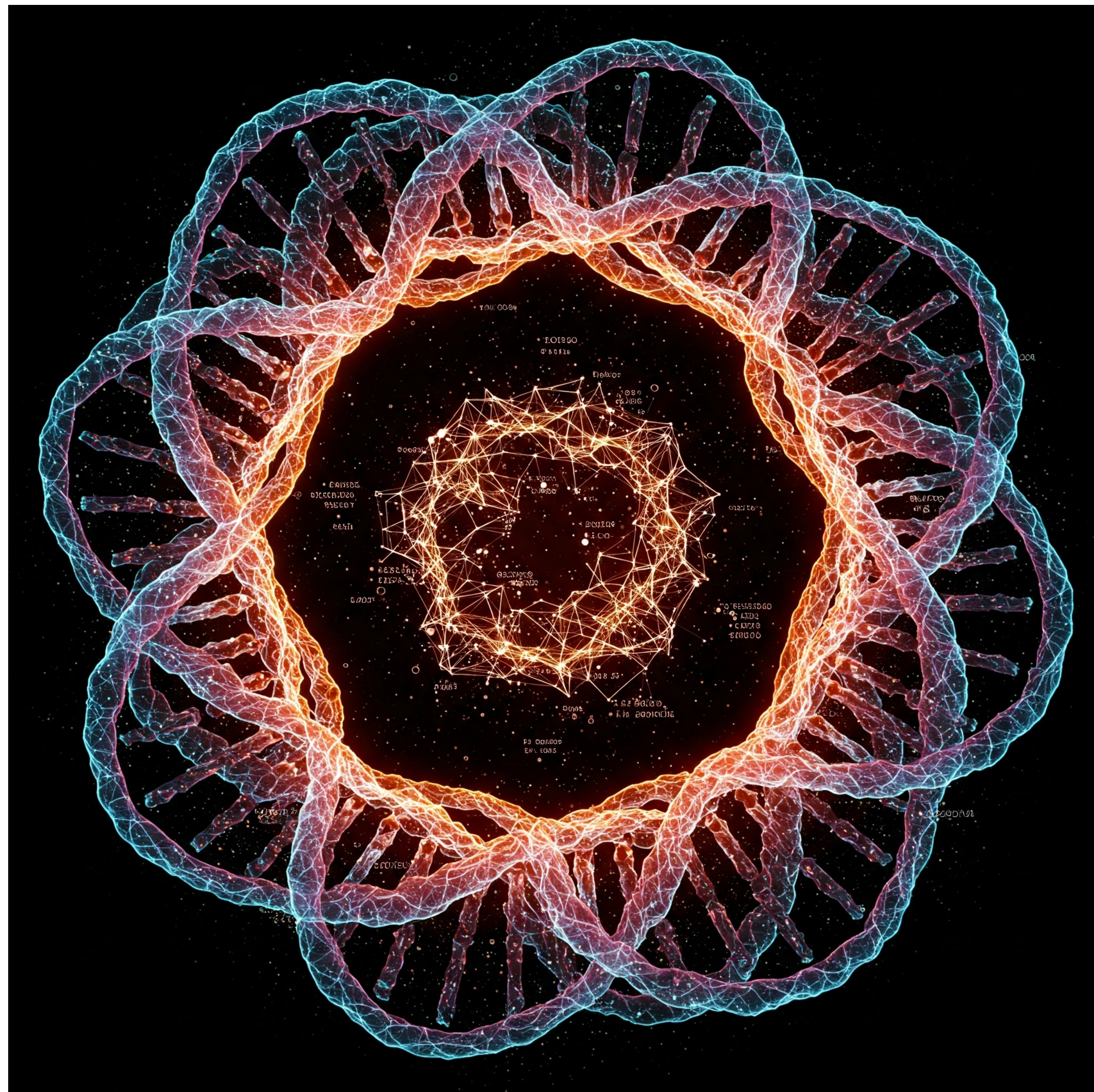
Sequence format and pre-processing

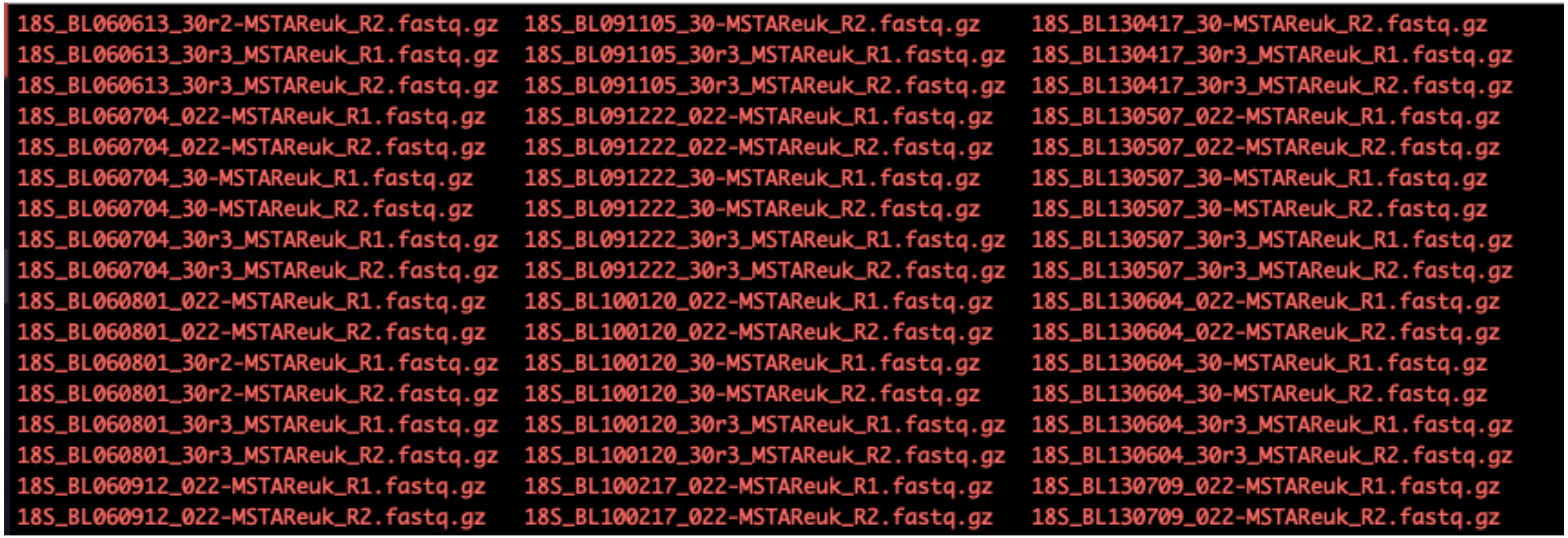
Ramiro Logares, ICM, Barcelona



Illumina short reads

- We receive files in fastq format from the sequencing center
- Two files per sample, 1 forward (R1) and 1 reverse (R2)
- Normally, reads overlap
- Depending on the library preparation, all reads are in the same direction (5’-3’) or their directions are mixed in both R1 and R2
- We usually work with gzipped fastq files to save space

How files look like when they are received



```
18S_BL060613_30r2-MSTAReuk_R2.fastq.gz  18S_BL091105_30-MSTAReuk_R2.fastq.gz    18S_BL130417_30-MSTAReuk_R2.fastq.gz
18S_BL060613_30r3_MSTAReuk_R1.fastq.gz  18S_BL091105_30r3_MSTAReuk_R1.fastq.gz  18S_BL130417_30r3_MSTAReuk_R1.fastq.gz
18S_BL060613_30r3_MSTAReuk_R2.fastq.gz  18S_BL091105_30r3_MSTAReuk_R2.fastq.gz  18S_BL130417_30r3_MSTAReuk_R2.fastq.gz
18S_BL060704_022-MSTAReuk_R1.fastq.gz   18S_BL091222_022-MSTAReuk_R1.fastq.gz   18S_BL130507_022-MSTAReuk_R1.fastq.gz
18S_BL060704_022-MSTAReuk_R2.fastq.gz   18S_BL091222_022-MSTAReuk_R2.fastq.gz   18S_BL130507_022-MSTAReuk_R2.fastq.gz
18S_BL060704_30-MSTAReuk_R1.fastq.gz    18S_BL091222_30-MSTAReuk_R1.fastq.gz    18S_BL130507_30-MSTAReuk_R1.fastq.gz
18S_BL060704_30-MSTAReuk_R2.fastq.gz    18S_BL091222_30-MSTAReuk_R2.fastq.gz    18S_BL130507_30-MSTAReuk_R2.fastq.gz
