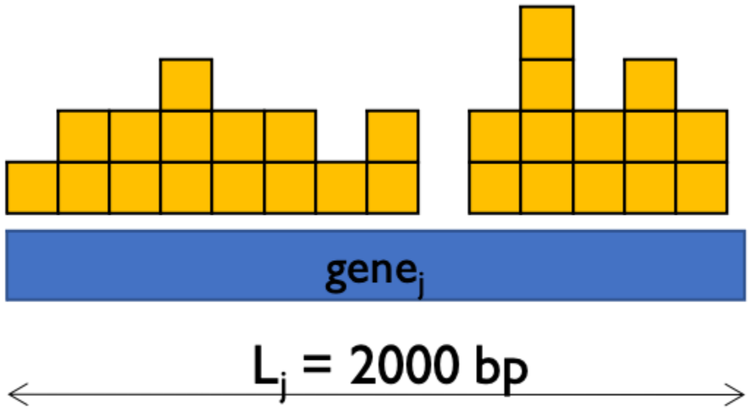


# How can we make fair between- and within-sample comparisons?

- **Normalized expression units**: expression values adjusted for factors like library size, gene length
  - e.g.RPKM/FPKM, TPM, CPM
  - useful for visualization / clustering
- **Normalization factors**: scalar values representing relative library size of each sample
  - e.g. TMM, DESeq size factors
  - useful to include in models of raw counts to adjust for library size
- For analysis (e.g. DE) it is ideal to start with **raw counts**
  - raw counts required for many methods
  - can always compute normalized values from raw counts (but not vice versa)

# Normalized expression units

- **RPKM/FPKM**: reads/fragments per kb of exon per million mapped reads




$R_{ij} = 28$ reads in gene $j$, sample $i$

$\Sigma_j R_{ij} = 11$ million reads in sample i

$$RPKM_{ij} = \frac{R_{ij}}{\frac{L_j}{10^3}\frac{\Sigma_j R_{ij}}{10^6}} = \frac{28}{\frac{2000}{10^3}\frac{1.1\times10^7}{10^6}} = 1.27$$

- FPKM is the more appropriate term for paired-end data

# Normalized expression units, continued

- **TPM**: Transcripts per million

$$TPM_{ij} = \frac{R_{ij}}{L_j} \frac{10^6}{\Sigma_j R_{ij}/L_j} = \frac{FPKM_{ij}}{\Sigma_j FPKM_{ij}/10^6}$$

- **CPM**: Counts per million

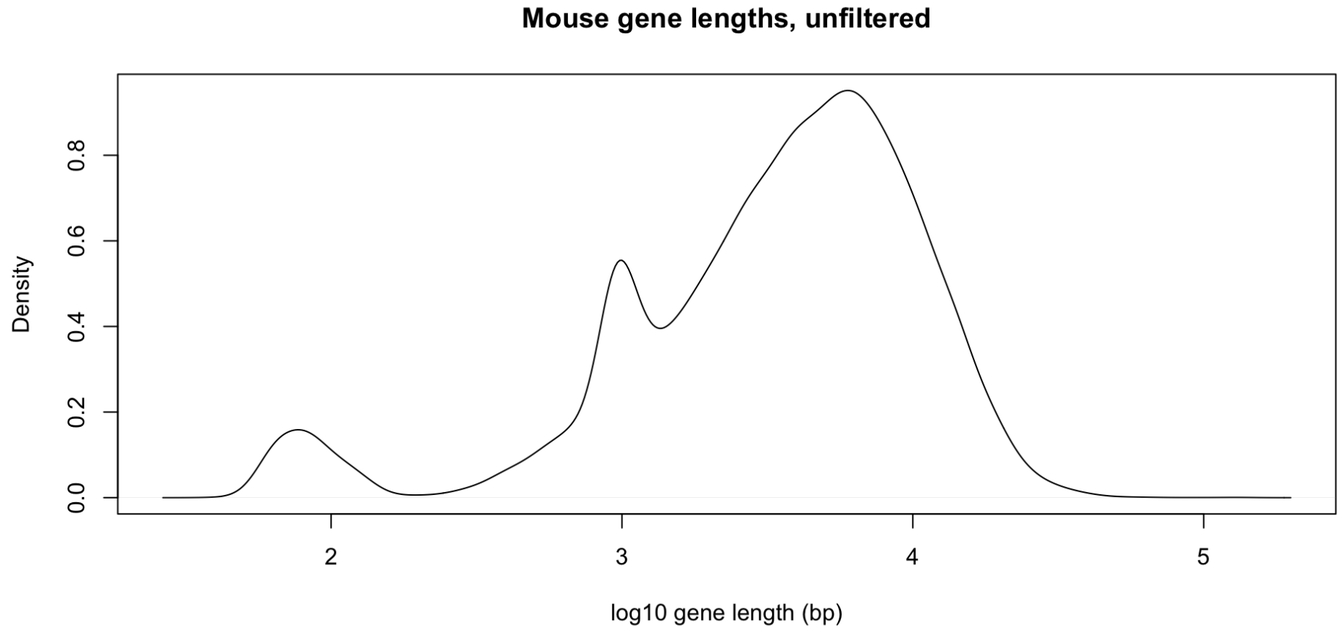
$$CPM_{ij} = \frac{R_{ij}}{\Sigma_j R_{ij}/10^6}$$

- See this useful blog post on relationship between these units
- Which of these measures are between-sample normalization measures? Within-sample? Both?

# How much does “gene length” vary?

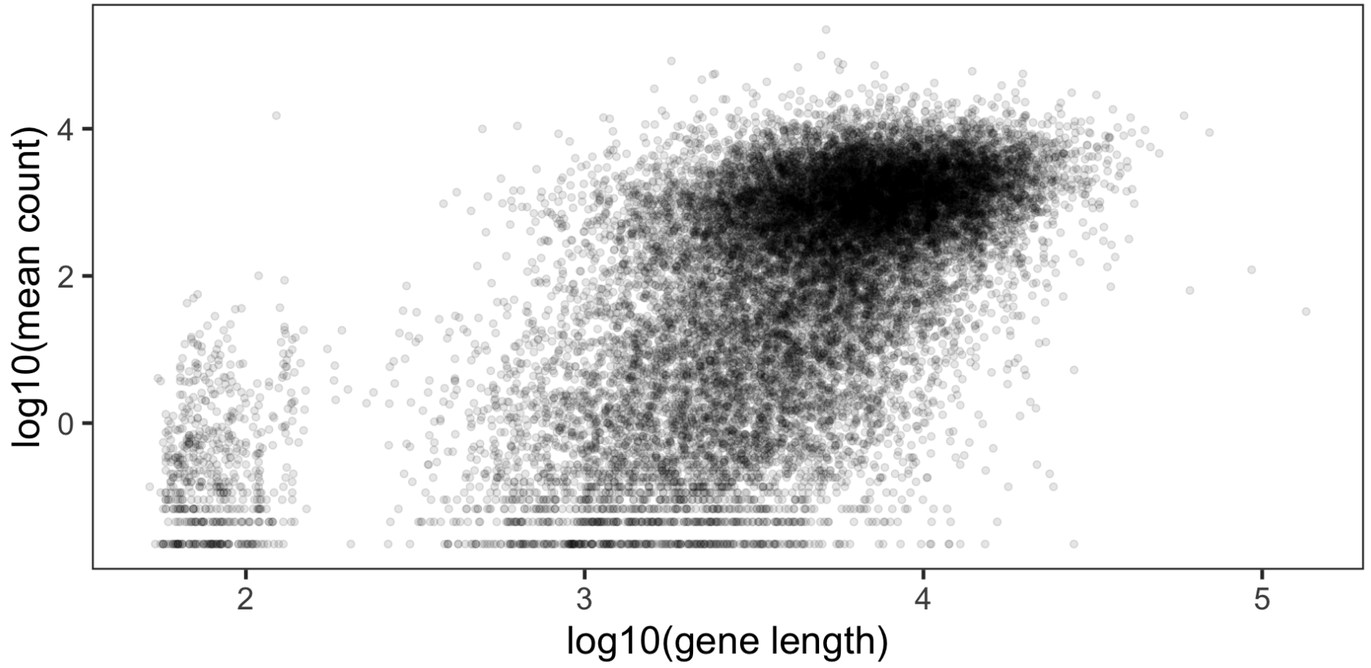
- Really we mean “total effective length of transcript used in assigning reads to genes”
- If all genes are same lengths, FPKM won’t do anything interesting
- In mouse, “gene length” varies mostly between ~2.5Kb - 4.3Kb; your organism may vary

▶ Code



# How does gene length relate to counts?

▶ Code



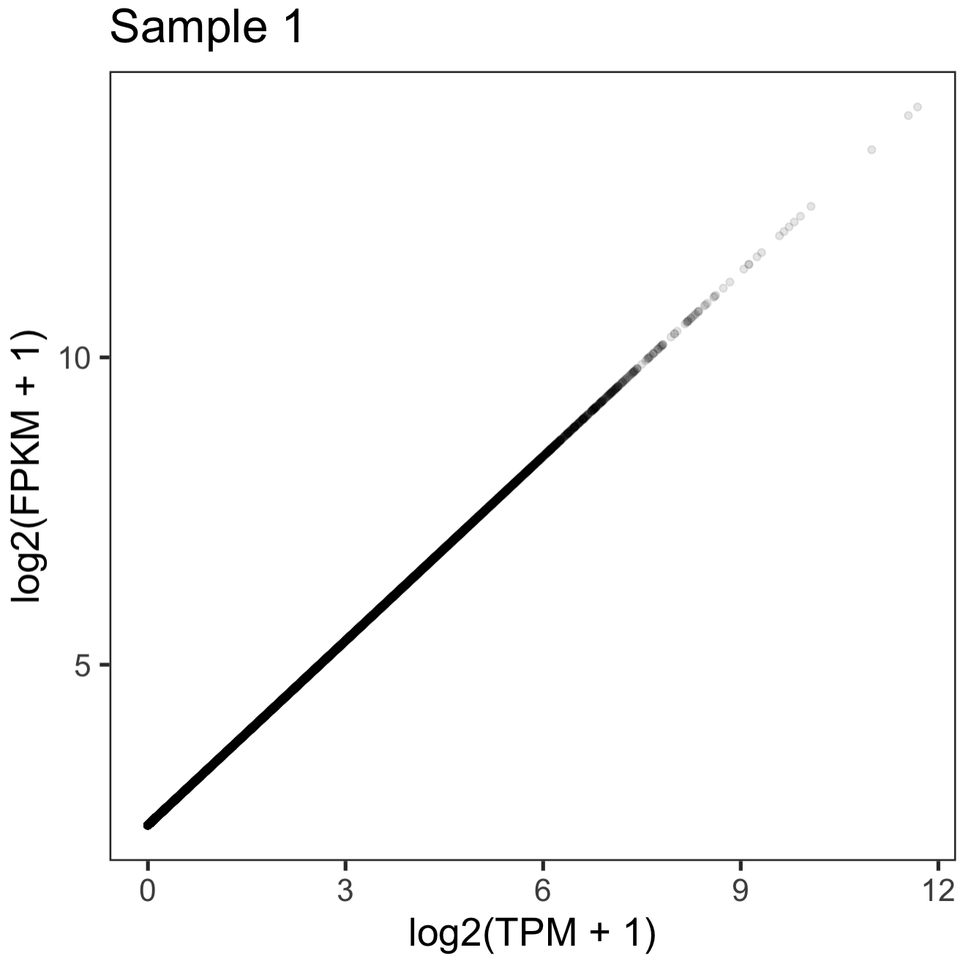

- If all genes were expressed at same level (same # molecules/cell), expect a 1:1 relation
- Of course they are not, so the effect of length is less obvious
- Rank correlation between length and mean expression in our example data is 0.573

# FPKM vs TPM

These metrics both enable comparison of expression levels of different genes within sample.

Any doubt about “gene length” will be propagated to both measures.

