Differential Expressed Genes

Differential expression (DE) analysis refers to the identification of genes (or other types of genomic features, such as, transcripts or exons) that are expressed in significantly different quantities in distinct groups of samples, be it biological conditions (drug-treated vs. controls), diseased vs. healthy individuals, different tissues, different stages of development, or something else.

Although genes (if we focus on those for a while) are of course not expressed independent of each other, differential expression analysis is typically done on one gene at a time (although information is sometimes borrowed across genes, as we will see below) in a ***univariate* way.**

WHY?

the number of *examples* is much smaller than the number of *features*, which makes it harder to fit a statistical model that considers all genes as a whole.

Multivariate dimension reduction methods such as principal component analysis (PCA) can be used to construct **low-dimensional representations** of the expression profiles that retain some of the properties of the complete data set and are thus often useful for visualization

Differential Expressed Genes – Replicates

The purpose of **replication** is to be able to estimate the variability between and among groups, which is important for, for example, hypothesis testing. Technical replication is used to estimate the variability of the measurement technique, for example, RNA-seq. **Biological replication is used to find out the variability within a biological group**. Roughly speaking, a change observed in gene expression between two groups can only be called significant if the difference between the groups is large compared to the variability within the group, while taking the sample size into consideration.

How many replicates should you use? This depends on the specifics of the experiment. The biological homogeneity of the different samples, the purpose of the experiment and the desired level of statistical power, among other things, will affect the number of replicates needed.

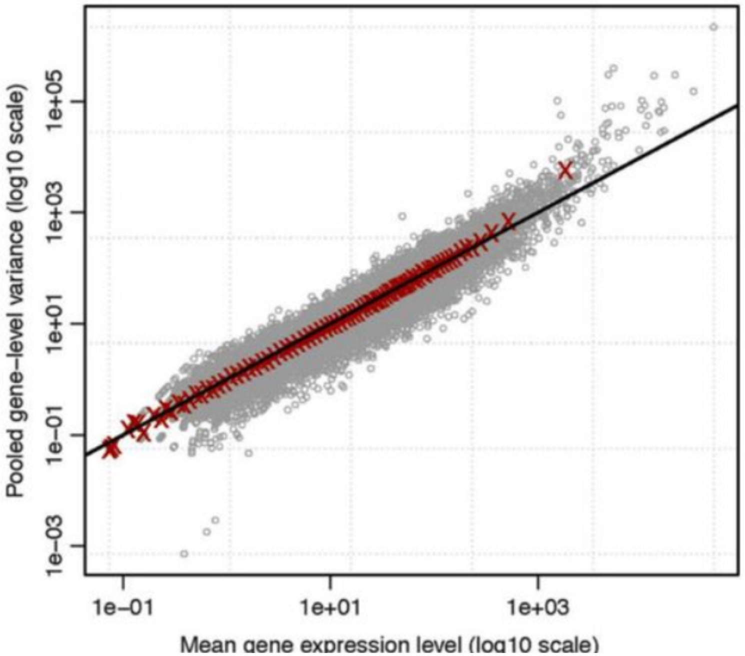
Many sequencing core facilities require or suggest using at least three or four replicates per group to be compared; two is almost always too few. With three, there is the risk that at least one sample will fail in library preparation or sequencing and you still end up with only two replicates in one of the groups.

Human blood and some tissue samples used for clinical case–control transcriptomics studies seem to exhibit considerable variation between individuals. Particularly for complex diseases, very large numbers of replicates (perhaps hundreds or thousands) may be needed to observe differential expression between cases and controls. For cell lines or samples from distinct tissues, only a few replicates may be needed.

Differential Expressed Genes – Statistical Distribution

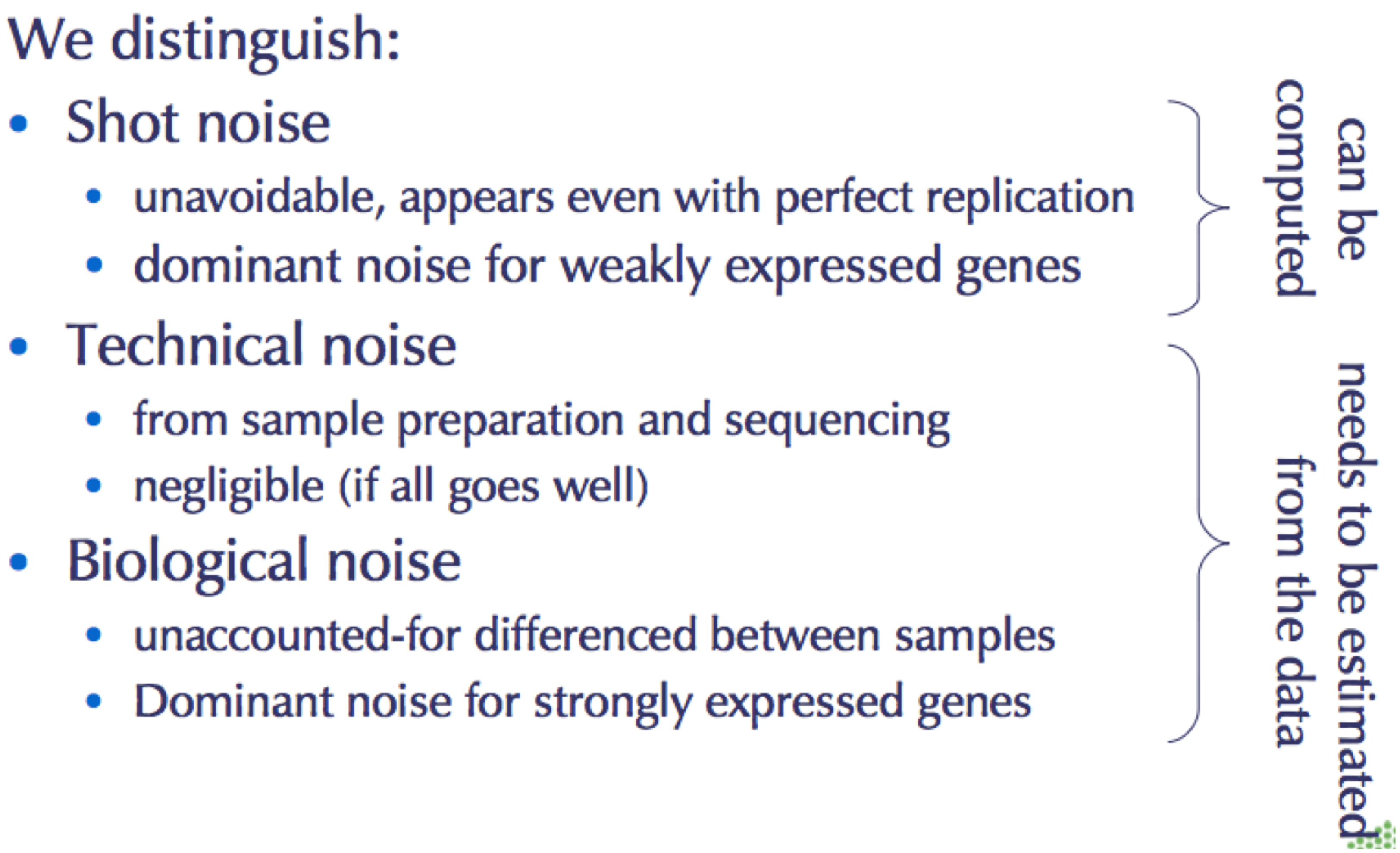
For RNA-seq experiments, where one might assume that sequences are sampled at random from the sequencing library, the raw read counts would be expected to be **Poisson-distributed**.

> You would expect to get slightly different counts even for the same library in an idealized scenario where it was sequenced twice under the same conditions. This inevitable noise which arises from the sampling process is called ***shot noise***, and often the variability between technical replicates in RNA-seq can be described quite well by this type of Poisson noise



Mean–variance plot for Marioni et al. dataset (Marioni et al. 2008). The variability in technically replicated RNA-seq data can be adequately captured using a Poisson model. The grey points in this plot shows the mean and pooled variance for each gene, scaled to account for differences in library size between samples. The black line displays the theoretical variance under the Poisson model where the variance is equal to the mean. The red crosses show binned variance, where genes are grouped by mean level.

Differential Expressed Genes – Noise



We distinguish:

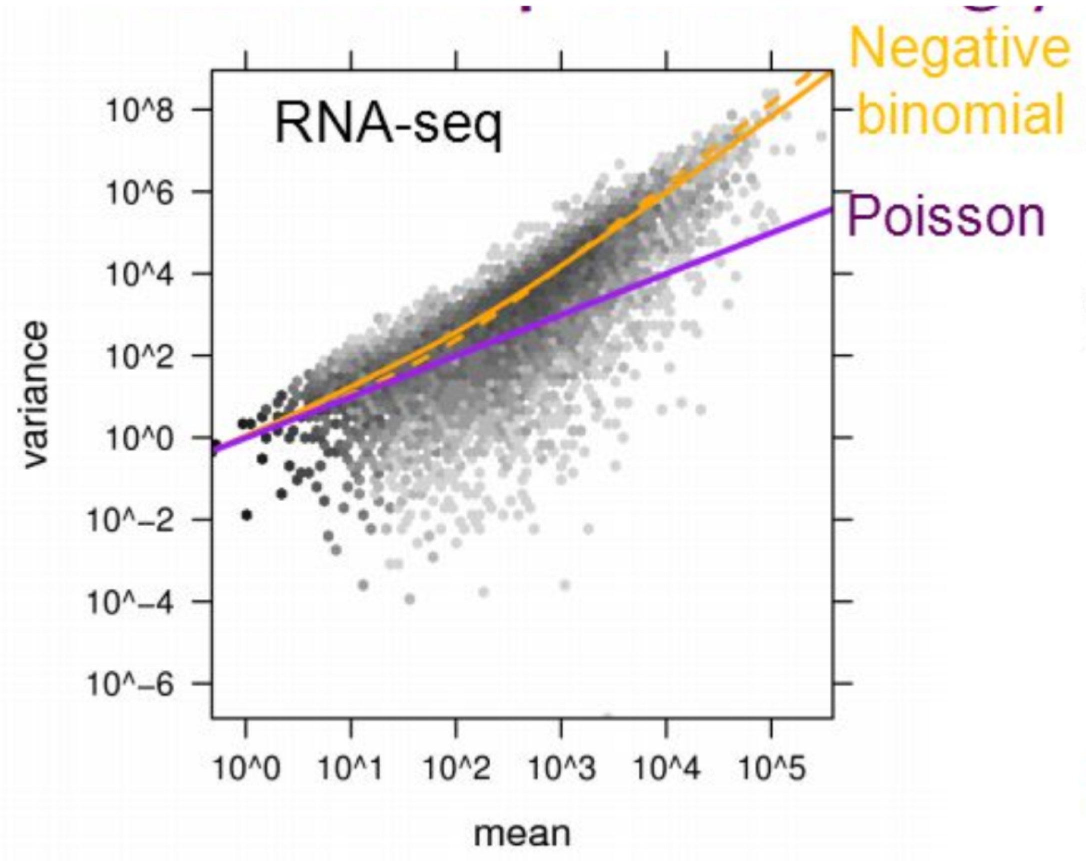
- Shot noise
  - unavoidable, appears even with perfect replication
  - dominant noise for weakly expressed genes
- Technical noise
  - from sample preparation and sequencing
  - negligible (if all goes well)
- Biological noise
  - unaccounted-for differenced between samples
  - Dominant noise for strongly expressed genes

can be computed

needs to be estimated from the data

Differential Expressed Genes – Statistical Distribution

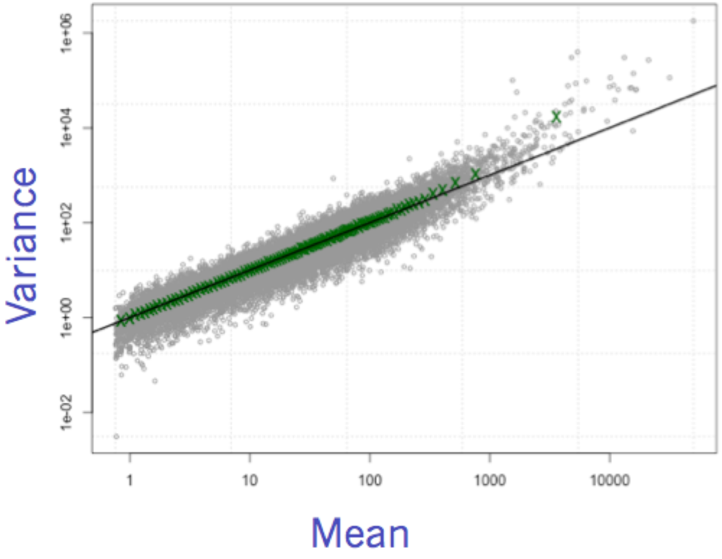
When samples are taken from biologically distinct sources, such as different individuals, the variability between them has often been modeled by a ***negative binomial* distribution** (sometimes called gamma-Poisson distribution). This distribution can be described as an *overdispersed* Poisson distribution



In RNAseq genes with high mean counts, because they are long or highly expressed, tend to show more variance between sample than genes with low mean counts. Thus this data fits a Negative Binomial Distribution.

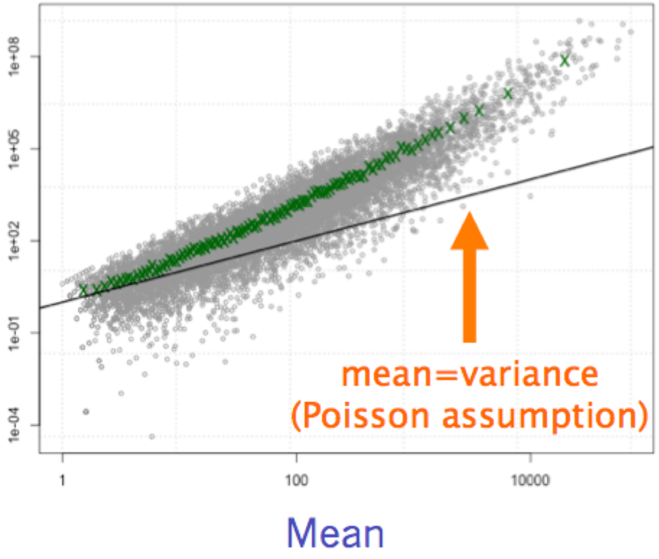
Differential Expressed Genes – Statistical Distribution

Technical replicates



data from Marioni et al. Gen Res 2008

Biological replicates



mean=variance
(Poisson assumption)

data from Parikh et al. *Genome Bio* 2010

Counts for the same gene from different **technical replicates** have variance equal to the mean (Poisson)

Counts for the same gene from different **biological replicates** have a variance exceeding the mean (overdispersion)

Differential Expressed Genes – Normalization, DESeq2

If sample A has been sampled deeper than sample B, we expect counts to be higher.

Naive approach: Divide by the total number of reads per sample

Problem: Genes that are strongly and **differentially expressed may distort the ratio of total reads**.

To compare more than two samples:

Form a “**virtual reference sample**” by taking, for each gene, the geometric mean of counts over all samples

Normalize each sample to this reference, to get one scaling factor (“size factor”) per sample.

Remember RPKM, FPKM and TPM? Those nice methods for adjusting for differences in overall read counts among libraries?

DESeq2 doesn't use those methods (neither does edgeR)…
Why not?

There are two main problems in library normalization, so let's talk about them.

Problem #1: adjusting for differences in library sizes

	Sample #1	Sample #2
Gene	635 reads	1,270 reads
A1BG	30	60
A1BG-AS1	24	48
A1CF	0	0
A2M	563	2126
A2M-AS1	5	10
A2ML1	13	26

The read counts for each gene in Sample #2 are twice the read counts in Sample #1.

This difference is not due to biology, but to sequencing depth.

RPKM, FPKM, TPM and CPM all deal with this.

However, there is another problem…

Problem #2: Adjusting for differences in library composition

RNA-seq (and other high-throughput sequencing) is often used to compare one tissue type to another. For example, liver vs. spleen.

It could be that there are a lot of liver specific genes transcribed in liver but not in the spleen.

You can also imagine seeing differences in library composition in the same tissue type if you knock out a transcription factor.

Problem #2: Adjusting for differences in library composition

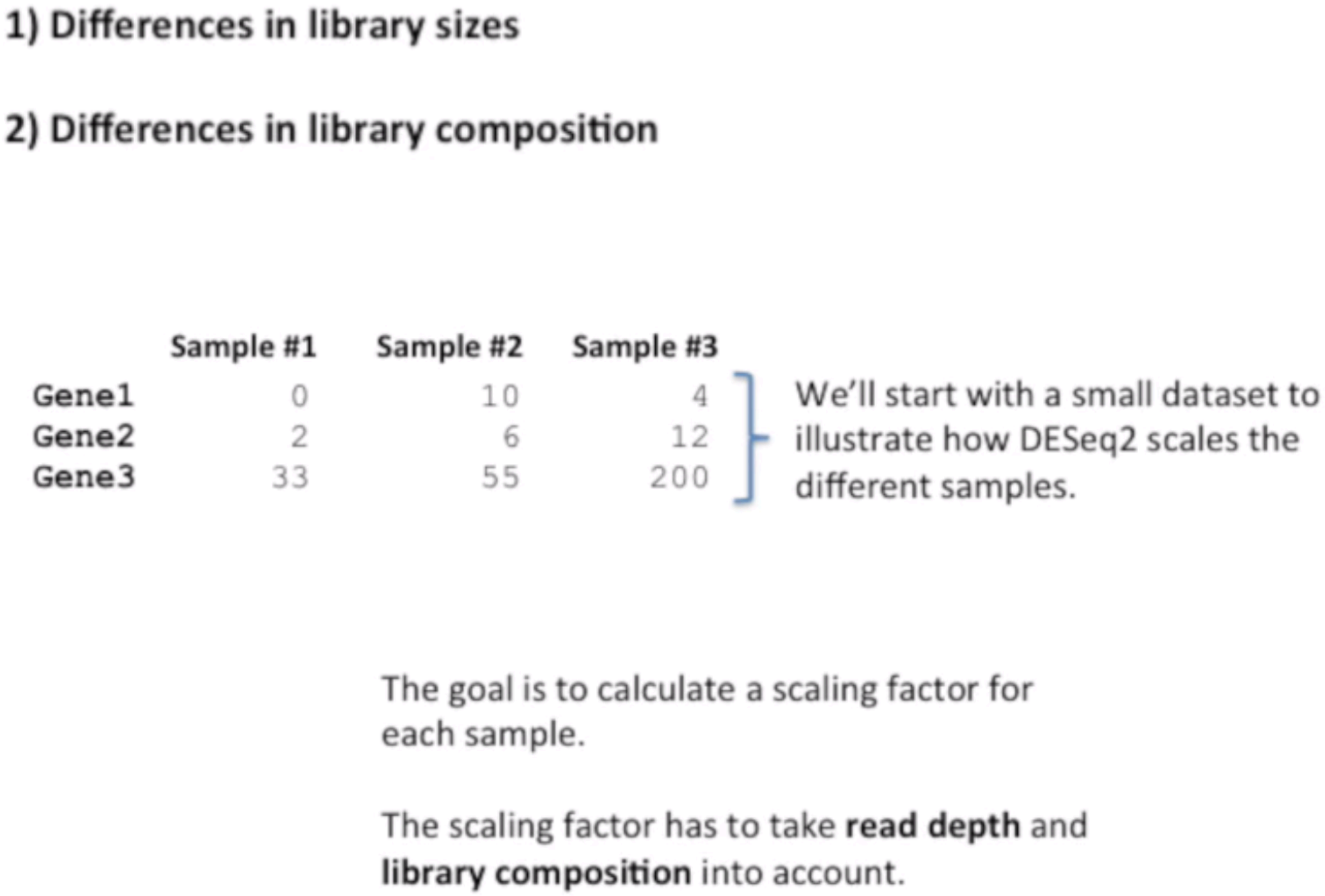
	Sample #1	Sample #2
Gene	635 reads	635 reads
A1BG	30	235
A1BG-AS1	24	188
A1CF	0	0
A2M	563	0
A2M-AS1	5	39
A2ML1	13	102

The read counts for everything but *A2M* are high in Sample #2

Assume that only Sample #1 transcribes *A2M*

This means that the 563 reads used up by *A2M* in Sample #1 will be distributed to other genes in Sample #2

DeSeq2 normalisation step want handle:



1) Differences in library sizes

2) Differences in library composition

	Sample #1	Sample #2	Sample #3
Gene1	0	10	4
Gene2	2	6	12
Gene3	33	55	200

We'll start with a small dataset to illustrate how DESeq2 scales the different samples.

The goal is to calculate a scaling factor for each sample.

The scaling factor has to take **read depth** and **library composition** into account.

	Sample #1	Sample #2	Sample #3
Gene1	0	10	4
Gene2	2	6	12
Gene3	33	55	200

$e^{2.3} = 10$

Step 1: Take the log of all the values

	log(Sample #1)	log(Sample #2)	log(Sample #3)
Gene1	-Inf	2.3	1.4
Gene2	0.7	1.8	2.5
Gene3	3.5	4.0	5.3

DESeq2 uses $\log_e$ ("log base *e*"), so these numbers are what we would need to raise *e* to in order to get the original value.

Notice that log(0) = -Infinity

This is just because R defines log(0) to be −Infinity.

	Sample #1	Sample #2	Sample #3
Gene1	0	10	4
Gene2	2	6	12
Gene3	33	55	200

One thing cool about the average of log values is that this average is not easily swayed by outliers.

To see this, let's calculate the average read count for **Gene3**

Step 1: Take the log of all the values

	log(Sample #1)	log(Sample #2)	log(Sample #3)
Gene1	-Inf	2.3	1.4
Gene2	0.7	1.8	2.5
Gene3	3.5	4.0	5.3

Step 2: Average Each Row

	Average of log values
Gene1	-Inf
Gene2	1.7
Gene3	4.3

	Sample #1	Sample #2	Sample #3
Gene1	0	10	4
Gene2	2	6	12
Gene3	33	55	200

Avg(**Gene3**) = 96

Now convert the average log value for **Gene3** into a normal number.

Step 2: Average Each Row

	Average of log values
Gene1	-Inf
Gene2	1.7
Gene3	4.3

Remember that logs are exponents, and in this case they are exponents of *e*, so we have to raise *e* by 4.3 to get a "normal number".

The average calculated with the logs is smaller, and thus, not swayed as much by the outlier.

$e^{4.3} = 73.7$

NOTE: Averages calculated with logs are called "Geometric Averages"

Step 2: Average Each Row

	Average of log values
Gene1	-Inf
Gene2	1.7
Gene3	4.3

Step 3: Filter Out Genes with Infinity

	Average of log values
Gene2	1.7
Gene3	4.3

In general, this step filters out genes with zero read counts in one or more samples.

If you are comparing liver and spleen, this will remove all of the genes only transcribed in liver (or spleen).

In theory, this helps focus the scaling factors on the house keeping genes – genes transcribed at similar levels regardless of tissue type.

Step 4: Subtract the average log value from the log(counts)

	log(Sample #1)	log(Sample #2)	log(Sample #3)
Gene2	0.7	1.8	2.5
Gene3	3.5	4.0	5.3

	Average of log values
Gene2	1.7
Gene3	4.3

	Sample #1	Sample #2	Sample #3
Gene2	-1.0	0.1	0.5
Gene3	-0.8	-0.3	1.3

So we're really checking out the ratio of the reads in each sample to the average across all samples.

Remember:

$$\log(\text{reads for gene X}) - \log(\text{average for gene X}) = \log\left(\frac{\text{reads for gene X}}{\text{average for gene X}}\right)$$

Step 5: Calculate the median of the ratios for each sample

$$\log\left(\frac{\text{reads for gene X}}{\text{average for gene X}}\right)$$

	Sample #1	Sample #2	Sample #3
Gene2	-1.0	0.1	0.5
Gene3	-0.8	-0.3	1.3
median =	-0.9	-0.1	0.9

NOTE: Using the median is another way to avoid extreme genes from swaying the value too much in one direction.

Genes with huge differences in expression have no more influence on the median than genes with minor differences.

Since genes with huge differences will most likely be rare, the effect is to give more influence to moderate differences and "house-keeping" genes.

Step 6: Covert the medians to "normal numbers" to get the final scaling factors for each sample

Scaling factor for Sample #1: $e^{-0.9} = 0.4$ Sample #2: $e^{-0.3} = 0.7$ Sample #3: $e^{0.9} = 2.5$

Step 7: Divide the original read counts by the scaling factors

Original read counts

	Sample #1	Sample #2	Sample #3
Gene1	0	10	4
Gene2	2	6	12
Gene3	33	55	200

Scaled read counts

	Sample #1	Sample #2	Sample #3
Gene1	0	14	2
Gene2	5	9	5
Gene3	83	79	80

Scaling factor for Sample #1: $e^{-0.9} = 0.4$ Sample #2: $e^{-0.3} = 0.7$ Sample #3: $e^{0.9} = 2.5$

Summary of DESeq2's Library Size Scaling Factor

Logs eliminate all genes that are only transcribed in one sample type (liver vs spleen).
They also help smooth over outlier read counts (via the Geometric Mean).

The **median** further downplays genes that soak up a lot of the reads, putting more emphasis on moderately expressed genes.

Differential Expressed Genes – Generalized linear models

Two sample groups, treatment and control.

Assumption:

Count value for a gene in sample j is generated by Negative Binomial distribution with mean μj and dispersion α.

Null hypothesis:

All samples have the same μj .

Alternative hypothesis:

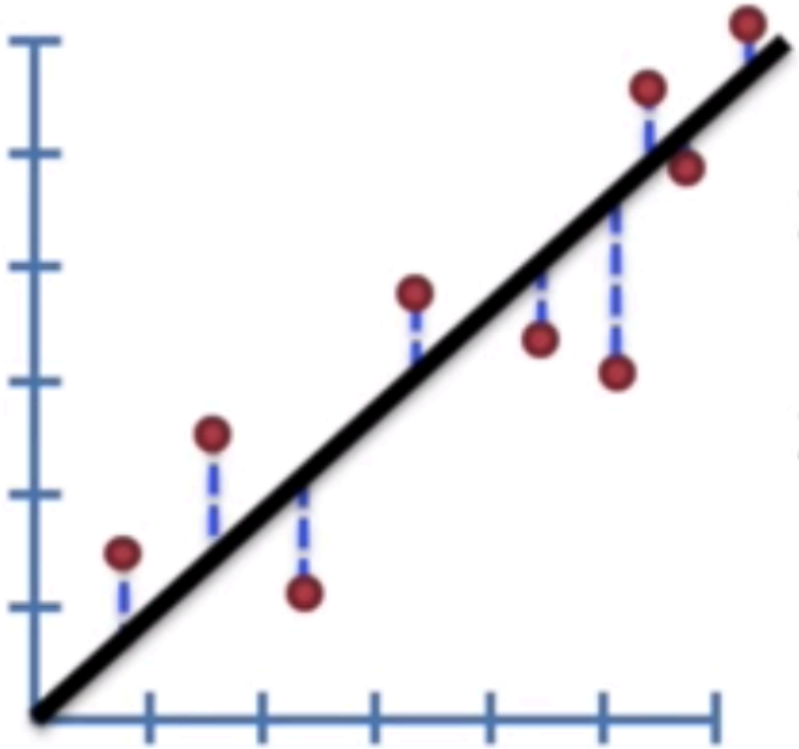
Mean is the same only within groups:

$$\log \mu j = \beta C + xj\ \beta T$$

where $xj = 0$ if j is control sample

$xj = 1$ if j is treatment sample

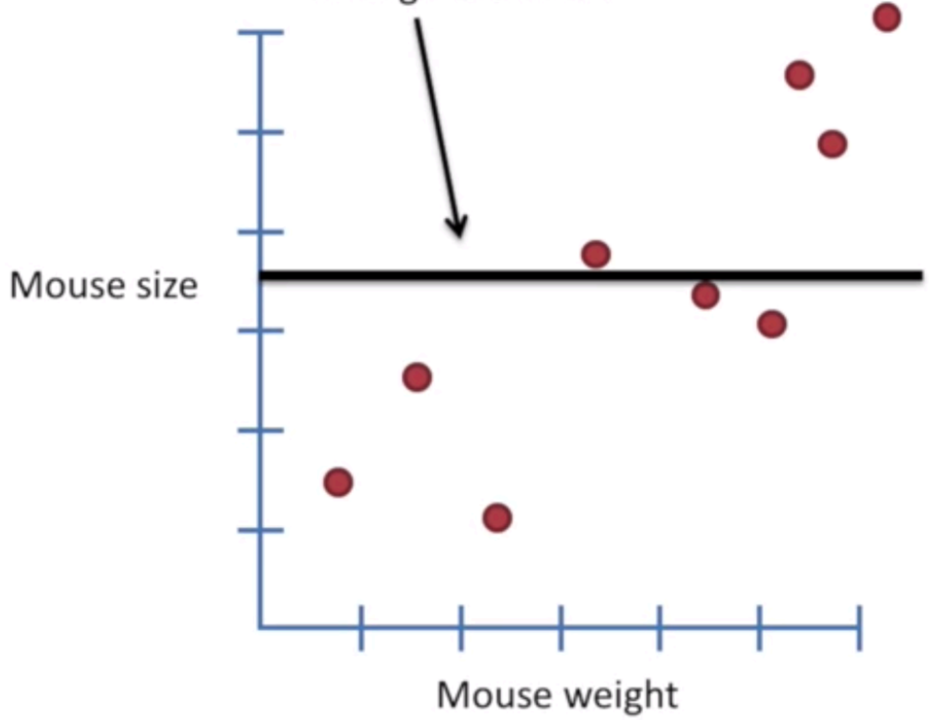
1) Use least-squares to fit a a line to the data.

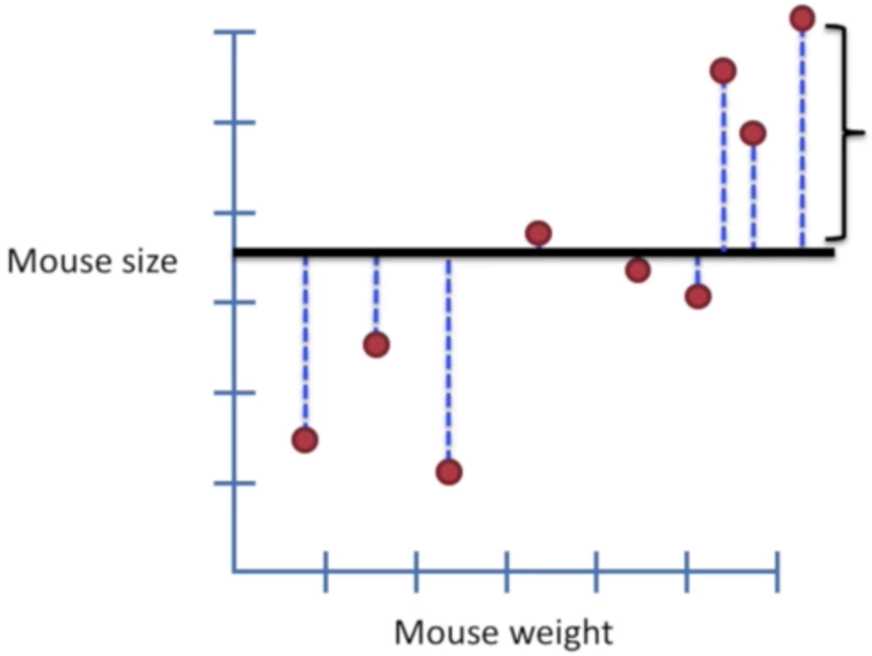


2) Calculate R^2

3) Calculate a *p*-value for R^2

First, draw a line through the data…

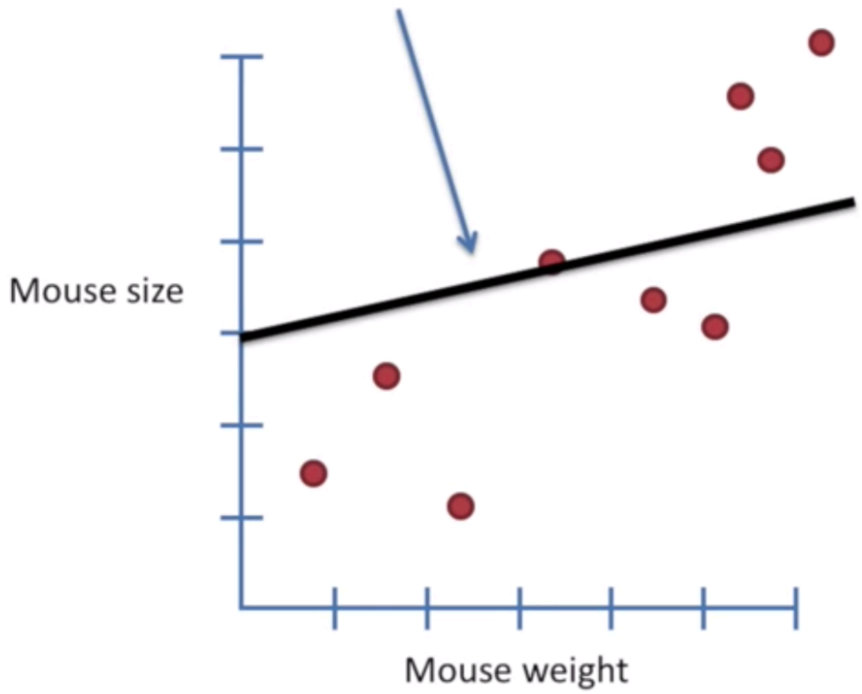


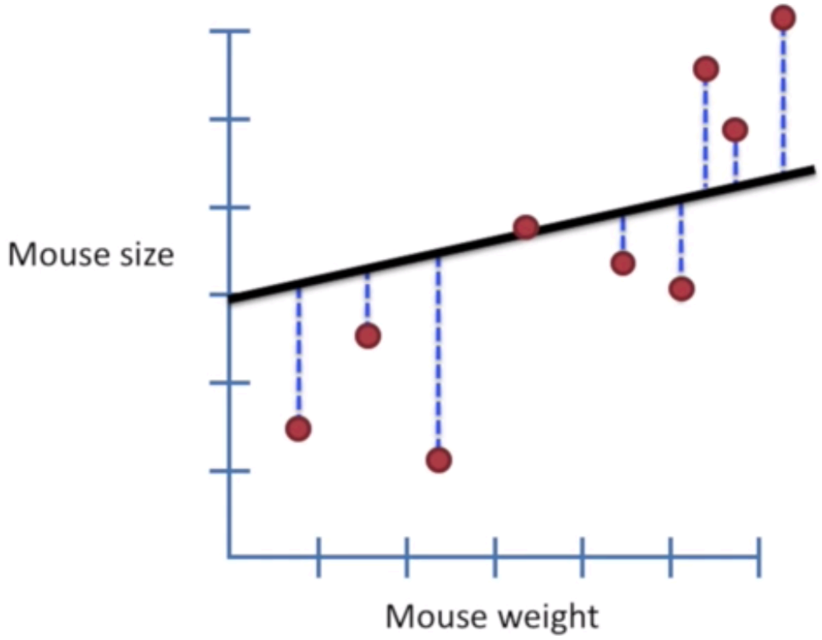


Second, measure the distance from the line to the data, square each distance, and then add them up.

