Statistical Inference for RNA-seq - Part II

Keegan Korthauer

14 February 2022

with slide contributions from Paul Pavlidis

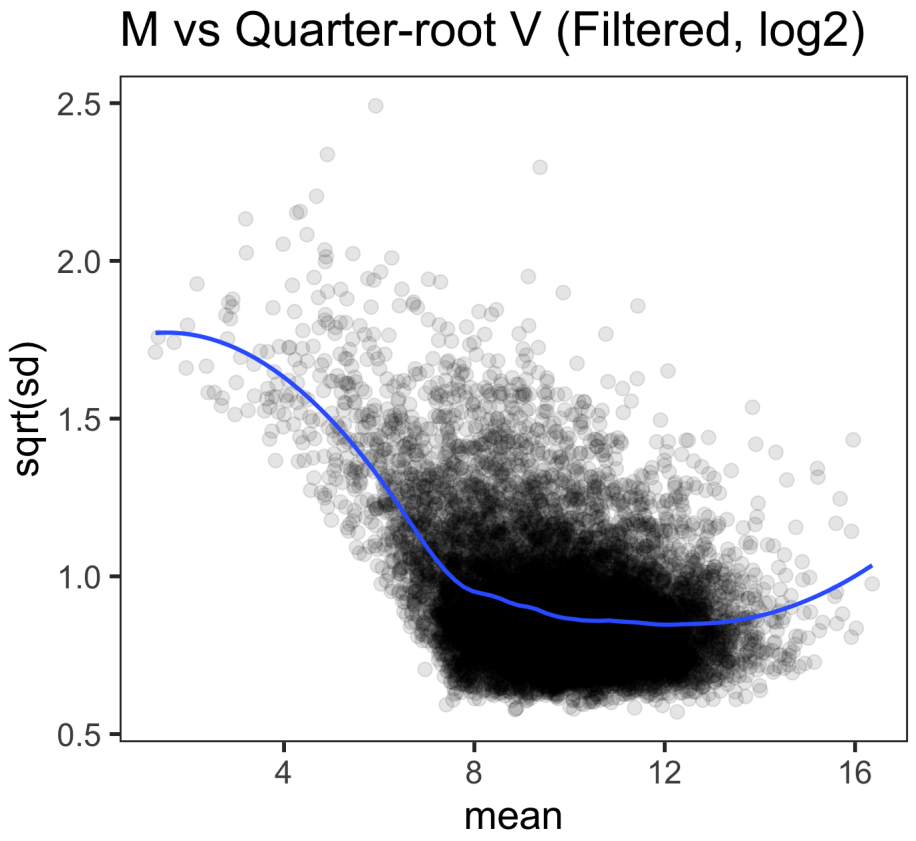
Learning objectives (lectures 11 and 12)

- 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

How do we handle these M-V relationships in our analysis?

Options we discussed last time:

- Use a non-parametric test
- Make adjustments and model as usual
- Use a model specific for count data



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 take into account the M-V relationship
- Key idea 1: heteroskedasticity leads to higher variance observations getting more weight in minimization of error than they should

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 take into account the M-V relationship
- Key idea 1: heteroskedasticity leads to higher variance observations getting more weight in minimization of error than they should
- Key idea 2: modeling the mean-variance relationship is more important than getting the probability distribution exactly right (i.e. don't bother with other distributions like Poisson, Binomial, etc)
- 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` package: `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)

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

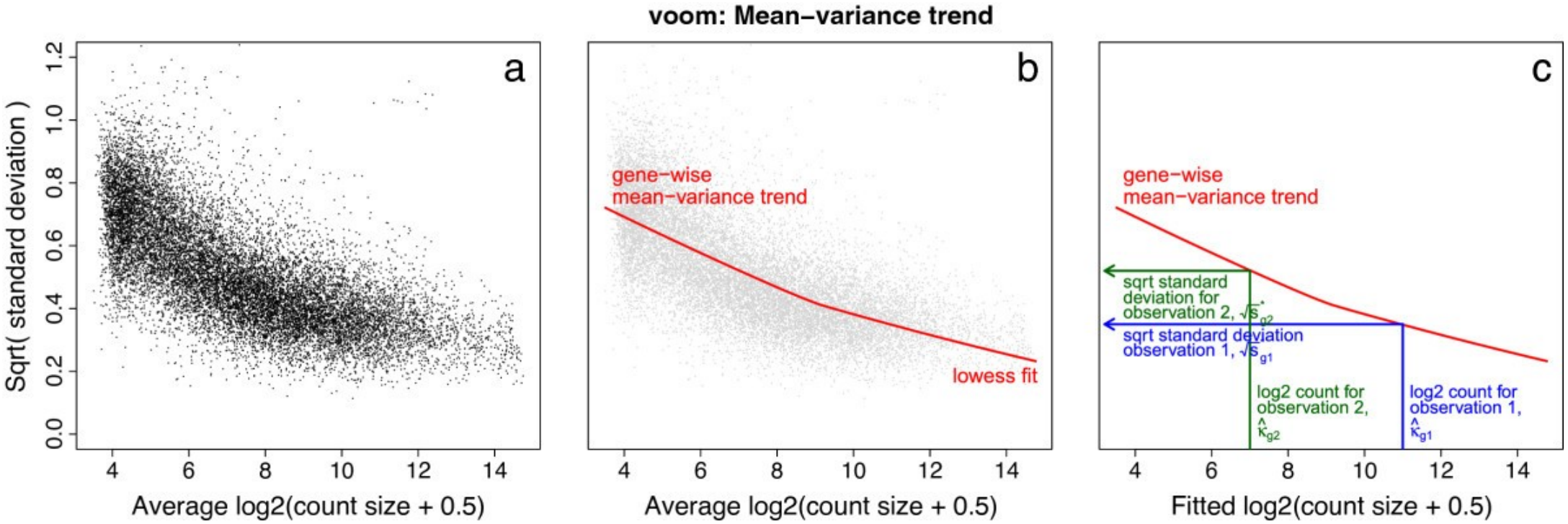


Figure 2, [Law et. al, 2014]

