Gene expression statistical analysis

Davide Risso

24/01/2018

Introduction

What we will cover

I will give you a *brief* introduction to the *statistical analysis of RNA-seq*.

We will focus on differential expression analysis using *R/Bioconductor*.

We will start from a *matrix of gene-level read counts*.

We will cover the two most popular packages, `DESeq2` and `edgeR`.

I will also show you how to deal with unwanted variation using the `RUVSeq` package.

What we will not cover

I will not show you the R code, but *focus on the statistical concepts*.

The R code that I used for the plots in these slides is available online at https://github.com/drisso/canazei.

Only in Bioconductor, there are 150 packages for QC/EDA, 69 for normalization, and 241 for differential expression!

Hence, this is not a comprehensive account on how to perform these steps, but rather an *introduction to the statistical methods* behind some of them.

Useful links

- These slides: https://github.com/drisso/canazei
- Example dataset: https://github.com/drisso/peixoto2015_tutorial
- The edgeR user guide https://bioconductor.org/packages/edgeR
- The DESeq2 vignette https://bioconductor.org/packages/DESeq2
- The F1000 Research Bioconductor gateway https://f1000research.com/gateways/bioconductor
- Bioconductor support forum https://support.bioconductor.org

Contact me!

Email: dar2062@med.cornell.edu

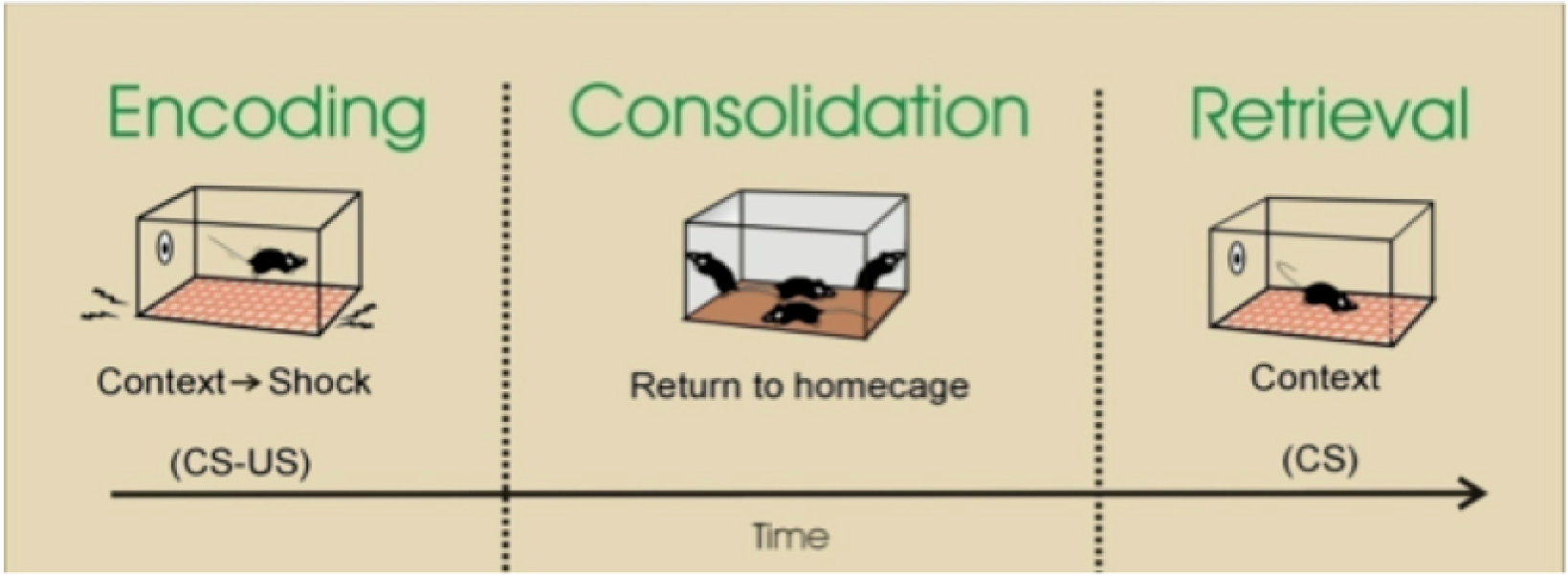
Github: https://github.com/drisso

Twitter: @drisso1893

Gene-level read counts

```
##                     CC3  CC5  CC6  CC7  CC8  FC3  FC5  FC6  FC7  FC8  RT3
## ENSMUSG00000000001 2034 2232 1253 2024 1510  994 1703 1796 1502 2145 1600
## ENSMUSG00000000028   81   93   77   91   85  106   81   84   70   95  121
## ENSMUSG00000000037   52   59   28   52   36   12   40   34   41   56   26
## ENSMUSG00000000049   15   32   15   18   14   49   10   18   11   24   21
## ENSMUSG00000000056 3125 3256 2175 3283 2553 1638 2276 2900 2223 3179 2504
## ENSMUSG00000000058 1412 1324  819 1243  668  446  821  815  786 1646  817
##                     RT5  RT6  RT7  RT8
## ENSMUSG00000000001 1734 1834 1982 1316
## ENSMUSG00000000028   92  102  102   60
## ENSMUSG00000000037   44   46   40   45
## ENSMUSG00000000049   22   17   11    9
## ENSMUSG00000000056 3045 3106 3441 1940
## ENSMUSG00000000058  945 1031 1170  990
```

An example dataset



		Dissections					Dissections
	10 am	10:30 am	4 pm	10 pm	10 am	10 am	10:30 am
Fear Conditioning		FC30	FC4	FC12	FC24	Test	RT30
Control		CC30	CC4	CC12			

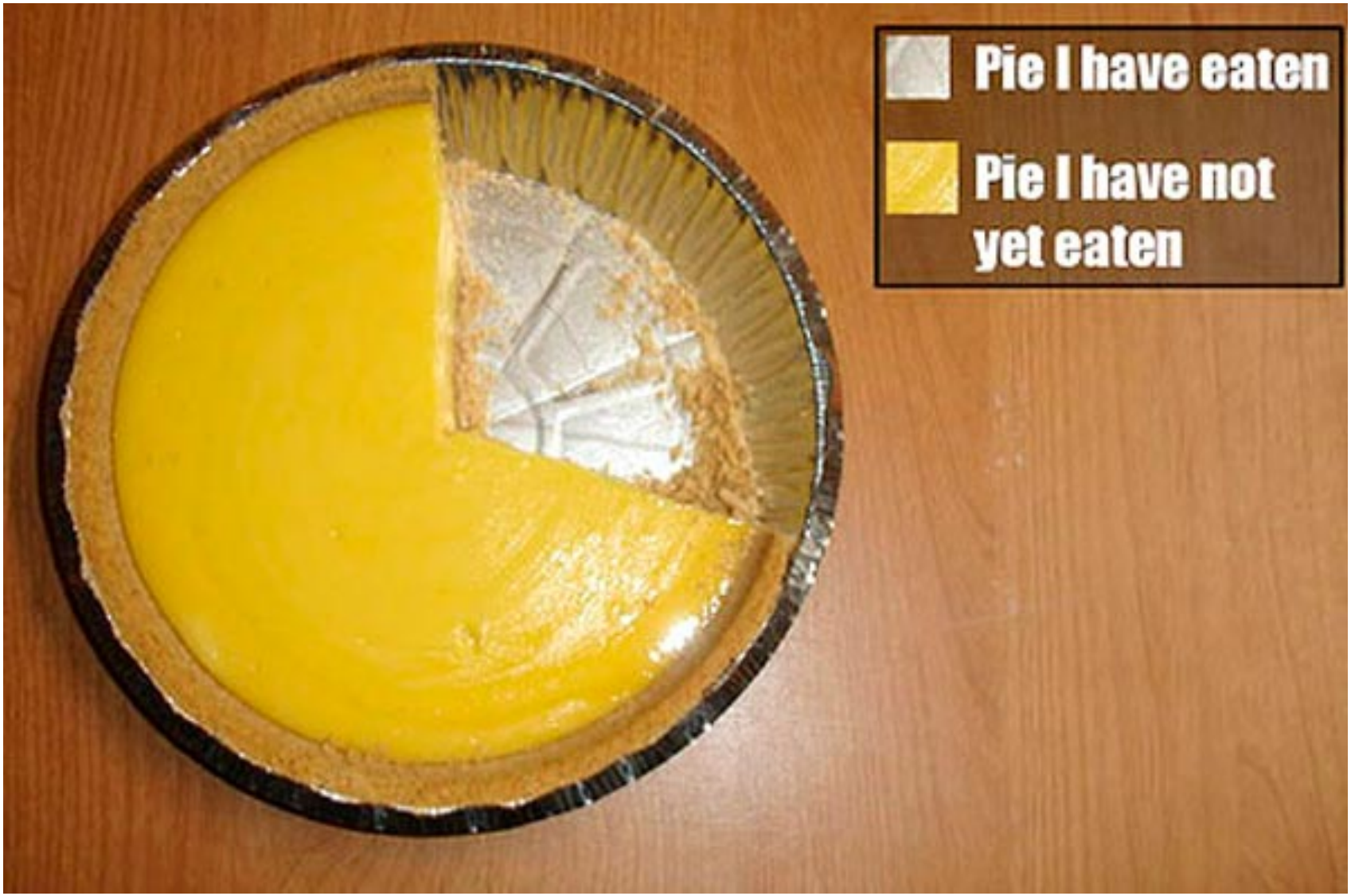
An example dataset

- C57BL/6J adult male mice (2 months of age).
- Five animals per group: fear conditioning (FC), memory retrieval (RT), and controls (CC).
- Illumina 100bp paired-end reads mapped to the mouse genome (mm9) using GMAP/GSNAP.
- Ensembl (release 65) gene counts obtained using HTSeq.

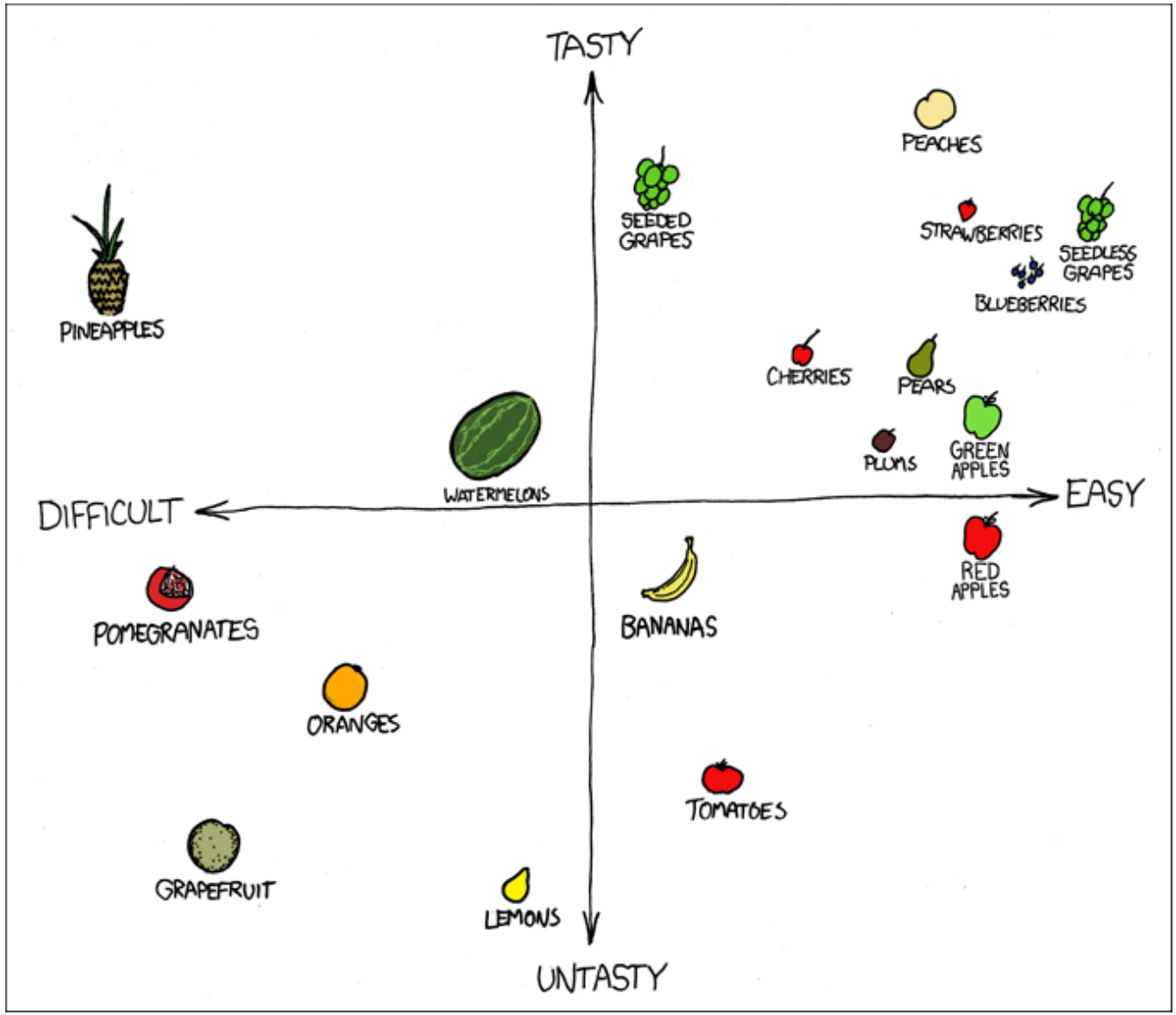
Peixoto et al. (2015). NAR.

Exploratory Data Analysis

Exploratory Data Analysis (EDA)



Exploratory Data Analysis (EDA)



Exploratory Data Analysis (EDA)

Key step of any data analysis and of statistical practice in general.

Examine dataset to

- get a *"first impression"* of the data
- reveal expected and *unexpected* characteristics of the data.
- reveal *outlying observations*
- check plausibility of the *assumptions*

EDA of Gene Expression Data

We will cover two important graphical summaries of the data, extremely useful for EDA.