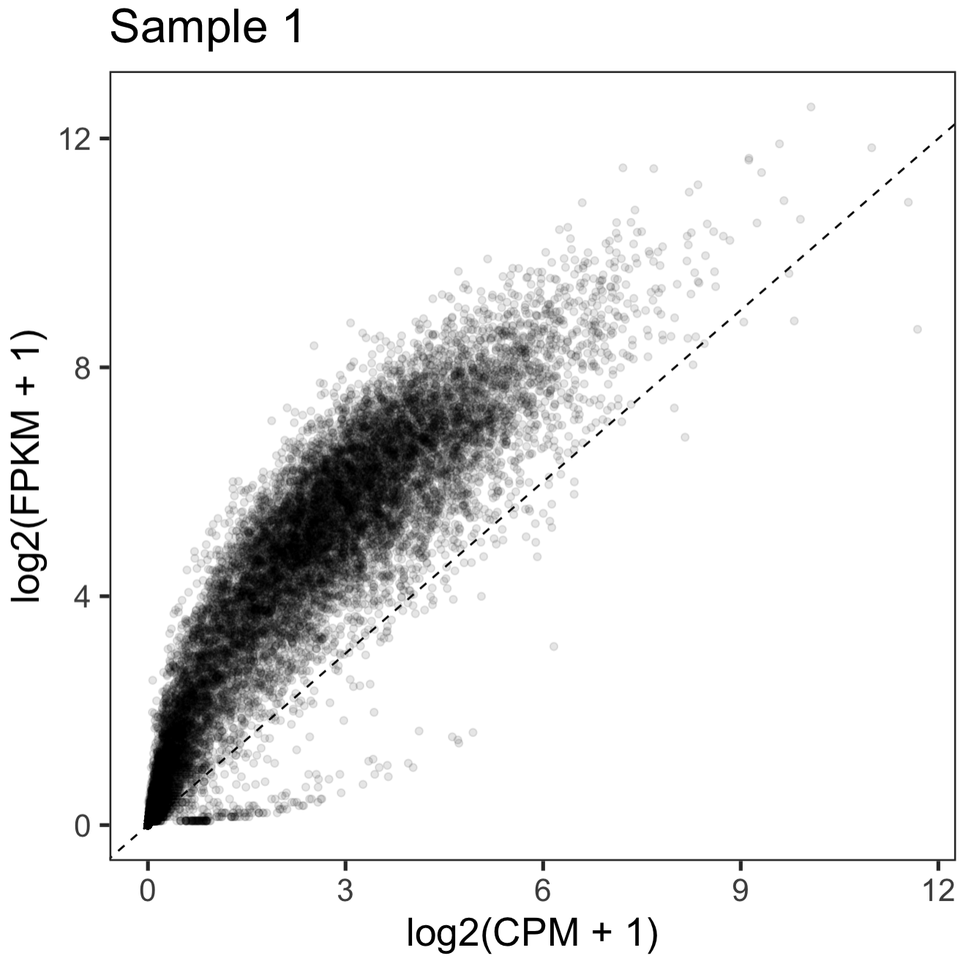
▶ Code



# FPKM vs CPM

If we're comparing samples to each other, there's no important difference between FPKM/TPM and CPM so long as we assume "effective gene length" is constant across samples

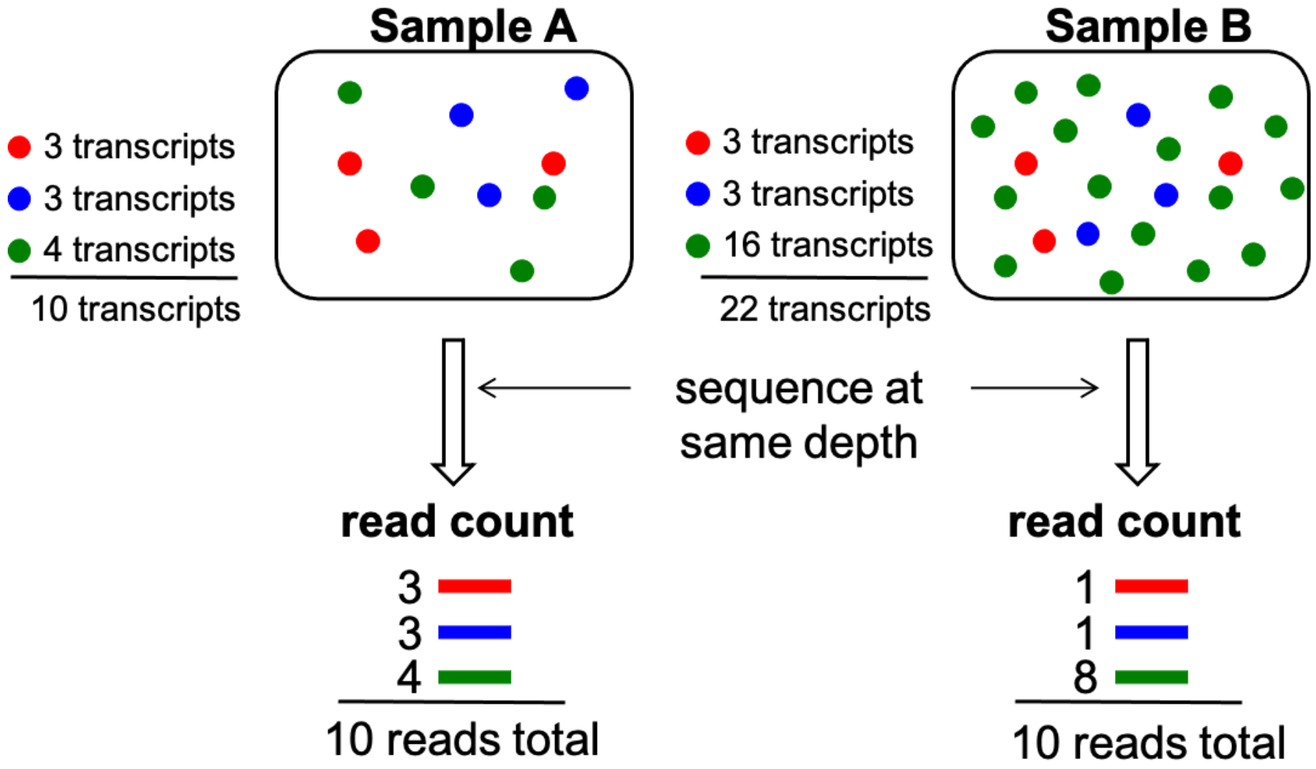
▶ Code



# Between-sample normalization

- Computing FPKM, CPM or TPM largely corrects for differences in library size
- However, there is a complication: “Sequence space”
  - Finite number of reads implies that observing reads for one gene decreases ability to observe reads for other genes
  - This is a fundamental difference from microarrays, where each spot is essentially independent
- This isn’t a major problem unless there are large differences in composition between samples, but should be inspected
  - Normalization factors are generally robust to this

# Effect of sequence space



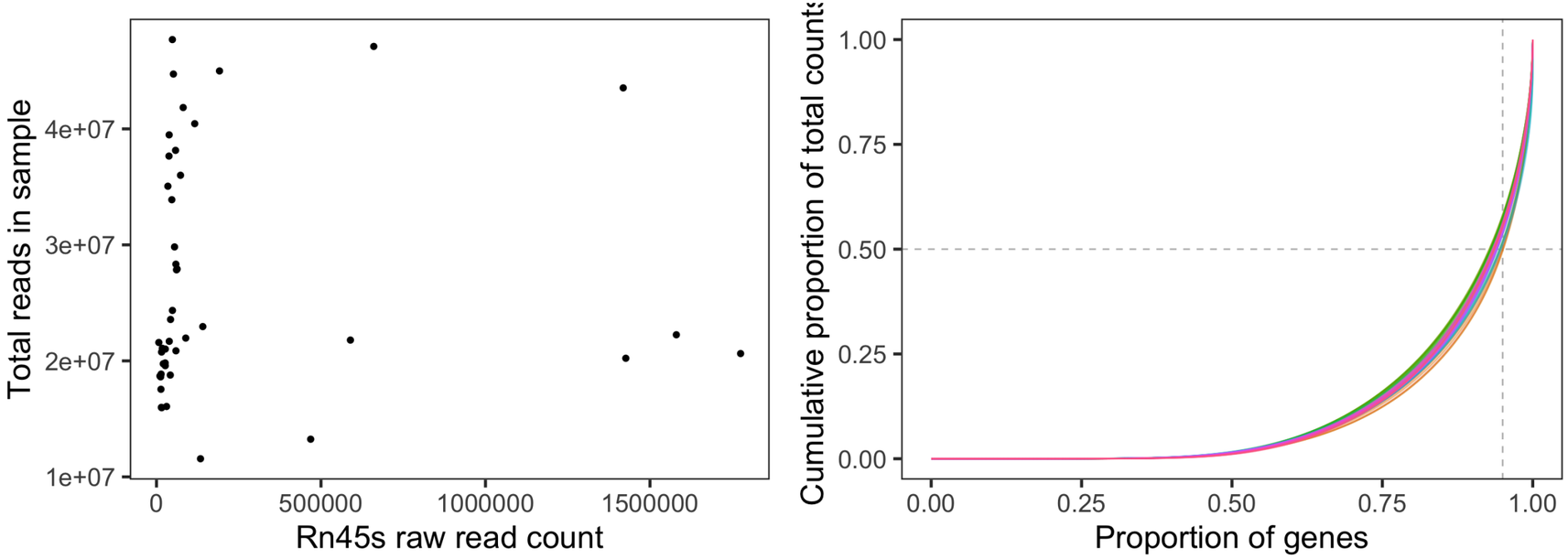

- By CPM or FPKM, red & blue appear down-regulated in sample B (only green is DE)
- Adjusting expression levels in Sample B by a factor of 3 would be needed

See Robinson and Oshlack (2010)

# Sequence space in our example data

- Rn45s gene has $> 100,000$ mean reads per sample ($> 5\%$ of reads in some samples)
- ~5% of the genes take up ~50% of reads, but this is consistent across samples
- Side note: Rn45s is potentially a contaminant - a ribosomal RNA that should have been removed during sample prep, which involved poly-A selection

▶ Code



# Normalization factors

# Preprocessing: filtering lowly expressed genes

- Common step which can be beneficial for a few reasons:
  - Genes with very low mean expression across samples may be uninteresting
  - Fitting models on a smaller number of genes can be faster
  - May obtain a more ‘well-behaved’ association between mean and variance, which might affect some methods (e.g. Voom)
- No universal threshold; original study: keep genes with $\geq 2$ samples that have CPM $> 10$

```
assays(sumexp)$cpm <- cpm(counts, log = FALSE, normalized.lib.sizes = FALSE)
keep <- which(rowSums(assays(sumexp)$cpm > 10) >= 2)
length(keep)
```

```
[1] 12158
```

```
sumexp <- sumexp[keep,]
```

# Differential expression: Why we need new methods

- **Goal**: accurate p-values for our hypothesis tests
  - Accurate: “Uniform under the null”
  - Properties relied upon for inference from $t$-statistics may not hold for count data
- Perhaps most important: **Heteroskedasticity** and **Overdispersion**
  - Strong mean-variance relationship expected with count data
    - violation of constant variance assumption of linear models
    - over- or under- shrinkage of genes, depending on variance levels
  - Biological variance over and above binomial sampling variance

# Properties of expression data: counts

> **Important**
>
> We are focused on the distribution of expression values for a gene across technical or biological replicates - for this discussion we care less about comparing two genes within a sample

**Microarray**:

- Signal is fundamentally counts (deep down: photon detection)
- But values are averaged across pixels and counts are high
- Never really have zero: background
- “Continuous-like”

**Sequencing**:

- Unit of measurement is the read; no such thing as 0.2 read
- Counts of reads start at 0
- As counts get high, the distinction with microarrays should decrease

# Statistics of counts: Binomial

- Number of reads observed for gene $g$ in a given sample is a random variable
- Say RNA for gene $g$ is present "in the cell" at about 1 out of every 1,000,000 molecules
  - Abundance $a_g = 1/1,000,000 = 1 \times 10^{-6}$ ("probability of success")
- If we randomly pick $R_i = \Sigma_g R_{ig} = 1,000,000$ molecules ("reads" = "trials"), how many gene $g$ RNAs will we see? $E(R_{ig}|R_i) = ?$

- But could get 0, 2, 3, 4, … etc just by chance: this is a **Binomial** distribution
  - probability distribution of the number of successes in $n$ trials, each with probability of success $p$ is $(Binomial(n, p))$
  - mean = $np$
  - our example: $R_{ig} \sim Binomial(R_i, a_g)$ where $n = R_i$ and $p = a_g$

