Bioinformatics Engineering Take-Home Challenge

Introduction

Feel free to use any language/libraries/frameworks that you are comfortable with.

As a bioinformatics engineer at Octant, an important responsibility is to build robust and reusable pipelines that process our unique experimental data. A common task is for a pipeline to pull the raw data coming off of a machine in our lab, join it with a file containing our experimental design, and then process the data for downstream use.

These pipelines typically contains several steps, including removing outliers, outputting QC metrics, fitting a model to data, and producing analysis-ready data. We have two major concerns for our internal pipelines:

- Make it easy to run and re-run the pipeline.
    - For example, we want to run the pipeline on every new experiment — several times a week.
- Ensure that we can swap in different pieces of code for different steps.
    - For example, we might want to update how we model dose responses from a LOESS to a log-logistic model

The Data

For this challenge, we would like you to build a simple pipeline that processes Luciferase experiments (see https://www.goldbio.com/articles/article/a-deep-dive-into-the-luciferase-assay-what-it-is-how-it-works-and-more for more information if you're interested). The inputs to the pipeline will be a CSV containing the experimental setup and a CSV containing the raw luciferase data.

The experimental setup will come in a CSV with the following columns:

```
Plate_ID: string
chem_ID: string
```

```
chem_M: float
cell_line: string
neg_control: string
pos_control: string
well: string
```

- Control chemicals are represented by a string starting with a capital C and containing a dash and a number: `C-##` . Note that control chemicals can show up in wells where they are NOT acting as the control.
    - `pos_control` and `neg_control` will both be control chemicals. `chem_ID` can be either a control chemical or an experimental chemical.
- Experimental chemicals are represented by a string starting with a capital O and containing a dash and a number: `O-##`
- Concentrations are NA for negative controls and positive floats otherwise
- Wells are represented as an alphanumeric string such as `A01` or `D23`.

The raw luciferase data will come in a CSV with the following columns:

```
plate: string
channel: string
row: int
col: int
value: int
```

- Both row and column are 1-indexed
- Value is guaranteed to be positive
- `channel` represents the wavelength of the light that the assay is detecting

Requirements

Please create a multi-step pipeline that performs the following tasks, outputting intermediate files at each stage. Each step in this pipeline should be its own executable:

1. Join experimental setup and raw data
    - Output: joined CSV

2. Calculate percentage of outliers in the negative controls and display what wells contain those outliers
    - Output: Quality Control (QC) report
3. Fit a LOESS model to the data
    - Output: Fitted model (someone else should be able to load the model using just the output)
4. Load the model in and produce plots
    - Output: Plots of the model fit and original data in a single document

We have attached an example of an experimental setup and luciferase output file to this email. Please use those files to help develop against, but realize that this pipeline should be general to any example that corresponds to the types listed above. We should be able to easily run this pipeline on a different set of data without changing any of your code.

Please make sure to return reasonable error codes and messages for any invalid data.