- Relative Log Expression (RLE) plots
- Principal Component Analysis (PCA)

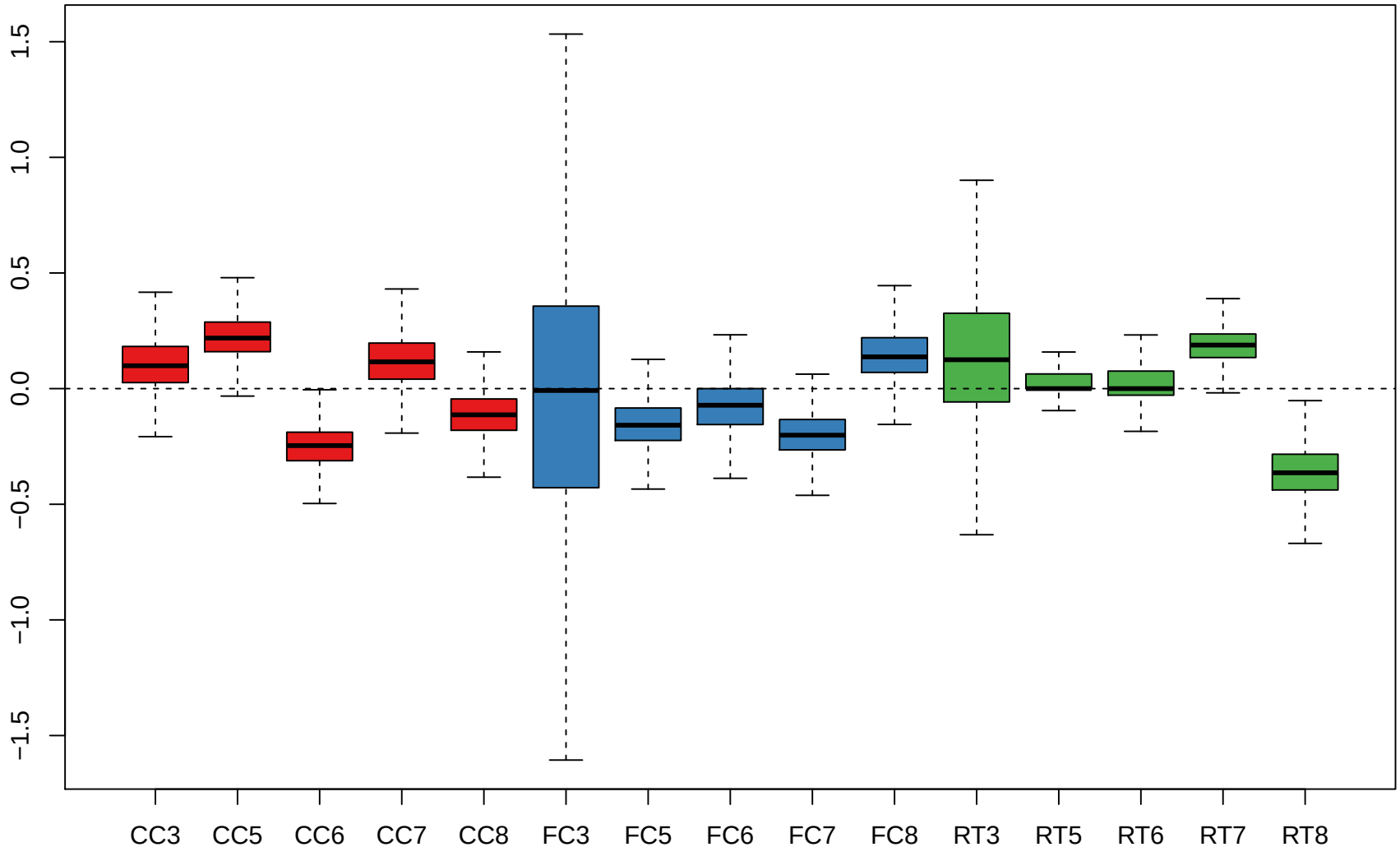
Relative Log Expression (RLE)

For each gene

1. Compute the *median* count *across all samples*.
2. Take the *log-ratio* of the read count to the median.
3. *Visualize* the distribution across all genes.

Comparable samples should have *similar RLE distributions* centered around zero.

Example: RLE plots



Dimensionality Reduction

Number of genes (variables): $J \approx 20{,}000$.

Number of samples (sample size): $n \ll J$.

Dimensionality reduction: representing the data using *fewer than* J variables.

Useful for *summarizing* and *visualizing* the data.

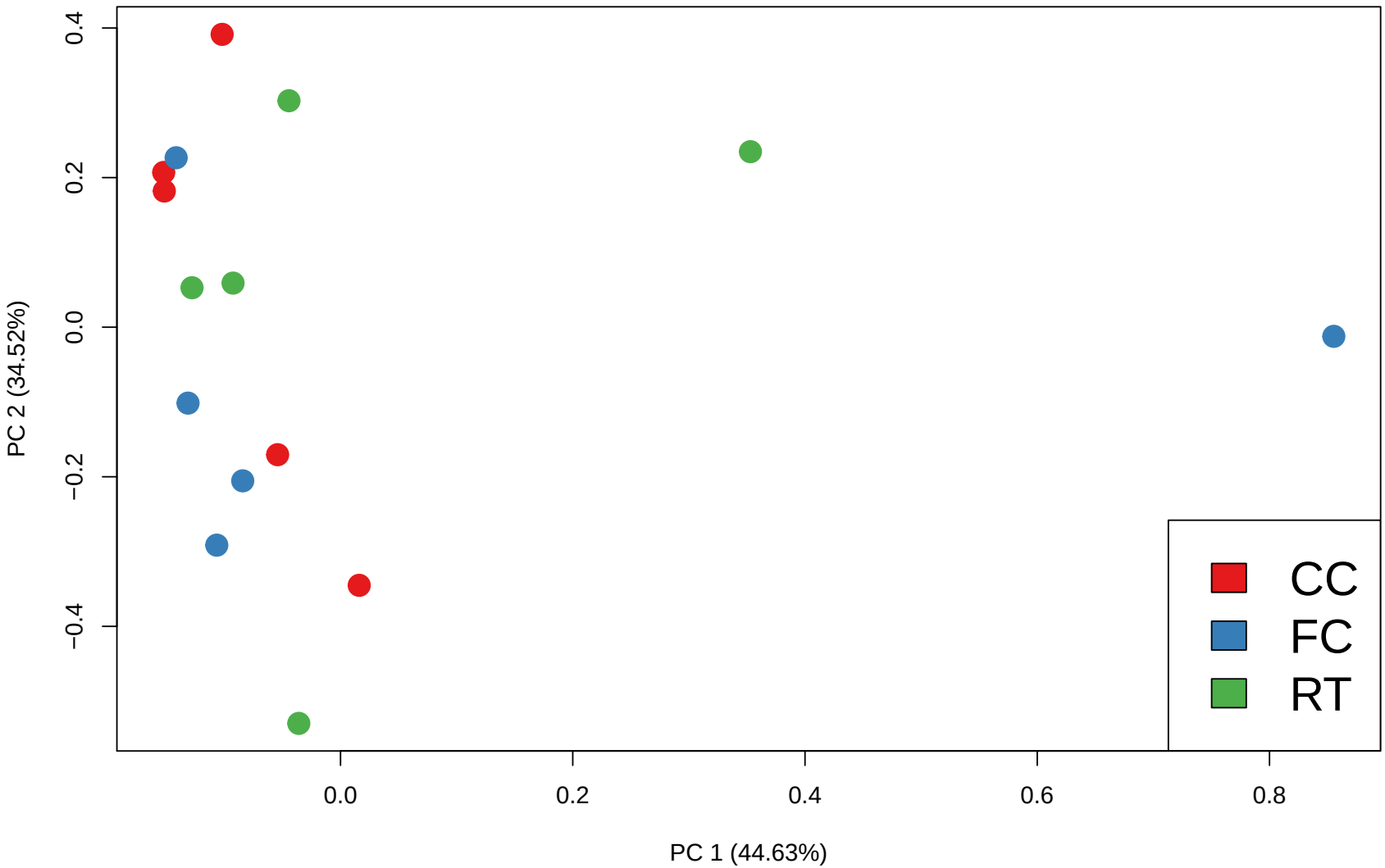
Principal Component Analysis (PCA)

Principal components are *linear combinations* of the original variables, that are

- *orthogonal* and
- have successively *maximal variance*.

Such linear combinations seek to "separate out" the observations, while loosing as little information as possible.

Example: PCA



Normalization

Sequencing and systematic biases

RNA-seq experiments are *inherently stochastic*: reads are *randomly sampled* from a pool of amplified cDNA molecules.

Our interest: the expression level of each gene.

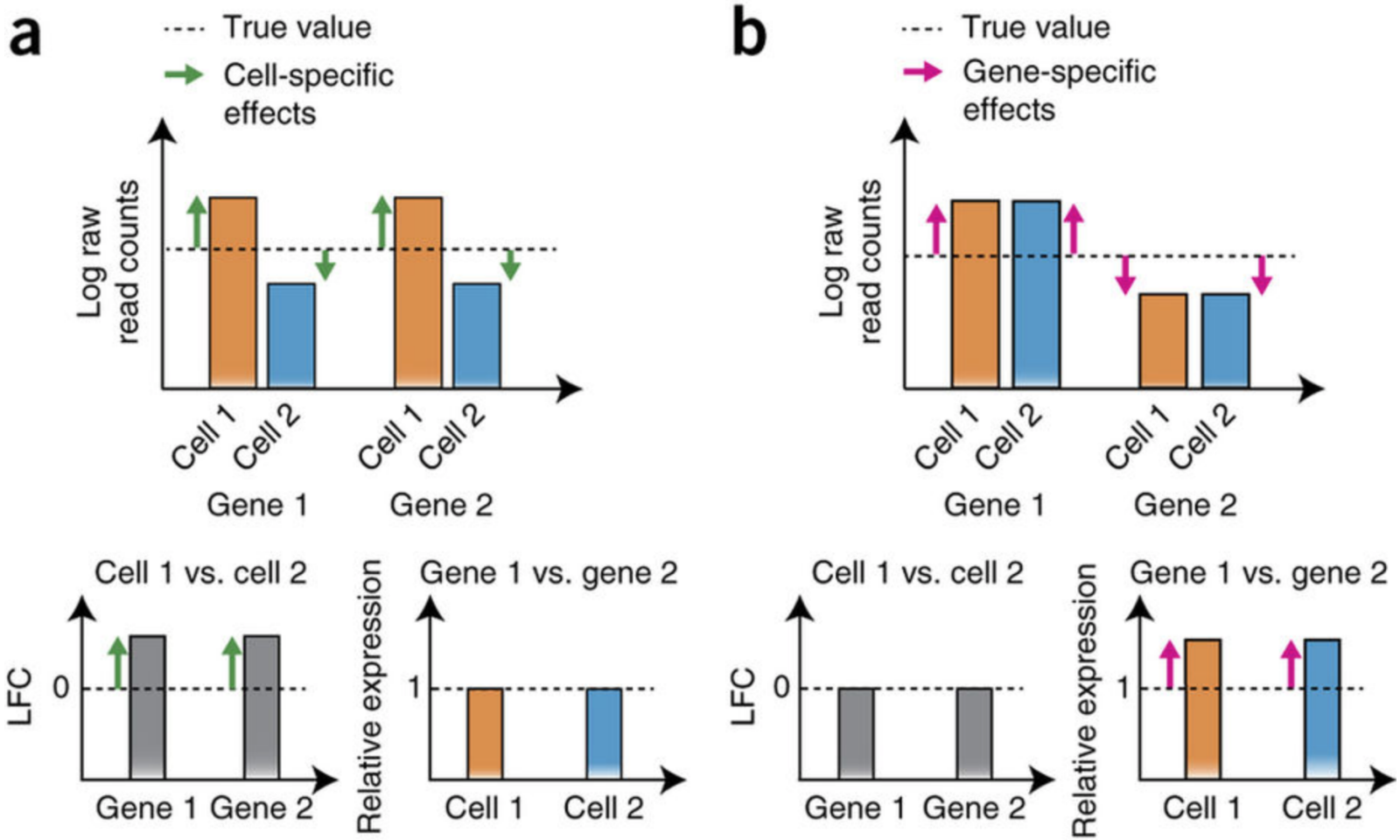
More specifically: the *relative abundance* of mRNA molecules for a gene within the population of mRNA molecules in each sample.

Sequencing and systematic biases

Several experimental sources of systematic biases affect measurements of gene expression.

We need to *normalize* the data for gene- and sample-specific biases.

Gene- and Sample-specific effects



Vallejos et al. (2017). Nature Methods

Normalization

We distinguish between two types of normalization:

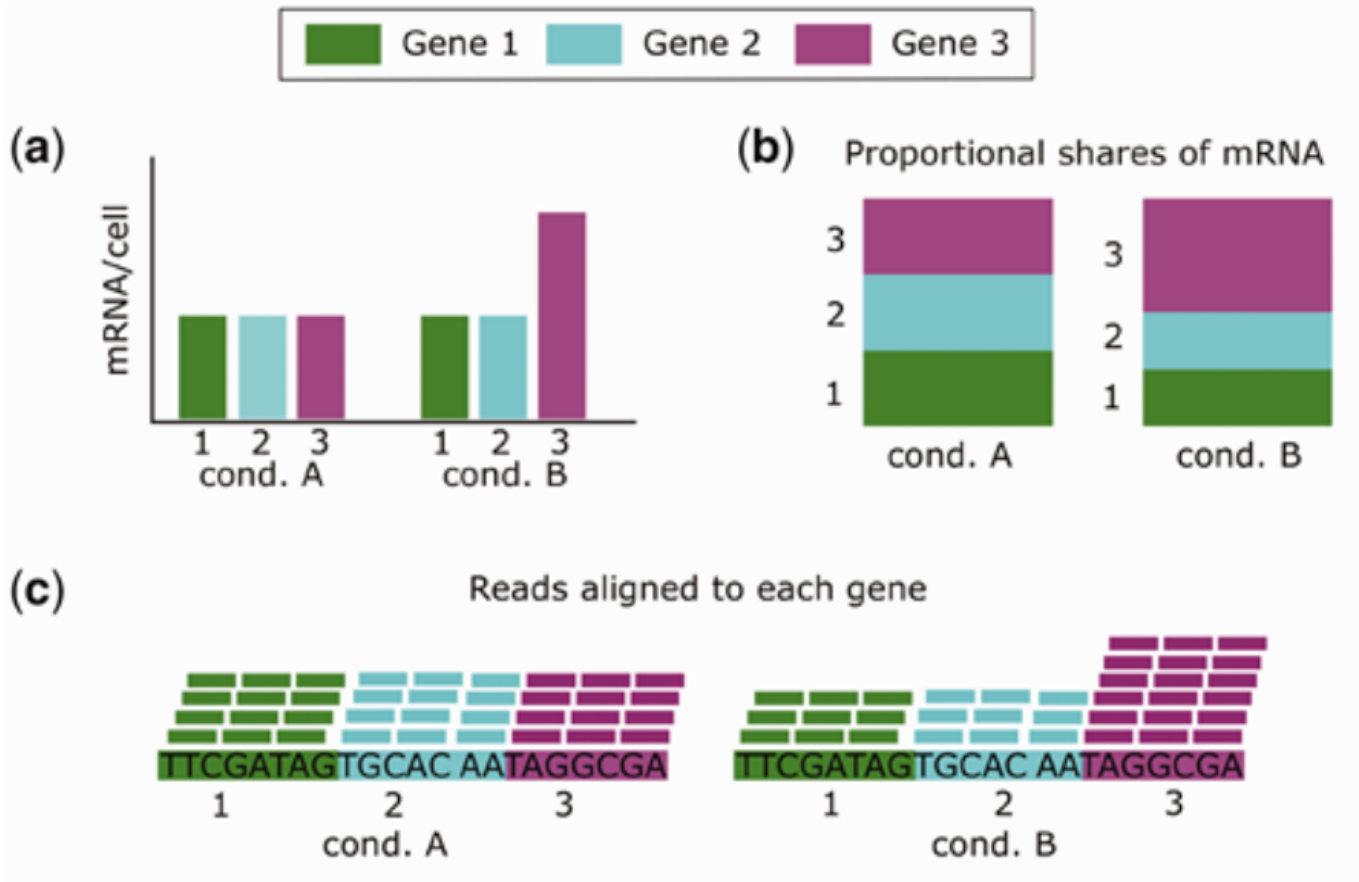
- *within-sample normalization* removes gene-specific biases (e.g., due to GC-content)
- *between-sample normalization* adjusts for effects related to distribu- tional differences in read counts between samples.