18S_BL060704_30r3_MSTAReuk_R1.fastq.gz  18S_BL091222_30r3_MSTAReuk_R1.fastq.gz  18S_BL130507_30r3_MSTAReuk_R1.fastq.gz
18S_BL060704_30r3_MSTAReuk_R2.fastq.gz  18S_BL091222_30r3_MSTAReuk_R2.fastq.gz  18S_BL130507_30r3_MSTAReuk_R2.fastq.gz
18S_BL060801_022-MSTAReuk_R1.fastq.gz   18S_BL100120_022-MSTAReuk_R1.fastq.gz   18S_BL130604_022-MSTAReuk_R1.fastq.gz
18S_BL060801_022-MSTAReuk_R2.fastq.gz   18S_BL100120_022-MSTAReuk_R2.fastq.gz   18S_BL130604_022-MSTAReuk_R2.fastq.gz
18S_BL060801_30r2-MSTAReuk_R1.fastq.gz  18S_BL100120_30-MSTAReuk_R1.fastq.gz    18S_BL130604_30-MSTAReuk_R1.fastq.gz
18S_BL060801_30r2-MSTAReuk_R2.fastq.gz  18S_BL100120_30-MSTAReuk_R2.fastq.gz    18S_BL130604_30-MSTAReuk_R2.fastq.gz
18S_BL060801_30r3_MSTAReuk_R1.fastq.gz  18S_BL100120_30r3_MSTAReuk_R1.fastq.gz  18S_BL130604_30r3_MSTAReuk_R1.fastq.gz
18S_BL060801_30r3_MSTAReuk_R2.fastq.gz  18S_BL100120_30r3_MSTAReuk_R2.fastq.gz  18S_BL130604_30r3_MSTAReuk_R2.fastq.gz
18S_BL060912_022-MSTAReuk_R1.fastq.gz   18S_BL100217_022-MSTAReuk_R1.fastq.gz   18S_BL130709_022-MSTAReuk_R1.fastq.gz
18S_BL060912_022-MSTAReuk_R2.fastq.gz   18S_BL100217_022-MSTAReuk_R2.fastq.gz   18S_BL130709_022-MSTAReuk_R2.fastq.gz
```

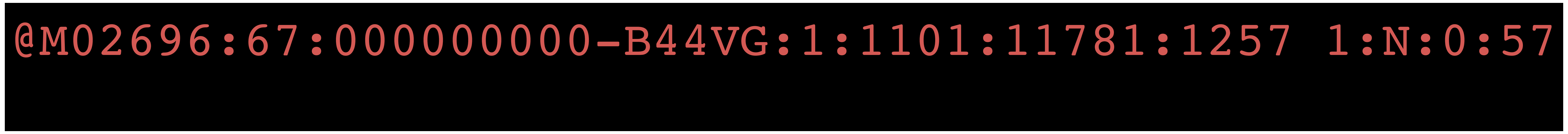
fastq format

- Four sequences per line

1. @sequence.ID
2. ACTGACTGACTG # nucleotide sequence
3. + (separator)
4. Quality scores (Phred +33: normally 0-41)

