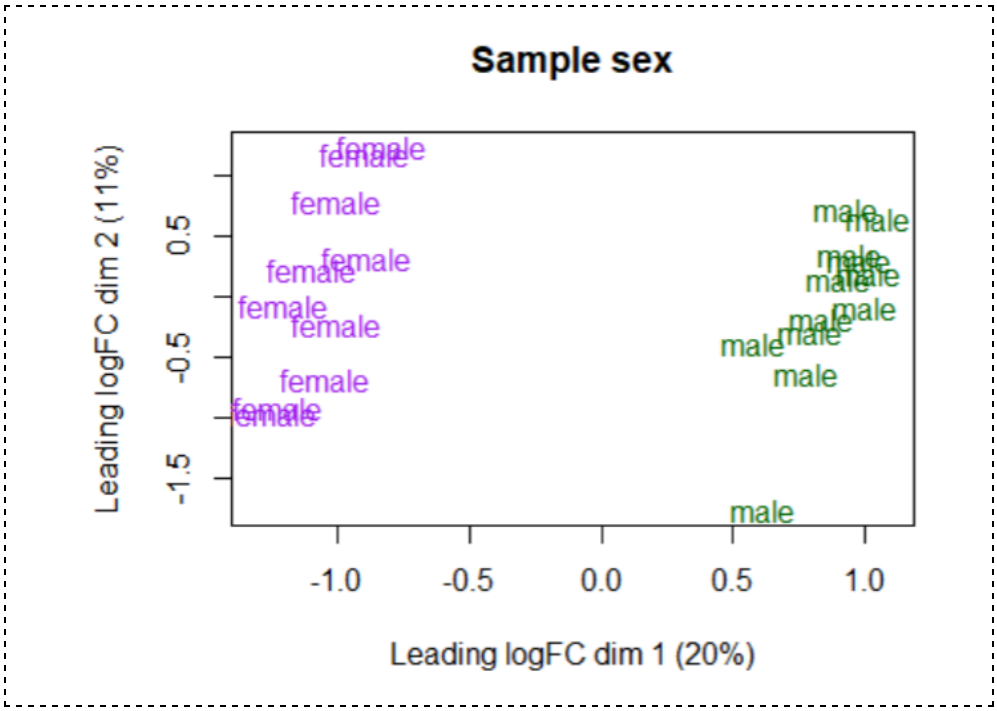
Analysis and Troubleshooting DE results for all trimmed data sets

As results from running the DE pipeline code on the 9 trimming data sets started rolling in, we saw that all the trimming parameters gave us 0 differentially expressed genes. Joelle was the first to share her trimming results with us and we began to think of reasons why we are seeing no DE genes. The following outline shows how I followed up on these discussions and how I solved the problem. Since we are already on Week 5 of 7, a corrected version of the script and the results are in our class shared drive in the female_vs_male directory. The results of the corrected script show that all the trimming parameters performed similarly to the untrimmed data set. We plan to discuss this in detail in our lab meeting so come with (or Slack) your questions.