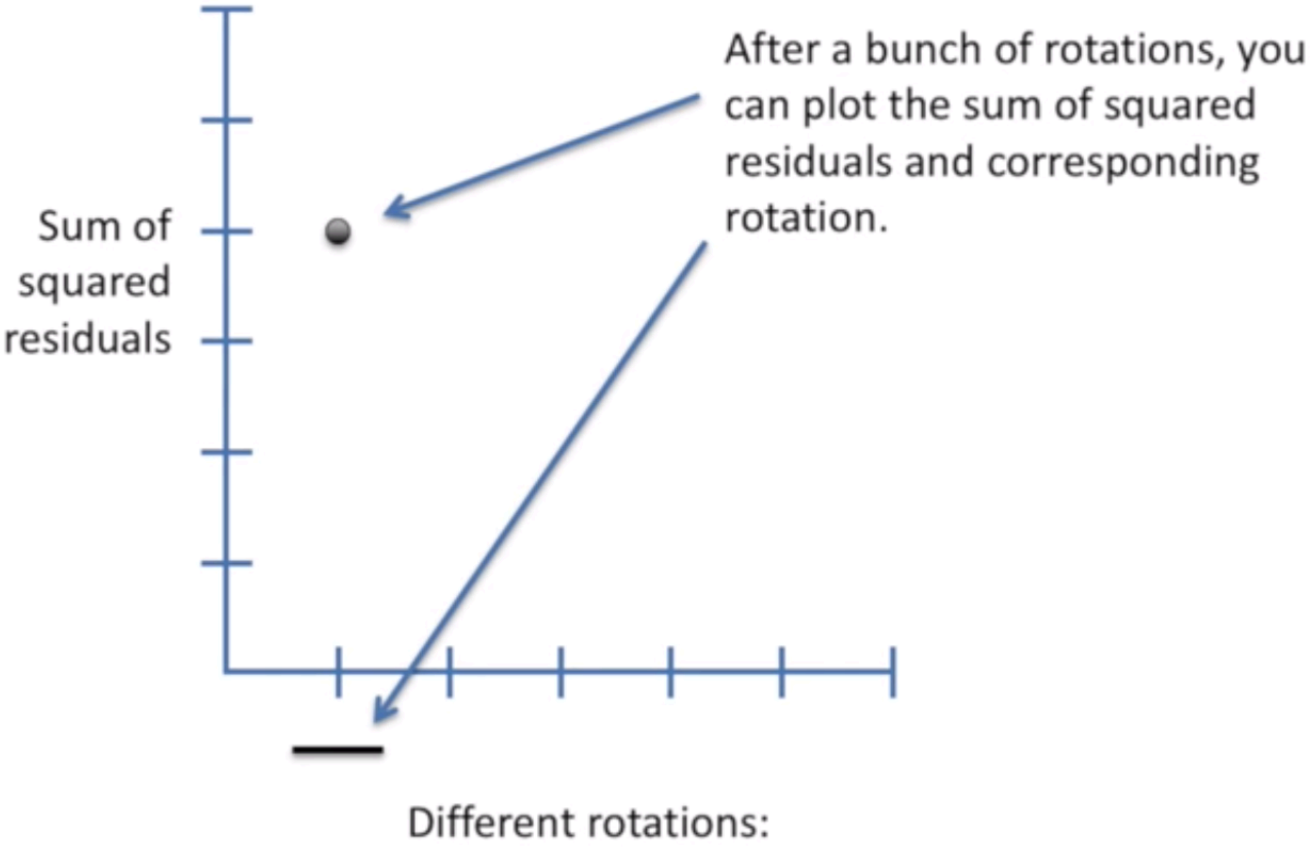
The distance from a line to a data point is called a "**residual**".

Third, rotate the line a little bit…

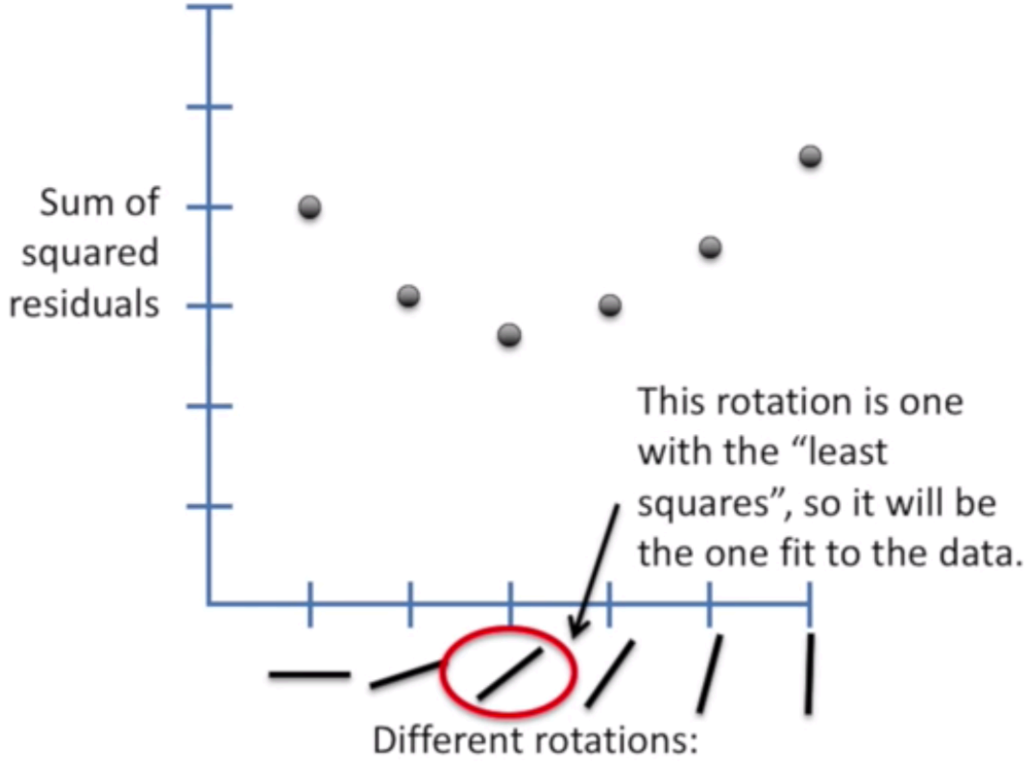




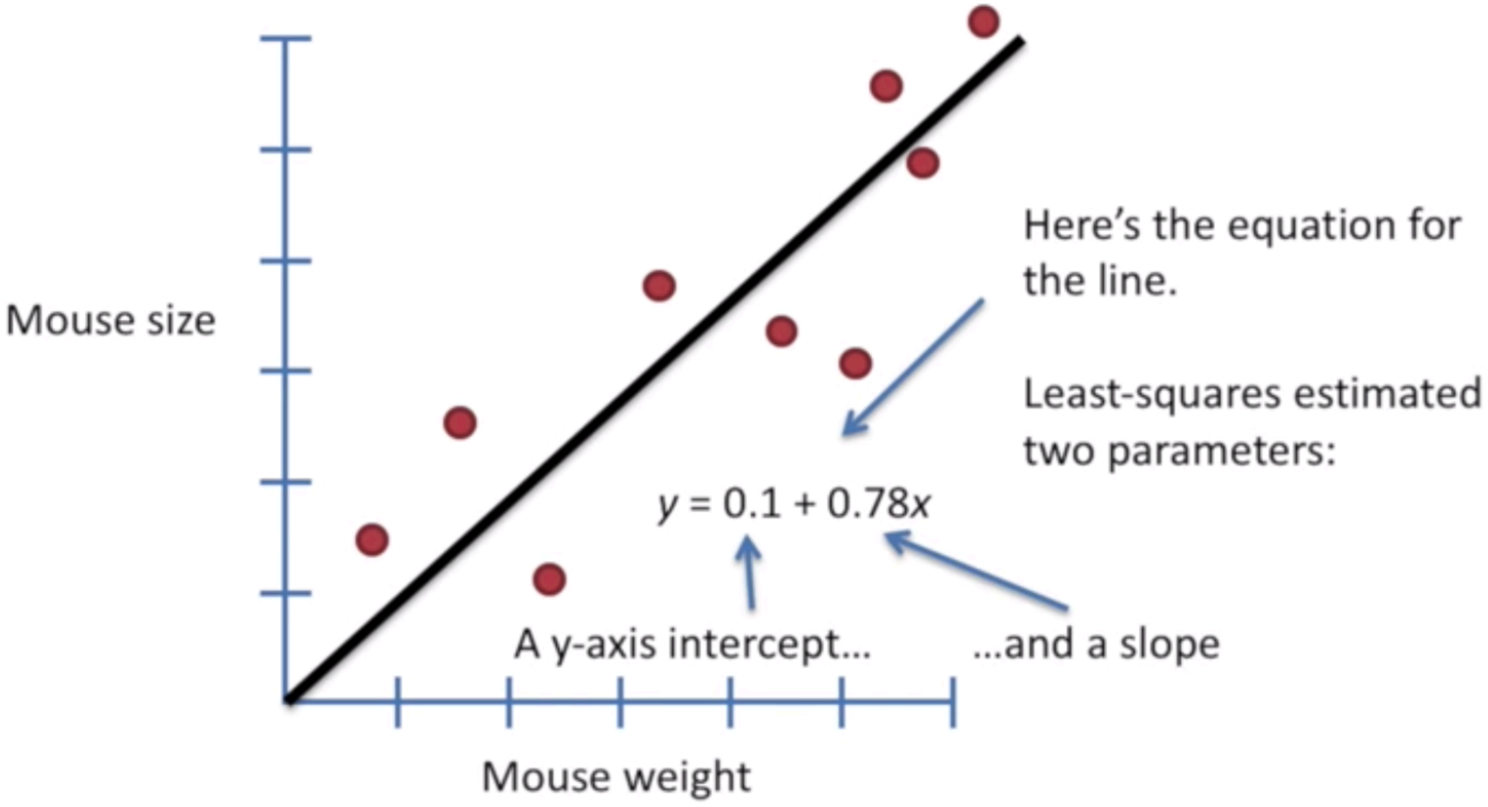
With the new line, measure the residuals, square them, and then sum up the squares.



After a bunch of rotations, you can plot the sum of squared residuals and corresponding rotation.



This rotation is one with the "least squares", so it will be the one fit to the data.



Here's the equation for the line.

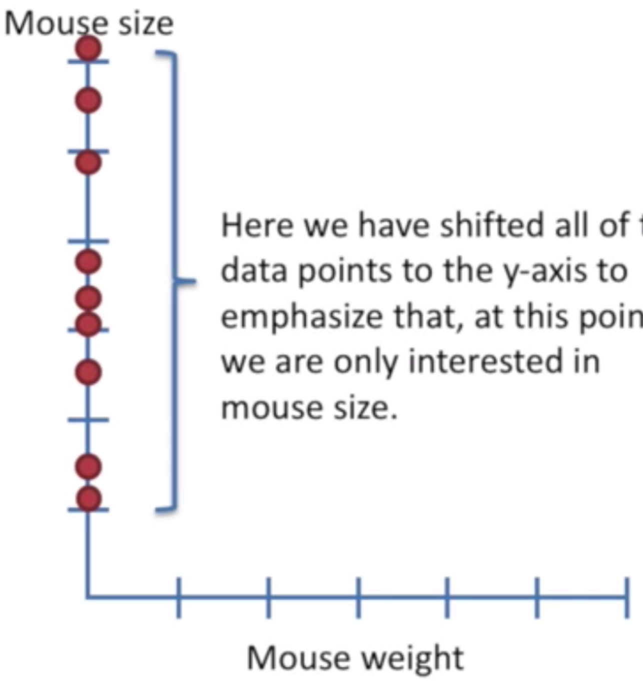
Least-squares estimated two parameters:

$y = 0.1 + 0.78x$

A y-axis intercept...

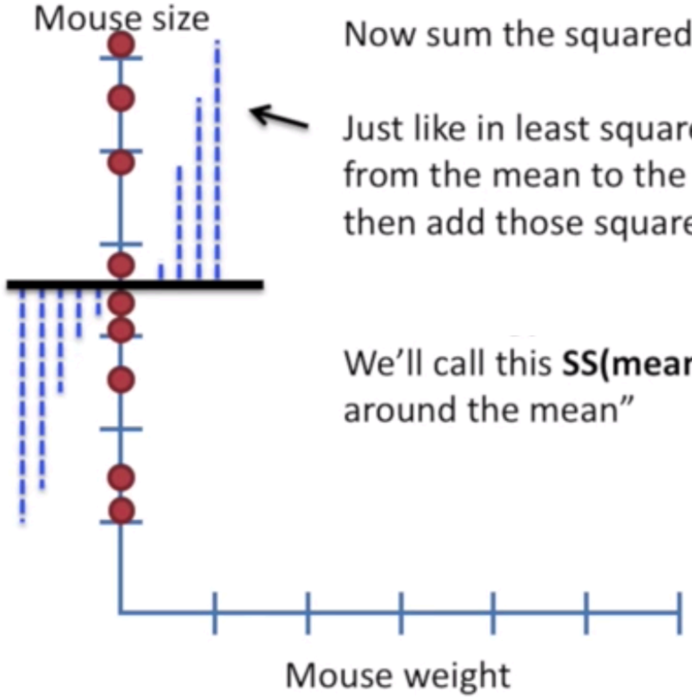
...and a slope

Calculating R^2 is the first step in determining how good that guess will be.



First, calculate the average mouse size.

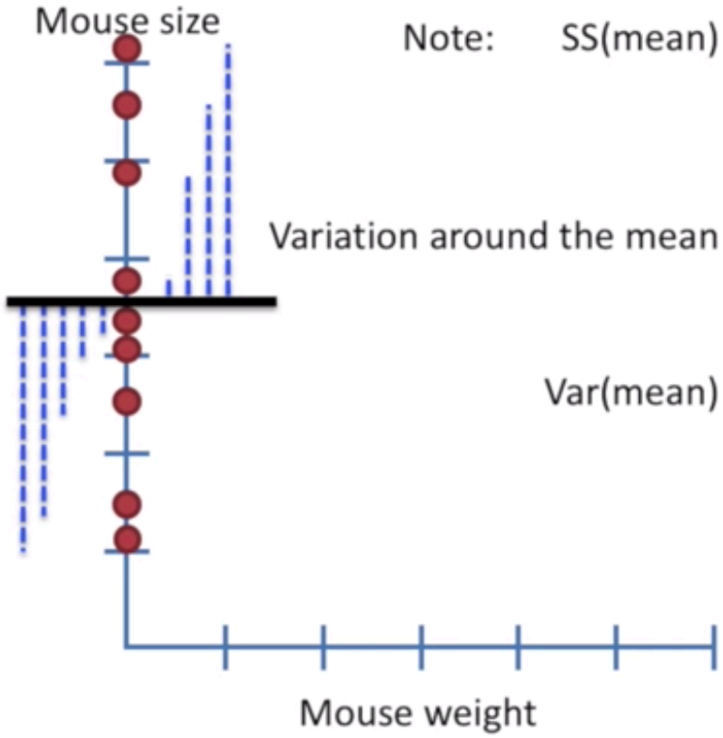
Here we have shifted all of the data points to the y-axis to emphasize that, at this point, we are only interested in mouse size.



Now sum the squared residuals…

Just like in least squares, we measure the distance from the mean to the data point and square it, then add those squares together.

We'll call this **SS(mean)**, for "sum of squares around the mean"



Note: $\text{SS(mean)} = (\text{data - mean})^2$

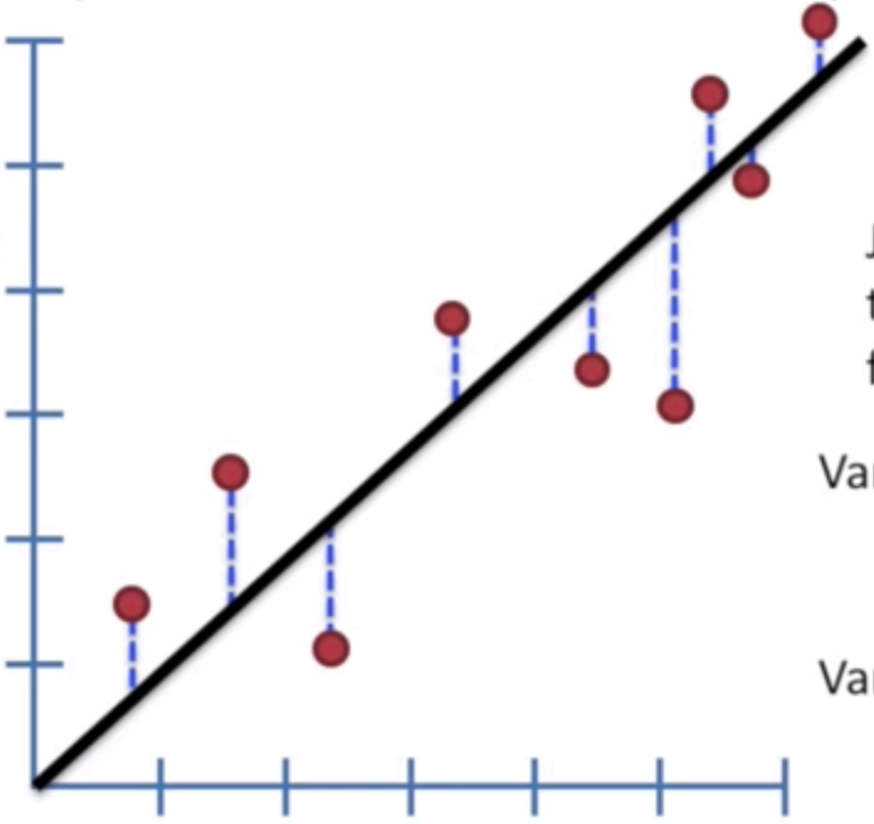
$$\text{Variation around the mean} = \frac{(\text{data - mean})^2}{n}$$

$$\text{Var(mean)} = \frac{\text{SS(mean)}}{n}$$

Now go back to the original plot.
Sum up the squared residuals around our least-squares fit.

We'll call this **SS(fit)**, for the sum of squares around the least-squares fit.

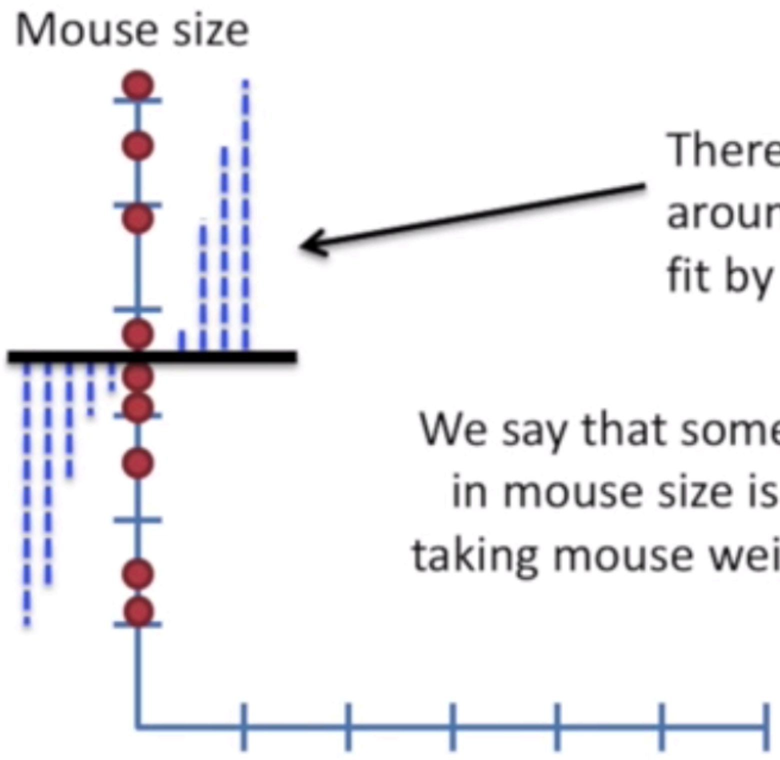
SS(fit) = (data - line)^2



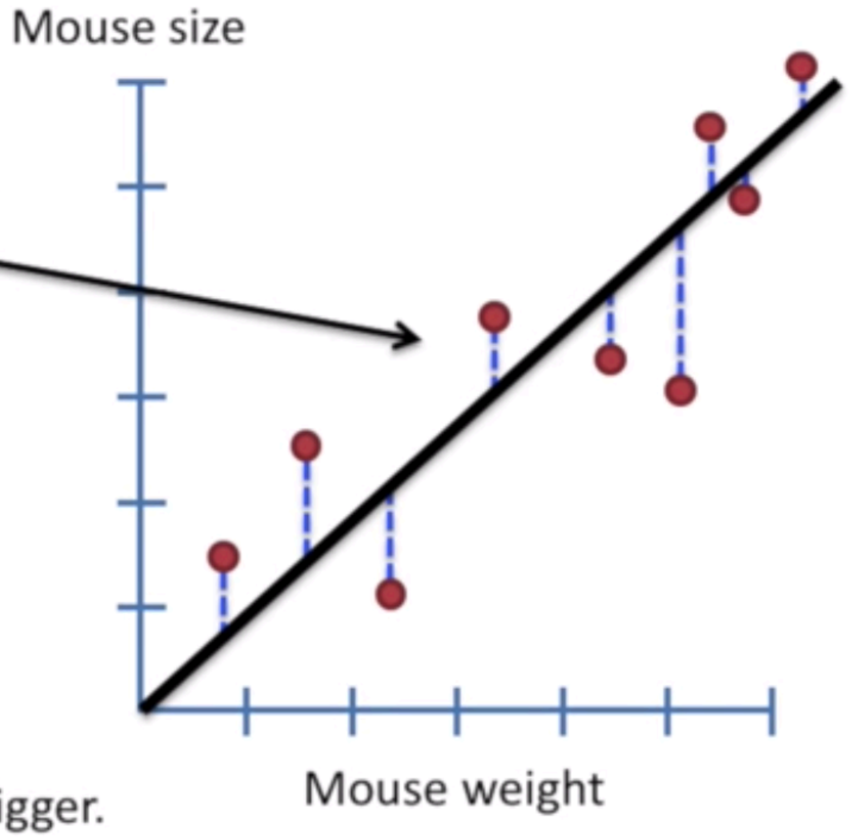
Just like with the mean, the variance around the fit...

$$\text{Var(fit)} = \frac{(\text{data - line})^2}{n}$$

$$\text{Var(fit)} = \frac{\text{SS(fit)}}{n}$$



There is less variation around the line that we fit by least-squares.

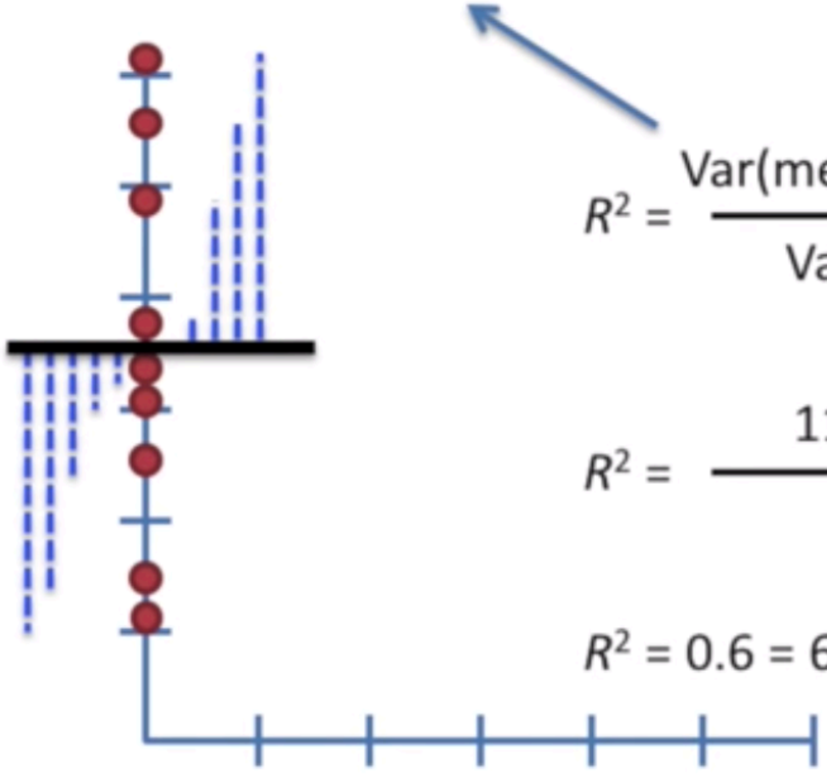


We say that some of the variation in mouse size is "explained" by taking mouse weight into account.

Heavier mice are bigger.
Lighter mice are smaller.

Var(mean) = 11.1

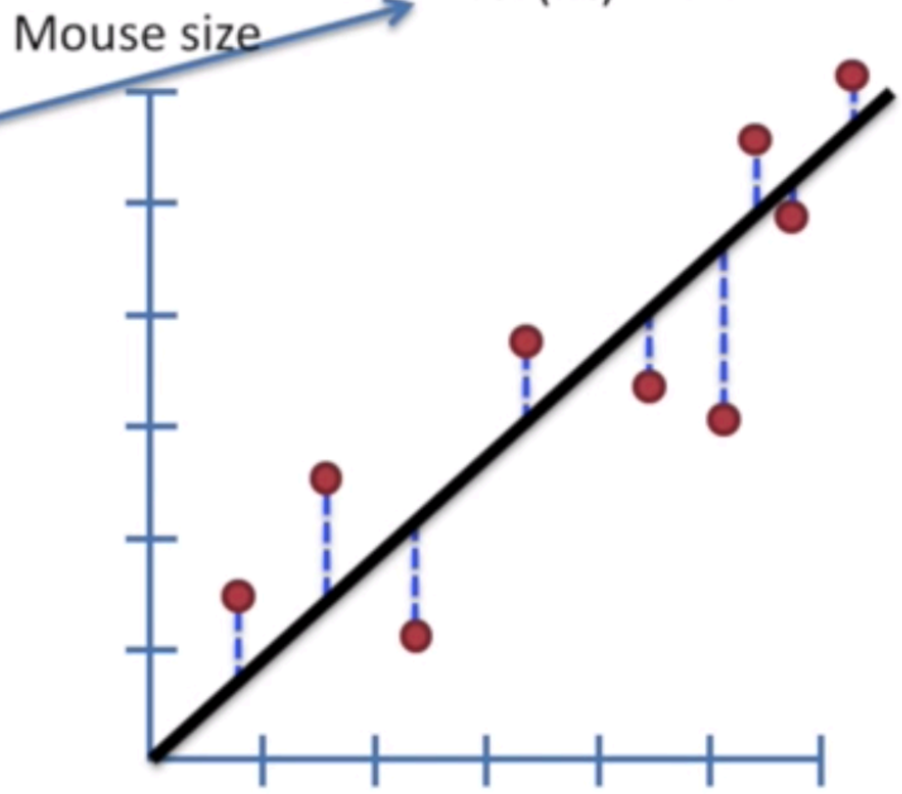
Var(fit) = 4.4



$$R^2 = \frac{\text{Var(mean)} - \text{Var(fit)}}{\text{Var(mean)}}$$

$$R^2 = \frac{11.1 - 4.4}{11.1}$$

$R^2 = 0.6 = 60\%$

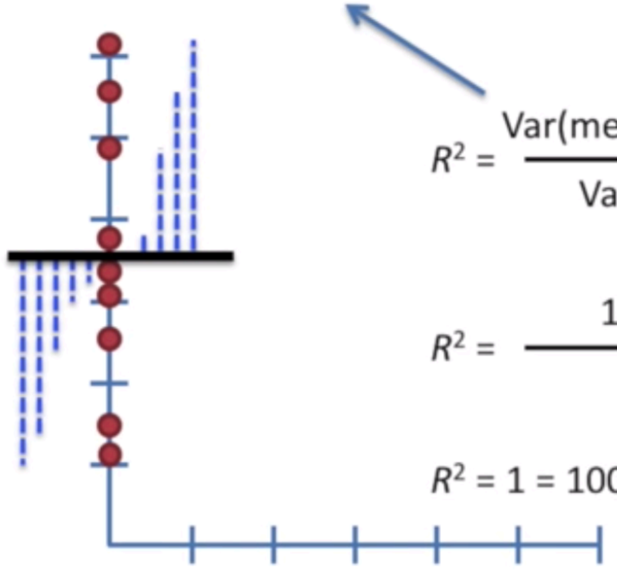


There is a 60% reduction in variance when we take the mouse weight into account.

Alternatively, we can say that mouse weight "explains" 60% of the variation in mouse size.

Var(mean) = 11.1

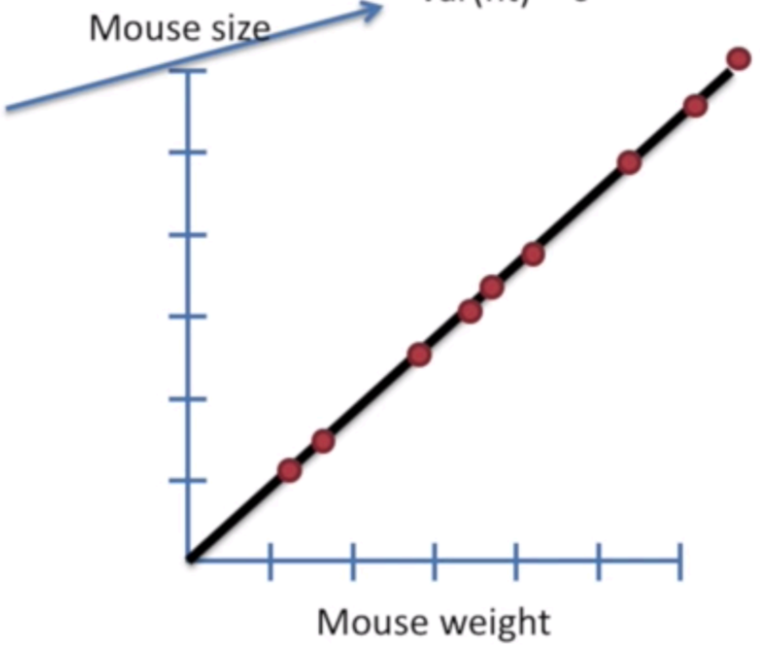
Var(fit) = 0



$$R^2 = \frac{\text{Var(mean)} - \text{Var(fit)}}{\text{Var(mean)}}$$

$$R^2 = \frac{11.1 - 0}{11.1}$$

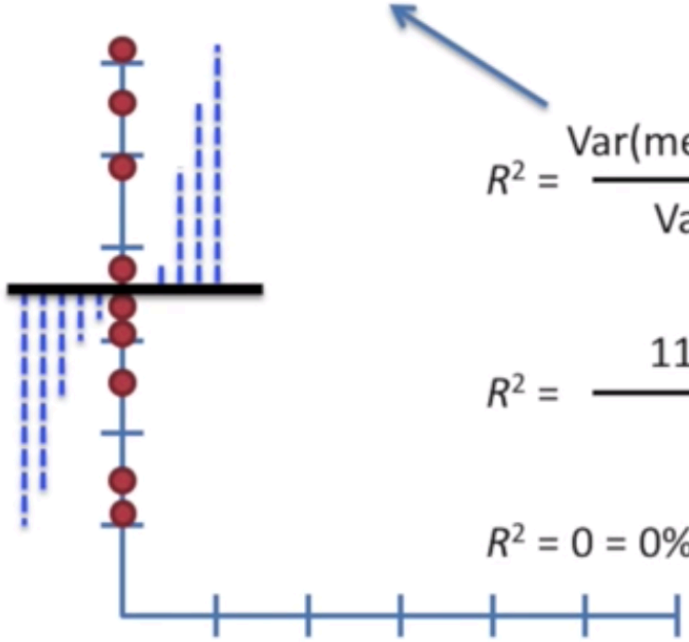
$R^2 = 1 = 100\%$



In this case, mouse weight "explains" 100% of the variation in mouse size.