Voom illustration

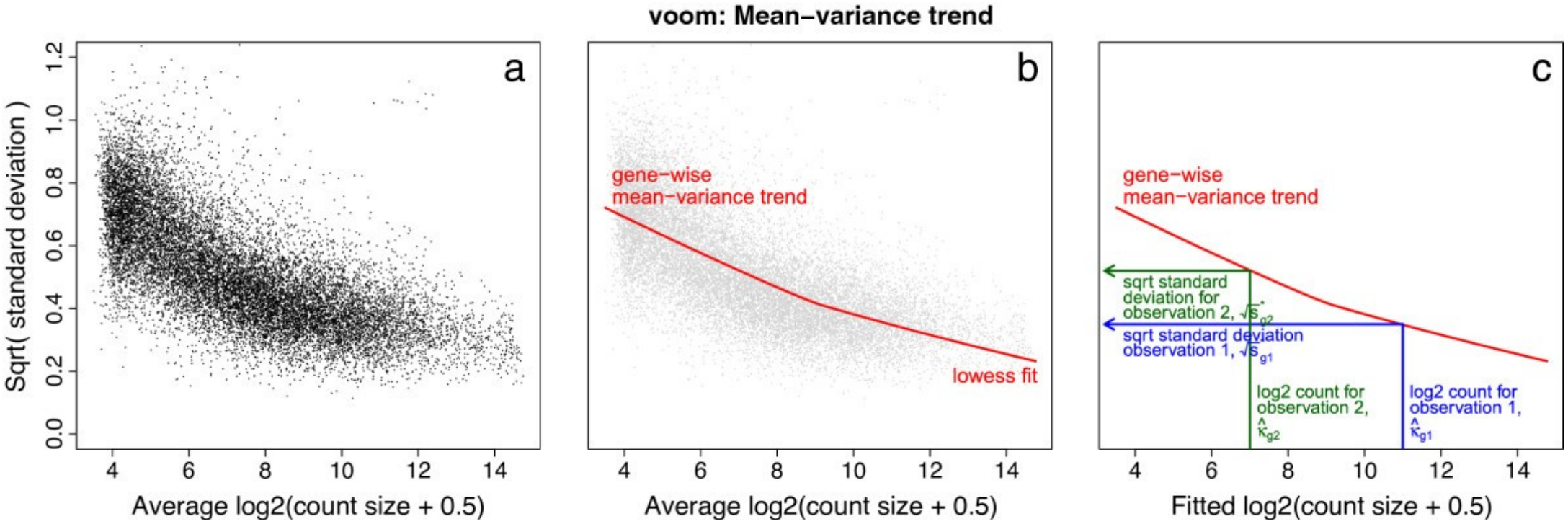


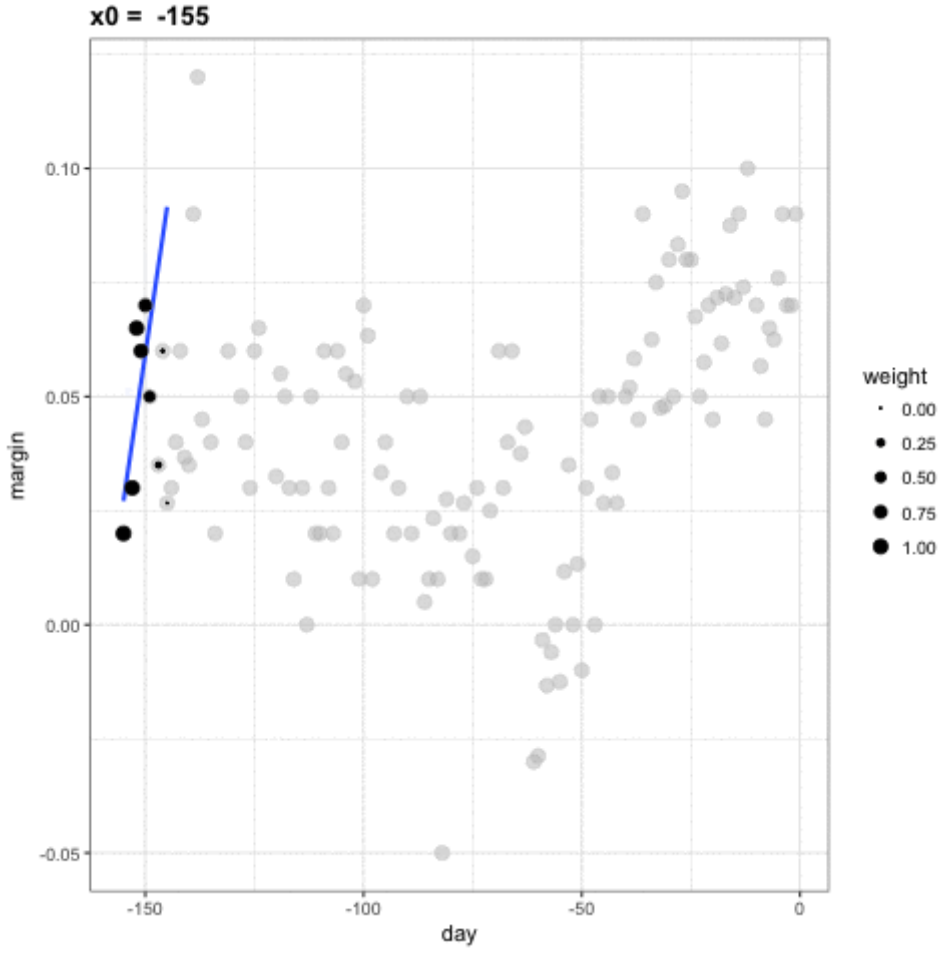
Figure 2, [Law et. al, 2014]

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

lowess

- locally weighted regression fits a smooth curve to approximate the relationship between independent and 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 and inclusion of multiple independent variables: `loess`

GIF source (not rendered in PDF):
"Introduction to Data Science" by Irizarry



Why quarter-root variance?

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

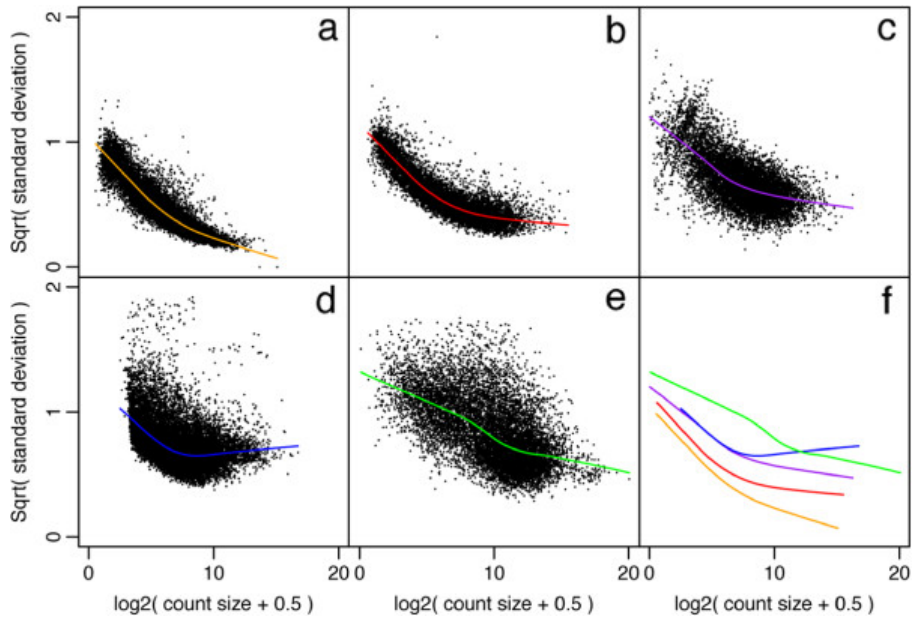
Why quarter-root variance?

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

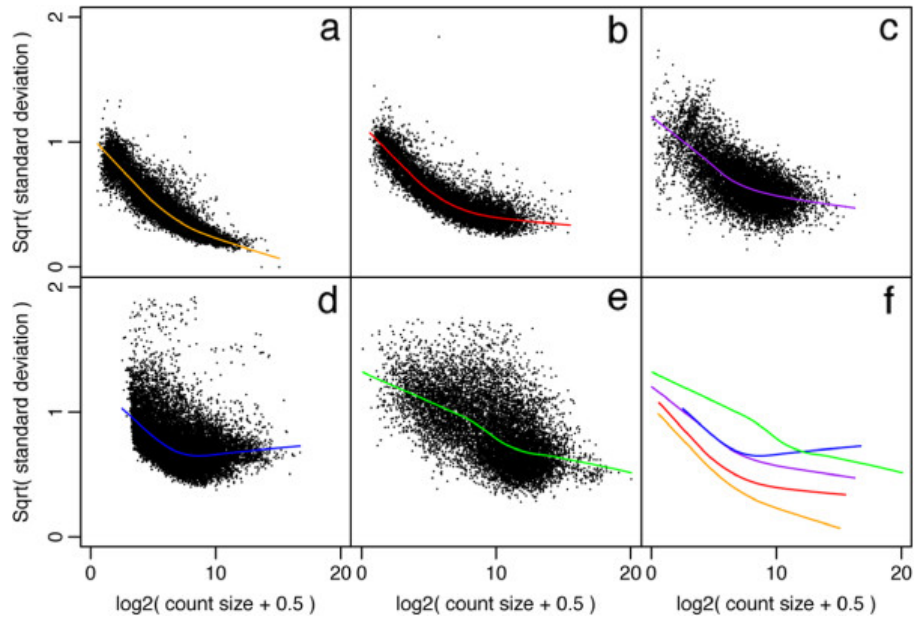
CV of RNA-seq counts should be a decreasing function of count size for small to moderate counts, and asymptote to a value that depends on biological variability



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

Why quarter-root variance?