Important information in the sequence ID

```
@M02696:67:000000000-B44VG:1:1101:11781:1257 1:N:0:57
CCAGCAGCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTTAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATTCTTCATGCCACTTTTATACTGA
TTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
+
CCCCCGGGGFG@CGG;FDEFFGEFGGGG9E@CFGCGGGEFG<EFGFEFGGGGFGGGEG<FC@@@6@F8@FCGAFFFFF,6C6EC@FCFGGGGGGGGGGCFGGDF:CFFFAFF,BCE<CFFEFF7F8?,CF<EBCF,AFDGFAFF<
9@BEFEG?FC9,CE<FD?A7CGEG:FDFG,3A;,CDFGGGFF,=CF,6,6BFGF,6+4@EEGGGG7>EC?FGGF@FCGED8CFFGG79D9CCF<?C4713?FFFCDE
@M02696:67:000000000-B44VG:1:1101:8695:1347 1:N:0:57
CCAGCACCCGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATTCTTCATGCCACTTTTATACTGA
TTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
+
CCCCCGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFGGGGGGGGGGGFGGGGGGGGGGGGGGGGGGGFGGGFGEFGGGGGGFGGGGGGGGGGGGGGGGGGGGGGGGGFFGGFFGGGGGGFGGGFGGDGGGGGGDFGGGGGGGGGGGGC
FFGGGGGGGGG9FGGGGGGGGGGGGFFGGGFGGGFFGGGGGGGGGGFFFGGG6CEEFGDGDEGEEDFFFCFFE@CFF;ACD7:F8CFGGFCFFFFFFGCA*::
@M02696:67:000000000-B44VG:1:1101:22691:1423 1:N:0:57
CCAGCACCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATCCGTGATTCTTCATGCCACTTTTATACTGA
TTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
+
CCCCCGGGGGGGGGGEGFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFFGGGGGGGGGGGGGC@D8,CFGGCFGGGGEF<FF9FGF6EGGFG<FG<BFGFGGEGCDGGGGGGGGGG?FGGGFFGCGGGGGGGGGGFG9@FA,EFF<
FFGGFFGGGGG,AFGGDGGDFFEACDFFDE@DDFAFC<;FD>=@FCFGGCDGG6ED;AEGGCEDG5FFFCF,9=CFFF)=,,=,BFFCGF4CD76:C>9ACE>
@M02696:67:000000000-B44VG:1:1101:19965:1620 1:N:0:57
CCAGCAGCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATTCTTCATGCCACTTTTATACTGA
TTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
+
CCCCCGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
FGGGGGGGGGGGGGGGGGGGGGGGGGGFGGGGGGGGGGGGFFFGFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFGGGGGGGGGGGFFDFFGGGFFFFF
@M02696:67:000000000-B44VG:1:1101:17122:1672 1:N:0:57
CCAGCAGCCGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATTCTTCATGCCACTTTTATACTGA
TTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
+
CCCCCGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGEGGGGGGGGGGEGGGGGGGGGGGFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
GGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFCFFGFFEGGGGGGGGGGGGGDGCGGGGGCFGGGGFFGGFGGGDFFGGGGGGGGGFGGGGFGFF
@M02696:67:000000000-B44VG:1:1101:21438:1779 1:N:0:57
CCAGCACCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATTCTTCATGCCACTTTTATACTGA
TTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
+
CCCCCGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
GGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGFFGGGFGGGGGFDCEEGGGGFGGGGGGD?FGGFGGGGFGGFGCEGGGFFFDGGGGFFGGGGFGED
```



```
@M02696:67:000000000-B44VG:1:1101:11781:1257 1:N:0:57
```

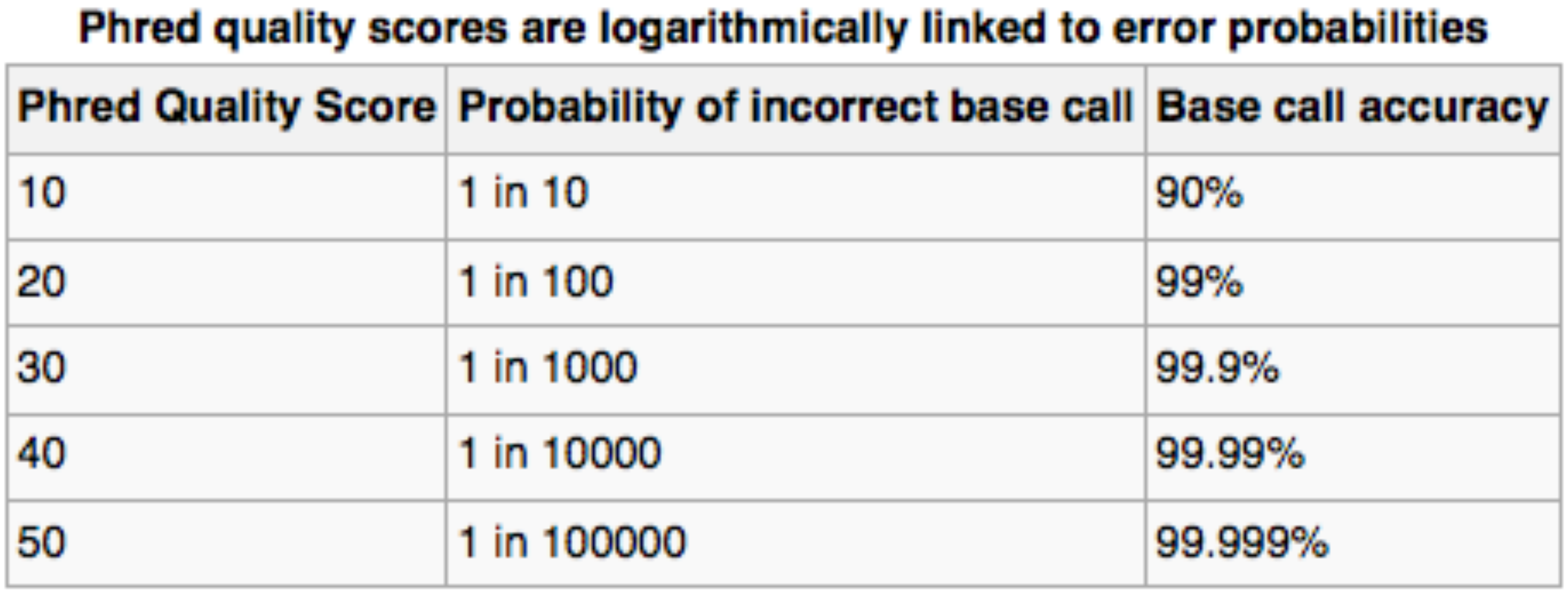
The first line, identifying the sequence, contains the following elements.

```
@<instrument>:<run number>:<flowcell ID>:<lane>:<tile>:<x-pos>:<y-pos>:<UMI> <read>:<is filtered>:<control number>:<index>
```