Between-sample normalization

The simplest approach is to simply *divide each count by the total number of reads in the sample*.

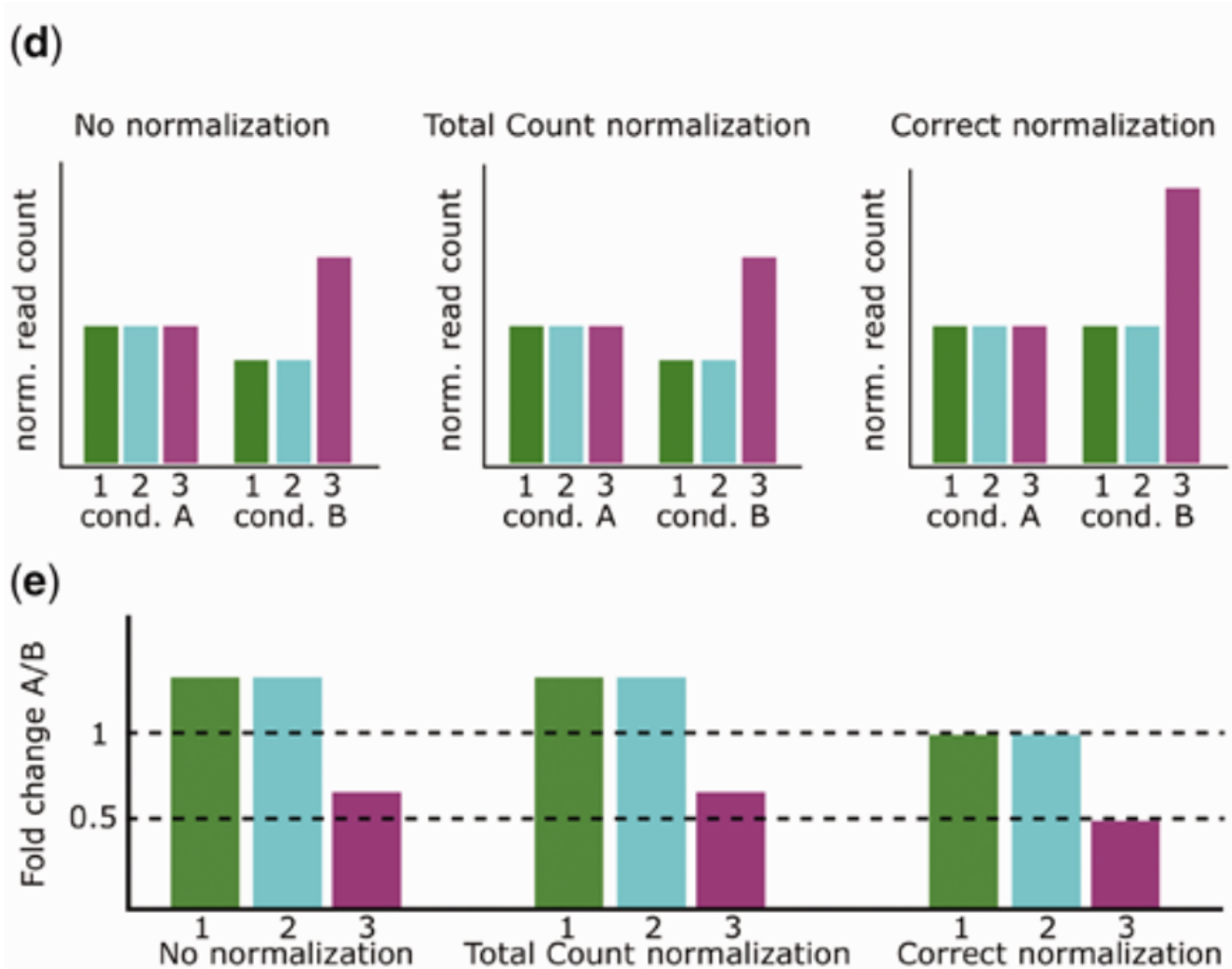
This is sometimes called *total count normalization* and is employed in popular summaries such as *FPKM* and *TPM*.

Between-sample normalization



Evans et al. (2017). Briefings in Bioinformatics.

Between-sample normalization



Evans et al. (2017). Briefings in Bioinformatics.

Between-sample normalization

In the context of differential expression, two approaches are the most popular and have been demonstrated to work well in practice.

Geometric mean scaling. Default in the *DESeq2* package.

Trimmed Mean of M-values (TMM). Default in the *edgeR* package.

Both approaches compute *normalization factors* by comparing the samples to a *reference sample*.

Normalization factors are then adjusted to multiply to 1.

Geometric mean scaling

Reference sample: the geometric mean of all samples.

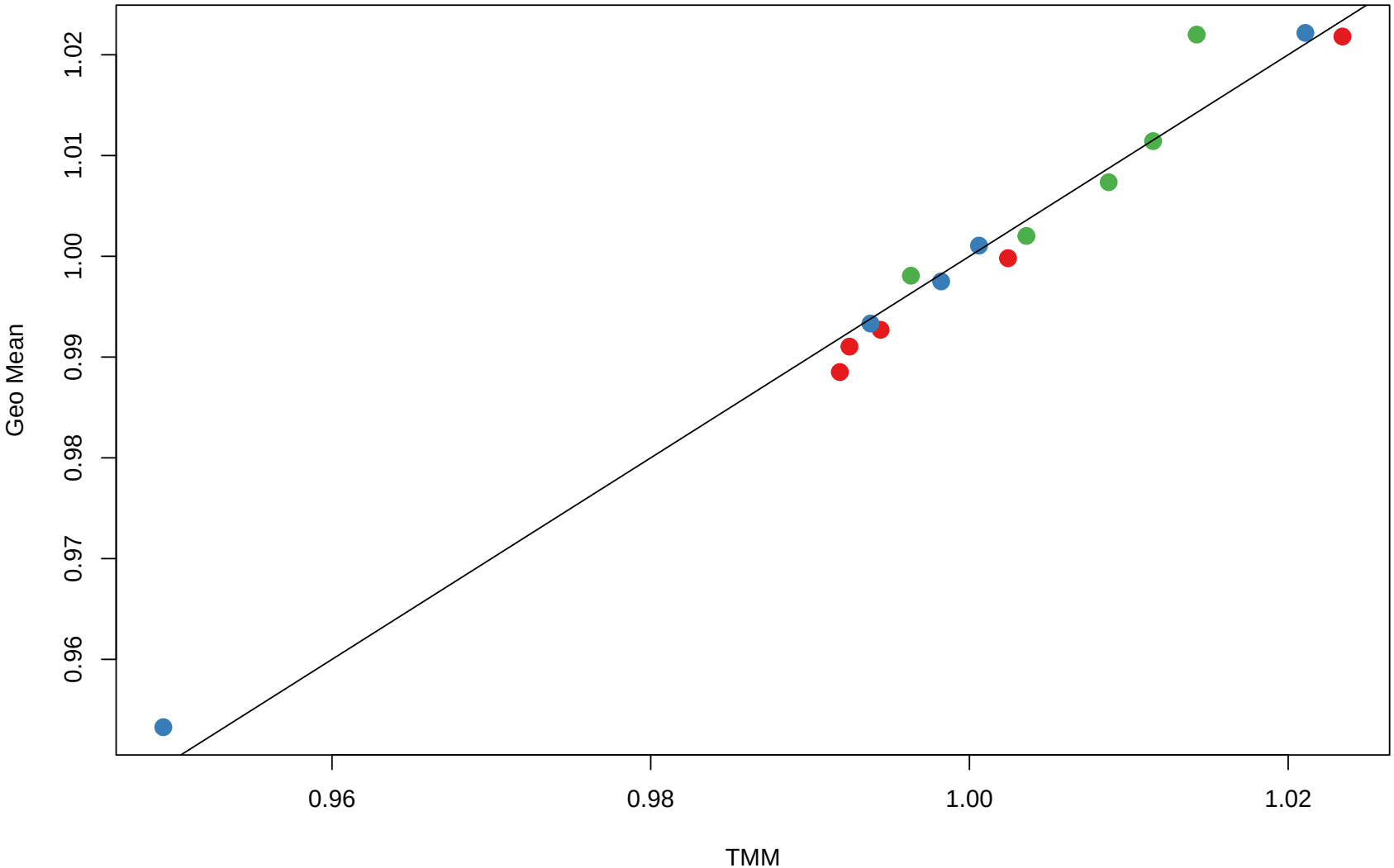
Normalization factors are computed by the *median ratio* of each sample to the reference sample.

TMM

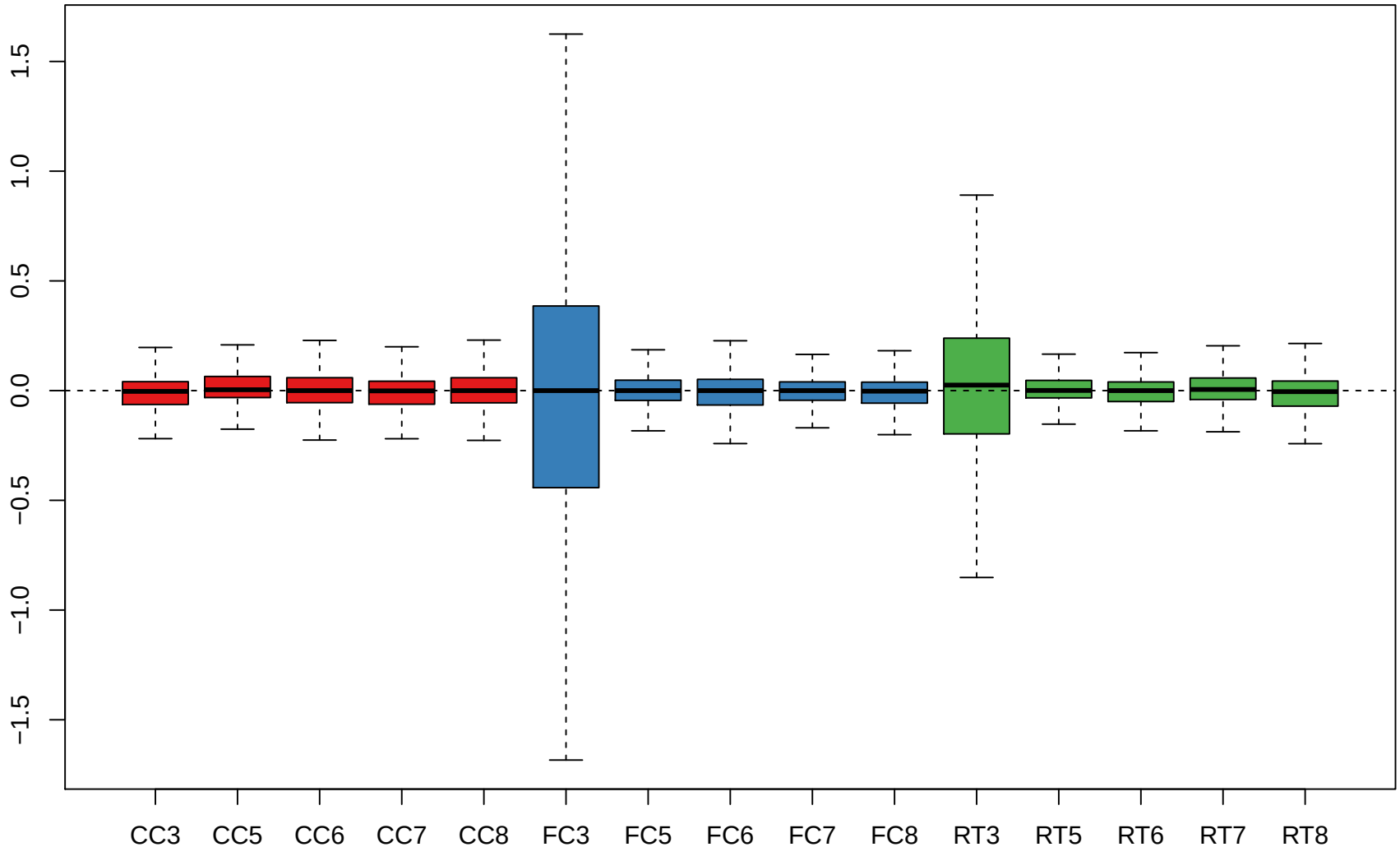
Reference sample: the sample whose upper quartile is closest to the mean upper quartile.

Normalization factors are computed by the *weighted mean log-ratio* of each sample to the reference sample, after trimming away genes with extreme log-ratios.

