Part3

October 16, 2018

1 Demultiplexing, part 3

```
In [1]: %%html
        <! BEHAVE YOURSELVES, TABLES !>
        <style>
            table {
                display: inline-block
            }
        </style
```

```
<IPython.core.display.HTML object>
```

1.0.1 Results by index

index name	# read pairs	% of good read pairs	% of all read pairs
A1	31767724	7.448	4.770
A3	62925596	1.656	1.061
A5	22417604	3.414	2.186
A11	69309924	6.761	4.331
A12	15410740	2.409	1.543
B1	32472024	3.489	2.235
B2	29657244	3.186	2.041
B3	139877596	15.03	9.627
B4	35315996	3.795	2.431
B7	44732500	4.807	3.079
B9	35478748	3.812	2.442
C1	26342088	2.831	1.813
C3	20253760	2.176	1.394
C4	305408144	32.82	21.02
C7	16764380	1.801	1.154
C9	42512680	4.568	2.926
TOTAL GOOD	930646748		64.05
BAD	522340192		35.95
TOTAL	1452986940		

1.0.2 Results by outcome

Note that index paris are singly categorized by the first attribute (in descending order) that is true

outcome	# read pairs	% read pairs
mean index quality < 15	2671281	0.7353
'N' in index	4163786	1.146
invalid index	123326319	33.95
hopped (mismatched) indices	423662	0.1105
good index pairs	232661687	64.05
TOTAL	363246735	99.99

1.0.3 With quality cutoff vs. without

I was curious as to the cost/benefit breakdown for checking score quality. Even if the mean quality scores are low, if the two reads agree, we probably have the right index. Additionally, we're already checking for 'N' in the sequence -- will that catch most of the low-quality scores?

method	# good pairs	% good pairs	# bad pairs	% bad pairs
cutoff = 15	232661687	64.05	130585048	35.95
no cutoff	232701897	64.06	130544838	35.94

That's only a difference of 40210 read pairs, or ~0.01107% of the total reads. Removing the quality filtering also cut the total run time from 128 minutes to 108 minutes, a savings of 20 minutes or roughly 15.6% over the version.

1.0.4 Repo summary

- SLURM script: *demult.srun*
- Python script: *demult.py*
- Unit test inputs: *UT_index1.fastq.gz, UT_index2.fastq.gz, UT_read1.fastq.gz, UT_read2.fastq.gz*
- Unit test outputs: in *UT_outputs.tar.gz*