Table 1 FASTQ File Elements

Element	Requirements	Description
`@`	@	Each sequence identifier line starts with @.
`<instrument>`	Characters allowed: a–z, A–Z, 0–9 and underscore	Instrument ID.
`<run number>`	Numerical	Run number on instrument.
`<flowcell ID>`	Characters allowed: a–z, A–Z, 0–9	
`<lane>`	Numerical	Lane number.
`<tile>`	Numerical	Tile number.
`<x_pos>`	Numerical	X coordinate of cluster.
`<y_pos>`	Numerical	Y coordinate of cluster.
`<UMI>`	Restricted characters: A/T/G/C/N	Optional, appears when UMI is specified in sample sheet. UMI sequences for Read 1 and Read 2, seperated by a plus [+].
`<read>`	Numerical	Read number. 1 can be single read or Read 2 of paired-end.
`<is filtered>`	Y or N	Y if the read is filtered (did not pass), N otherwise.
`<control number>`	Numerical	0 when none of the control bits are on, otherwise it is an even number. On HiSeq X and NextSeq systems, control specification is not performed and this number is always 0.
`<index>`	Restricted characters: A/T/G/C/N	Index of the read.

Sanger Phred quality scores



Phred quality scores are logarithmically linked to error probabilities

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10000	99.99%
50	1 in 100000	99.999%

$$Q = -10 \log_{10} P$$

Q = Phred quality scores
P = base calling error probability

Calculating Phred scores

- To determine quality scores, Phred first calculates several parameters related to peak shape and peak resolution at each base

- Phred then uses these parameters to look up a corresponding quality score in lookup tables

- These lookup tables were generated from sequence traces where the correct sequence was known, and are hard coded in Phred; different lookup tables are used for different sequencing chemistries and machines

- Quality scores are encoded in ASCII
  (American Standard Code for Information Interchange)

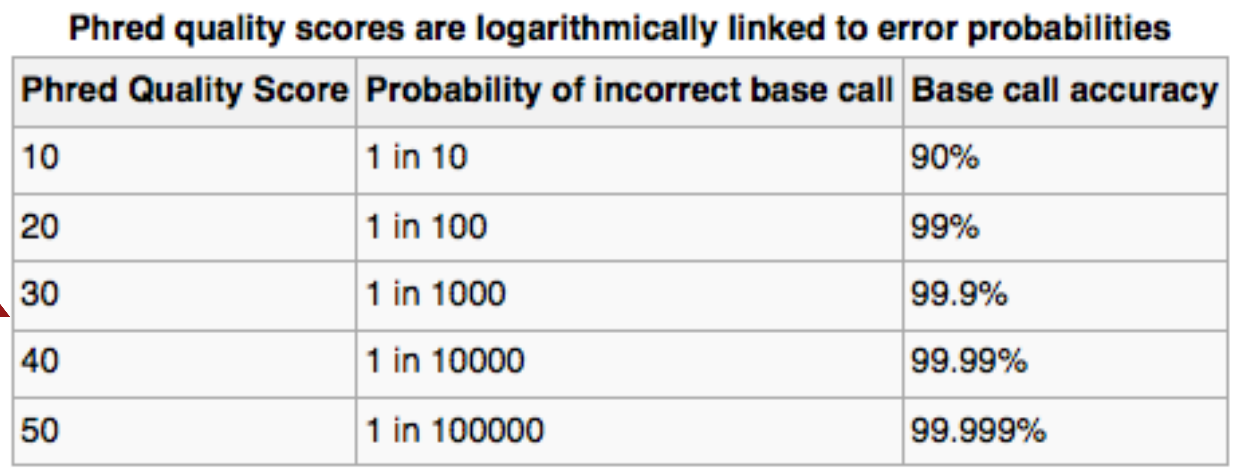
- They start in character 33 (Phred+33)

Quality

```
CCCCCGGGGFG@CGG;FDEFFGEFGGGG9E@CFGCGGGEFG<EFGFEFGGGGFGGGEG<FC@@@6@F8@FCGAFF
FFF,6C6EC@FCFGGGGGGGGGCFGGDF:CFFFAFF,BCE<CFFEFF7F8?,CF<EBCF,AFDGFAFF<
9@BEFEG?FC9,CE<FD?A7CGEG:FDFG,3A;,CDFGGGFF,=CF,6,6BFGF,6+4@EEGGGG7>EC?
FGGF@FCGED8CFFGG79D9CCF<?C4713?FFFCDE
```