Var(mean) = 11.1

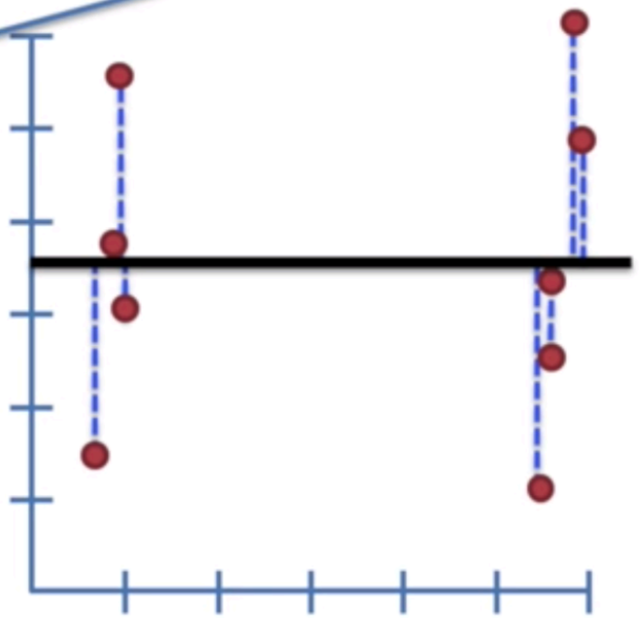
Var(fit) = 11.1



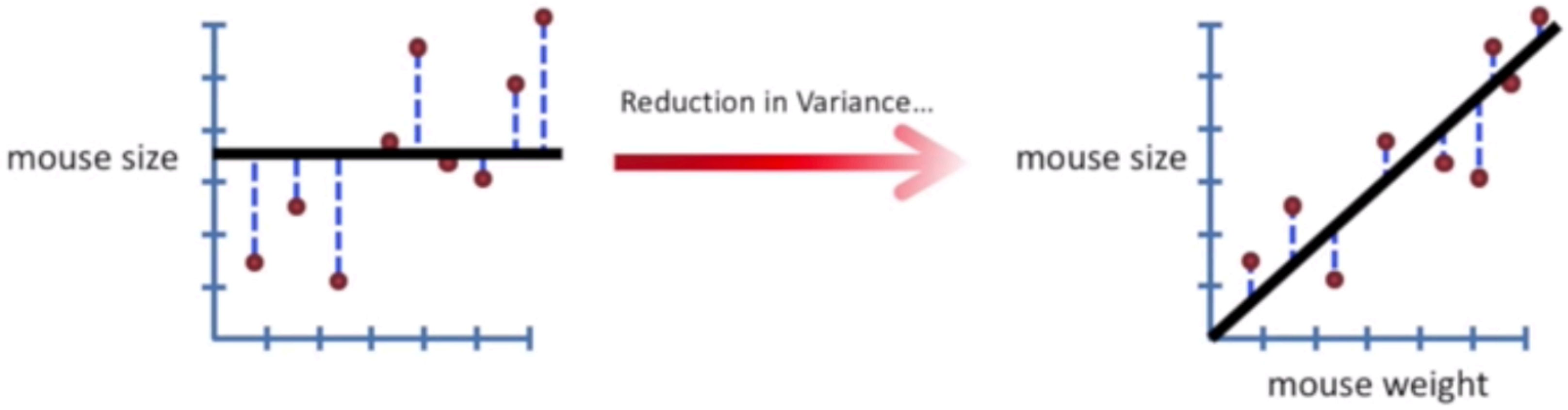
$$R^2 = \frac{\text{Var(mean)} - \text{Var(fit)}}{\text{Var(mean)}}$$

$$R^2 = \frac{11.1 - 11.1}{11.1}$$

$R^2 = 0 = 0\%$



In this case, mouse weight doesn't "explain" any of the variation around the mean.



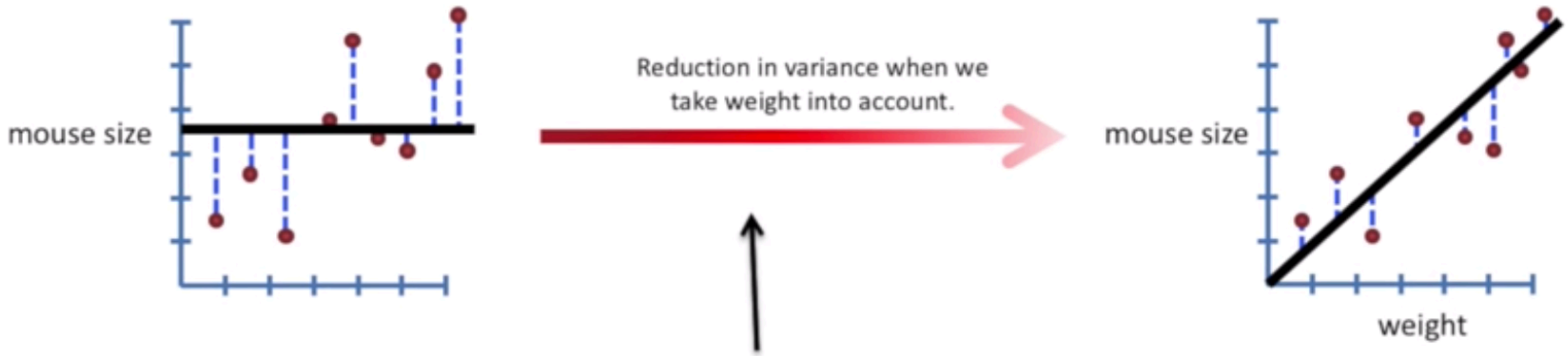
In this particular example, $R^2 = 0.6$, meaning we saw a 60% reduction in variation once we took mouse weight into account.

$$R^2 = \frac{\text{The variation in mouse size explained by weight}}{\text{The variation in mouse size without taking weight into account}}$$

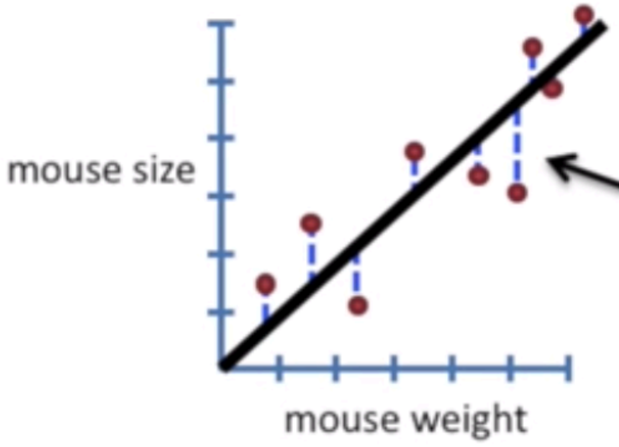
calculating a *p*-value

$$F = \frac{\text{The variation in mouse size explained by weight}}{\text{The variation in mouse size not explained by weight}}$$

The *p*-value for R^2 comes from something called "*F*"



$$F = \frac{\text{The variation in mouse size explained by weight}}{\text{The variation in mouse size not explained by weight}}$$



These dotted lines (residuals) represent the variation that remains after fitting the line. **This is the variation that is not explained by weight.**

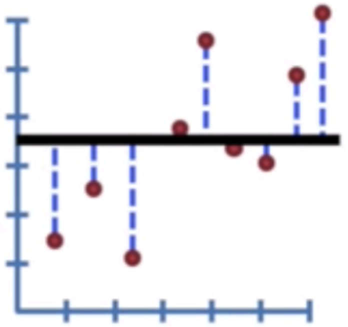
$$F = \frac{SS(\text{mean}) - SS(\text{fit}) / (p_{\text{fit}} - p_{\text{mean}})}{SS(\text{fit}) / (n - p_{\text{fit}})}$$

This equation will tell us if R^2 is significant.

$$F = \frac{SS(\text{mean}) - SS(\text{fit}) / (p_{\text{fit}} - p_{\text{mean}})}{SS(\text{fit}) / (n - p_{\text{fit}})}$$

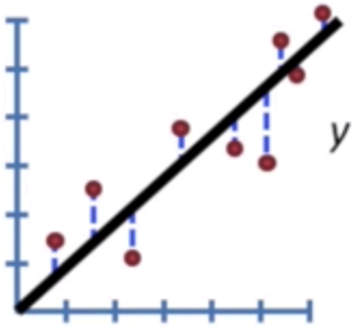
These numbers over here are the "degrees of freedom".

They turn the sums of squares into variances.



y = y-intercept

1 parameter



y = y-intercept + slope x

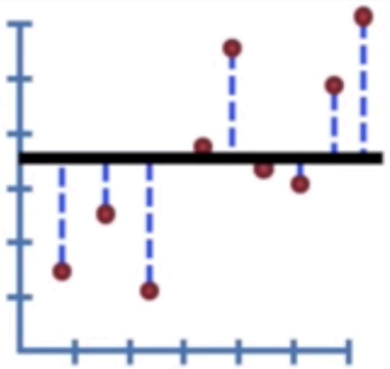
2 parameters

$p_{\text{fit}} = 2$

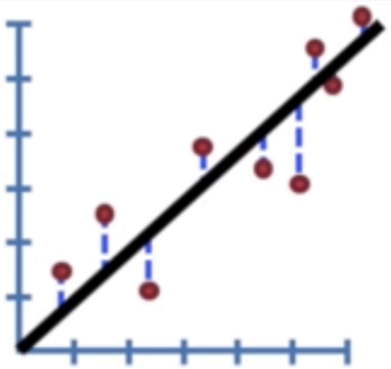
$$F = \frac{SS(\text{mean}) - SS(\text{fit}) / (p_{\text{fit}} - p_{\text{mean}})}{SS(\text{fit}) / (n - p_{\text{fit}})}$$

p_{fit} is the number of parameters in the fit line...

p_{mean} is the number of parameters in the mean line.



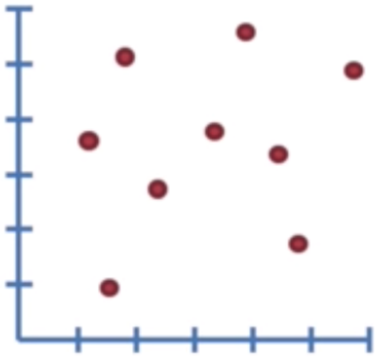
If the “fit” is good, then…



$$F = \frac{\text{The variation explained by the extra parameters in the “fit”}}{\text{The variation not explained by the extra parameters in the “fit”.}} \rightarrow \frac{\text{large number}}{\text{small number}}$$

F = really large number

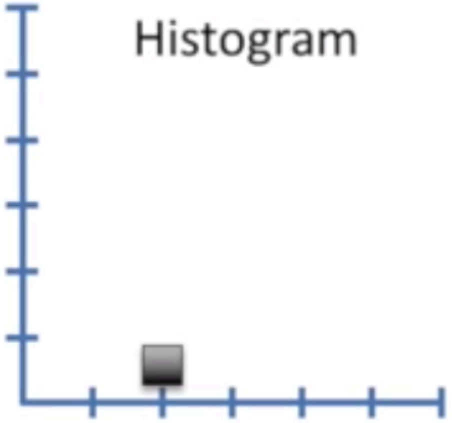
How do we turn this number in to a *p*-value?



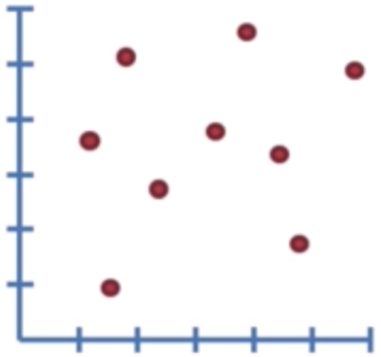
Generate a set of random data…

…calculate the mean and SS(mean)…

…calculate the “fit” and SS(fit)…

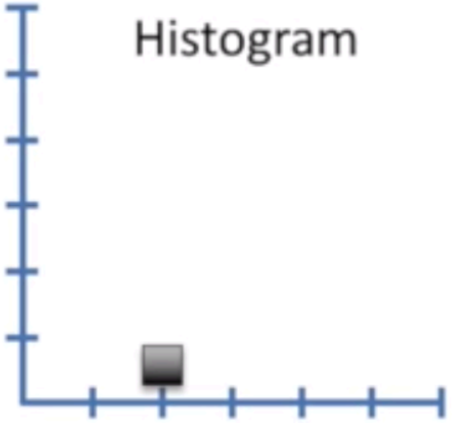


$$F = \frac{SS(\text{mean}) - SS(\text{fit}) / p_{\text{extra}}}{SS(\text{fit}) / (n - p_{\text{fit}})} \longrightarrow F = 2$$

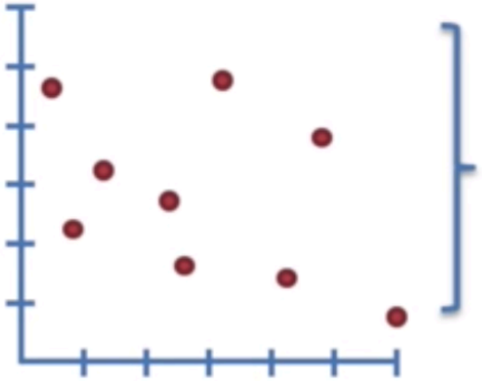


...calculate the mean and SS(mean)...

...calculate the “fit” and SS(fit)...



$$F = \frac{\mathrm{SS(mean)} - \mathrm{SS(fit)} / p_{\mathrm{extra}}}{\mathrm{SS(fit)} / (n - p_{\mathrm{fit}})} \longrightarrow F = 2$$

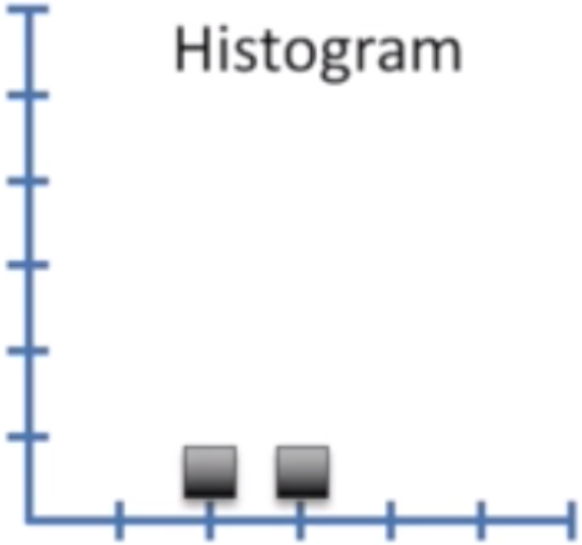


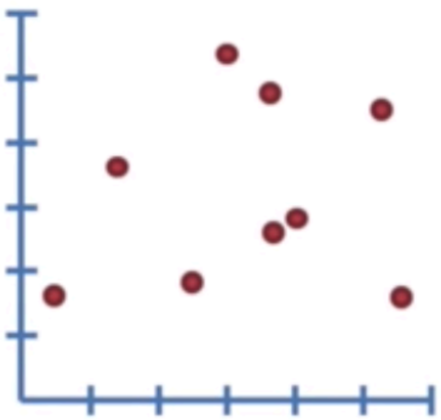
Generate another set of random data.

...calculate the mean and SS(mean)...

...calculate the “fit” and SS(fit)...

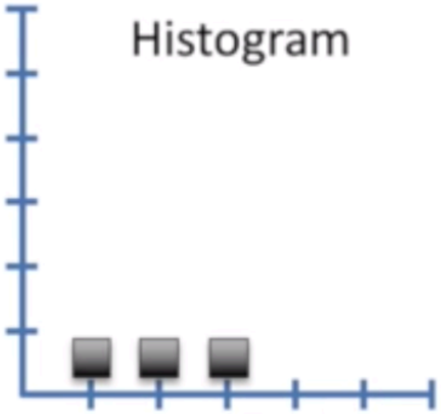
$$F = \frac{\mathrm{SS(mean)} - \mathrm{SS(fit)} / p_{\mathrm{extra}}}{\mathrm{SS(fit)} / (n - p_{\mathrm{fit}})} \longrightarrow F = 3$$

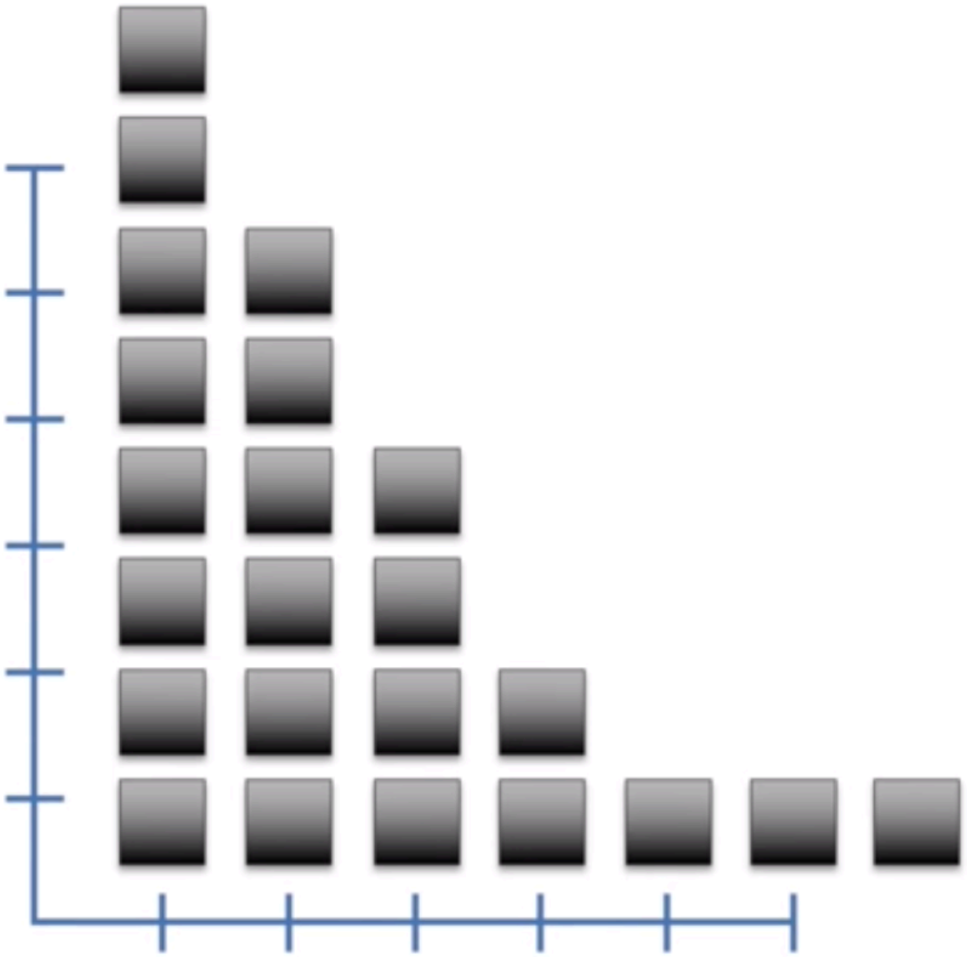


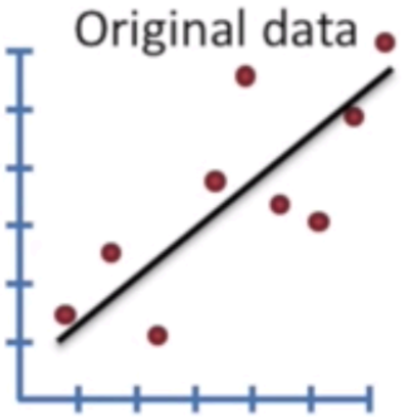


Repeat with yet another set of random data.

$$F = \frac{\text{SS(mean)} - \text{SS(fit)} / p_{\text{extra}}}{\text{SS(fit)} / (n - p_{\text{fit}})} \rightarrow F = 1$$



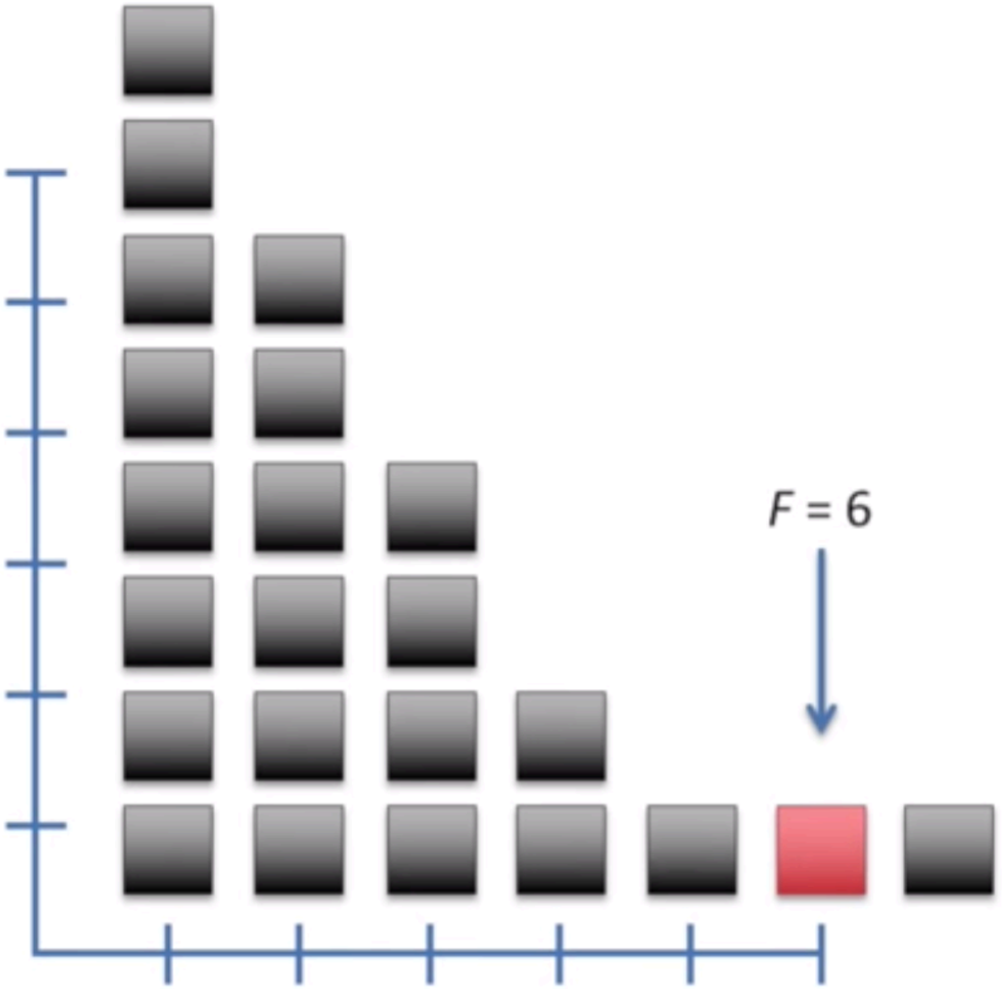




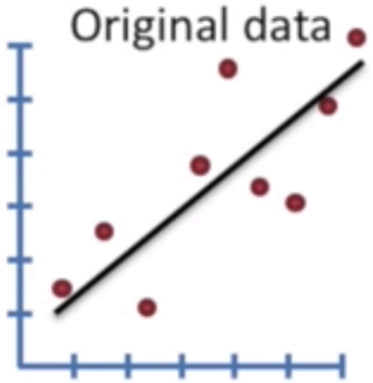
Original data

$$F = \frac{\text{SS(mean)} - \text{SS(fit)} / p_{\text{extra}}}{\text{SS(fit)} / (n - p_{\text{fit}})}$$

$= 6$

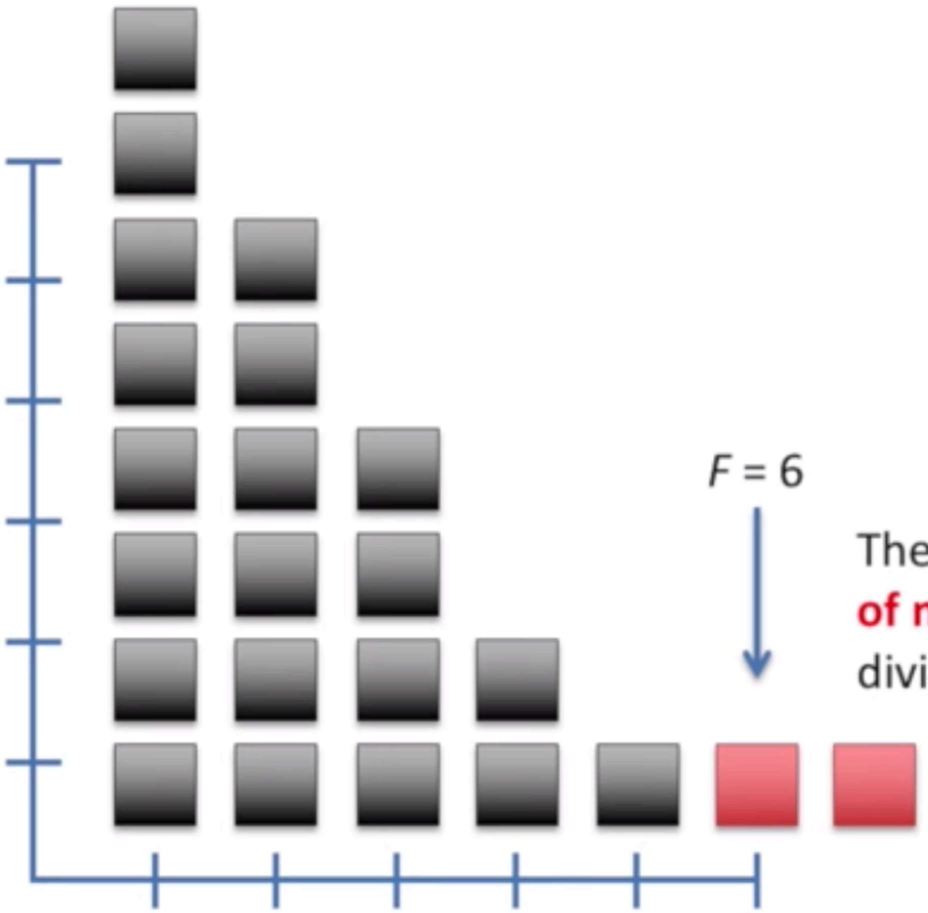


$F = 6$

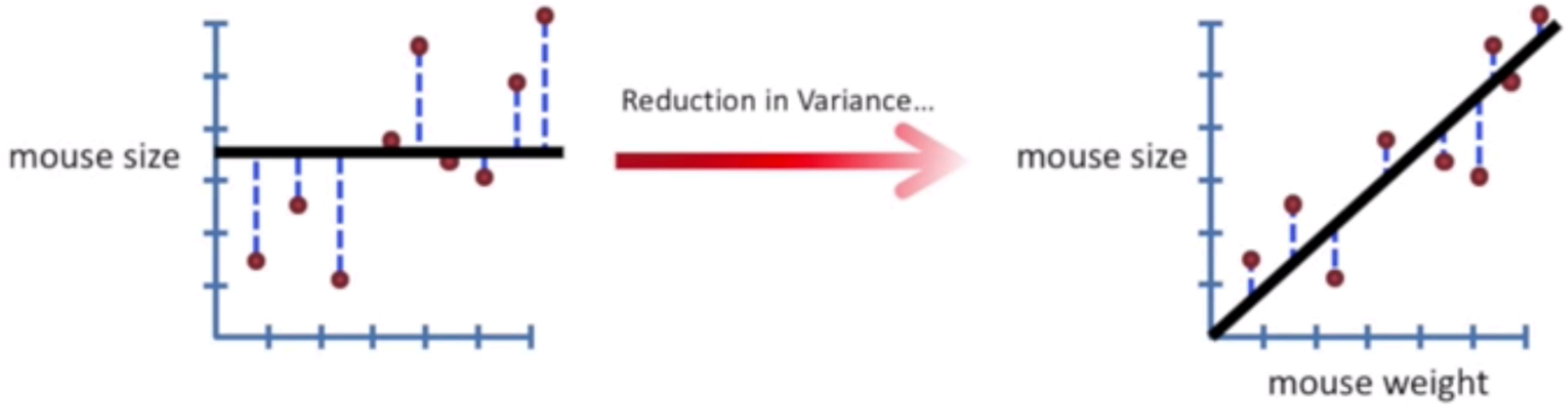


$$F = \frac{\text{SS(mean)} - \text{SS(fit)} / p_{\text{extra}}}{\text{SS(fit)} / (n - p_{\text{fit}})}$$

$$= 6$$



The *p*-value is **number of more extreme values** divided by all the values.



In this particular example, $R^2 = 0.6$, meaning we saw a 60% reduction in variation once we took mouse weight into account.

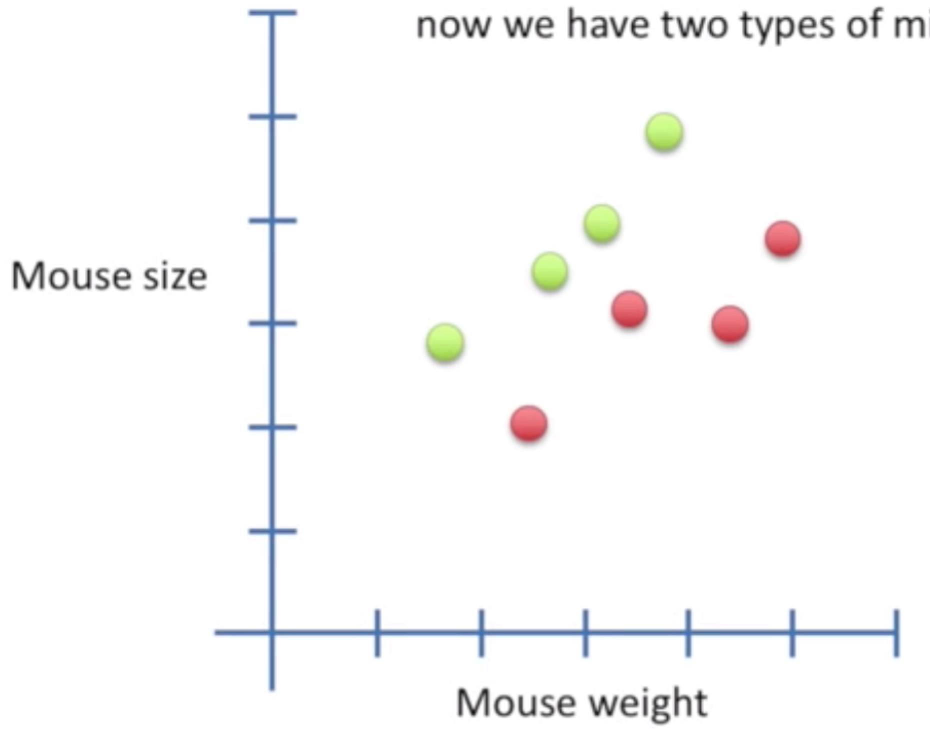
$$R^2 = \frac{\text{The variation in mouse size explained by weight}}{\text{The variation in mouse size without taking weight into account}}$$

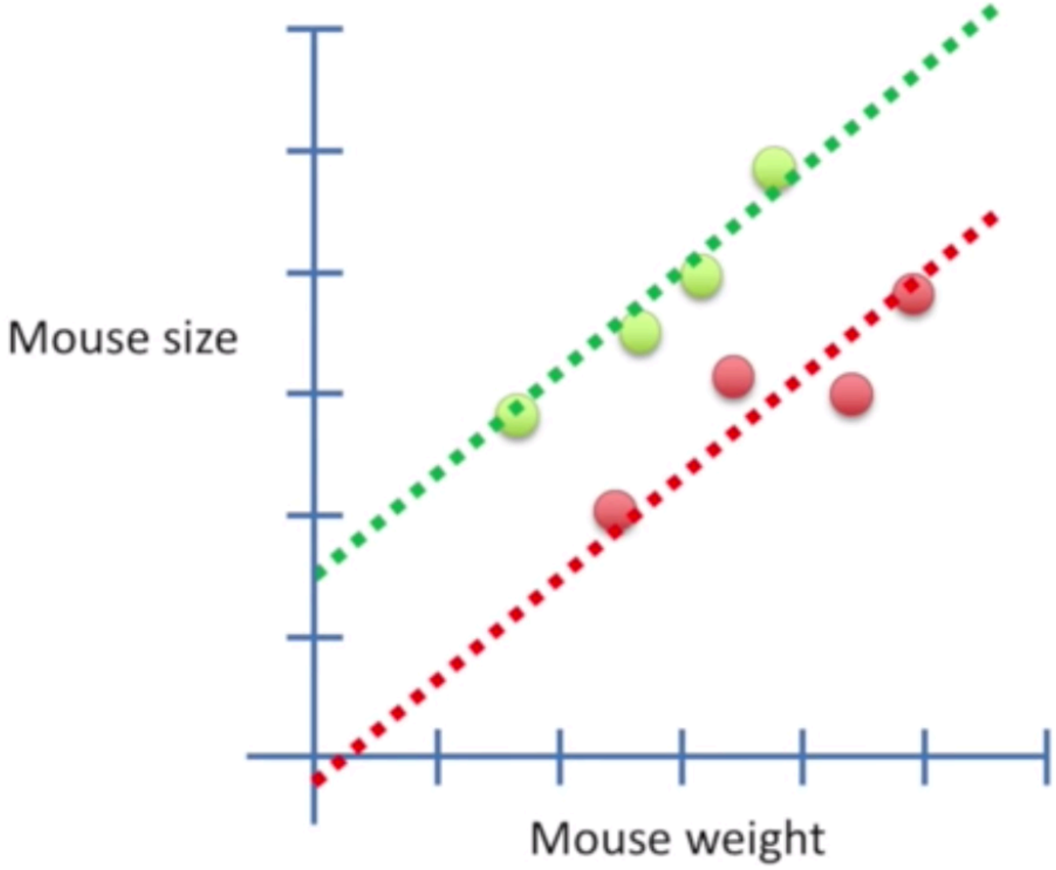
$$F = \frac{\text{SS(mean)} - \text{SS(fit)} / p_{\text{extra}}}{\text{SS(fit)} / (n - p_{\text{fit}})}$$

$$= 6$$

The *p*-value is **number of more extreme values** divided by all the values.