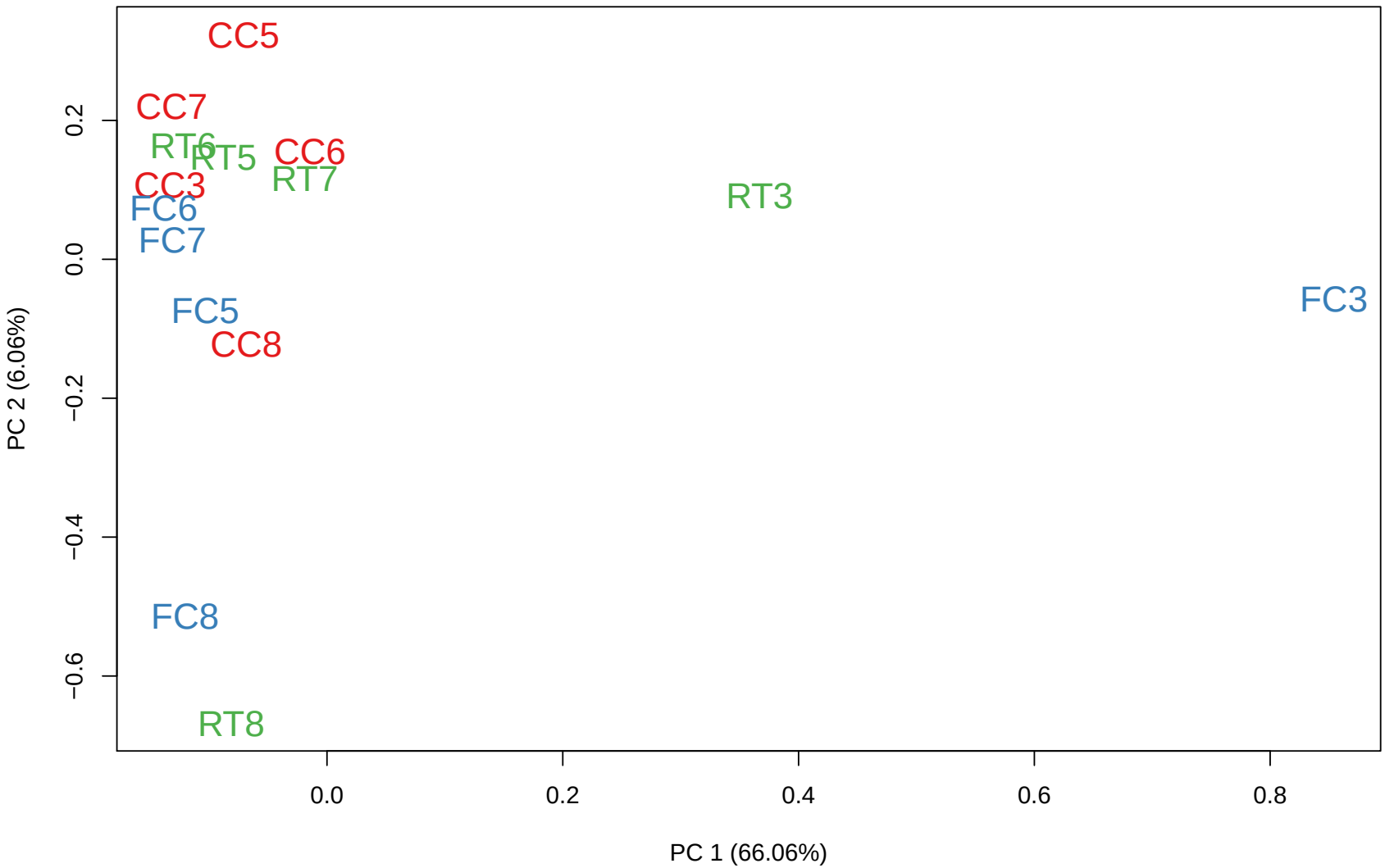
Example: normalization factors



Evaluating effectiveness of normalization



Evaluating effectiveness of normalization



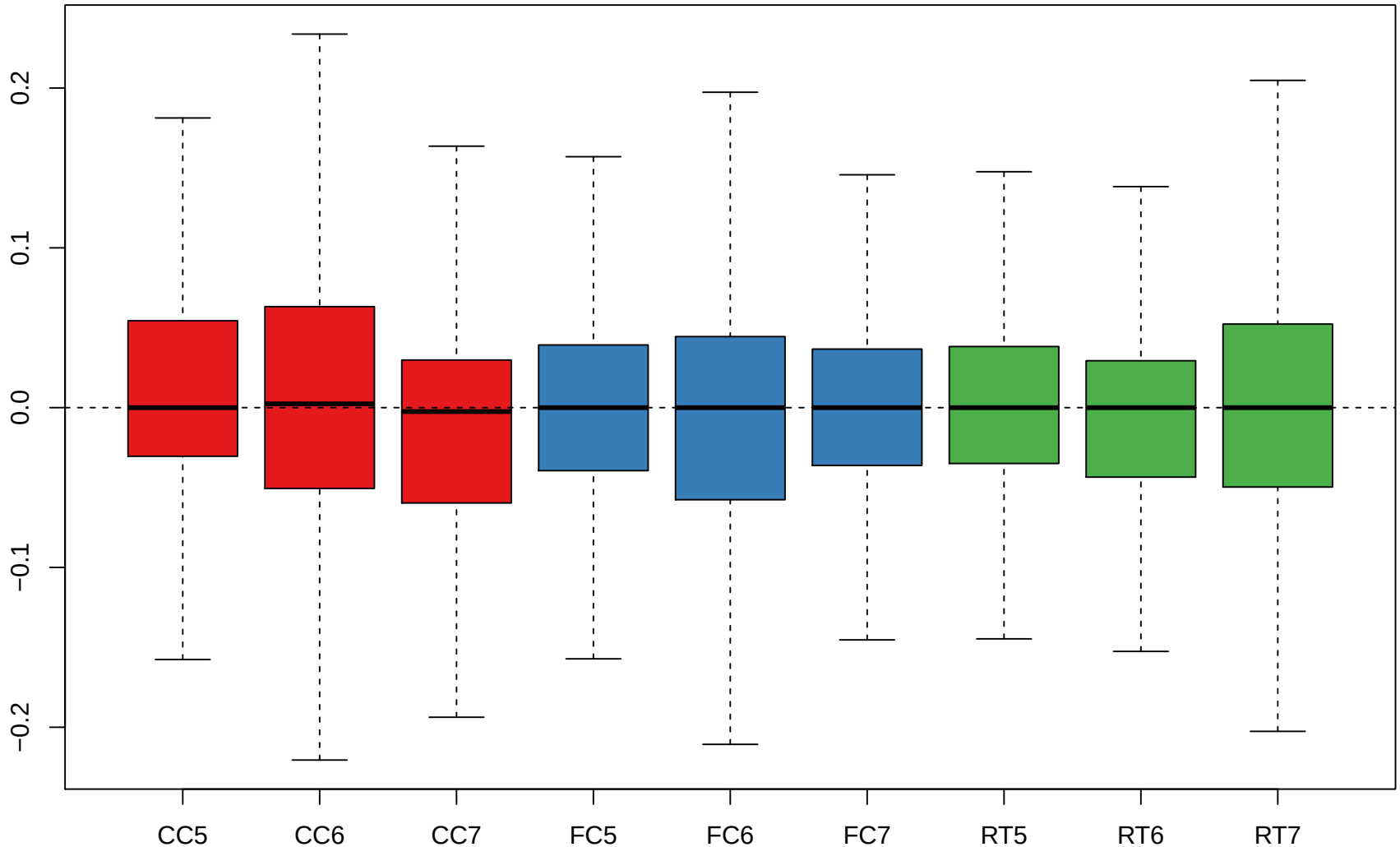
Evaluating effectiveness of normalization

Even after normalization, the distribution of the RLE and the PCA are not great.

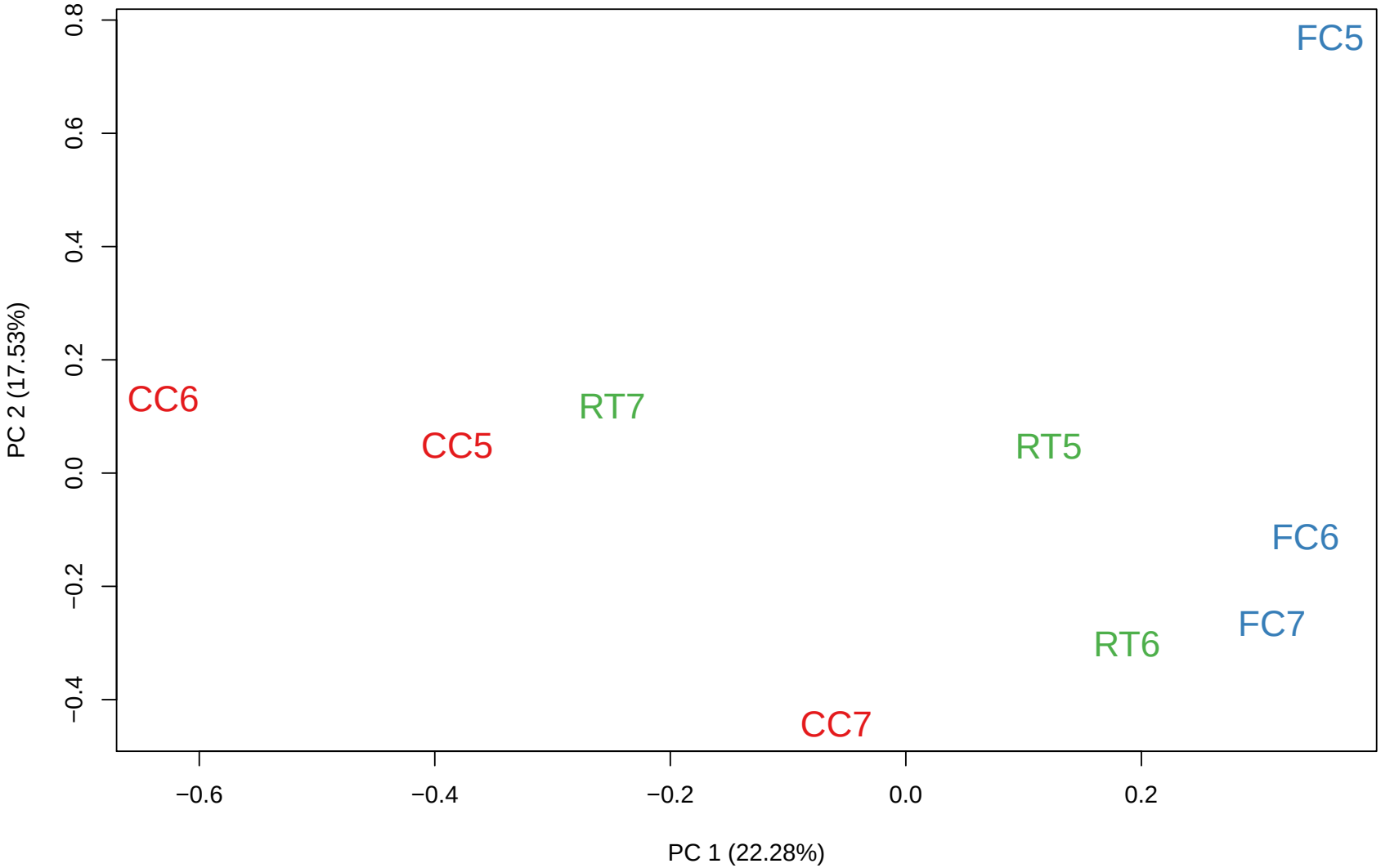
From the PCA we see that experiments carried out at day 3 and 8 are not quite like the others.

We may decide to exclude these samples from the analysis.

Evaluating effectiveness of normalization



Evaluating effectiveness of normalization



Differential Expression

Differential Expression (DE)

Goal: Find genes which are *significantly different* between the conditions.

What do we mean by *significantly* different?

Statistical significance can be evaluated using *hypothesis testing*.

For simplicity, let's assume for now that we want to compare two groups.

Statistical tests of hypothesis

Hypothesis testing is a formal statistical procedure in which we test which of *two mutually exclusive hypotheses are true*.

Importantly, the two hypotheses are not treated equally, but each has a different role.

The *null hypothesis*, H_0, represents the *status quo*, or the expected result that we want to accept as true if the data provides no convincing evidence against it.

The *alternative hypothesis*, H_1, is a statement about an effect that we want to prove.

Hypothesis testing in RNA-seq

In RNA-seq, we want to simultaneously test tens of thousands of hypotheses (one for each gene).

For each gene, the hypotheses are:

- H_0: the gene is expressed at the same level in both groups.
- H_1: the gene is *differentially expressed* between the two groups.

Assumptions

Needed in any statistical model.

We need to assume a *data generating distribution*.

For instance, if we assume that the data are distributed as a Gaussian random variable, we can use *linear regression models* and *t-tests*.

Nature of RNA-seq data

For each gene, we *count* how many reads can be mapped to a particular sequence.

The Gaussian distribution is a very general model to describe *continuous data*.

Since we have counts, we will need an alternative distribution.