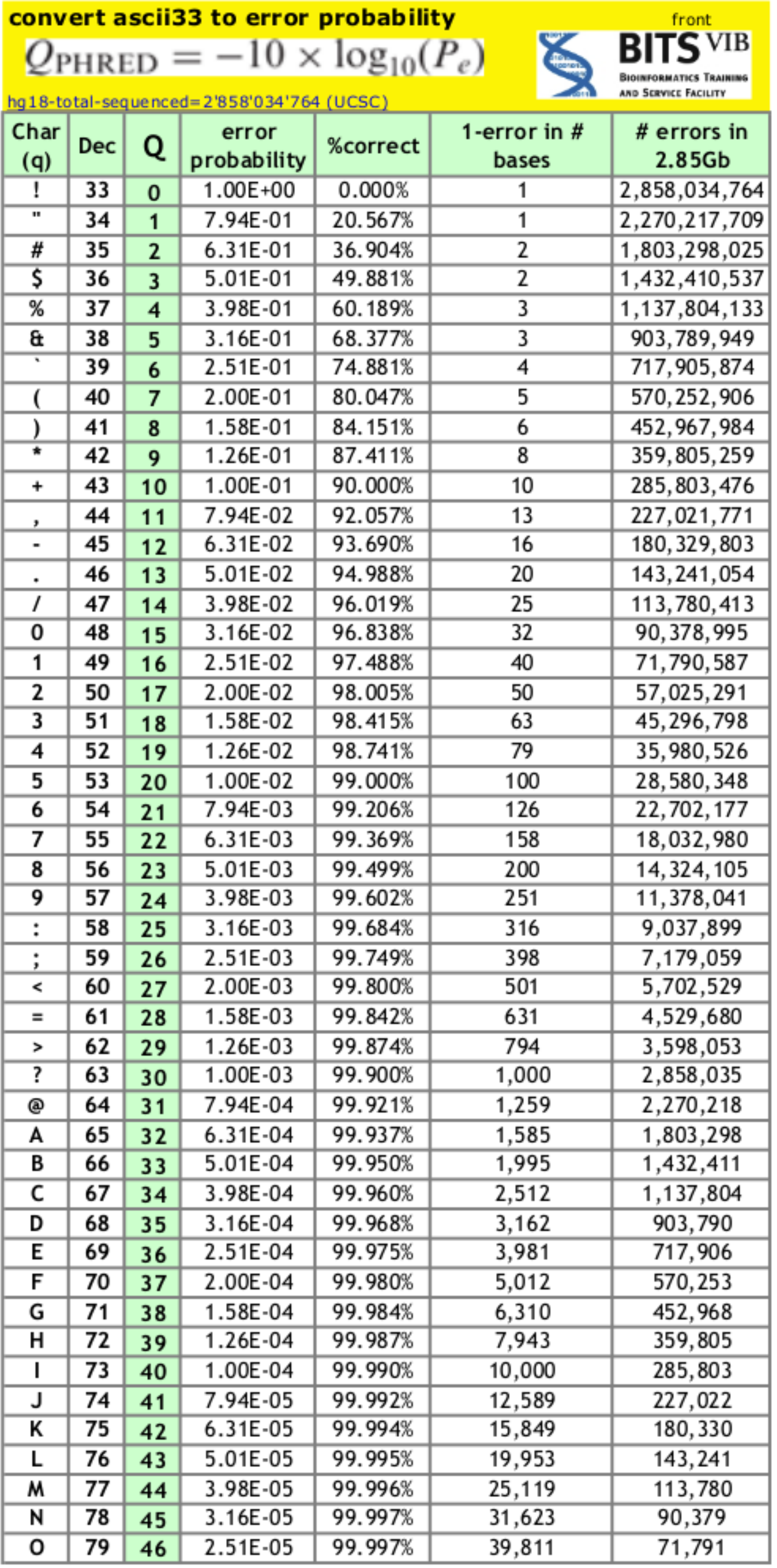
Q=ASCII code - 33

Example:
C = 67
Q = 67 -33 = 34



Phred quality scores are logarithmically linked to error probabilities

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10000	99.99%
50	1 in 100000	99.999%