We're back to the relationship between mouse weight and mouse size. However, now we have two types of mice…

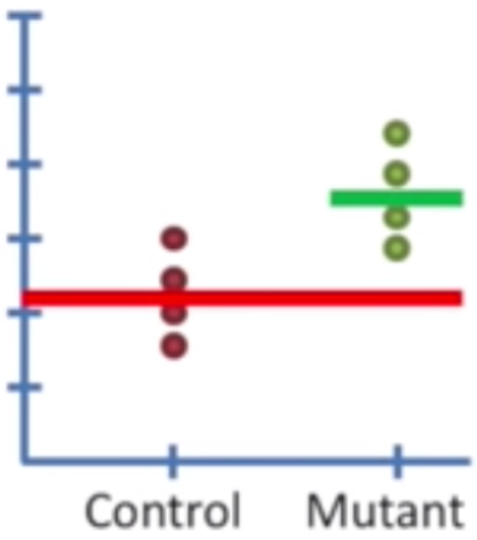


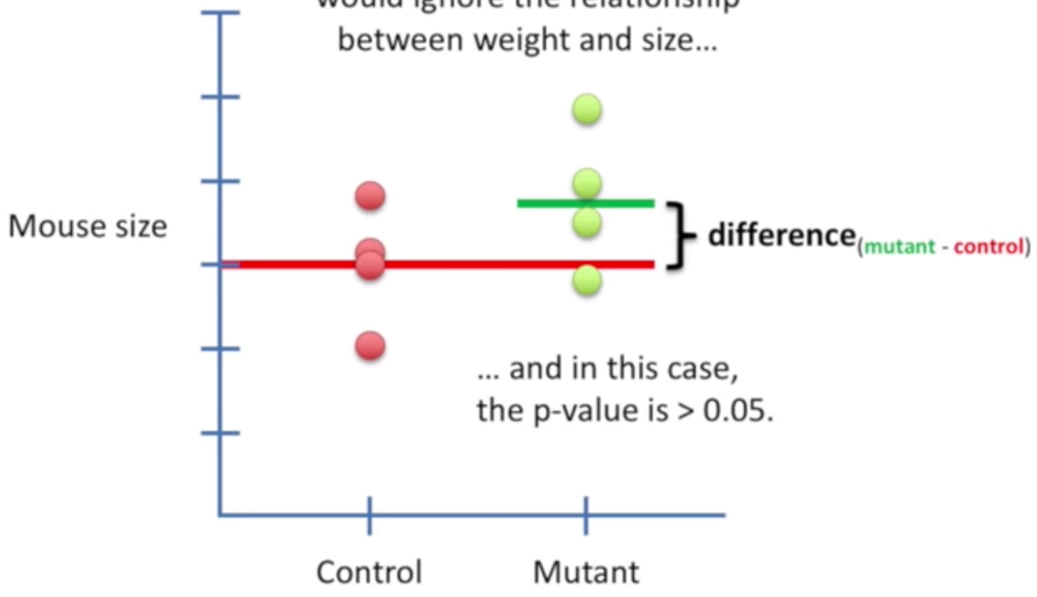


Can we use statistics to test if there is a significant difference between the two types of mice?

t-test

$y = \text{mean}_{\text{control}} + \text{difference}_{(\text{mutant - control})}$



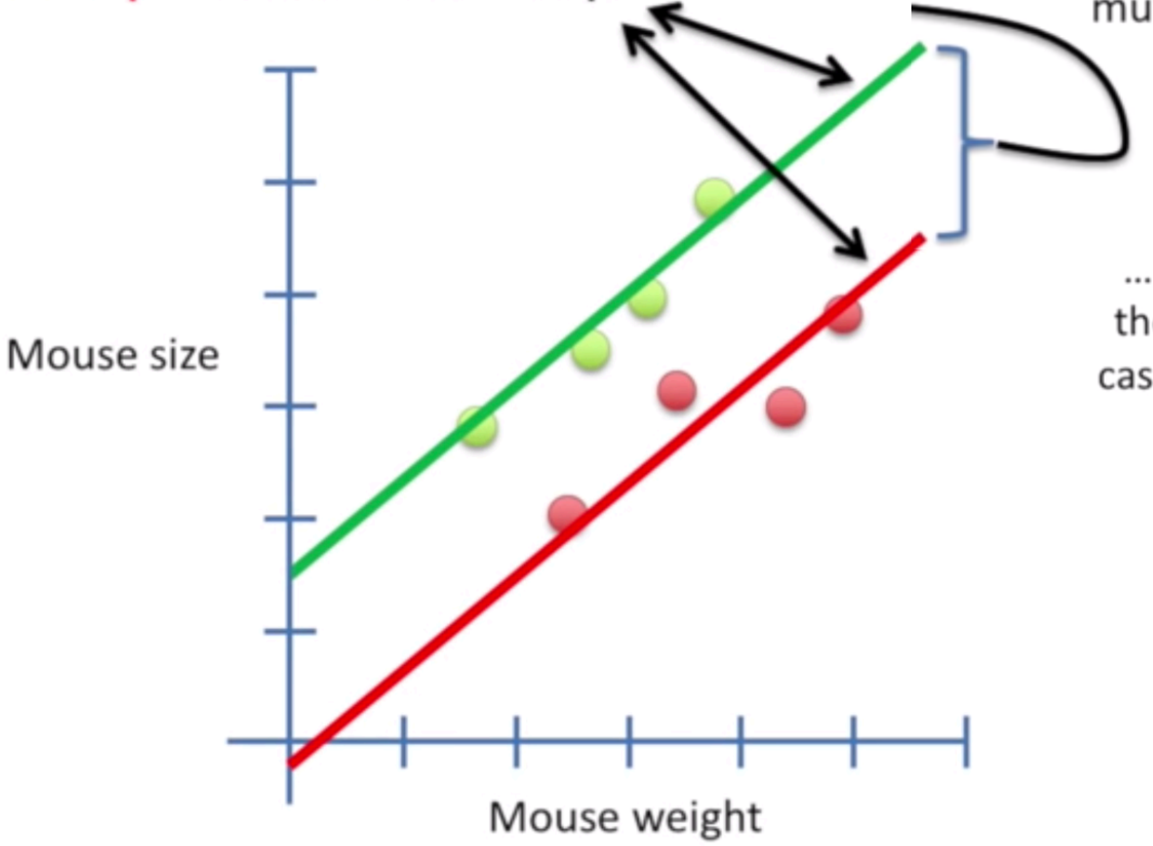


On the other hand, a normal t-test would ignore the relationship between weight and size...

... and in this case, the p-value is > 0.05.

generalised linear regression

y = control intercept + mutant offset + slope



... a term for the mutant mouse offset...

... and lastly, a term for the slope (which, in this case, is the same for both types of mice).

Sample	Treatment
Sample1	Treatment A
Sample 2	Control
Sample 3	Treatment A
Sample 4	Control
Sample 5	Treatment A
Sample 6	Control

Let's now consider this parameterization:

C= Baseline expression
T_A= Baseline expression + effect of treatment

So the set of parameters are:

C = Control (mean expression of the control)
a = T_A – Control (mean change in expression under treatment

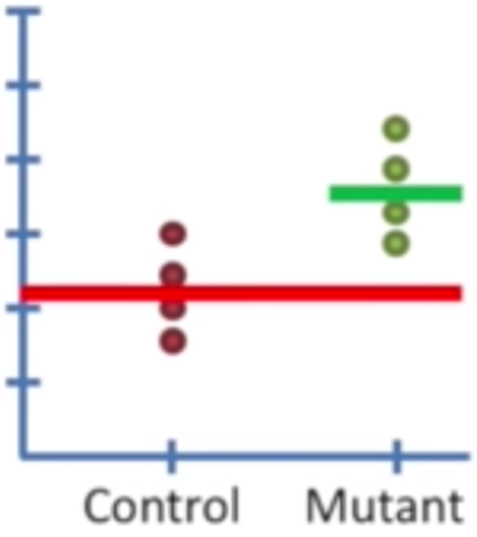
$$\begin{array}{l} \text{Sample 1}\\ \text{Sample 2}\\ \text{Sample 3}\\ \text{Sample 4}\\ \text{Sample 5}\\ \text{Sample 6} \end{array} \begin{bmatrix} S1\\ S2\\ S3\\ S4\\ S5\\ S6 \end{bmatrix} = \begin{array}{cc} \text{Intercept} & \text{Treatment A}\\ \hline 1 & 1\\ 1 & 0\\ 1 & 1\\ 1 & 0\\ 1 & 1\\ 1 & 0 \end{array} \begin{bmatrix} \beta_0\\ a \end{bmatrix}$$

Parameters (coefficients, levels of the variable)

Design Matrix

Intercept measures the baseline expression.
a measures now the differential expression between Treatment A and Control

t-test



$y = \textbf{mean}_{\textbf{control}} + \textbf{difference}_{(\textbf{mutant - control})}$

$y = 1 \times \textbf{mean}_{\textbf{control}} + 0 \times \textbf{difference}_{(\textbf{mutant - control})}$

design matrices in the context of using 1's and 0's to turn parts of the equation "on" or "off"...

$$\boxed{\begin{matrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{matrix}}$$

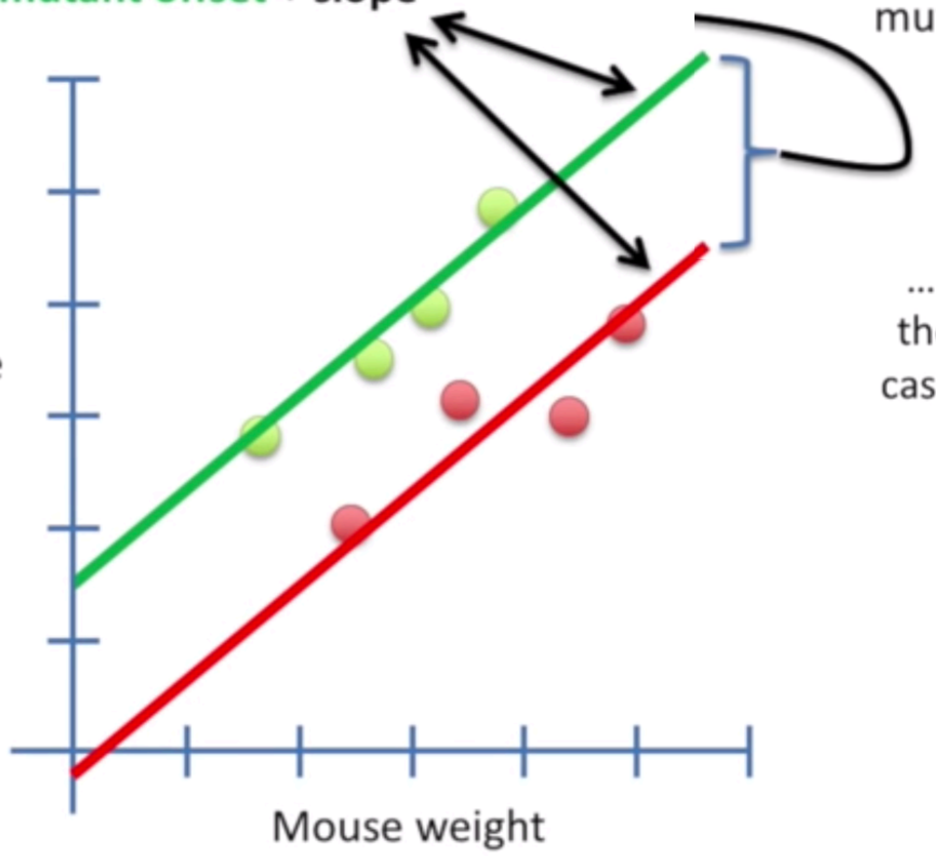
Remember that the numbers in the first column are multiplied by $\textbf{mean}_{\textbf{control}}$

...and the numbers in the second column are multiplied by $\textbf{difference}_{(\textbf{mutant - control})}$

Multiplying $\textbf{mean}_{\textbf{control}}$ by 1 "turns it on" by just letting it be.

$\textbf{difference}_{(\textbf{mutant - control})}$ by 0 makes it 0 and that "turns it off".

y = control intercept + mutant offset + slope



... a term for the mutant mouse offset...

... and lastly, a term for the slope (which, in this case, is the same for both types of mice).

This means we need a design matrix where the first column is 1's...

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

This means that both lines intercept the y-axis at some point...

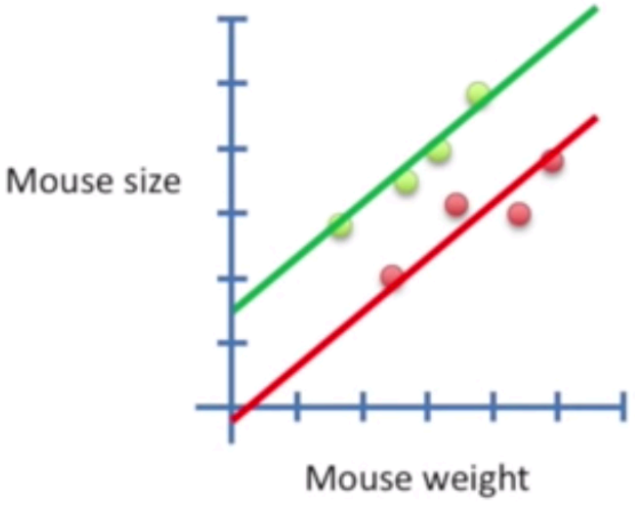
...the second column indicates whether the mutant offset is on or off...

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

mutant offset is "off" for the control mice...

...and "on" for the mutant mice. This allows the mutants to have their own y-intercept.

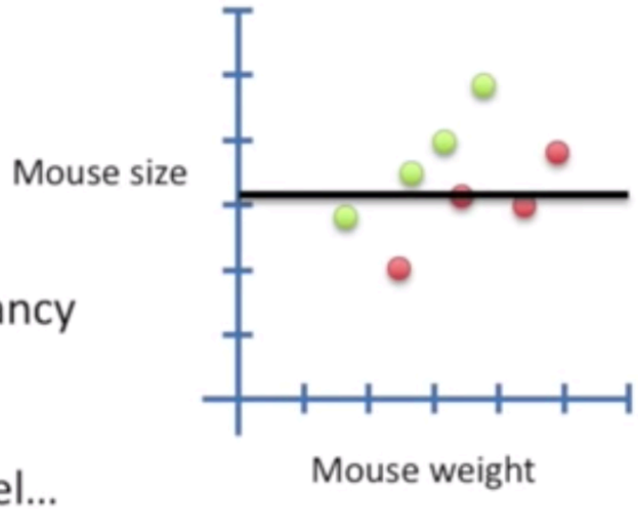
y = **control intercept** + **mutant offset** + **slope**



Now compare the fancy model...

...to a simpler model...

y = **overall mean**

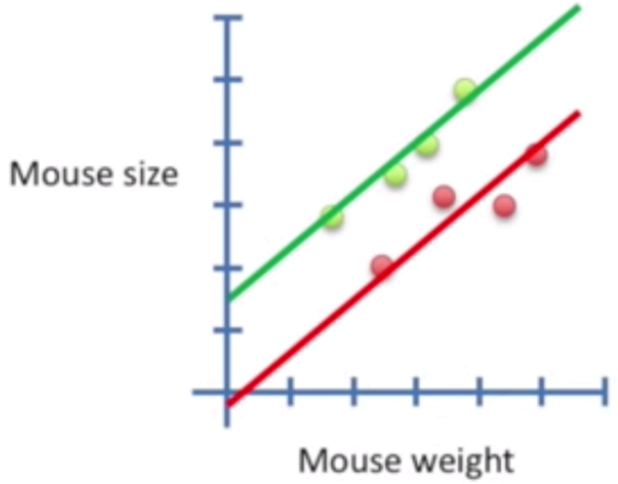


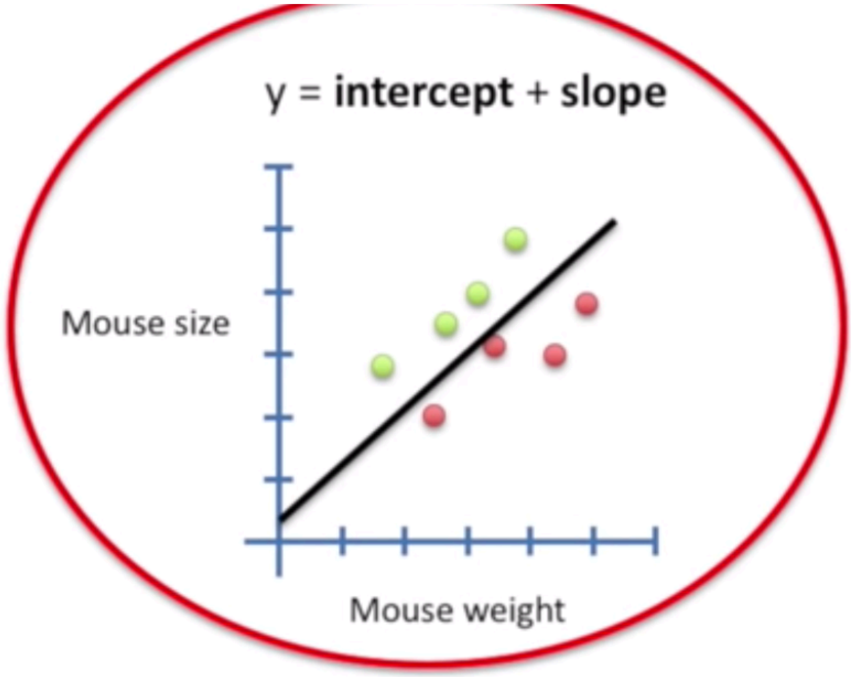
$$F = \frac{SS(\text{simple}) - SS(\text{fancy}) / (p_{\text{fancy}} - p_{\text{simple}})}{SS(\text{fancy}) / (n - p_{\text{fancy}})} = 21.88$$

p-value = 0.003

The small p-value says that taking weight and mouse type into account is significantly better at predicting size than just using the average size.

y = **control intercept** + **mutant offset** + **slope**




y = **intercept** + **slope**

This model takes weight into account, but ignores the fact that some mice are normal and others are mutants.

$$F = \frac{SS(\text{simple}) - SS(\text{fancy}) / (p_{\text{fancy}} - p_{\text{simple}})}{SS(\text{fancy}) / (n - p_{\text{fancy}})} = 32.6$$

p-value = 0.0023

This small p-value suggests that using both weight and mouse type is better at predicting mouse size than weight alone.

Linear Models

- The observed value of Y is a linear combination of the effects of the independent variables

Arbitrary number of independent variables

$$E(Y) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k$$

Polynomials are valid

$$E(Y) = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \ldots + \beta_p X_1^p$$

$$E(Y) = \beta_0 + \beta_1 \log(X_1) + \beta_2 f(X_2) + \ldots + \beta_k X_k$$

We can use functions of the variables if the effects are linear

Smooth functions: not exactly the same as the so-called ***additive models***

- If we include categorical variables the model is called **General Linear Model**

In DeSeq2

RNA-seq raw count data follows a negative binomial distribution, as reported in the previous slide.

The DESeq2 authors model the data i.e. imply that for each gene is built a regression model of the data such that it is possibile to make statistical inferences from the data.

The normalised counts, are used to compute a logistic regression model fro each gene **with the negative binomial distribution.**

Once modelled each gene, the way to derive a P value for each model coefficient is by the Wald Test.

In DeSeq2

RNA-seq raw count data follows a negative binomial distribution, as reported in the previous slide.

The DESeq2 authors model the data i.e. imply that for each gene is built a regression model of the data such that it is possibile to make statistical inferences from the data.

The normalised counts, are used to compute a logistic regression model fro each gene **with the negative binomial distribution.**

Once modelled each gene, the way to derive a P value for each model coefficient is by the Wald Test.

The likelihood ratio (LTR) test

We are working with models, therefore we would like to do hypothesis tests on coefficients or contrasts of those models:

- We fit two models M1 without the coefficient to test and M2 with the coefficient.

- We compute the likelihoods of the two models (L1 and L2) and obtain LRT=-2log(L1 /L2) that has a known distribution under the null hypothesis that the two models are equivalent. This is also known as model selection

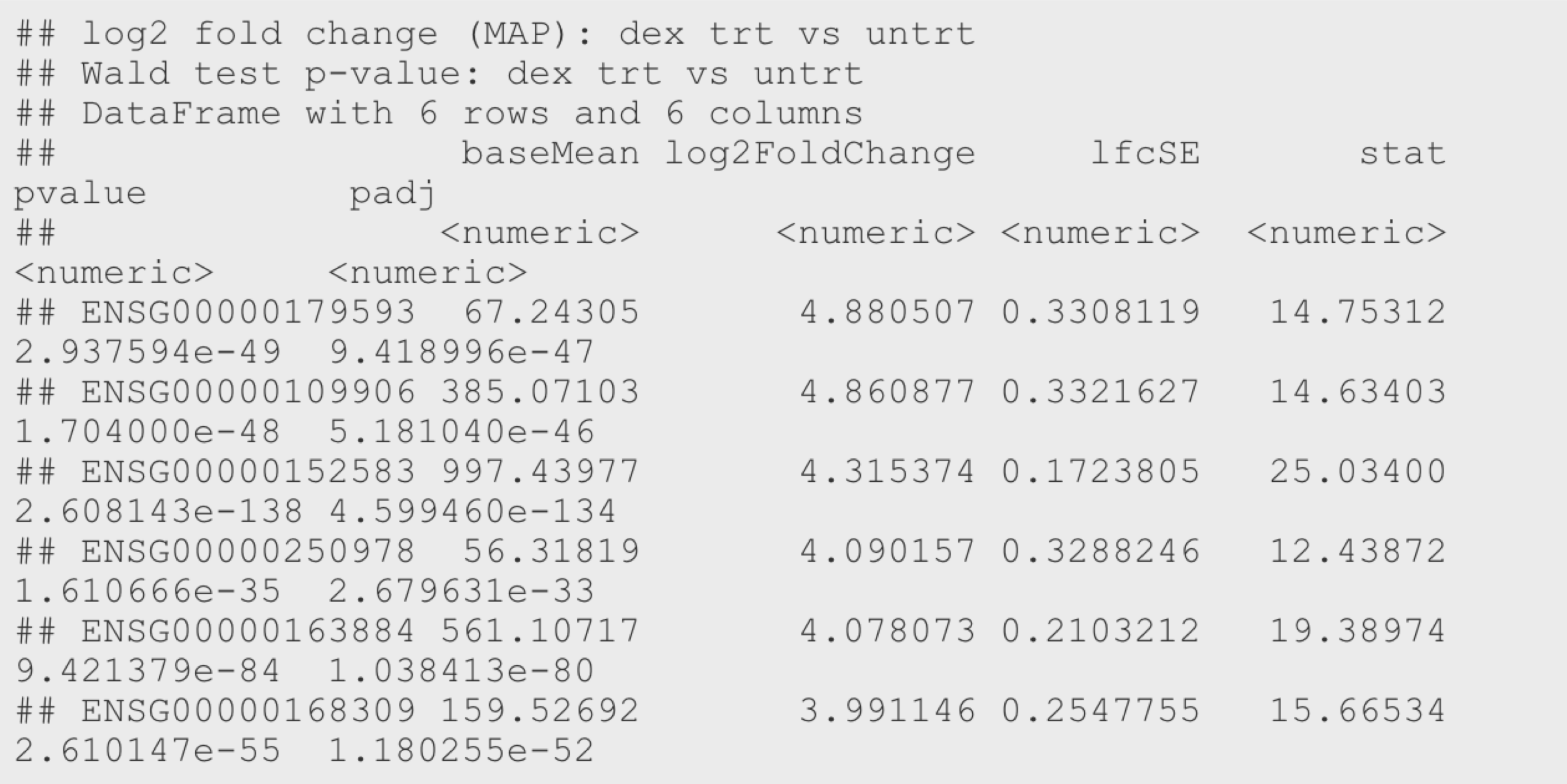
```
ddsLRT = DESeq(dds, test="LRT", full=~sex+age+smoke+disease, reduced=~sex+age+smoke)
```

The LRT It tests whether the increase in the log likelihood from the additional coefficients would be expected if those coefficients were equal to zero. It doesn't mean the reduced model is a good model or a good fit.

The adjusted p-value computed stay for: if it is small, then for the set of genes with those small adjusted p-values, the additional coefficient in full and not in reduced increased the log likelihood more than would be expected if their true value was zero.

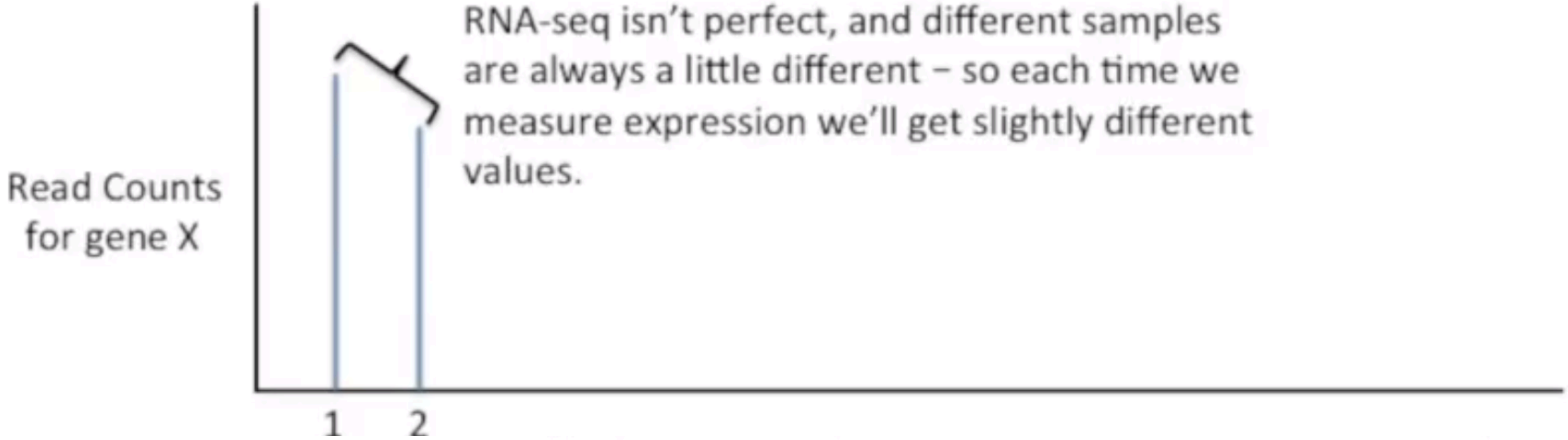
```
ddsLRT = DESeq(dds, test="LRT", full=~sex+age+smoke+geneA+disease, reduced=~sex+age+smoke+disease)
```

Differential Expressed Genes – FDR

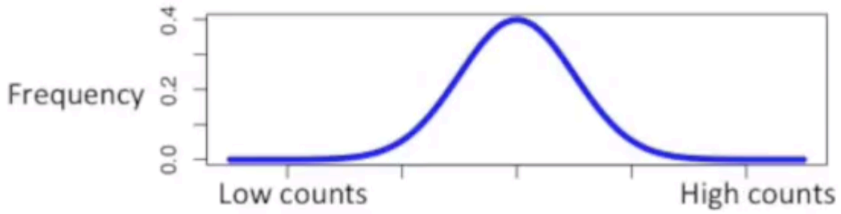


```
## log2 fold change (MAP): dex trt vs untrt
## Wald test p-value: dex trt vs untrt
## DataFrame with 6 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE       stat
pvalue         padj
##                 <numeric>      <numeric> <numeric>  <numeric>
<numeric>     <numeric>
## ENSG00000179593  67.24305       4.880507 0.3308119   14.75312
2.937594e-49  9.418996e-47
## ENSG00000109906 385.07103       4.860877 0.3321627   14.63403
1.704000e-48  5.181040e-46
## ENSG00000152583 997.43977       4.315374 0.1723805   25.03400
2.608143e-138 4.599460e-134
## ENSG00000250978  56.31819       4.090157 0.3288246   12.43872
1.610666e-35  2.679631e-33
## ENSG00000163884 561.10717       4.078073 0.2103212   19.38974
9.421379e-84  1.038413e-80
## ENSG00000168309 159.52692       3.991146 0.2547755   15.66534
2.610147e-55  1.180255e-52
```

Measuring gene expression in RNA-seq experiments



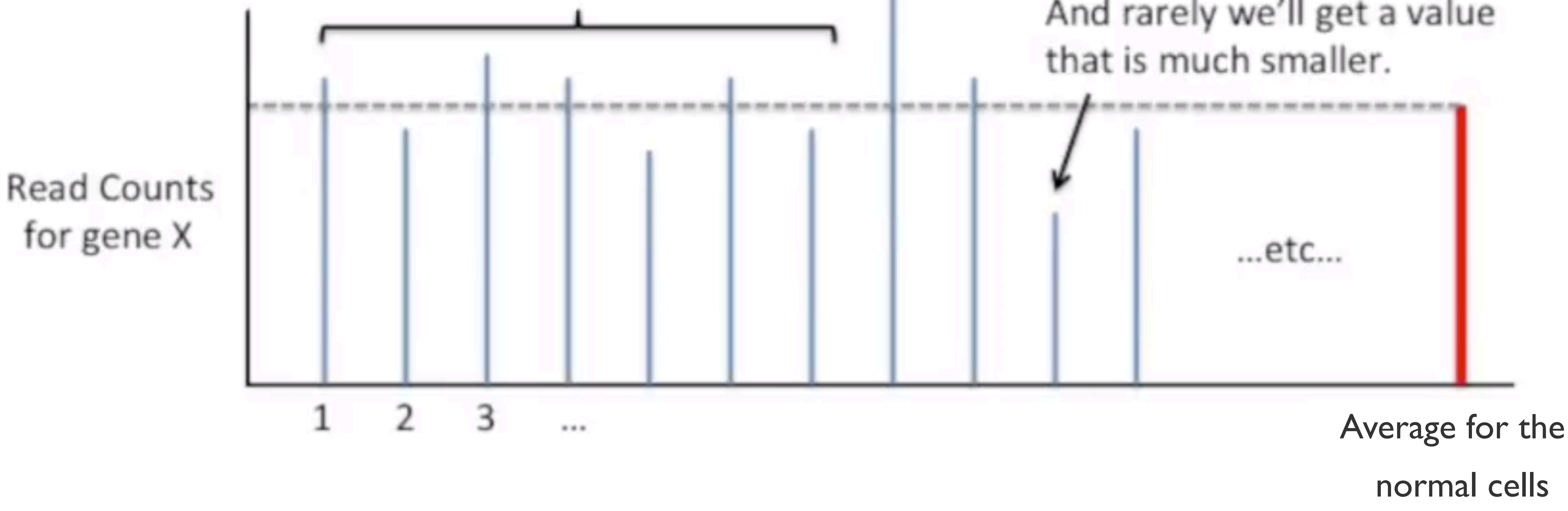
RNA-seq isn’t perfect, and different samples are always a little different – so each time we measure expression we’ll get slightly different values.

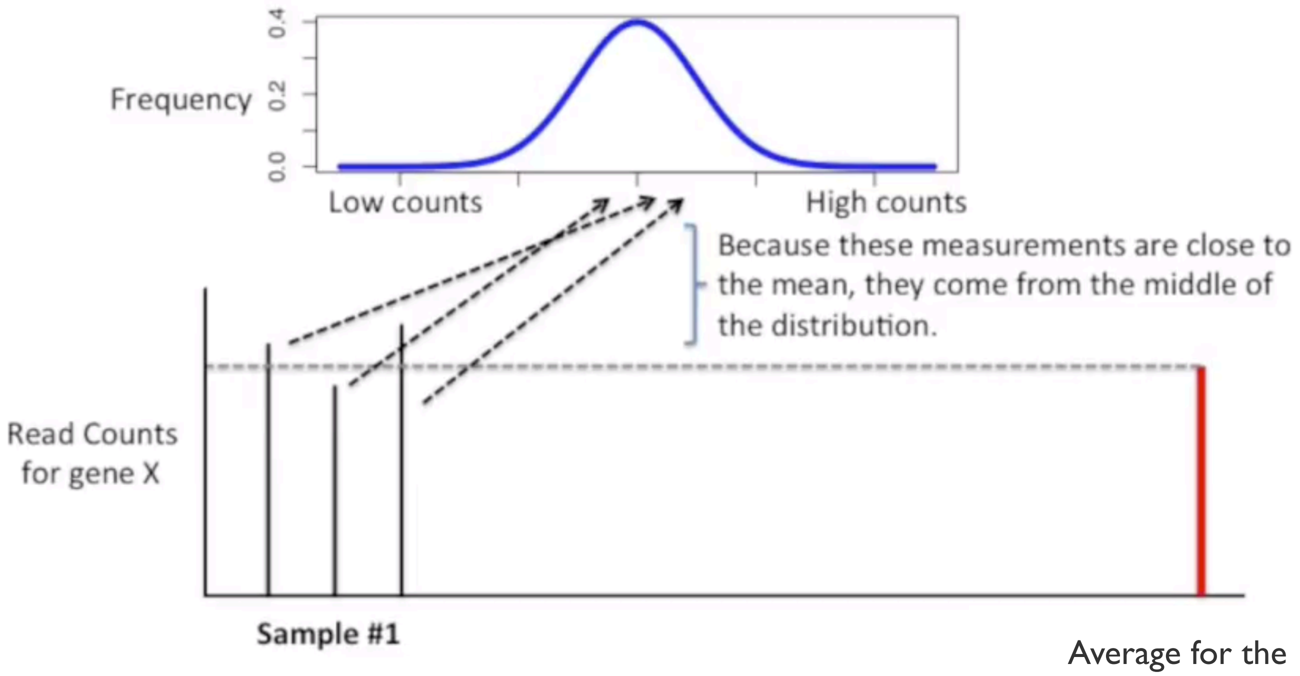


Most values are close to the mean.