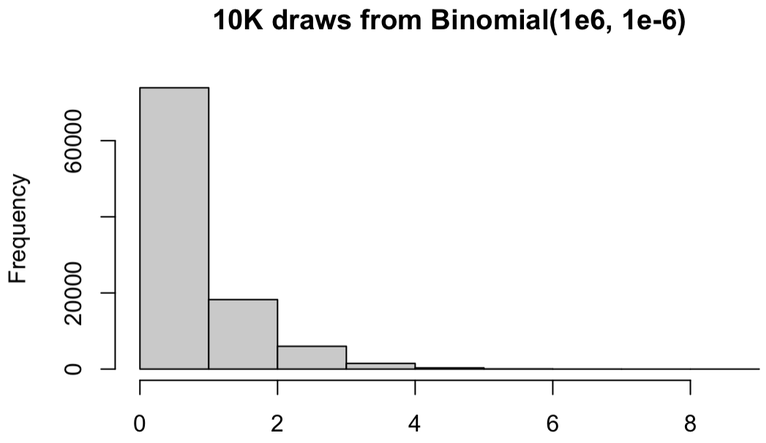
# Statistics of counts: Poisson

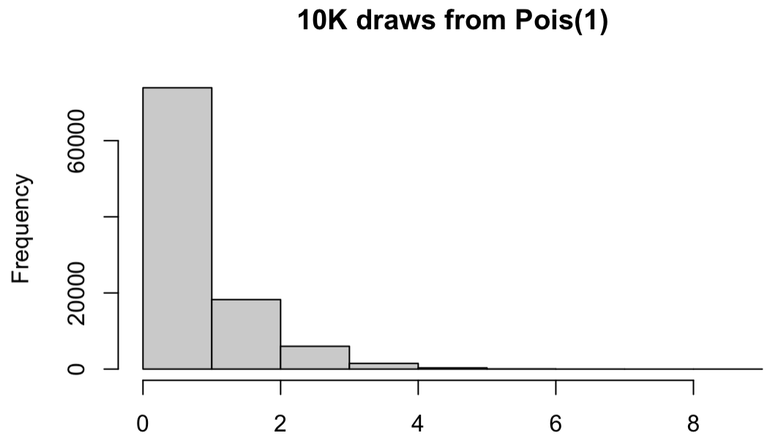
- Poisson distribution counts discrete occurrences along a continuous interval of time/space
  - parameterized by a rate parameter $\lambda$
  - key difference from Binomial: number of events can be infinitely large
- For count data, the variance is a function of the mean (*very* different from a normal)
  - Binomial: mean $= np$, variance $= np(1 - p)$
  - Poisson: mean = variance = $\lambda$

# Statistics of counts: approximations

- **Binomial approximation of Poisson**: for large $n$ and small $np$ (rule of thumb: $n > 20$ & $np < 5$)
  - Approximately $R_{ig} \sim Poisson(R_i a_g)$
- **Binomial approximation of Normal**: For large $np$ (rule of thumb: $np$ & $n(1-p) > 5$)
  - Approximately $R_{ig} \sim Normal(R_i a_g,\ R_i a_g(1-a_g))$

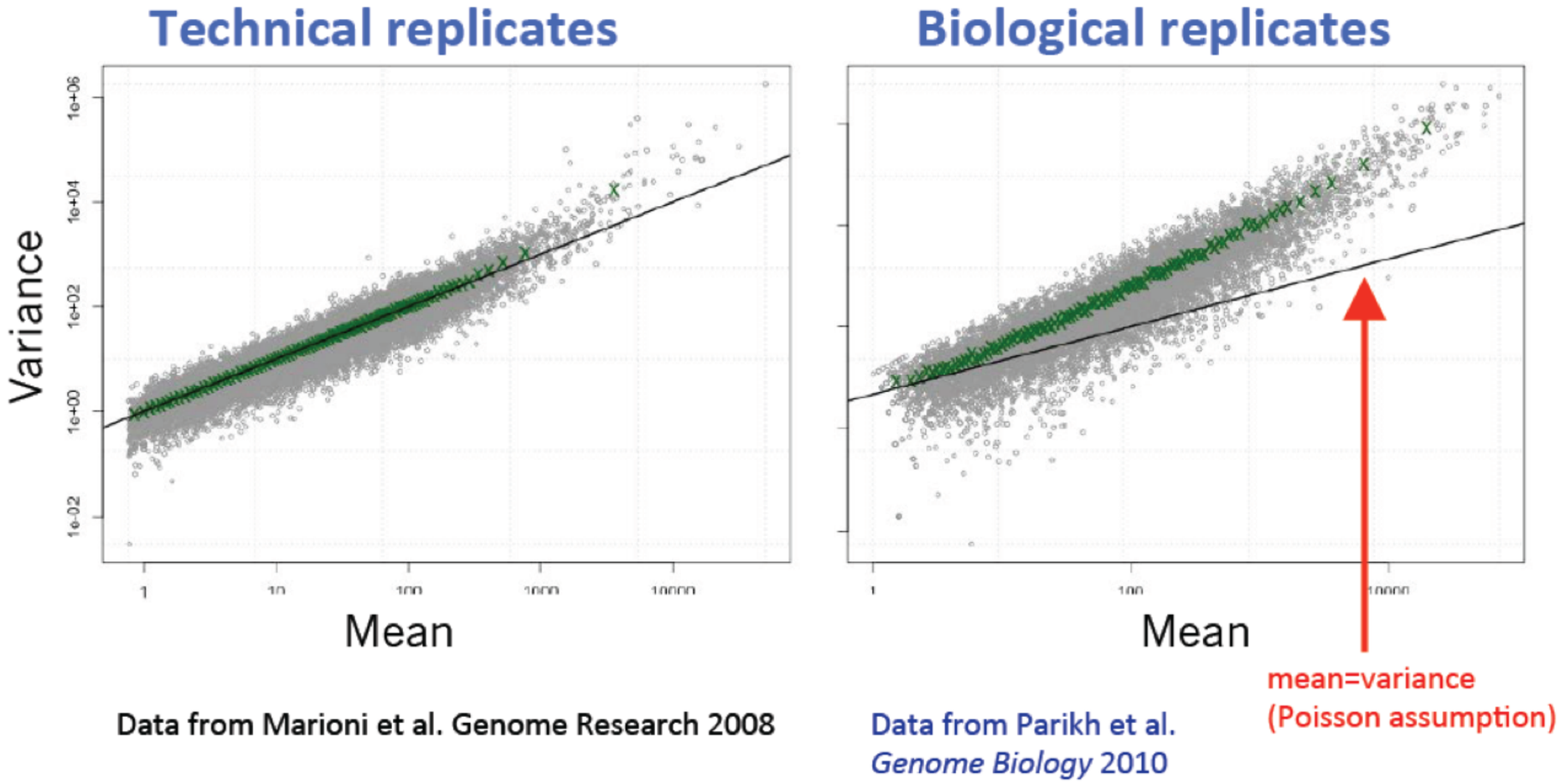
▶ Code



# Overdispersion

Poisson OK for technical replicates, but **does not capture biological variability**



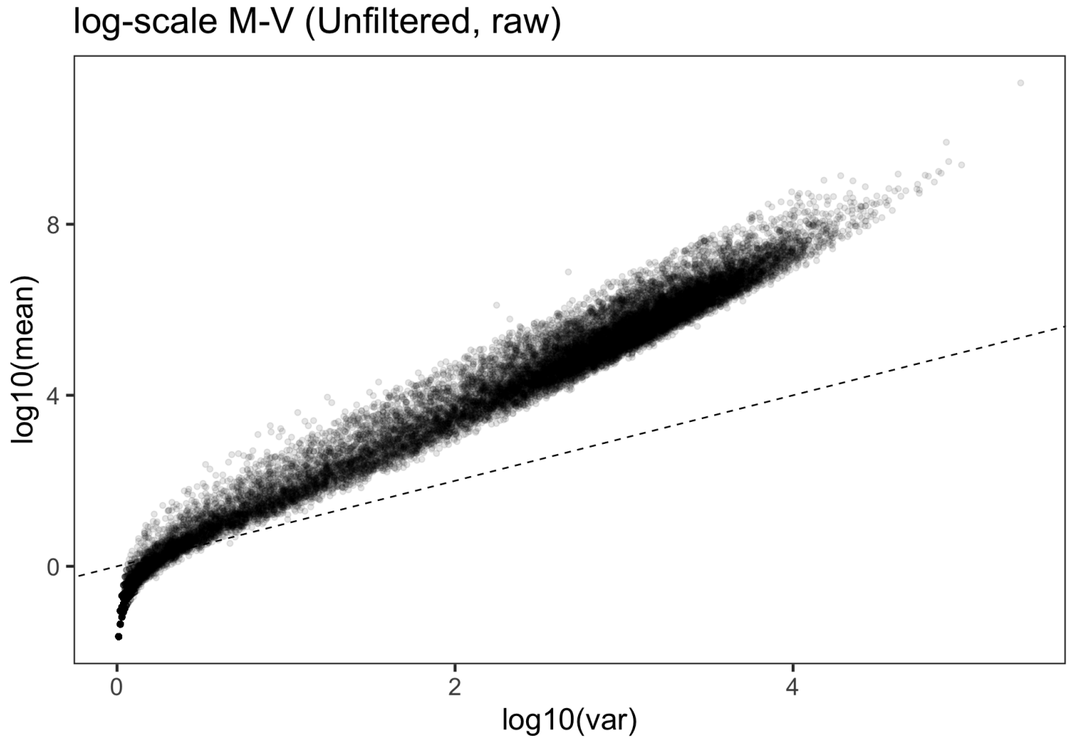

Data from Marioni et al. Genome Research 2008

Data from Parikh et al. *Genome Biology* 2010

# Impact of heteroskedasticity

- OLS: assume all errors have the same variance (within gene)
- If not true, higher variance observations get more weight in minimization of error than they should (since less precise)
  - Standard errors of parameter estimates will be poor estimates
  - Recall: $t = \frac{\hat{\beta}}{se(\hat{\beta})}$
  - …So p-values will also be wrong - in case of positive relationship, too small

▶ Code

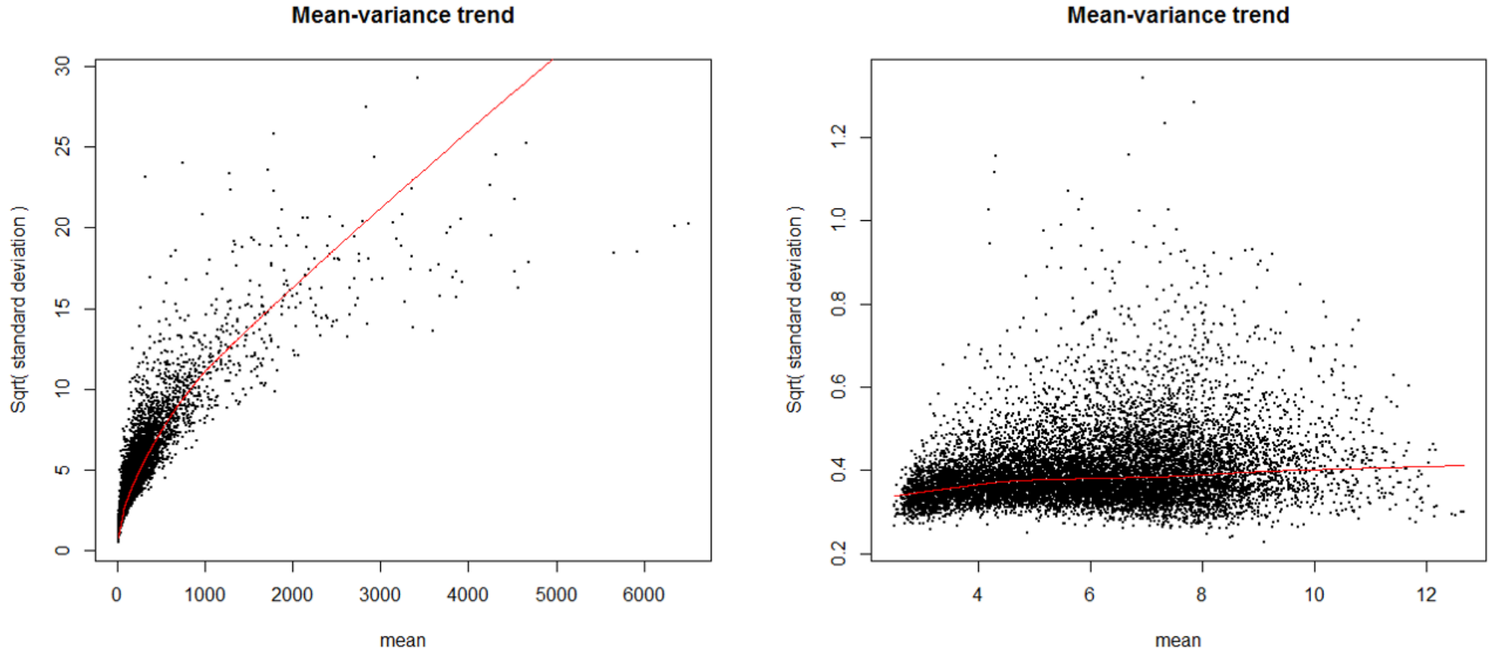


# Options for DE analysis on counts

- **Summary of the problem**: Count data is expected to violate both normality and constant variance assumptions
- Even microarray data usually has some mean-variance relation!
- Possibilities for coping:
  - Use a non-parametric test (e.g. SAMseq – based on Wilcoxon)
  - Make adjustments and model as usual
  - Use a model specific for count data

