1 NGS Data Formats

No exercises in this section.

2 Data formats for NGS

Navigate to the data directory:

```
cd ~/course_data/data_formats/data
```

Q1: How many sequences are there in the fasta file example.fasta?

There are 10 sequences in this file. To count all the header lines, we can use

```
grep -c ">" example.fasta
```

Q2: How many reads are there in the file example.fastq?

There are 8 reads in this file. We can use `grep` to search for /1 or /2:

```
grep -c "/1" example.fastq
```

Alternatively, we can use `wc -l` to count the lines in the file and then divide this by 4.

Q3: What does RG stand for?

RG = Read Group

Q4: What platform was used to produce the data?

Illumina. See the **PL** field.

Q5: Where was the sequence data produced?

SC (Sanger Centre). See the **CN** field.

Q6: What is the expected insert/fragment size?

2kbp. See the **PI** field.

Q7: What is the mapping quality of ERR003762.5016205?

The quality is 48. We can use `grep` to find the id, followed by `awk` to print the fifth column:

```
grep "ERR003762.5016205" example.sam | awk '{print $5}'
```

Q8: What is the CIGAR string for ERR003814.6979522?

The CIGAR is 37M. We can use `grep` and `awk` to find it:

```
grep ERR003814.6979522 example.sam | awk '{print $6}'
```

Q9: What is the inferred insert/fragment size for ERR003814.1408899?

213, The ninth column holds the insert size, so we can use awk to get this:

```
grep "ERR003814.1408899" example.sam | awk '{print $9}'
```

Q10: What does the CIGAR from Q8 mean?

The CIGAR was 37M, meaning all 37 bases in the read are either matches or mismatches to the reference.

Q11: How would you represent the following alignment with a CIGAR string?

CIGAR: 4M 4I 8M. The first four bases in the read are the same as in the reference, so we can represent these as 4M in the CIGAR string. Next comes 4 insertions, represented by 4I, followed by 8 alignment matches, represented by 8M.

Q12: What version of the human assembly was used to perform the alignments?

NCBI build v37

Q13: How many sequencing runs/lanes are in this BAM file?

There are 15 lanes in the file. We can count the @RG lines manually, or use standard Linux commands such as:

```
samtools view -H NA20538.bam | grep ^@RG | wc -l
```

or

```
samtools view -H NA20538.bam | awk '{if($1=="@RG")n++}END{print n}'
```

Q14: What programs were used to create this BAM file? (Hint: have a look for the program record, @PG)

Looking at the @PG records ID tags, we see that three programs were used: GATK IndelRealigner, GATK TableRecalibration and bwa.

Q15: What version of bwa was used to align the reads?

The @PG records contain a the tag **VN**. From this we see that bwa version 0.5.5 was used.

Q16: What is the name of the first read?

The first collumn holds the name of the read: ERR003814.1408899

Q17: What position does the alignment start at?

Chromosome 1, position 19999970. Column three contains the name of the reference sequenceand the fourth column holds the leftmost position of the clipped alignment.

Q18: How many reads are mapped to region 20025000-20030000 on chromosome 1?

320 reads are mapped to this region. We have already sorted and indexed the BAM file, so now we can search for the region using **samtools view**. Then we can pipe the output to **wc** to count the number of reads in this region:

```
samtools view NA20538_sorted.bam 1:20025000-20030000 | wc -l
```

Q19: What version of the human assembly do the coordinates refer to?

The reference version is 37. In the same way that we can use `-h` in samtools to include the header in the output, we can also use this with bcftools:

```
bcftools view -h 1kg.bcf | grep "##reference"
```

Q20: How many samples are there in the BCF?

There are 50 samples in the file. The `-l` option will list all samples in the file:

```
bcftools query -l 1kg.bcf | wc -l
```

Q21: What is the genotype of the sample HG00107 at the position 20:24019472? (Hint: use the combination of -r, -s, and -f options)

The genotype is A/T. With `-f` we specify the format of the output, `-r` is used to specify the region we are looking for, and with `-s` we select the sample.

```
bcftools query -f'%POS [ %TGT]\n' -r 20:24019472 -s HG00107 1kg.bcf
```

3 Converting between formats

Navigate to the data directory:

```
cd ~/course_data/data_formats/data
```

Q1: Since CRAM files use reference-based compression, we expect the CRAM file to be smaller than the BAM file. What is the size of the CRAM file?

The file is 18MB.

Q2: Why do we need to provide the reference genome when converting to CRAM format?

For the reference based compression algorithm. The reference genome sequence is needed in order to encode and decode the sequence reads in the CRAM file.

Q3: Convert the CRAM file back to a BAM file called yeast_2.bam?

```
samtools view -b -T Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa -o
yeast_2.bam yeast.cram
```

Q4: Convert the BCF file 1kg.bcf to a compressed VCF file called 1kg.vcf.gz

```
bcftools view -O u -o 1kg.vcf.gz 1kg.bcf
```