

## RNA Sequencing

If you are not familiar with the basics of gene expression and RNA-Seq technology, please watch the *RNA Sequencing* video before continuing.

Next, please read Do, K.-A. et al., *Advances in Statistical Bioinformatics*, 2013, chapter 4 (it starts on page 77): sections 4.1.1, 4.1.2, 4.2, 4.2.3, 4.3.1 until equation 4.4 (including that equation), 4.4 and 4.4.3. Return to this lecture when you have finished reading.

**Question 1: (not for forum discussion) In the context of an mRNA sequencing experiment, the term differential gene expression refers to the situation when**

- different genes produce different number of copies of mRNA molecules
- different mRNAs produce different number of protein molecules
- different mRNAs produce different number of reads
- the same gene produces different number of copies of mRNA molecules in different samples (or groups of samples)
- the same mRNA molecule produces different number of protein molecules in different samples (or groups of samples)
- the same mRNA molecule produces different number of reads in different samples (or groups of samples)

**Question 2: (not for forum discussion) One of the goals of mRNA-Seq data analysis is to**

- detect it when identical mRNA molecules produce different numbers of reads
- get around the problem that identical mRNA molecules produce different numbers of reads

**Question 3: In the absence of alternative splicing, heterozygosity and paralogous genes unobserved read counts  $r_i$  mentioned in equation 4.4 of the reading becomes**

- 0
- 1
- $-\infty$
- $+\infty$
- observed/observable
- uninformative

## The Use of the Poisson Distribution for RNA-Seq Data Analysis

In the absence of *biological variation*, read counts for a specific transcript obtained in an RNA-Seq experiment follow a Poisson distribution.

In fact, the number of events that occur in a fixed interval of time follows the Poisson distribution if these events are independent and have the same probability of a “success”. An RNA-Seq experiment is conducted over a fixed duration of time and it counts numbers of reads that are sequenced during this time.

Each location of each copy of each transcript has the same probability of being sequenced. All reads are sequenced independently from each other and the probability of a “success” (a read originating from the transcript) only depends on the length of the transcript and its abundance in the mixture.

Therefore, the probability of exactly  $k$  reads originating from a specific transcript is

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

where  $\lambda$  is a transcript specific parameter. It is the average number of reads that originate from the

transcript during the sequencing experiment.

Each read starts at a random location of a random copy of a random transcript in the mixture. The more copies of a transcript are present in the mixture and the longer the transcript, the higher the probability that a read will originate from the transcript.

$$\lambda \sim \mu l$$

or

$$\lambda = b\mu l$$

where  $\mu$  is the level of gene expression,  $l$  is the length of the transcript minus the average length of a read plus one and  $b$  is a constant that is the same for all transcripts in the experiment.

The equations above apply to a transcript, not to a gene. If we ignore the possibility of alternative splicing, one transcript will correspond to one gene. So, if, for a specific gene, we have read counts from several repetitions of the same sequencing experiment then we have a set of numbers generated from the same Poisson distribution. The mean and variance of this distribution are  $\lambda$  and, knowing the value of  $\lambda$ , the length of the gene and constant  $b$ , we can find  $\mu$ , which is the level of gene expression.

**Question 4: (not for forum discussion) What is the approximate value (within  $\pm 20$  from the actual value) of the parameter  $\lambda$  of a Poisson distribution if it has generated numbers 257 245 263 244 265 251 243 220 222 260 (please round to the nearest whole number).**

If alternative splicing is possible, this simple method can no longer be used. With alternative splicing, a read cannot be mapped to a specific isoform. It is possible to map a read to an exon, but this exon may belong to several alternatively spliced transcripts. Therefore, reads that map to a gene come from a mixture of gene isoforms and it's not known how the read count is split between the isoforms. The number of reads that originate from a mixture of isoforms doesn't necessarily follow the Poisson distribution and the equation above cannot be applied.

One approach to solve this problem is to try to figure out how each count is split between the isoforms and we will discuss this approach in module 13. Another way to account for alternative splicing is to fit read counts with a distribution that is similar to the Poisson distribution, but is more spread out.

The negative binomial distribution is widely used to fit RNA-Seq data because it is an overdispersed Poisson distribution. The increased variance accounts for alternative splicing and other types of biological variation regardless of the exact nature of this variation.

