Modern statistics for modern biology

Chapter 8. High-throughput count data

Presented by Seungeun Lee

Aug 7, 2024

Bioinformatics and Genomics Lab

Goal of this chapter

- Detect and quantify systematic changes between samples from different conditions

  (= Can we distinguish systematic changes from sampling/experimental variation?)

- Keywords:

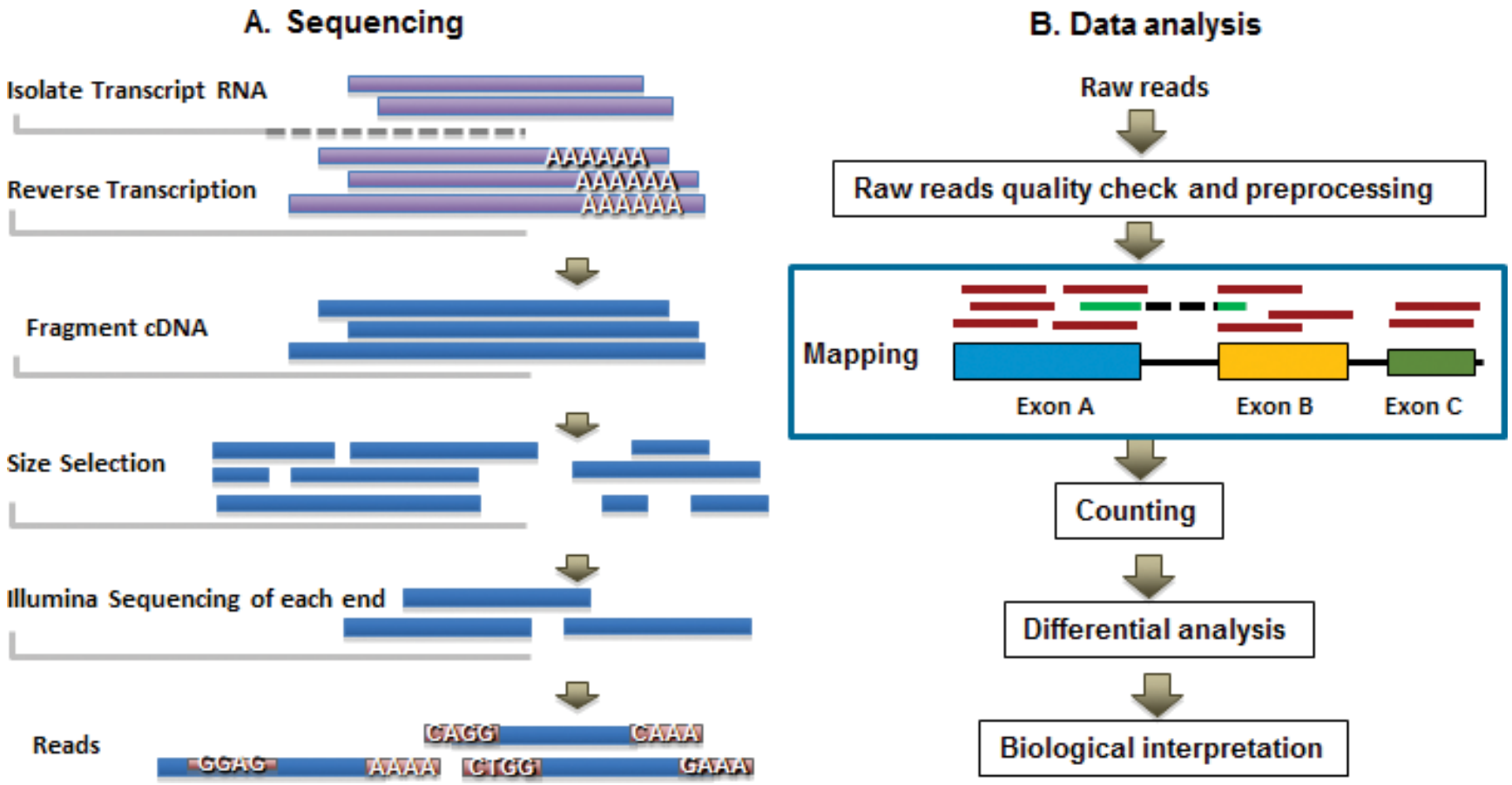
  1) Multifactorial designs, linear models, analysis of variance

  2) Generalized linear models

  3) Robustness and outlier detection

  4) Shrinkage estimation

Some core concepts – an RNA-seq experiment and data analysis pipeline



https://www.rna-seqblog.com/bioinformatics-for-rna%E2%80%90seq-data-analysis/

Example dataset – *pasilla* data

- RNA-seq count data of the splicing factor *pasilla* knock-down in *Drosophila* (Brooks et al., Genome Research 2011)

```
fn = system.file("extdata", "pasilla_gene_counts.tsv",
                 package = "pasilla", mustWork = TRUE)
counts = as.matrix(read.csv(fn, sep = "\t", row.names = "gene_id"))
```

```
dim(counts)
```

