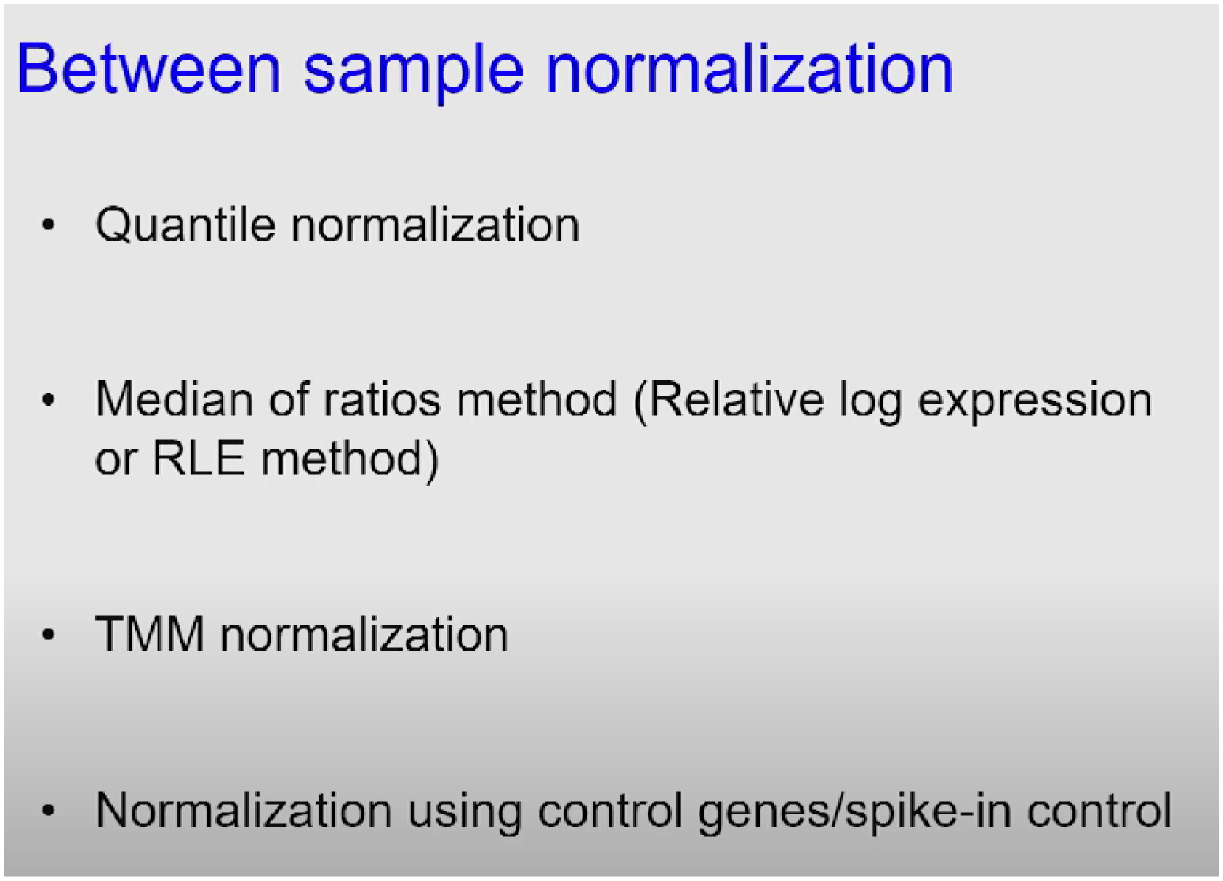
Next Generation Sequencing Technologies: Data Analysis and Applications

Data Normalization Methods (Contd.)

Dr. Riddhiman Dhar, Department of Biotechnology

Indian Institute of Technology, Kharagpur

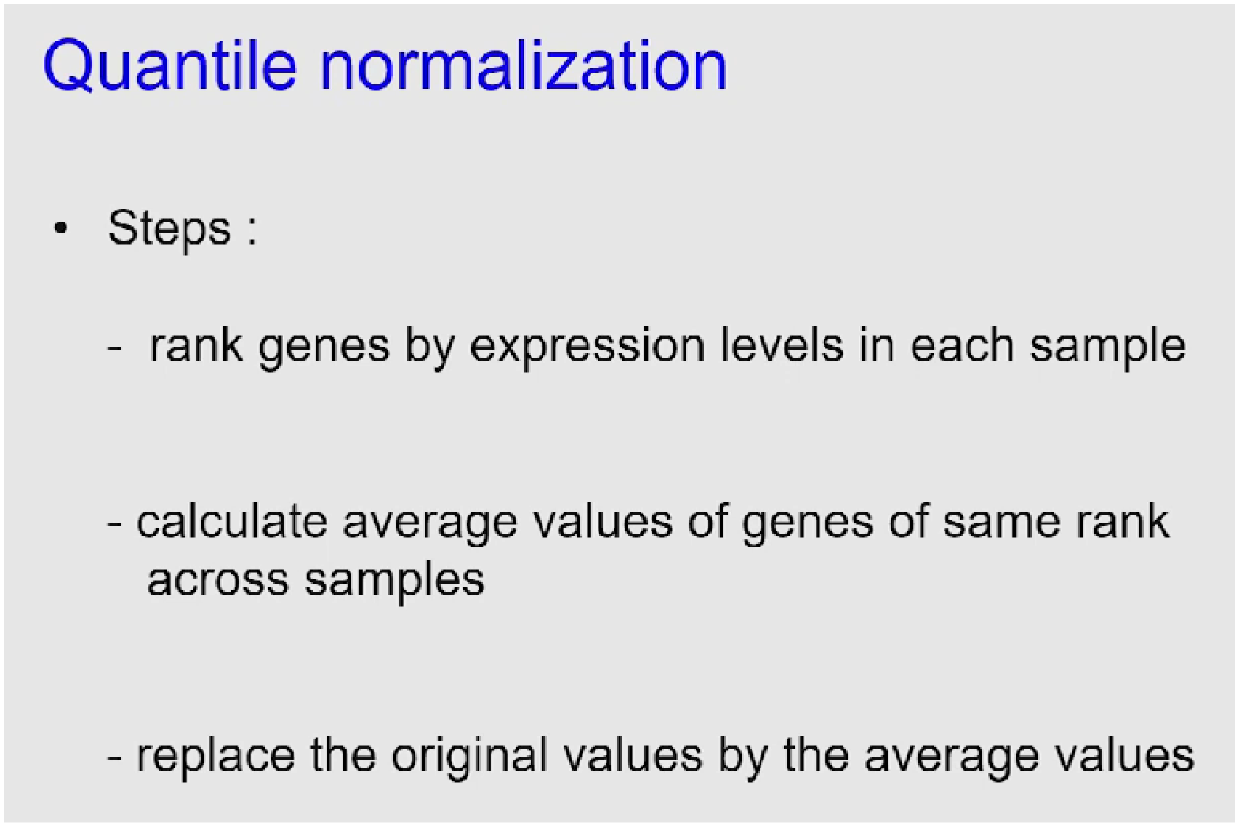
Good day, everyone. Welcome to the course on next-generation sequencing technologies, data analysis, and applications. In the last class, we started talking about data normalization methods.



Between sample normalization

- Quantile normalization
- Median of ratios method (Relative log expression or RLE method)
- TMM normalization
- Normalization using control genes/spike-in control

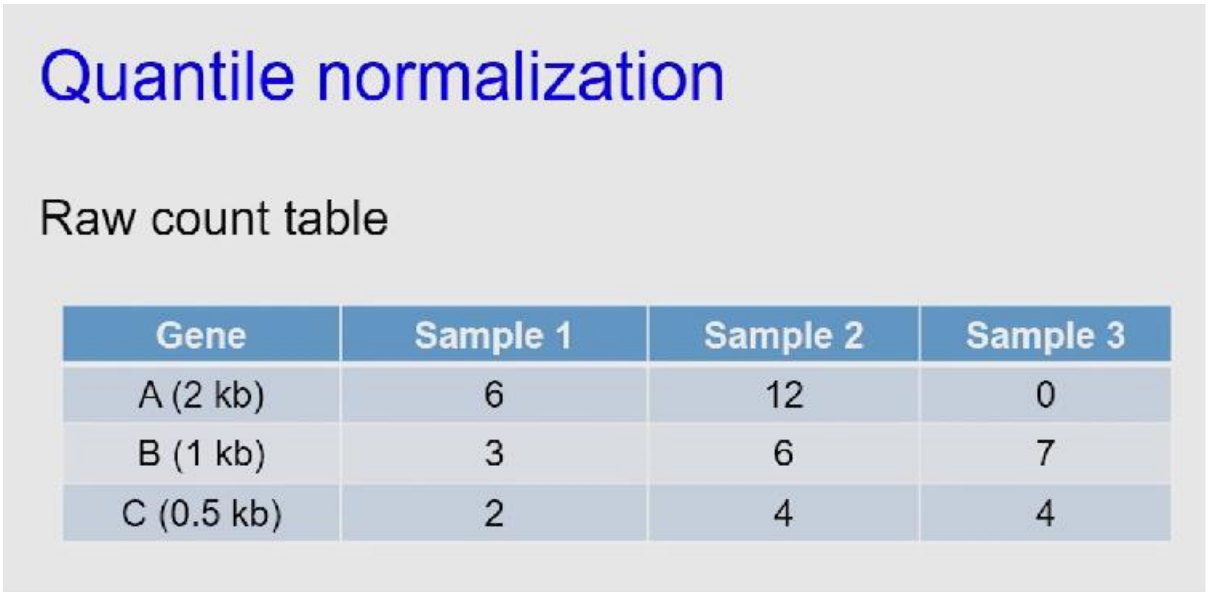
We talked about some of the methods that address this gene length bias as well as the sequencing depth variation. And then we talked about sample normalization. So, we will continue on that in this class, and we will discuss more sophisticated normalization methods. So, this is the agenda for this class. So, we will talk about sample normalization, and then we will talk about batch correction. So, in between sample normalizations, we have already discussed two methods. So, the first one is median normalization, the second one is UQ normalization, and we have also seen their drawbacks. So, one of the drawbacks is that they assume that their count data distribution differs by a constant across samples and this does not hold in most cases, and they also do not address this RNA composition bias. So, in this class, we will talk about different types of methods that can perhaps address these issues. The first method that we will talk about is called quantile normalization. Then we will discuss the median of ratios method, which is also sometimes referred to as or called the relative log expression or RLE method. It will give you identical results, and then we will talk about TMM normalization, and finally, we will talk about normalization using control genes or spikes in controls. So, what is quantum normalization? So, again, it starts with an assumption, ok? So, the assumption is that the read count distributions are identical across samples. So, this is a specific assumption, right? Maybe their genes are not showing the same expression, but the distributions are identical. They will not change between samples or, in other words, the distribution of gene expression levels is identical. Now, if you think about this assumption, you might have some questions, but keep those to yourself for the moment, because we will discuss a bit more about it. So, when you assume that the aim of normalization is that we want to make these read count distributions identical, So, if we assume they should be identical, in reality they are not. So, we would have to do some sort of transformation. So, these distributions become identical, right? So, that becomes the aim of this normalization method, and this is a normalization method that was employed for microarray and has also been adapted for RNA-Seq data.



Quantile normalization

- Steps :
  - rank genes by expression levels in each sample
  - calculate average values of genes of same rank across samples
  - replace the original values by the average values

So, what are the steps for quantile normalization? So, the first step is to rank genes by expression levels in each sample, then calculate the average values of genes of the same rank across samples and then replace the original values by these average values. So, we can actually take an example, and we can see how these steps are executed and the results that you get at the end. We go back to the same example, right?

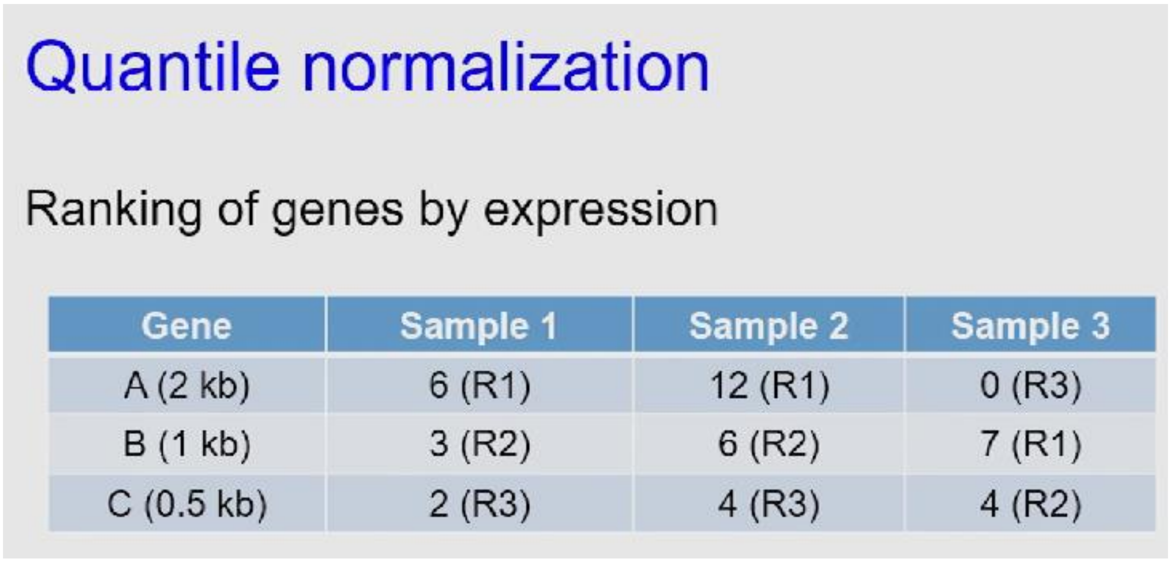


Quantile normalization

Raw count table

Gene	Sample 1	Sample 2	Sample 3
A (2 kb)	6	12	0
B (1 kb)	3	6	7
C (0.5 kb)	2	4	4

Because you are already familiar with this, we have the genes A, B, and C; we have the samples 1, 2, and 3; and we have the raw counts. I have not given the total because the total is not really important here, ok? So, what is the first step? So, we have to rank genes by expression, right?