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)

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

Weighted least squares (WLS) regression

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

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

Weighted least squares (WLS) regression

- OLS: $\hat{\boldsymbol{\beta}}_g = (\boldsymbol{X^T X})^{-1} \boldsymbol{X^T y}_g$
- WLS: $\hat{\boldsymbol{\beta}}_g = (\boldsymbol{X^T W}_g \boldsymbol{X})^{-1} \boldsymbol{X^T W}_g \boldsymbol{y}_g$, where $\boldsymbol{W}_g$ is a diagonal matrix of weights for gene g
- Intuition: in minimizing the sum of squared errors, 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 for this purpose: inverse variance
- Remedies heteroskedasticity
- Note: parameter estimates $\hat{\boldsymbol{\beta}}_g$ 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:

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

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 t statistics are then calculated using the shrunken gene-specific variance estimates: $\tilde{t}_g = \frac{\hat{\beta}_{ig}}{\tilde{s}_g \sqrt{v_{ii}}}$
  - recall that under OLS, v_{ii} is the i^{th} diagonal element of $\boldsymbol{(X^T X)^{-1}}$
  - under WLS, v_{ii} is the i^{th} diagonal element of $(\boldsymbol{X^T W}_g \boldsymbol{X})^{-1}$
- Degrees of freedom for moderated t statistic: $n - p + d_0$
- If d_0 is large compared to $n - p$, moderated statistics have a bigger effect compared to using regular t 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)

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

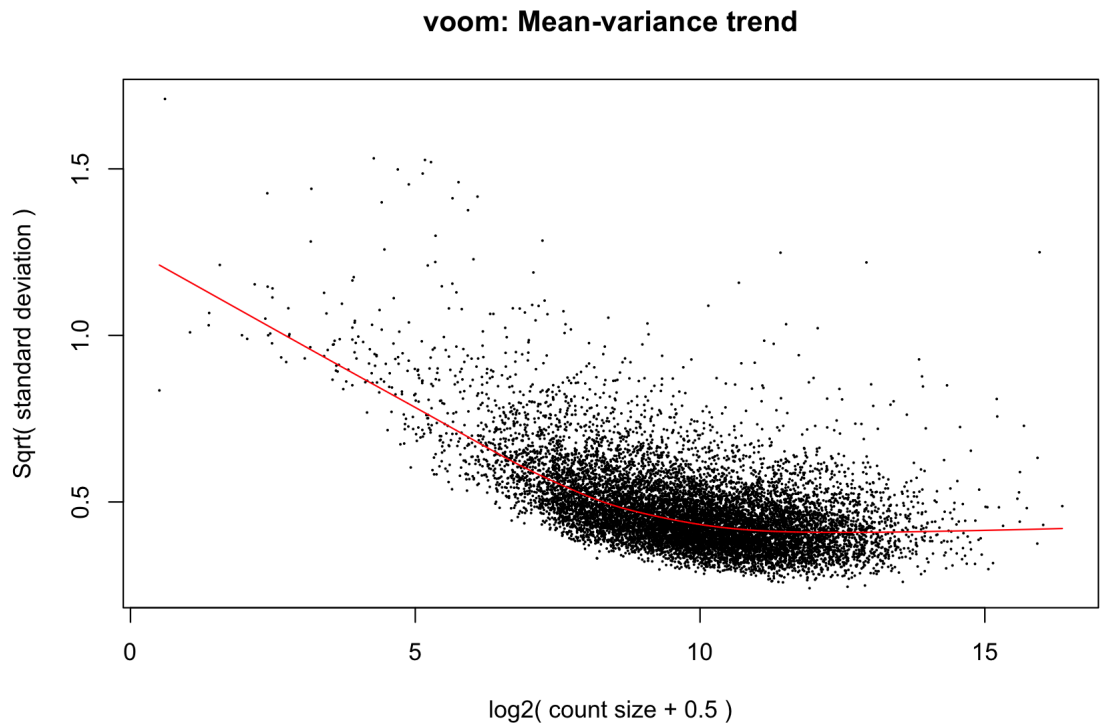
$$x_{i,Mut} = \begin{cases} 1 \text{ if sample } i \text{ is Mutant} \\ 0 \text{ otherwise} \end{cases}, \quad x_{i,F} = \begin{cases} 1 \text{ if sample } i \text{ is Female} \\ 0 \text{ otherwise} \end{cases}, \quad x_{i,D\#} = \begin{cases} 1 \text{ if sample } i \text{ is DPC}\# \\ 0 \text{ otherwise}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

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

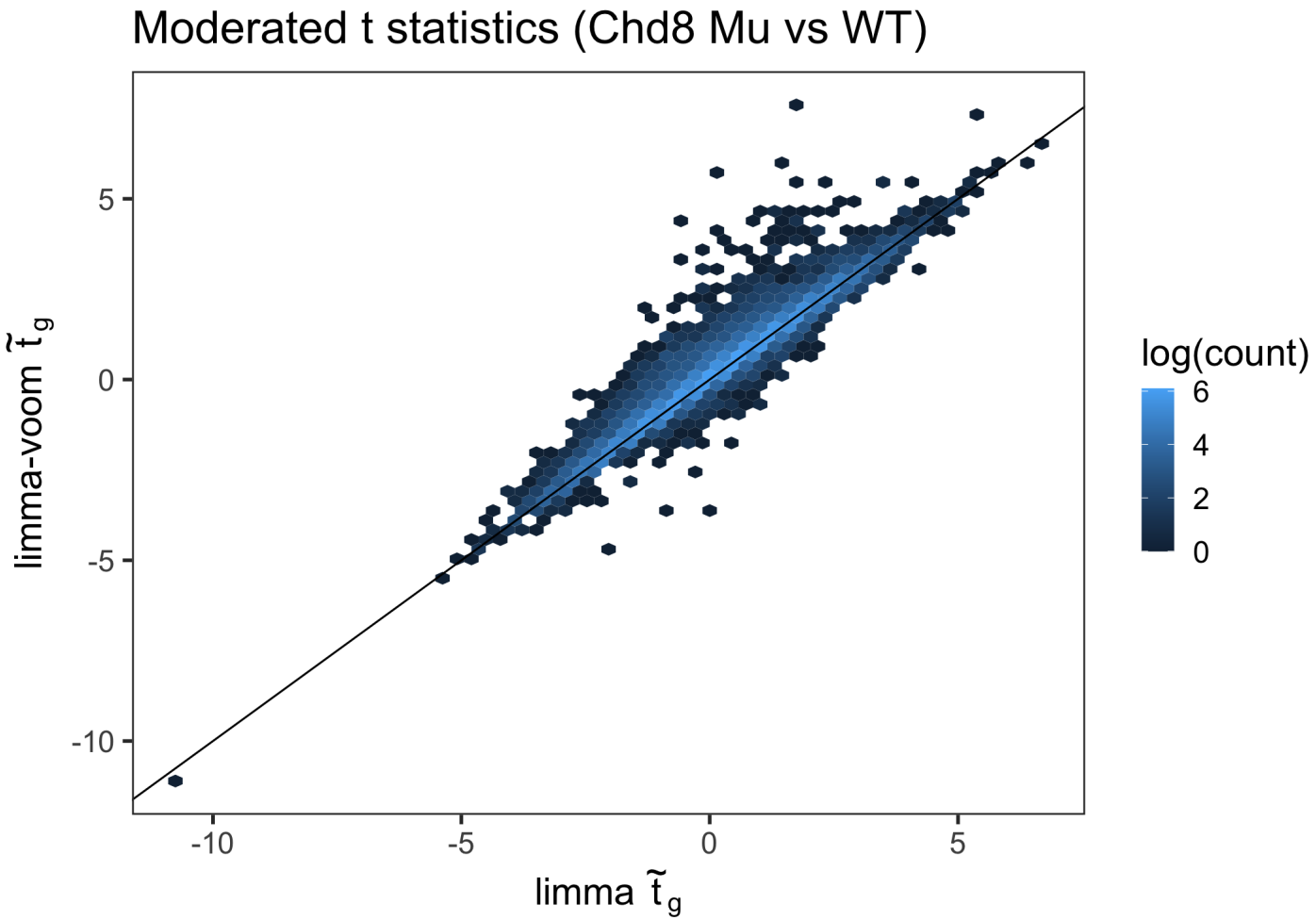
- Our model has $n - p = 44 - 7 = 37$ degrees of freedom
- We will focus on the null hypothesis of the main effect of Group $H_0 : \tau_{Mut} = 0$

limma-voom in action

```
vw <- voom(assays(sumexp)$counts,
           design = model.matrix(~ Sex + Group + DPC, data = colData(sumexp)),
           plot = TRUE, span = 0.5)
```



limma-voom vs limma



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:

$$\tilde{s}_g^2 = \frac{d_0 s_{0g}^2 + d s_g^2}{d_0 + d}$$

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:

$$\tilde{s}_g^2 = \frac{d_0 s_{0g}^2 + d s_g^2}{d_0 + d}$$

- Notice the g subscript on s_{0g}^2! The prior variance is different for each gene (unlike in regular limma)
- Based on the M-V trend, s_{0g}^2 is (typically) higher for lowly expressed genes

limma-trend vs voom?