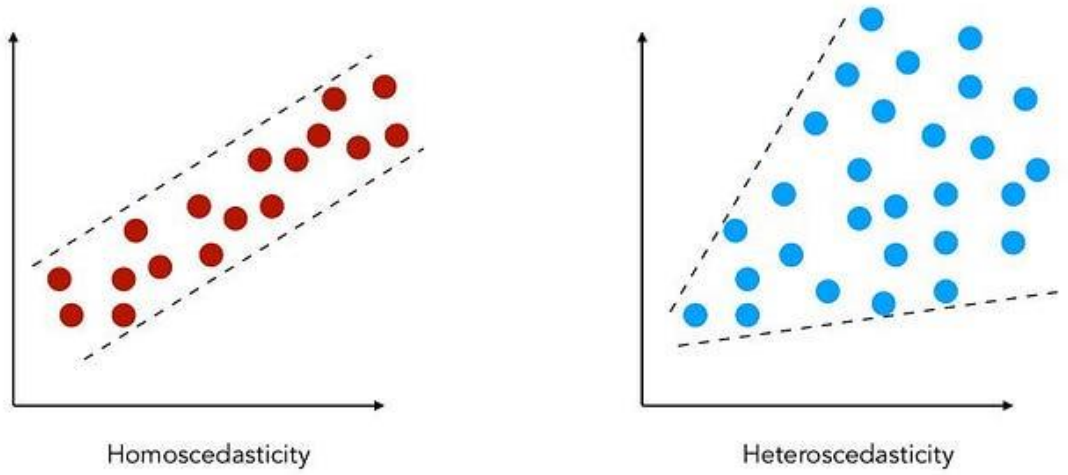
```
[1] 14599     7
```

```
counts[ 2000+(0:3), ]
```

```
           untreated1 untreated2 untreated3 untreated4 treated1 treated2
FBgn0020369       3387       4295       1315       1853     4884     2133
FBgn0020370       3186       4305       1824       2094     3525     1973
FBgn0020371          1          0          1          1        1        0
FBgn0020372         38         84         29         28       63       28
           treated3
FBgn0020369     2165
FBgn0020370     2120
FBgn0020371        0
FBgn0020372       27
```

The challenges of count data

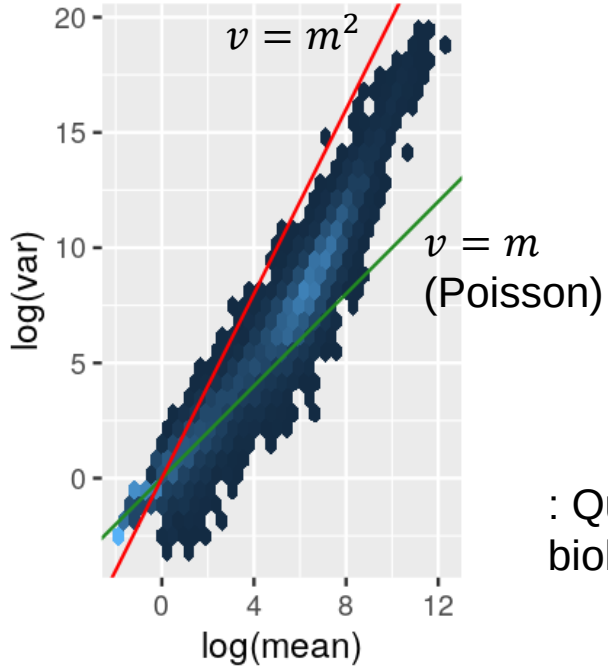
- Heteroskedasticity: the distribution shape of the data in different parts of the range are very different
- Distribution not symmetric
- Systematic sampling biases (e.g. sequencing depth) and stochastic experimental variation



https://medium.com/riga-data-science-club/learn-heteroskedasticity-in-1-minute-42f678911389

Modeling count data

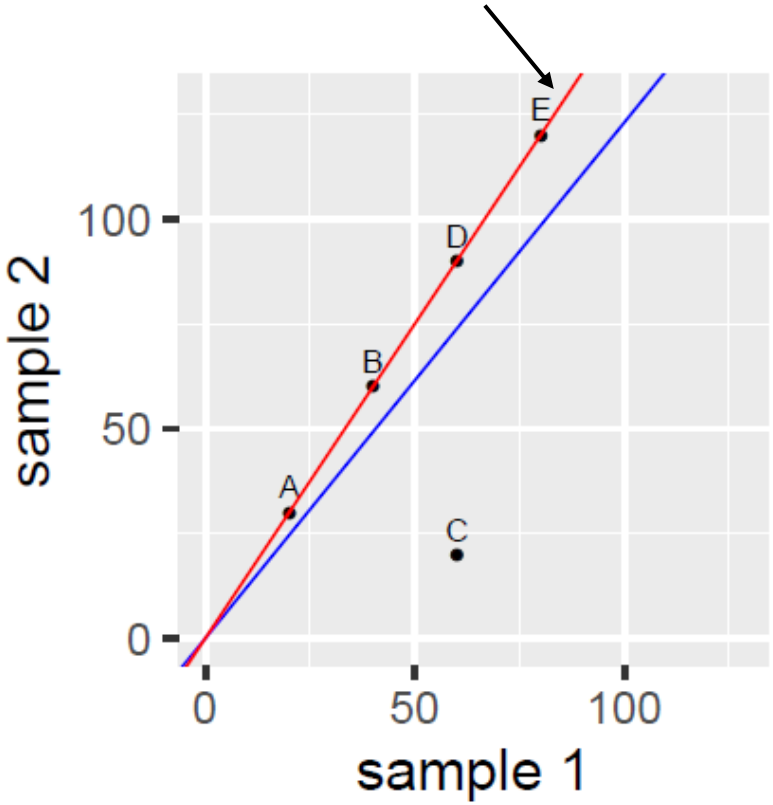
- Read count for gene i *within* a single library can be modeled by a Poisson distribution.
- However, our usual interest is comparing the read counts *between* libraries.
- Replication experiments vary more than what the Poisson distribution predicts.
- Gamma-Poisson distribution suits our need:



: Quadratic mean-variance relationship in biological replicates in the *pasilla* dataset

Normalization with proportionality factors to adjust the systematic bias

DESeq2's approach



- We first need to adjust for variations in the total number of reads in each sample.
- Two different ways of size factor estimation:

  Naïve approach considering all genes (blue line) vs. robust regression (red line, preferred)