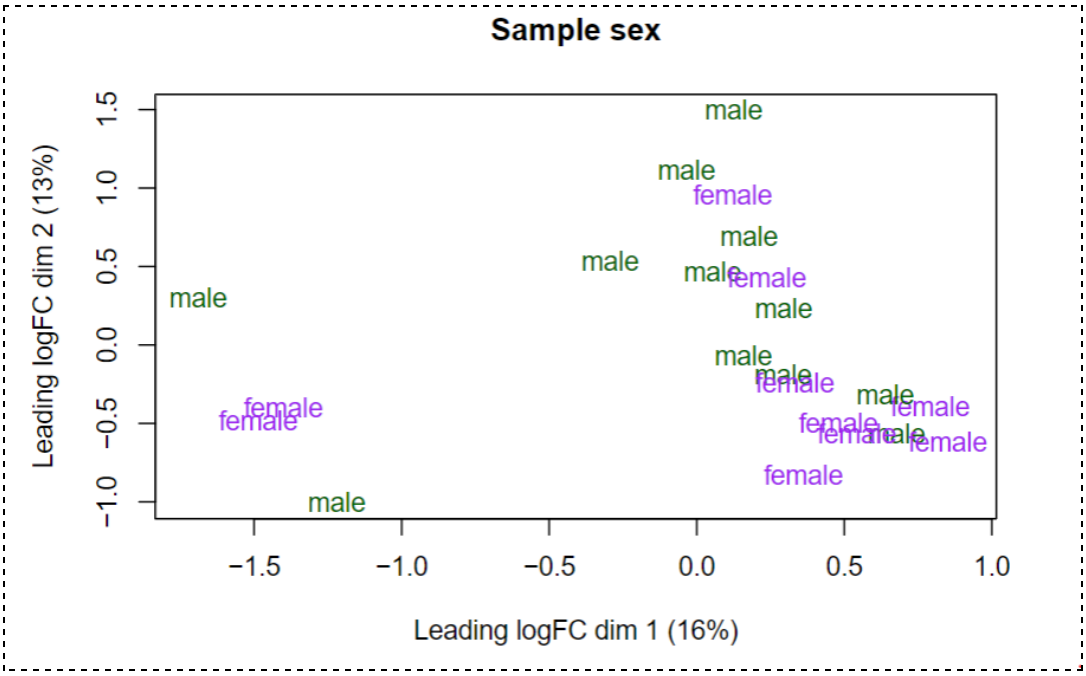
Inner monologue provided in blue italics in case that is helpful

1) **Reasons to be suspicious of the results we were getting running the DE pipeline on trimmed data**
    a) Untrimmed data gave separation of females and males on MDS plots but trimq_0_minlen_10 which is very little trimming (practically a negative control) did not at all separate by sex

MDS plot from untrimmed data report:



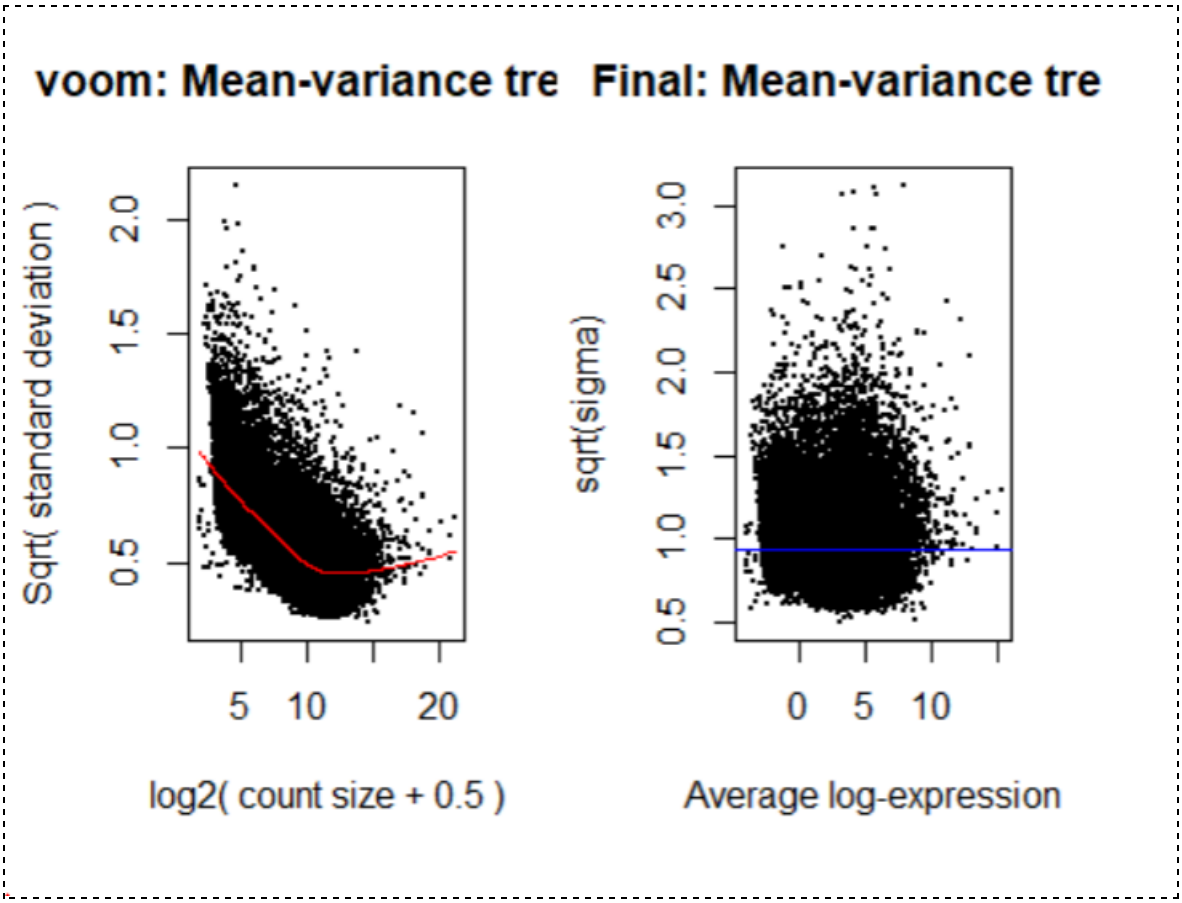
MDS plot from trimq_0_minlen_10 report:



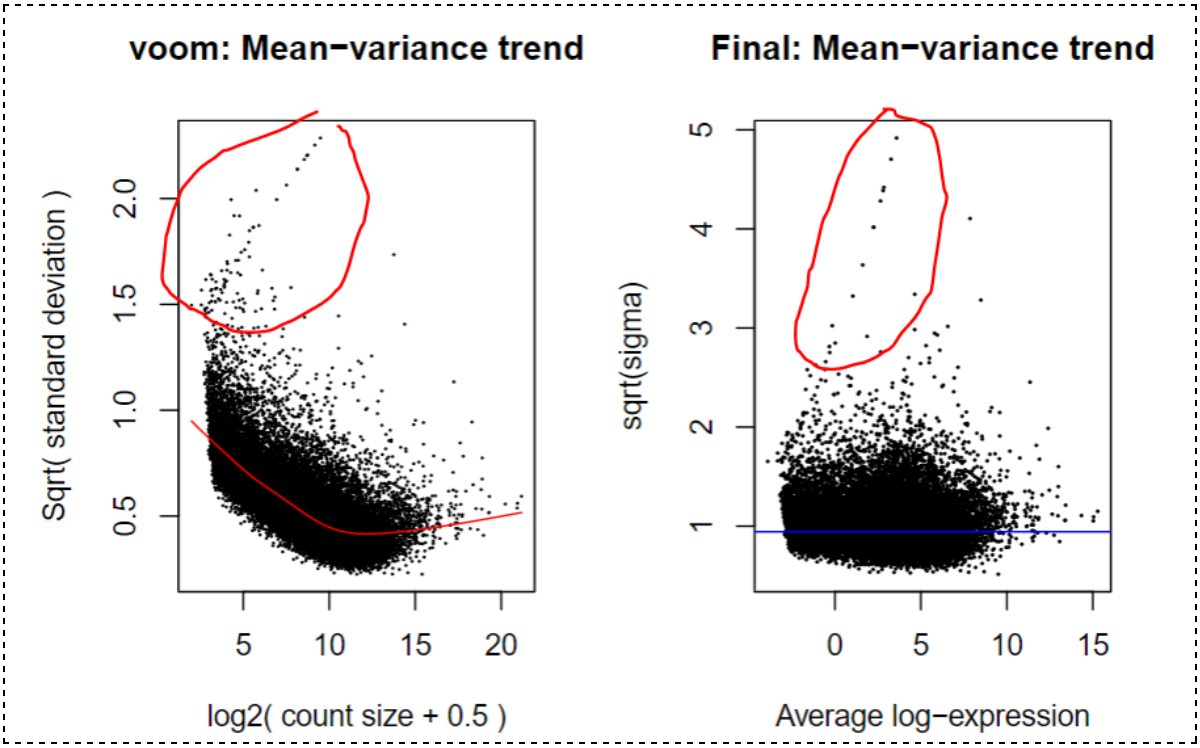
How could such a small change in trimming have such a drastic effect?