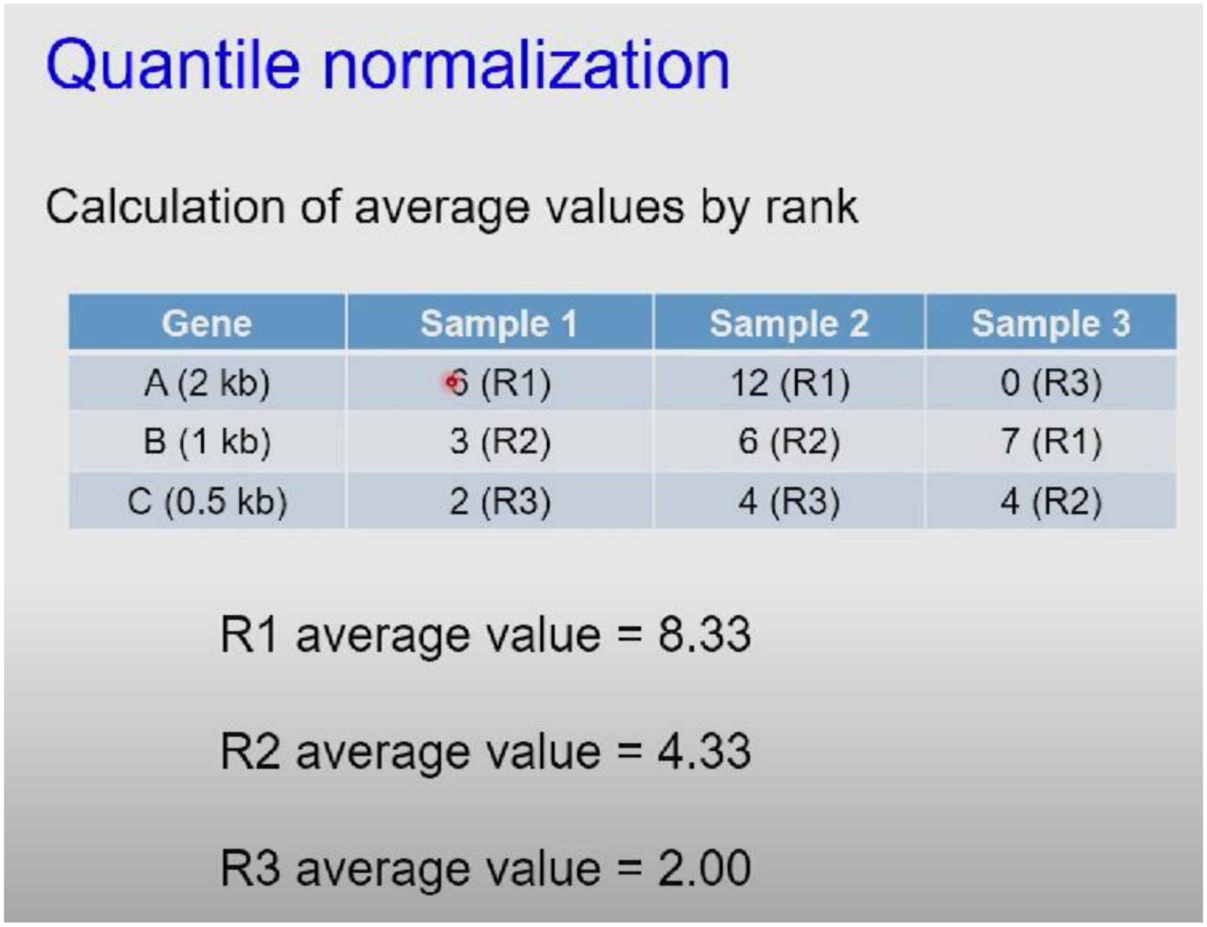


Quantile normalization

Ranking of genes by expression

Gene	Sample 1	Sample 2	Sample 3
A (2 kb)	6 (R1)	12 (R1)	0 (R3)
B (1 kb)	3 (R2)	6 (R2)	7 (R1)
C (0.5 kb)	2 (R3)	4 (R3)	4 (R2)

You can do low to high ranking or high to low ranking, right? So, here I have done high to low ranking, right? So, the numbers are the letters and the numbers that you see. So, R 1 stands for rank 1 right, R 2 stands for rank 2, and R 3 stands for rank 3. So, I have ranked these genes; we have only 3 genes. So, it is easy, right? So, R 1 is the rank 1 gene with the highest expression ok, and again, we have sample 2 again, rank 1 with the highest read count ok, and sample 3 we have also done the ranking ok. Now, the next step is to calculate the average values by rank ok. So, for rank 1, what is the average value? So, for rank 1, we have these 3 rank 1s. So, this is rank 1, this is rank 1, and this is rank 1. So, the total count is 6 plus 12 plus 7 right, and the average will be that divided by 3 right.



Quantile normalization

Calculation of average values by rank

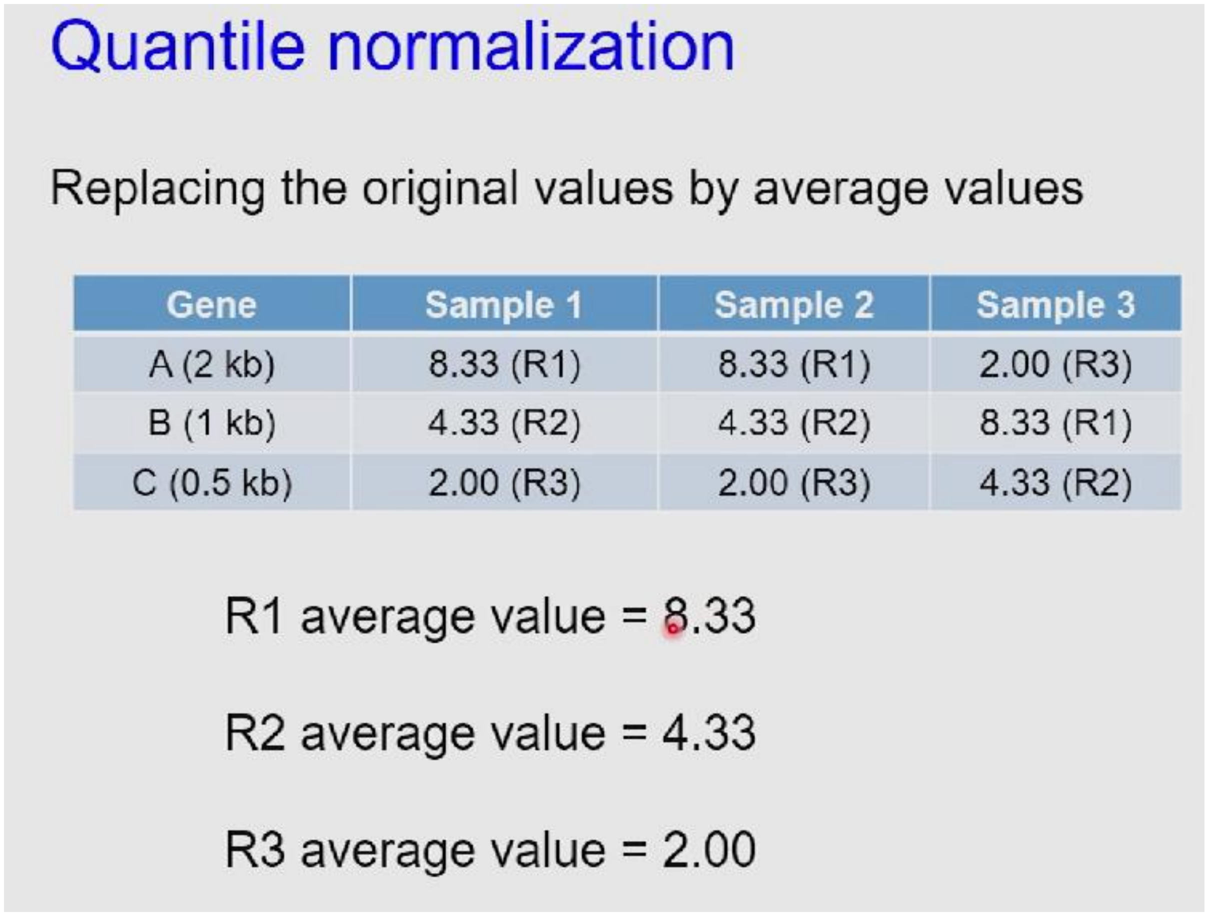
Gene	Sample 1	Sample 2	Sample 3
A (2 kb)	6 (R1)	12 (R1)	0 (R3)
B (1 kb)	3 (R2)	6 (R2)	7 (R1)
C (0.5 kb)	2 (R3)	4 (R3)	4 (R2)

R1 average value = 8.33

R2 average value = 4.33

R3 average value = 2.00

So, for rank 1, the average value is 8.33, and similarly we can calculate the R 2 average value and the R 3 average value. It is following the same process; we are calculating the average expression values by rank. So, this is what we get at the end. So, what is the next step? So, we have to replace the original values with the average values. So, average values of that rank are okay. So, again if we go back, right? So, this was rank 1, and we are replacing this with the actual value that we had earlier, right with this 8.33, which is the average value for rank 1. And similarly, here we are; we have also replaced this with 8.



Quantile normalization

Replacing the original values by average values

Gene	Sample 1	Sample 2	Sample 3
A (2 kb)	8.33 (R1)	8.33 (R1)	2.00 (R3)
B (1 kb)	4.33 (R2)	4.33 (R2)	8.33 (R1)
C (0.5 kb)	2.00 (R3)	2.00 (R3)	4.33 (R2)