## The Use of the Negative Binomial Distribution for RNA-Seq Data Analysis

The *negative binomial distribution* is a distribution of the number of successes in a sequence of Bernoulli trials before a specified number of failures (denoted  $r$ ) occur.

Do you remember the problem about a gambler from module 1? A coin is tossed until a tail appears. The casino pays  $\$2^N$  if  $N$  is 0, 1, 2, 3 or 4 and  $\$17$  otherwise where  $N$  is the number of heads. The number of heads follows the negative binomial distribution.

**Question 5: The probability of a success in each Bernoulli trial in the gambler problem is (please enter as a decimal number)**

**Question 6: The parameter  $r$  of the negative binomial distribution in the gambler problem is**

If the casino used a negative binomial distribution with  $r=2$ , the game would continue until two tails

appear (not necessarily consecutively).

The probability mass function of the negative binomial distribution is

$$P(X = k) = \binom{k+r-1}{k} p^k (1-p)^r$$

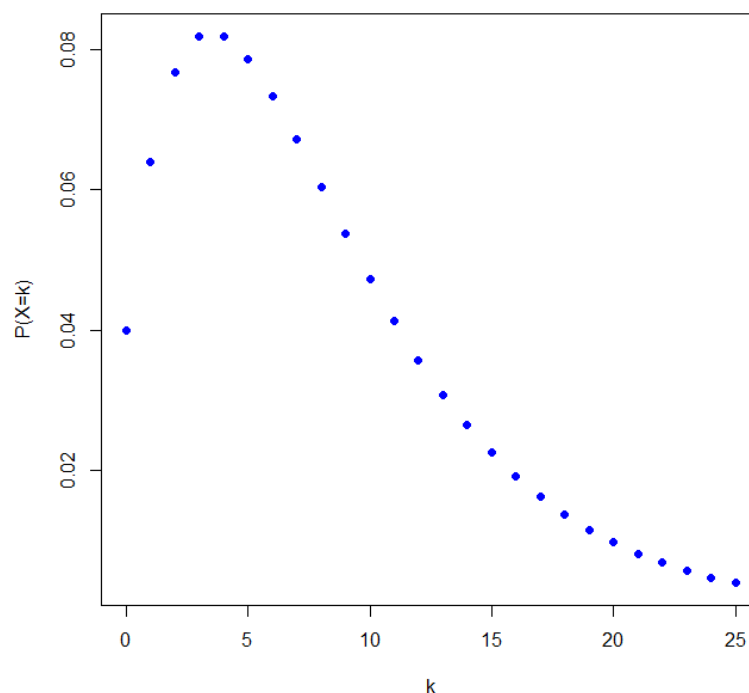
where  $p$  is the probability of a success in each Bernoulli trial.

**Question 7: (not for forum discussion, 2 points)** Is the gambler expected to win more or less money if the payoff is based on the number of heads before the second tail appears, but the coin was unfair with the probability of head 0.35? Compare two scenarios, the first scenario is the same as in module 1 (a fair coin, game ends after the first tail, the payoff equal to  $\$2^N$  for  $N=0, 1, 2, 3$  or 4 and  $\$17$  for  $N>4$  where  $N$  is the number of heads), the second scenario includes an unfair coin with the probability of head being 0.35 and the payoff equal to  $\$2^N$  for  $N=0, 1, 2, 3$  or 4 and  $\$17$  for  $N>4$  where  $N$  is the number of heads before the second tail.

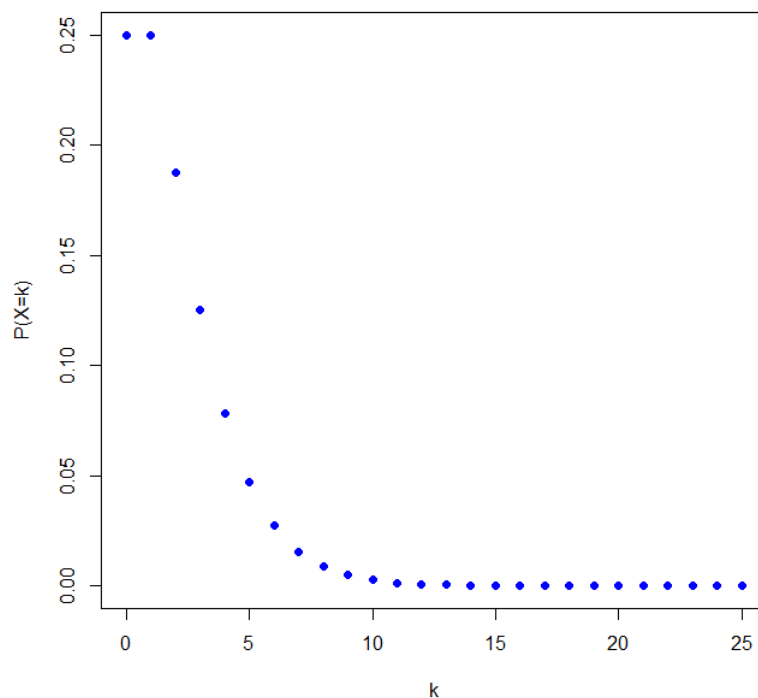
- the expected payoff is bigger in the second scenario
- the expected payoff is smaller in the second scenario
- the expected payoff is the same in both scenarios