convert ascii33 to error probability

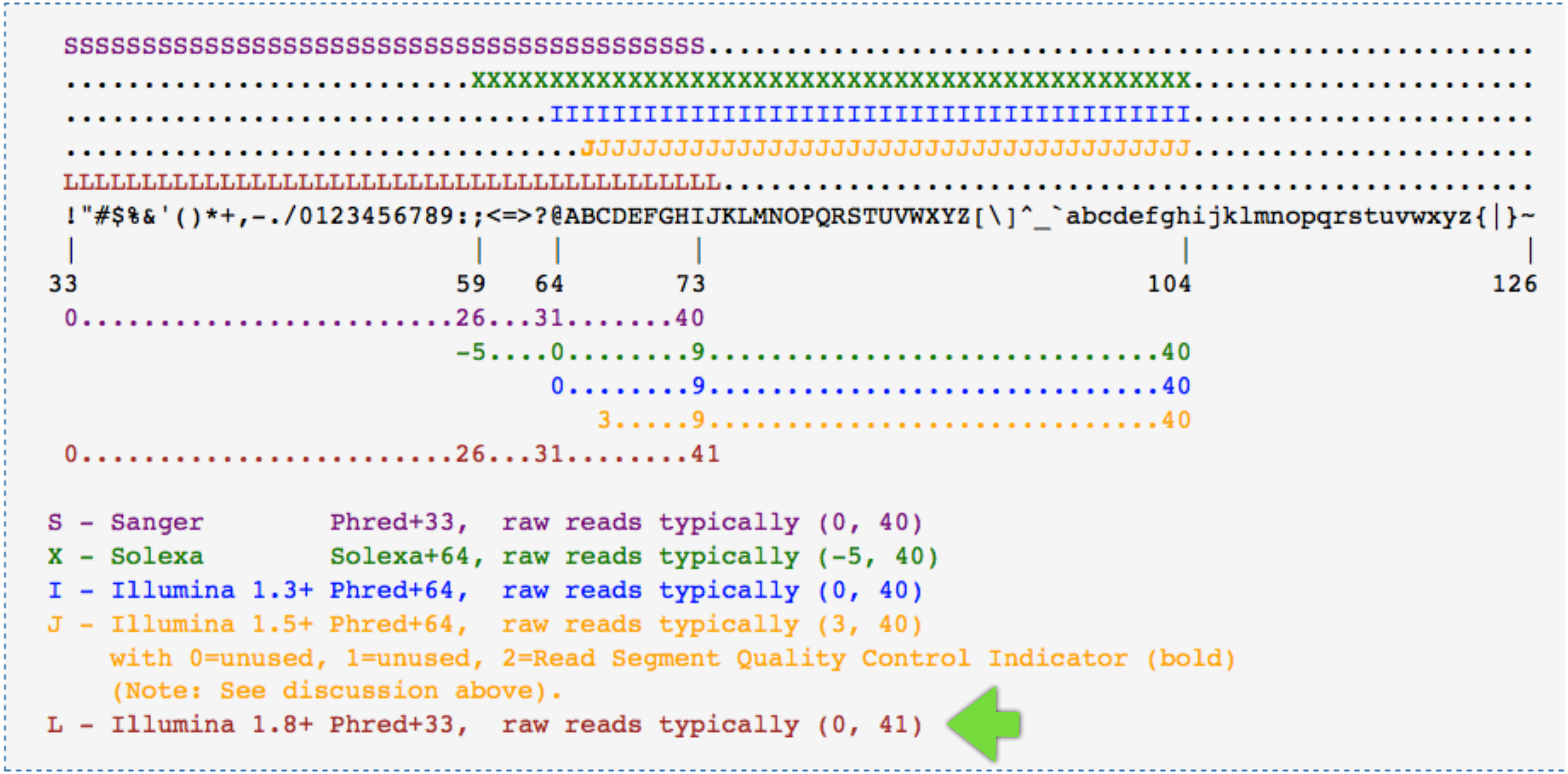
front

$$Q_{\mathrm{PHRED}} = -10 \times \log_{10}(P_e)$$

hg18-total-sequenced=2'858'034'764 (UCSC)

Char (q)	Dec	Q	error probability	%correct	1-error in # bases	# errors in 2.85Gb
!	33	0	1.00E+00	0.000%	1	2,858,034,764
"	34	1	7.94E-01	20.567%	1	2,270,217,709
#	35	2	6.31E-01	36.904%	2	1,803,298,025
$	36	3	5.01E-01	49.881%	2	1,432,410,537
%	37	4	3.98E-01	60.189%	3	1,137,804,133
&	38	5	3.16E-01	68.377%	3	903,789,949
`	39	6	2.51E-01	74.881%	4	717,905,874
(	40	7	2.00E-01	80.047%	5	570,252,906
)	41	8	1.58E-01	84.151%	6	452,967,984
*	42	9	1.26E-01	87.411%	8	359,805,259
+	43	10	1.00E-01	90.000%	10	285,803,476
,	44	11	7.94E-02	92.057%	13	227,021,771
-	45	12	6.31E-02	93.690%	16	180,329,803
.	46	13	5.01E-02	94.988%	20	143,241,054
/	47	14	3.98E-02	96.019%	25	113,780,413
0	48	15	3.16E-02	96.838%	32	90,378,995
1	49	16	2.51E-02	97.488%	40	71,790,587
2	50	17	2.00E-02	98.005%	50	57,025,291
3	51	18	1.58E-02	98.415%	63	45,296,798
4	52	19	1.26E-02	98.741%	79	35,980,526
5	53	20	1.00E-02	99.000%	100	28,580,348
6	54	21	7.94E-03	99.206%	126	22,702,177
7	55	22	6.31E-03	99.369%	158	18,032,980
8	56	23	5.01E-03	99.499%	200	14,324,105
9	57	24	3.98E-03	99.602%	251	11,378,041
:	58	25	3.16E-03	99.684%	316	9,037,899
;	59	26	2.51E-03	99.749%	398	7,179,059
<	60	27	2.00E-03	99.800%	501	5,702,529
=	61	28	1.58E-03	99.842%	631	4,529,680
>	62	29	1.26E-03	99.874%	794	3,598,053
?	63	30	1.00E-03	99.900%	1,000	2,858,035
@	64	31	7.94E-04	99.921%	1,259	2,270,218
A	65	32	6.31E-04	99.937%	1,585	1,803,298
B	66	33	5.01E-04	99.950%	1,995	1,432,411
C	67	34	3.98E-04	99.960%	2,512	1,137,804
D	68	35	3.16E-04	99.968%	3,162	903,790
E	69	36	2.51E-04	99.975%	3,981	717,906
F	70	37	2.00E-04	99.980%	5,012	570,253
G	71	38	1.58E-04	99.984%	6,310	452,968
H	72	39	1.26E-04	99.987%	7,943	359,805
I	73	40	1.00E-04	99.990%	10,000	285,803
J	74	41	7.94E-05	99.992%	12,589	227,022
K	75	42	6.31E-05	99.994%	15,849	180,330
L	76	43	5.01E-05	99.995%	19,953	143,241
M	77	44	3.98E-05	99.996%	25,119	113,780
N	78	45	3.16E-05	99.997%	31,623	90,379
O	79	46	2.51E-05	99.997%	39,811	71,791

Phred encoding in different sequencers



```
 SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................................
 ..........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX......................
 ...............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII......................
 .................................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ......................
 LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL....................................................
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
 |                         |    |        |                              |                     |
33                        59   64       73                            104                   126
 0........................26...31.......40
                          -5....0........9.............................40
                                0........9.............................40
                                   3.....9.............................40
 0........................26...31........41

S - Sanger        Phred+33,  raw reads typically (0, 40)
X - Solexa        Solexa+64, raw reads typically (-5, 40)
I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
    with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
    (Note: See discussion above).
L - Illumina 1.8+ Phred+33,  raw reads typically (0, 41)
```

Removing primers

- The sequences received from the sequencing center may contain primers used to amplify them
- Primers need to be removed as they normally contain ambiguous positions that can interfere with DADA2
- DADA2 assumes primers have been removed

Checking if sequences have primers in Unix

We know the primer sequence expected in the reads

Fw: CCAGCA[ACGT]C[ACGT]GCGGTAATTCC

Rv: ACTTTCGTTCTTGAT[AGCT][AGCT][AGCT]

Ambiguities are included to match sequences
We use zgrep to match the primer against the gzipped sequences

```
[rlogares@marbits raw]$ ls
BL100525E-MSTAReuk_R1.fastq.gz  BL100706E-MSTAReuk_R1.fastq.gz  BL100914E-MSTAReuk_R1.fastq.gz  primers_R1_in_reads
BL100525E-MSTAReuk_R2.fastq.gz  BL100706E-MSTAReuk_R2.fastq.gz  BL100914E-MSTAReuk_R2.fastq.gz  primers_R2_in_reads
BL100622E-MSTAReuk_R1.fastq.gz  BL100803E-MSTAReuk_R1.fastq.gz  clipping_primers.sh
BL100622E-MSTAReuk_R2.fastq.gz  BL100803E-MSTAReuk_R2.fastq.gz  cutadapt.o40252
```

```
[rlogares@marbits raw]$ zgrep -c --color  CCAGCA[ACGT]C[ACGT]GCGGTAATTCC BL100525E-MSTAReuk_R1.fastq.gz
23843
```