A basic analysis – data preprocessing

- DESeq2 uses a specialized data container called DESeqDataSet to store the datasets to work with.
- It helps users keep related data together.

```
annotationFile = system.file("extdata",
  "pasilla_sample_annotation.csv",
  package = "pasilla", mustWork = TRUE)
pasillaSampleAnno = readr::read_csv(annotationFile)
pasillaSampleAnno
```

```
> pasillaSampleAnno
# A tibble: 7 × 6
  file         condition type        `number of lanes` `total number of reads` `exon counts`
  <chr>        <chr>     <chr>                   <dbl> <chr>                          <dbl>
1 treated1fb   treated   single-read                 5 35158667                    15679615
2 treated2fb   treated   paired-end                  2 12242535 (x2)               15620018
3 treated3fb   treated   paired-end                  2 12443664 (x2)               12733865
4 untreated1fb untreated single-read                 2 17812866                    14924838
5 untreated2fb untreated single-read                 6 34284521                    20764558
6 untreated3fb untreated paired-end                  2 10542625 (x2)               10283129
7 untreated4fb untreated paired-end                  2 12214974 (x2)               11653031
```

```
library("dplyr")
pasillaSampleAnno = mutate(pasillaSampleAnno,
condition = factor(condition, levels = c("untreated", "treated")),
type = factor(sub("-.*", "", type), levels = c("single", "paired")))
```

```
mt = match(colnames(counts), sub("fb$", "", pasillaSampleAnno$file))
stopifnot(!any(is.na(mt)))

pasilla = DESeqDataSetFromMatrix(
  countData = counts,
  colData   = pasillaSampleAnno[mt, ],
  design    = ~ condition)
class(pasilla)
```

```
[1] "DESeqDataSet"
attr(,"package")
[1] "DESeq2"
```

A basic analysis – differential expression analysis

- Aim: identify genes that are differentially abundant between the treated and the untreated cells
- A test that is conceptually similar to the t-test is applied
- Standard analysis steps are wrapped into a single function `DESeq`.

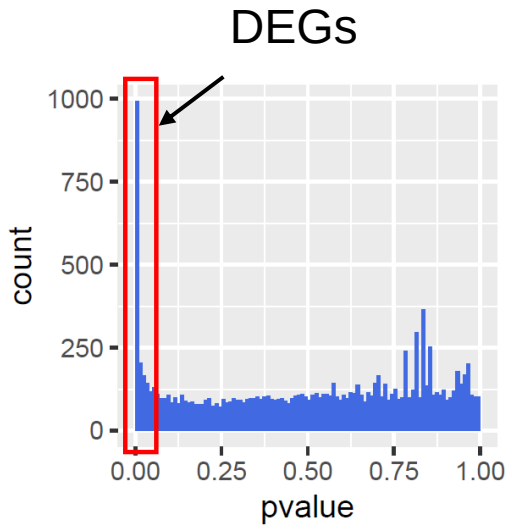
```
pasilla = DESeq(pasilla)
```

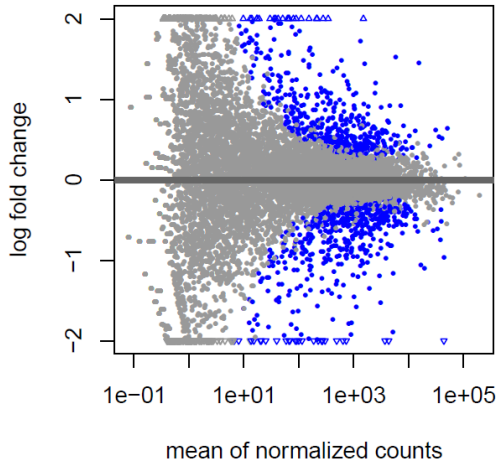
```
res = results(pasilla)
res[order(res$padj), ] |> head()
```

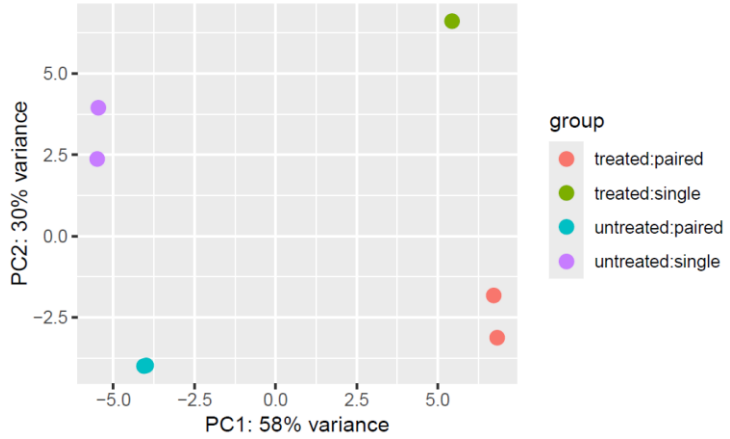
```
log2 fold change (MLE): condition treated vs untreated
Wald test p-value: condition treated vs untreated
DataFrame with 6 rows and 6 columns
             baseMean log2FoldChange     lfcSE      stat       pvalue
            <numeric>      <numeric> <numeric> <numeric>    <numeric>
FBgn0039155   730.596       -4.61901 0.1687068  -27.3789 4.88599e-165
FBgn0025111  1501.411        2.89986 0.1269205   22.8479 1.53430e-115
FBgn0029167  3706.117       -2.19700 0.0969888  -22.6521 1.33042e-113
FBgn0003360  4343.035       -3.17967 0.1435264  -22.1539 9.56283e-109
FBgn0035085   638.233       -2.56041 0.1372952  -18.6490  1.28772e-77
FBgn0039827   261.916       -4.16252 0.2325888  -17.8965  1.25663e-71
                   padj
              <numeric>
FBgn0039155 4.06661e-161
FBgn0025111 6.38497e-112
FBgn0029167 3.69104e-110
FBgn0003360 1.98979e-105
FBgn0035085  2.14354e-74
FBgn0039827  1.74316e-68
```