b) Voom plot showed a strange hook after transformation

Voom mean-variance plot from report for DE pipeline with untrimmed data:



Voom mean-variance from report of DE pipeline on trimq_0_minlen_10 data:



Something must be off with the data itself to give this strange line of points
Seems like some kind of artifact

c) All trimmed data sets were giving 0 differential expressed genes
If it was just one trimming parameter set, that might be believable but all 9?!
Something is fishy

2) Testing the data/code: Thinking of places there could be an error

a) Merging gene counts files – are we somehow not making the gene counts input file correctly?
b) Are we pulling in the right input file?
   i) Was there some kind of copy paste error from where the files were generated to the class shared drive?
   ii) Are the paths correct in the Rmd?
c) Are the sample labels or sample phenotypes wrong?
   i) This would affect both the MDS plot and the DE genes, so we should look carefully at this

3) Checking everything step by step

a) Merged count files
   i) Manually checked that the gene counts listed in the merged data file matches the gene count in the individual featureCount files for each sample
b) Pulling in the right input files

i) Regenerated the bbduk trimming commands used to generate the input data files and confirmed that they were working as expected (correctly passing in all combinations of 3 trimq values and 3 minlen values we were testing)

ii) Checked all the paths in the Rmds to make sure I had set things correctly

c) Sample labels and phenotypes (sex, rep)

i) Loaded trimmed data into differentially expressed pipeline

ii) Loaded pheno.csv file

iii) Clicked to view them in RStudio

(1) Noticed that the way sample names are given in the columns of the gene counts file do not match the ones in pheno.csv → this helped me figure out that they were not being cross referenced to get the sex/rep of each sample

Variable pheno after loading untrimmed gene counts file:

	sample	sex	batch	rep
1	OBG0158-2_untrimmed.XY	male	1	OBG0158
2	OBG0116-2_untrimmed.XY	male	1	OBG0116
3	OBG0166-2_untrimmed.XX	female	1	OBG0166
4	OBG0126-1_untrimmed.XY	male	1	OBG0126
5	OBG0132-1_untrimmed.XY	male	1	OBG0132
6	OBG0112-1_untrimmed.XY	male	1	OBG0112
7	OBG0130-1_untrimmed.XY	male	1	OBG0130

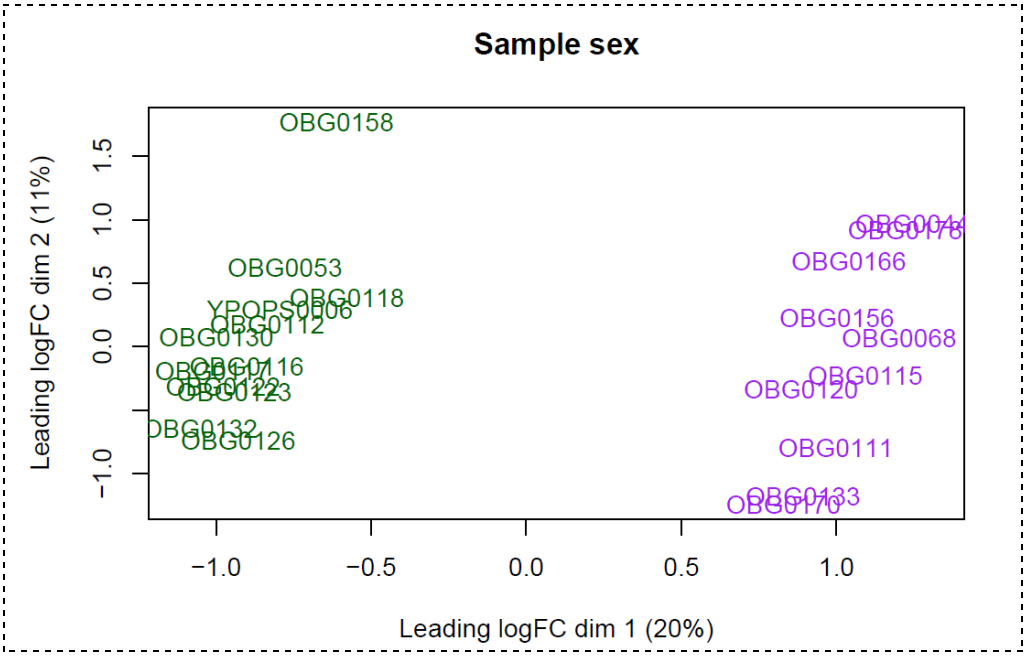
Variable counts after loading trimq_0_minlen_10 gene counts file:

	trimq_0_minlen_10/processed_bams/OBG0130-1_trimmed.XY.sort.mkdup.rdgrp.bam	trimq_0_minlen_10/processed_bams/YPOPS0006-2_trimmed.XY.sort.mkdup.rdgrp.bam	trimq_0_minlen_10/processed_bams/OBG0115-2_trimmed.XX.sort.mkdup.rdgrp.bam	trimq_0_2_trimm
DDX11L1	8	12	4	
WASH7P	383	421	371	
MIR6859-1	14	20	15	
MIR1302-2HG	0	0	1	

Uh-oh, the sample names don't match so DGEList can't be looking up the phenotype info like sex and rep. And the samples in the pheno table are in a different order! I think this is doing something wrong. Instead of trying to reorder or lookup the pheno table, let's just pull the phenotype info sex and rep (the ones we need) from the column headings of the counts table. That way no matter what order the samples are in in the counts table, we are good. This might not be enough of a fix if we want to use other phenotypes for something in the future, but this will do for the moment.

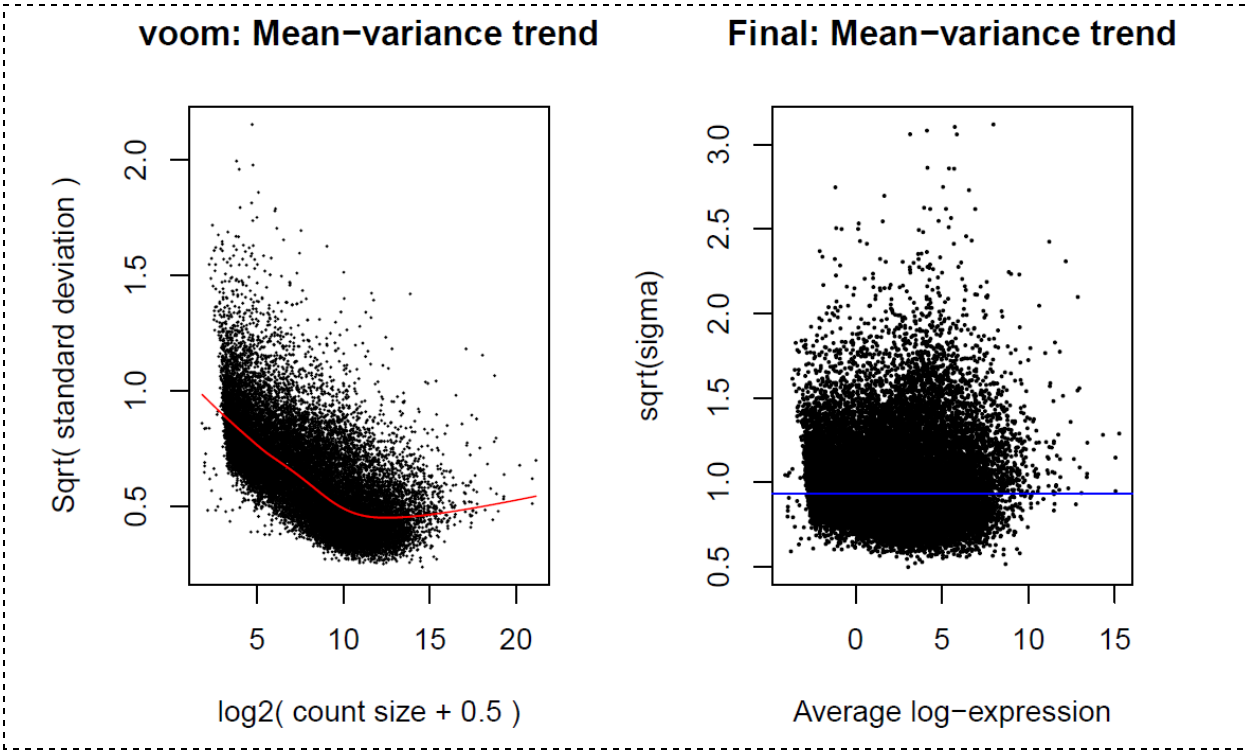
(2) Looking at the column headings of the merged gene counts files which are the path to the merged gene count file and seeing that students were having trouble setting paths to get the pheno.csv file, thought it would be simpler to simply extract the sex and replicate ID from the column headers of the counts files
    (a) Simpler (only one input file)
    (b) Know that the sample phenotype info would be correctly matched to sample (same order)
(3) Ran and saw results that made a lot more sense
    (a) MDS plots show separation by sex
    (b) No more hook in limma voom plot