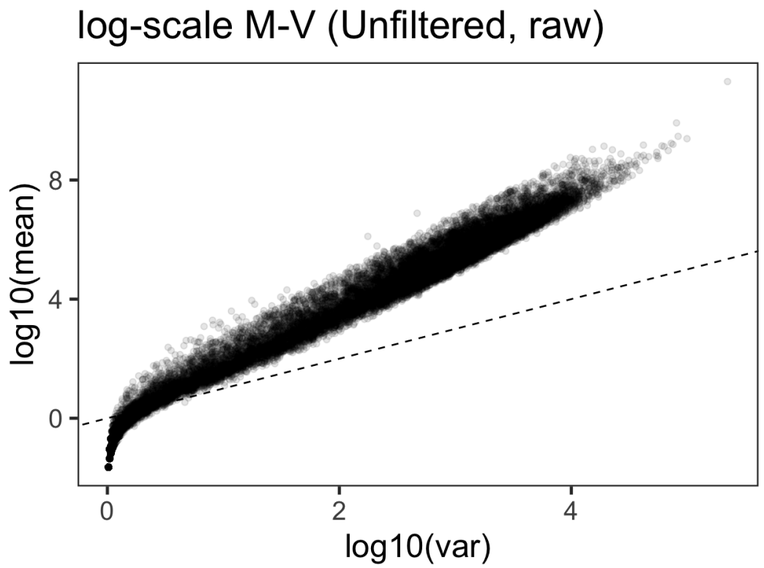
# Make adjustments & model as usual: transformation

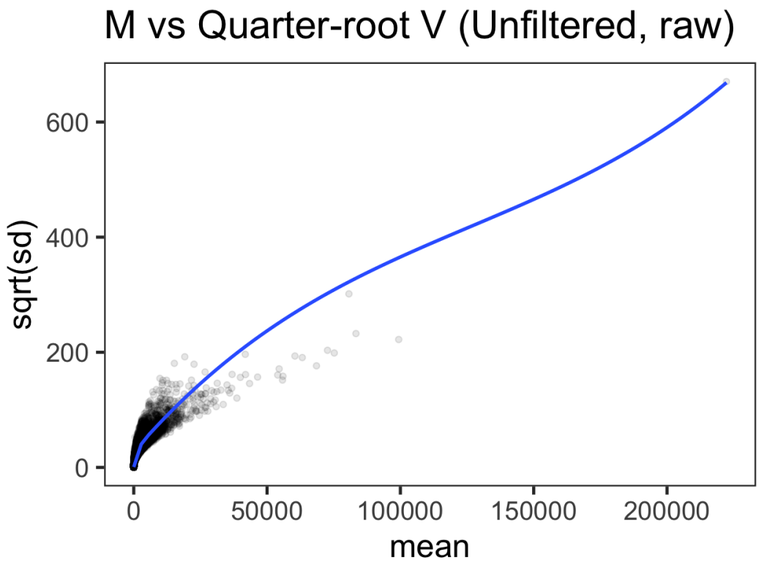
- For microarray data, taking logs is often deemed sufficient to reduce M-V trends
- We'll use plots like this which are mean vs $\sqrt{sd}$ (quarter root variance) instead of mean vs variance (you'll see why later on)
- Behaviour of Nrl microarray data set (raw on left, log-tranformed on right):

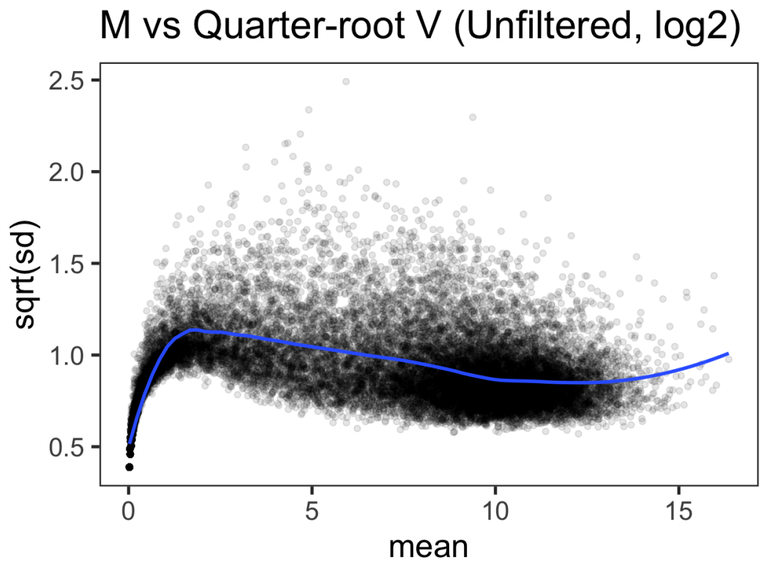


# Chd8 data & effect of log transform

▶ Code



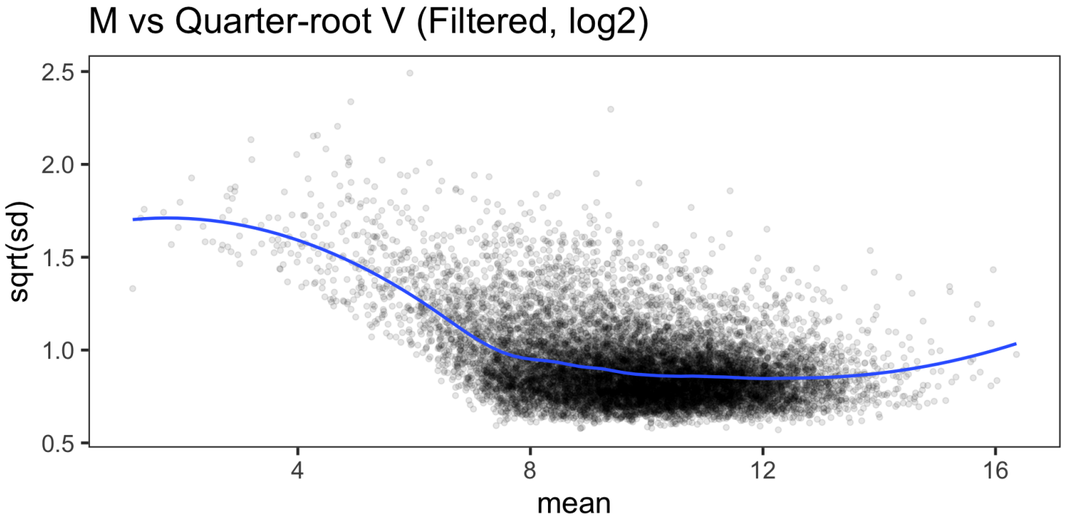



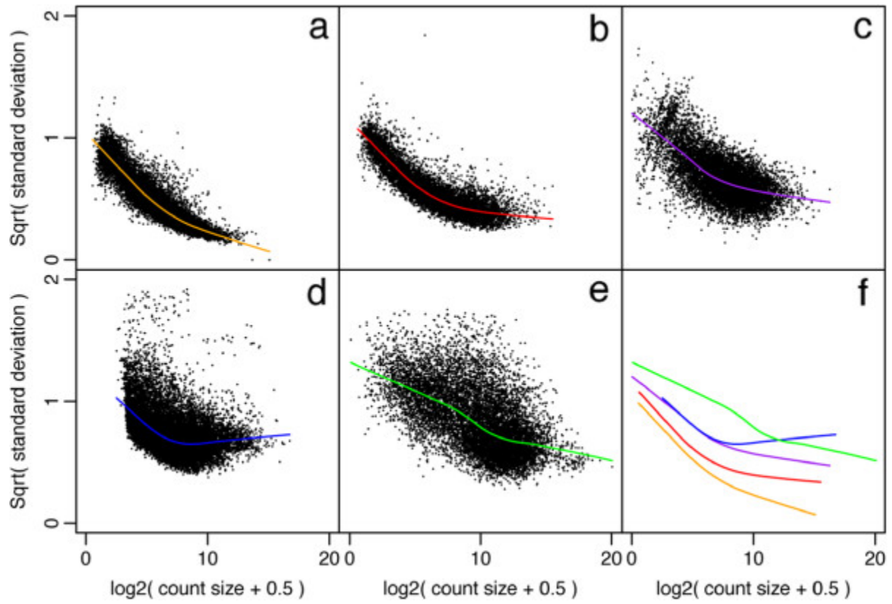

**For RNA-seq data, log-transformation doesn't reliably improve the trends**

# Mean variance trends in various RNA-seq datasets

▶ Code




Chd8 dataset (Filtered to remove lowly expressed genes, log2-transformed)




Panels (a)-(e) represent datasets with increasing expected biological variability

Source: Law et al. 2014

# One option: Voom

## Mean-variance modelling at the observational level

- Falls under the category *"Make adjustments and model as usual"*
- Specifically, adjustment to regular `lm` to account for M-V relationship + `limma`

**Key ideas of Voom:**

1. heteroskedasticity leads to higher variance observations getting more weight in minimization of error than they should
2. modeling the mean-variance relationship is more important than getting the probability distribution exactly right (i.e. don't bother with distributions like Poisson, Binomial, etc that lead to more complicated likelihoods)

- Proposed in "voom: precision weights unlock linear model analysis tools for RNA-seq read counts" by Law et al. (2014)

# Voom implementation

- Input:
  1. **raw counts** (required to estimate M-V relationship), but modeling is done on log-transformed CPM values ($(log_2(CPM + 0.5)$ to be precise)
  2. design matrix
- Output: precision weights and moderated $t$-statistics
- Implemented in `limma::voom()` function

# Voom steps

1. Fit linear model to $log_2(CPM_{ig} + 0.5)$ values (samples $i$) for each gene $g$
2. Extract the fitted quarter-root error variance estimates $s_g^{1/2} = \sqrt{sd(\hat{\varepsilon}_{ig})}$
3. Fit a smoothed line $\hat{f}$ to the trend between mean log counts and $s_g^{1/2}$ using lowess (locally weighted regression)
4. Use the fitted lowess curve to estimate **precision weights**: $w_{ig} = \frac{1}{\hat{f}(\hat{c}_{ig})^4}$ where $\hat{c}_{ig}$ are the $log_2$ *fitted* counts (estimated from model in step 1)
5. Fit linear model to $log_2(CPM_{ig} + 0.5)$ values using **precision weights** $w_{ig}$
6. Compute moderated $t$-statistics as before (using `eBayes` from `limma`)