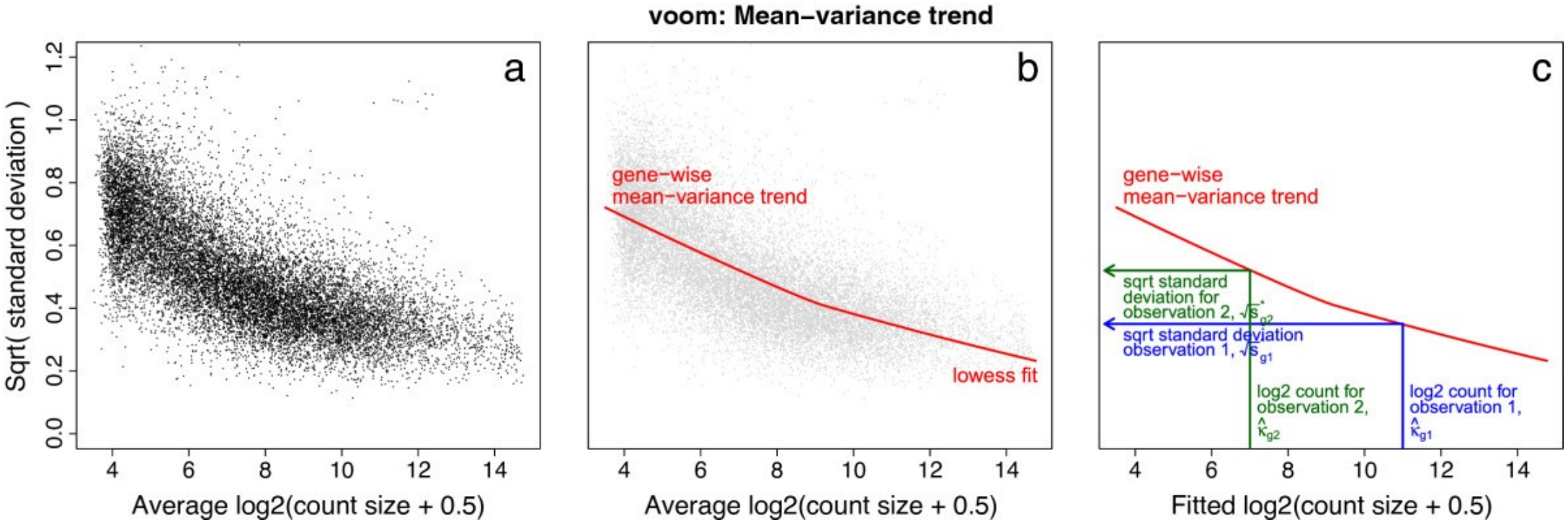


Figure 2, [Law et. al, 2014]

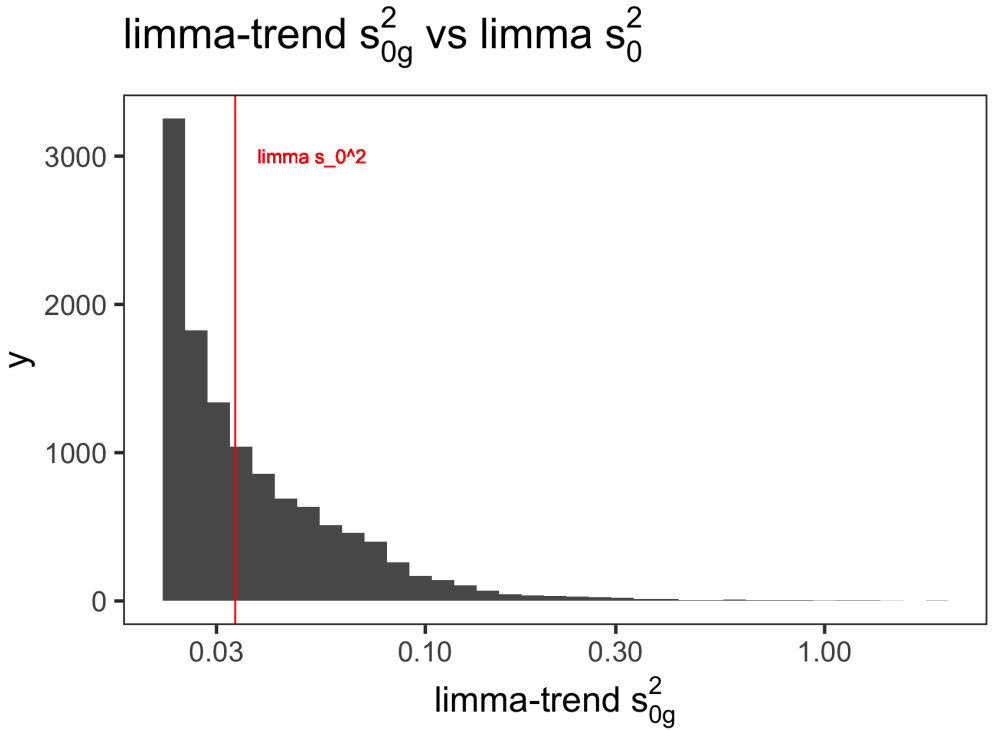
limma-trend in action

```
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 [1:12021] 0.0287 0.051 0.0274 0.023
0.0268 ...
##  - attr(*, "names")= chr [1:12021]
"0610007P14Rik" "0610009B22Rik" "0610009O20Rik"
"0610010F05Rik" ...
```

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

```
##  num 0.0334
```

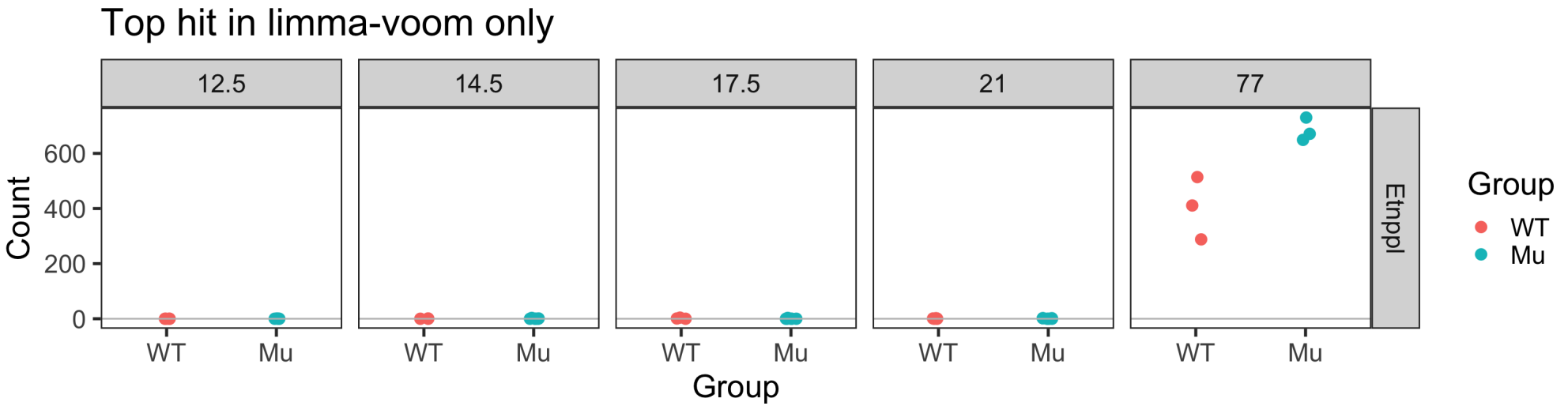


Nuances for limma-trend and limma-voom

- 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, s^2_{0g} will be constant across genes
- Even if M-V isn't flat, impact is most prominent in lowly expressed genes