Let's plot the probability mass function of the distribution with the probability of a success  $p=0.8$  and  $r=2$ :

```
x<-seq(from=0, to=25)
# size is the number of failures (size=r) and the parameter p in dnbinom() is
# the probability of a failure, which is 1-p in the probability mass function above
# with the negative binomial distribution, different authors
# denote the parameters differently, but the distribution itself is the same
y <-dnbinom(x, size=2, p=1-0.8)
plot(x, y, col="blue", xlab="k", ylab="P(X=k)", pch=16)
```



The shape of the distribution depends on both  $p$  and  $r$ .



**Question 8: The probability mass function above defines a negative binomial distribution with  $r=2$  and  $p$  (the probability of a success) approximately ( $\pm 0.1$ ) equal to**

It can be shown that the mean of the negative binomial distribution is

$$\mu = E(X) = \frac{pr}{1-p}$$

where  $p$  is the probability of a success and the variance is

$$Var(X) = \frac{pr}{(1-p)^2}$$

Combining the two equations above, the variance can also be written as

$$Var(X) = \frac{pr}{(1-p)^2} = \frac{pr}{1-p} + \frac{1}{r} \left( \frac{pr}{1-p} \right)^2 = \mu + \frac{1}{r} \mu^2$$

The distribution is defined by  $r$  and  $p$ . Alternatively, it is defined by  $r$  or  $\phi = \frac{1}{r}$ , which is called dispersion and the mean.

**Question 9: If  $\mu=10$  and the dispersion  $\phi=0.2$  then what is the value of  $r$ ? (please round to three decimal places, if applicable)**

**Question 10: If  $\mu=10$  and the dispersion  $\phi=0.2$  then what is the value of  $p$  (the probability of a success)? (please round to three decimal places, if applicable)**

The `dnbinom()` function in R can use  $r$  and  $1-p$  as parameters or it can accept  $r$  and  $\mu$ .

`dnbinom(x, size=5, mu=10)`

is equivalent to

`dnbinom(x, size=5, p=?)`

**Question 11: Please replace the question mark with the correct number (rounded to three**

**decimal places, if applicable)**

**A hint: don't forget that `dnbinom()` defines  $p$  differently than the probability mass function above and the equations derived from it.**

Why do we need two sets of parameters for the negative binomial distribution? The difference between the binomial and Poisson distribution on one hand and the negative binomial distribution on the other hand is that the former are used to describe the processes that are binomial or Poisson in nature and the latter is just a distribution with a convenient shape.

When the Poisson distribution is used to describe gene expression quantification, it is used because the sequencing process resembles a Poisson process. Each stretch of RNA (or the corresponding cDNA) is sequenced with some frequency which depends on the abundance of that RNA in the sample and doesn't depend on whether or not it has been sequenced before. The sequencing experiment has a fixed duration. Therefore, the count of the sequencing events which actually happen follows the Poisson distribution. The real process differs a bit from this idealistic picture because of alternative splicing and other biological variation, but there is still clear logic behind using the Poisson distribution.

The negative binomial distribution is used to approximate read counts because it is similar to the Poisson distribution (the Poisson distribution is a special case of the negative binomial distribution), but it has an additional parameter which controls its variance. Hence, this distribution is selected for its shape, not for the logic behind it. This use of the negative binomial distribution is not limited by this specific application. While many processes resemble binomial and Poisson processes, it is hard to find a real life process where successes until a fixed number of failures are counted.