# Voom illustration

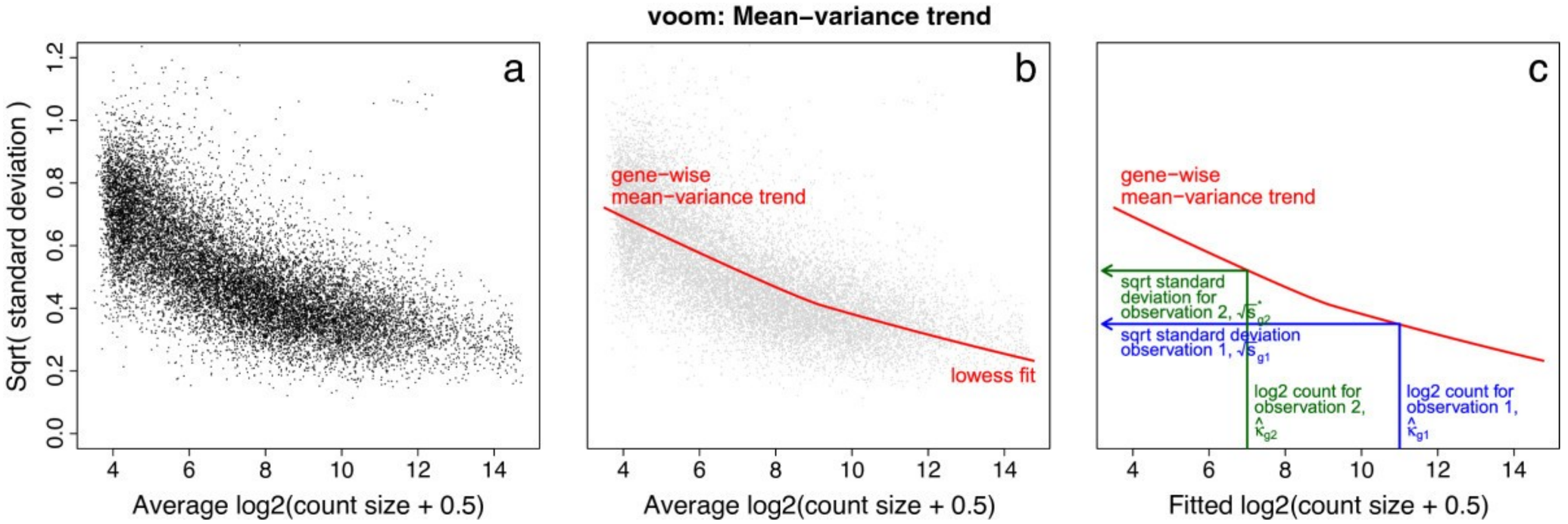
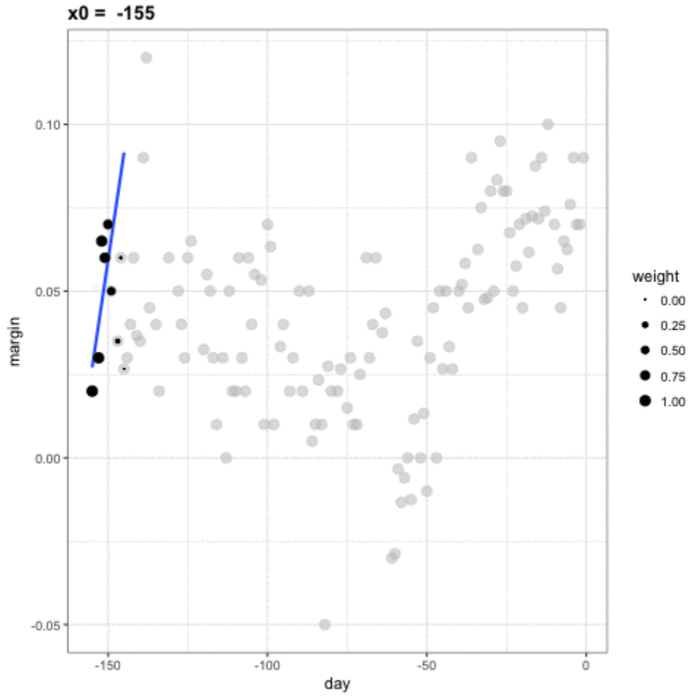



Figure 2, Law et. al, 2014

$$w_{ig} = \frac{1}{\hat{f}(\hat{c}_{ig})^4} = \frac{1}{(\sqrt{s_{ig}})^4} = \frac{1}{s_{ig}^2}$$

# lowess

- locally weighted regression fits a smooth curve to approximate the relationship between independent & dependent variables
- Each smoothed value is given by a weighted linear least squares regression over the **span** (a neighborhood of the independent variable)
- Smoothing span is adjustable
- Generalization to locally weighted polynomial regression & inclusion of multiple independent variables: `loess`




GIF source: “Intro to Data Science” by Irizarry

# Why quarter-root variance?

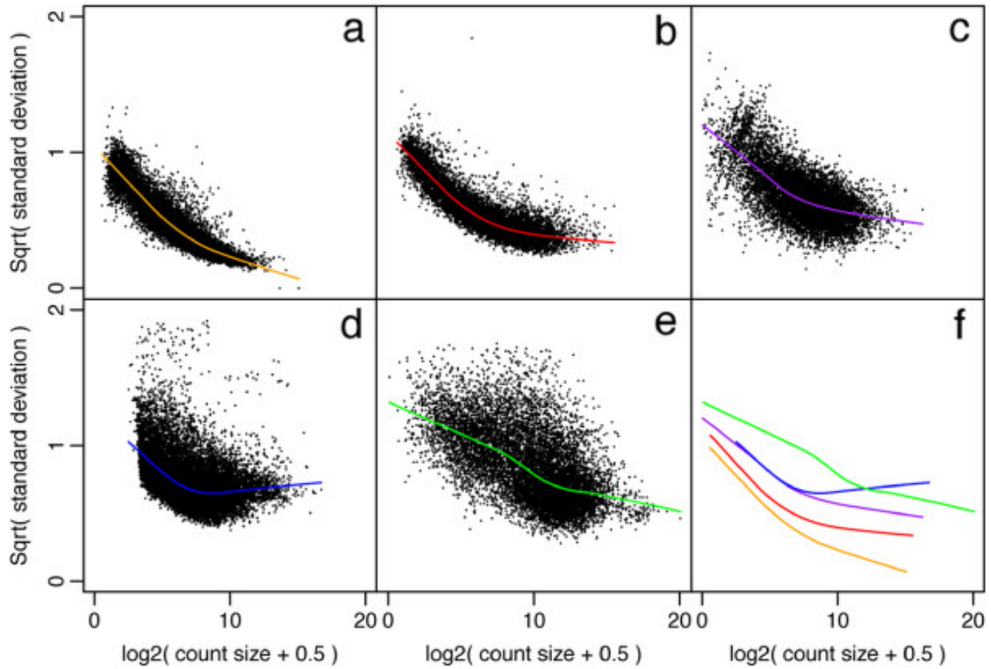
- The **coefficient of variation** ($CV = \frac{\sigma}{\mu}$) for RNA-seq counts is roughly $\sqrt{\frac{1}{\lambda} + \phi}$
  - $\lambda$ : expected size of count; arises from technical variability associated with sequencing and gradually decreases with increasing count size
  - $\phi$ : measure of biological variation (*overdispersion*); roughly constant

- Standard deviation of $log_2(CPM)$ is approximately equal to CV of the counts (by Taylor's theorem)

$$sd(log_2(CPM)) \approx \sqrt{\frac{1}{\lambda} + \phi}$$

# Why quarter-root variance?

> **Note**
>
> Coefficient of variation (CV) of RNA-seq counts should be a decreasing function of count size for small to moderate counts, and asymptote to a value that depends on biological variability




Law et al. 2014: Panels (a)-(e) represent datasets with increasing expected biological variability

Square root of standard deviation used as distribution is more symmetric (i.e. less skewed)

# What do we do with these ‘precision weights’?

How can we actually incorporate these precision weights in the regression fit?

# Weighted least squares (WLS) regression

- OLS: $\hat{\boldsymbol{\beta}}_g = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{y}_g$
- WLS: $\hat{\boldsymbol{\beta}}_g = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{W}_g\boldsymbol{X})^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{W}_g\boldsymbol{y}_g$, where $\boldsymbol{W}_g$ is a diagonal matrix of weights for gene $g$
- **Intuition**: in minimizing the RSS, we put less weight on data points that are less precise:

$$\hat{\boldsymbol{\beta}}_g = argmin_{\beta_{g1},\ldots,\beta_{gp}} \left( \sum_{i=1}^{n} w_{ig}(x_{i1}\beta_{g1} + \ldots + x_{ip}\beta_{gp} - y_{ig})^2 \right)$$

- Optimal weights to correct for heteroskedasticity: inverse variance[1]

1. Note: parameter estimates $\hat{\boldsymbol{\beta}}$ assume weights (variances) are known

# limma-voom

- **limma-voom** is the application of `limma` to $log_2(CPM + 0.5)$ values, with inverse variance observational weights *estimated from the M-V trend*
- This alleviates the problem of heteroskedasticity and (hopefully) improves estimates of residual standard error
- Gene-specific variance estimates are 'shrunken' to borrow information across all genes:

$$\tilde{s}_g^2 = \frac{d_0 s_0^2 + d s_g^2}{d_0 + d}$$

- Note that $s_g^2$ estimates are affected by voom weights
  - recall that $s_g^2$ is the sum of squared residuals $\frac{1}{n-p}\hat{\boldsymbol{\epsilon}}_g^T \hat{\boldsymbol{\epsilon}}_g$
  - under WLS $\hat{\boldsymbol{\epsilon}}_g = \boldsymbol{y}_g - \boldsymbol{X}\hat{\boldsymbol{\beta}}_g = \boldsymbol{y}_g - \boldsymbol{X}(\boldsymbol{X}^T \boldsymbol{W}_g \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{W}_g \boldsymbol{y}_g$

# limma-voom, continued

- Moderated statistics are then calculated using the shrunken gene-specific variance estimates:
  - recall that under OLS, is the diagonal element of
  - under WLS, is the diagonal element of
- Recall:
  - Degrees of freedom for moderated statistic:
  - If is large compared to , moderated statistics have a bigger effect compared to using regular statistics (i.e. in general, shrinkage matters more for small sample sizes)

# Differential expression analysis on Chd8 data

- Recall: Our **additive** model for each gene to test for Group (Chd8 mutant vs WT) effect, and adjust for:
  - Sex (M vs F)
  - DPC (days post conception, 5 levels)

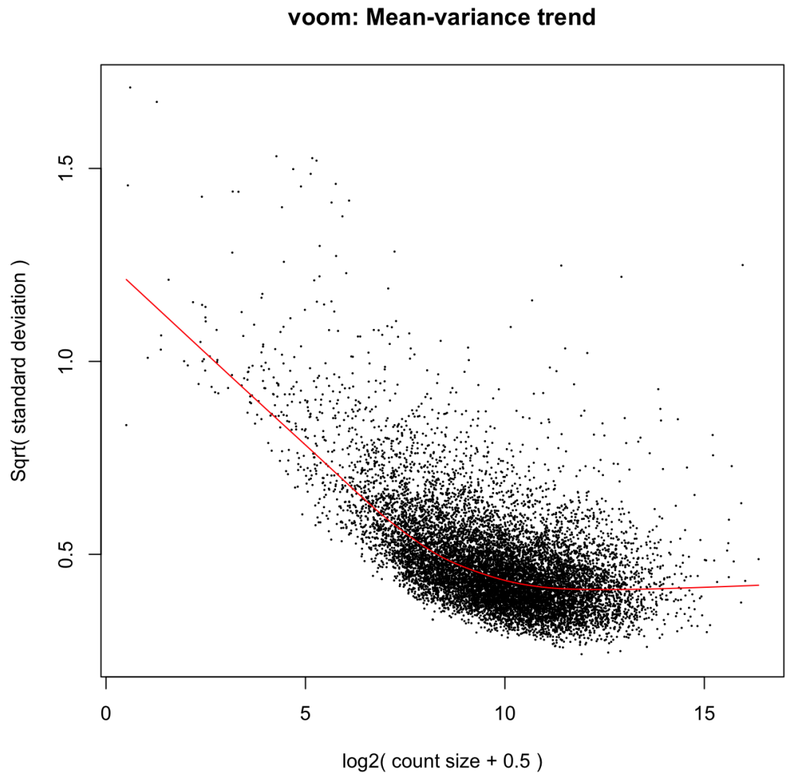
where

- Our model has degrees of freedom
- We will focus on the null hypothesis of the **main effect** of Group