Rarely, we’ll get a value that is much larger than the mean…

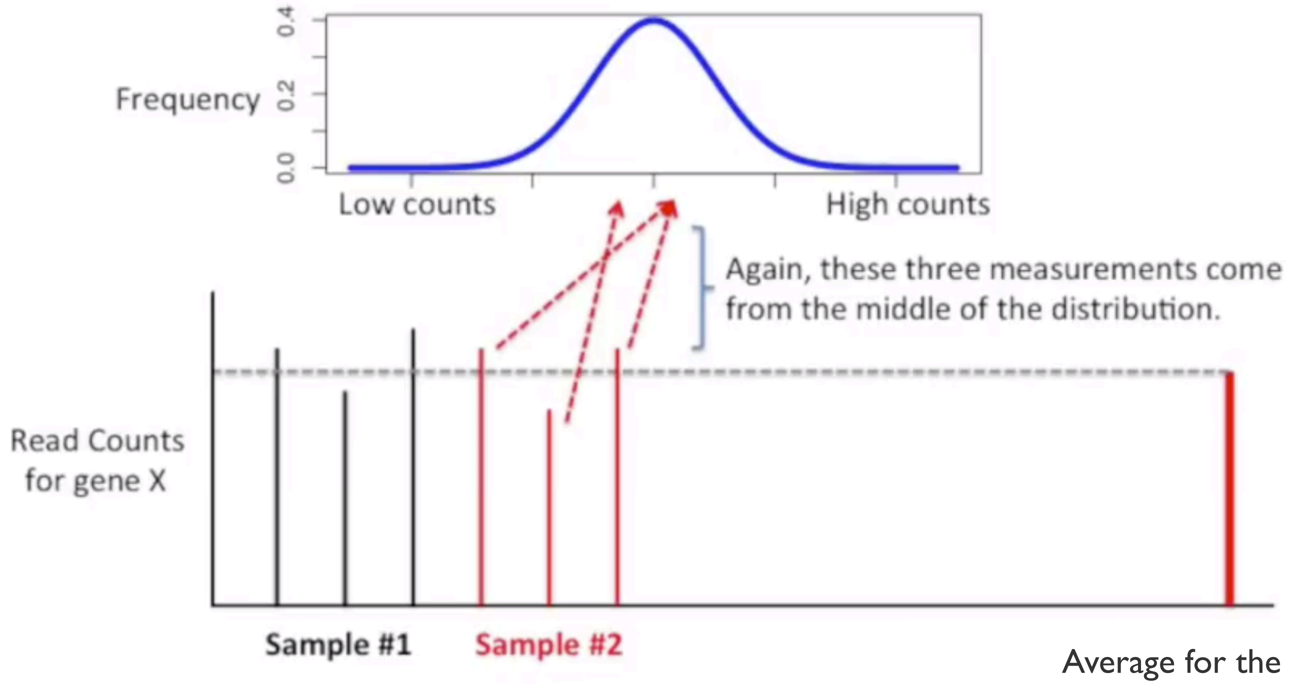
And rarely we’ll get a value that is much smaller.





Because these measurements are close to the mean, they come from the middle of the distribution.

Average for the normal cells

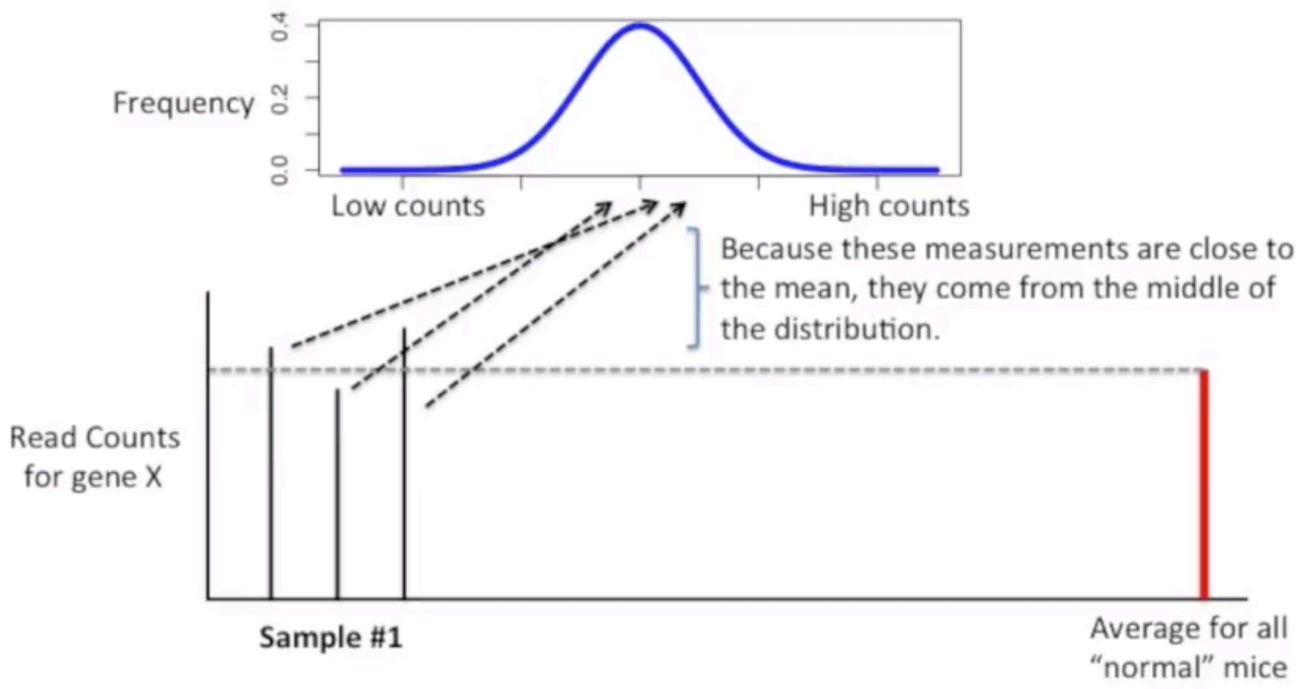


Again, these three measurements come from the middle of the distribution.

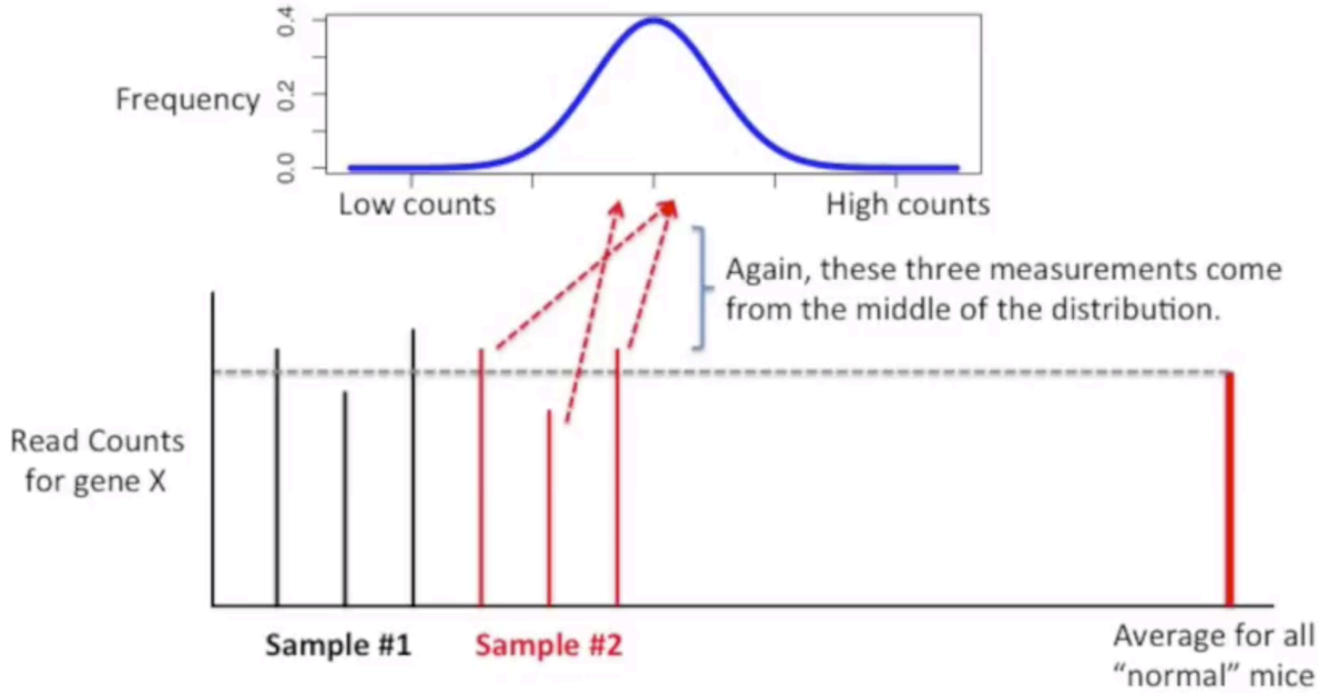
Average for the normal cells

Sample #1 normal cells: epithelian cells, reference genes or genes not specific of that cells.

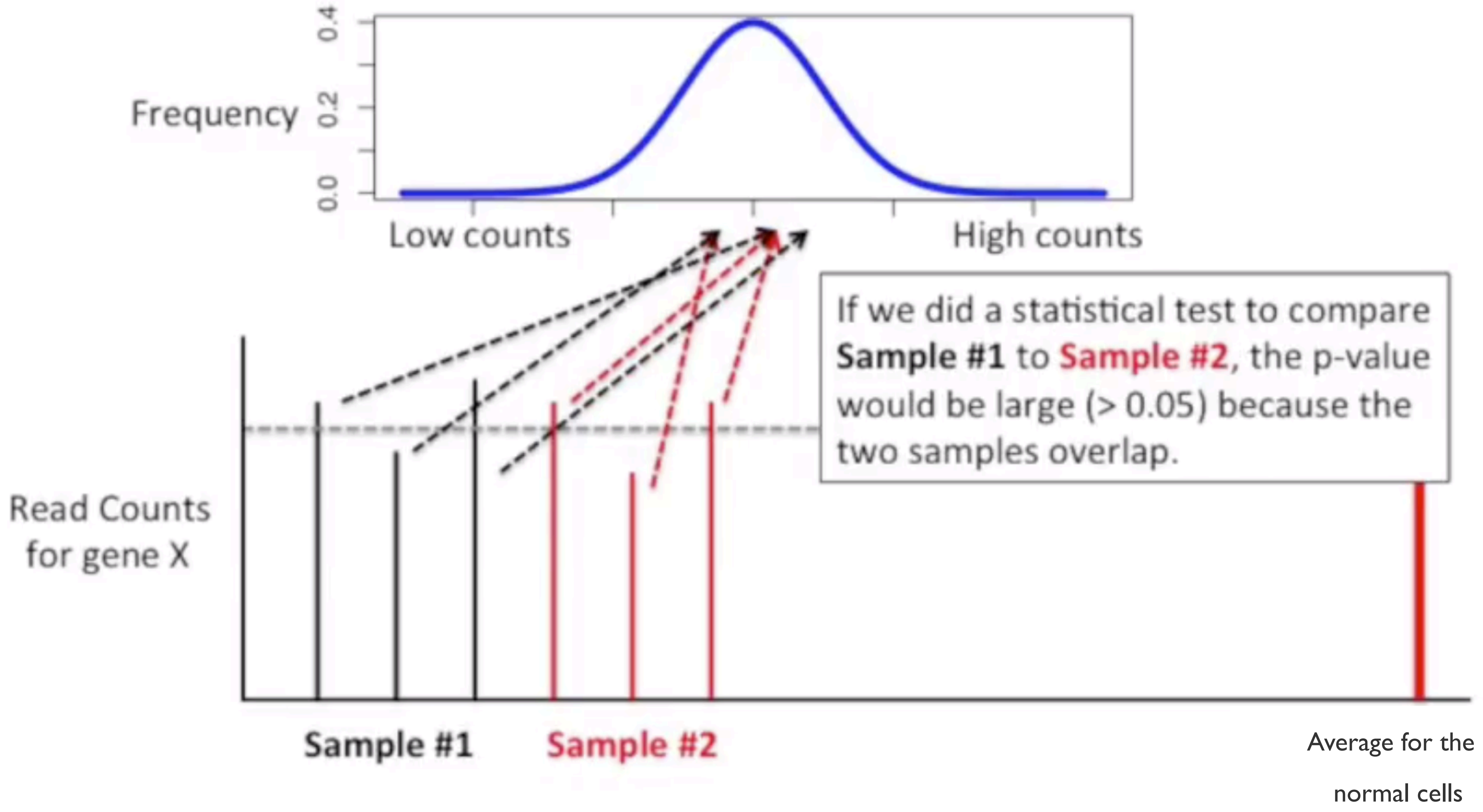
Sample #2 normal cells: red blood cells, reference genes or genes not specific of that cells.



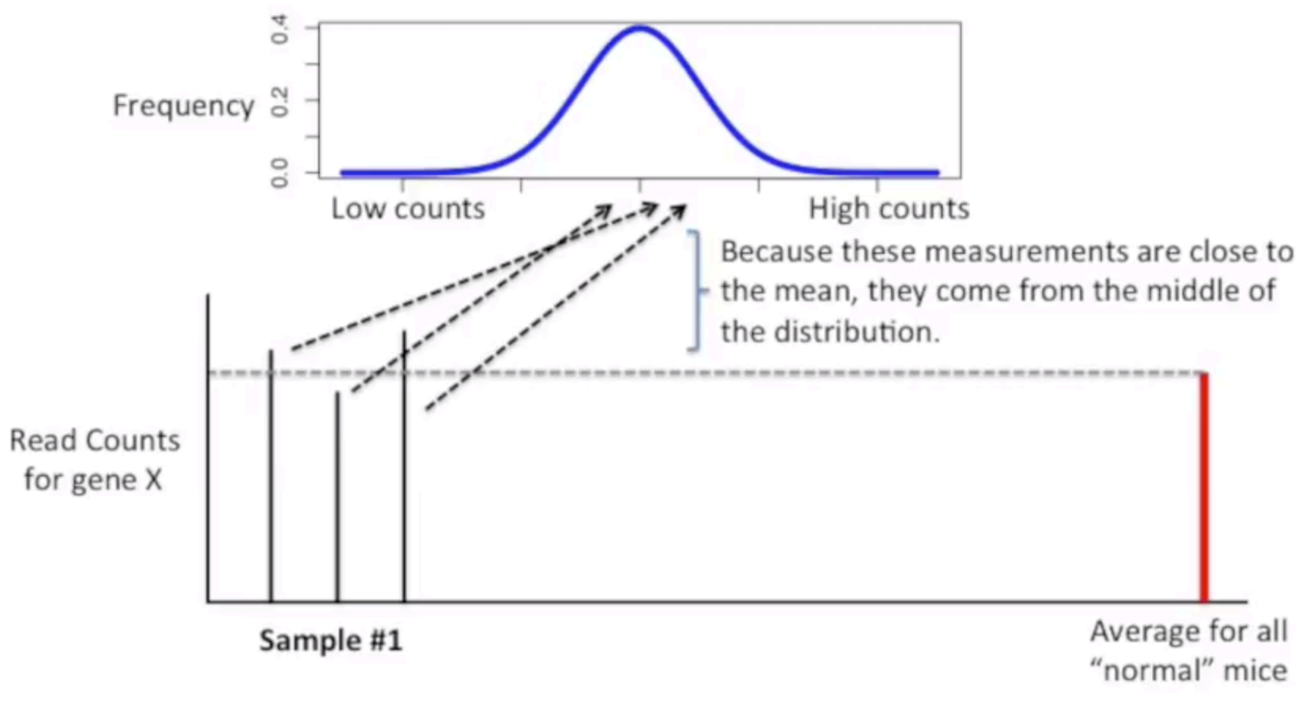
Because these measurements are close to the mean, they come from the middle of the distribution.



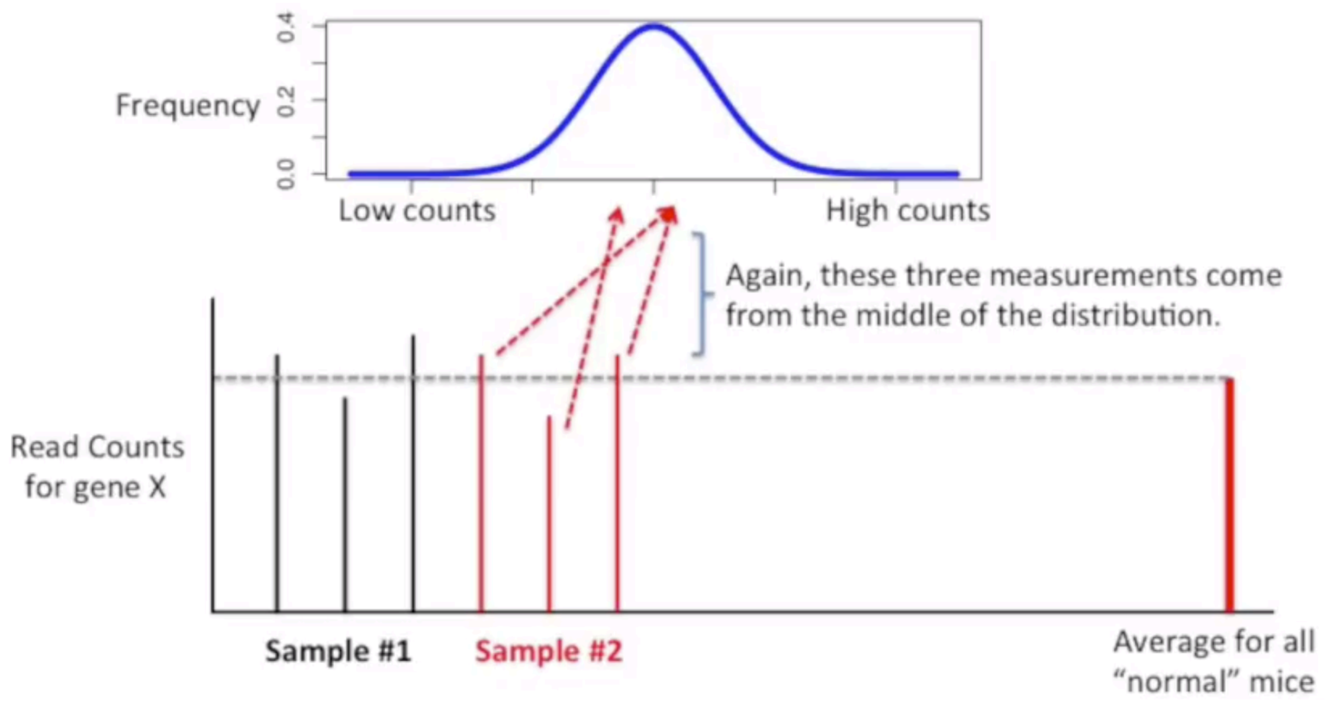
Again, these three measurements come from the middle of the distribution.



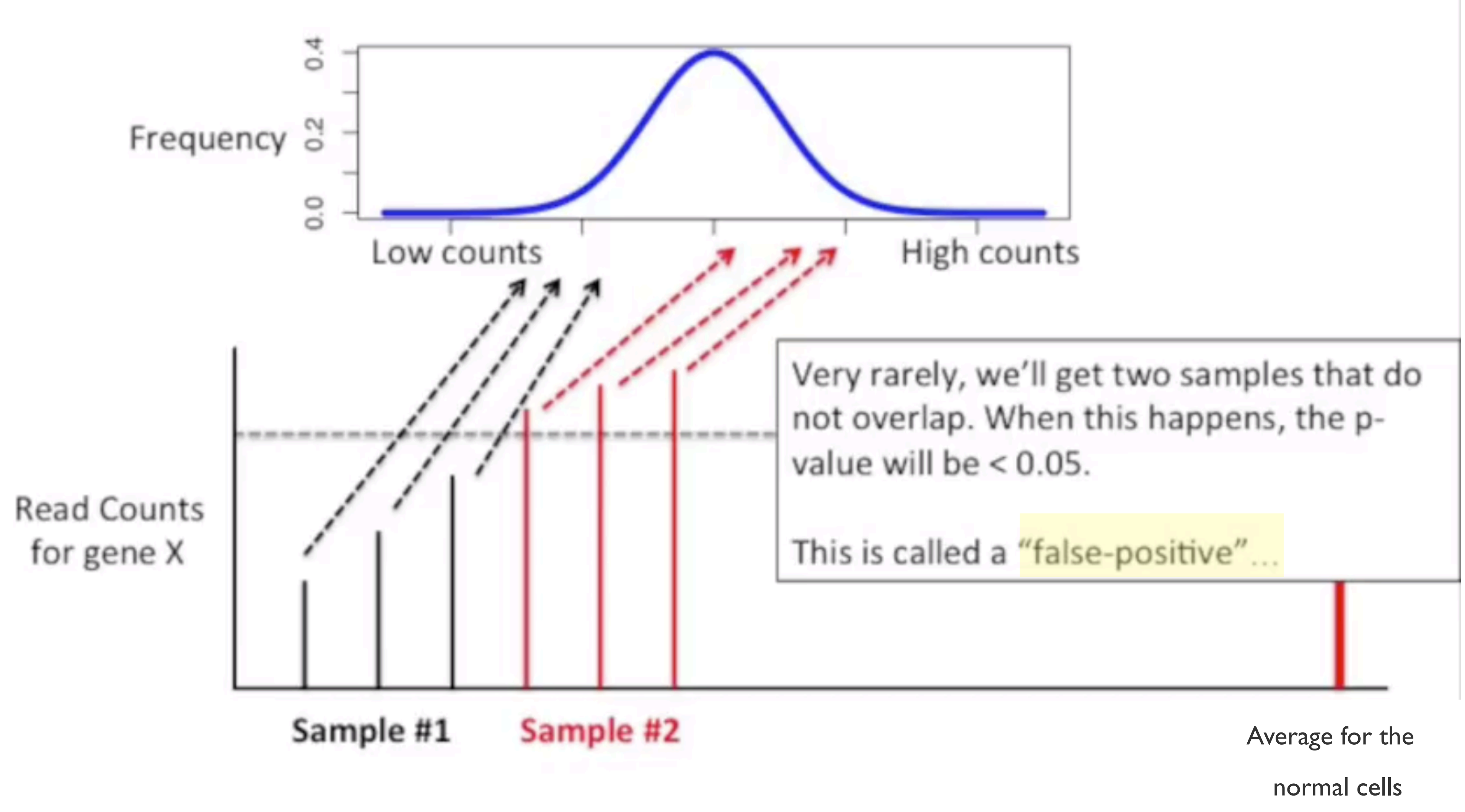
If we did a statistical test to compare **Sample #1** to **Sample #2**, the p-value would be large (> 0.05) because the two samples overlap.



Because these measurements are close to the mean, they come from the middle of the distribution.

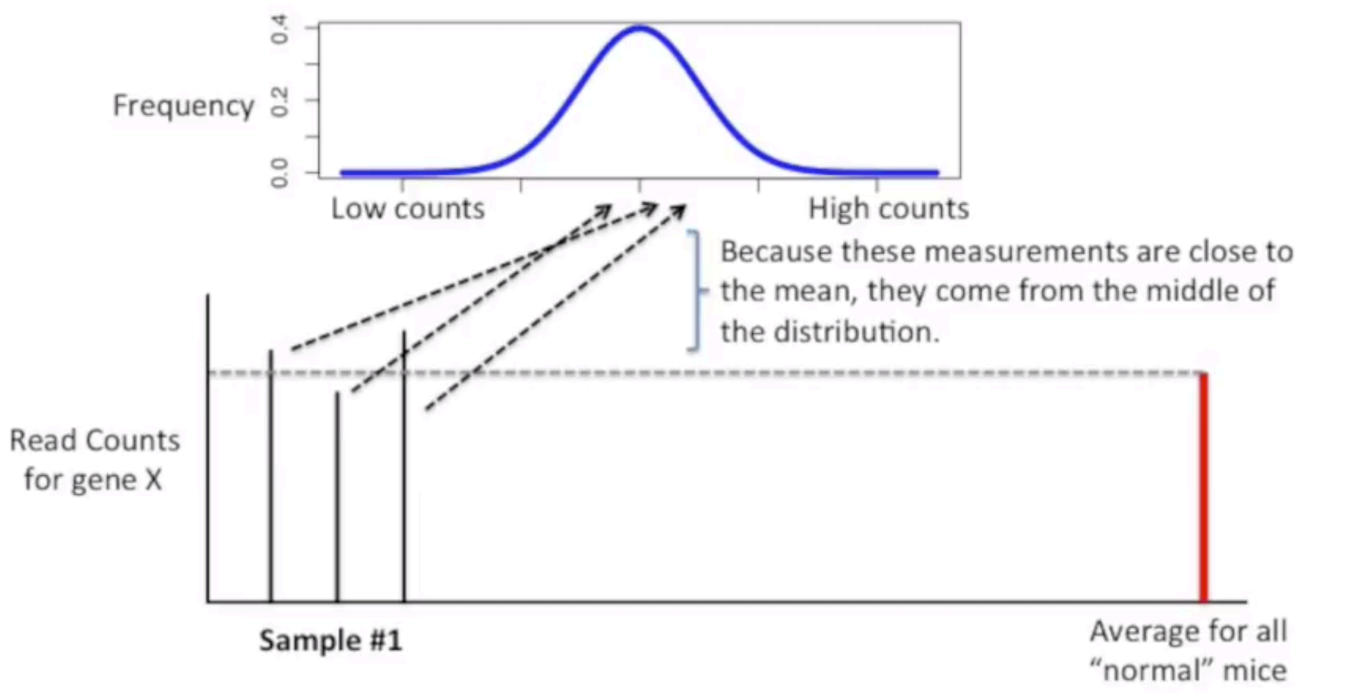


Again, these three measurements come from the middle of the distribution.

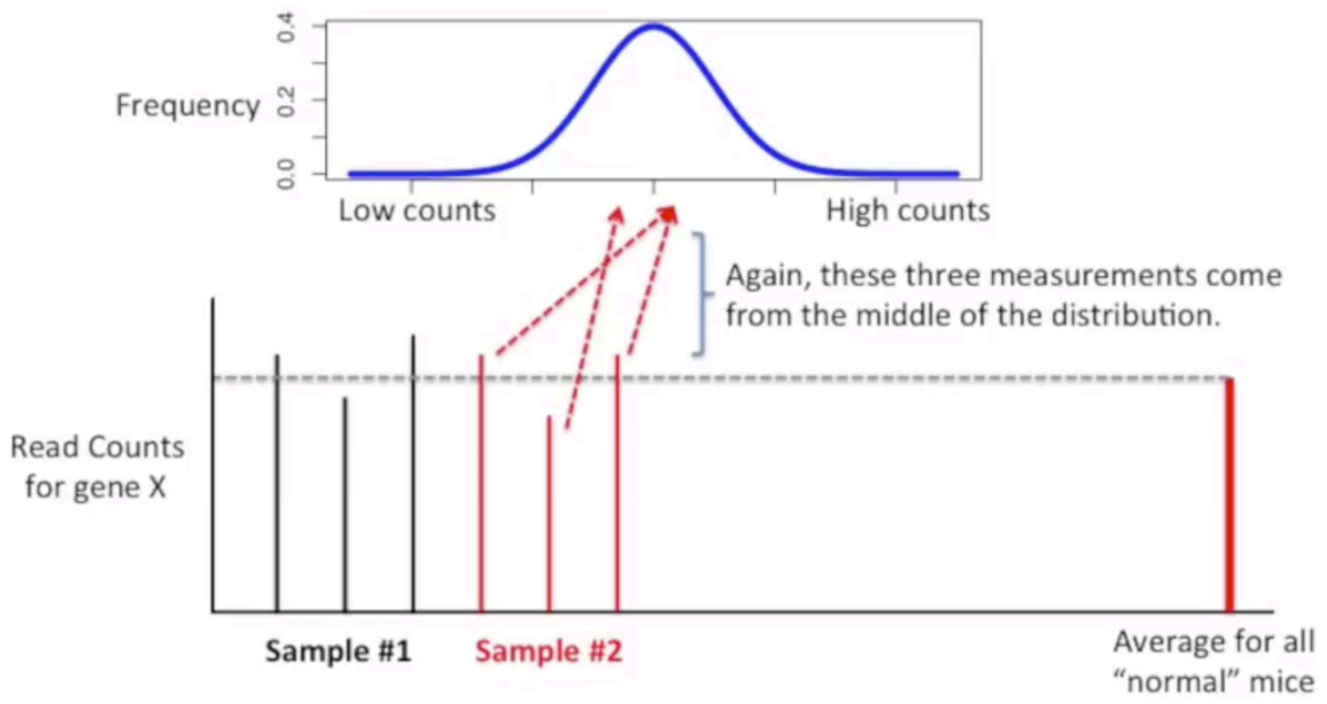


Very rarely, we'll get two samples that do not overlap. When this happens, the p-value will be < 0.05.

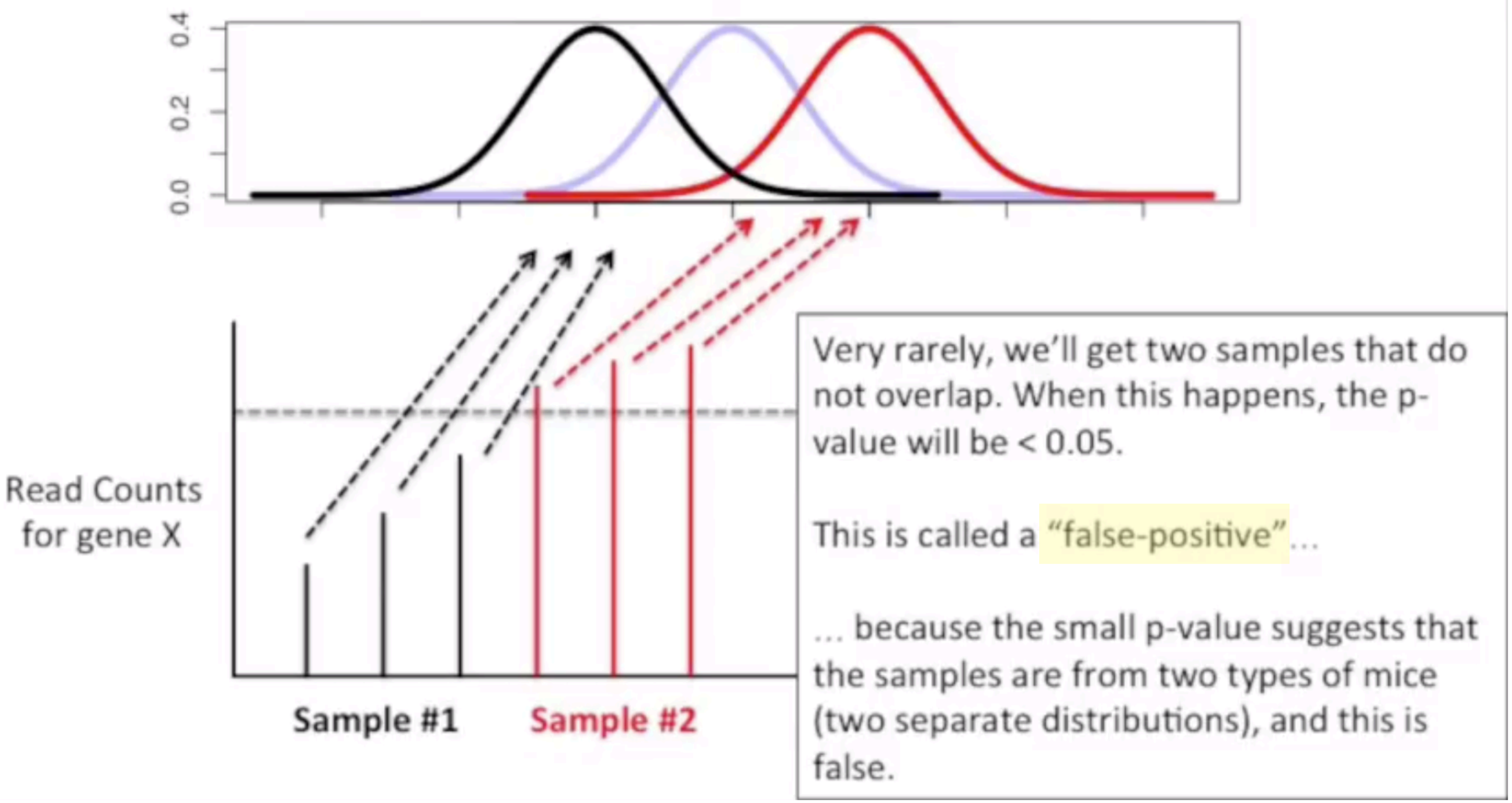
This is called a "false-positive"…



Because these measurements are close to the mean, they come from the middle of the distribution.



Again, these three measurements come from the middle of the distribution.