The Poisson Model

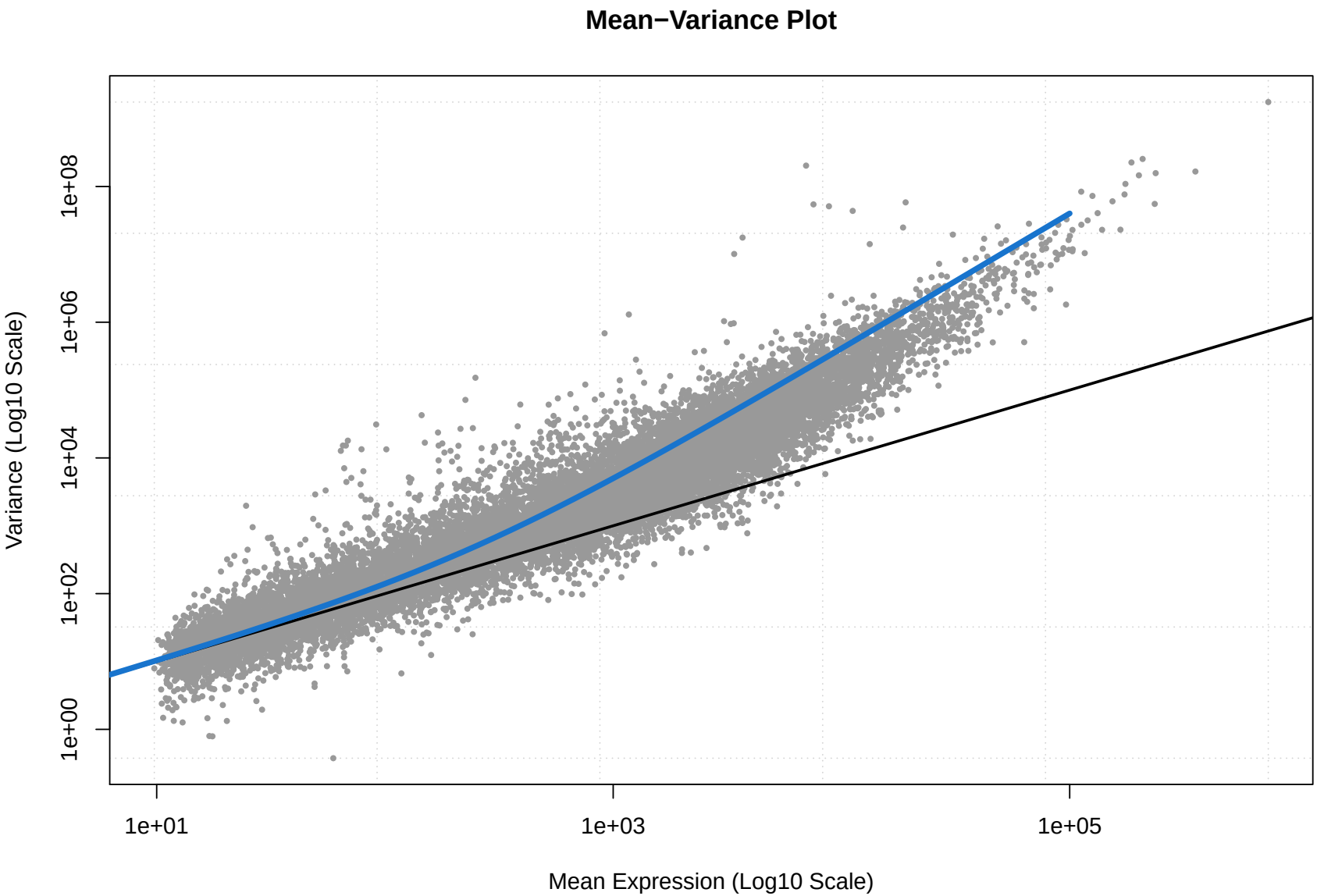
The Poisson distribution naturally arises from binomial calculations, with a large number of trials and a small probability.

However, it has a rather stringent assumption: **the variance is equal to the mean**!

$$Var(Y_{ij}) = \mu_{ij}$$

In real datasets the variance is greater than the mean, a condition known as **overdispersion**.

A real example



The Negative Binomial Model

A generalization of the Poisson model is the negative binomial, that assumes that the variance is a quadratic function of the mean.

$$Var(Y_{ij}) = \mu_{ij} + \phi_j \mu_{ij}^2$$

where ϕ is called the **dispersion parameter**.

Both `edgeR` and `DESeq2` assume that the data is distributed as a negative binomial.

The Negative Binomial Model

If we want to use the negative binomial model to fit our data, we need to *estimate* two parameters:

- The dispersion parameter ϕ_j
- And the mean μ_{ij}

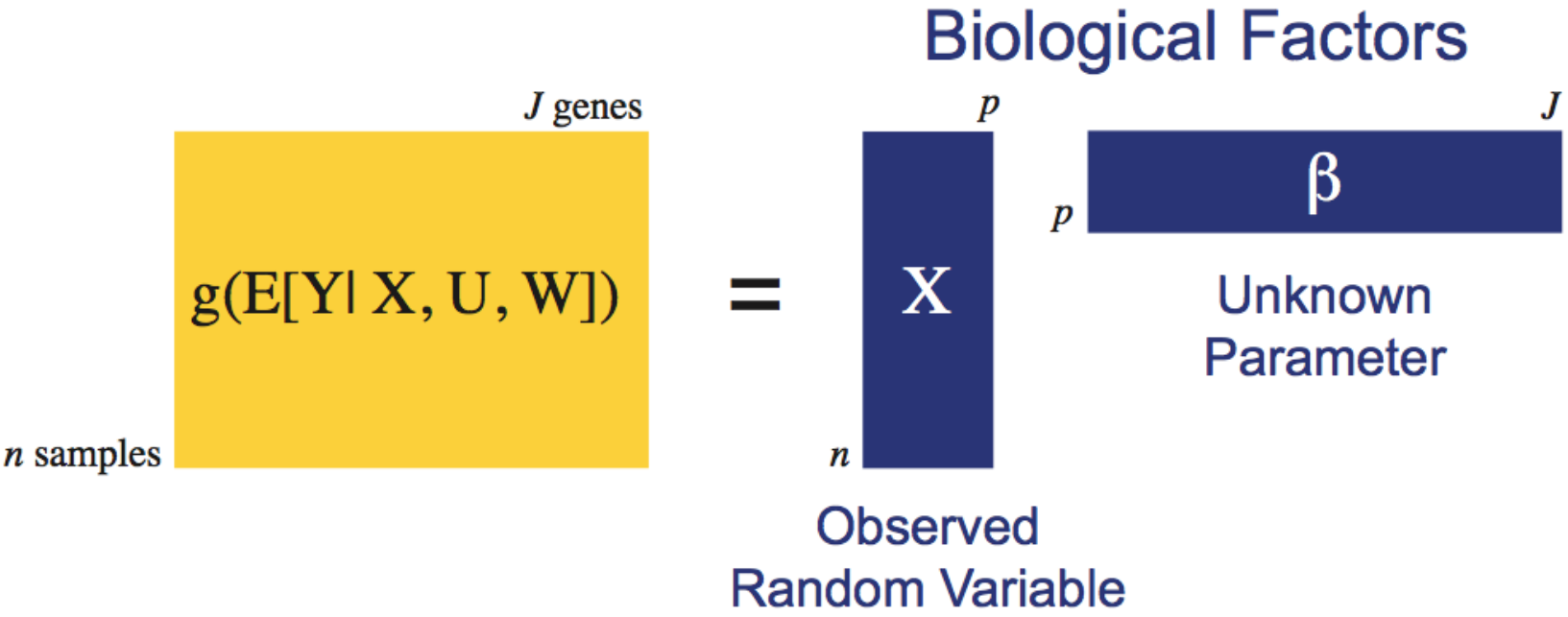
After estimating ϕ_j, we can use a *generalized linear model* (GLM) to estimate the mean and test for the difference between the groups.

Generalized Linear Models

GLMs are a generalization of linear models for distributions other than the Gaussian.

For the negative binomial distribution, GLMs take the form of a *log-linear model*, where we model the log of the mean as a linear function of the *covariates*.

Generalized Linear Models



How to specify the design matrix

The matrix X is called *design matrix* and it includes

- An *intercept* (a vector of ones) that captures the overall mean
- The *biological factors* that we want to test
- Possibly, *confounding factors* that we need to account for in the analysis

In the simplest case of *two group comparison*, X is simply an indicator variable of the group to be compared.

When we have more than two groups, we need more than one indicator.

Example: two group comparison

```
##   Intercept Group2
## 1         1      0
## 2         1      0
## 3         1      0
## 4         1      1
## 5         1      1
## 6         1      1
```

Example: three group comparison

```
##   Intercept FC RT
## 1         1  0  0
## 2         1  0  0
## 3         1  0  0
## 4         1  1  0
## 5         1  1  0
## 6         1  1  0
## 7         1  0  1
## 8         1  0  1
## 9         1  0  1
```

Interpreting the parameters

Back to the two-group case.

```
##   Intercept Group2
## 1         1      0
## 2         1      0
## 3         1      0
## 4         1      1
## 5         1      1
## 6         1      1
```

For each gene, the β parameter has two components $\beta = [\beta_0 \, \beta_1]$.

Interpreting the parameters

For a sample in Group 1

$$\log E[Y_{ij}|X = 0] = \beta_0.$$

For a sample in Group 2

$$\log E[Y_{ij}|X = 1] = \beta_0 + \beta_1$$

Hence,

$$\beta_1 = \log \frac{E[Y_{ij}|X = 1]}{E[Y_{ij}|X = 0]}$$

Interpreting the parameters

β_1: *average log-fold-change between the two groups*.

When more than three groups: each β is the *average log-fold-change* with respect to the *reference group*.

Example: interpreting the parameters

```
## Coefficient:  FC RT
##                          logFC.FC  logFC.RT
## ENSMUSG00000021250 2.4509340 2.1558649
## ENSMUSG00000034765 0.8242325 0.8514761
## ENSMUSG00000020423 0.8752627 0.7597942
## ENSMUSG00000022602 1.6668624 1.5562573
## ENSMUSG00000037868 2.7851375 2.3486688
## ENSMUSG00000024042 1.1568160 1.1587839
## ENSMUSG00000052837 1.1005269 1.0127417
## ENSMUSG00000085609 1.2747892 1.1263603
## ENSMUSG00000023034 1.0356528 0.9021138
## ENSMUSG00000020108 0.5087313 0.5764423
```

Test for differential expression

For each gene, our test for differential expression involves the following hypotheses.

- H_0: $\beta_1 = 0$;
- H_1: $\beta_1 \neq 0$.

We can test these hypotheses by using a

- *Likelihood Ratio Test* (default in `edgeR`);
- *Wald Test* (default in `DESeq2`).

If the p-value is less than $\alpha = 0.05$ we *reject the null hypothesis*, hence declaring the gene *differentially expressed*.

Testing error

Even if the null hypothesis is true, our test can still reject it.

This is called *type I error*, i.e., to reject the null hypothesis when it is true.

Adjusting for multiple testing

Suppose that $\alpha = 0.05$, and that the null hypothesis is indeed true (no DE genes).

With two tests (genes): the probability of finding at least one (false) positive result is $1 - 0.95^2 \approx 0.1$.

With 20 genes: $1 - 0.95^{20} \approx 0.64$.

That means that we have 64% probability to find at least one DE gene when there are none!

We are testing thousands of genes!

Adjusting for multiple testing

Multiple testing procedures can be used to *adjust the p-values* for multiplicity.

Here, we briefly cover the *Benjamini-Hochberg procedure* to control the *False Discovery Rate* (FDR).

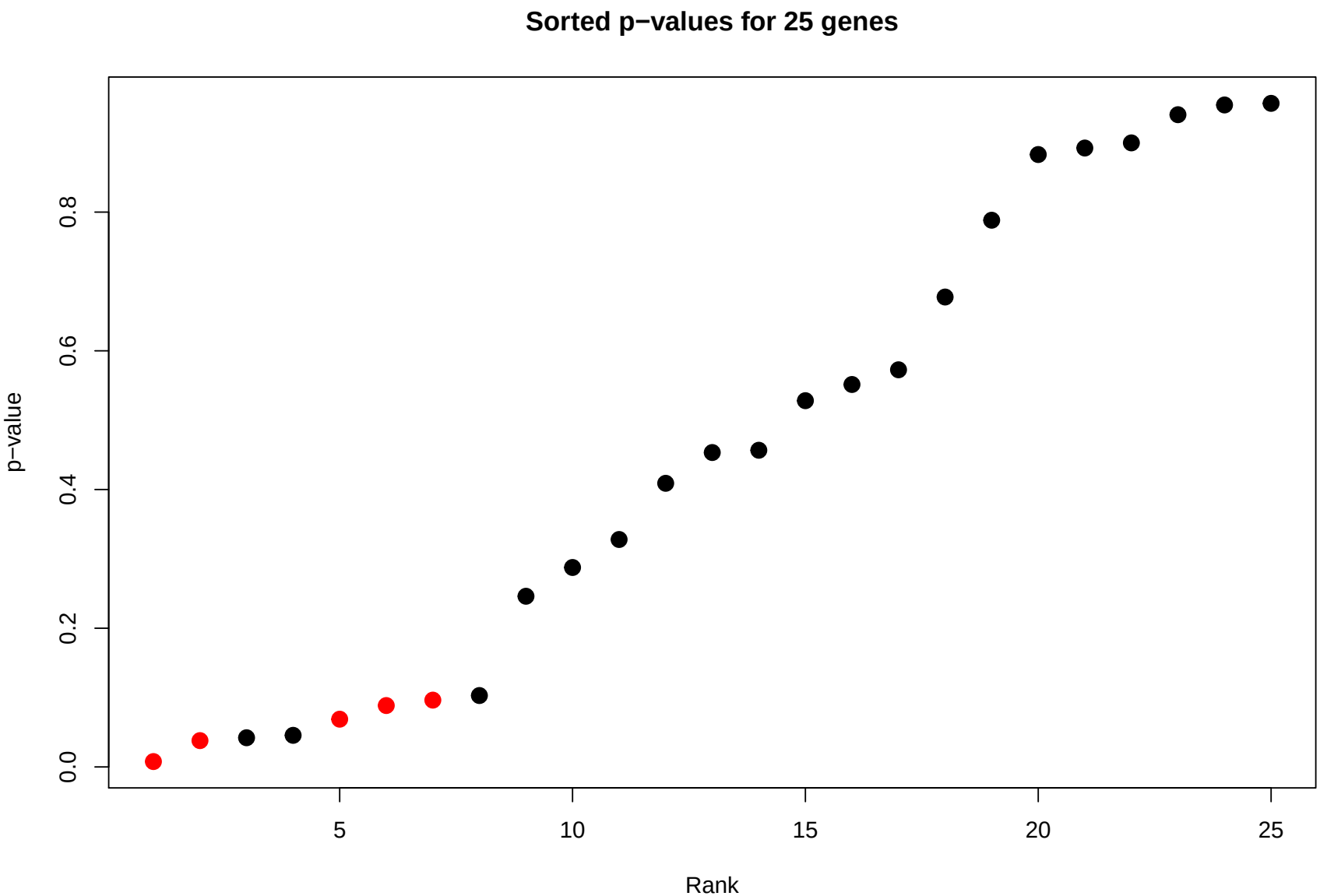
The False Discovery Rate

The False Discovery Rate (FDR) is defined as the *expected proportion of false discoveries* amongst the rejected null hypotheses.