# limma-voom in action

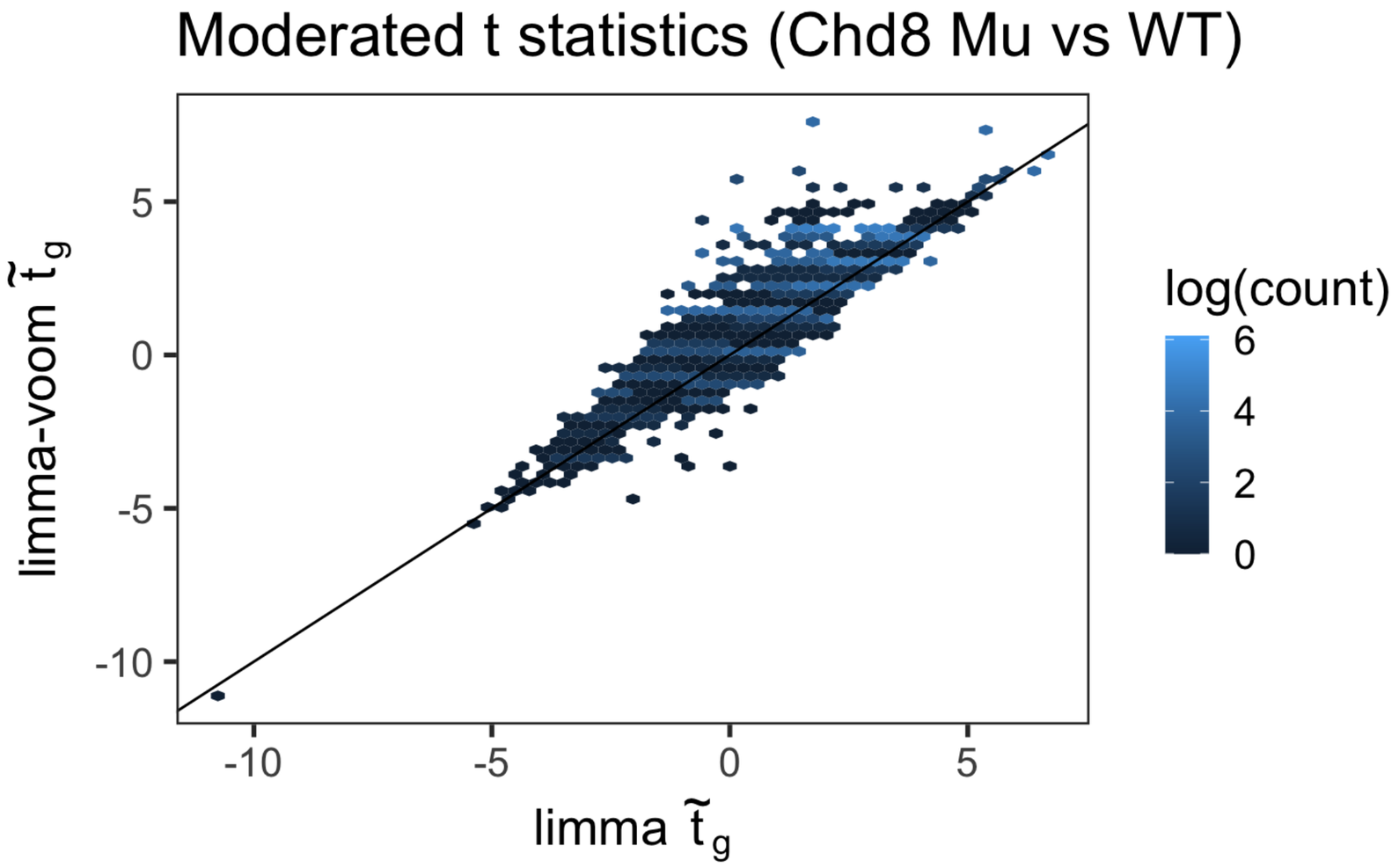
```r
# estimate voom weights; plot M-V trend
vw <- voom(assays(sumexp)$counts,
           design = model.matrix( ~ Sex + Group + DPC, data = colData(sumexp)),
           plot = TRUE, span = 0.5)
```

```r
# run limma with voom weights
lvfit <- lmFit(vw, model.matrix(~ Sex + Group + DPC, data = colData(sumexp)))
lvfit <- eBayes(lvfit)
```



# limma-voom vs limma

▶ Code



# Another option: limma-trend

## Limma-trend uses the M-V relationship at the gene level, whereas voom uses observational level trends (Law et. al, 2014)

- Gene-wise variances are shrunken toward a **global M-V trend**, instead of toward a constant pooled variance:

- Notice the subscript on ! The prior variance is different for each gene (unlike in regular limma)
- Based on the M-V trend, is (typically) higher for lowly expressed genes

# limma-trend vs voom?

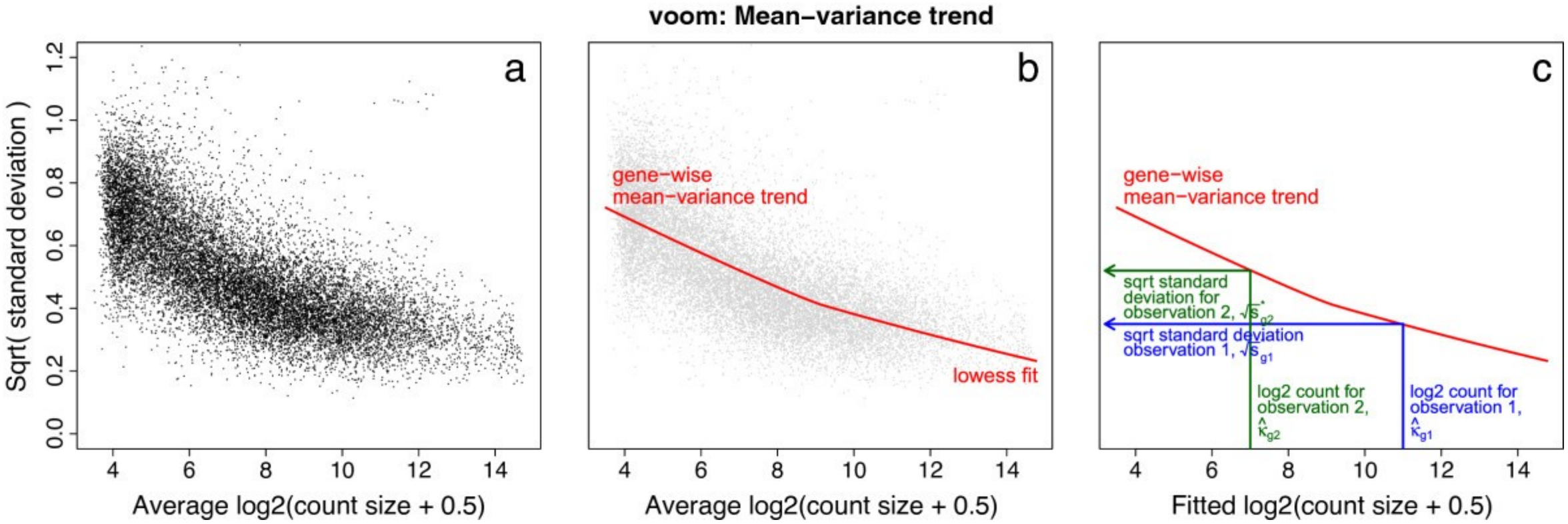



Figure 2, Law et. al, 2014

# limma-trend in action

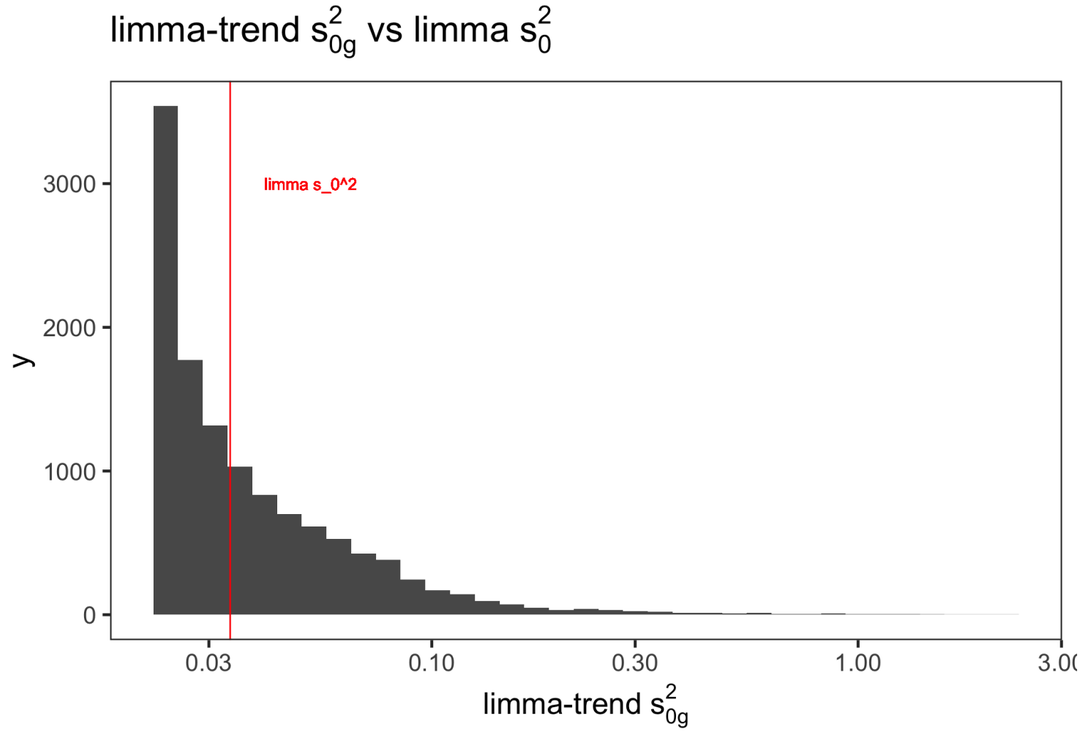
```
mm <- model.matrix(~ Sex + Group + DPC,
                   data = colData(sumexp))
ltfit <- lmFit(cpm(assays(sumexp)$counts,
                   log = TRUE),
                 design =  mm)
ltfit <- eBayes(ltfit, trend = TRUE)

# limma-trend s^2_{0g}
str(ltfit$s2.prior)
```

```
 Named num [1:12158] 0.0287 0.051 0.0274 0.023 0.0268 ...
 - attr(*, "names")= chr [1:12158] "0610007P14Rik"
"0610009B22Rik" "0610009O20Rik" "0610010F05Rik" ...
```

```
# regular limma s^2_0
str(lfit$s2.prior)
```

```
num 0.0337
```

▶ Code



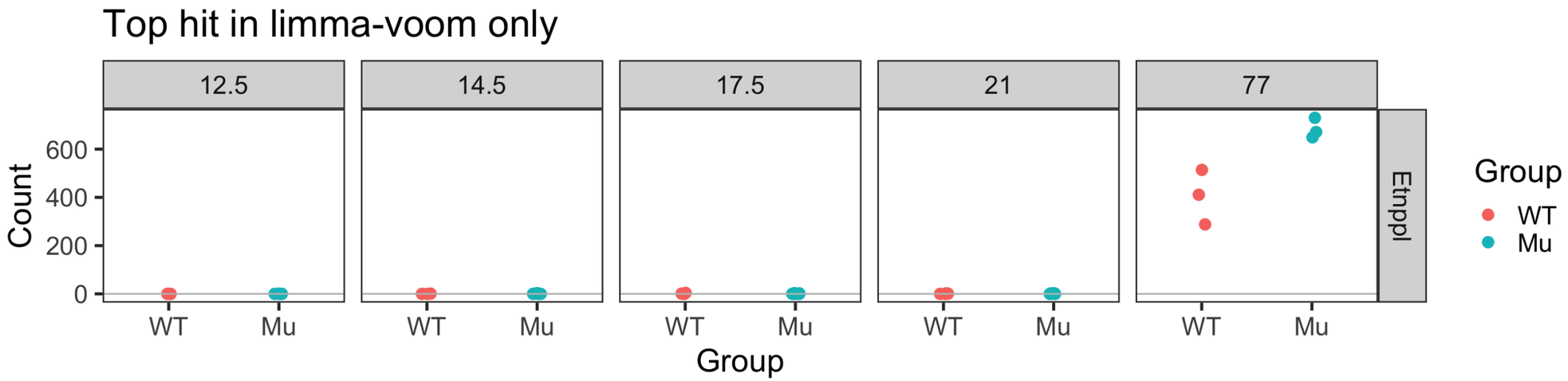
# Nuances for limma-trend and limma-voom

- If M-V relationship is flat, limma-voom and limma-trend have practically no effect
  - for limma-voom, weights will be all equal
  - for limma-trend, will be constant across genes
- Even if M-V isn’t flat, impact is most prominent in lowly expressed genes

# limma-voom ‘false positives’?

One of the top DE genes by Group according to voom (but not other methods):

▶ Code




- Why does this happen?
  - Voom weighting causes very low expression values to have little effect on model fit
  - Weights for this gene are about 30-40x higher for DPC 77 observations
- Whether this is a false positive is a matter of opinion, but lesson is: *always look at the data*

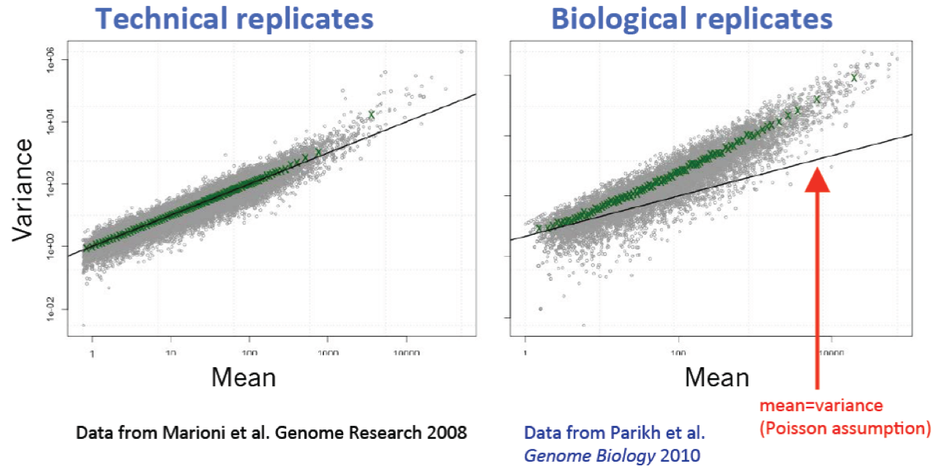
# Alternative option: use count models



- Methods: edgeR, DESeq2
- Both assume counts have underlying *Negative Binomial distribution* and fit **generalized linear models**
  - **Generalized linear models** (GLM) are a generalization of OLS that allows for response variables that have error distribution models other than a normal distribution
  - No closed-form solutions (iterative estimation)
- Still fit models gene-by-gene as we've discussed so far
- Many similarities with limma: empirical Bayes-based moderation of parameters and addressing the M-V trend