R1 average value = 8.33

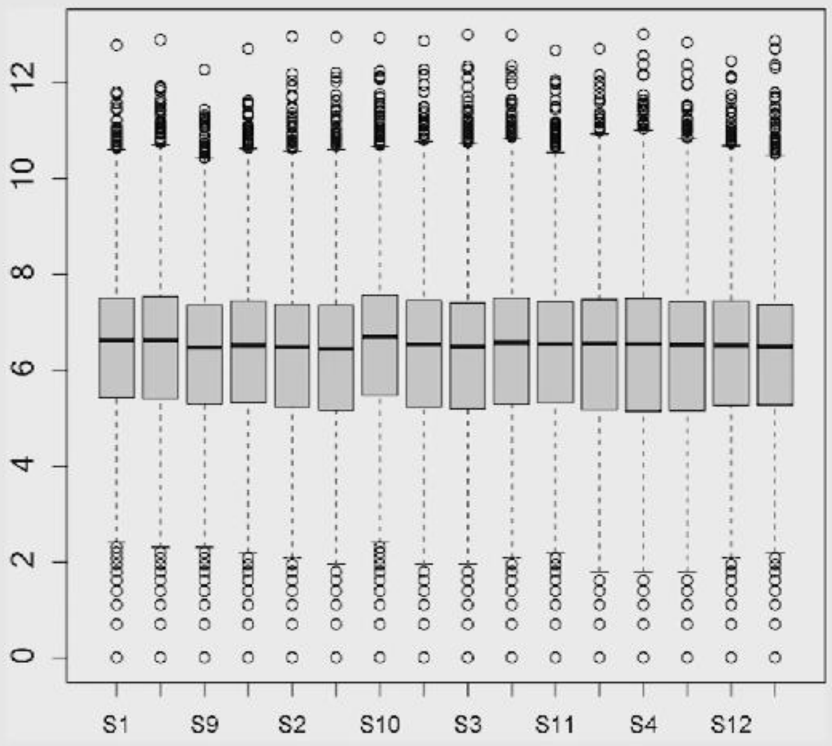
R2 average value = 4.33

R3 average value = 2.00

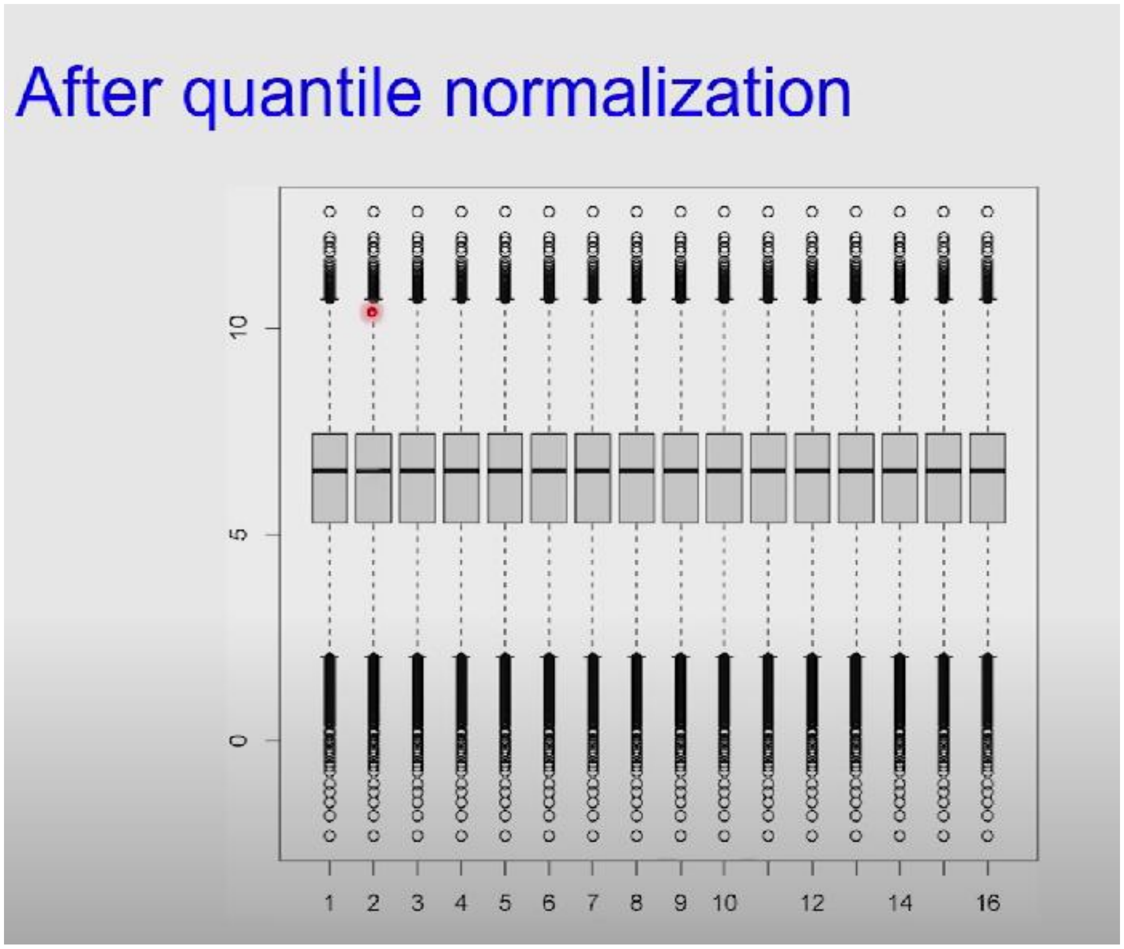
33 because this is rank 1. Similarly, here we have replaced this with rank 1 again, 8.33. Now, from what you see at the end of this normalization, the distributions are identical, right? For sample 1, you have 3 values: 8.

33, 4.33, and 2. For sample 2, you have 8.33, 4.33, and 2. For sample 3, you have 2, 8.33, and 4.33. The values are identical, but the genes may show differences in expression. That is what you see in this very small example. So, you follow this process for thousands of genes, and we generate identical distributions of read counts for each sample. And that is what I am going to show you now. Right here, I am showing you the raw count data.

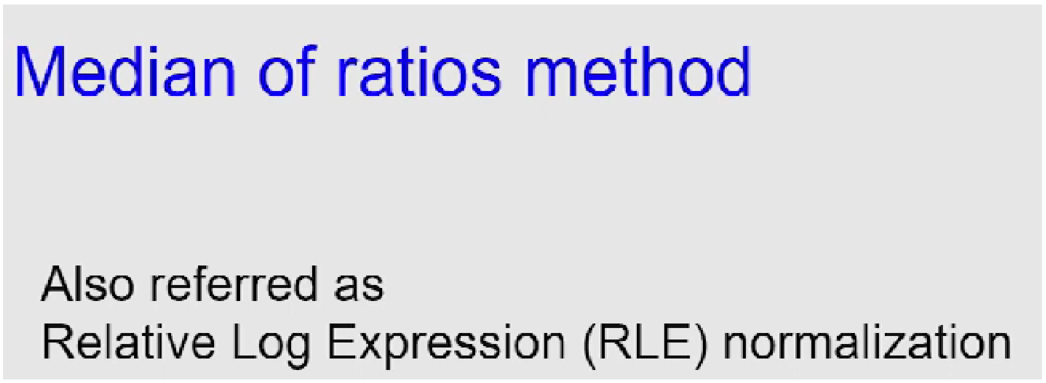
Before normalization (raw count data)



This is a simple box plot; this is not an RLE plot like I showed you earlier. This is a simple box plot where we are looking at the raw count data distribution for those samples again. And you can see that these distributions are slightly different. So, each dot here is a data point; these are the outliers, and you can see that the distributions are slightly different.



Now, after the normalization, right after the quantile normalization you will see the distributions will become identical, right? Even if you look at the points, they are at exactly the same place, and you see the median value is exactly identical. You have these boxes; they are exactly identical. So, they become identical because that is the idea of quantile normalization. We want to generate identical distributions for all samples. So, here you have 16 samples, and for each of them, you are getting these identical distributions. The genes may show differences in expression, but the distributions are identical. So, now, this is probably a question in your mind: can we assume this right? Can we assume that the distributions are identical? So, again, the answer is no. It may not always be the case that the distributions are identical because some genes may show very high expression, and that value may not be present in other samples. So, you can imagine different scenarios where this assumption will be violated. So, we come to the drawbacks, right? So, the assumption that the distributions are identical often does not hold right because of different scenarios. If you have, say, just two genes showing very high expression, you will not have those values in other samples. So, you are trying to artificially contain all those values by making these distributions identical. So, that is one of the first major assumptions, and this is a drawback, and again, we have not addressed this RNA composition bias. So, if you go back to our calculation, you can see right that we have not addressed this RNA composition bias, and if you compare again, it might appear that genes B and C are highly expressed in sample 3 compared to sample 1. So, this again makes it unsuitable for this comparative analysis between samples. So, we come to the next method, which is the median of ratios method.

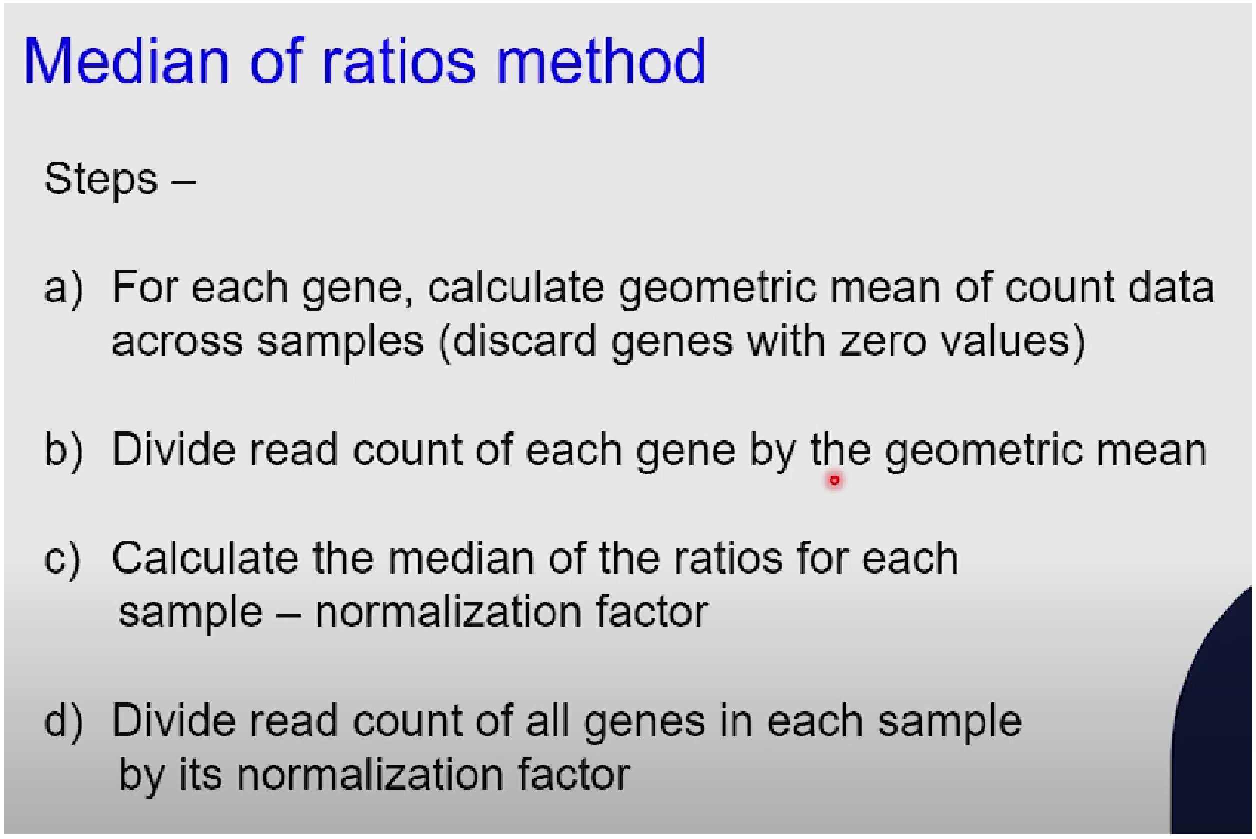


Median of ratios method

Also referred as
Relative Log Expression (RLE) normalization

So, this is a method that again does not assume anything right what it does is like a series of steps to actually address these RNA composition biases, along with other issues and biases that are present in the data. And sometimes it is also referred to as relative log expression normalization, although this was not introduced by the original authors. But if you see this term early normalization, you can probably then fall back. This is the same method as the median of ratios method. So, what are the steps here for this method? So, for each gene, we have to first calculate the geometric mean of count data across samples. So, this is a between-sample normalization.

So, we are looking at count data for each gene across samples, and then we calculate the geometric mean. And of course, in some cases if you have 0 values, the geometric mean would be 0. So, for genes where we get the geometric mean to be 0, we just discard them for the subsequent analysis. So, this is the first step. The next step is to divide the read count of each gene by the geometric mean. We will again take the example, and we will see the steps. Then you will understand this better, ok? So, we divide the read count of each gene by the geometric mean, and we calculate the median of the ratios that we get after the divisions in step b, and we use this median as the normalization factor for that sample. So, maybe this sounds very complex, but once we do the steps, it will be clear, ok? And so, we have calculated the normalization factor, and then we divide the raw read count of all genes in each sample by the normalization factor of that sample. So, these are the four steps we have. The first step is where we calculate the geometric mean for each gene.



Median of ratios method

Steps –

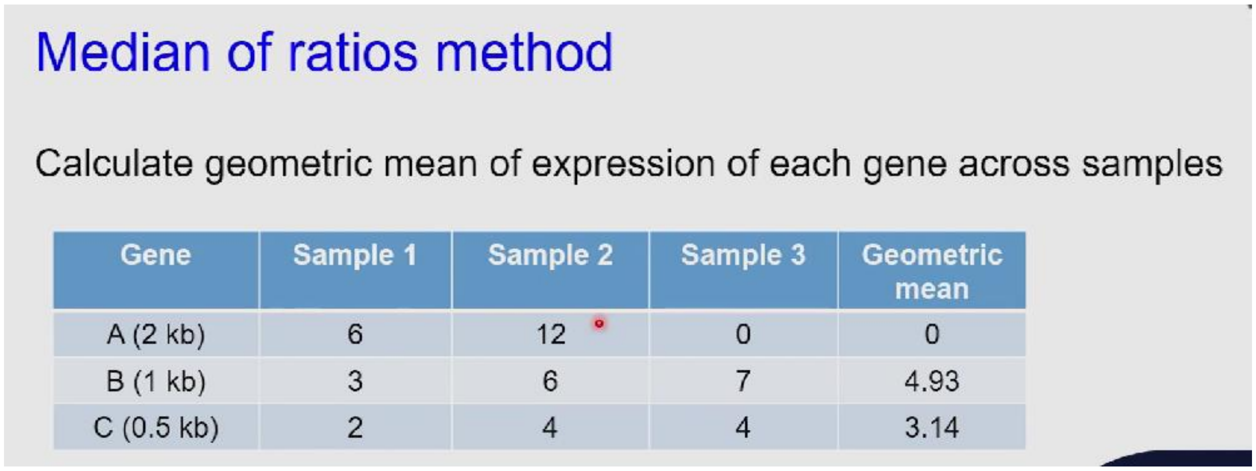
a) For each gene, calculate geometric mean of count data across samples (discard genes with zero values)

b) Divide read count of each gene by the geometric mean

c) Calculate the median of the ratios for each sample – normalization factor

d) Divide read count of all genes in each sample by its normalization factor

We divide the read count by the geometric mean, and once we get some ratios right, we get some values. We calculate the median of those ratios for each sample, and those median values for each sample will be the normalization factors for those samples. And using that normalization factor, you divide the read count of all genes in each sample. So, again, each sample will have its own normalization factor and we will use that to divide the read count, ok? So, let us now look at this right in action, and that will be clear, ok? We come back to the same example, right? So, we have sample 1, sample 2, and sample 3. Again, we have genes a, b, and c, and the first step is to calculate the geometric mean of expression of each gene across the samples.