Since it is the shape of the binomial distribution and its similarity to the Poisson distribution that we are after, the parameter  $p$  is no longer meaningful.

The dispersion  $\phi$  of a negative binomial distribution defines the difference between the mean and the variance of that distribution. If  $\phi=0$  then the mean and the variance are the same, just like in a Poisson distribution. In fact, a negative binomial distribution with zero dispersion is a Poisson distribution. Let's plot a Poisson distribution with mean = 250.

```
x<-seq(from=100, to=400, length=101)
y<-dpois(x, 250)
plot(x, y, col="blue", xlab="k", ylab="P(X=k)", pch=15)
```

**Question 12: What is the variance of this distribution? (please round to one decimal place, if applicable)**

Next, we will plot a negative binomial distribution with  $\phi=0$  (zero dispersion corresponds to an infinitely large value of  $r$ ):

```
y <- dnbinom(x, size=10e10, mu=250)
points(x, y, col="red", pch=16)
```

**Question 13: Do the probability mass functions of these two distributions look the same?**

Now, let's add a negative binomial distribution with  $\phi=0.005$

```
y <- dnbinom(x, size=1/0.005, mu=250)
points(x, y, col="green", pch=16)
```

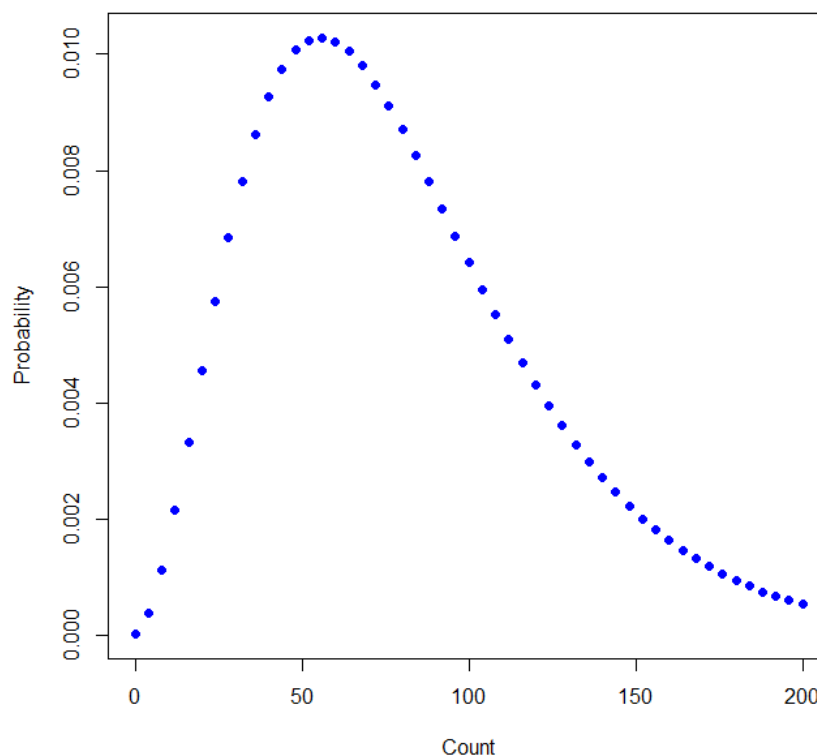
**Question 14: What is the variance of this distribution? (please round to one decimal place, if applicable)**

Assume that we know that read counts for a particular gene follow the negative binomial distribution with mean  $\mu$  and dispersion  $\phi$  and we want to know whether or not the count from a particular sample may have been generated by this distribution. We may want to know this if the sample comes from one group, for example, disease, and the distribution is defined by data that was obtained from another group, for example, control group. The null hypothesis is that the count was generated by the distribution.

Consider an example,  $\mu=80$ ,  $\phi=0.3$  and the count is 150. The count is higher than the mean of the distribution. However, there is always some random variation in actual counts even if the expression level is the same. To get an idea about how unusual this count is, we need to compute the p-value.