Forward primers

```
[rlogares@marbits raw]$ zgrep -c --color ACTTTCGTTCTTGAT[AGCT][AGCT][AGCT]  BL100525E-MSTAReuk_R2.fastq.gz
23856
```

Reverse primers

Let’s inspect the primer match visually

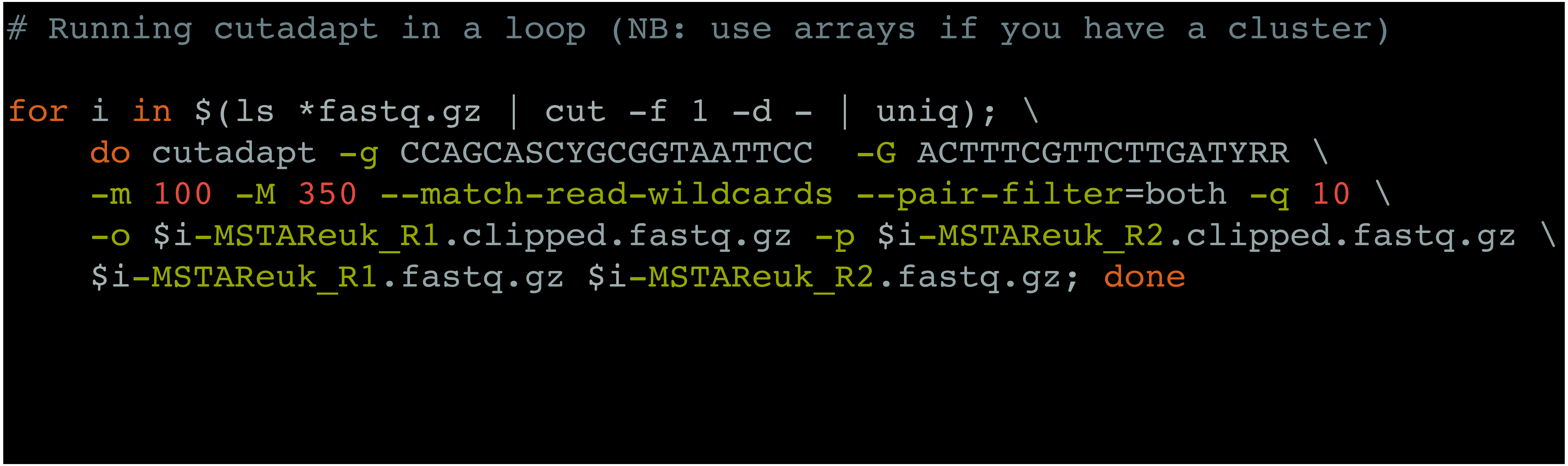
```
[rlogares@marbits raw]$ zgrep --color  CCAGCA[ACGT]C[ACGT]GCGGTAATTCC BL100525E-MSTAReuk_R1.fastq.gz
```

```
CCAGCACCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCACTATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
CCAGCACCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTATTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCCTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
CCAGCACCTGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAAAATAACGG
CCAGCACCCGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
CCAGCACCTGCGGTAATTCCGGCTCCTTCAGCCTGATGTAGAATTGTTGTAGTTAAAACGCTCGTATTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
CCAGCAGCCGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
CCAGCAGCCGCGGTAATTCCGGCTCCTTCAGCCTGAGGTAGAATTGTTGTAGTTAAAACGCTCGTAGTTGGATTTTGTAAGAGTTTTGTGTGTGTTGGTTGCGTATATATTCGTATATTCGTGATT
CTTCATGCCACTTTTATACTGATTGTGGATAATTTTCGGATTATTTGCAATATTACTGTGAGAAAAAGAGTGCGCTTAAGGGCGGCTTTATGCTAAGATCATTTAGCATGGAATAAACATAACGG
[rlogares@marbits raw]$
```

Forward primers

We use cutadapt to remove primers

- Program: https://cutadapt.readthedocs.io/en/stable/
- Runs in Unix (we will run this in Google Colab)
- Cutadapt will search for primers in R1 and R2 sequences and remove them
- It can also remove all sequences where primers have not been found



```
# Running cutadapt in a loop (NB: use arrays if you have a cluster)

for i in $(ls *fastq.gz | cut -f 1 -d - | uniq); \
    do cutadapt -g CCAGCASCYGCGGTAATTCC  -G ACTTTCGTTCTTGATYRR \
    -m 100 -M 350 --match-read-wildcards --pair-filter=both -q 10 \
    -o $i-MSTAReuk_R1.clipped.fastq.gz -p $i-MSTAReuk_R2.clipped.fastq.gz \
    $i-MSTAReuk_R1.fastq.gz $i-MSTAReuk_R2.fastq.gz; done
```

Ambiguities in primers are interpreted

Cutadapt options

```
#     -g ADAPTER, --front=ADAPTER
#                        Sequence of an adapter ligated to the 5' end (paired
#                        data: of the first read). The adapter and any
#                        preceding bases are trimmed. Partial matches at the 5'
#                        end are allowed. If a '^' character is prepended
#                        ('anchoring'), the adapter is only found if it is a
#                        prefix of the read.
#       Paired-end options:
#       The -A/-G/-B/-U options work like their -a/-b/-g/-u counterparts, but
#       are applied to the second read in each pair.
#
#        -G ADAPTER          5' adapter to be removed from second read in a pair.
#        -m LENGTH, --minimum-length=LENGTH
#                        Discard reads shorter than LENGTH. Default: 0
#        -M LENGTH, --maximum-length=LENGTH
#                        Discard reads longer than LENGTH. Default: no limit
#        --match-read-wildcards
#                        Interpret IUPAC wildcards in reads. Default: False
#        --pair-filter=(any|both)
#                        Which of the reads in a paired-end read have to match
#                        the filtering criterion in order for the pair to be
#                        filtered. Default: any
#        -q [5'CUTOFF,]3'CUTOFF, --quality-cutoff=[5'CUTOFF,]3'CUTOFF
#                        Trim low-quality bases from 5' and/or 3' ends of each
#                        read before adapter removal. Applied to both reads if
#                        data is paired. If one value is given, only the 3' end
#                        is trimmed. If two comma-separated cutoffs are given,
#                        the 5' end is trimmed with the first cutoff, the 3'
#                        end with the second.
#        -o output file R1
#        -p FILE, --paired-output=FILE
#                        Write second read in a pair to FILE.
```