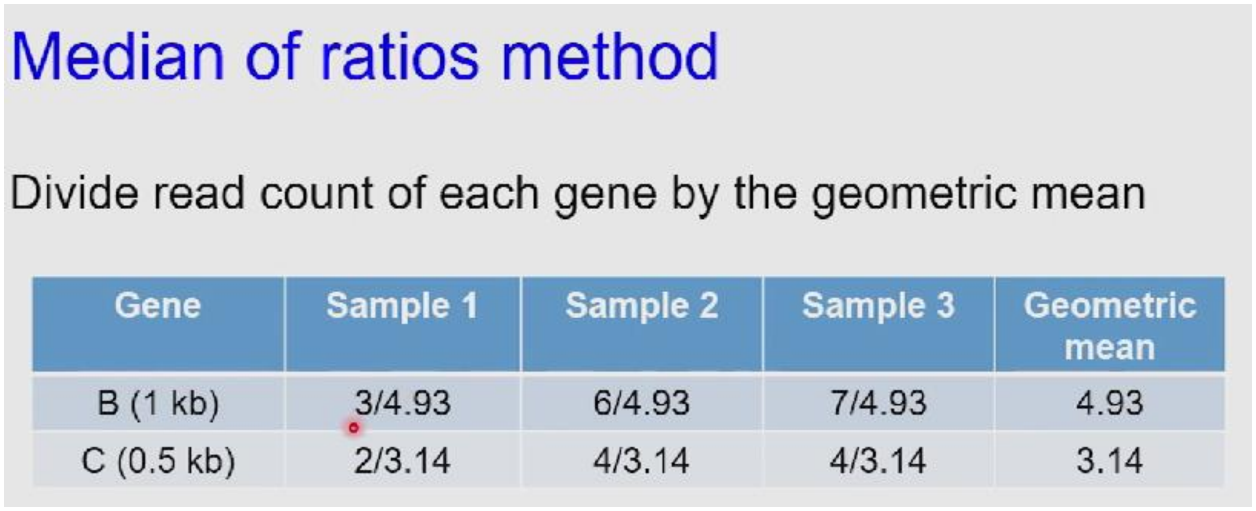
Median of ratios method

Calculate geometric mean of expression of each gene across samples

Gene	Sample 1	Sample 2	Sample 3	Geometric mean
A (2 kb)	6	12	0	0
B (1 kb)	3	6	7	4.93
C (0.5 kb)	2	4	4	3.14

So, we come here to this table, right? So, for gene A, we have to take these three values: 6, 12 and 0, and we have to calculate the geometric mean, which I have written on an extra column here, ok? So, for this sample, the read count is 0.

So, this geometric mean will be 0, right? So, that is something you are probably familiar with: how to calculate the geometric mean. For gene b, we have sample 1, sample 2, and sample 3. We calculate the geometric mean. If you know how to calculate the geometric mean, you should be able to do this. For gene C again, we take these three values and calculate the geometric mean, and you continue this. You do this for thousands of genes; if you have 10,000 genes, you do the same process for each of them. So, once we have done this, the next step is to divide the read count of each gene by the geometric mean.

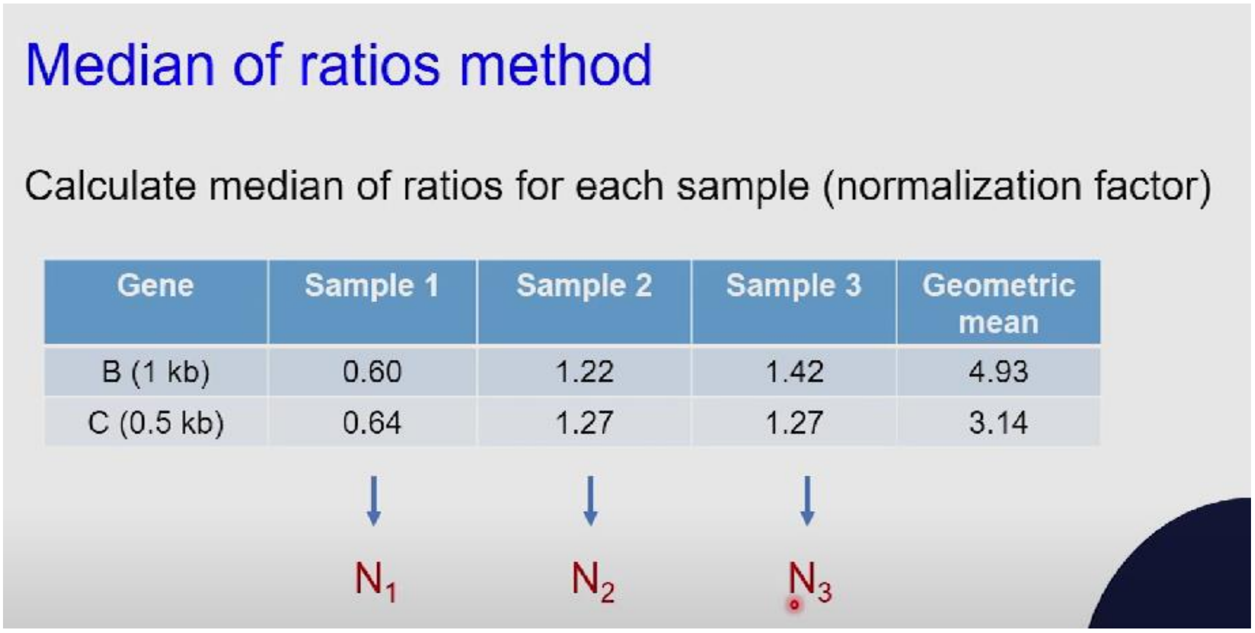


Median of ratios method

Divide read count of each gene by the geometric mean

Gene	Sample 1	Sample 2	Sample 3	Geometric mean
B (1 kb)	3/4.93	6/4.93	7/4.93	4.93
C (0.5 kb)	2/3.14	4/3.14	4/3.14	3.14

So, we have to divide these counts by the last column, okay? So, we have discarded the one with 0 value, right? Of course, you cannot divide by 0, and that is the first step. Remember, we have to discard genes for which the geometric mean is 0 ok. So, we are left with these two genes now, b and c, and for each of them, we have to divide by the geometric mean of that gene. So, for gene b, the geometric mean is 4.93. So, we are dividing the raw read count of 3 by 4.93, and similarly, we are doing the same for samples 2 and 3. For gene C, the geometric mean is 3.14. So, we divide the read counts by this number: 3.14. So, you can see this for sample 1, sample 2, and sample 3. We did the same, ok? Now, once we have done that right, we got the values. Now, after the division, these are the values. What you have to do now is calculate the median of the ratios for each sample, which will give us the normalization factor, sometimes also referred to as the scaling factor. So, you take the median for sample 1, right these ratios for sample 2, we take the median for sample 3, we take the median, and we get these three different medians. Of course, I will not take the median of two values; it does not make any sense, right? But you get the idea.

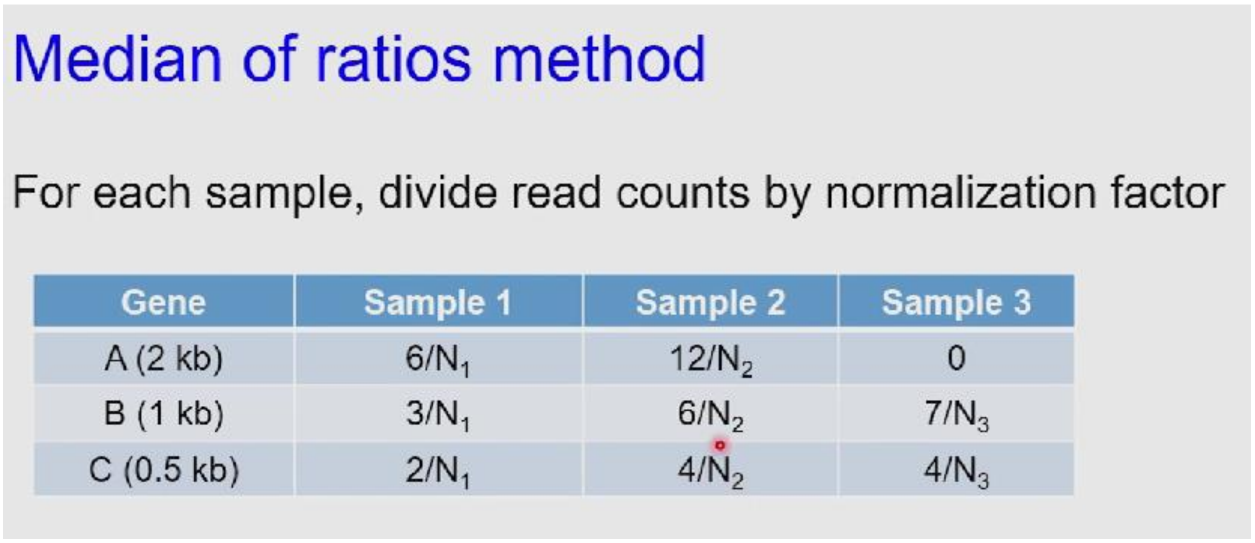


Median of ratios method

Calculate median of ratios for each sample (normalization factor)

Gene	Sample 1	Sample 2	Sample 3	Geometric mean
B (1 kb)	0.60	1.22	1.42	4.93
C (0.5 kb)	0.64	1.27	1.27	3.14
	$\downarrow$	$\downarrow$	$\downarrow$	
	N_1	N_2	N_3	

When you have thousands of genes, you take the median, and we will get some values right. So, I have simply given an example. Let us say N 1, N 2, and N 3 as medians for these samples. Once we get these medians, the next step is to divide these values by the raw counts that we have started with. So, we can now forget about this table. In this transform table, we go back to the raw count data, and then we can divide by N 1, N 2, and N 3. So, this is the next step. So, for example, divide the counts by the normalization factor, and for sample 1, the normalization factor is N 1. So, we are dividing these counts by N 1 for sample 2, and the normalization factor is N 2. So, we are dividing by this N 2 number here for sample 3, and the normalization factor is N 3. So, we are dividing by N3, ok. So, once we have done this, we get the normalized count data, ok.

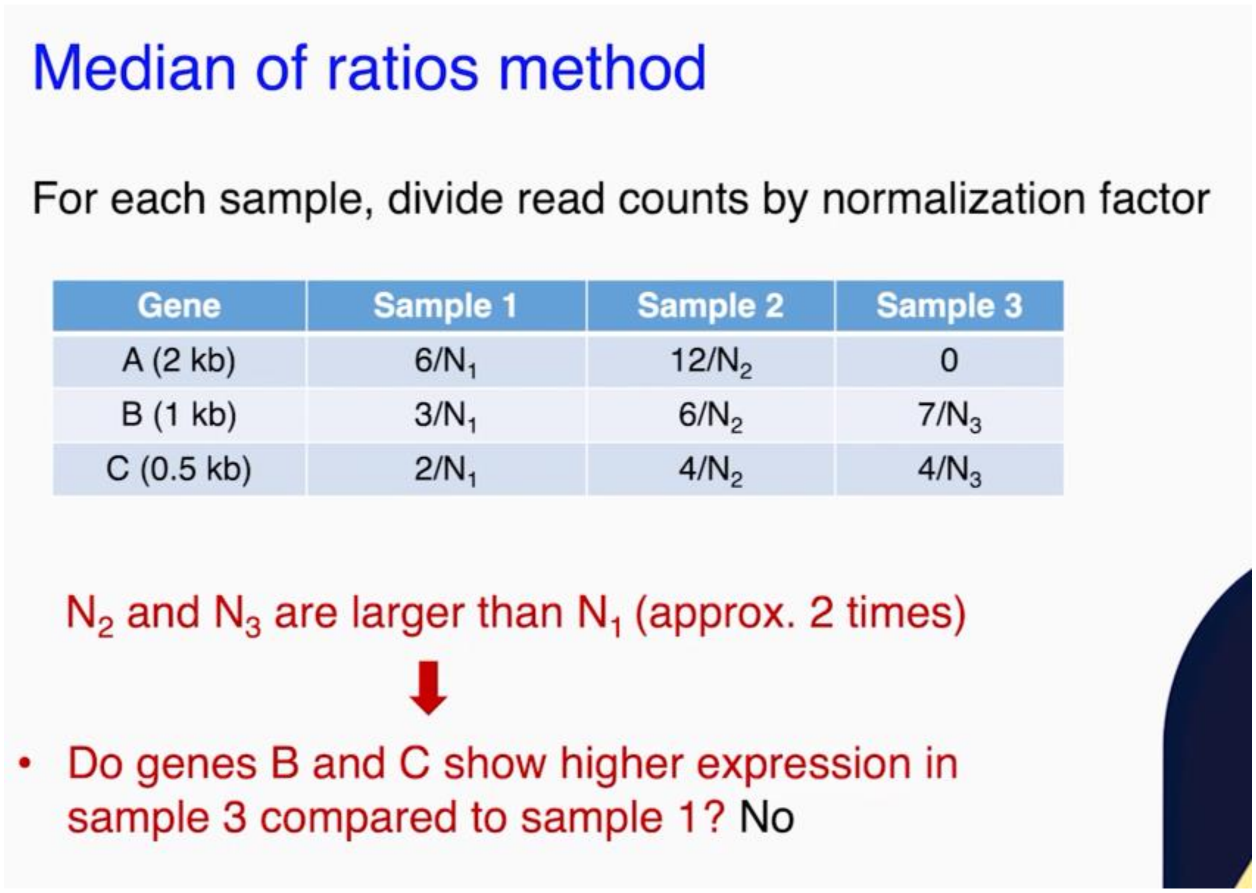


Median of ratios method

For each sample, divide read counts by normalization factor

Gene	Sample 1	Sample 2	Sample 3
A (2 kb)	$6/N_1$	$12/N_2$	0
B (1 kb)	$3/N_1$	$6/N_2$	$7/N_3$
C (0.5 kb)	$2/N_1$	$4/N_2$	$4/N_3$

So, one of the things you probably realize if you go back to this table, right? So, median, if you imagine that for samples 2 and 3, the median would be higher right compared to sample 1, it will probably be twofold, almost two times higher approximately, of course. Again, it does not make sense, but again, if you kind of think for a moment, this will be approximately two times right. So, N 2 and N 3 will be larger than N 1 in this example, as you can see. So, this is approximately two times. So, what it means, right, if you now look at these genes B and C here, So, if N 2 and N 3 are two times larger than N 1, So, once you have completed these divisions, So, 3 by N 1 will be almost similar in value to 7 by N 3, ok? And similarly, 2 by N 1 will be very similar to this 4 by N 3 value. What it means is that we have now addressed this RNA composition bias ok, that we were talking about for all the methods ok.