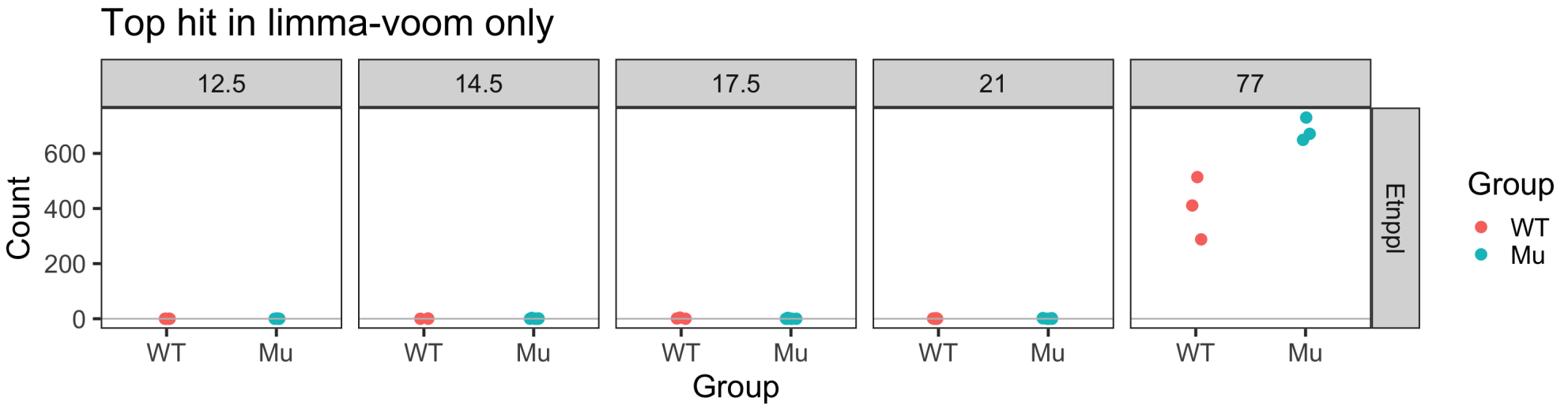
limma-voom 'false positives'?

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



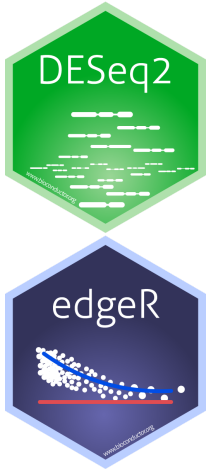
limma-voom 'false positives'?

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



- 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*

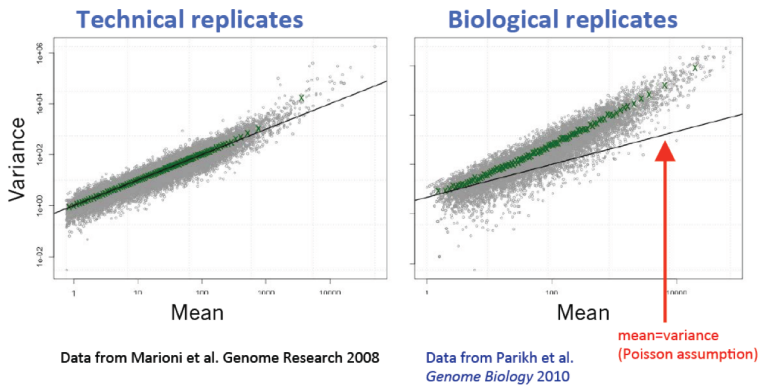
Another option: directly model counts



- 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
- 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: $\sigma_g^2 = \mu_g + \mu_g^2 \phi_g$
  - ϕ_g is the dispersion for gene g
  - if $\phi_g = 0$, get Poisson!
- We can perform inference about μ_g using GLM (e.g. using likelihood ratio tests)

Negative Binomial GLM

- Gene-specific variance under NB: $\sigma_g^2 = \mu_g + \mu_g^2 \phi_g$
  - ϕ_g is the dispersion for gene g
  - if $\phi_g = 0$, get Poisson!
- We can perform inference about μ_g using GLM (e.g. using likelihood ratio tests)
- To do so, we need to treat ϕ_g as known (*so first need to estimate it*)

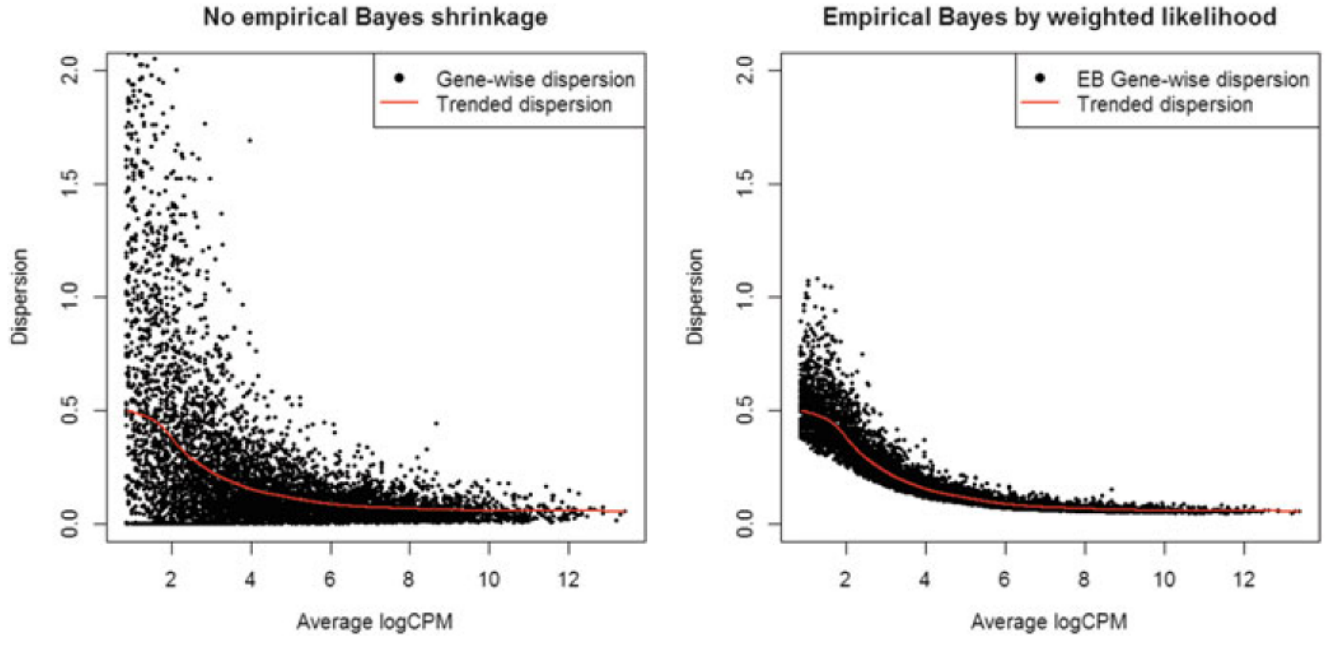
Negative Binomial GLM

- Gene-specific variance under NB: $\sigma_g^2 = \mu_g + \mu_g^2 \phi_g$
  - ϕ_g is the dispersion for gene g
  - if $\phi_g = 0$, get Poisson!
- We can perform inference about μ_g using GLM (e.g. using likelihood ratio tests)
- To do so, we need to treat ϕ_g as known (*so first need to estimate it*)

Estimation of dispersion is the main issue addressed by methods like edgeR and DEseq2

Disperson estimation

- One option is to assume ϕ_g is a set parametric function of the mean μ_g (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, Chapter 3 by Chen, Lun, and Smyth (2014)