A basic analysis – data visualization to assess for data quality

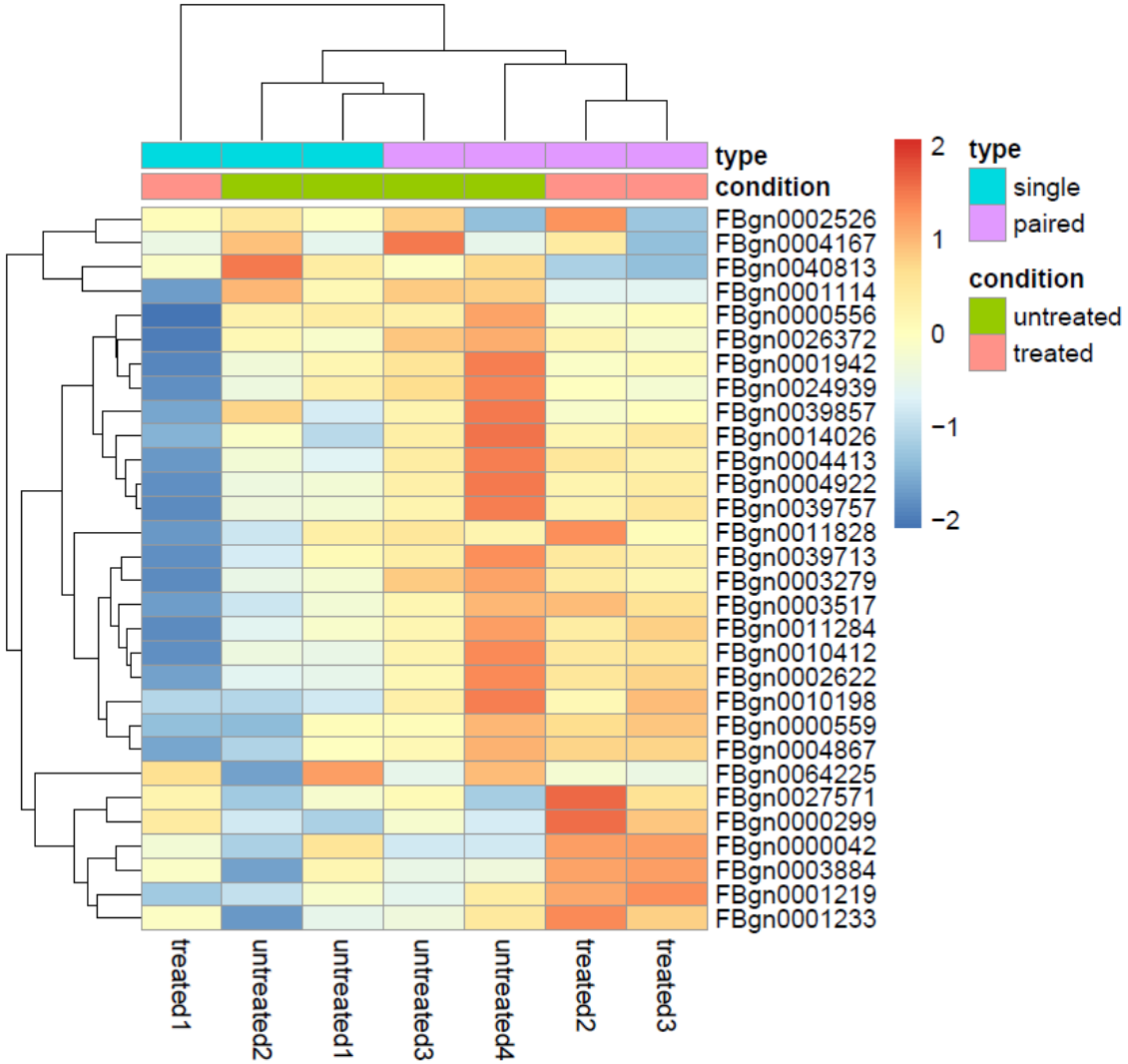
- 1) Histogram of p-values (indicative of FDR & batch effect)
- 2) MA plot
- 3) Orientation plot (useful for visualizing the overall effect of experimental covariates and/or detecting batch effects)







A basic analysis – data visualization to assess for data quality



- + a heatmap
- The clustering of the columns is dominated by the type factor.
- We should adjust for this “nuisance” factor to test for differentially expressed genes between conditions.

Critique of default choices and possible modifications

- (The few-changes assumption) DESeq2 default normalization and dispersion estimates assume that most genes are not differentially expressed.
- If this assumption is violated, we need to identify a subset of negative control genes.

- (Point-like null hypothesis) As a default, DESeq2 tests against the null hypothesis that each gene has the same abundance across conditions.
- As sample size gets bigger, spurious significance can arise → In this case, one can modify the test to use interval-based null hypothesis (8.10.4)

Multifactorial designs and linear models – What is a multifactorial design?

- A multifactorial design: investigate the effect of 1) siRNA knockdown of the *pasilla* gene and 2) effect of a certain drug on the expression of a certain gene
- Can simplify the design and result of the experiment in the following notation and the design matrix:

$$y = \beta_0 + x_1\beta_1 + x_2\beta_2 + x_1x_2\beta_{12}. \quad (8.1)$$

x_0	x_1	x_2
1	0	0
1	1	0
1	0	1
1	1	1

- y: expression level of a gene
- x_i: design factors, binary indicator variables. 1 if the experiment is done, 0 if not.
- Since in an RNA-seq experiment there are lots of genes, we'll have many copies of eq. 8.1.

Multifactorial designs and linear models – What is a multifactorial design?

$$y = \beta_0 + x_1\beta_1 + x_2\beta_2 + x_1x_2\beta_{12}. \qquad (8.1)$$

- β_0 (intercept): the base level of the measurement in the negative control
- β_1: change due to siRNA transfection
- β_2: change due to treatment with the drug
- $\beta_{1,2}$: interaction effect of siRNA and drug (We don't always care about interactions. $\beta_{1,2}$ is often not considered.)