# Why Negative Binomial distribution?

- Negative Binomial is also known as a **Poisson-Gamma** mixture
  - i.e. A Poisson with a rate parameter that is Gamma-distributed (instead of fixed)
  - The Gamma distribution on means captures the biological variance (overdispersion) that can't be accommodated by Poisson alone
- "Overdispersed Poisson" (variance mean)
- **Key problem**: estimating dispersion from small datasets is tricky



# Negative Binomial GLM

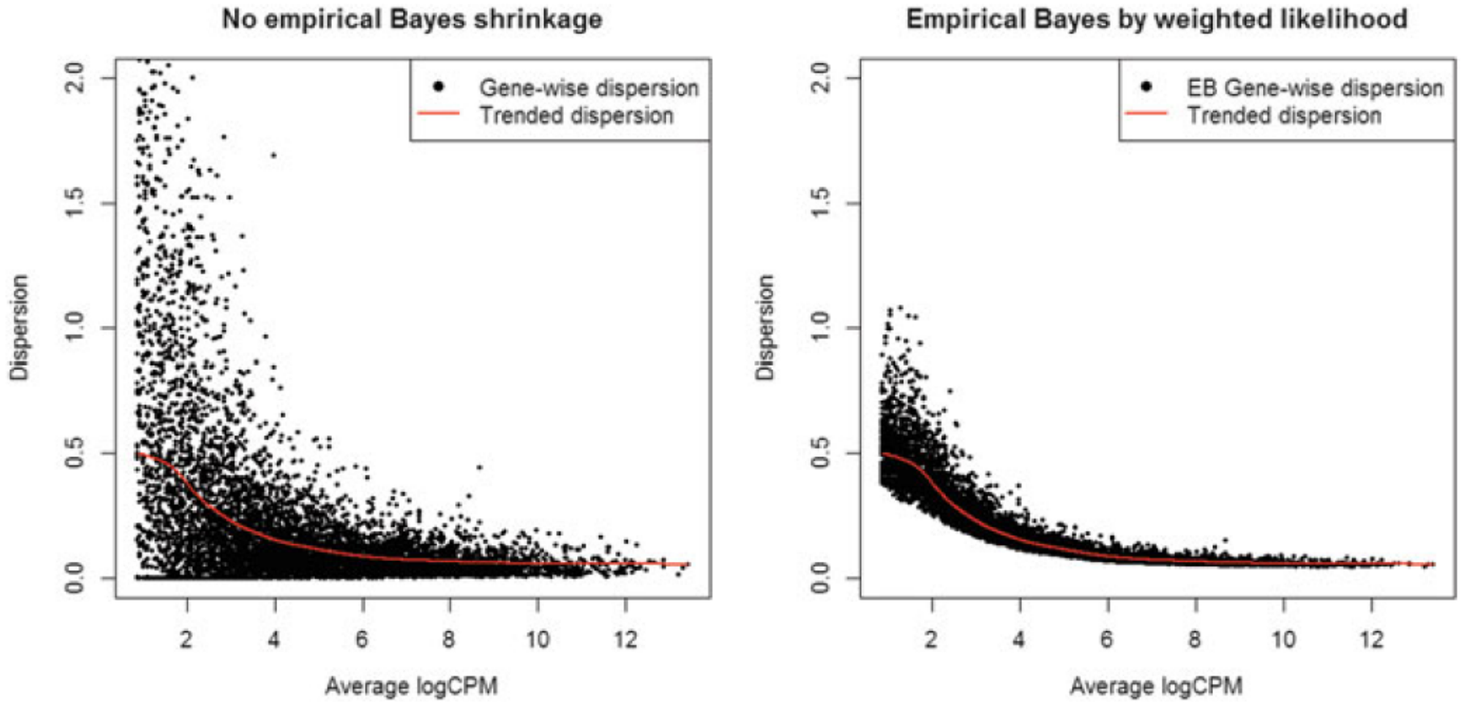
- Gene-specific variance under NB:
  - is the **dispersion** for gene
  - if , get Poisson!
- We can perform inference about using GLM (e.g. using likelihood ratio tests)

- To do so, we need to treat as known (*so first need to estimate it*)

> **Important**
>
> Estimation of dispersion is the main issue addressed by methods like `edgeR` and `DEseq2`

# Disperson estimation

- One option is to assume is a set parametric function of the mean (e.g. quadratic)
- More flexible approach is to use empirical Bayes techniques: dispersion is gene-specific but moderated toward the observed trend with the mean




source: Statistical Analysis of Next Generation Sequencing Data, by Chen et al.

# DESeq2 vs edgeR

- These methods are very similar overall
- Major differences between the methods lie in how they filter low-count genes, estimate prior degrees of freedom, deal with outliers in dispersion estimation, and moderate dispersion of genes with high within-group variance or low counts
  - Also slight differences in specific types of hypothesis tests (quasi-likelihood in edgeR and Wald test in DESeq2)
- Many of these choices can be altered by changing default parameter settings in both methods (see user manuals)

# DESeq2 vs edgeR

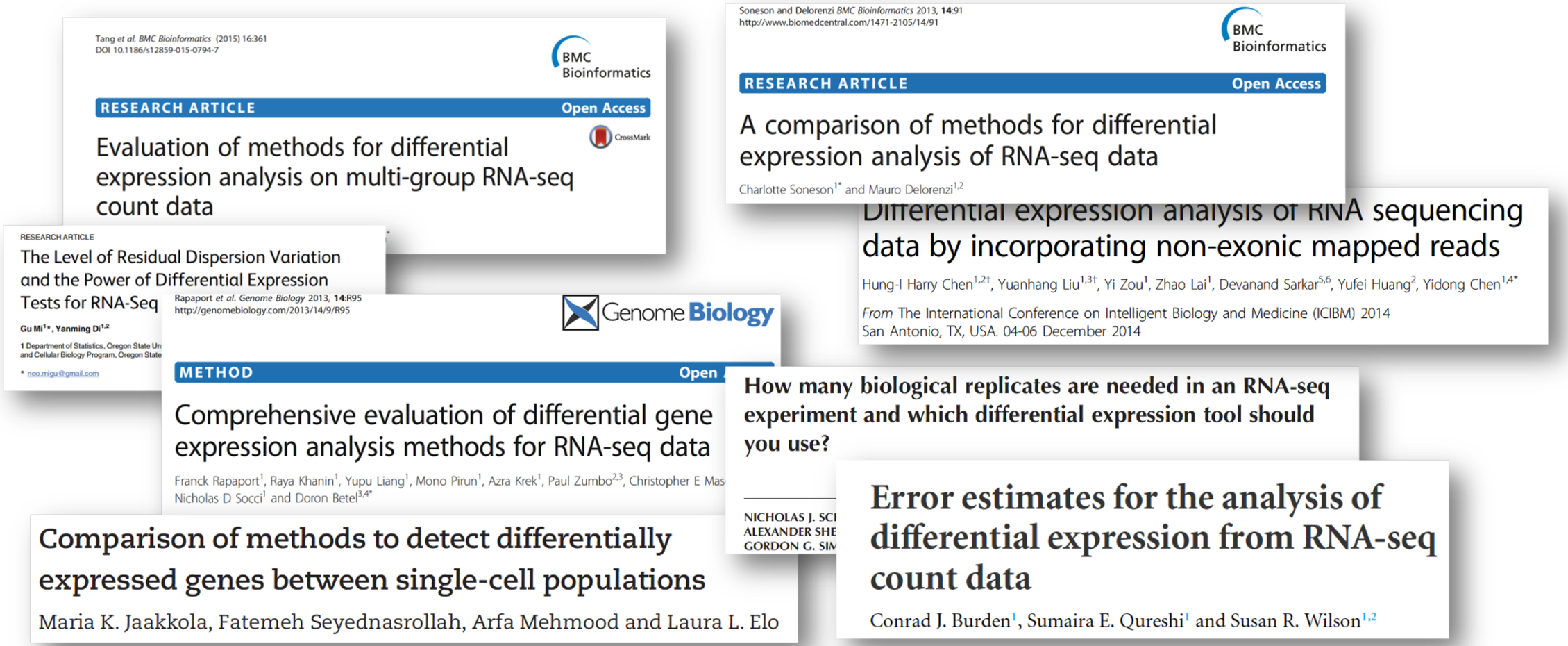
## edgeR

```
dge <- DGEList(counts = assays(sumexp)$counts,
               samples = colData(sumexp))
dge <- calcNormFactors(dge)
dge <- estimateDisp(dge,
  design = model.matrix(~ Sex + Group + DPC,
                        data = sumexp$samples),
  robust = TRUE)

edgeR_fit <- glmQLFit(dge,
  design = model.matrix(~ Sex + Group + DPC,
                        data = sumexp$samples))
```

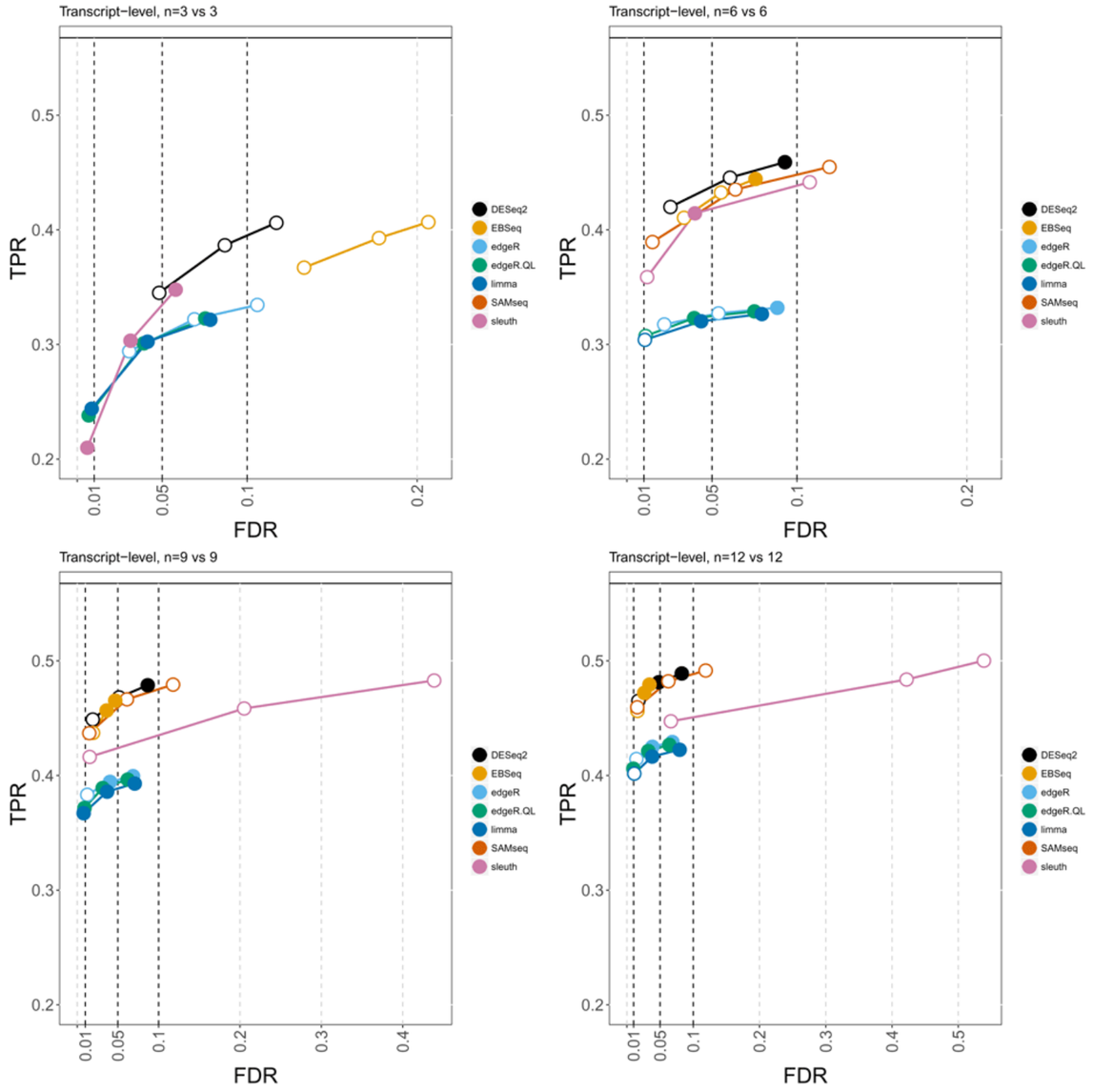
## DESeq2

```
dds <- DESeqDataSet(sumexp,
  design = model.matrix(~ Sex + Group + DPC,
  data = colData(sumexp)))
dds <- estimateSizeFactors(dds)
dds <- DESeq(dds)
```

