Tools for improving the quality of the data

**Cutadapt:** (Python script that you run at command line):

https://cutadapt.readthedocs.org/

- Cuts low quality bases from the 3’end of the reads
- Removes contamination
- Remove low quality reads
- Remove polyA tails
- Demultiplex: split reads over multiple fastq files based on barcodes
How cutadapt trims low quality bases

3‘ trimming of bases with quality threshold 20

G A T T T T A G T A A A A T T T A A T
12 19 40 36 37 12 33 34 24 26 12 14 18 15 23 3

Subtract threshold from quality scores

G A T T T T A G T A A A A T T T A A T
-8 -1 20 16 17 -8 13 14 4 6 -8 -6 -2 -5 3 -17

Calculate sum starting from 3‘ end
Stop when sum becomes positive

G A T T T T A G T A A A A T T T A A T
2 -11-25-29-35-27-21-19-14-17
How cutadapt trims low quality bases

Cutadapt needs to know:

ASCII encoding
  Default: +33
  --quality-base=64

Minimum quality to keep
  -q
  3 – 10 – 15 – 20

Which ends to trim
  Default: 3’
  5’+3’: -q 10,10
  5’ 3’
How cutadapt detects adapter contamination

Semi-global alignment of adapter to reads

Global alignment: full length

AGTCAGTGCAG

AG–C–T––––T–C

Semi-global: full length but gaps at ends are not penalized better when sizes differ

AGTCAG–TGCAG

–––AGCTTC–––
How cutadapt detects adapter contamination

Cutadapt needs to know:

Where adapter is ligated

- a adapter at 3’end

penalize gaps at 5’end of read
How cutadapt detects adapter contamination

Cutadapt needs to know:
Where adapter is ligated - b adapter at both sides
How cutadapt detects adapter contamination

Cutadapt needs to know:

**Adapter sequence(s)**

**Fastq file with reads**

**Name of output file**

Single end:

```bash
cutadapt -a AACCGG -a TAATTG in.fastq > out.fastq

cutadapt -a AACCGG -o out.fastq in.fastq
```

Paired end:

```bash
cutadapt -a fwadap -A revadap -o out1.fastq -p out2.fastq

reads1.fastq reads2.fastq
```
How cutadapt detects adapter contamination

Semi-global alignment of adapter to reads

---adapter
seqadap---

---adapter
seqadecter

---adapter
seqadap---

---longadapter
seqlongadacter

Alignment: 7 nt Error rate: 0.7 = 0
Alignment: 7 nt Error rate: 0.7 = 0
Alignment: 4 nt Error rate: 0.4 = 0
Alignment: 11 nt Error rate: 1.1 = 1

Cutadapt needs to know:

How many errors to allow

- \( e \) maximum error rate
default: 0.1 (\( \approx 10\% \) of alignment)
How cutadapt avoids errors

Semi-global alignment of adapter to reads

---adapter
seqadapter

--adapter
seqadap

---adapter
seqad

Adapter or read?

Cutadapt needs to know:

Minimum length of alignment - o
Minimum overlap default: 3
Cutadapt can discard reads with N bases

Cutadapt needs to know:

Maximum number of N’s  --max-n
Reads that are too short after trimming will map everywhere

By default, empty and short reads are kept and appear in the output

Cutadapt needs to know:
The minimum read length to keep \(-m\) cutadapt always discards both reads of a pair
Cutadapt report

Sequence: 'ACGTACGTACGTTAGCTAGC'; Length: 20; Trimmed: 2402 times.

No. of allowed errors: 0-9 bp: 0; 10-19 bp: 1; 20 bp: 2

Adapter is 20 nt
Error rate is 0.1
Matches up to a length of 9 bp are allowed to have no errors
Matches of lengths 10-19 bp are allowed to have 1 error
Matches of length 20 can have 2 errors

Overview of removed sequences

<table>
<thead>
<tr>
<th>length</th>
<th>count</th>
<th>expect</th>
<th>max.err</th>
<th>error counts</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>140</td>
<td>156.2</td>
<td>0</td>
<td>140</td>
</tr>
</tbody>
</table>

3 bases were removed in 140 reads
Randomly, you would expect this to occur 156.2 times
Maximum number of errors at that match length is 0
Additional steps that cutadapt can do

- Demultiplex
- Nextseq: additional step removing G’s at the ends of your reads
  --nextseq-trim
- Removing adapters with barcodes: adapNNNNNNNter