Multifactorial designs and linear models – What is a multifactorial design?

$$y = \beta_0 + x_1\beta_1 + x_2\beta_2 \qquad (8.1)$$

- β_i tell us how much the gene expression level changes when one of the predictor variables (x_i) changes with the others fixed.
- If $x_1 = 1$ and $x_2 = 0$:

$$\begin{aligned}\beta_1 = y - \beta_0 &= \log_2(\text{expression}_{\text{treated}}) - \log_2(\text{expression}_{\text{untreated}}) \\ &= \log_2 \frac{\text{expression}_{\text{treated}}}{\text{expression}_{\text{untreated}}}\end{aligned}$$

Multifactorial designs and linear models – noise and replicates

- Real data are affected by noise.

  We need replicates to estimate the levels of noise and assess the uncertainty of our estimated βs.

- We extend the equation by adding the index j and a new term ϵ_j (j for each replicate; ϵ is called residual, to absorb differences between replicates):

$$y_j = x_{j0}\,\beta_0 + x_{j1}\,\beta_1 + x_{j2}\,\beta_2 + x_{j1}\,x_{j2}\,\beta_{12} + \varepsilon_j. \qquad (8.5)$$

- If for each of the four conditions we perform three replicates, then j counts from 1 to 12.

Multifactorial designs and linear models – noise and replicates

- The system of 12 equations would be underdetermined without further information.

  It has more variables (12 ϵs and 4 βs) than its equations (12, one for each j).

- To fix this, we require that the ϵ_j to be small.
- A popular way to do this is to minimize the sum of squared residuals:

$$\sum_j \epsilon_j^2 \quad \rightarrow \quad \text{min.} \qquad (8.6)$$

- If this is satisfied, the βs represent the average effects of each of the experimental factors, while the residuals ϵ_j reflect the experimental fluctuations around the mean between the replicates (average of 0)
- This approach is called the **least sum of squares fitting.**

Multifactorial designs and linear models – analysis of variance

- A model like 8.5 is called a linear model:

$$y_j = x_{j0}\,\beta_0 + x_{j1}\,\beta_1 + x_{j2}\,\beta_2 + x_{j1}\,x_{j2}\,\beta_{12} + \varepsilon_j. \qquad (8.5)$$

- It decomposes the variability of y that we observed in the experiments into elementary components:

  baseline value β_0,

  its variability caused by the effect of the first variable (β_1),

  variability caused by the effect of the second variable (β_2),

  and variability that is unaccounted for (ϵ)

- The last is commonly called noise and the other ones systematic variability.

Multifactorial designs and linear models – robustness

- The sum 8.6 is sensitive to outliers in the data.

$$\sum_j \varepsilon_j^2 \quad \rightarrow \quad \text{min.} \qquad (8.6)$$

- To achieve a higher degree of robustness against outliers, other choices than the sum of squares can be used as the objective of minimization.
- DESeq2 packages uses the below approach (R is the quantity to be minimized):

$$R = \sum_j w_j \varepsilon_j^2$$

- Each data point is assessed using a measure called Cook's distance, and for those whose value is too large, the weight is set to 0, whereas the other data points get $w_j = 1$.

Generalized linear models

$$y_j = x_{j0}\,\beta_0 + x_{j1}\,\beta_1 + x_{j2}\,\beta_2 + x_{j1}\,x_{j2}\,\beta_{12} + \varepsilon_j. \qquad (8.5)$$

- We can generalize the assumptions used in the above equation:

  (1) consider data after some transformation (log2, variance stabilizing, …)

  (2) generalization concerns on the minimization criterion and probabilistic model (gamma-Poisson instead of normal)

$$\sum_j \varepsilon_j^2 \;\rightarrow\; \text{min.} \qquad \Leftrightarrow \qquad \prod_j p(\varepsilon_j) \;\rightarrow\; \text{max.}$$

$$p(\varepsilon_j) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\varepsilon_j^2}{2\sigma^2}\right)$$

(probability density function of a normal distribution)

Generalized linear models

- DESeq2 uses a generalized linear model of the form:

$$K_{ij} \sim \mathrm{GP}(\mu_{ij}, \alpha_i)$$

- $K_{i,j}$: count for gene i, sample j. $\mu_{i,j}$: mean, a_i: dispersion

  (By default, dispersion is different for each gene i, but the same across all samples → no index j)