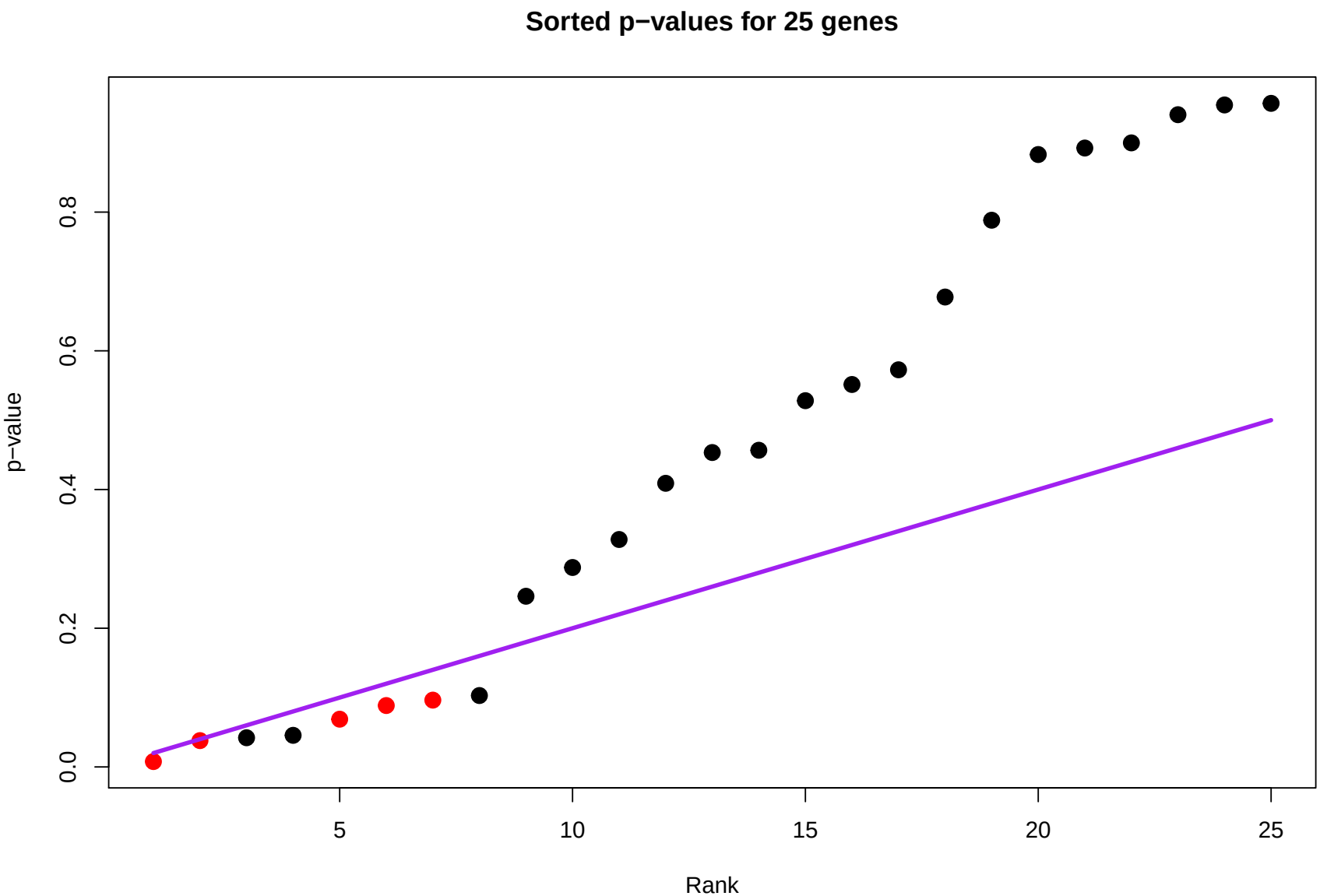
In terms of gene expression, this means the expected proportion of non-DE genes wrongly declared as DE.

We usually want to limit this proportion to 5% or 10%.