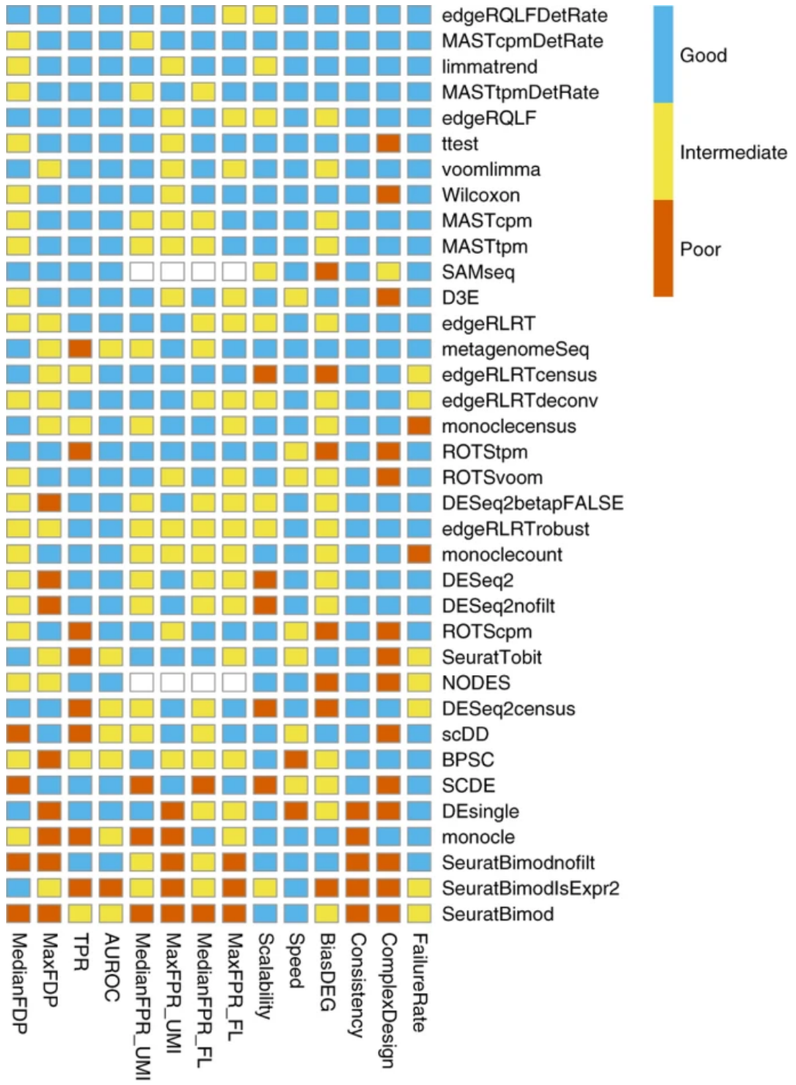
# How to choose a method?



# Example comparison 1: Love et al. (2018)



# Example comparison 2 (for single-cell RNA-seq)




Soneson & Robinson (2018)

# How to choose a method?

- No established gold standards
  - Simulations somewhat unsatisfying (depend on specific settings)
  - In real data, the truth is unknown

**The most popular and widely used methods tend to give similar results**

- **edgeR** and **DESeq2** are very similar in design
  - might be expected to work better for small sample sizes or low read depth
- **limma-trend** or **limma-voom** also sound choices
  - work equally well when library sizes don't vary much
  - might not do as well when sample size or depth is very low

# Comparing methods on the Chd8 dataset

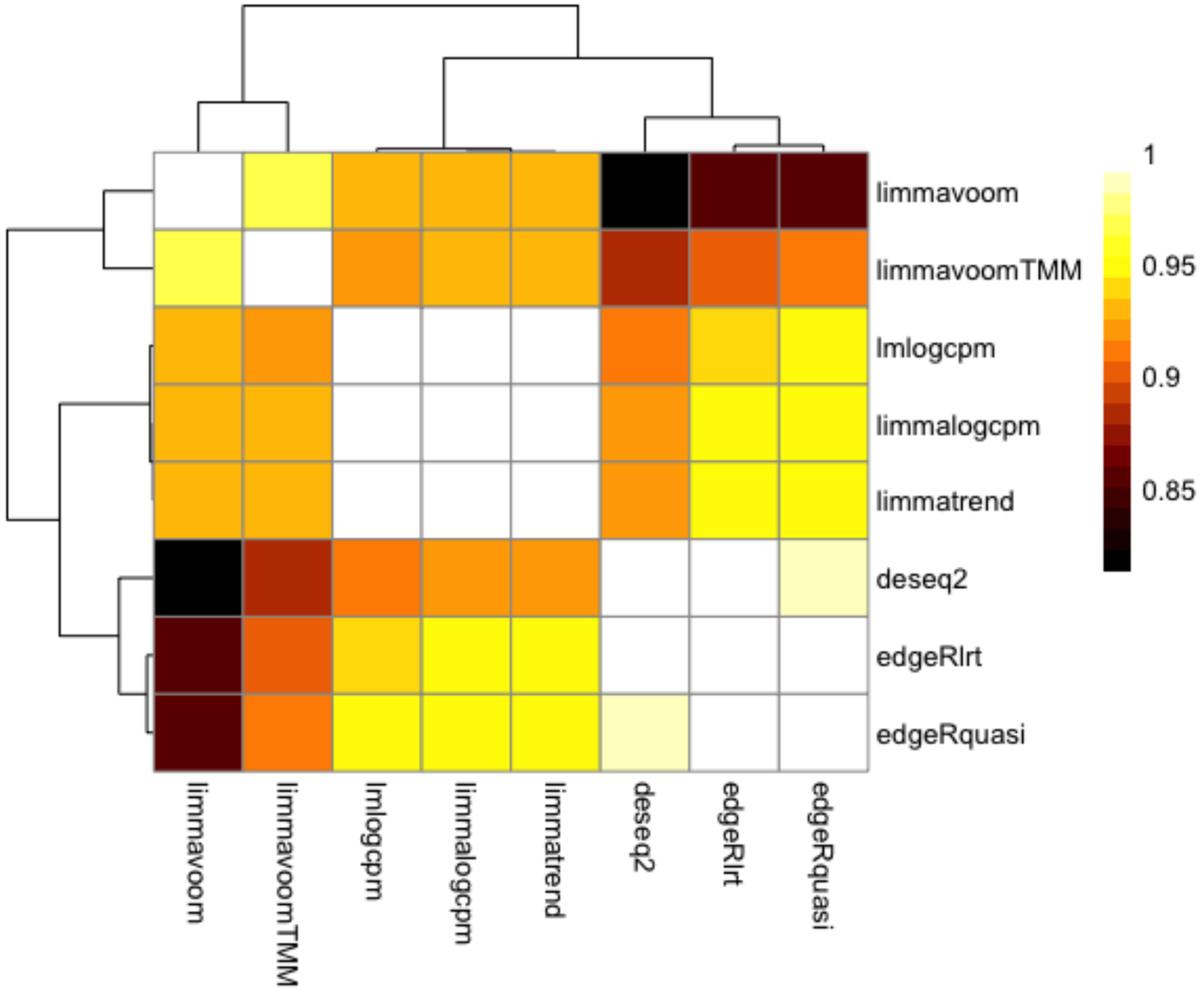
## tl;dr version: there isn't a big difference

Possible reasons why:

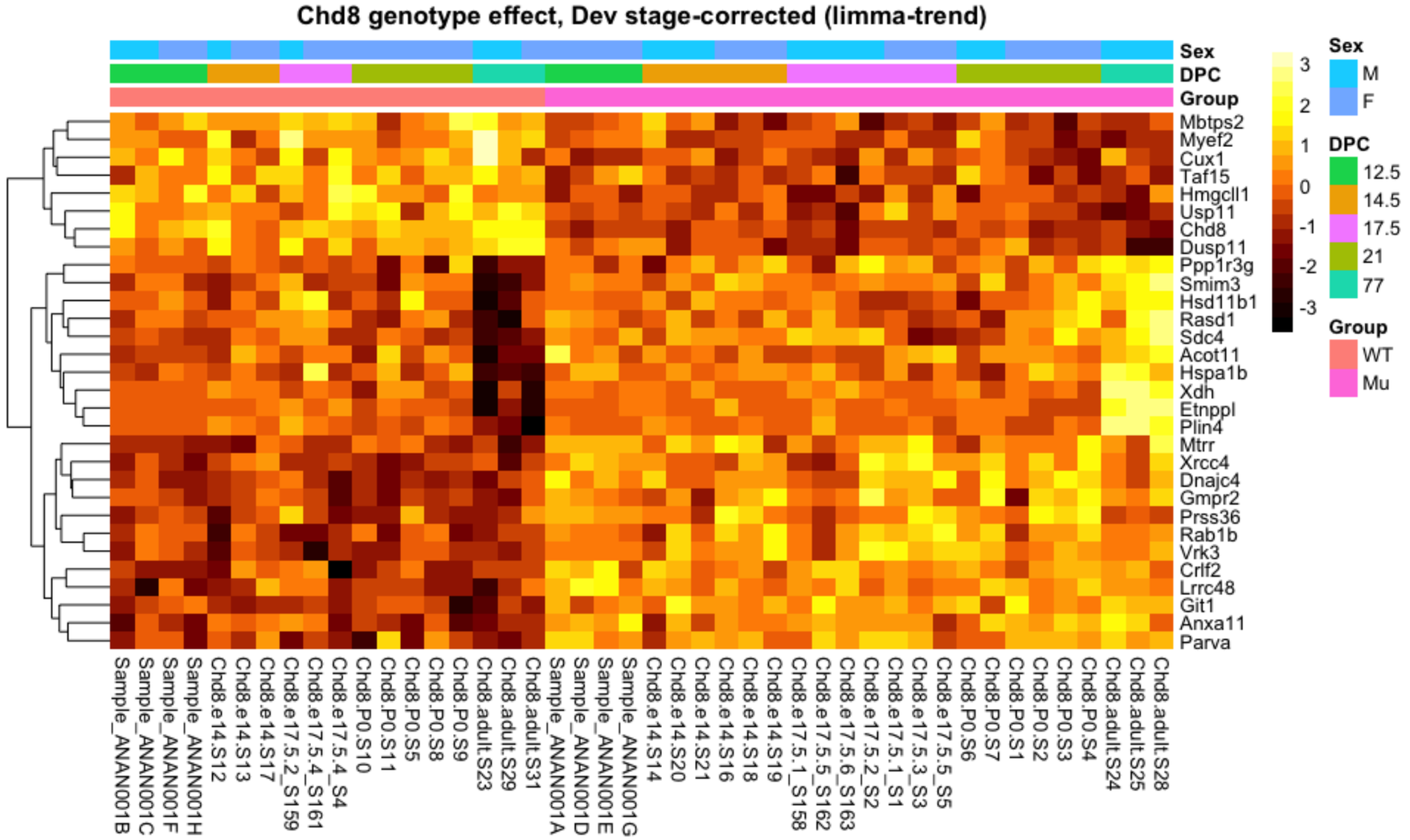
- methods have been converging in approach
- modeling count data directly with GLMs is more important for smaller samples sizes, lower read depth

Check out the comparisons in detail (including the results of `edgeR` and `DESeq2` in the companion notes

# Rank correlation of p-values for effect of Chd8 mutation



# Heatmap of top 30 genes by limma-trend, adjusted for DPC effect



# Additional resources

- Chapter 1: Generative Models for Discrete Data in Modern Statistics for Modern Biology by Holmes and Huber is a great review of count models
- Detailed comparison of these methods on the Chd8 dataset can be found here
- For all of the specific methods we discuss, refer to the Bioconductor pages (vignettes, reference manuals) for the most current and thorough details on implementation