Median of ratios method

For each sample, divide read counts by normalization factor

Gene	Sample 1	Sample 2	Sample 3
A (2 kb)	$6/N_1$	$12/N_2$	0
B (1 kb)	$3/N_1$	$6/N_2$	$7/N_3$
C (0.5 kb)	$2/N_1$	$4/N_2$	$4/N_3$

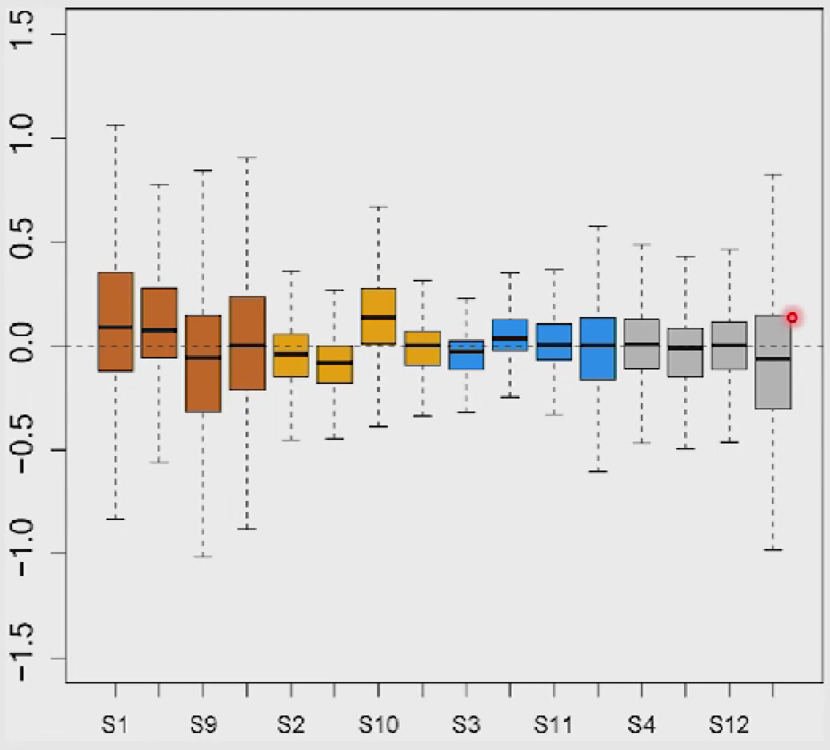
N_2 and N_3 are larger than N_1 (approx. 2 times)

⬇

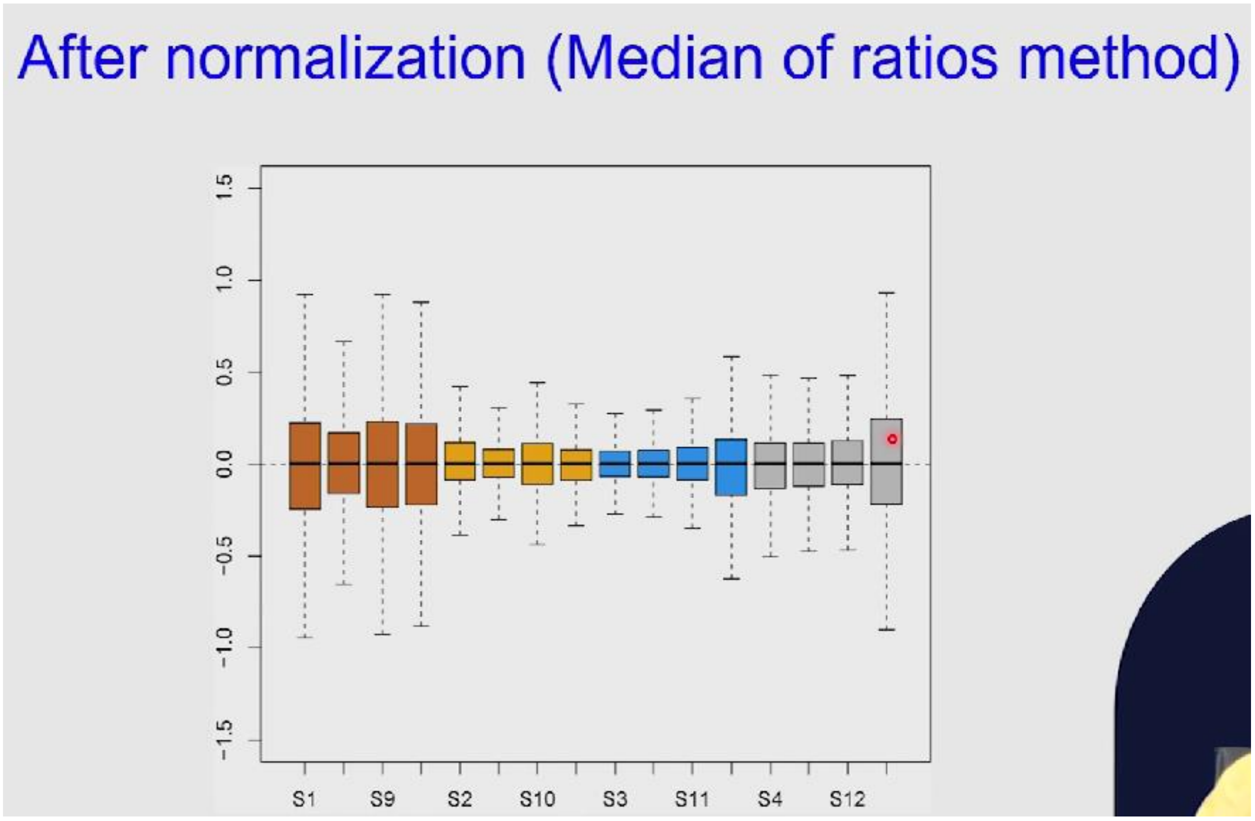
- Do genes B and C show higher expression in sample 3 compared to sample 1? No

By discarding this gene right where we have 0 value right we have calculated this geometric means we have calculated these ratios etcetera and taken the median we have now addressed this RNA composition bias here ok. So, this is something that is an advantage of this method, ok? So, now we can answer this question correctly, right? So, do genes B and C show lower expression in sample 3 compared to sample 1? That is actually no difference; there is almost no difference in expression between these two samples, sample 1 and sample 3. So, you can probably see now that the normalization factor N 3 is probably approximately two times greater than the normalization factor N 1, of course. Again, if you have to do this for a large number of genes, you cannot calculate just the median, but just as an example, this holds ok.

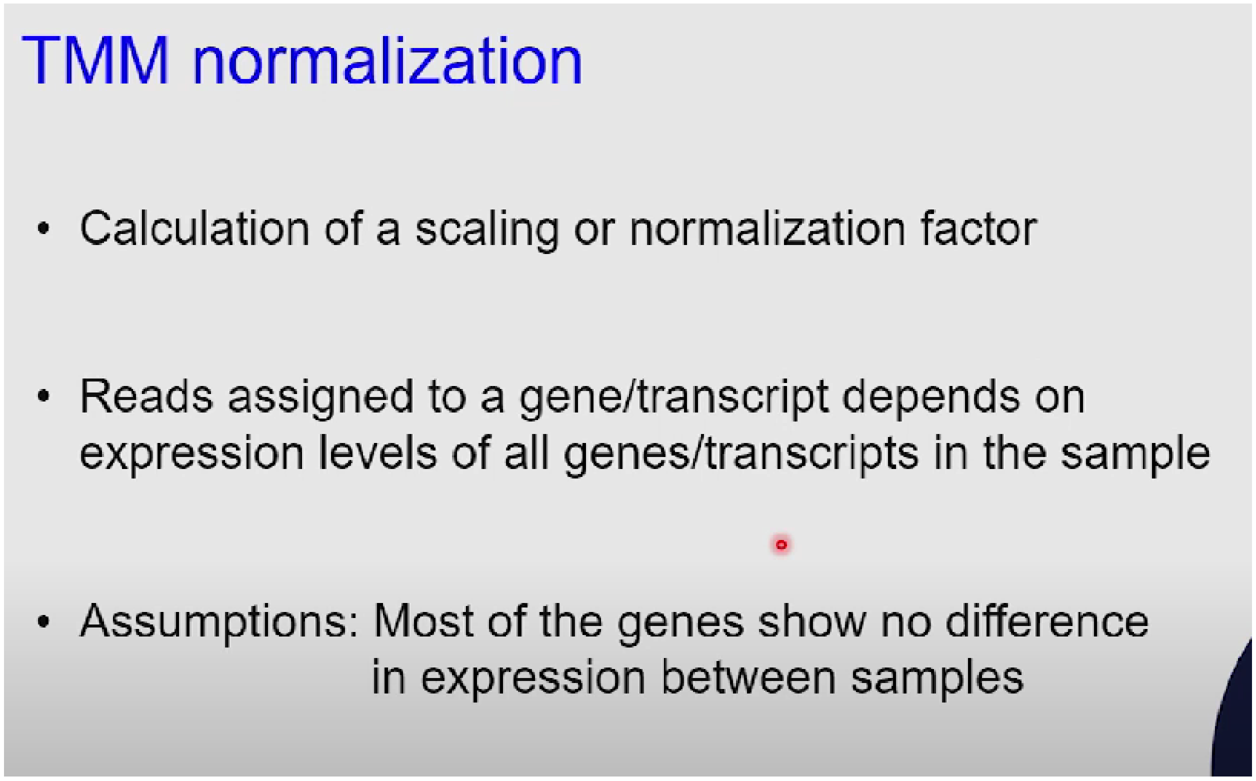
Before normalization



So, again, we go back to those RLE plots, and I can show you what the data would look like after normalization, right? So, this is before normalization this is the raw data right again transformed because of this RLE method.



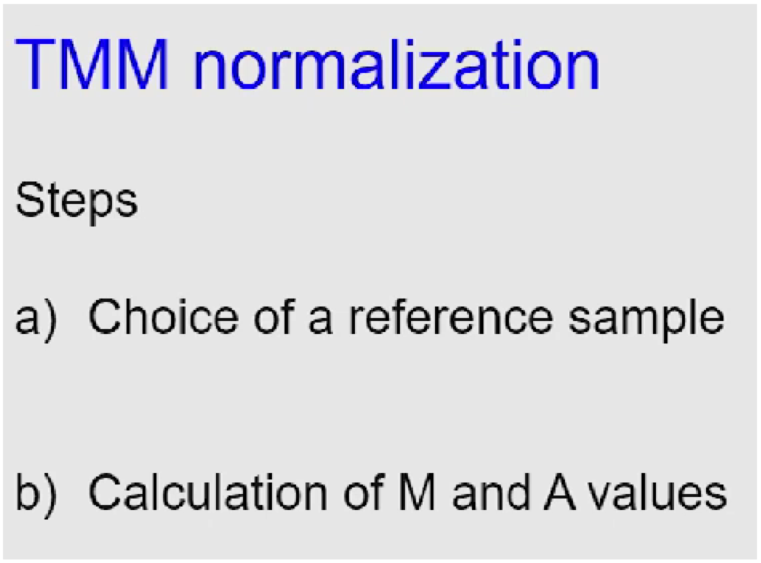
And then this is after normalization using this median of ratios method. And here you see, right again, that the medians are more or less identical and the distributions have changed. So, you can go back, right? These are the raw data distributions, and after this normalization, you get this kind of distribution for each sample. Now, as you can probably guess, the RLE plot is actually related to this RLE term right relative log expression. So, when you do this and make this plot, you kind of follow the same procedure: you calculate the geometric mean and then divide by the geometric mean to get this calculation of normalization factors, and then you get these plots. So, that is why they are kind of centered around 0, okay? So, what are the what are the drawbacks of this median of ratios method? We have seen that this can address the RLE composition bias issue, right? So, one of the major drawbacks, of course, is that there is no consideration for gene length. So, this means we cannot do a comparison within a sample, right? So, we cannot compare the expression levels of genes within a sample. So, this will require some other method right within the sample normalization method. So, the median of ratios method is strictly between sample normalization, right? o, we can compare across samples, but not within samples. So, the next normalization method that is also very popular is called TMM normalization. So, again, let us look at this. So, what it does in its full form is, of course, the stream mean of M values, right? So, this TMM is right.



