Homework Project 1 - 10 kilobase alignment

BIOINFO M122/M222

Due: Thursday April 11th, 2019, 11:59 pm

This short programming assignment is designed to help you get an understanding for the basics of sequence alignment. You can use any language for this project, but Python is strongly recommended, and you will receive starter code in Python. You will submit your response to https://cm124.herokuapp.com/upload as a `.zip` file.

Overview

In the first two programming assignments for this class, you will solve the computational problem of re-sequencing, which is the process of inferring a donor genome based on reads and a reference.

You are given a reference genome in FASTA format, and paired-end reads.

The first line of each file indicates which project the data relates to. In the reference file, the genome is written in order, 80 bases (A's, C's, G's, and T's) per line.

The paired end reads are generated from the unknown donor sequence, and 10 percent of the reads are generated randomly to mimic contamination with another genetic source. These reads are formatted as two 50 bp-long ends, which are separated by a 90-110 bp-long separator.

Your task is to map the reads to the reference genome and then determine where the reads indicate there is a difference (a variant). The different kinds of variants that may be found in the donor genome are explained at https://cm124.herokuapp.com/variant_doc

Starter Code

Starter code for all the class projects is available at https://github.com/eeskin/CM122_starter_code. It is strongly recommended that you use git for these programming assignments, and to set up your own account on github.com. The tutorials at http://try.github.io/ might come in handy. If you are completely unfamiliar with git and github, you can obtain a copy of the code by running

```
git clone https://github.com/eeskin/CM122_starter_code.git
```

This will create a folder named CM122_starter_code in your current directory.

Dependencies

The starter code for project 1uses the `numpy` library, which can be installed in your local environment using

```
conda install numpy
```

Tutorial

The starter code provided first aligns the paired end reads to the reference genome using `basic_aligner.py` and then calls SNPs using `basic_pileup.py`. Bear in mind that while this code works ok for the practice data set, it may not work so well for the real data.

You should download the practice and "for-credit" data from https://cm124.herokuapp.com/h1_data_files. If you want to follow the tutorial below, download and extract these files into the HP1 folder. Via command line that would look like this:

```
cd CM122_starter_code
cd HP1
wget http://studentdownloads.s3.amazonaws.com/practice_W_1.zip
wget http://studentdownloads.s3.amazonaws.com/hw1_W_2.zip
unzip practice_W_1.zip
unzip hw_1_W_2.zip
```

Obviously, you can just also open the browser and download and unzip the files using your file manager.

There are two scripts to look at: 1. `basic_aligner.py` takes in a reference genome, a set of reads and an output file and outputs the reads aligned to the reference genome into the output file 2. `basic_pileup.py` takes in aligned reads and an output file and outputs the SNPs called based on the aligned reads.

Running each of the above scripts with the `-h` option should be self explanatory, but here is an example of running them to create a file that can be submitted on the website for the practice data.

```
python basic_aligner.py -g practice_W_1/ref_practice_W_1_chr_1.txt \
-r practice_W_1/reads_practice_W_1_chr_1.txt -o test_aligner.txt
```

The step above will take ~30 minutes!

The next step uses the output file called `test_aligner.txt` generated by `basic_aligner.py`

```
python basic_pileup.py -a test_aligner.txt -o test_pileup.txt -t practice_W_1_chr_1
```

This will generate the file test_pile_up.txt.zip that you can submit on the website. It also generates a .txt file that you can look at. **Note that the -t parameter HAS to be practice_W_1_chr_1 when submitting the practice data and hw1_W_2_chr_1 when submitting the real assignment. This will let the online submission system know on which leaderboard to place you.**

Windows Users : Command line on Windows is a little tricky. There are several ways you can run the commands above, but probably the simplest one is to use the Anaconda prompt that came with your Anaconda installation. Another option is to set command line parameters using PyCharm: https://stackoverflow.com/questions/33102272/pycharm-and-sys-argv-arguments. Yet another option is to go into the code and hardcode the file names you want to use and remove the section that takes in command line arguments so you can run the code in Pycharm without adding any command line arguments. This last option is unfortunate and I hope you don't resort to it.

Read the content of HP1, and see if you can understand what it is doing. You can submit your results as many times as you want to achieve a passing score.

I/O Details

https://cm124.herokuapp.com/ans_file_doc should handle most of your questions on reading and writing output.

Sequence Alignment

The trivial sequence alignment algorithm is to "slide" the read along the reference genome. This is written simply as a for loop:

```
... get reads as input ...

for read in reads:
    for i in range(len(genome) - len(read)):
        mismatches = [0 if genome[i + j] == read[j] else 1 for j in range(len(read))]
        n_mismatches = sum(mismatches)
        if n_mismatches < 3:
            ...
            add this to the output
            ...
            break

.... construct consensus sequence ...
```

The statement above has some useful Python tricks; instead of iterating only over indices, Python allows you to iterate over objects. If you store your reads in an array called `reads`, you can get each one using the word `read`, rather than using many different indices.

The statement `mismatches = [0 if genome[i + j] == read[j] else 1 for j in range(len(read))` is called a list comprehension, which borrows from functional programming. Essentially, this code is the same as:

```
output = []
for j in range(len(read)):
    if genome[i + j] == read[j]:
        output.append(0)
    else:
        output.append(1)
```

Questions to consider

The genome from which the reads are generated has not only SNPs, but insertions, deletions, and repeated sequences that are not present in the reference.

What will these non-SNP mutations look like when you try to align them to the genome? Try writing it out on a piece of paper.

More generally, what is the "signature" of a SNP mismatch in the consensus sequence? What is the "signature" of an insertion or deletion?

Grading

Remember to submit your solutions to https://cm124.herokuapp.com/upload as a `.zip` file. You can submit as many times as you want without penalty.

You will be graded on your performance on the test set, which can be found at https://cm124.herokuapp.com/h1_data_files and under week 1 on CCLE. You can also submit your solutions for the practice data to https://cm124.herokuapp.com/upload to see how your solution is performing.

Undergrads will get full credit with a score of 45 on SNPs, and no credit for a score of 25 or below. Grad students will get full credit with a score of 60 on SNPs, and no credit for a score of 40 or below.