Let's start with plotting the probability mass function of the distribution:

```
# plot only points with x = 0, 4, 8, ..., 200 to avoid too many points on the graph
x <- seq(from=0, to=200, length = 51)
# the corresponding probabilities
y <- dnbinom(x, size=1/0.3, mu=80)
plot(x, y, col="blue", xlab="Count", ylab="Probability", pch=16)
```



**Question 15: The probability that the process with this probability mass function generates a count 150 or higher is equal to**

- the probability that the process generates counts from 0 to 149
- the probability that the process generates counts from 1 to 149
- one minus the probability that the process generates counts from 0 to 149
- one minus the probability that the process generates counts from 1 to 149

**Question 16: The probability that the process generates a count 150 or higher can be calculated using this R code (please select from the list):**

```
- x<-seq(from=0, to=149)
y<-dnbinom(x, size=1/0.3, mu=80)
pval1tail<-sum(y)
- x<-seq(from=1, to=149)
y<-dnbinom(x, size=1/0.3, mu=80)
pval1tail<-sum(y)
- x<-seq(from=0, to=149)
y<-dnbinom(x, size=1/0.3, mu=80)
pval1tail<-1-sum(y)
- x<-seq(from=1, to=149)
y<-dnbinom(x, size=1/0.3, mu=80)
pval1tail<-1-sum(y)
```

**Question 17: The probability that the process generates a count 150 or higher is (please round to 4 decimal places)**

**Question 18: Unless we are only looking for overexpressed genes, we need to count the extreme values on the left of the distribution as well (2-tailed test), so, in order to find the p-value, we need to multiply the obtained value by (please enter the correct number)**

**Question 19: At a 0.05 level of significance, 2-tailed test, can we conclude that the given read count (150) couldn't have originated from the given distribution, and, therefore, the gene is overexpressed?**

Above, we used one read count to estimate gene expression. Normally, there will be at least a few and we need to compare several samples from one group (for example, treatment) to several samples from another group (control). So, instead of finding the probability of the distribution to produce a number equal to or more extreme than the given count, we will be calculating the probability of the distribution to produce several numbers that, as a whole, are more extreme than the given set of counts.

Luckily, a sum of negative binomial distributions is also a negative binomial distribution and the parameters of this group distribution can be easily found. A group count can be evaluated using a group distribution just like a single count in the example above was evaluated using a single negative binomial distribution.

How do we know the mean and dispersion of the distribution? As long as there are at least a few samples with non-zero counts, we have a set of counts that can be fitted by a negative binomial distribution. The distribution can be built from the control group or both groups (the null hypothesis is that there is no difference in gene expression between the two groups, so it is OK to merge the groups together when estimating the parameters). A separate distribution can be built for each gene or genes can be pulled together to estimate the dispersion.

These are the general ideas behind using the Poisson and the negative binomial distributions to analyze RNA-Seq data. Every specific implementation is different. In the next section, we will analyze real life data using the edgeR package.

## The edgeR Package

edgeR is a package for R that uses the negative binomial distribution to find genes with differential expression. Let's install this package. In R, type

```
source("http://bioconductor.org/biocLite.R")
biocLite("edgeR")
```

Say yes to an update of outdated packages if prompted.

We will start with creating and analyzing random data. First, please type

```
set.seed(0)
```

to ensure that the “random” data generated in your R environment is the same as in this lecture. Next, generate random read counts from the negative binomial distribution with mean 75 and dispersion 0.2. We could have used any random data here, but it would be interesting to check whether or not edgeR will recover the value of the dispersion from the data correctly.

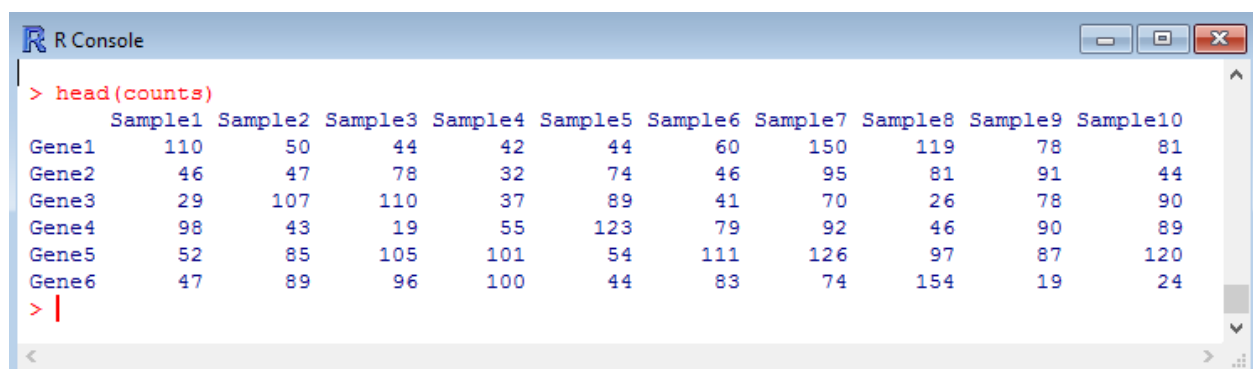
```
counts <- matrix(rnbinom(100000, size=1/0.2, mu=75), ncol=10)
```