$$\mu_{ij} = s_j \, q_{ij}$$

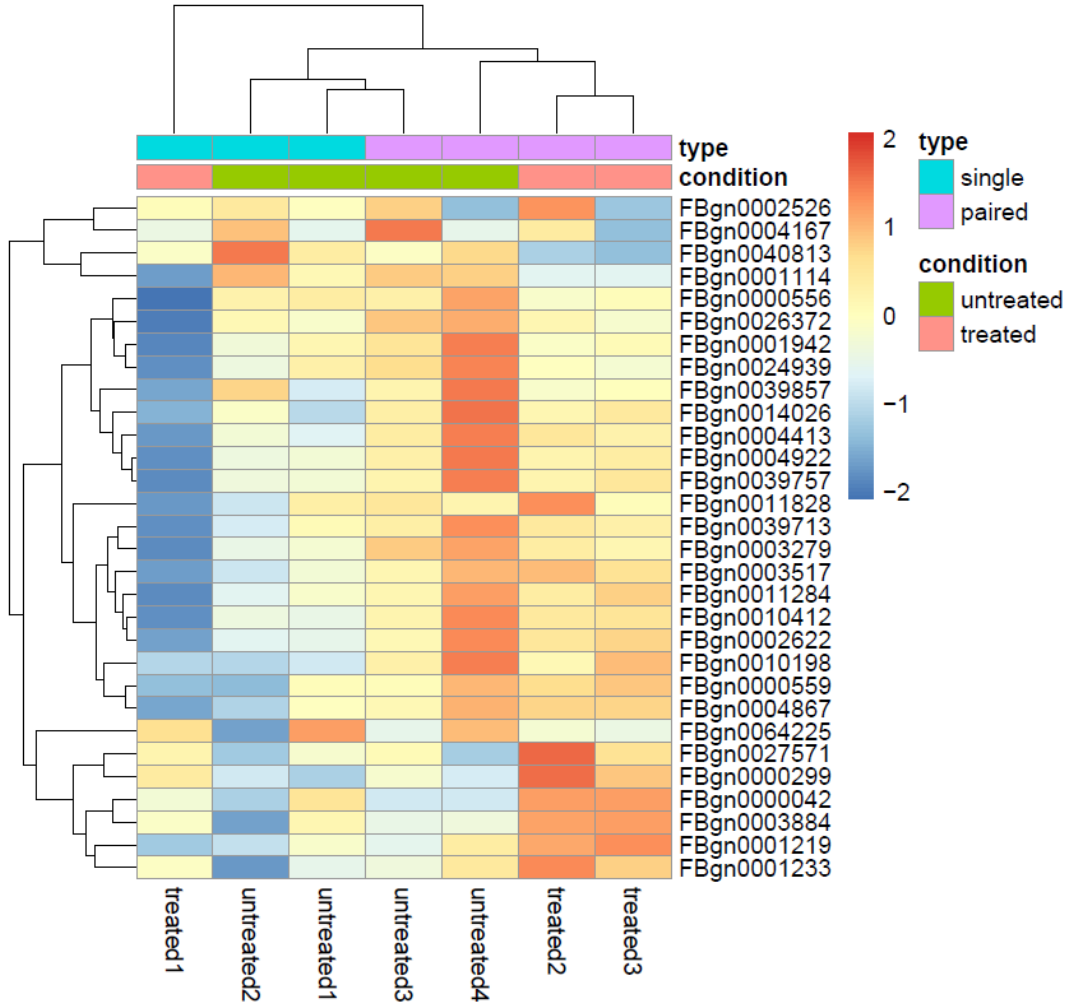
- s_j: normalization constant, $q_{i,j}$: expected concentration of fragments for gene i in sample j

$$\log_2(q_{ij}) = \sum_k x_{jk} \beta_{ik}.$$

- k: experimental factors, $\beta_{i,k}$: log2 fold changes for gene i for each column of the design matrix X

Two-factor analysis of the *pasilla* data

- Recall that `type` had a considerable systematic effect on the data



Two-factor analysis of the *pasilla* data

- Two-factor analysis considering `type` (`condition` is the one of primary interest):

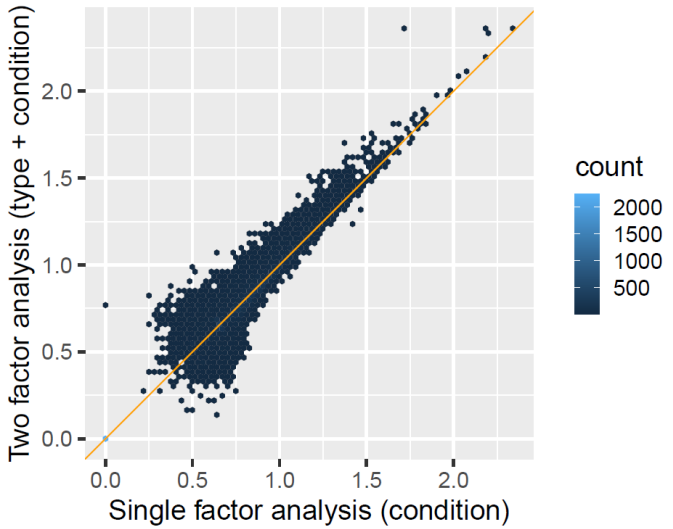
```
pasillaTwoFactor = pasilla
design(pasillaTwoFactor) = formula(~ type + condition)
pasillaTwoFactor = DESeq(pasillaTwoFactor)
```

```
res2 = results(pasillaTwoFactor)
head(res2, n = 3)
```

```
log2 fold change (MLE): condition treated vs untreated
Wald test p-value: condition treated vs untreated
DataFrame with 3 rows and 6 columns
             baseMean log2FoldChange     lfcSE       stat    pvalue      padj
            <numeric>      <numeric> <numeric>  <numeric> <numeric> <numeric>
FBgn0000003  0.171569      0.6745518  3.871091  0.1742537  0.861666        NA
FBgn0000008 95.144079     -0.0406731  0.222215 -0.1830351  0.854770  0.951975
FBgn0000014  1.056572     -0.0849880  2.111821 -0.0402439  0.967899        NA
```

Two-factor analysis of the *pasilla* data

- Comparing the p-values from 1) simple comparison and 2) two-factor analysis:



- Note that p-values are transformed as $f(p) = (-\log_{10} p)^{\frac{1}{6}}$:

  if $p = 10^{-1}$, $f(p) = 1$. if $p = 10^{-64}$, $f(p) = 2$.

- P-values in the two-factor analysis are generally smaller.

Two-factor analysis of the *pasilla* data

- We can also see the difference by counting the number of genes passing an FDR threshold of 10%:

```
compareRes = table(
   `simple analysis` = res$padj < 0.1,
   `two factor` = res2$padj < 0.1 )
addmargins( compareRes )
```

```
> addmargins(compareRes)
                two factor
simple analysis FALSE TRUE  Sum
          FALSE  6973  289 7262
          TRUE     25 1036 1061
          Sum    6998 1325 8323
```

- The two-factor analysis has more detection power.