TMM normalization

- Calculation of a scaling or normalization factor
- Reads assigned to a gene/transcript depends on expression levels of all genes/transcripts in the sample
- Assumptions: Most of the genes show no difference in expression between samples

So, we will talk about what these M values are that you get. So, this method also calculates a scaling or normalization factor, like the earlier method that we discussed. And so, what happens is that we now understand that the count that we get for a gene or transcript is not just proportional; it is not just dependent on the expression of that gene. So, it also depends on the expression levels of all gene transcripts in the sample. We have seen this in our small example, ok? And one of the assumptions this method makes is that there is no difference in the expression of most of the genes between samples. So, if you take, let us say, 1000 genes, probably 80 percent or 70 percent do not show any difference in expression, ok? So, this is an assumption in this method, and maybe 20 percent, 30 percent, or so they show a difference in expression between samples, ok. Based on the assumption, this method calculates a scaling or normalization factor, ok. So, let us see the steps right how it works. So, the first step in this method is that you have to choose a reference sample. So, when you are comparing two samples, you choose a reference sample against which you will compare another sample. So, once you have chosen this reference sample, you are comparing these other samples, and you can calculate something called M and A values. So, these M and A values will come later on as well for different purposes when you talk about differential expression analysis, etcetera.



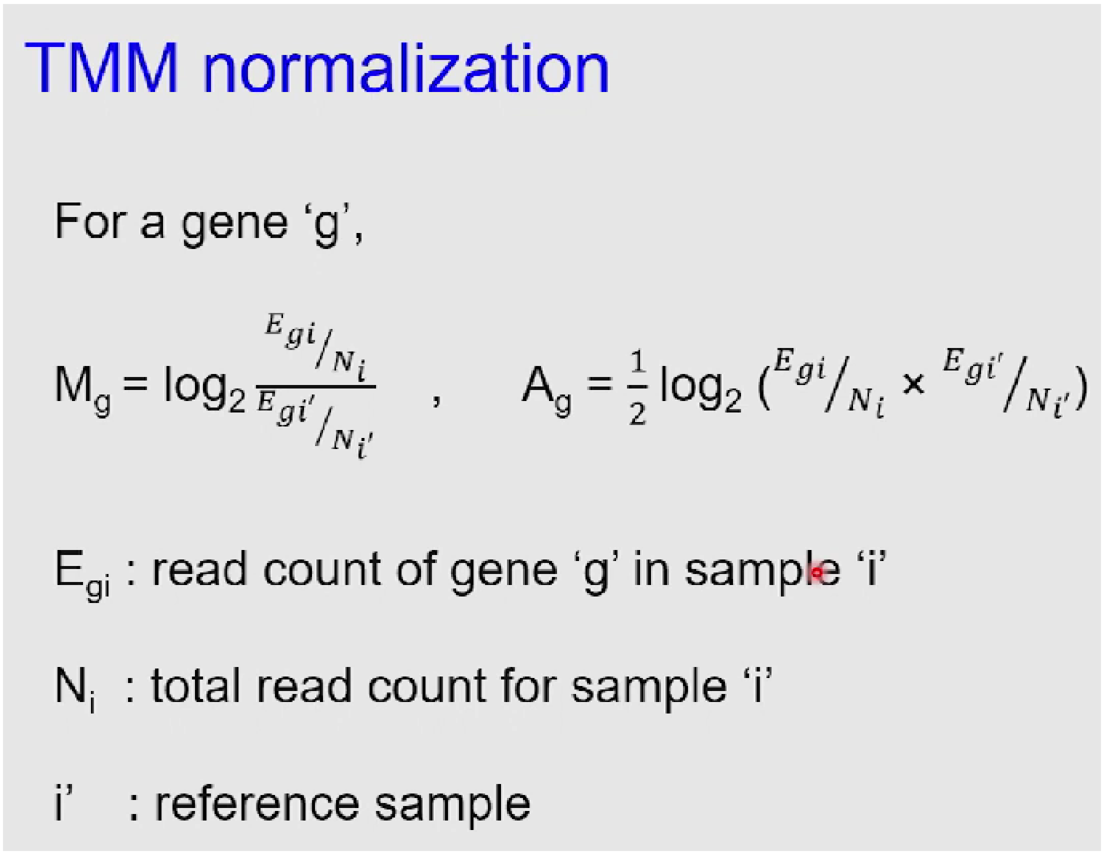
TMM normalization

Steps

a) Choice of a reference sample

b) Calculation of M and A values

So, how do you calculate these M and A values? So, for gene G right, this is M G right, so these M and A values are calculated for every gene right that is present in the data. So, M G is simply the log of this E g i divided by N i and divided by the E g i prime divided by N i prime.



TMM normalization

For a gene ‘g’,

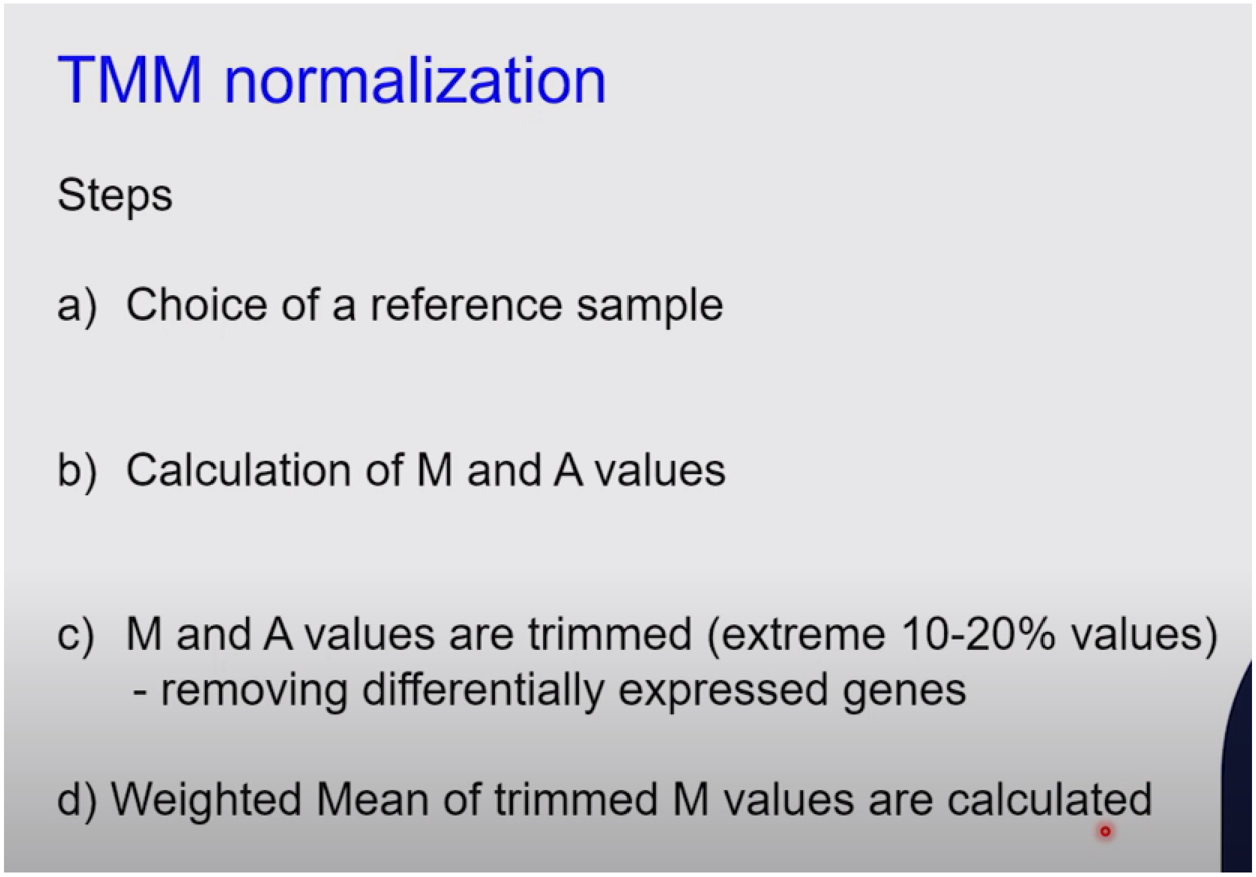
$$M_g = \log_2 \frac{E_{gi}/N_i}{E_{gi'}/N_{i'}} \quad , \qquad A_g = \frac{1}{2}\log_2 \left(E_{gi}/N_i \times E_{gi'}/N_{i'} \right)$$

E_{gi} : read count of gene ‘g’ in sample ‘i’

N_i : total read count for sample ‘i’

i’ : reference sample

Now, this E g i is the read count or expression of gene G in sample i, and N i is the total read count of sample i. And the lower part, E g i prime and N i prime, is for the reference sample, right? So, in the first step, you choose the reference sample, and then you get all these values. Similarly, you can calculate this A g; this is kind of an average expression, right? So, again, you have this half log of 2 E g i divided by N i times E g i prime times N i prime. Again, we know that E g i is the read count for gene G in sample i, and N i is the total read count. Then, for similar measures, we have E g i prime and N i prime. This is for the reference sample. So, once you have calculated this right, the next step is to trim these M and A values, which are usually extreme 10, 20 percent, or 30 percent, and so trim out those genes that show these extreme 10, 20 percent values. So, you get a distribution and trim out these 10, 20, or 30 percent values, ok? And what it means is that you are removing genes that are showing differential expression, ok? And they are showing differences in expression between samples, and you are removing those genes before you calculate the normalization factor, ok? Once we have removed these genes, we calculate the weighted mean of the trimmed M values. So, we do not take these extreme genes, and they are M values; we just take the filtered M values or trimmed M values, as we call them, and take the weighted mean. Now, the question is: What is this weight? Right? How do you calculate this weighted mean? So, weights are usually inversely proportional to the variance in read count. And this is to avoid genes that show a lot of fluctuation between samples. Still, even after filtering, they should not bias the mean too much. So, that is why the weights are inversely proportional to the variance in read count.



TMM normalization

Steps

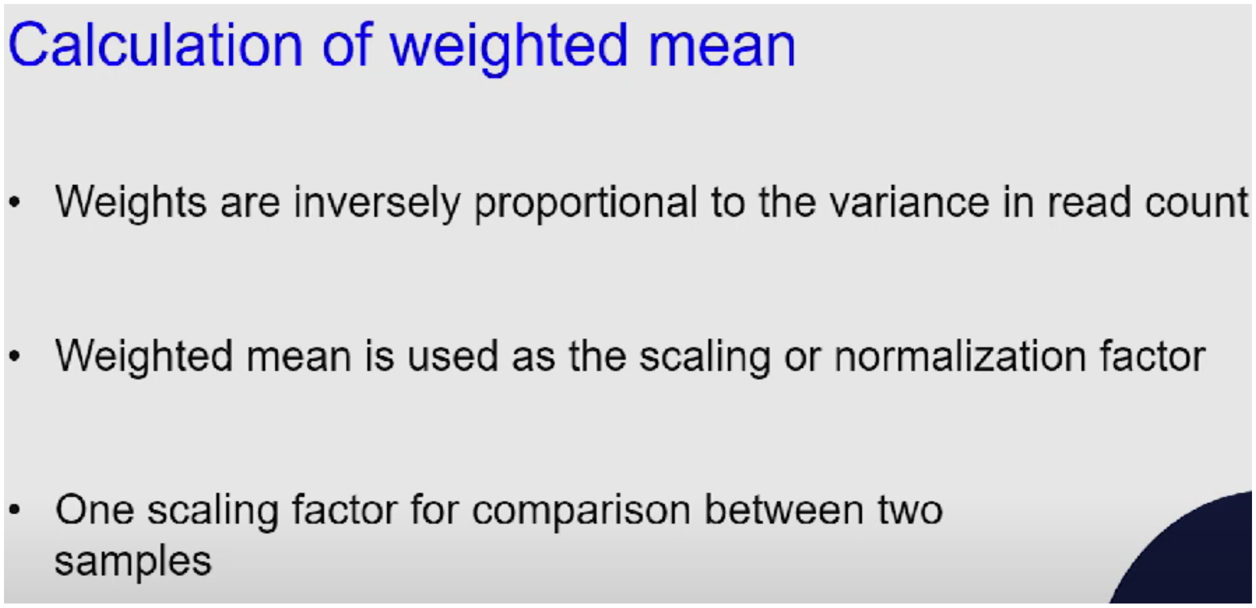
a) Choice of a reference sample

b) Calculation of M and A values

c) M and A values are trimmed (extreme 10-20% values)
 - removing differentially expressed genes

d) Weighted Mean of trimmed M values are calculated

And again, the idea behind this is that the genes that show constant expression across samples should have the highest weight. They are clearly not differentially expressed. So, they should have the highest weight in determining the normalization factor, okay? And this weighted mean is then used as the scaling or normalization factor. And so, you are comparing two samples, and what you will end up with is one scaling factor for comparison between these two samples.