Matrix counts has 10,000 rows that correspond to 10,000 genes and 10 columns that correspond to 10 samples. Let's name the rows and columns correspondingly

```
rownames(counts) <- paste("Gene", seq(1:10000), sep="")
colnames(counts) <- paste("Sample", seq(1:10), sep="")
```

Type head(counts) to view the first few rows of the data

As long as you generate the matrix with random data right after typing set.seed(0), it will be exactly as in this picture



	Sample1	Sample2	Sample3	Sample4	Sample5	Sample6	Sample7	Sample8	Sample9	Sample10
Gene1	110	50	44	42	44	60	150	119	78	81
Gene2	46	47	78	32	74	46	95	81	91	44
Gene3	29	107	110	37	89	41	70	26	78	90
Gene4	98	43	19	55	123	79	92	46	90	89
Gene5	52	85	105	101	54	111	126	97	87	120
Gene6	47	89	96	100	44	83	74	154	19	24

We will assign samples 1 to 5 to the first group and samples 6 to 10 to the second group and will try to find genes with different expression between these two groups. We know that there is no real differential expression here, any difference comes from random variation, but edgeR doesn't know it.

```
group <- c(rep("Group1", 5), rep("Group2", 5))
```

Load edgeR package

```
library(edgeR)
```

Create an edgeR object

```
edgeRlist <- DGEList(counts, group = group)
```

Type edgeRlist to view this object

### Question 20: lib.size is

- the total number of reads for each gene
- the total number of reads for each sample
- the total number of reads for each group

We'll filter out genes with low read counts and only keep those genes that have at least 1 read per million in at least 3 samples. This won't remove any data from our random dataset, but it will make a difference with real data where zero counts are common.



```
edgeRlist <- edgeRlist[rowSums(1000000*  
edgeRlist$counts/expandAsMatrix(edgeRlist$samples$lib.size, dim(edgeRlist)) > 1) >= 3, ]
```

Next, we'll apply normalization to correct for the different compositions of the samples.

```
edgeRlist <- calcNormFactors(edgeRlist)
```

Now we can estimate the common dispersion which is the dispersion across all genes

```
edgeRlist <- estimateCommonDisp(edgeRlist)
```

The resulting dispersion is written into the edgeRlist object

**Question 21: What is the value of the common dispersion? (please round to two decimal places)**

**A hint: please make sure you are using the same counts dataset as in the picture.**

**Question 22: Did edgeR get the dispersion correctly (within  $\pm 0.01$ ), i.e., is it the same as the dispersion that we used to generate the data?**

Estimate the dispersion for each gene

```
edgeRlist <- estimateTagwiseDisp(edgeRlist)
```

Finally, compute the p-values

```
edgeRresult <- exactTest(edgeRlist, dispersion="tagwise", pair = c( "Group1" , "Group2" ) )
```

and view the genes with the lowest p-values

```
topTags( edgeRresult, n = 30, sort.by = "PValue" )
```

The number of genes displayed is controlled by the parameter n.

**Question 23: (not for forum discussion, 2 points) How many genes have p-values equal or less than 0.001?**

**A hint: if you want to output many rows then, depending on your current settings, you might need to change the maximum number of rows printed, for example, `options(max.print=100)`**

Before proceeding, let's go back and discuss the data that we have been analyzing. These data were generated randomly. All genes in all samples had exactly the same level of expression, 75. We just added some random noise to account for the variation that is due to the nature of the sequencing process.

We know that there is no differential expression, but we have detected a number of genes with p-values equal or less than 0.001! A  $p = 0.001$  looks significant; if we didn't know that there were no differential expression in our data, we may have decided that the corresponding gene is over- or underexpressed.