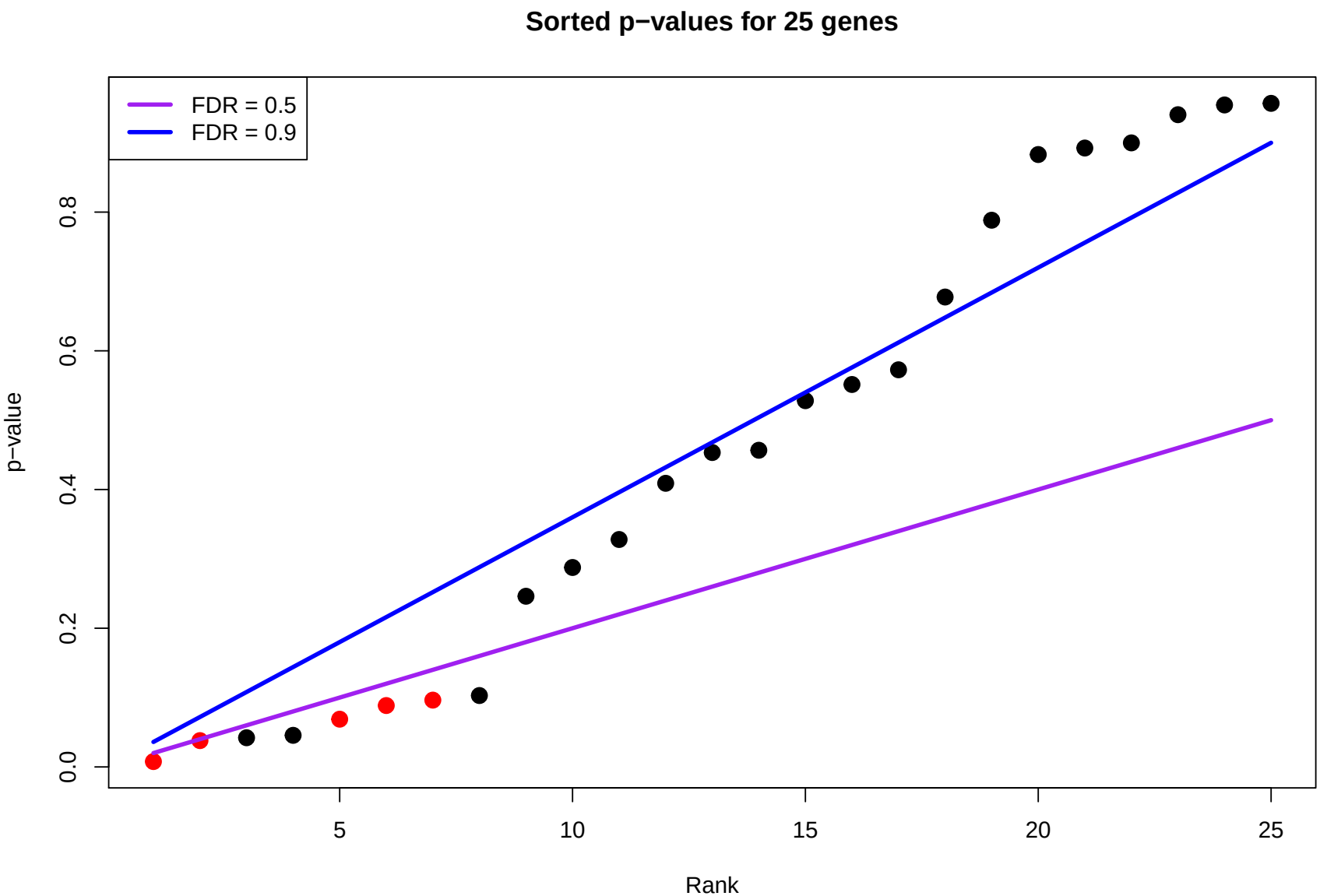
Example: FDR control



Example: FDR control



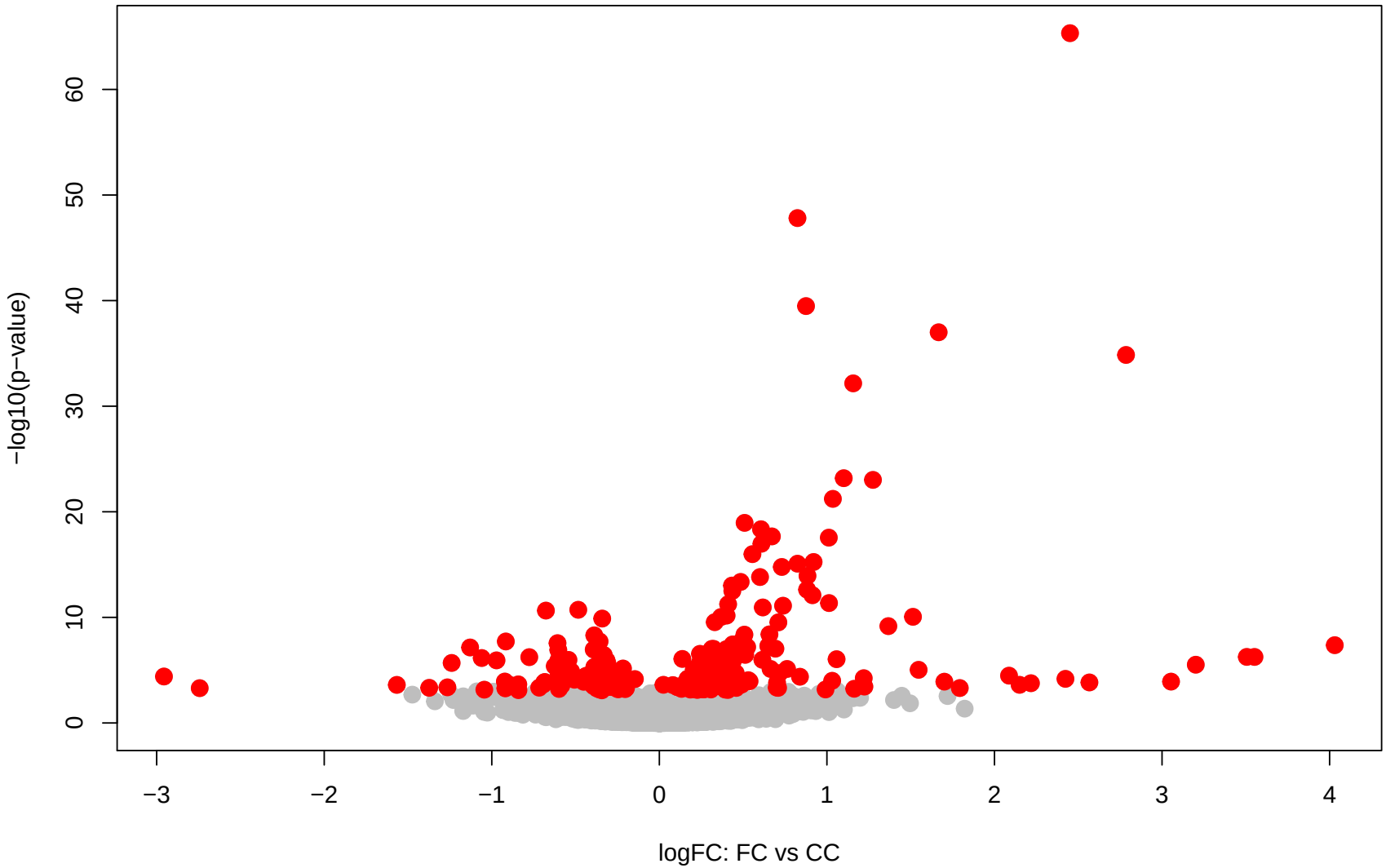
Example: FDR control



Example: Differentially Expressed Genes

```
##
## out of 17513 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 109, 0.62%
## LFC < 0 (down)     : 22, 0.13%
## outliers [1]       : 7, 0.04%
## low counts [2]     : 1358, 7.8%
## (mean count < 26)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Volcano plots



Diagnostics: the p-value histogram

To see if the model that we used correctly fit the data, we can check the distribution of the (unadjusted) p-values.

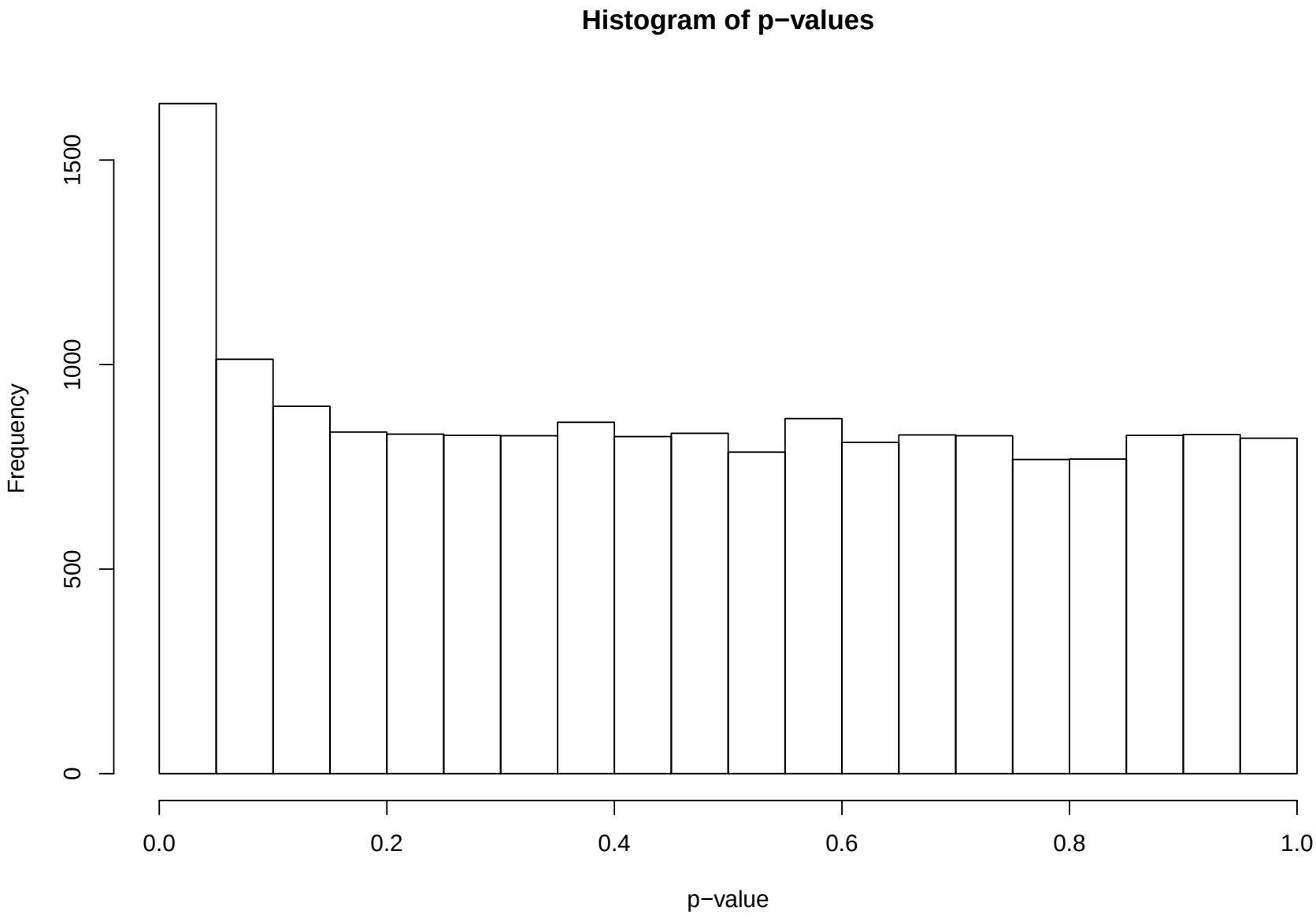
We know from theory that if all the genes are non-DE the distribution of the p-values should be uniform.

For the DE genes, we expect p-values very close to 0.

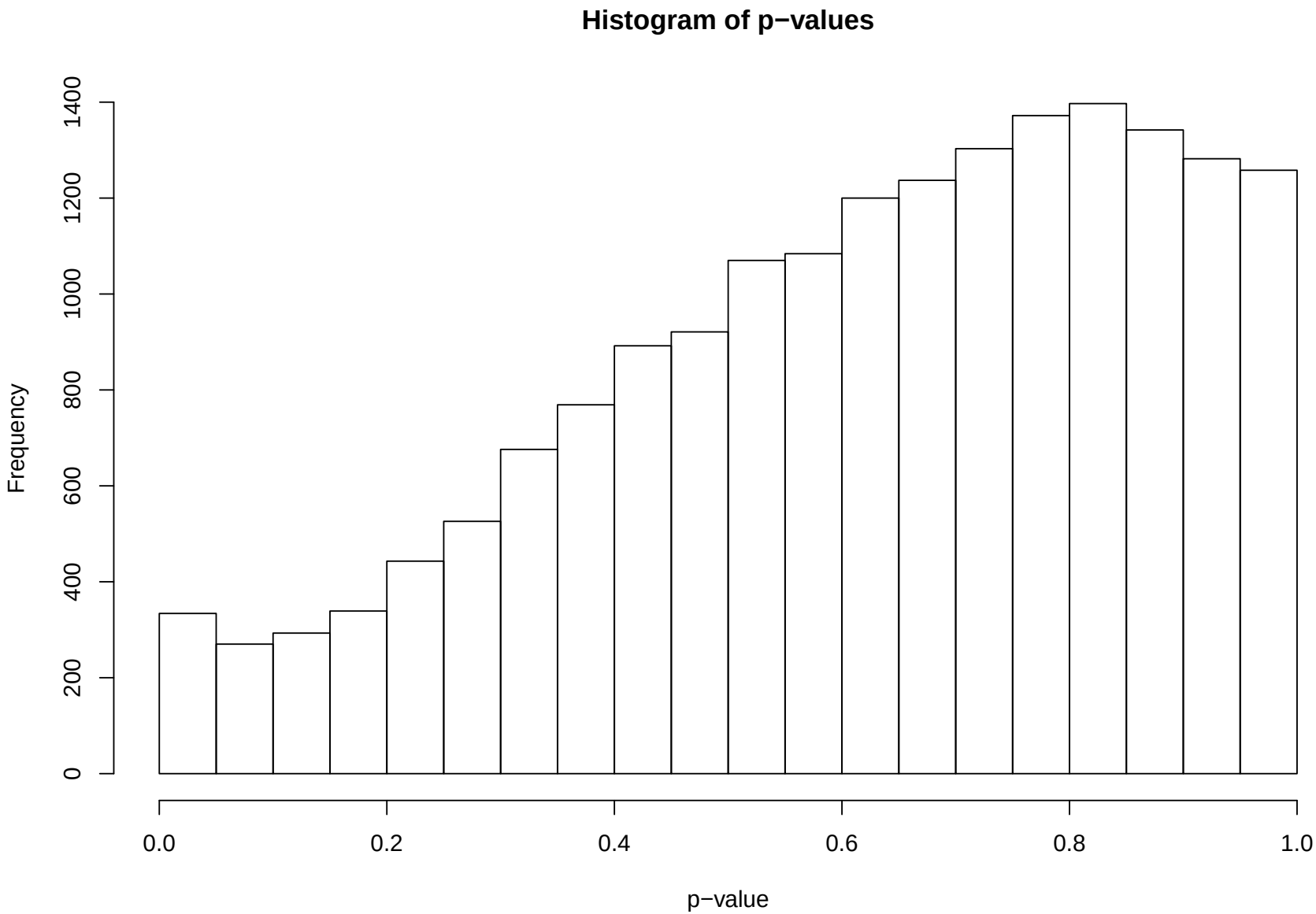
Hence, we expect to see a mixture of a uniform distribution and a "spike" at zero.

Non-uniform distributions are an indication of batch effects or other *unwanted variation* that affects the data.

Example of “good” p-value distribution



Example of “bad” p-value distribution



Batch Effects

Accounting for batch effects

As for any high-throughput genomic technology, RNA-seq is affected by complex, non-linear effects that are not removed by global scaling normalization.

These effects are collectively known as *batch effects*.

Accounting for known batch effects

Sometimes, we know (or suspect) where these effects come from.

For instance, we might have processed the samples in different days, or at a different time of day.

In large collaborative studies, samples might be processed in different labs.

Or we may have used different machines, or protocols.