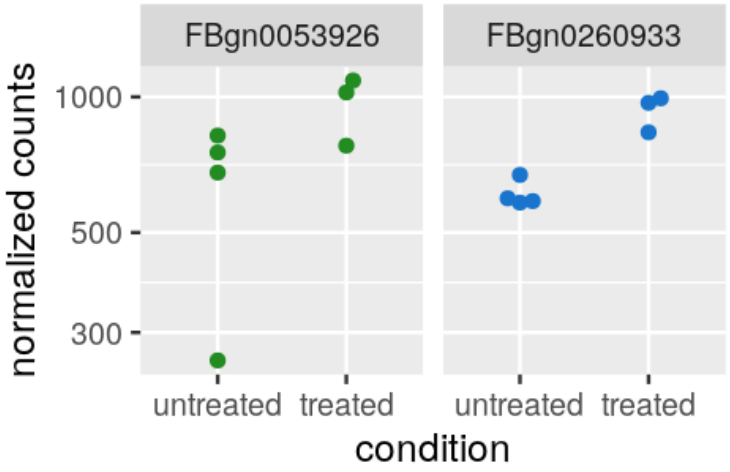
- Why this happens?

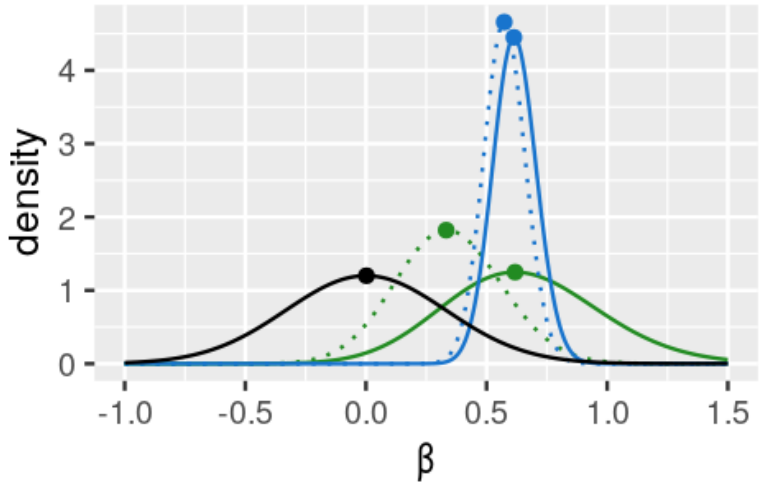
  ϵ absorbing the variance of the blocking factor vs. more parameters to be estimates

  (→ depends on the data …)

Further statistical concepts – sharing of dispersion information across genes

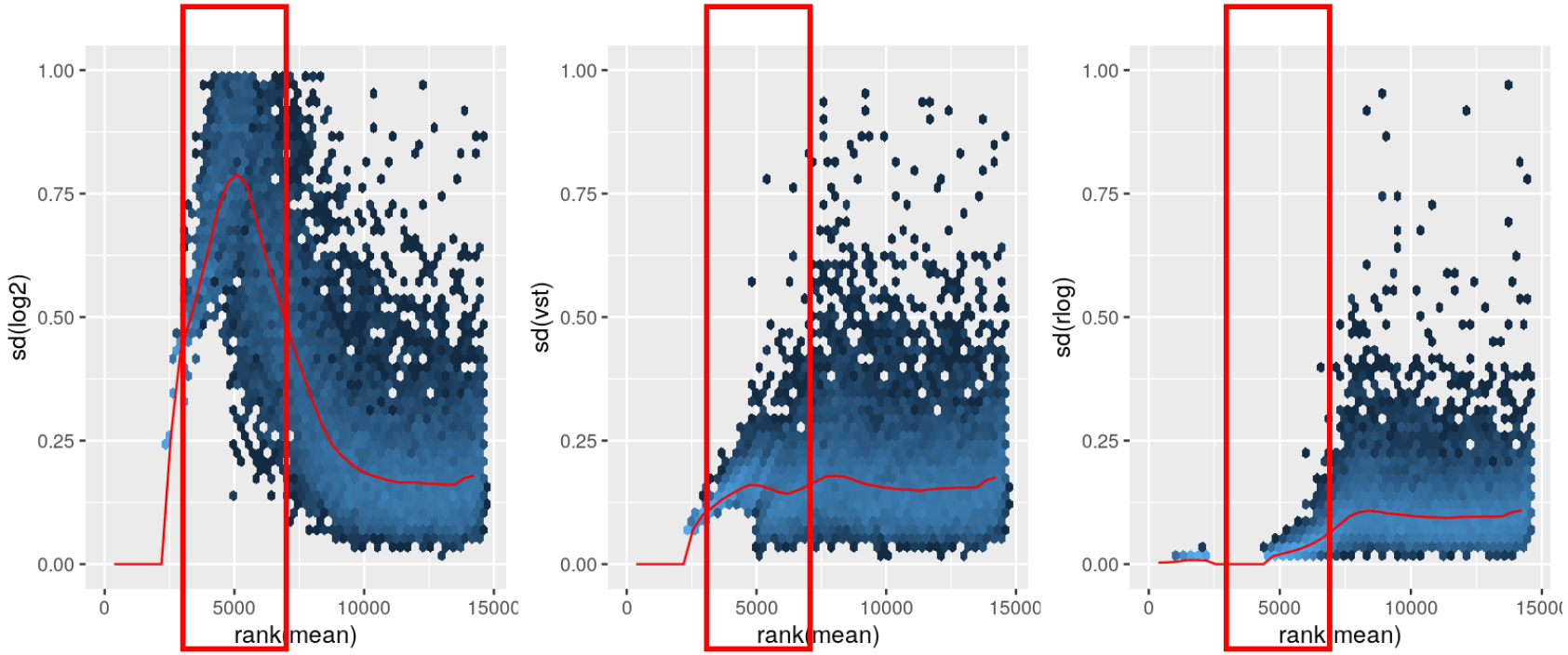
- DESeq2 uses empirical Bayesian approach to estimate the dispersion parameters and (optionally) the logarithmic fold changes.
- The priors are taken from the distributions of the maximum likelihood estimates across all genes.
- The empirical Bayes machinery then shrinks each per-gene MLE towards the prior peak.
- More dispersed → more influence of prior





Further statistical concepts – count data transformations

- Sometimes it is more useful to transform the data to a scale where the data are more homoscedastic.
- **log2 transformation** vs. **variance-stabilizing transformation** vs. **regularized logarithm transformation**