The main reason why we shouldn't make the above conclusion is that a p-value indicates the significance of the result if there is only one experiment. For example, if we have a reason to believe that a specific gene is overexpressed, measure the expression of just this one gene and find that the p-value is 0.001, then it is a significant finding. However, if we repeat our experiment many times, we will eventually get  $p=0.001$  even if there is no differential expression. If we test many genes, we will eventually find one with  $p=0.001$  even if there is no real differential expression for any of these genes.

**Question 24: On average, if there is no differential expression, how many genes out of 10,000 (the size of our random set) should have p-values equal or less than 0.001?**

**A hint: the answer is based on the definition of the p-value.**

Therefore, the number of genes with the p-values equal or less than 0.001 is of the same order of magnitude as expected. The reason why it is higher than expected is that, because the EdgeR model has many parameters (a separate value of dispersion for each gene), there is some overfitting. If `edgeRresult <- exactTest(edgeRlist, dispersion="common", pair = c( "Group1" , "Group2" ) )` were used instead of `edgeRresult <- exactTest(edgeRlist, dispersion="tagwise", pair = c( "Group1" , "Group2" ) )` then the number of genes with the p-values equal or less than 0.001 would be about the same as expected.

EdgeR has a built-in function `p.adjust()` that allows one to adjust the p-values to account for multiple hypothesis testing. For example, the code below provides adjusted p-values based on the FDR (False Discovery Rate) adjustment method

```
pAdjusted<-p.adjust(unlist(edgeRresult$stable["PValue"]), 'fdr')
```

and here is Bonferroni's method:

```
pAdjusted<-p.adjust(unlist(edgeRresult$stable["PValue"]), 'bonferroni')
```

There are several adjustment methods and they provide different adjusted p-values because they use different algorithms and assumptions.

To get an idea on what these algorithms are based upon, we will continue evaluating the significance of the results manually, by comparing the expected number of genes with low p-values with the actual number of genes with low p-values, as we did in questions 23 and 24. This is the idea behind Bonferroni's method.

We are ready to analyze real data now. The data file can be downloaded from here

[https://sites.google.com/site/davismcc/useful-documents/pnas\\_expression.txt](https://sites.google.com/site/davismcc/useful-documents/pnas_expression.txt)

The data are from the experiment that is described here

<http://www.pnas.org/cgi/doi/10.1073/pnas.0807121105>

After you have downloaded the data file, set the working directory to where the data file is located. In windows, use `\\` or `/` instead of `\` in the directory path

```
setwd("<the path to the data file>")
```

Read the data file and rearrange the data to fit edgeR requirements

```
data <- read.table( file = "pnas_expression.txt" , header = TRUE )
counts <- data[ , -c(1, ncol(data))]
rownames(counts) <- data[, 1]
colnames(counts) <- paste("Sample", seq(1:7), sep="")
```

Please compute the p-values with all the same settings and options (including tagwise dispersion) as we used for the random set. The only difference is that there are seven samples now. The first four samples are controls; they are from untreated cancer cells. The last three samples are from treated cancer cells. The last column in the data file is not used.

**Question 25: What is the value of the common dispersion (please round to 2 decimal places)?**

**Question 26: Which gene has the lowest tagwise dispersion (please enter the id of that gene, as listed in the first column of the data file)?**

**A hint: to display tagwise dispersions, use `edgeRlist$tagwise.dispersion`. Some R functions that might be helpful include `min()` and `which.min()`. Elements of a vector are accessed with `[]` in R, so gene ids can be retrieved using `row.names(edgeRlist$counts)[<some number>]`.**

**Question 27: Which gene has the highest tagwise dispersion (please enter the id of that gene,**

as listed in the first column of the data file)?

**Question 28:** For how many genes the p-values are equal to or less than  $10^{-5}$ ?

**Question 29:** On average, if there is no differential expression, how many genes in this set (don't count the filtered out genes with zero counts) should have p-values equal to or less than  $10^{-5}$ ? (please round to the nearest whole number)

**Question 30:** For how many genes the p-values are equal to or less than  $10^{-10}$ ?

**Question 31:** On average, if there is no differential expression, how many genes in this set (don't count the filtered out genes with zero counts) should have p-values equal to or less than  $10^{-10}$  (please round to the nearest whole number)?

**Question 32:** Which five genes are the best candidates for being differentially expressed? (1 to 3 sentences are expected)

**Question 33:** How are these results different from those obtained from random data? (2 to 4 sentences are expected)