BIO720, Fall 2024, Individual Assignments

Ian Dworkin

2024-09-15

BIO720, Fall 2024, Individual Assignments

Due Date: Monday, September 23rd, 9AM. Send the rendered document of your assignment as a PDF, and email it to Dr. Golding, as per instructions that Dr. Golding discussed in class and available on the class github README.

Please code independently. Additionally, there is no use of large language models or other generative AI for this or any other assignments, unless specifically stated otherwise. This class provides the opportunity to learn how to think computationally, and to learn the practical skills of scientific computing. To develop the fundamentals of the skills requires you to work through the thought processes, problem-solving and practice of computation. Any evidence of group work OR use of generative AIs is cheating on assignments.

DataCamp modules

By the time this assignment is due, you should have already completed the following DataCamp modules (based on their individual due dates as assigned in DataCamp).

Introduction to R, **Intermediate R**, **Foundations of Git**, **Reporting with R Markdown**

```
sessionInfo()
```

Copy the information that is rendered as a PDF and email it to Dr. Golding (along with the information for question 4 below).

Note: Some people may (not sure why) not have the `rstudioapi` package already installed. If it gives you an error about this you can install it (and then re-run the code above). i.e.

```
#install.packages("rstudioapi")  # notice what you might need to change here!
```

The remainder of this assignment follows directly from our in-class activities with the RNAseq data set from the heads of *Drosophila melanogaster*, so you will need to use the information from those activities to complete your work. In particular loading in, properly setting up, and having the meta-data set up as we did by the end of class.

Note, even if the question just asks to report an answer, you are expected to show your code, so that the evaluator can re-run your analysis.

Q1

A. Write a *function* that calculates and outputs mean expression (using the `mean()` function) for a given data column. Provide the code of your named function so it can be run and checked by the person evaluating your assignment. Before trying your function on a full column (representing a single sample) of RNAseq count data, try it on only the first 10 rows (remember rows are genes) for the first biological sample (column) and report the mean.

B. Using a `for` loop, generate an object with the computed e mean read number for each of the RNAseq samples, based on your function.

C. Repeat the activity we did in 1B, but using a member of the apply-family of functions. There are several which could be used. Check whether your outputs from 1B and 1C are computationally identical to one another (and show how you checked it).

D. Repeat this computation again, but with a function (that already exists in base R) that automatically can work on columns or rows. Again, check whether the output objects (the vector of mean number of reads per sammple) are computationally identicial to one another.

Q2

Q2. Now write a new *function*, revising the first function (but give it a different, informative name) that calculates and can output mean expression (using the `mean()` function) for a given data column, but computes the mean of that column either on raw counts (as provided) or transformed to log_2 values of the counts. You should write this with a *logical* argument (i.e. setting a logical flag) that allows the user to choose whether to do this on the raw scale or log_2 scale (i.e. log_2 transform data, then calculate the mean). Make sure all functions you write for this assignment have this argument. We often use log_2 scale for RNAseq data!

A. Before trying your new function on a full column of RNAseq count data, try it on only the first 10 rows (remember rows are genes) for the first biological sample (columns) and output the mean. What happened? You may wish to examine the log_2 transformed values of these first 10 observations.

B. Based on the issue you discovered in Q2A, we need to make a change to our new function. In most real RNAseq analyses we have more sophisticated approaches to deal with this issue. What we will do for now is write our function so that it automatically adds one read count to each cell from the raw RNAseq count matrix, before using the log_2 transformation.

So for instance, the first gene (FBgn0031081) for the first sample has a raw count number of 2294. You will revise your function so that before it computes log2 of this value it will add one to it (2295) and then log2 transform it (and then computes the mean). The second row (gene FBgn0052826) has a raw read count of 0, so your function will add one to this (for a total value of 1) and do the log_2 transformation on this. This should "fix" the problem (computationally anyways).

C. Confirm that your new function works to generate the mean of read number, with the option (flag) of either doing the computation on the raw counts or log_2 transformed counts. Provide your code so the person evaluating your assignment can run it and check your work.

Q3

Q3. Evaluating sample-to-sample variation in sequencing depth is very important. We also want to get a sense of the average number of reads we have for each gene, across all of the samples. In other words, which genes on average have relatively low expression, moderate expression, or high expression. Modify what you have done so far so you compute the vector of mean expression (based on read counts) by gene. Please don't print all 17611 values to the screen! Instead just report the mean values for the final 10 genes (bottom 10 rows) in the data set. Again (always) share your code, not just the answer, so the person evaluating you can check your code works.

Q4

Q4. Using the `hist()` function, plot the distribution of values for the means of read counts by gene (this time for all 17611 genes). Using the help information in R, make sure that your axes are appropriately labelled. Show the plots (and code). Do this for both the mean expression based on raw reads and a second plot based on log2 transformed counts (again make sure to label plots so we know which is which).

Q5

Q5. One thing which you should get a sense of from looking at these plots is that there are many genes that have relatively low numbers of reads across virtually all of the samples. It is common practice to *pre-filter* your data and remove genes or transcripts that have very low abundance across almost all samples, prior to performing any formal analysis. We will do this now.

A. How many genes have a sum of zero raw counts (no reads in any sample)? Show your work.

B. How many genes have a sum of less than 100 reads across all samples? (This is less than one read per sample, as we have 142 unique samples in this dataset).

C. Deciding where to pre-filter takes some judgement. However, for now are going to decide to start by filtering out any genes that have less than 1000 total reads summed across all samples (still less than 10 reads per sample). Generate a new dataset of counts (with a new name) only with the genes that are being kept (a sum of more than 1000 reads across all samples). How many genes are retained post-filtering? (show your work)

D. This type of filtering is actually very naive, and can frequently fail. If a single sample has contamination (say from another nearby tissue), then that sample could have many 1000s of reads from transcripts from a particular gene, due to this contamination, while it is not present in any other sample. Alternatively, you may have a small number of samples (4-5) representing a particular treatment group that all show some expression of a gene, but it is completely absent in expression in all other samples. So we may wish to filter to account for some of these issues.

So for the final part of this question, perform pre-filtering based on the following critera. We will retain genes for further analysis if :

- At least 8 samples in the dataset show a "reasonable amount" of expression for a given gene, with a minimum of 30 reads in each of those 8 samples.
- and the sum of the expression counts of the gene across all samples is greater than 500 counts.

First write out your pseudo-code to show the logic of your approach. Then provide how many genes are retained (along with your code) by this filtering step.

Q6

Q6. Finish off by running `sessionInfo()` again. It is always good to include this information in every report!