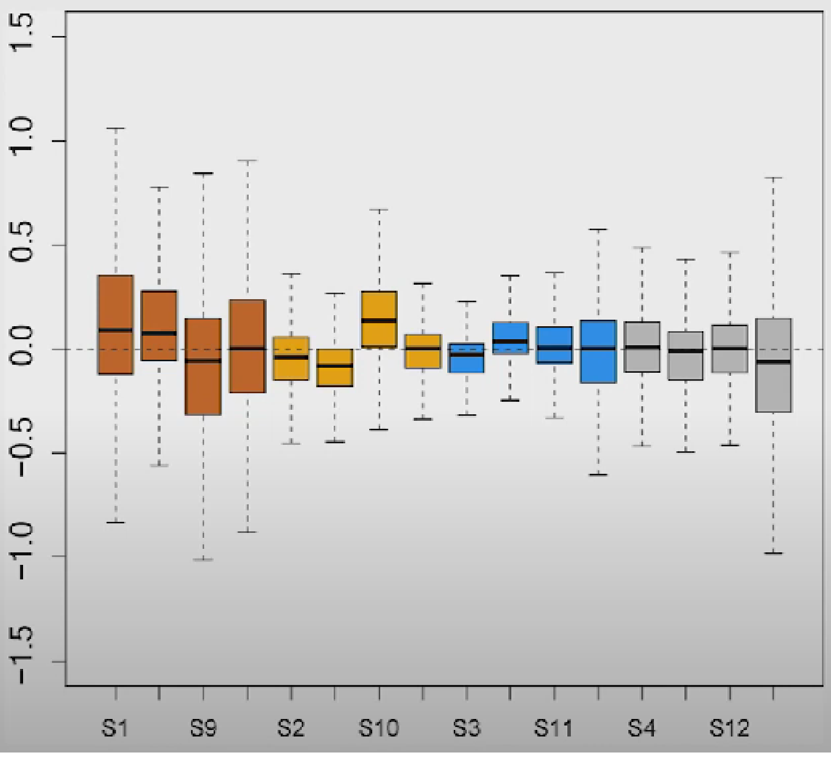


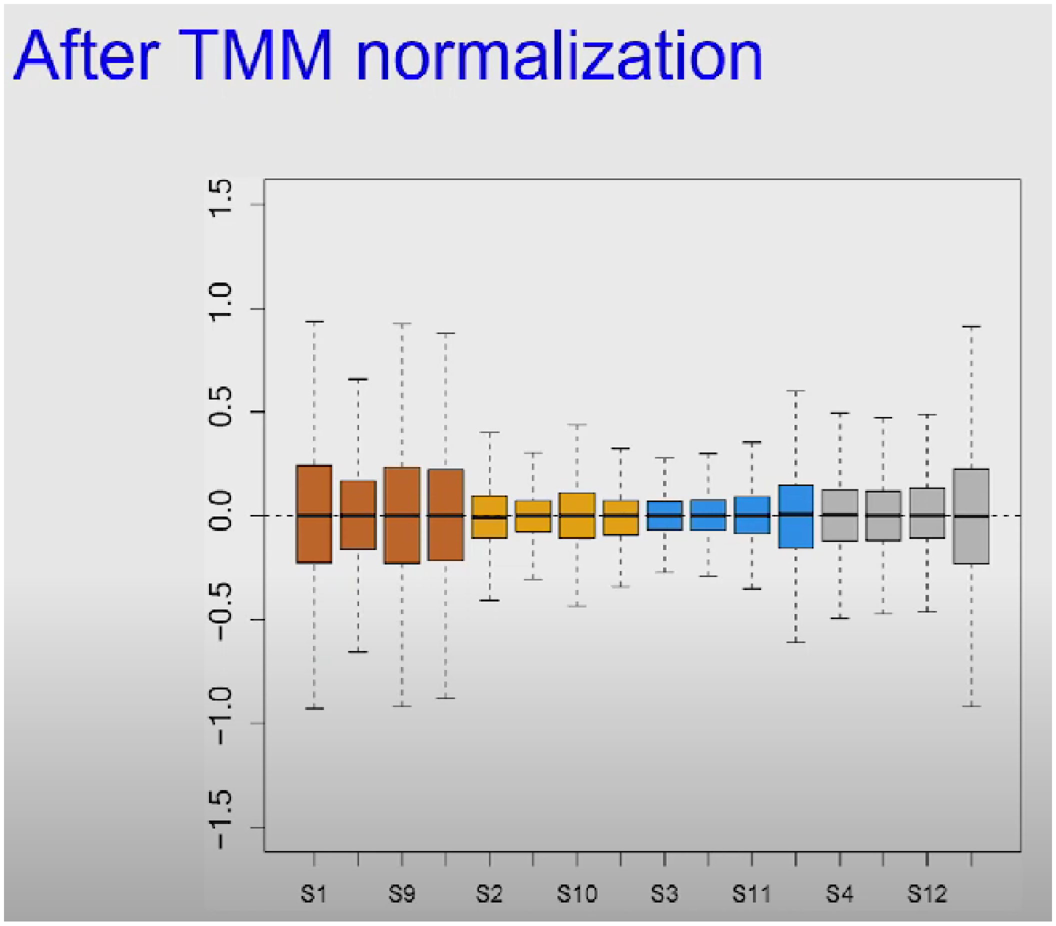
Calculation of weighted mean

- Weights are inversely proportional to the variance in read count
- Weighted mean is used as the scaling or normalization factor
- One scaling factor for comparison between two samples

And you can then use this scaling factor to transform this data into a reference as well as a non-reference sample. If you are comparing between multiple samples, we choose one reference, and then we can calculate scaling factors for each pair-wise comparison. So, in that way, you can handle these multiple comparisons as well, alright? So, let us again go back to the example, right? This is the real data set.

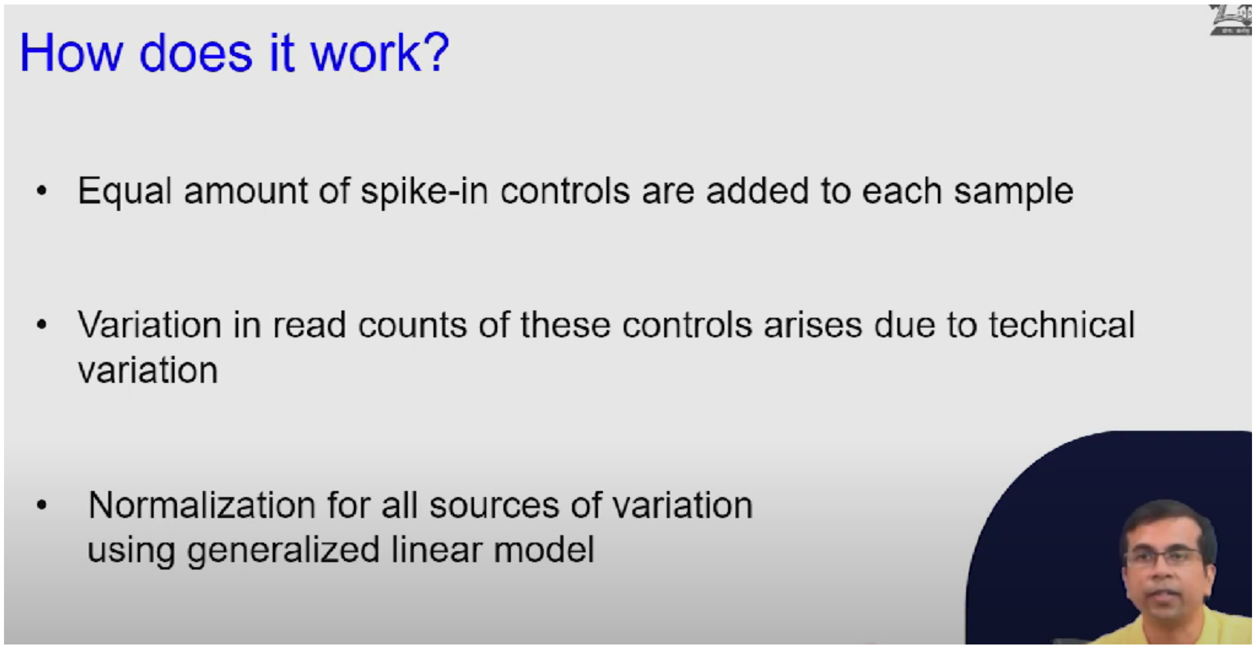
Before normalization





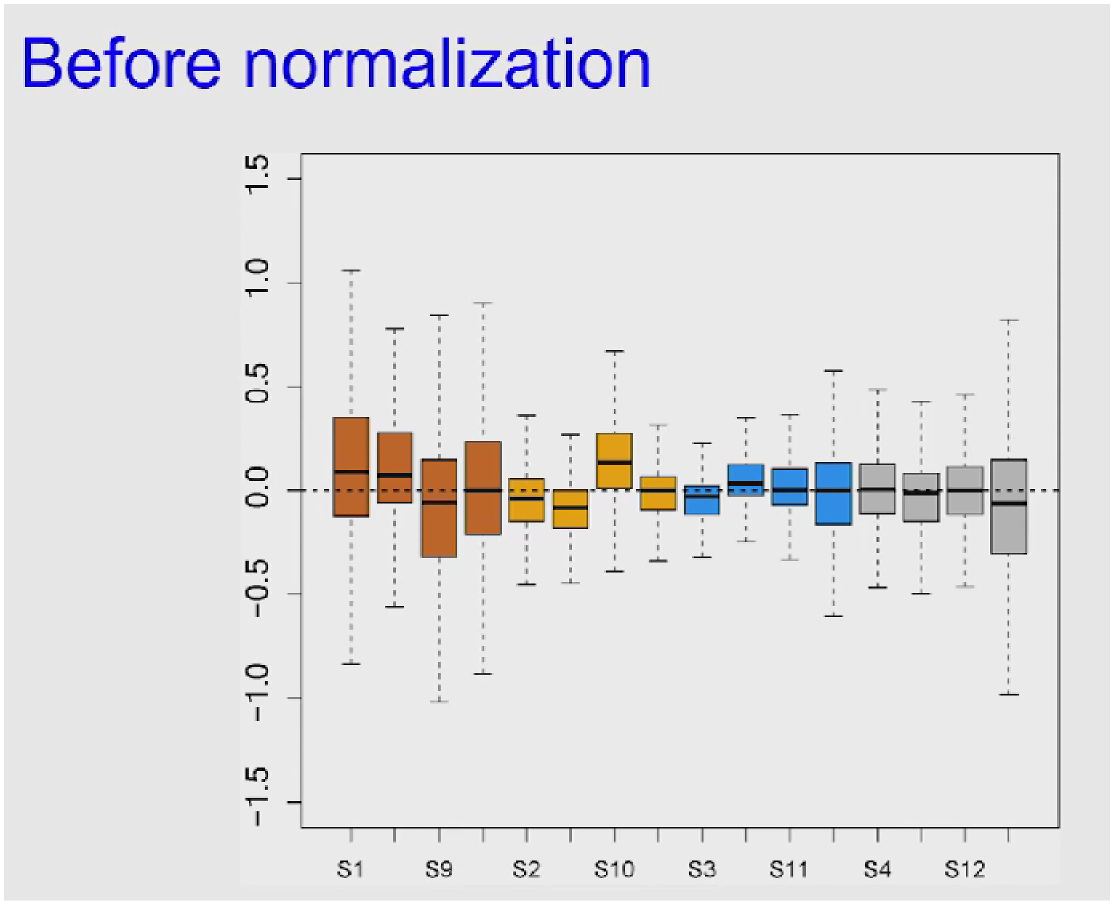
So, again, before normalization, this is the distribution. These are the distributions for the 16 samples that we have, and this is after TMM normalization. You can see this right in the distribution; the medians become identical, and again, the values kind of come close to each other, right? So, we have seen this right. So, just to again give you something to think about, right here is the rock on data in our case. So, you can think about how you proceed with this TMM normalization with this data set. We will not do this, of course. It does not really make sense to do this on a very small data set because, again, you cannot throw out extreme values, but you can at least think about the steps yourself, right? So, for example, if we choose a reference sample, we can calculate these M and A values, and then, after discarding those extreme genes, we can then calculate this weighted mean, etcetera. So, you can think about the steps with a very simple example, and you will get this idea right behind all those complex formulas, etcetera, that we used. So, we will talk about the next normalization, and this is quite different from what we have discussed so far, ok? So, this normalization method uses something called spike in controls. So, what are these spikes in controls? So, spike-in is an external RNA molecule that is added to RNA-seq samples. So, these are not part of your sample, right? You do not isolate them from the tissue or cells, right? These are synthetic molecules, data, and external molecules that we add to our samples, ok? The question is, why do we add them? So, again, there is a consortium that actually has designed these spikes in control; these are synthetic RNA molecules, and kind of taking care that these are synthetic; they do not show any similarity with RNA molecules that are present in organisms, etcetera. So, you will see this term ERCC spike in controls or ERCC spike in mix quite often, and you understand what this actually means. So, this is usually a mix of synthetic RNA molecules of different sizes, and these are also mixed in different numbers. This is not a single molecule; you have many molecules like that, and they are of different sizes lengths, and they are mixed in different numbers. So, just to see like how they are actually how they actually come out after you do the RNA sequencing experiment ok. So, how does it work actually how does this method work ok. So, what we do in this case is add an equal amount of spike to each sample. So, if you have 16 samples we add this equal amount exactly equal amount of the spike in to these samples right.