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

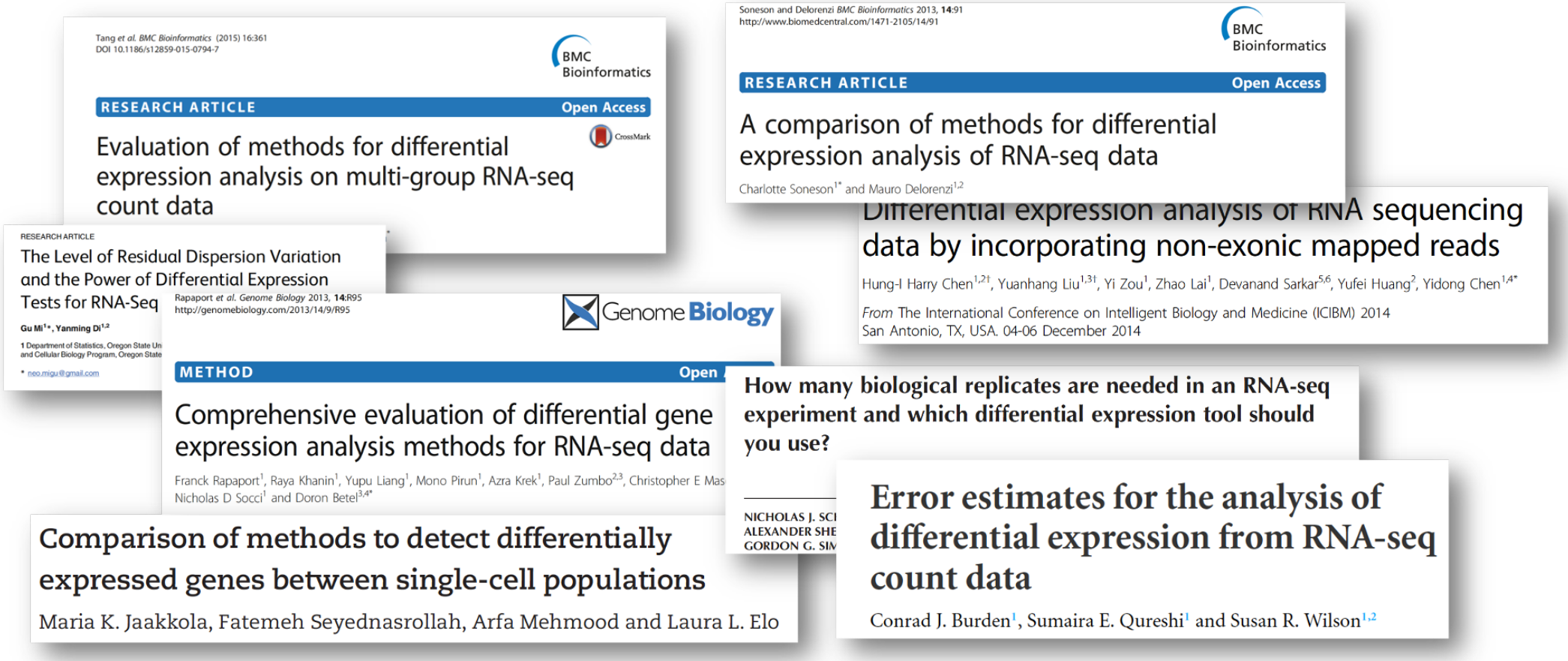
```
dge <- DGEList(assays(sumexp)$counts)
dge <- calcNormFactors(dge)
dge <- estimateDisp(dge,
                    design = model.matrix(~ Sex + Group + DPC,
                                          data = colData(sumexp)),
                    robust = TRUE)

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

DESeq2

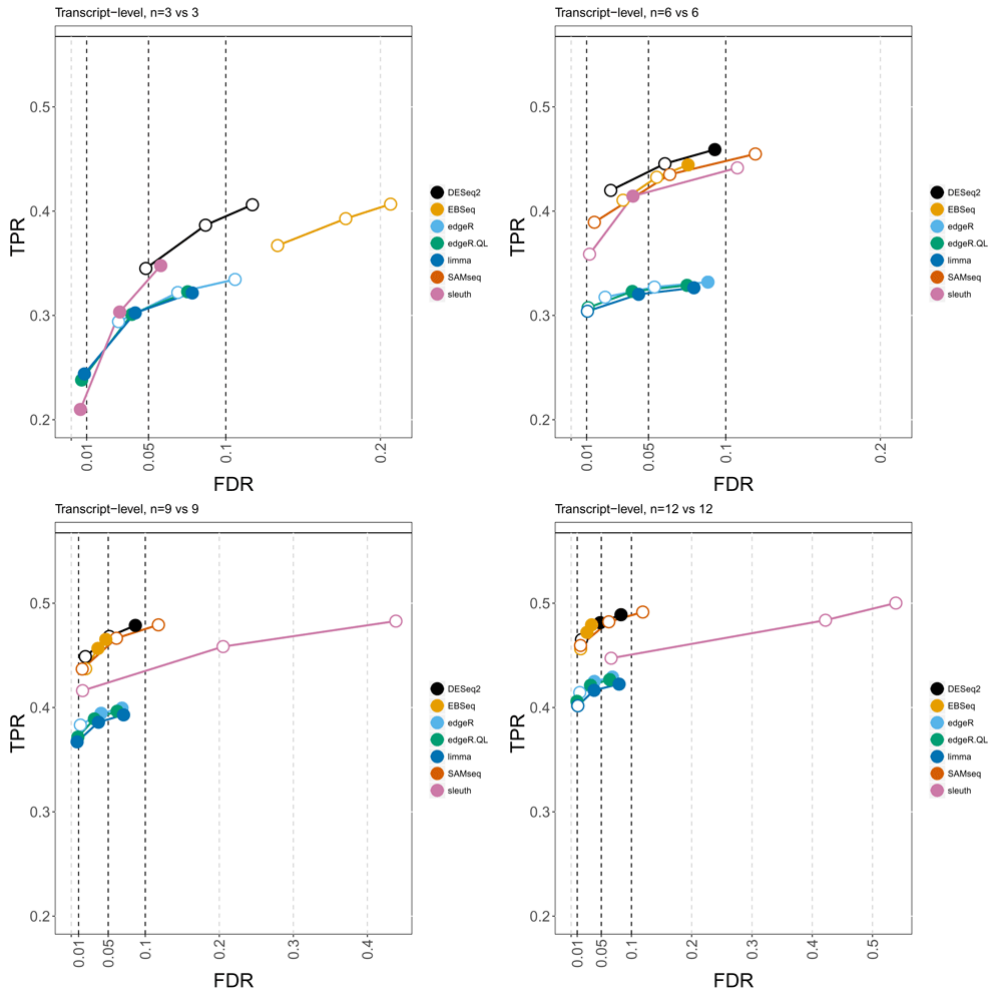
```
dds <- DESeqDataSet(sumexp,
                    design = model.matrix(~ Sex + Group + DPC,
                                          data = colData(sumexp)))
dds <- estimateSizeFactors(dds)
dds <- DESeq(dds)
```

```
## using supplied model matrix

## using pre-existing size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

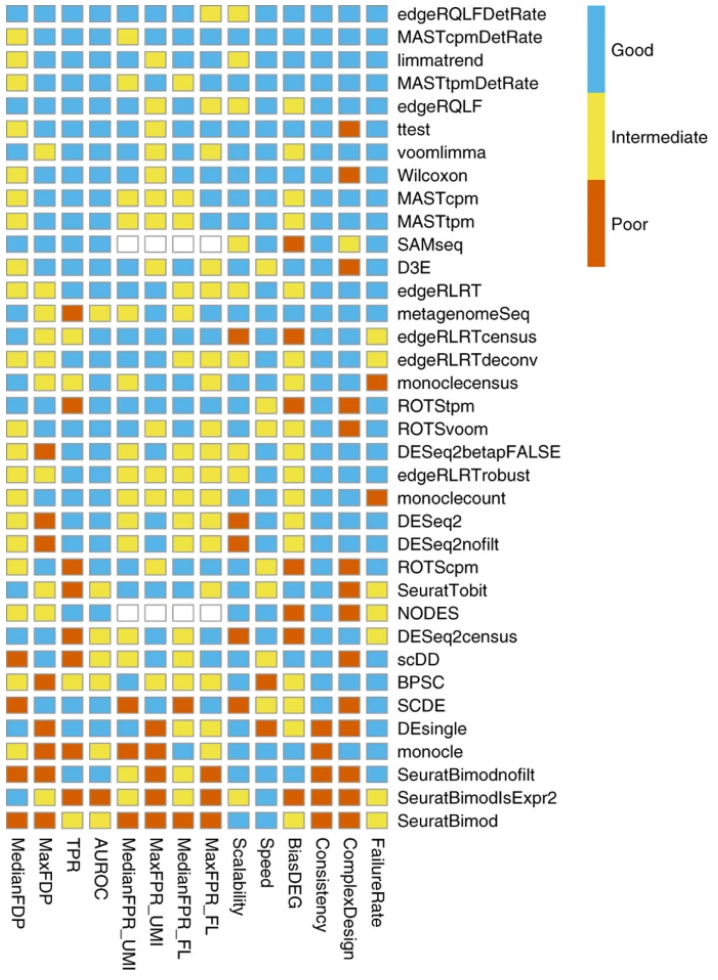
How to choose a method?