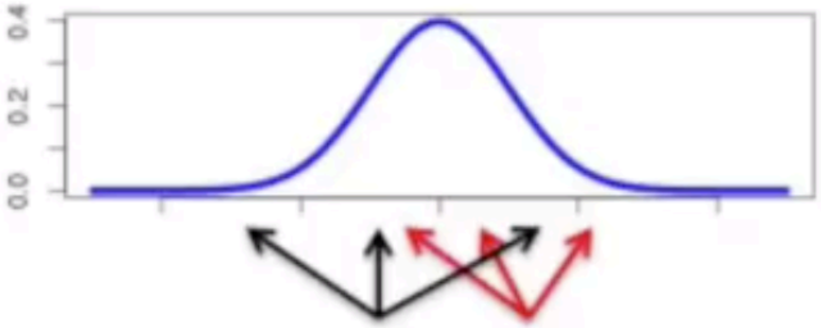
Very rarely, we'll get two samples that do not overlap. When this happens, the p-value will be < 0.05.

This is called a "false-positive"...

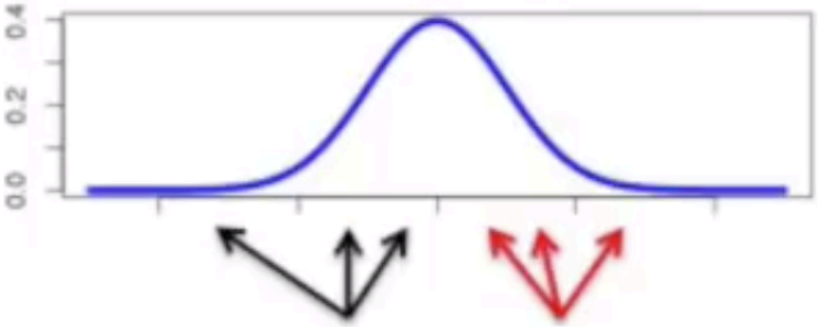
... because the small p-value suggests that the samples are from two types of mice (two separate distributions), and this is false.

Normally, false positives are rare

95% of the time the samples will overlap.



5% of the time they don't.



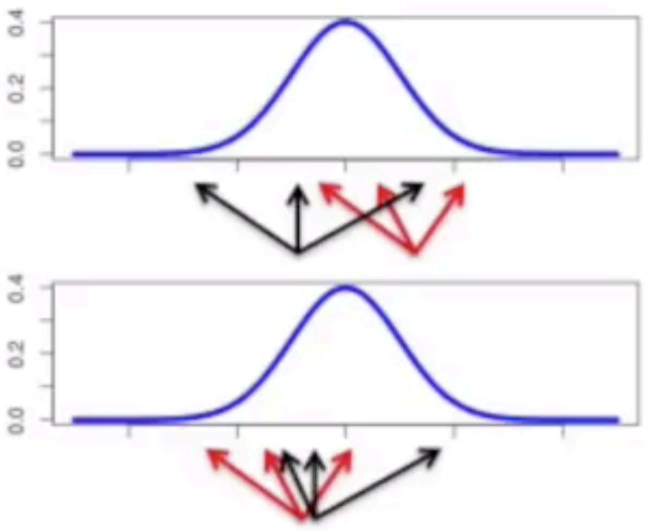
But human and mouse cells have at least 10,000 transcribed genes. If we took two samples from the same type of mice and compared all 10,000 genes…

5% of 10,000 = 500 false positives – 500 genes that appear interesting, even when they are not.

The False Discovery Rate (FDR) can control the number of false positives.

Technically, the FDR is not a method to limit false positives, but the term is used interchangeably with the methods. In particular, it is used for the "Benjamini-Hochberg method".

We'll start by generating 10,000 p-values from samples taken from the same distribution.



Test #1

p-value = 0.83

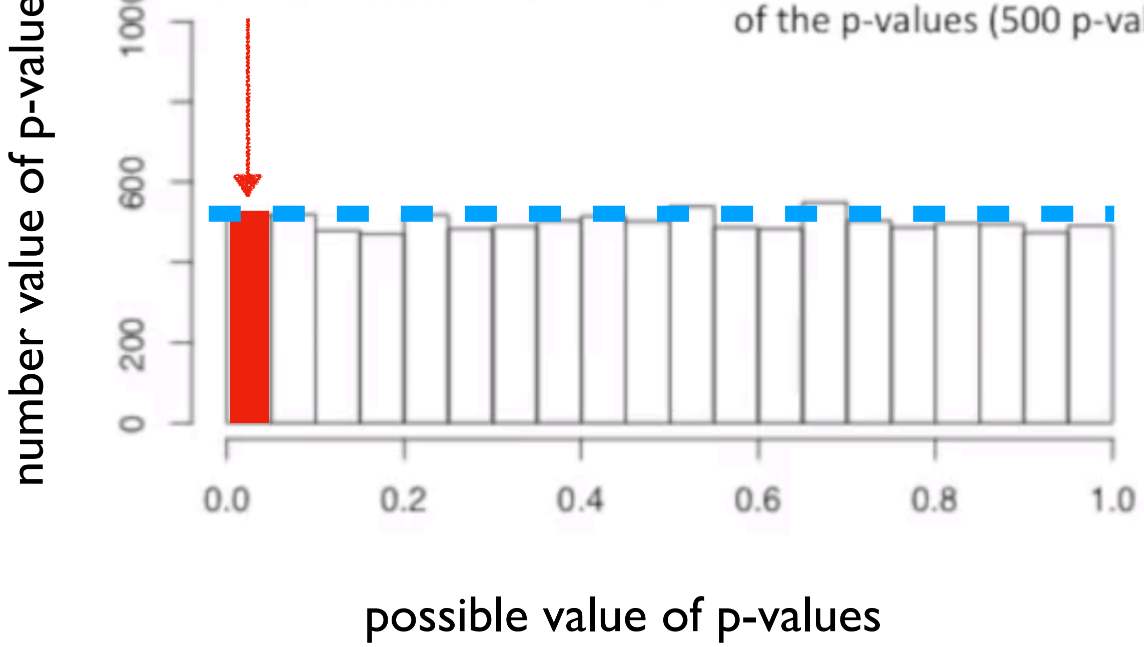
Test #2

p-value = 0.98

A histogram of 10,000 p-values generated by testing samples taken from the same distribution.

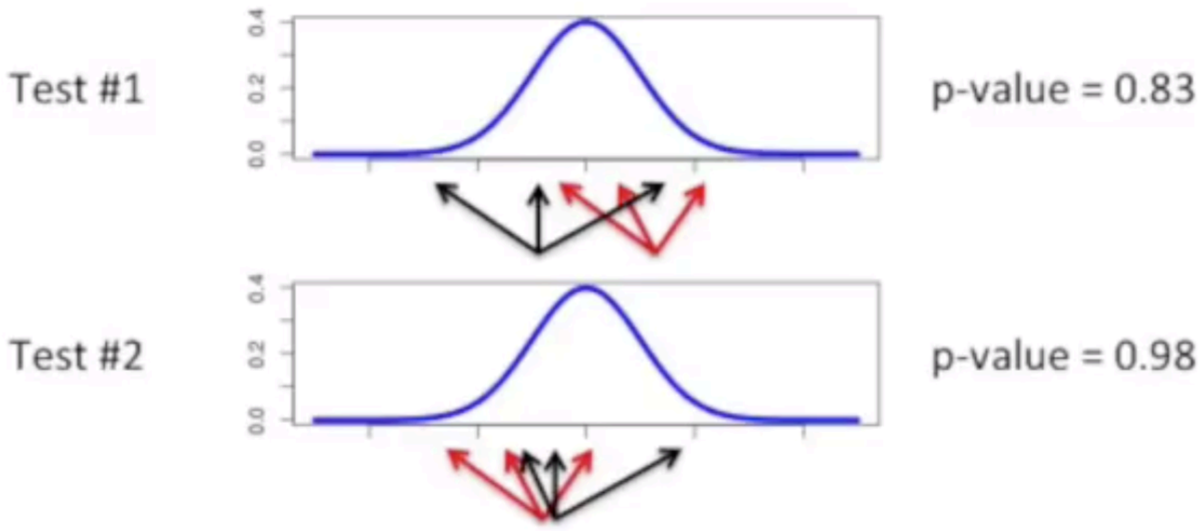
510 p-values (5.1%) are < 0.05. . .

each "bin" contains about 5% of the p-values (500 p-values per bin).

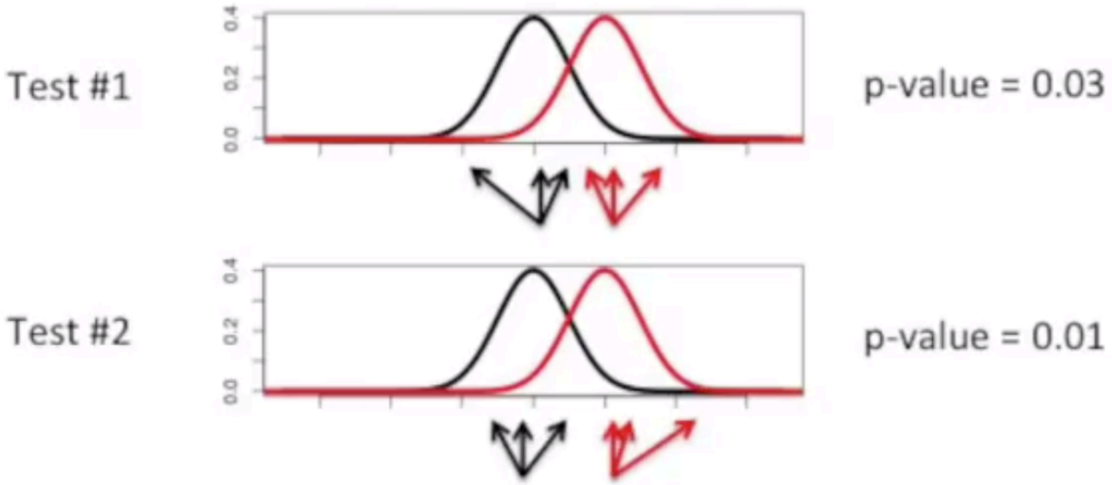


Since the p-values are uniformly distributed, there's an equal probability that a test's p-value falls into any one of these bins.

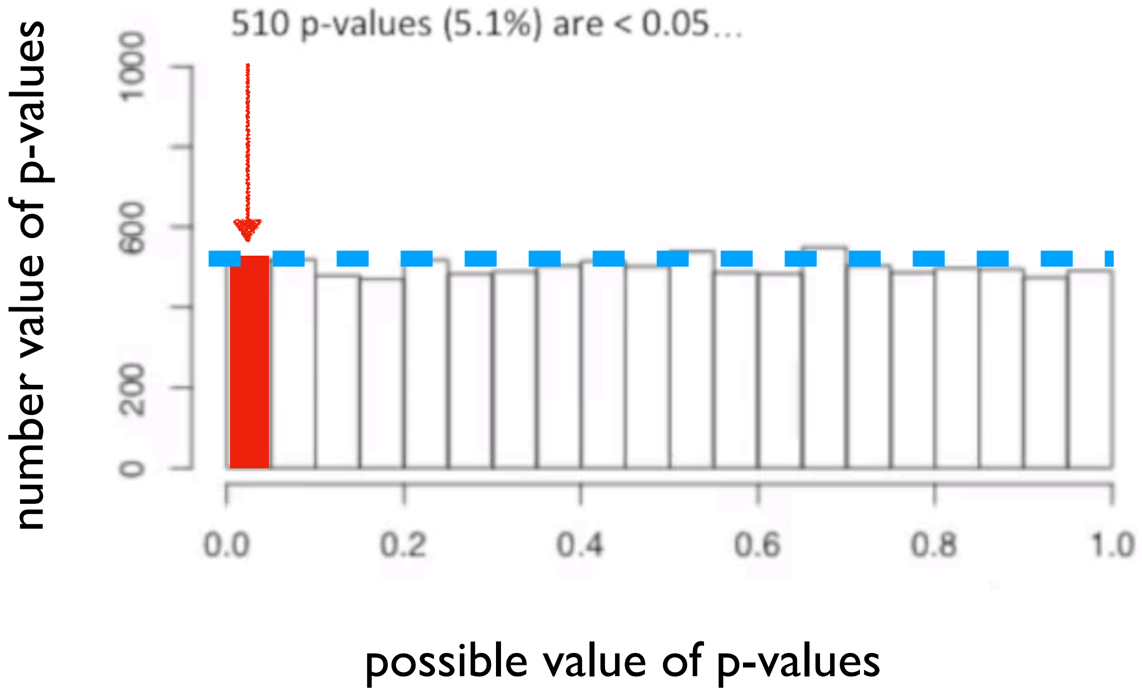
We'll start by generating 10,000 p-values from samples taken from the same distribution.



Now let's look at how p-values are distributed when they come from two different distributions.

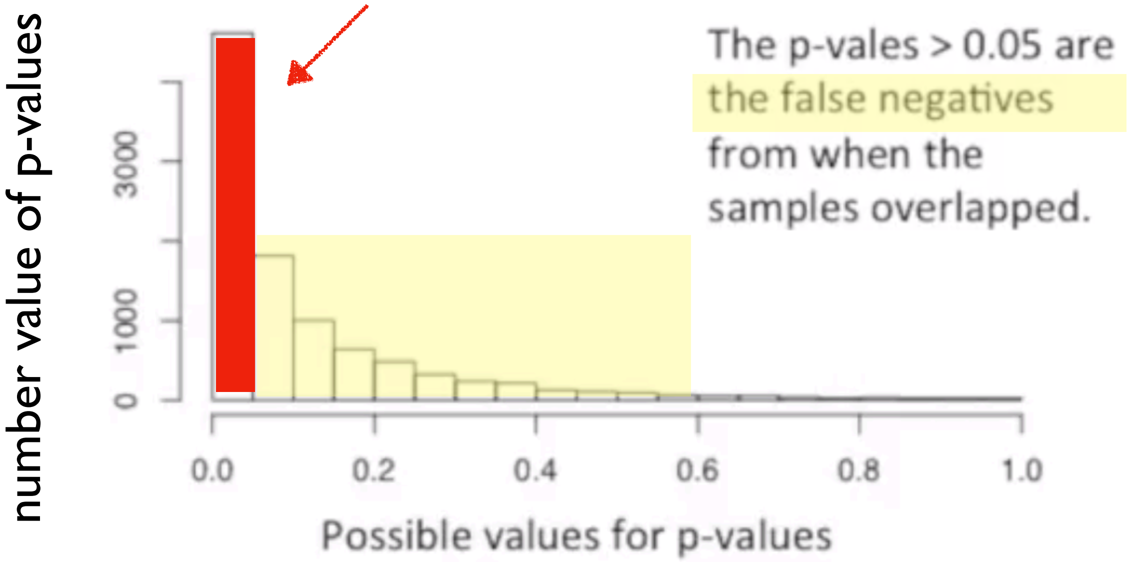


A histogram of 10,000 p-values generated by testing samples taken from the same distribution.


510 p-values (5.1%) are < 0.05…

A histogram of 10,000 p-values generated by testing samples taken from two different distributions.

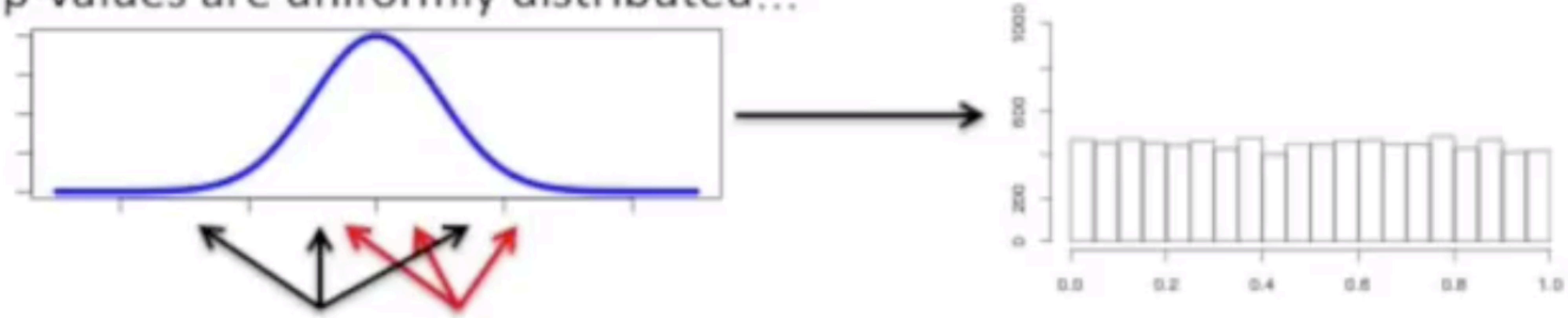
Most of the p-values are < 0.05.


The p-vales > 0.05 are the false negatives from when the samples overlapped.

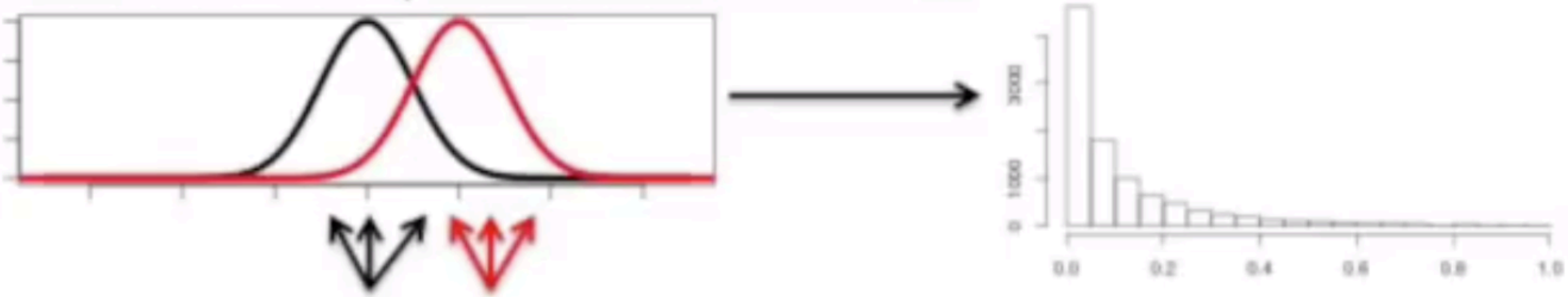
the false negative can be reduced increasing the sample size

To summarize what we know so far…

When samples come from the same distribution, the p-values are uniformly distributed…



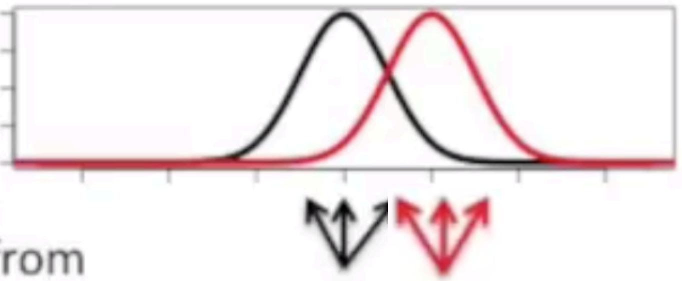
When samples come from different distributions, the p-values are heavily skewed and closer to 0…



Experiment: all the active genes in the neuronal cells,

one set of neuronal cells is treated with a drug the other set is not.

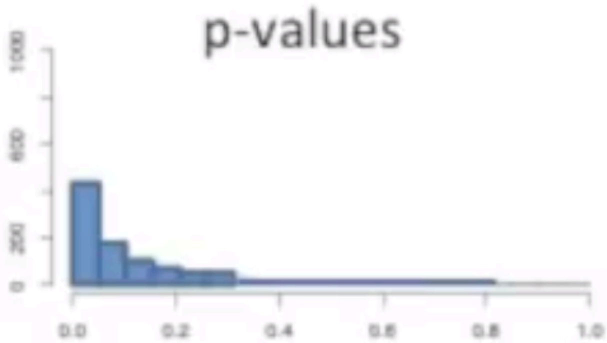
The drug might affect 1,000 genes…



The measurements for these genes will come from two different distributions.

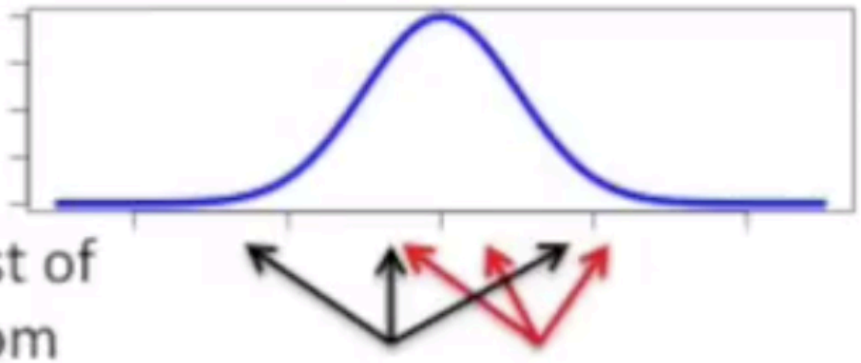
The **black** sample is from the control cells.

The **red** sample is from the cells treated with the drug.

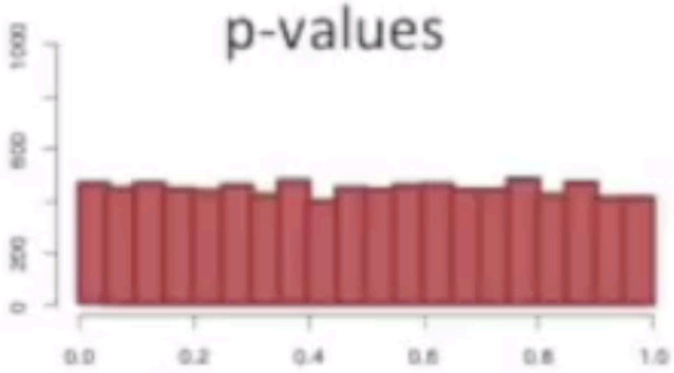


Since the samples come from different distributions, the p-values are skewed.

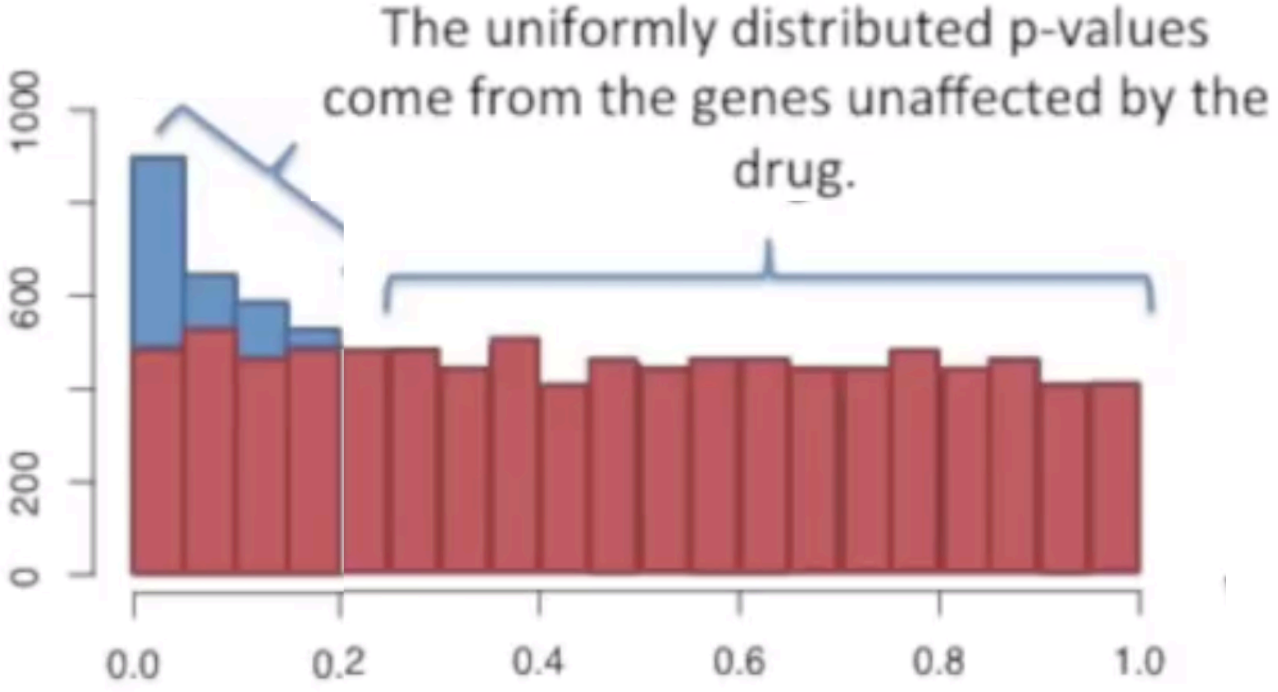
The remaining 9,000 active genes might not be affected by the drug…

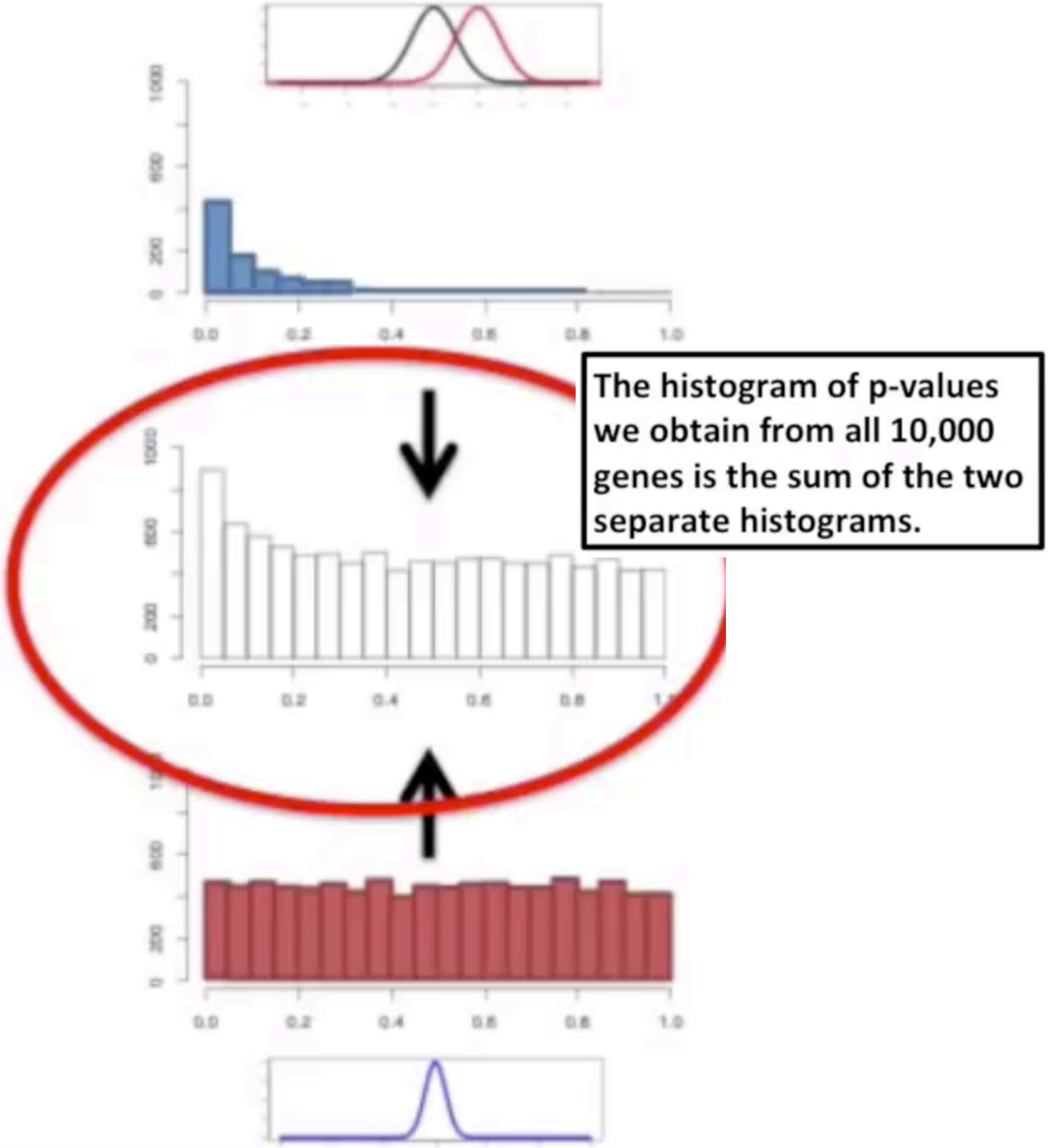


This means the measurements for most of the genes will come from the same distribution.

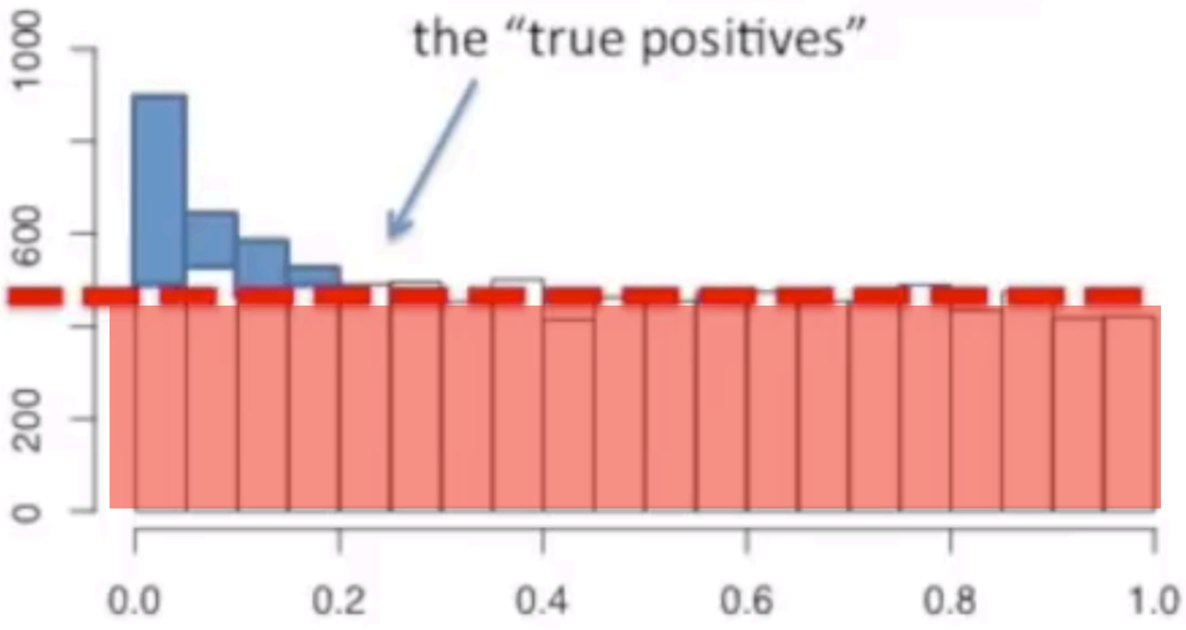


The p-values on the left side are a mixture from genes affected and genes unaffected by the drug.

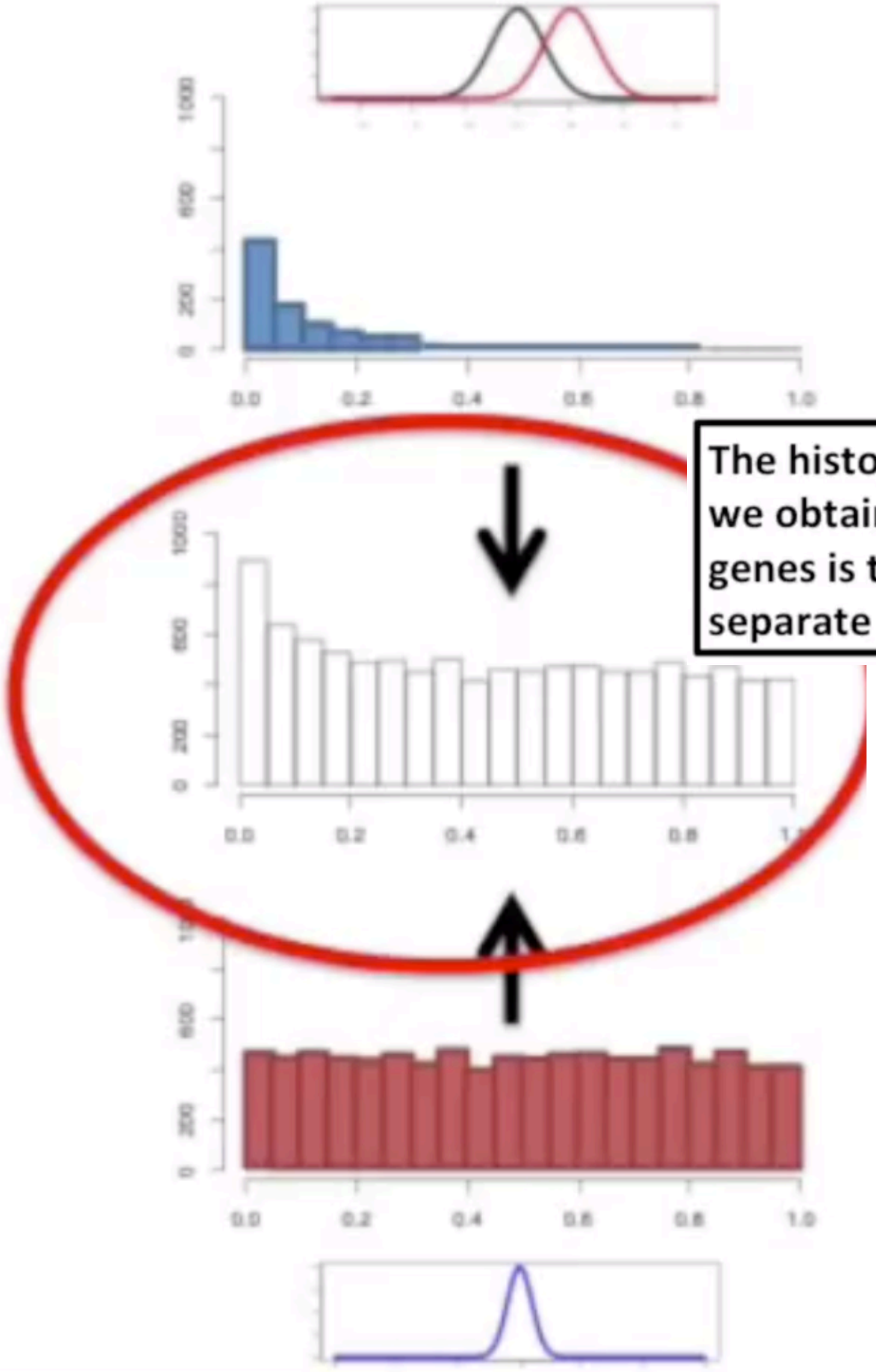

The uniformly distributed p-values come from the genes unaffected by the drug.