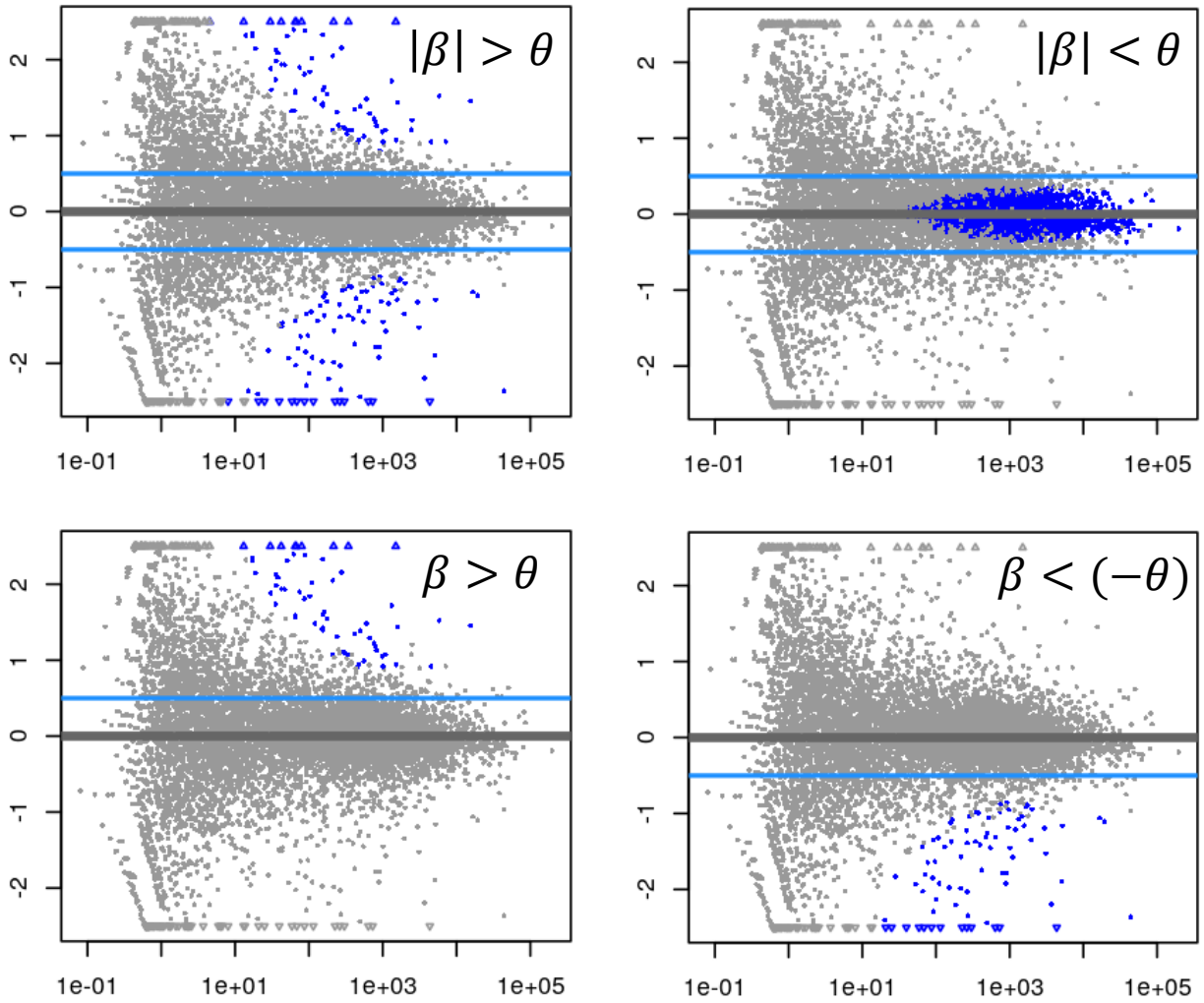

```
vsp = varianceStabilizingTransformation(pasilla)
```

```
library("vsn")
rlp = rlogTransformation(pasilla)

msd = function(x)
  meanSdPlot(x, plot = FALSE)$gg + ylim(c(0, 1)) +
     theme(legend.position = "none")

gridExtra::grid.arrange(
  msd(log2(counts(pasilla, normalized = TRUE) + 1)) +
    ylab("sd(log2)"),
  msd(assay(vsp)) + ylab("sd(vst)"),
  msd(assay(rlp)) + ylab("sd(rlog)"),
  ncol = 3
)
```

Further statistical concepts – tests of log2-fold change above or below a threshold



β : log2 fold change
θ: threshold

```
par(mfrow = c(4, 1), mar = c(2, 2, 1, 1))
myMA = function(h, v, theta = 0.5) {
  plotMA(pasilla, lfcThreshold = theta, altHypothesis = h,
         ylim = c(-2.5, 2.5))
  abline(h = v * theta, col = "dodgerblue", lwd = 2)
}
myMA("greaterAbs", c(-1, 1))
myMA("lessAbs",    c(-1, 1))
myMA("greater",          1)
myMA("less",          -1   )
```

- “Banded” null hypothesis: we can detect effects that have a strong enough size (as opposed to statistically significant but with small effect size).

Summary

- We have seen how to analyze count tables from high-throughput sequencing for differential abundance, in various designs (basic two group comparison, multifactorial designs, experiments with covariates).

[Problems with count data]

1. Discrete distributions, tend to be skewed with highly different variances
2. Different sequencing depths for replicates
3. The number of replicates too small to estimate the dispersion parameter
4. The null hypothesis (each gene has the same abundance across conditions, or the effect size is exactly zero) is almost never true

→

[Proposed solutions]

1. Generalized linear models, various transformations
2. Size factors s_i were estimated for each sample for normalization purpose
3. Shrinkage or empirical Bayes techniques
4. May be overcome by considering effect size as well as statistical significance