In these cases, it might be enough to include the appropriate variable in the design matrix to adjust for these *known* effects.

Accounting for unknown batch effects

More often, we do not have a clear idea of which aspect of the data generation introduced the batch effects.

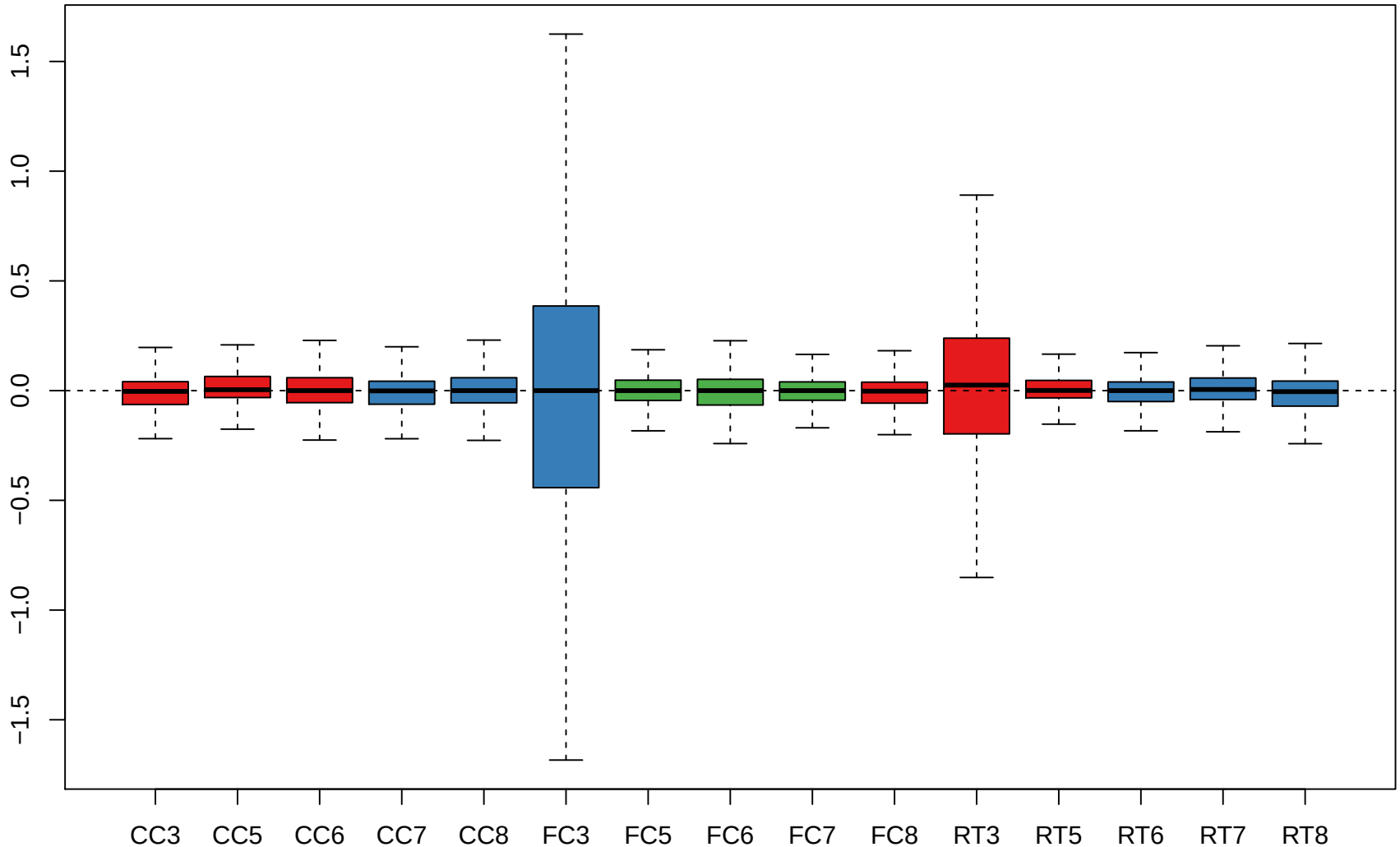
Or it may be a combination of too many factors to all include in the model.

In such cases, we can estimate such effects from the data.

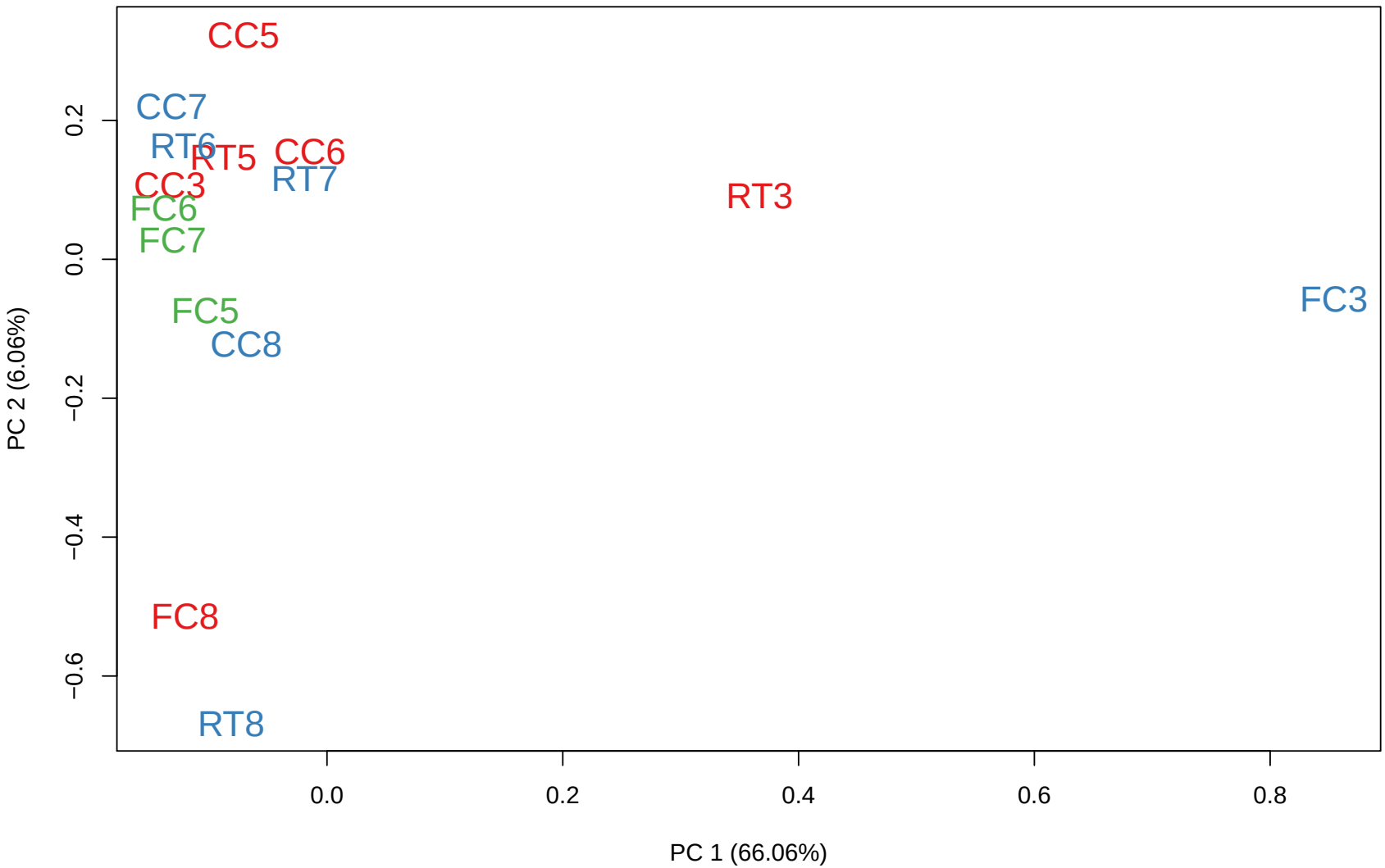
One way to do so is by using the *Remove Unwanted Variation* (RUV) approach, which uses *negative control genes* to estimate batch effects from the data.

This approach is implemented in the Bioconductor package `RUVSeq`.

Example of unwanted variation



Example of unwanted variation



Accounting for unwanted variation

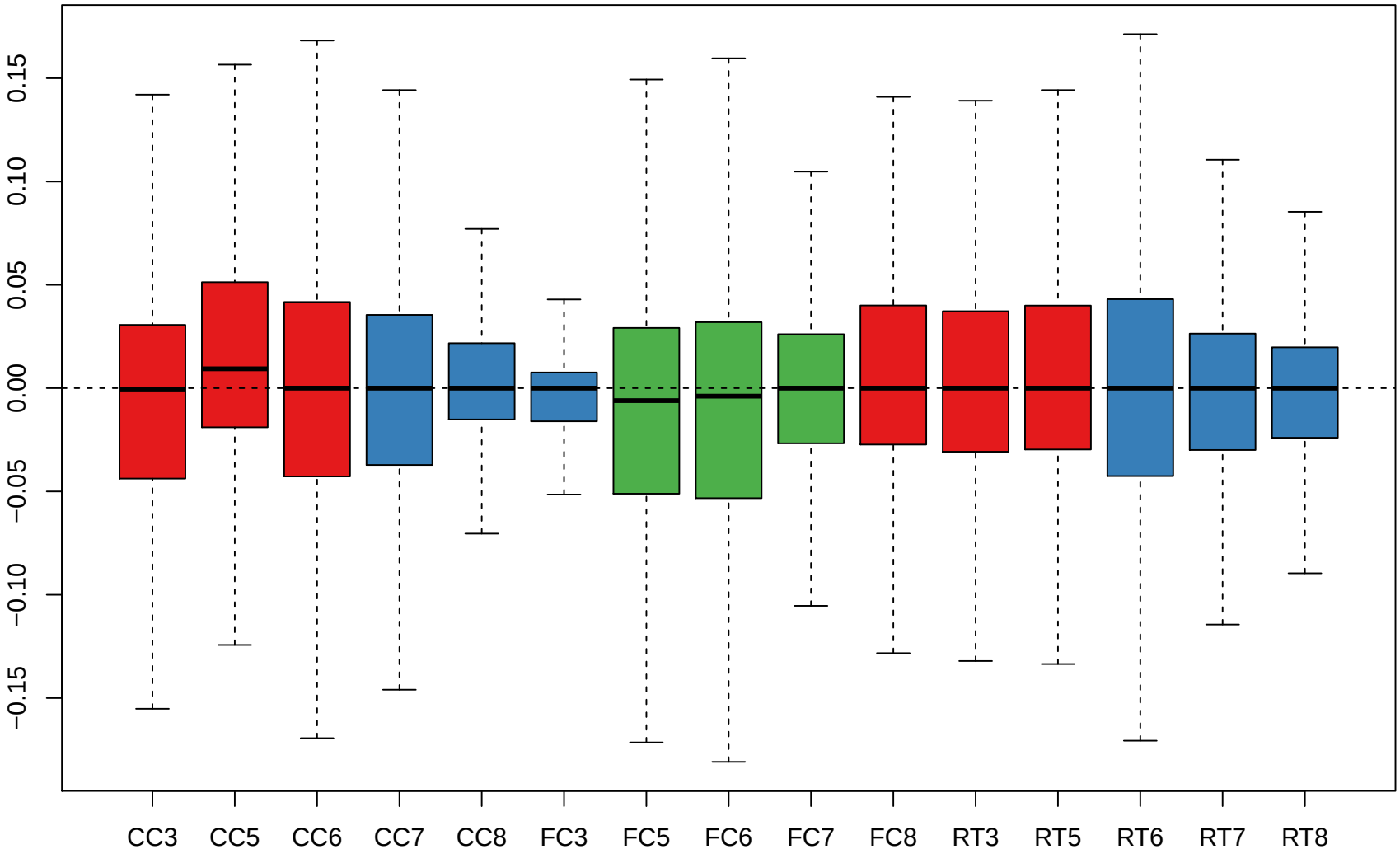
Rather than filtering out the unusual samples, we can try to adjust (or normalize) their values using the expression levels of *negative control genes*.

The assumption is that we can identify a set of negative controls that are *not affected by biology*.

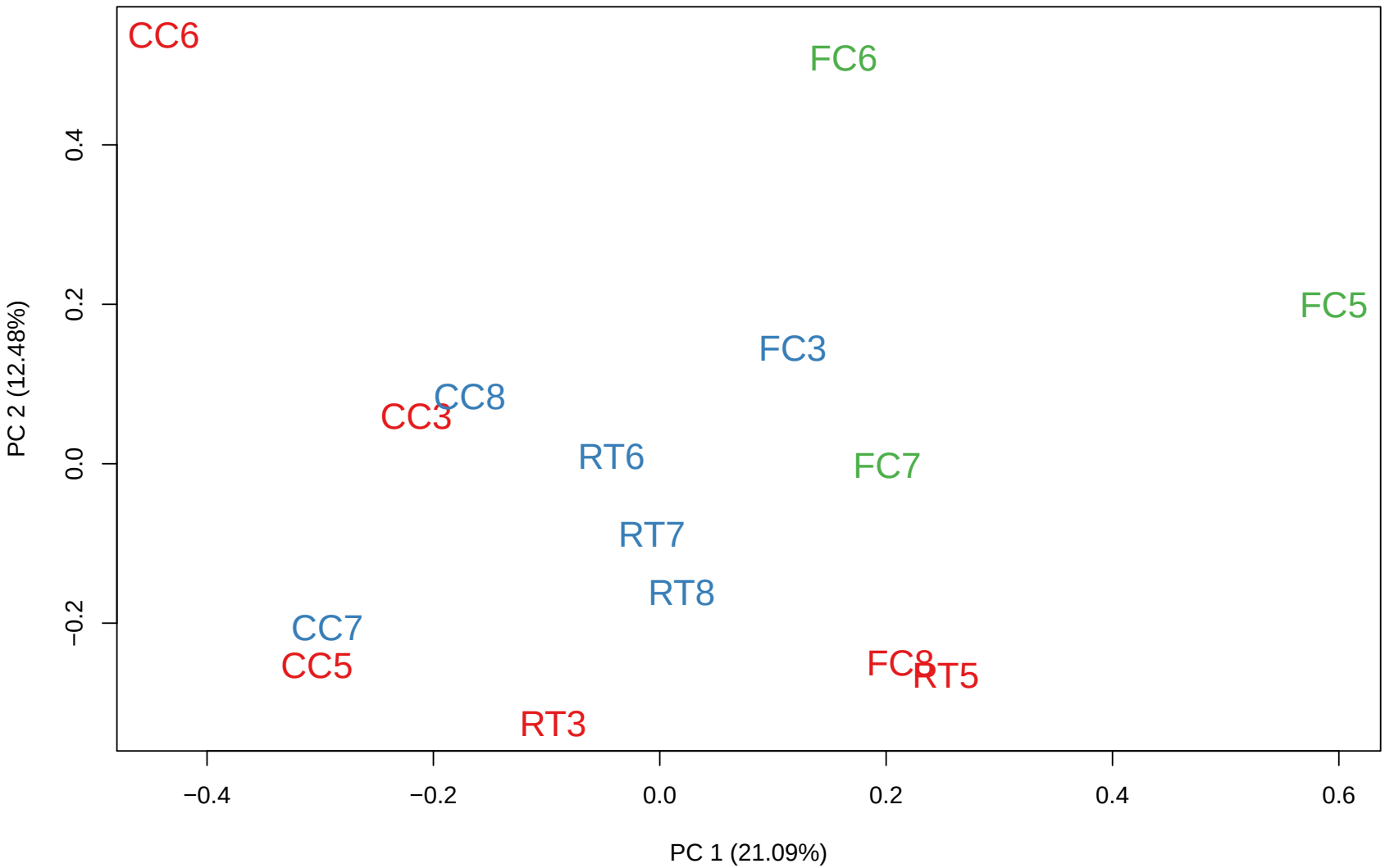
Hence, all the signal captured by these genes is *driven by unwanted variation*.

RUV estimates k factors of unwanted variation that can be included in the design matrix for the DE analysis. In this case, we used $k = 5$.

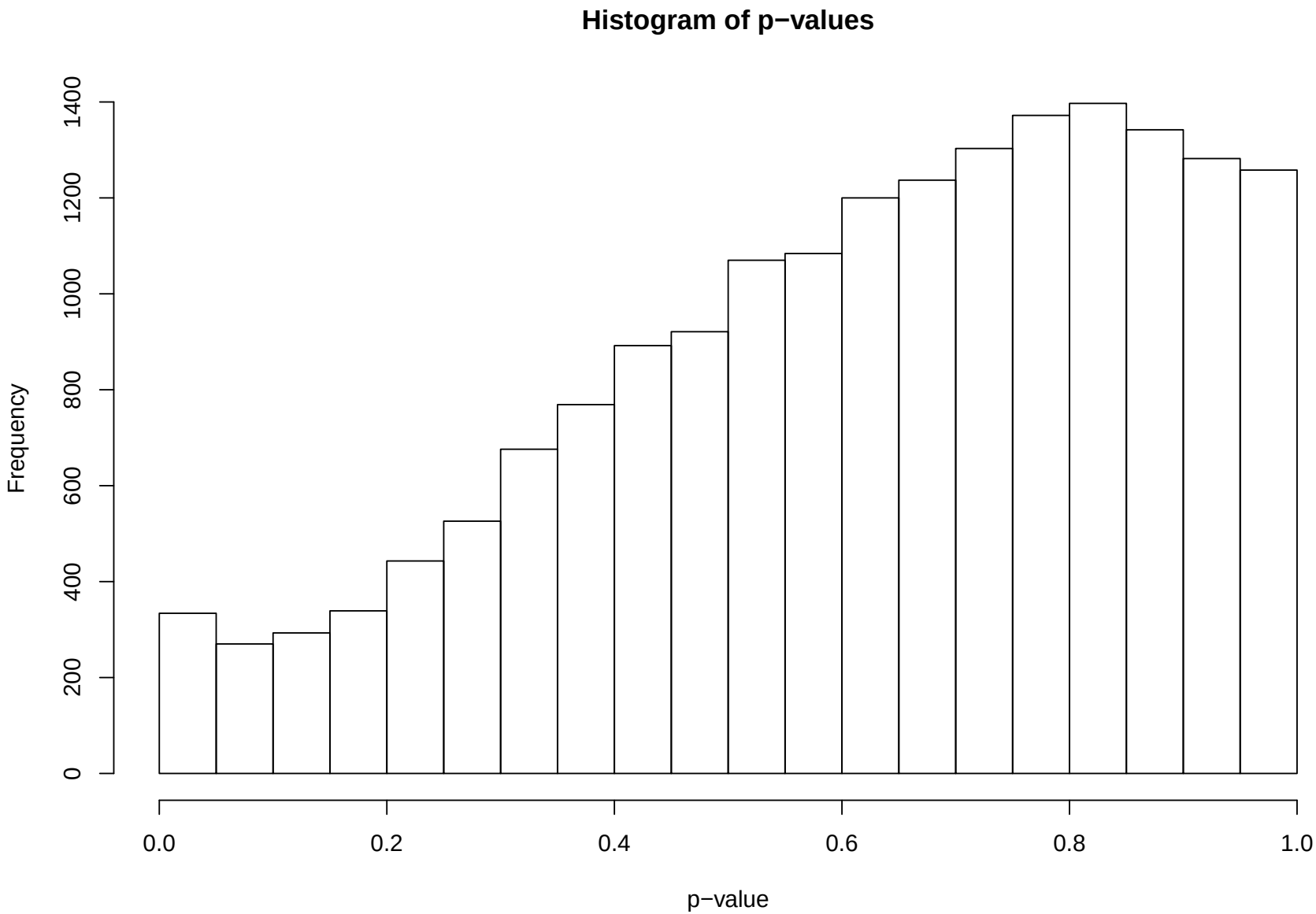
Accounting for unwanted variation



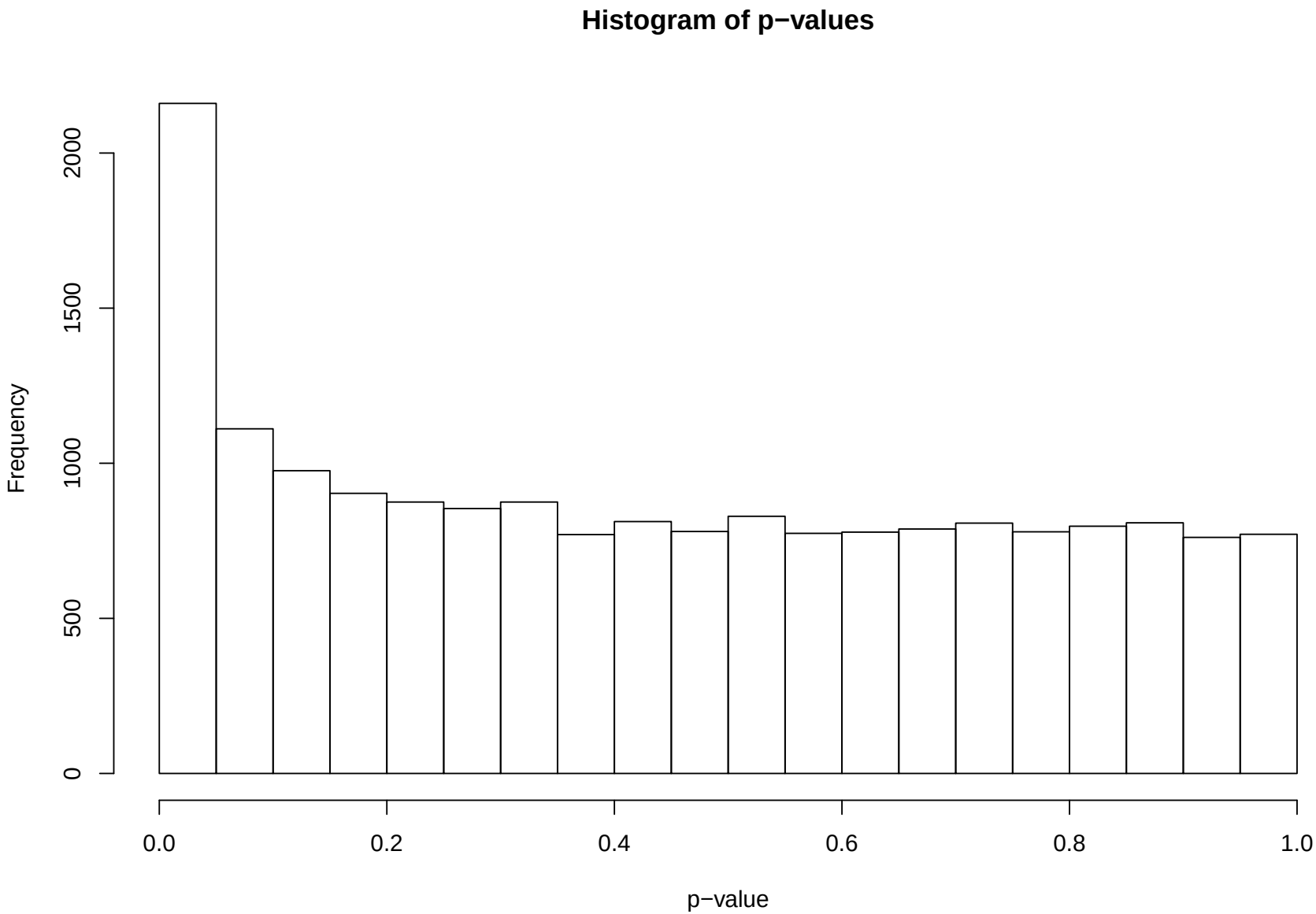
Accounting for unwanted variation



Without batch correction



With batch correction



Thank you for your attention!

Email: dar2062@med.cornell.edu

Slides and code: github.com/drisso/canazei