The histogram of p-values we obtain from all 10,000 genes is the sum of the two separate histograms.

We can extend this line and use it as a cutoff to identify the "true positives"

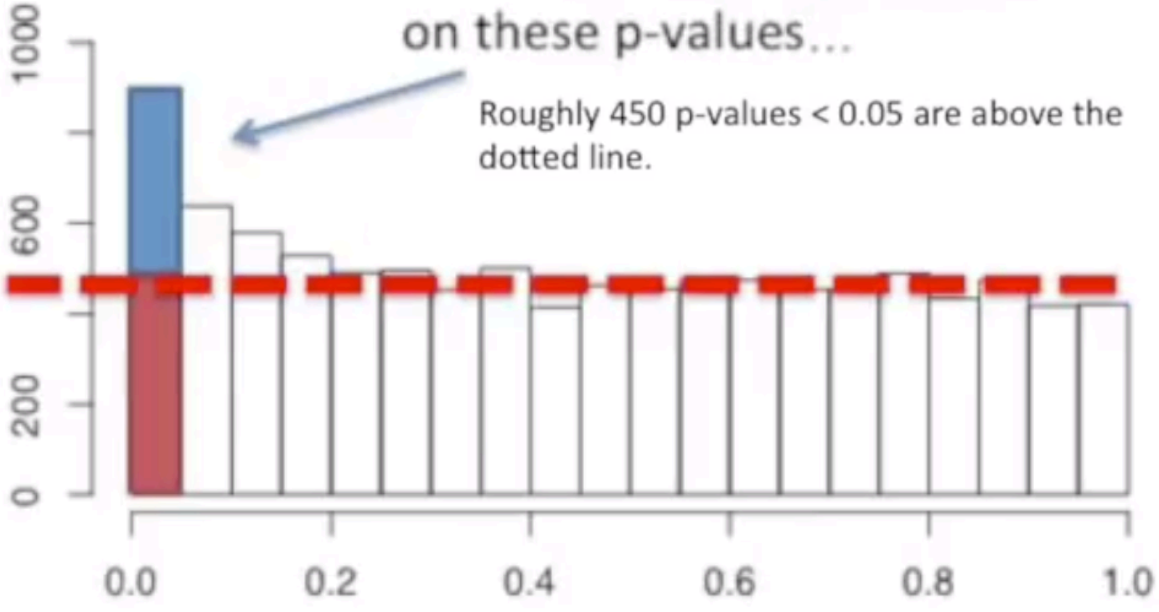


By eye, we can see where the p-values are uniformly distributed and determine how many tests are in each bin.



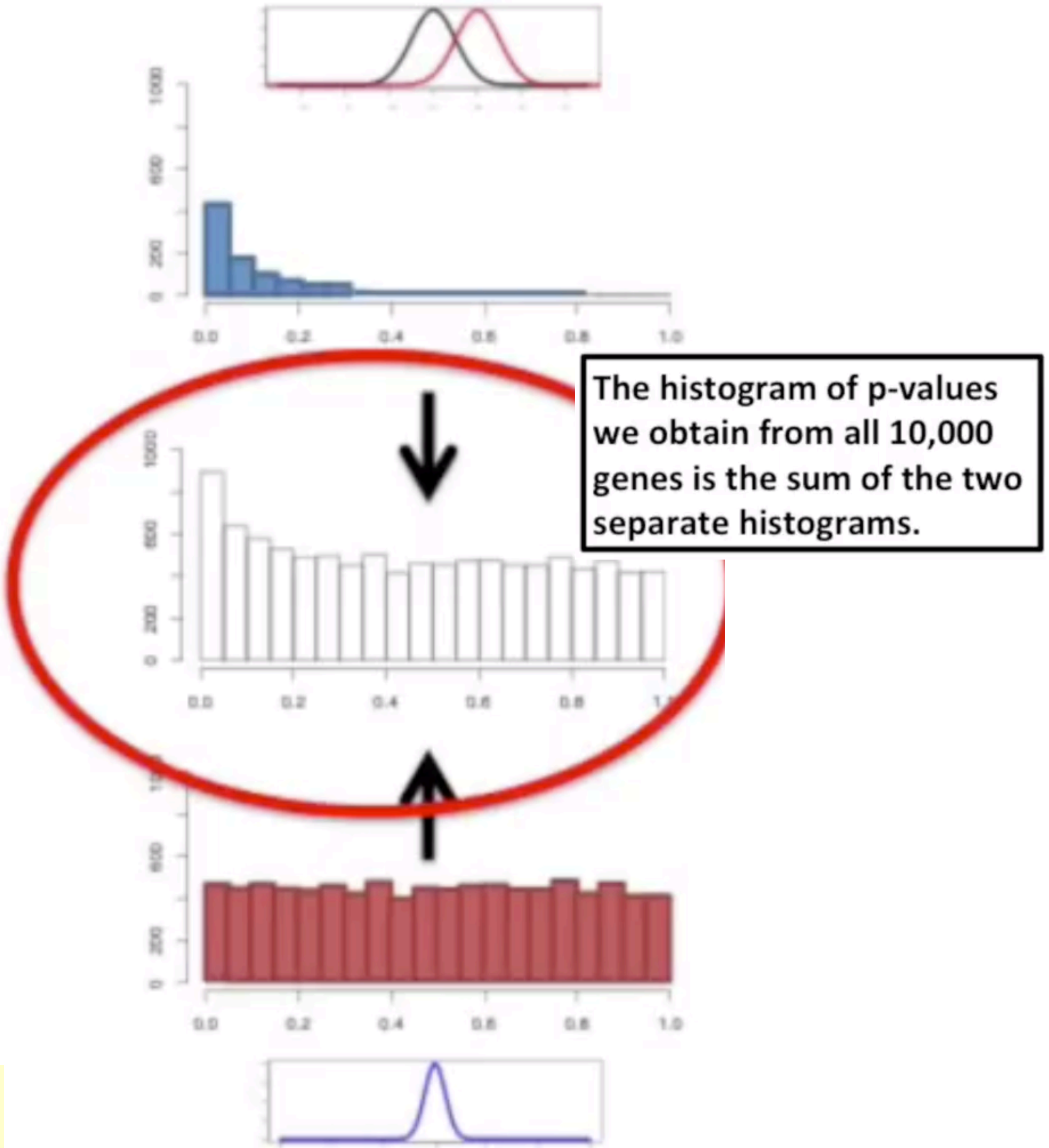
The histogram of p-values we obtain from all 10,000 genes is the sum of the two separate histograms.

Since we usually use a cutoff of 0.05, we're going to focus on these p-values…



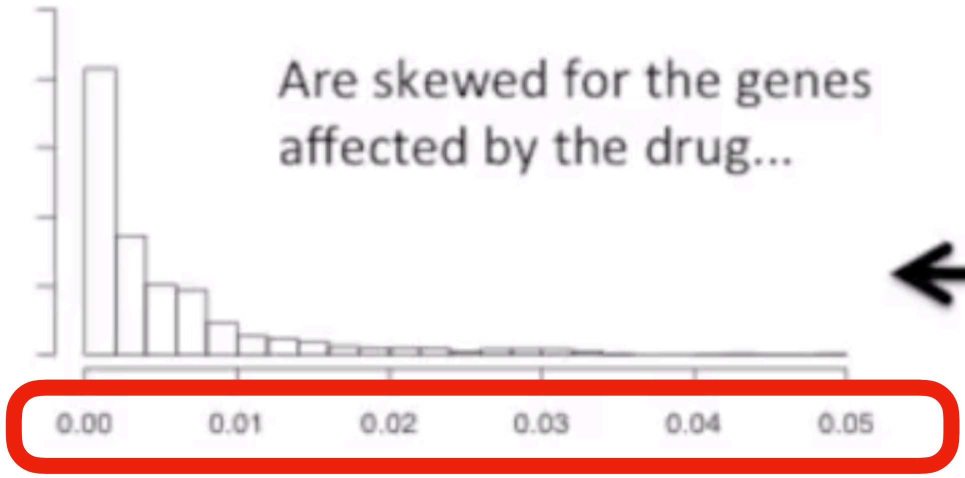
Roughly 450 p-values < 0.05 are above the dotted line.

Roughly 450 p-values < 0.05 are below the dotted line.

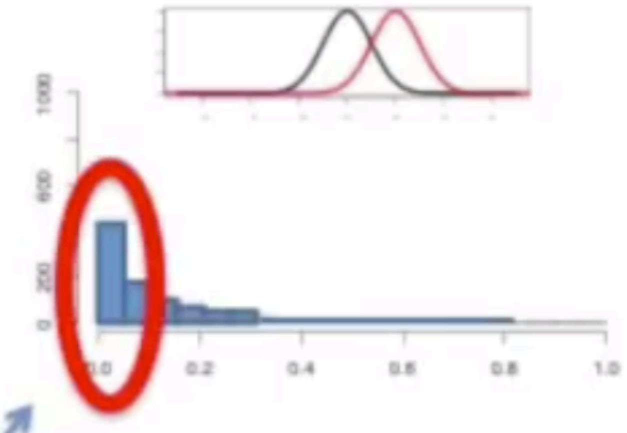


The histogram of p-values we obtain from all 10,000 genes is the sum of the two separate histograms.

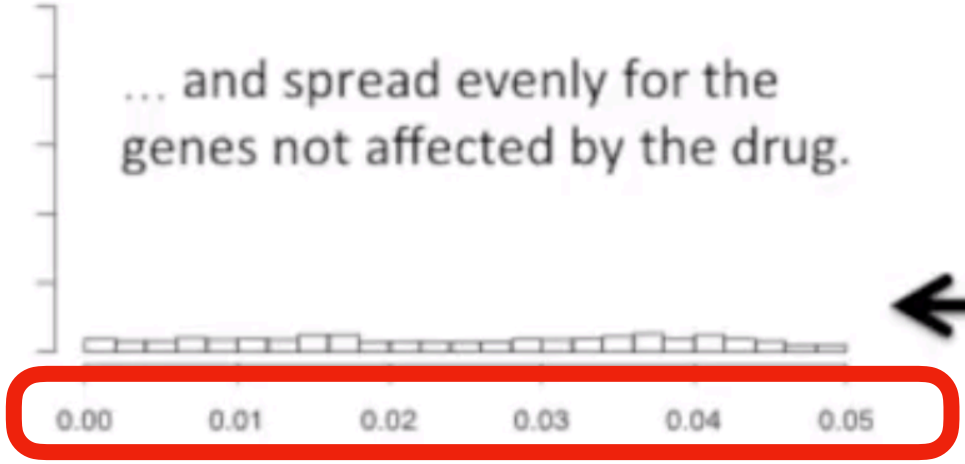
One way to isolate the true positives (genes affected by the drug) from the false positives would be to only consider the smallest 450 p-values.



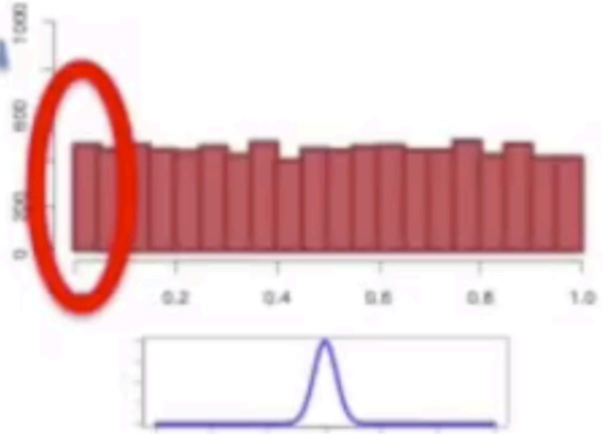
Are skewed for the genes affected by the drug...



This procedure works fairly well because the p-values within the bins...



... and spread evenly for the genes not affected by the drug.

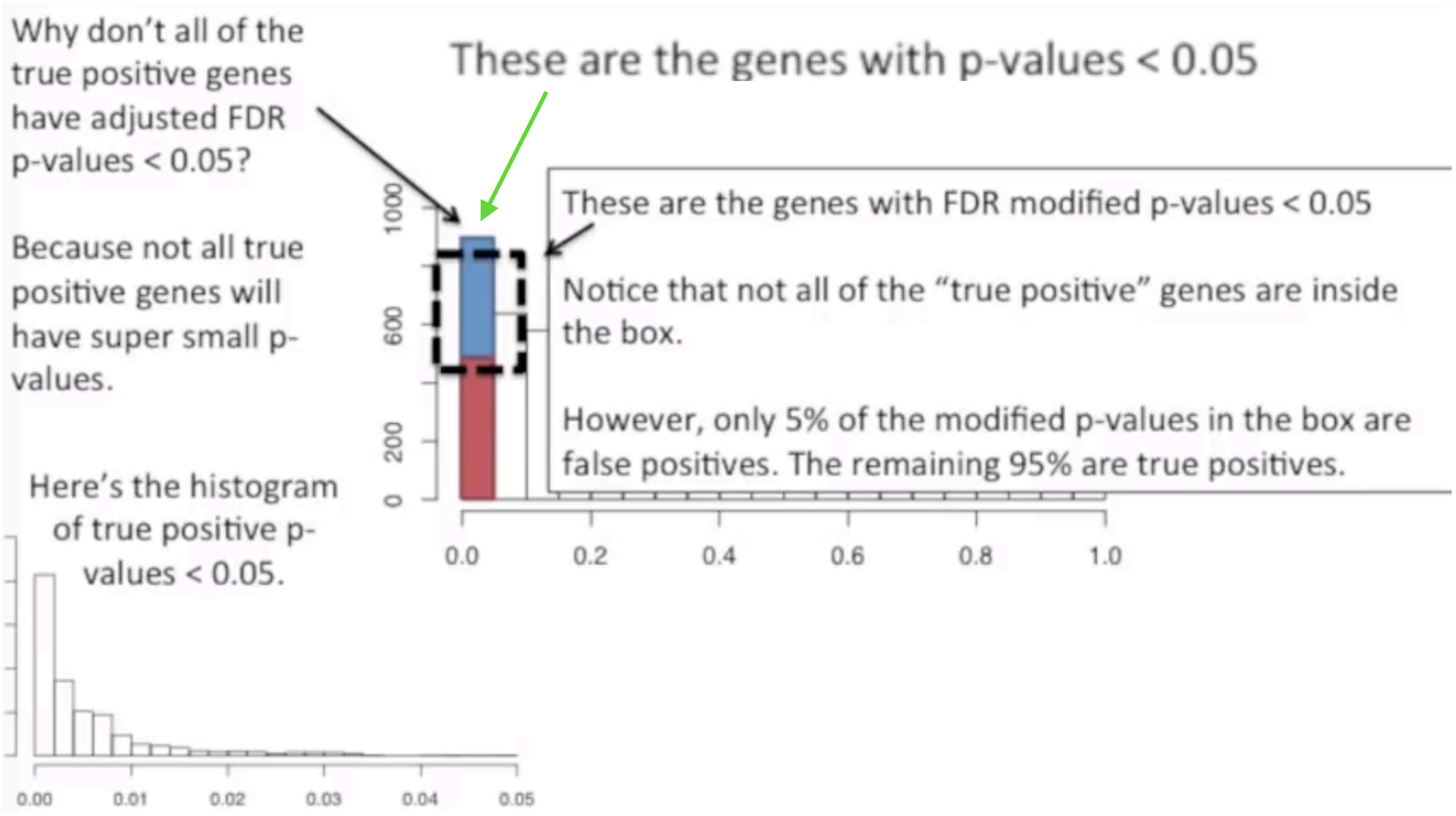


The Benjamini-Hochberg method

- Is based on the "eyeball" method we just saw.
  - We'll go over how it really works in just a bit.
- If your cutoff for significance is FDR < 0.05, then less than 5% of the "significant" results will be false positives.
- It **adjusts p-values** in a way that limits the number of false positives that are reported as "significant".

"adjusts p-values" means that it makes them larger.

For example, before the FDR correction, your p-value might be 0.04 (significant)
After the FDR correction, your p-value might be 0.06 (no longer significant)



Why don't all of the true positive genes have adjusted FDR p-values < 0.05?

Because not all true positive genes will have super small p-values.

These are the genes with p-values < 0.05

These are the genes with FDR modified p-values < 0.05

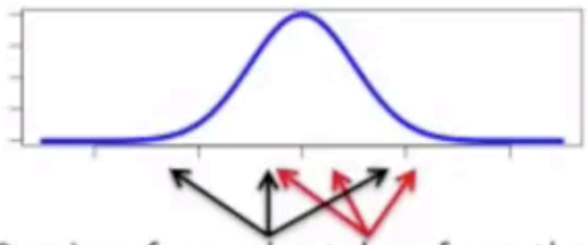
Notice that not all of the "true positive" genes are inside the box.

However, only 5% of the modified p-values in the box are false positives. The remaining 95% are true positives.

Here's the histogram of true positive p-values < 0.05.

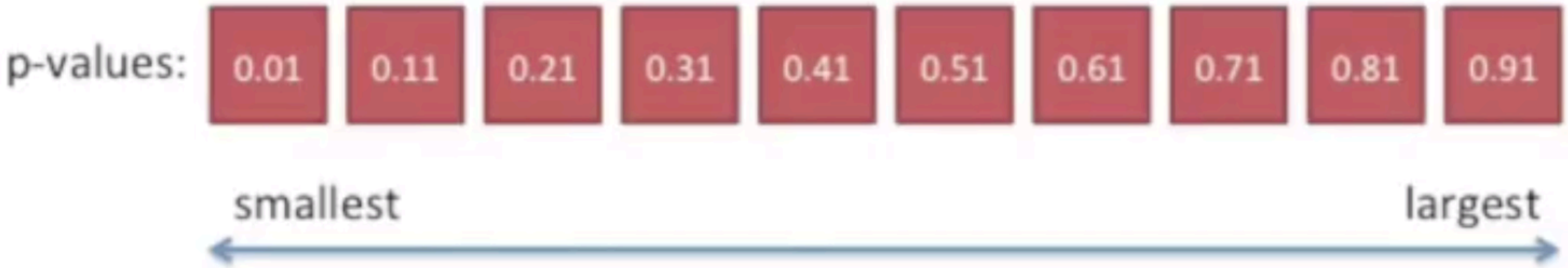
The Benjamini-Hochberg method

A simple example



10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.

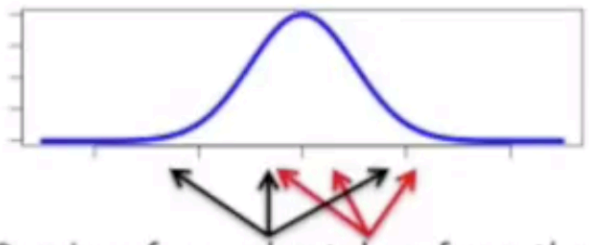


p-values: 0.01 0.11 0.21 0.31 0.41 0.51 0.61 0.71 0.81 0.91

smallest largest

The Benjamini-Hochberg method

A simple example



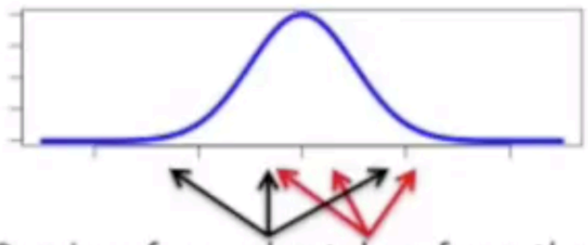
10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.
2. Rank the p-values

p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8	9	10

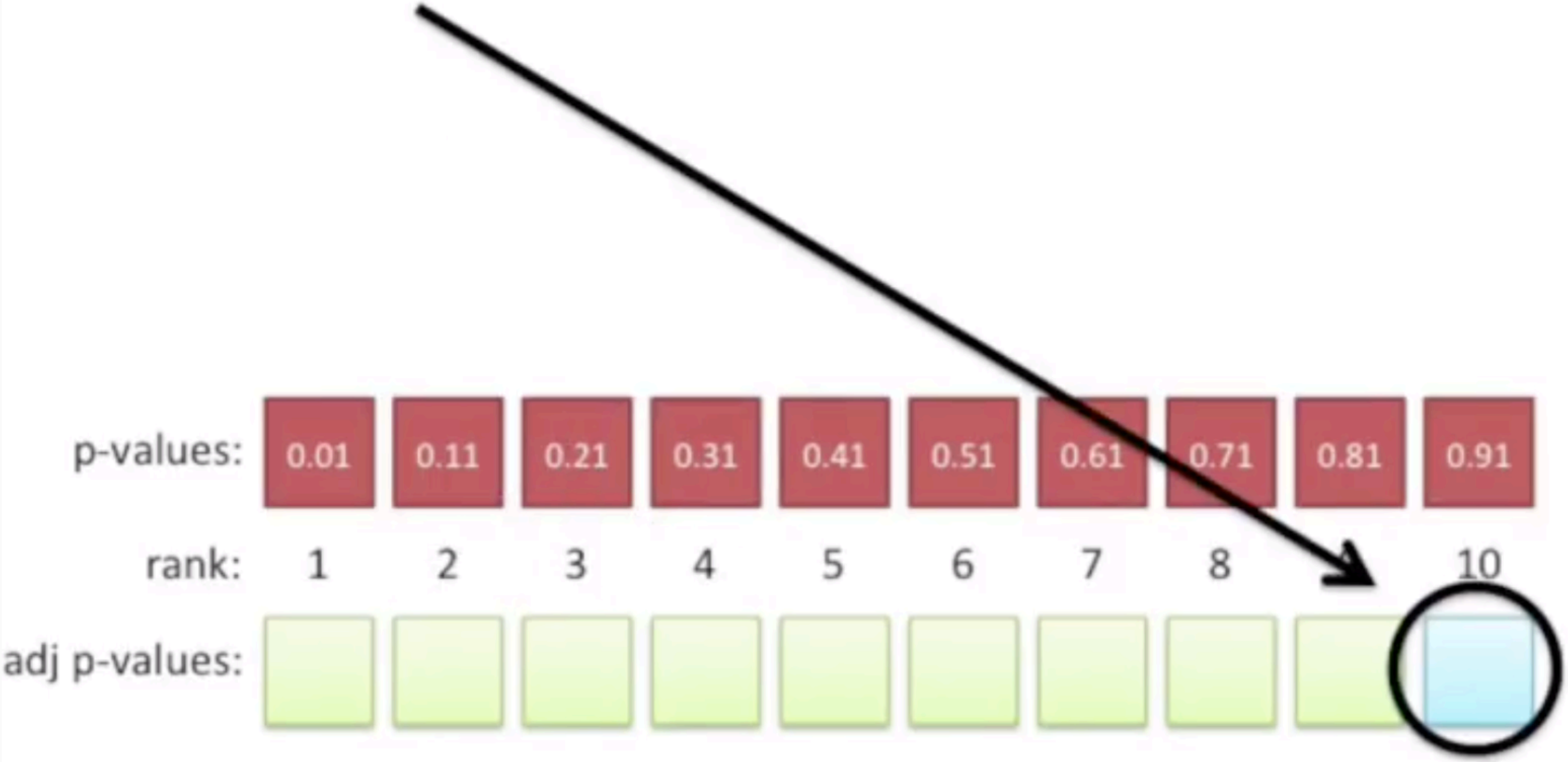
The Benjamini-Hochberg method

A simple example



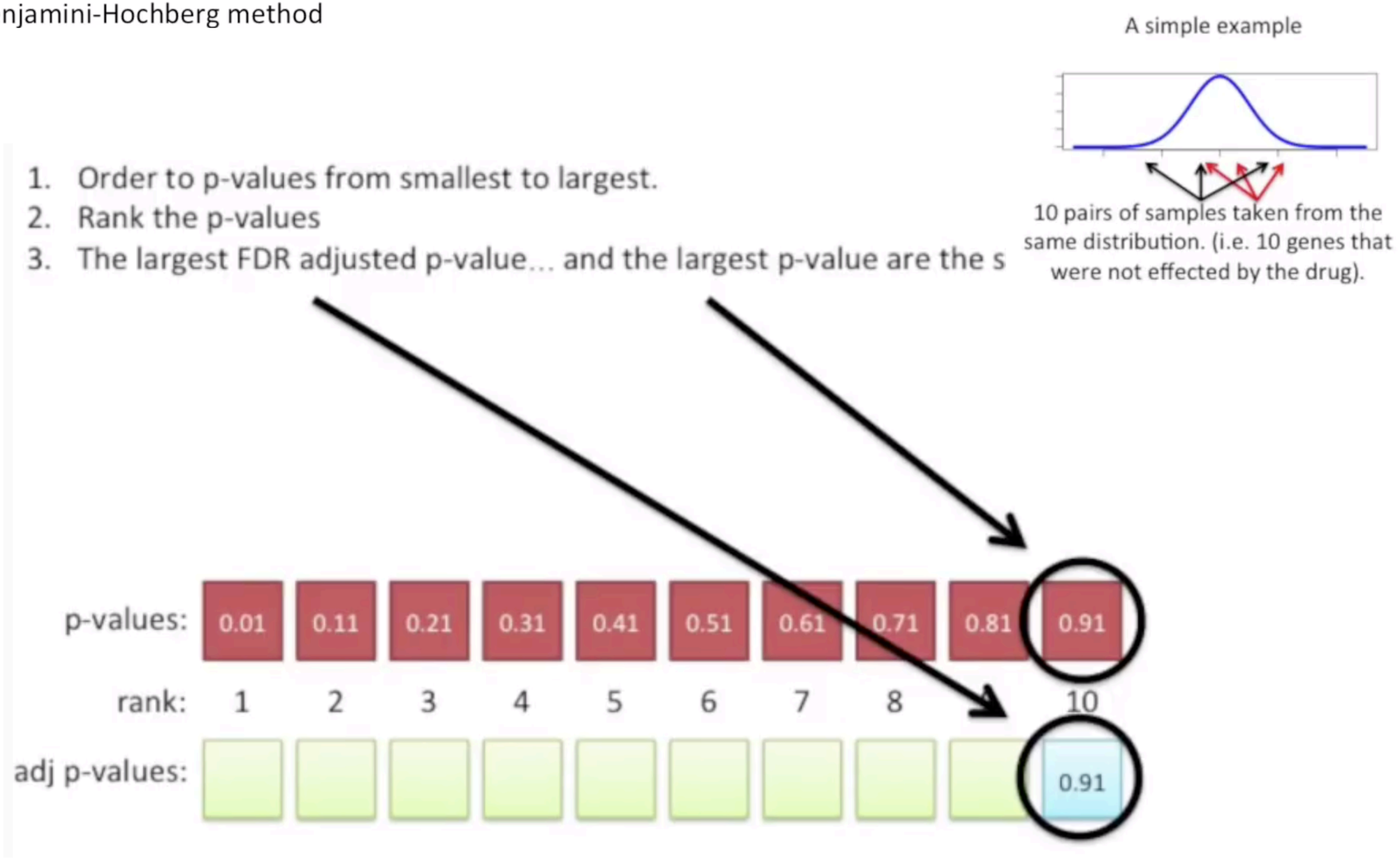
10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.
2. Rank the p-values
3. The largest FDR adjusted p-value…



p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8		10
adj p-values:										

The Benjamini-Hochberg method



A simple example

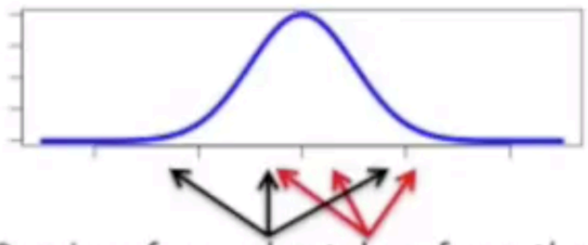
10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.
2. Rank the p-values
3. The largest FDR adjusted p-value… and the largest p-value are the s

p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8	9	10
adj p-values:										0.91

The Benjamini-Hochberg method

A simple example



10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.
2. Rank the p-values
3. The largest FDR adjusted p-value… and the largest p-value are the same
4. The next largest adjusted p-value…

…is the smaller of two options:

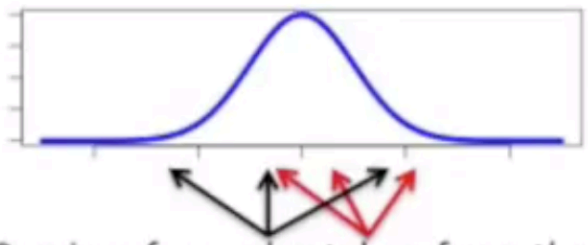
a: The previous adjusted p-value = 0.91

b: the current p-value $* \left(\frac{\text{total \# of p-values}}{\text{p-value rank}} \right)$

p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8	9	10
adj p-values:										0.91

The Benjamini-Hochberg method

A simple example



10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.
2. Rank the p-values
3. The largest FDR adjusted p-value… and the largest p-value are the same
4. The next largest adjusted p-value…

…is the smaller of two options:

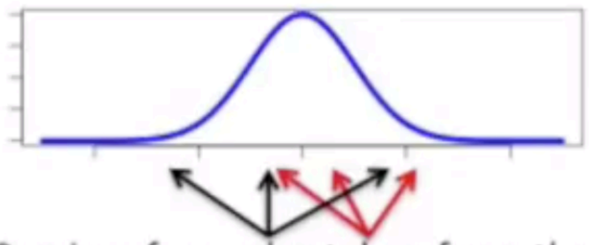
a: The previous adjusted p-value = 0.91

b: $0.81 * \left(\frac{10}{9} \right) = 0.90$

p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8	9	10
adj p-values:									0.90	0.91

The Benjamini-Hochberg method

A simple example



10 pairs of samples taken from the same distribution. (i.e. 10 genes that were not effected by the drug).

1. Order to p-values from smallest to largest.
2. Rank the p-values
3. The largest FDR adjusted p-value... and the largest p-value are the same
4. The next largest adjusted p-value...

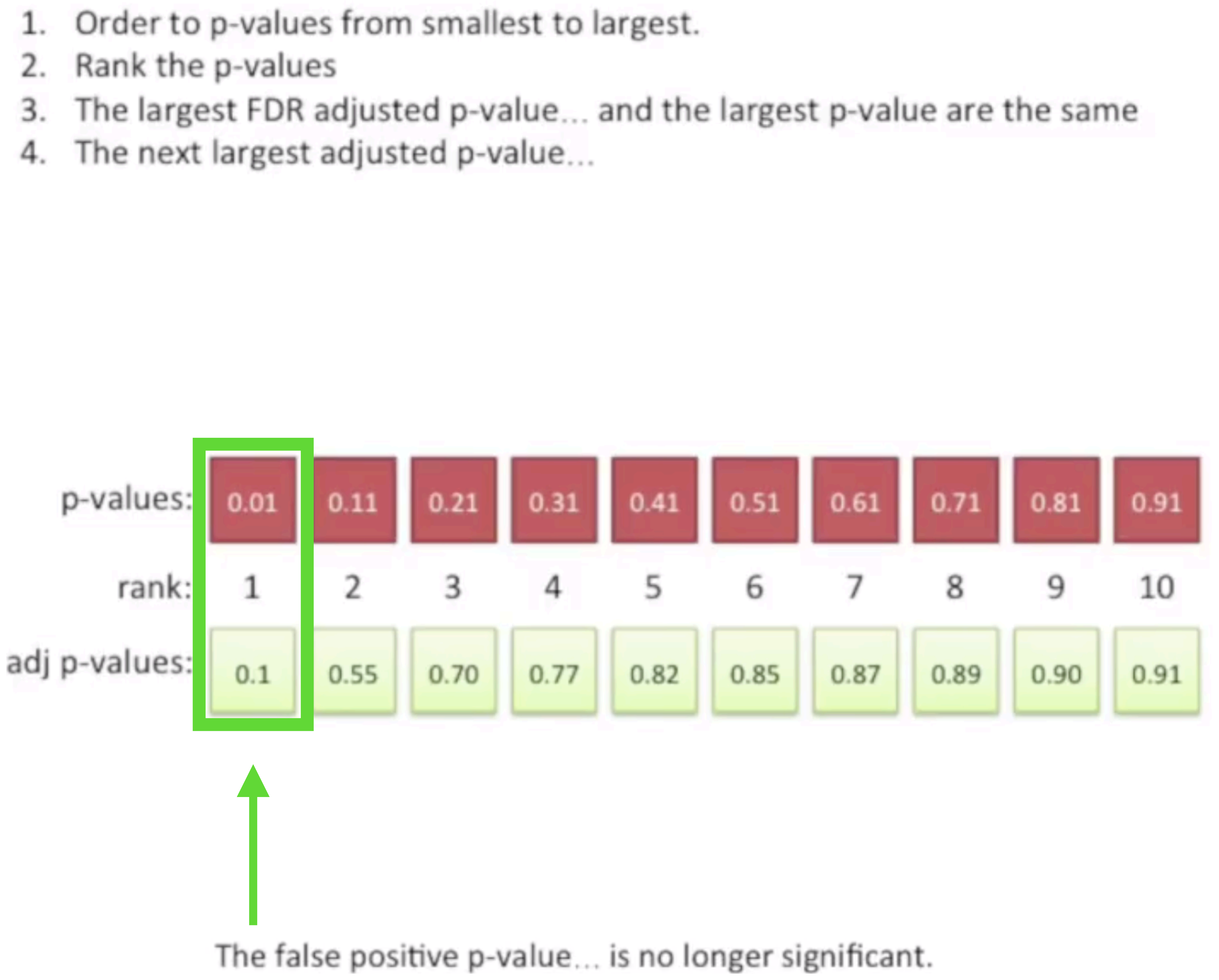
...is the smaller of these two options:

a: The previous adjusted p-value = 0.90

b: $0.71 * \left(\frac{10}{8} \right) = 0.89$

p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8	9	10
adj p-values:									0.90	0.91

The Benjamini-Hochberg method



1. Order to p-values from smallest to largest.
2. Rank the p-values
3. The largest FDR adjusted p-value… and the largest p-value are the same
4. The next largest adjusted p-value…

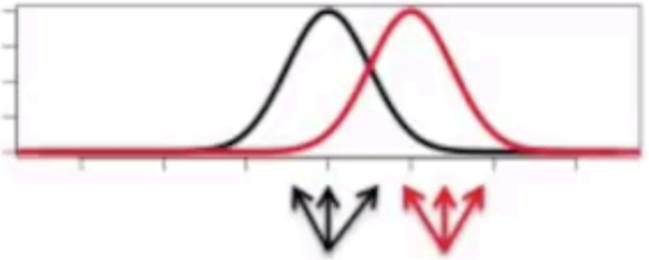
p-values:	0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
rank:	1	2	3	4	5	6	7	8	9	10
adj p-values:	0.1	0.55	0.70	0.77	0.82	0.85	0.87	0.89	0.90	0.91

The false positive p-value… is no longer significant.