MDS plot from trimq_0_minlen_10 after fix:



Ok this is more like it. Females (purple) and males (green) separate on the first axis with a similar variance explained as the untrimmed case (20%). I had changed the display to be the sample name instead to make it easier to tell which was which if any samples looked different.

Voom mean-variance plots for trimq_0_minlen_10 after fix:



Yep, that weird line of points is gone. This looks more correct.

iv) Implemented changes/improvements to code
   (1) Removed input of pheno.csv
   (2) Wrote code to get the sex and replicate ID from the column headings of gene counts files
   (3) Made it possible to run the untrimmed and the trimmed to make it so that we can run either data set using one script
   (4) Since a couple of students got confused between changing the input file and changing the prefix for the output files, changed the script to automatically create an output file label
   (5) Recreated all output for students
   (6) Posted to shared data directory on /data drive on Agave

4) Important lessons

a) Visualizing your data well, often, and consistently helps to ensure that you are doing what you think you are doing
   i) In this case, there was no R error indicating that something was not working properly
   ii) Programs will do exactly what you tell them to do as long as the input is in the right format → it's up to you to make sure that everything is working as expected

b) It's important to take time to think about whether your results make sense

i) Talking with others (especially your PI/supervisor) can help you to do that as they are looking at your results with fresh eyes
ii) Don't brush it off when someone says that they don't understand how something you did works
    (1) Take time to understand it yourself
    (2) Explain it as that's a great way to test your understanding
    (3) Great way to build connections with your labmates

c) When working in groups it can take a little bit of time to figure out when there is an error and where it comes from
    i) In this case, when one trimmed data set come back without differentially expressed genes we were a little concerned/confused, but when all of them come back that way that it felt like something could be wrong
    ii) If I had run all of these myself, I might have come to know something was off sooner, but the design of this project was to divide and conquer this research and there are many projects where time and resources might make it impractical to do everything yourself
    iii) All code has nuances that only the original programmer will understand, so computational research using computational tools will very likely involve a lot of troubleshooting and double checking

d) When there is an error with one part of the analysis, you have go back and rethink all the other analysis that dominos after that
    i) Each time you make a fix you have to think about how to explain that to anyone using your code and how best to aide in the recovery after that fix is made (documentation!!)
    ii) Think about any other applications that have used this code and make sure that the error is not present there as well