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

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

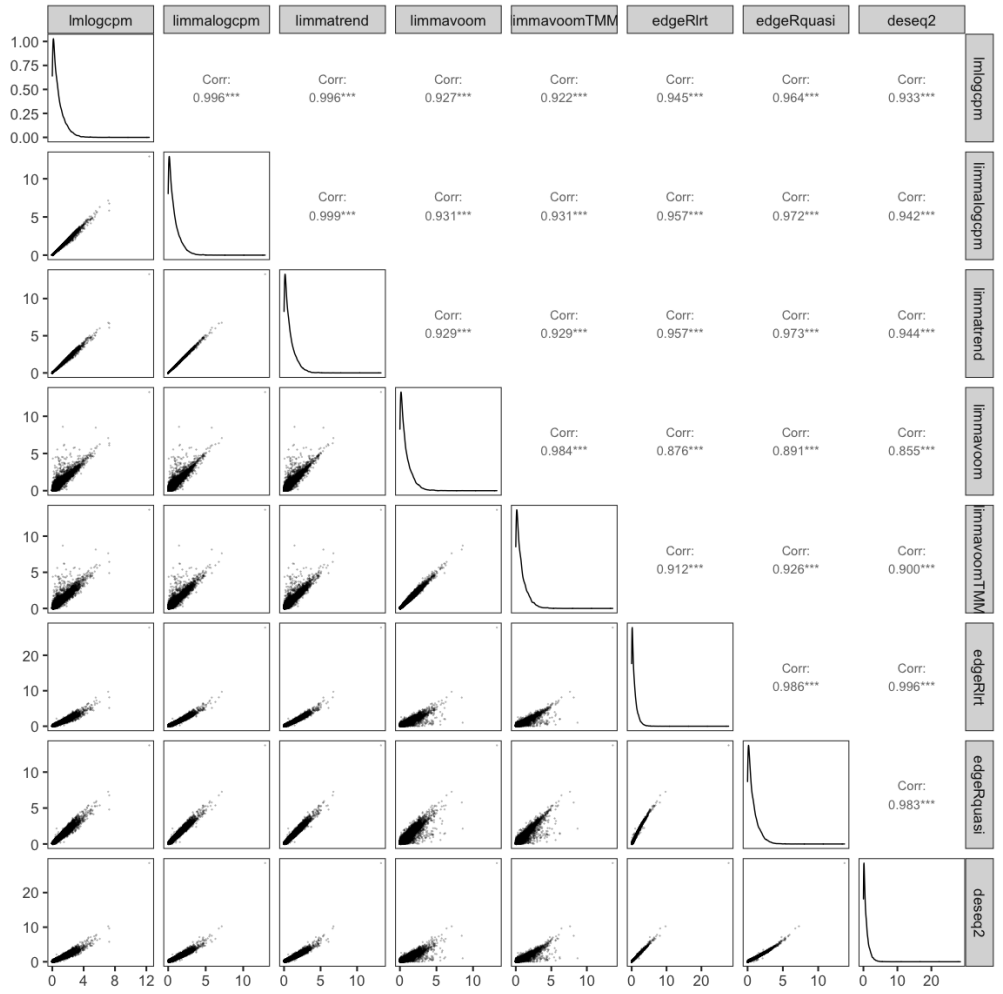
tl;dr version: there isn't a big difference

Possible reasons why:

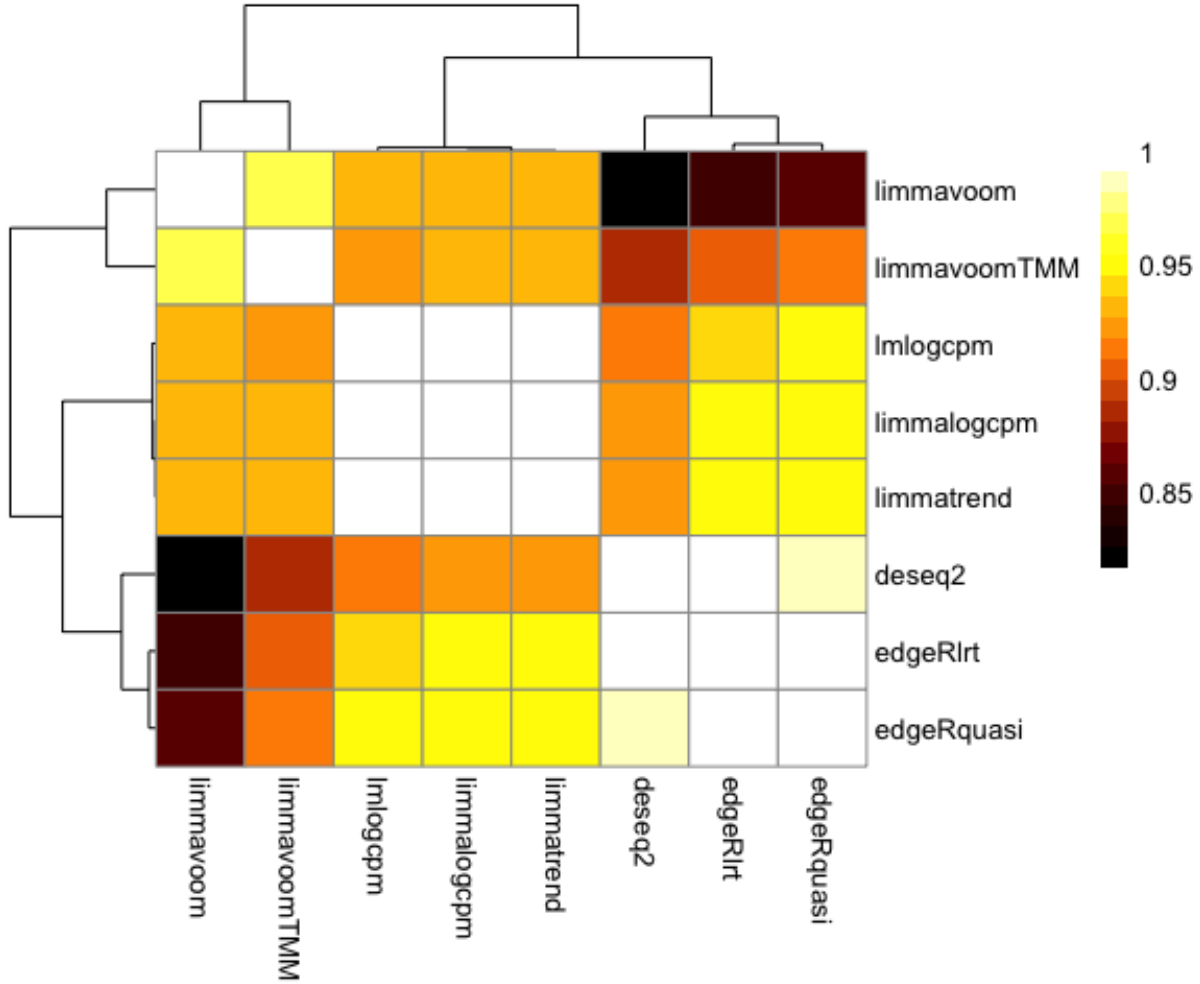
- methods have been converging in approach
- modeling count data directly with GLMs is more important for smaller samples sizes, lower read depth