So, there is no biological variation here, ok, because we are adding this manually right after the RNA extraction. After we have purified the RNA, we add this, and then we do the same steps right for the library preparation, sequencing, etcetera. So, any variation that we see in the read counts right that that will arise purely due to technical variation; there is no biological variation here, ok? It will purely come from the experimental steps that we are following. Using this control, we can now normalize all sources of technical variation using a generalized linear model. So, we can set a generalized linear model, and researchers have done that. There are packages, and we will see these things later on when you actually do the hands-on. We will actually do this process right, and we can remove these unwanted technical variations.

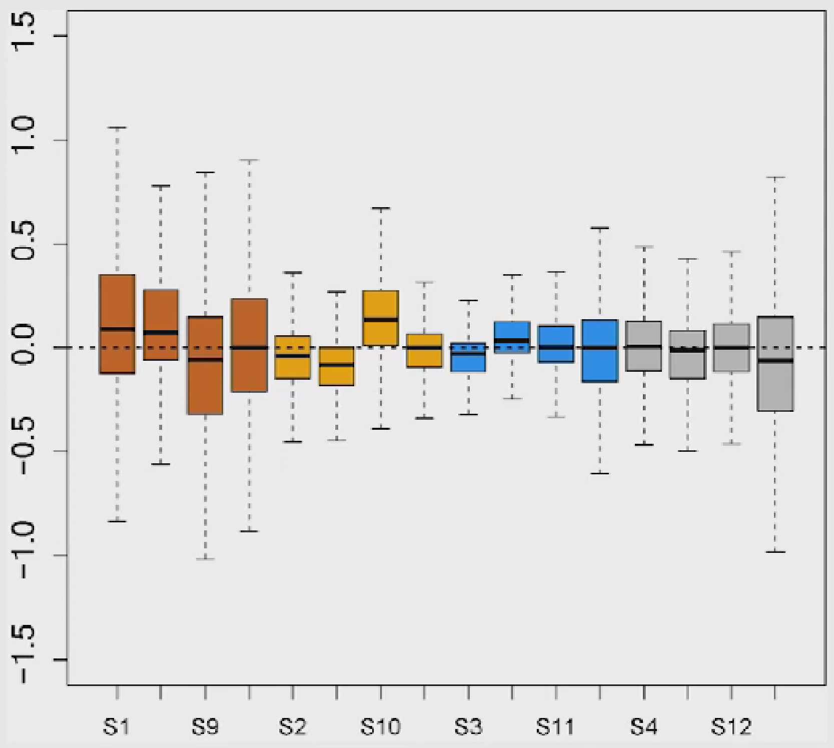


How does it work?

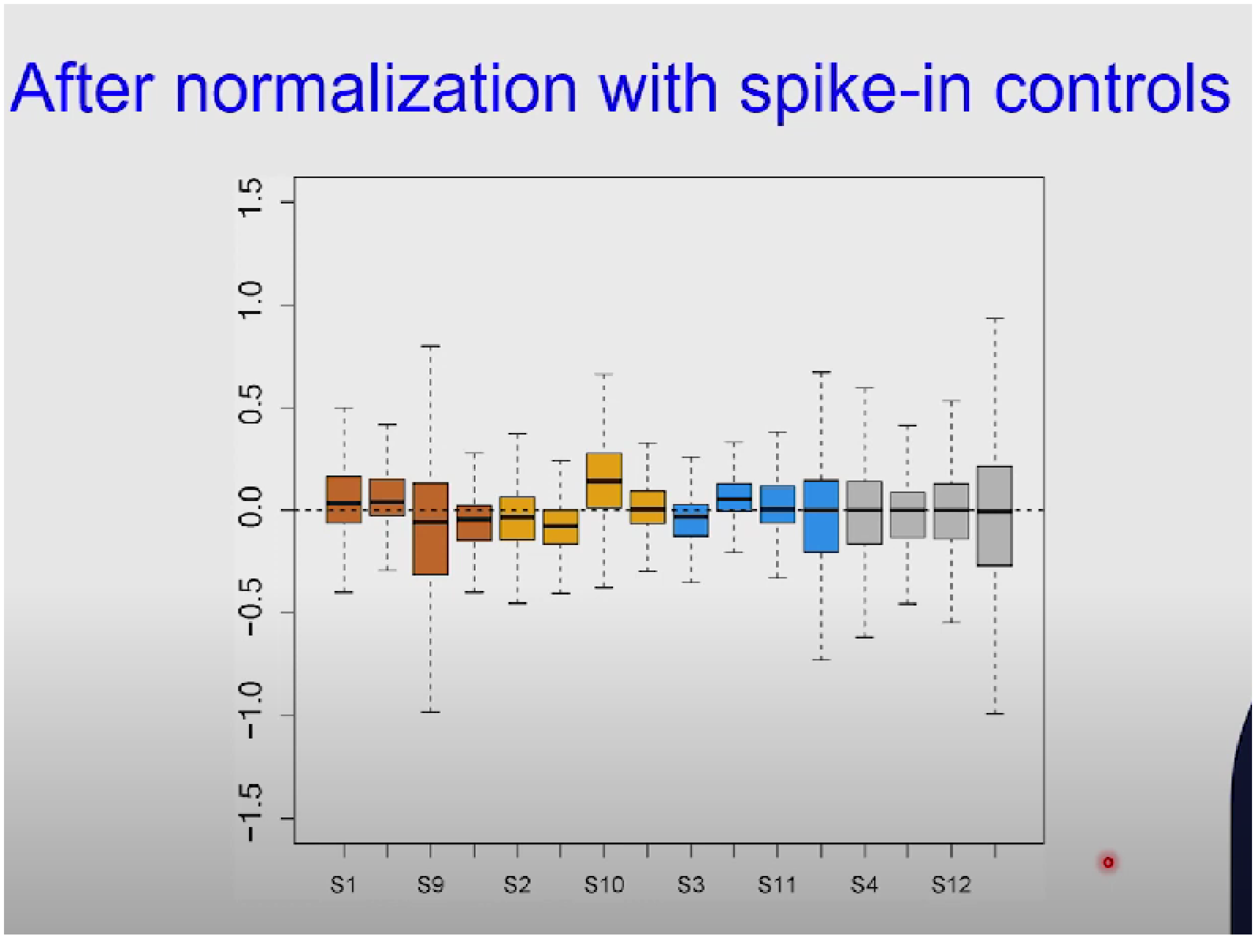
- Equal amount of spike-in controls are added to each sample
- Variation in read counts of these controls arises due to technical variation
- Normalization for all sources of variation using generalized linear model



Before normalization

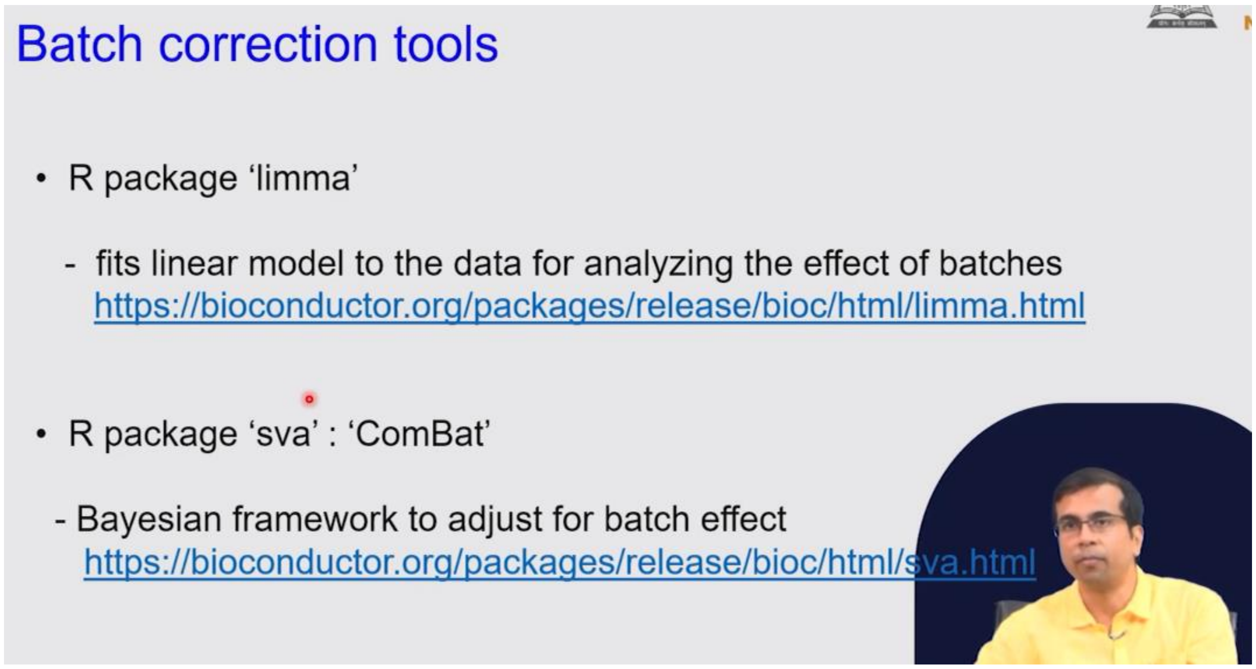


So, I will just show this data again because this data set had this ERCC spike in controls. So, you can see this here, ok? Again, this is the raw data. You will see the RLE plot right again for all the samples, and this data had ERCC spikes. You will be able to see that if you see the raw count data, then these names of ERCC spike ins will come out right, and after normalization is spike in controls, this is what we get, ok?


After normalization with spike-in controls

And you can see this is actually very different from what we got earlier across all different types of normalization. We still see, for example, that in this sample, they look a bit like outliers, but they do not show a very similar distribution across this data set. So, that may actually be the case, right? Because there was some sort of variation that happened during the experiment that actually led to this kind of situation. So, the biggest question is, which normalization method should you choose? We have discussed a lot of these methods, and the question is which one you would choose. Again, there is no unique answer here; it depends on your application and what you want to do. If you want to do this within sample comparison, of course, then you have to choose a method that does this within sample normalization.

But if you want to do this between sample comparisons only, then you have to choose a proper method that will be able to handle this RNA composition bias across samples and also look at differences in other factors. So, it can model all these technical variations. So, you would have to choose something like that, but if you want to do both, then you have to probably combine some of these methods one after another. So, what I will do is not go into too many details about this, but this has been kind of searched around, and so, here are some papers and some references that you can explore to better understand these issues. So, there are some papers that actually compare different normalization methods for different applications, different sets, and different types of analysis, and they found out different things. So, you can go into these references and explore them. So, one final thing that I wanted to discuss is something called batch correction. So, what happens in this when you want to combine different data sets of the same experiments ok. So, when you want to combine these different data sets right they can show different read counts expression levels of same genes and this is something that we very often see ok. And because these experiments are done on different time points maybe years apart or in different labs you tend to get different results because the experimental conditions are slightly different etcetera the people who are doing the experiment they are also different right it also varies a lot based on the person who is doing the experiment ok. And again as I mentioned this is mostly experimental variation right also the some of the conditions may have changed and which leads to this difference in expression levels.



Batch correction tools

- R package 'limma'
  - fits linear model to the data for analyzing the effect of batches https://bioconductor.org/packages/release/bioc/html/limma.html
- R package 'sva' : 'ComBat'
  - Bayesian framework to adjust for batch effect https://bioconductor.org/packages/release/bioc/html/sva.html

So, you can actually correct this using these tools. These are two tools I will mention right now. I will not go into all the details, but the first one is part of the lImma package in R. So, it feeds linear models to the data for analyzing the effect of batches. If you have batch data you can actually analyze this using this package lemma. This was actually designed for microarrays but can be extended to RNA-seq. And the second one is a function called ComBat, which again corrects for batch effects. It uses a Bayesian framework to adjust for batch effects. I will not go into details, but if you are interested, of course, you can explore this move. Here are the references for this class to summarize. We have talked about very sophisticated sample normalization methods that can address this RNA composition bias and, of course, other issues.

And we have talked about two methods that can do this very well: the median of ratios method and TMM normalization. We have also talked about the normalization method using RNA spikes in controls, and they enable estimation of technical variation. We have seen the normalization data right after normalization. And we have also talked about batch correction, which allows us to combine datasets and compare across datasets. Thank you.