- After cutadapt, sequences are ready to be used in dada2

- It is good to double check that primers are gone using the same zgrep command used before

- We don’t analyze the overall quality of the sequences, as this will be done later with dada2

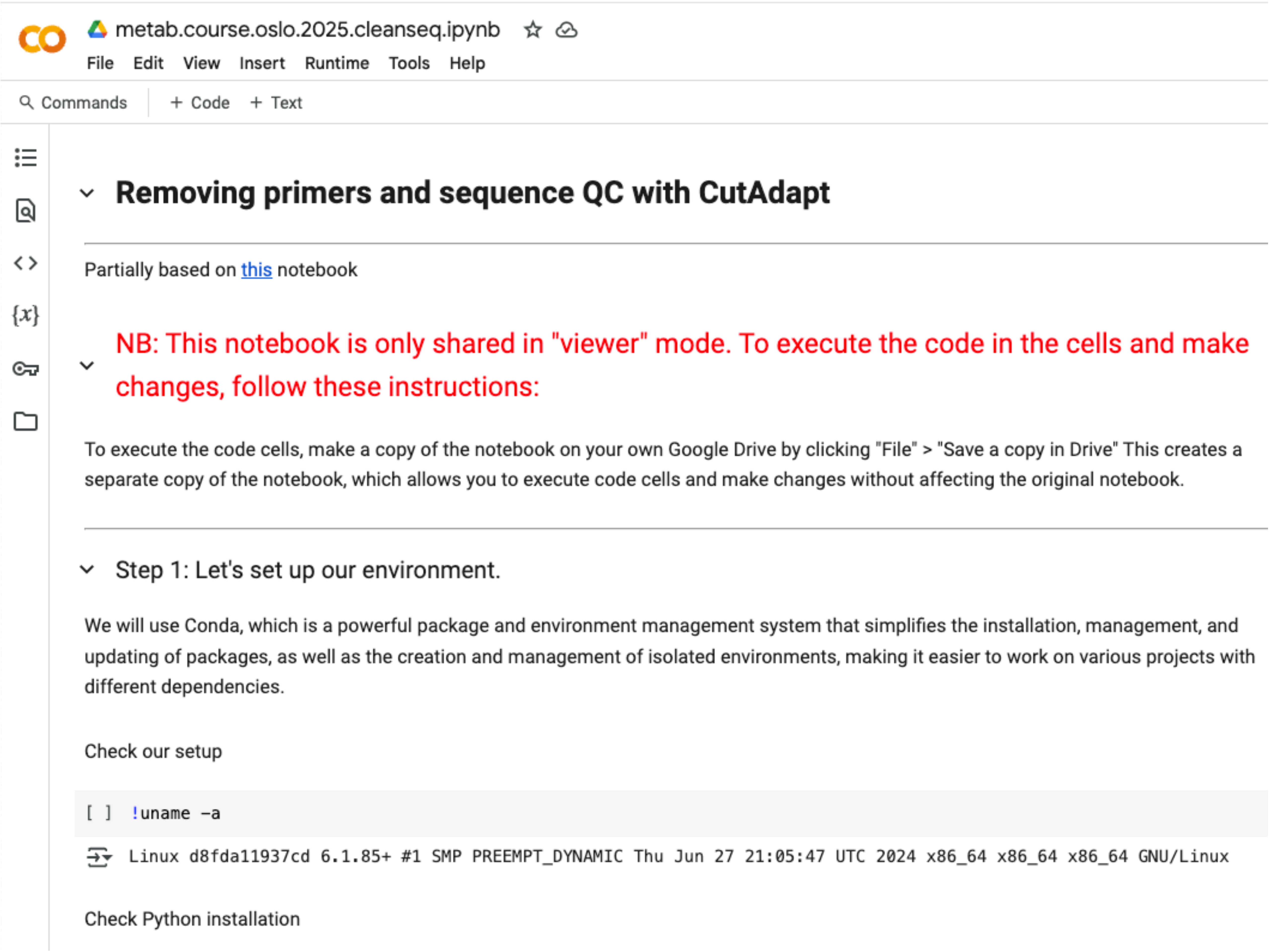
- We only remove entire sequences that look bad with cutadapt

- It is important to consider whether sequences come from sequencers with 4- or 2-color chemistries, as this will change cutadapt parameters

- There are alternative tools, such as Trimmomatic

Tutorial

https://colab.research.google.com/drive/1M68Qbti_auj_dF8yep7brjLDCzF03V2o?usp=sharing


metab.course.oslo.2025.cleanseq.ipynb
File Edit View Insert Runtime Tools Help
Commands
+ Code
+ Text

Removing primers and sequence QC with CutAdapt

Partially based on this notebook

NB: This notebook is only shared in "viewer" mode. To execute the code in the cells and make changes, follow these instructions:

To execute the code cells, make a copy of the notebook on your own Google Drive by clicking "File" > "Save a copy in Drive" This creates a separate copy of the notebook, which allows you to execute code cells and make changes without affecting the original notebook.

Step 1: Let's set up our environment.

We will use Conda, which is a powerful package and environment management system that simplifies the installation, management, and updating of packages, as well as the creation and management of isolated environments, making it easier to work on various projects with different dependencies.

Check our setup

```
[ ] !uname -a
```

```
Linux d8fda11937cd 6.1.85+ #1 SMP PREEMPT_DYNAMIC Thu Jun 27 21:05:47 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
```

Check Python installation