Check out the comparisons in detail in the companion notes

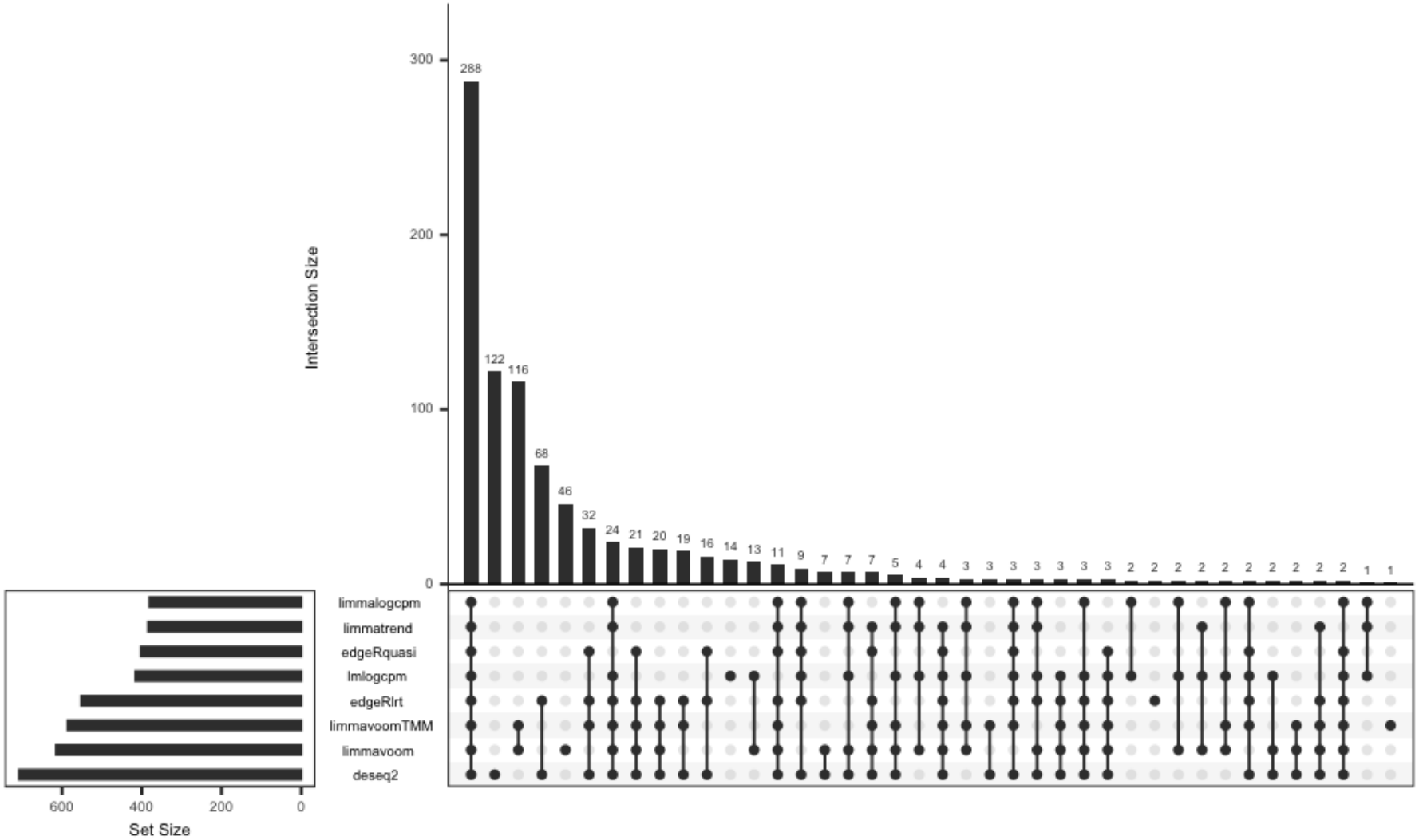
Comparisons of p-values for Chd8 mutation effect



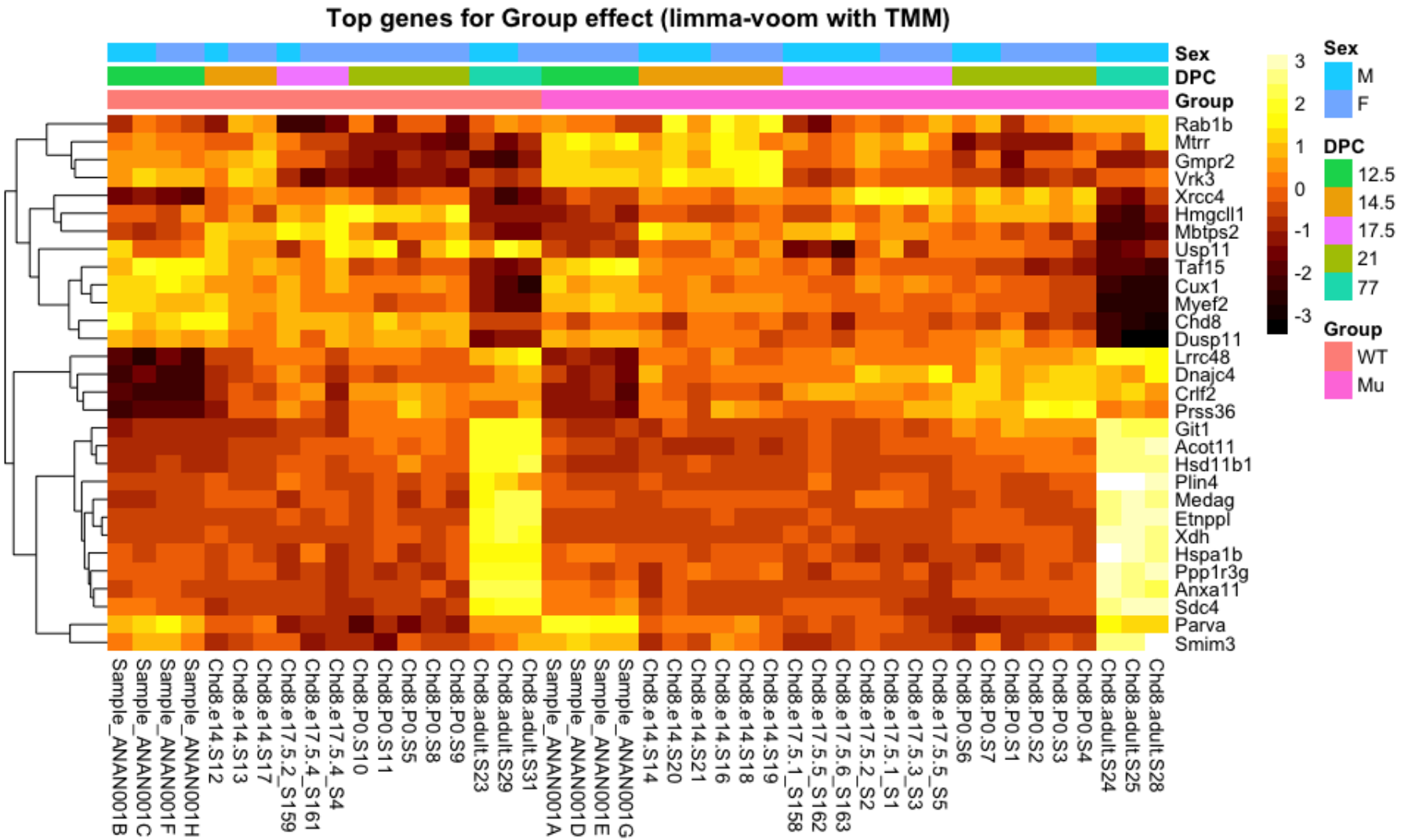
Correlation of p-value ranks for the effect of Chd8 mutation



Overlap of genes with FDR < 0.05 for the effect of Chd8 mutation



Heatmap of top 30 genes by limma-voom applied to TMM



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