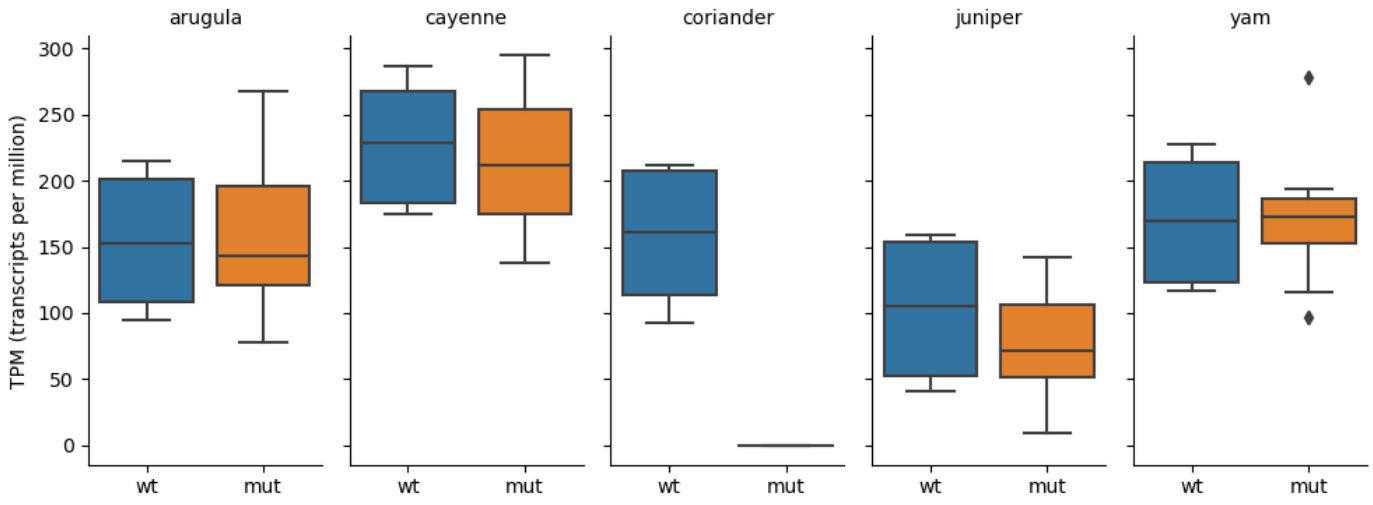
Problem Set 2: The adventure of the missing phenotype

The sand mouse gene *Coriander* is supposed to encode an important transcription factor. However, in a new paper from Watson *et al.* in the *Sand Mouse Journal*, an RNA-seq analysis of wild-type versus *Coriander* mutant sand mice shows that loss of *Coriander* function results in no significant effects on the mean expression of any sand mouse gene transcript (other than *Coriander* itself, of course).

the data

For example, Figure 2 from the paper shows mean expression levels and standard deviations for four sand mouse genes (*Arugula*, *Cayenne*, *Juniper*, and *Yam*), showing no expression differences. They also show that the *Coriander* mutant is indeed a knockout, null for RNA expression. The figure is based on RNA-seq measurements on ten biological replicates each for in wild type versus mutant sand mice.



You decide to dig a little deeper into Watson's data. Fortunately, Watson *et al.* seem to be practioners of reproducible computational science, because they've made their data available in their Supplementary Data Table 1: pset2-data.tbl.

Download the data file, and have a look. It's a column-justified, whitespace-delimited file. Looks like the table starts with some comment lines and commented column headers, followed by one line of RNA-seq expression levels (in units of TPM, transcripts per million) per gene, with 21 fields per line: the gene name, 10 replicates for wild type, and 10 replicates for mutant.

1. downsample the data set by reservoir sampling

Write a script that takes a random sample of 10 data (non-comment) lines, using the reservoir sampling algorithm.

2. look at outliers; validate the formatting

Write a script that checks over the whole file and:

- Find and print the data line that contains the maximum expression level.
- Find and print the data line that contains the minimum expression level.
- Find lines that don't have the right format. ("Right" is deliberately left vague. Validation is an art.)

3. clean the data

Write a script that removes the data lines that have the problems you just detected.

(hint: there are two problems in the data file, affecting 36 data lines. Your cleaning should leave you with 19,995 data lines for that many sand mouse genes.)

4. tidy the data

Write a script that converts the data table to a new file in tidy data format.

Both Pandas and Seaborn (which we're about to use next) are happiest with tidy data.

5. visualize the data

Write a script to read your tidy data file and visualize the distribution of the raw data points using *beeswarm plots* (also known as *swarm plots*) – that is, 20 points per gene, 10 biological replicates for wild type versus mutant – for any list of chosen sand mouse genes, and use it to display a plot of 10 randomly sampled genes. The Seaborn package has a good swarm plot function… which is why we're using Seaborn.

Also have a look at Seaborn's *catplot* function too, which gives you the ability to plot lots of genes in a "facet grid" in the same figure.

Instead of a swarm plot, another way to do it is a Seaborn **strip plot** with jitter=True.

6. your conclusions

What's going on in the *Coriander* mutant?

notes

- Tabular text files will often come with comma-separated values (CSV) or tab-delimited values, and sometimes with whitespace-delimited values. Column-justified, whitespace-delimited files are great because they're so human readable. A major theme of the course is that biological data are complex, artifacts will come at you from unexpected directions, and you have to *look* at the data, watching for problems that your scripts don't catch. The obvious drawback of whitespace-delimited files is that it limits your ability to have fields that consist of more than one "word".

hints

- Watson *et al.* say they validated the data file, to make sure it contains no bad data, and they've even provided the script they used. This is another example (like last week) showing how to open a file, read it one line at a time, and split each line into whitespace-delimited fields that you can look at one at a time. (You may also develop some worries about their data validation strategy.) This is a bare-bones starting point. Data science modules like Pandas give you more powerful tools for data file input and parsing.

- There's two reasonable ways to make the 'tidy' data set with different advantages and disadvantages. One form ('wide'), with each individual gene as a variable, is only mostly-tidy, but makes more sense for many kinds of analysis; another form ('long') with gene_name as a variable is fully tidy in Wickham's sense, and is easier to manipulate and plot in a Seaborn catplot, at least for this exercise. Hint: check out the Pandas melt method,

- Need an even more massive hint on using Seaborn to produce swarm plots and jittered strip plots? First, try your friend, Google; you should be able to find examples of people using (struggling to use) Seaborn to make swarm plots or related plots such as boxplots. But if you get stuck… Watson *et al.* have also provided the script they used to produce their boxplots, and a tidy data file that it will work on. This is a massive hint because it only takes a small tweak to make this produce your swarm plot – so use it only if you're really stuck and frustrated.