Part 1 Demultiplexing | BIO 622

Jared Galloway

Part 1

1.

Filename	Read
$1294_{S}1_{L}008_{R}1_{0}01.fastq.gz$	Read 1
$1294_{S}1_{L}008_{R}2_{0}01.fastq.gz$	Index 1
$1294_{S}1_{L}008_{R}3_{0}01.fastq.gz$	Index 2
$1294_{S}1_{L}008_{R}4_{0}01.fastq.gz$	Read 2

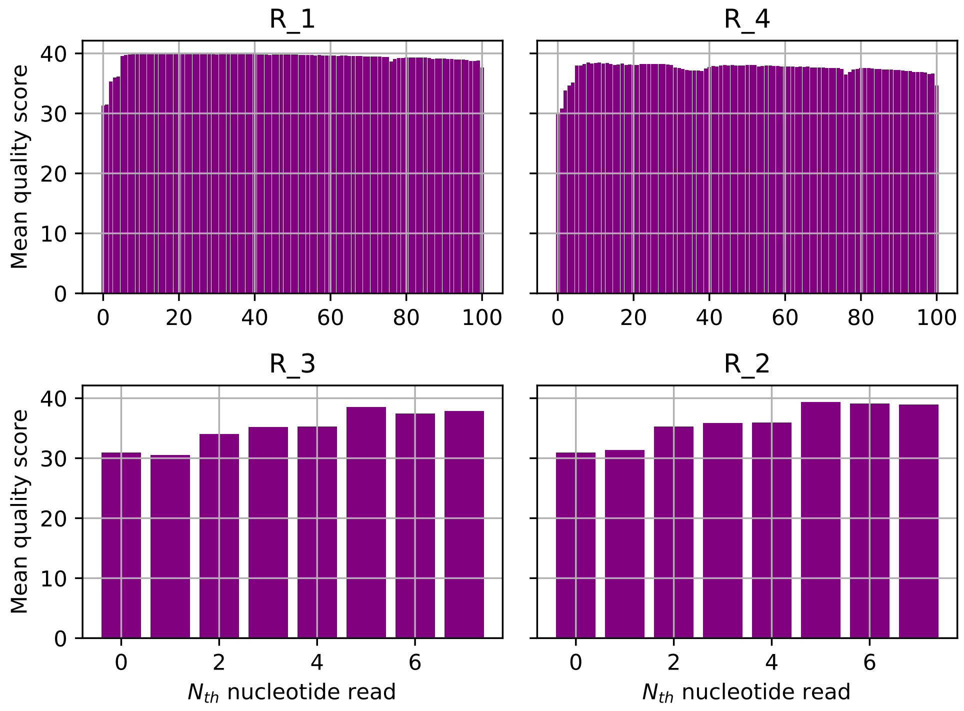

Figure 1: Distribution of average quality score R1

Due to the first two biologcal reads – for both R1 and R2 – are quite low $\approx < 36$, I would set the cutoff for 36, to make sure we are getting high quality biolocial reads.

To find the total number of N's within the two index reads, I used the following bash command

```
zcat emp_files/1294_S1_L008_R[2-3]* | awk 'NR%4==2' | grep N | wc -l > countN_Index.txt
```

Part 2

Problem: this problem is defined by seperating out reads on a flow cell based upon the reported barcode. This means that both index reads from each pair must match and be high quality in order to be succefully demultiplexed. If the Index pairs are *valid* (meaning above the quality cutoff, and exist in pre-defined set of indices) yet they do match for both pairs of a given read, then we put them in a hopped fastq file. for *not valid* indices, we put the read in an undefined fastq file.

Output: For this, I'm going to make two output fastq files for each valid index for each one of the paired end reads. And two files for undefined and two files for hopped.

Test Files: I'm going to make four test files which have three respective reads in each, one for each of the catagories mentioned above.

```
(bgmp_py3) n221 13:30 [demultiplexing-jgallowa07] $ ./P2.py -fq test_files/R[1-4]* -fi emp_files/indexes.txt
Number of hopped reads = 1 Number of undefined reads = 1 Number of valid reads = 1
```

Algorithm & High-Level functions : Found in psuedo_code/ folder