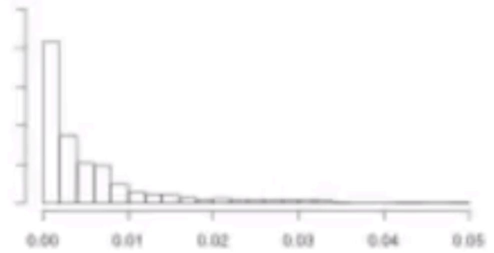
The Benjamini-Hochberg method

.001									
.002									
.003	0.05								
.004	0.06	0.15							
0.01	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
0.02	0.12	0.22	0.32	0.42	0.52	0.62	0.72	0.82	0.92
0.03	0.13	0.23	0.33	0.43	0.53	0.63	0.73	0.83	0.93
0.04	0.14	0.24	0.34	0.44	0.54	0.64	0.74	0.84	0.94

These are the p-values from when the samples came from two separate distributions.

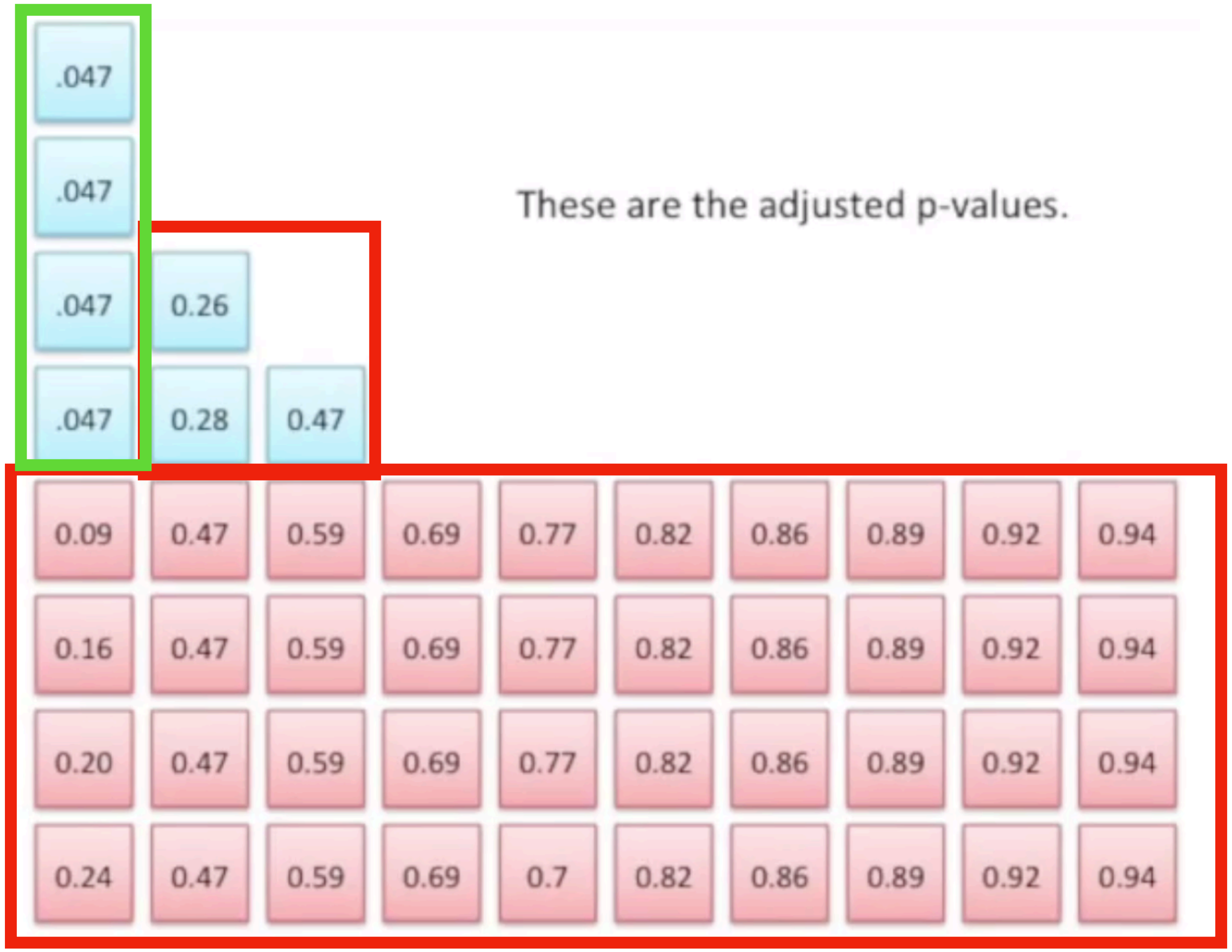


I've made these p-values smaller to reflect their normal skew.



NOTE! We've got some false positives!

The Benjamini-Hochberg method



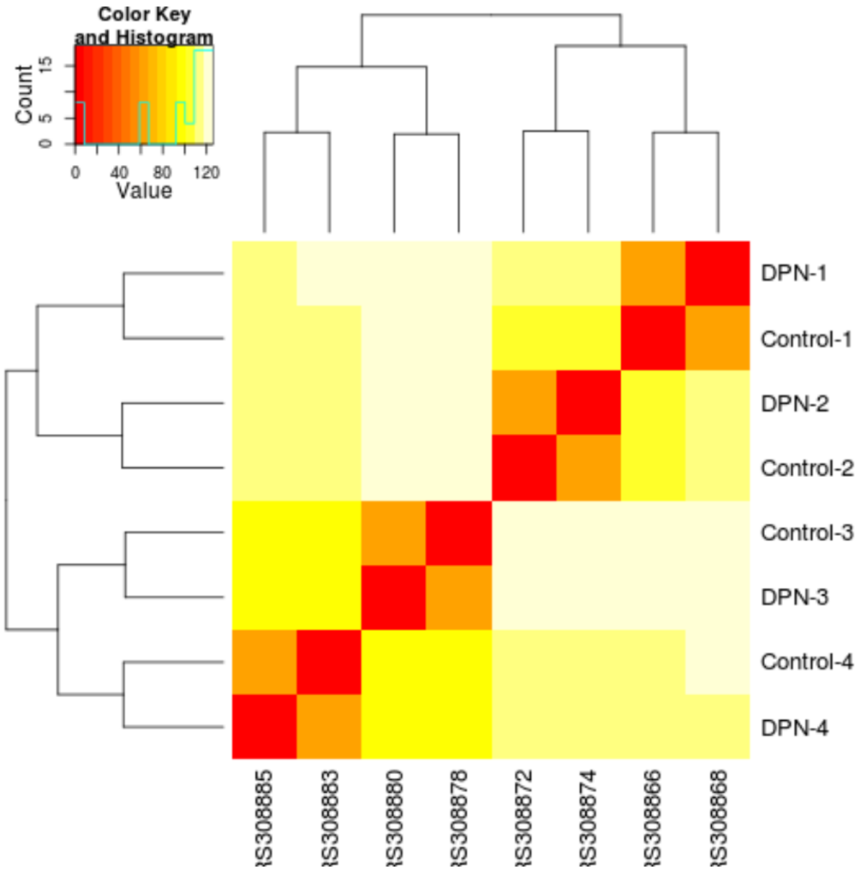
These are the adjusted p-values.

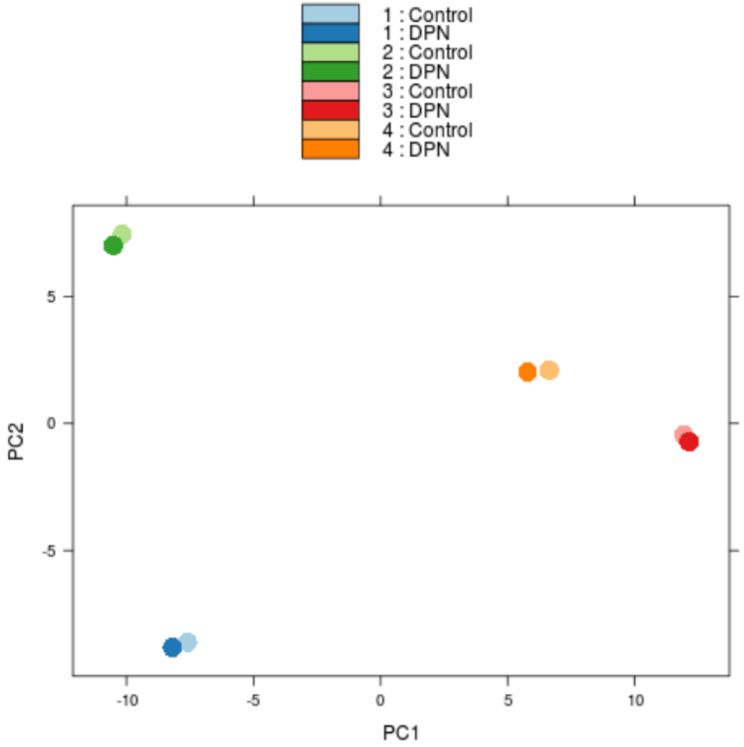
But these true positives remain < 0.05

The false positives are now > 0.05

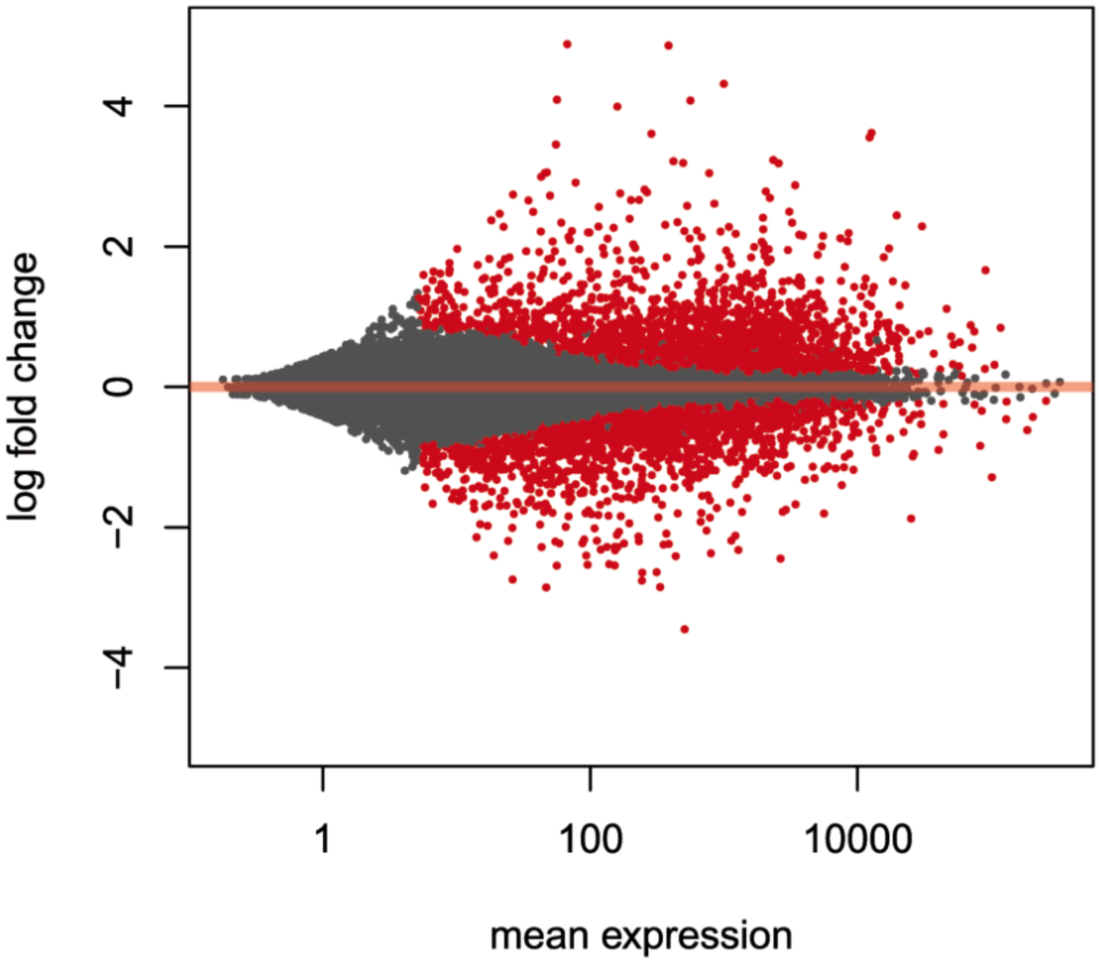
.047									
.047									
.047	0.26								
.047	0.28	0.47							
0.09	0.47	0.59	0.69	0.77	0.82	0.86	0.89	0.92	0.94
0.16	0.47	0.59	0.69	0.77	0.82	0.86	0.89	0.92	0.94
0.20	0.47	0.59	0.69	0.77	0.82	0.86	0.89	0.92	0.94
0.24	0.47	0.59	0.69	0.7	0.82	0.86	0.89	0.92	0.94

Differential Expressed Genes – Visualization





Differential Expressed Genes – Visualization

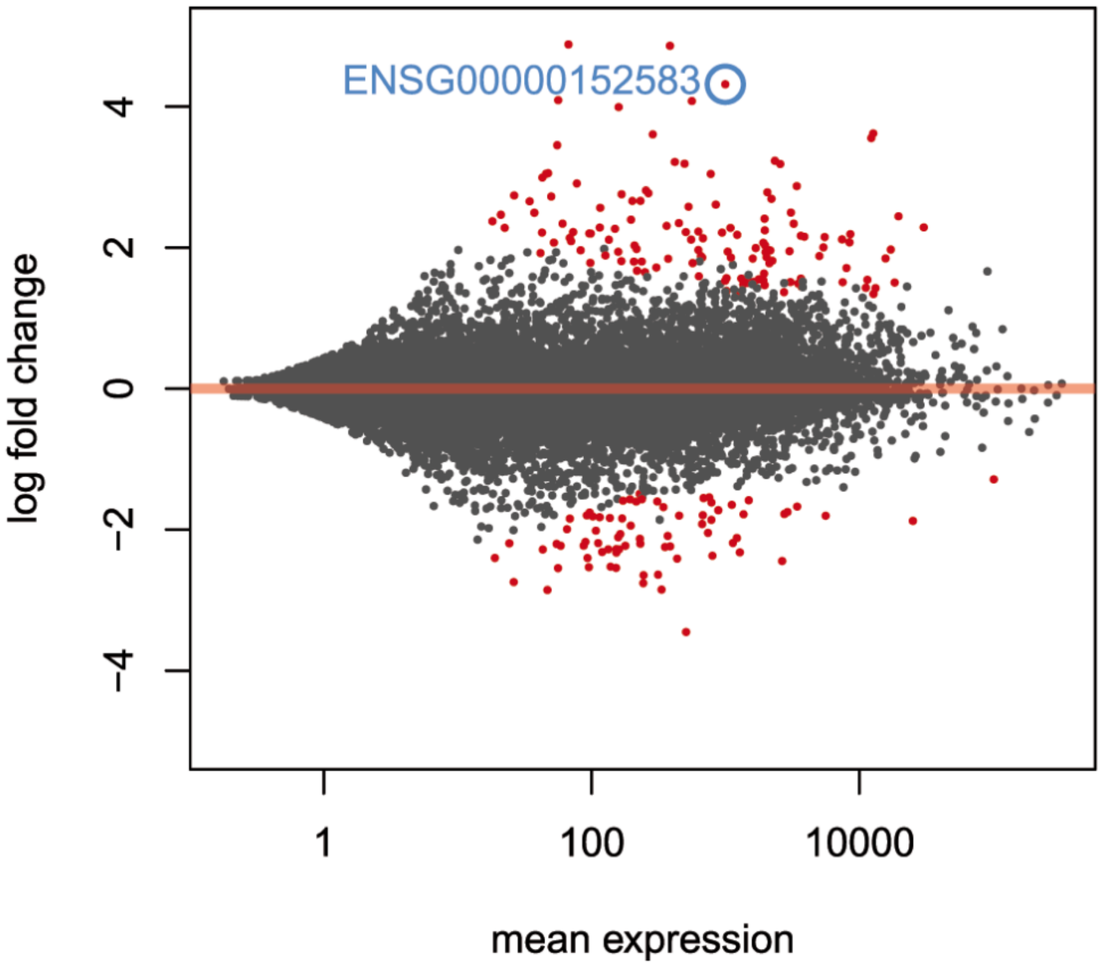


MA-plot of changes induced by treatment.

The log2 fold change for a particular comparison is plotted on the y-axis and the average of the counts normalized by size factor is shown on the x-axis (“M” for minus, because a log ratio is equal to log minus log, and “A” for average). Each gene is represented with a dot.

Genes with an adjusted *p* value below a threshold (here 0.1, the default) are shown in red.

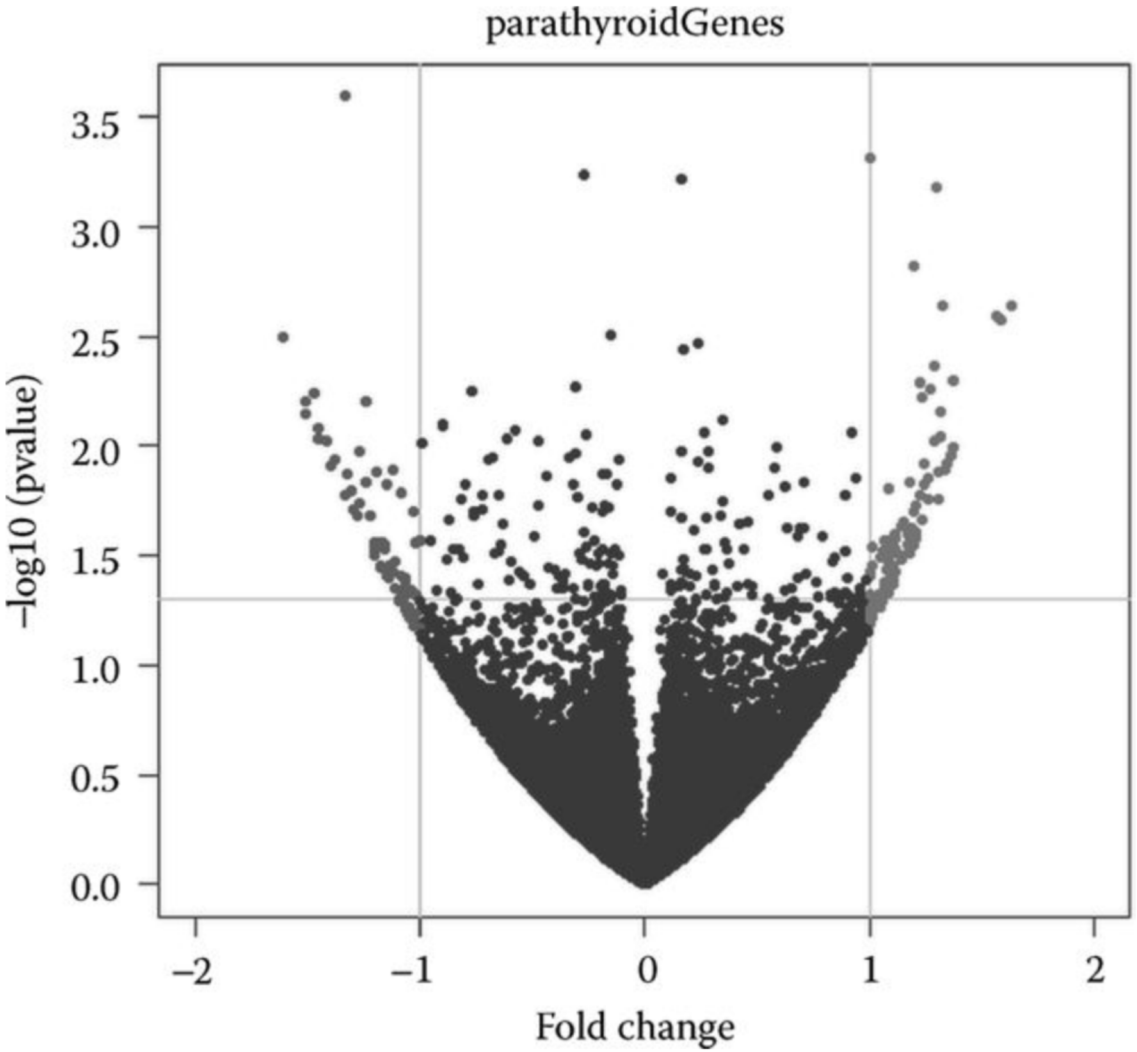
Differential Expressed Genes – Visualization



The red points indicate genes for which the log2 fold change was significantly higher than 1 or less than -1 (treatment resulting in more than doubling or less than halving of the normalized counts) with adjusted *p* value less than 0.1.

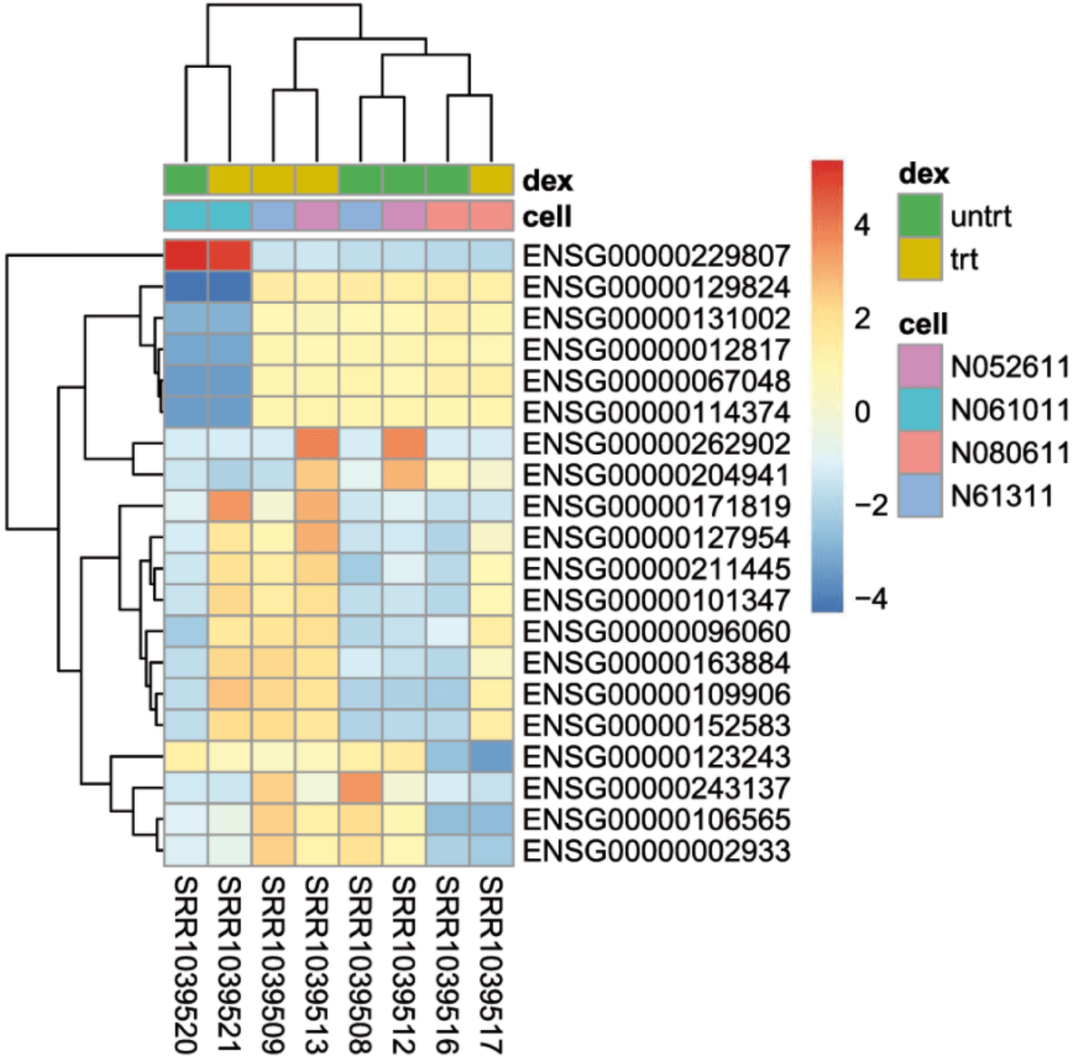
The point circled in blue indicates the gene with the lowest adjusted *p* value.

Differential Expressed Genes – Visualization



A **Volcano plot** is simply a scatterplot that has the fold change values for all features on the horizontal (*x*) axis, and the −log 10-transformed *p*-value on the vertical (*y*) axis.

Differential Expressed Genes – Visualization

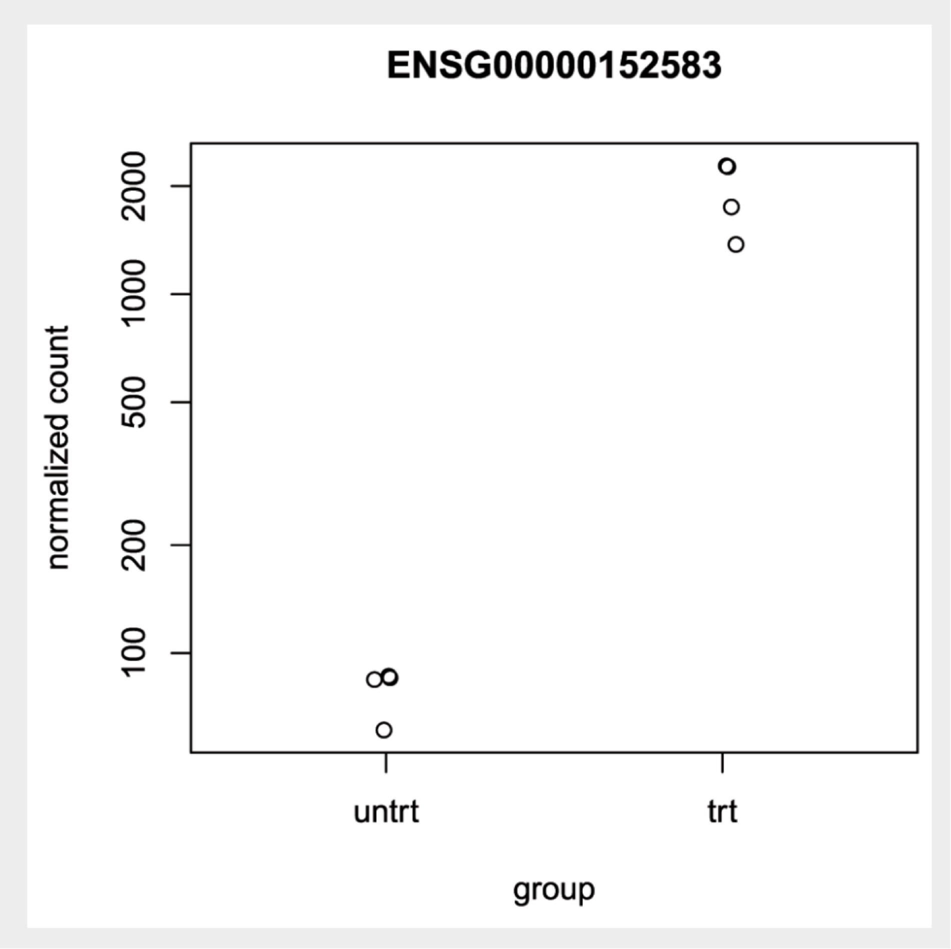


Heatmap of relative rlog-transformed values across samples.

Treatment status and cell line information are shown with colored bars at the top of the heatmap.

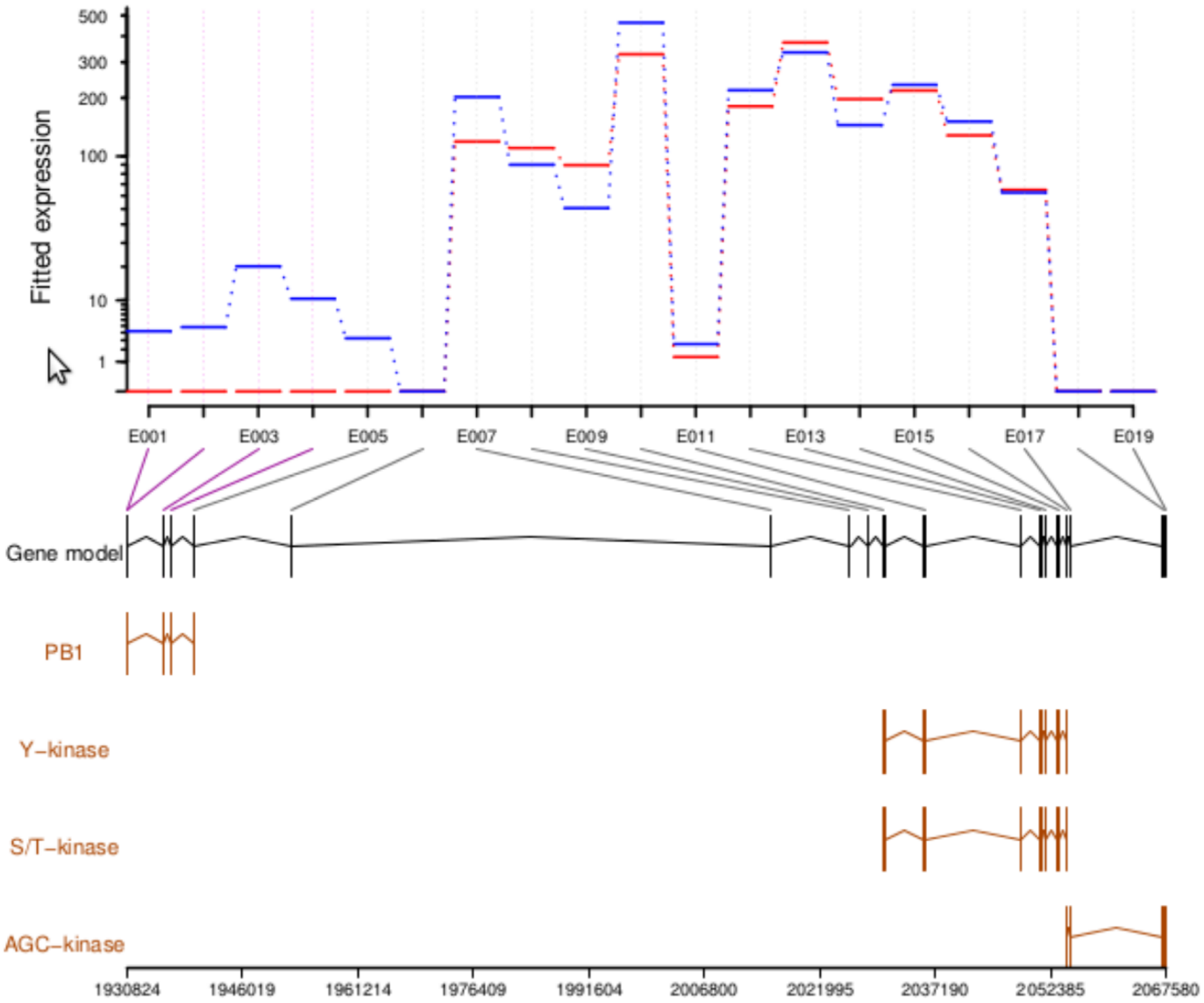
Note that a set of genes at the top of the heatmap are separating the N061011 cell line from the others. In the center of the heatmap, we see a set of genes for which the dexamethasone treated samples have higher gene expression.

Differential Expressed Genes – Visualization



Normalized counts for a single gene over treatment group.

Differential Expressed Genes – Visualization



Exon expression by DEXSeq