Alignment of NGS reads, samtools and visualization
Hands-on

Software used in this practical

- BWA MEM : Burrows-Wheeler Aligner. A software package for mapping low-divergent sequences against a large reference genomes.
- SAMtools, http://samtools.sourceforge.net/
- Shell commands: wc, cat, head, tail, cut, sort, uniq, grep, awk

File formats explored in this practical

- SAM : Sequence Alignment/Map format. A TAB-delimited text format storing the alignment information. A *header section* is optional.

- Mandatory fields are:

    1. QNAME: read name
    2. FLAG: bitwise FLAG
    3. RNAME: reference sequence name (chromosome)
    4. POS: 1-based leftmost mapping position (where the read starts mapping)
    5. MAPQ: mapping quality
    6. CIGAR: CIGAR string
    7. RNEXT: name of the mate/next read (for paired ends)
    8. PNEXT: Position of the mate/next read (BP O )
    9. TLEN: observed Template LENgth
    10. SEQ: read sequence
    11. QUAL: read quality (phred-scale +33)

Data used in this practical
- mt.fa: mitochondrial sequence (reference genome)
- mt_adeno_infect_1.fq and mt_adeno_infect_2.fq: reads from a mitochondrial sequence infected by an adeno virus.

Exercise
Create an empty directory to work in the exercise and copy or download the raw data to it

mkdir data
cd data

## Exploring the FASTQ read files

Download or copy the FASTQ read files (mt_adeno_infect_1.fq and mt_adeno_infect_2.fq) to the working directory.

**Q1: How many reads do we have in our experiment?**

*wc -l *.fq*
(and divide by four)

or

*grep "^@" mt_adeno_infect_1.fq | wc -l*
*grep "^@" mt_adeno_infect_2.fq | wc -l*

**Q2: Do the names of the paired end files match?**

*grep "^@" mt_adeno_infect_1.fq | head*
*grep "^@" mt_adeno_infect_2.fq | head*

*grep "^@" mt_adeno_infect_1.fq | tail*
*grep "^@" mt_adeno_infect_2.fq | tail*

**Q3: What does it mean the symbol '^' used in the previous commands?**

## Exploring the reference genome

Explore the reference genome mt.fa

**Q4: Display the chromosome names.**

*grep "^>" mt.fa*

**Q5: How long is the reference genome in nucleotides?**

*grep -v "^>" mt.fa | awk 'BEGIN{c = 0;} { c = c + length($0)} END{ printf("length = %i\n", c); }'*

**Q6: What does it mean the option '-v' used in the previous command?**

**Q7: What does it mean the function 'length' used in the previous command?**

## BWA MEM

Lets try to align the FASTQ read files (mt_adeno_infect_1.fq and mt_adeno_infect_2.fq) using bwa to the mitochondrial genome (mt.fa).

*mkdir bwa-index*
*cp mt.fa bwa-index/mt.fa*

**Q8: Create the BWA index**

*bwa index bwa-index/mt.fa*

**Q9: Map the FASTQ files using BWA MEM**

*bwa mem bwa-index/mt.fa mt_adeno_infect_1.fq mt_adeno_infect_2.fq  > out.sam*

## Exploring the SAM files

The SAM format consists of two sections, header and alignments.
Header lines starts with "@". For the alignment the fields are: Read name, flag, reference it mapped to, position, mapping quality, cigar, mate reference map, mate position, template length, read sequence and read qualities. See SAM-specification (http://samtools.sourceforge.net/SAM1.pdf) for a thorough description.

**Q10: Display the SAM header.**

Using the grep command:
*grep "^@" out.sam*

Using the samtools:
*samtools view -SH out.sam*

**Q11: What does it mean the option '-S' used in commands above (find the documentation by executing 'samtools view')?**

**Q12: What happends if you do not include the '-S' option in the command?**

**Q13: Which information is in the @SQ line?**

**Q14: How many @SQ lines would we have if we had more than two chromosome?**

**Q15: Can you find out the specifications of the program which did create the alignment?**

**Q16: Convert SAM file to BAM file and compare file sizes**

*samtools view -bS out.sam > out.bam*

*ls -lh out.**

*file out.**

**Q17: What does it mean the option '-b' used in commands above ?**

**Q18: Display the BAM header.**

*samtools view -H out.bam*

**Q18: Convert BAM file to SAM file**

*samtools view -h out.bam > out2.sam*

**Q19: What does it mean the option '-h' used in command above ?**

**Q20: What happends if you do not include the '-h' option in the previous command?**

Explore the alignment section of the file.

You should see that some flags are "16" meaning that it mapped on the reverse strand, other flag values are “0”= mapped forward strand, "4" = unmapped... A translator for the flags can be found here https://broadinstitute.github.io/picard/explain-flags.html. Try typing in 16 and see what you get out of it. You also see that some reads have a mapping quality of 54 (this is phred-scale) and the "cigar" is "25S75M". This means 25 unmapped nucleotides and 75 matches or mismatches. You can see the sequence of the read and the qualities as well.

After the qualities there are additional fields that gives information on the alignment, eg. AS:i: means the alignment score for the read, NM:i:6 means that the edit distance for the alignment is 6, eg. we need to change 6 bases in the read to have a perfect match to the reference sequence.

**Filtering the alignments by FLAG**

We can use samtools view to filter the reads according to their FLAG using the options '-f' and '-F', check the documentation by executing 'samtools view'.

For exploratory purposes we could extract the FLAG column using samtools and the shell command cut. We know form the specification of the SAM/BAM file that the FLAG is in the **second column** or field of the file.

**Q21: Display all the flags.**

*samtools view out.bam | cut -f 2*

**Q21: Display the UNIQUE values of the FLAG.**

samtools view out.bam | cut -f 2 | sort | uniq

Remember there are some tools to help you understand such flags: http://broadinstitute.github.io/picard/explain-flags.html

**Q22: Get the unmapped reads.**

*samtools view -f 4 out.bam*

**Q23: What does it mean the '4'?**

**Q24: Get the reads mapped to the reverse strand.**

samtools view -f 16 out.bam

**Q25: Does the command above include the header in its output?**

**Q26: How would you do to include such header into the file?**

samtools view -h -f 16 out.bam

**Q27:** How do you confirm that the returned file is filtered?

samtools view -c -f 16 out.bam

**Q28: What does it mean the option 'c'?**

**Q29: How does the option '-F' work?**

## Filtering the alignments by mapping quality

We can use samtools view to filter alignments according to their **quality**. The -q option lets you set up a minimum quality threshold for reads.

Remember the Quality of the alignment is in the **fifth column** of the BAM file. You can use cut to extract just such column of the file

**Q30: Display all the mapping qualities.**

*samtools view out.bam* | *cut -f 5*

**Q31: Display the UNIQUE values of the mapping quality.**

*samtools view out.bam* | *cut -f 5* | *sort* | *uniq*

Then we can keep just the good quality reads using the '-q' option.

**Q32: Get reads with mapping quality >= 30.**

*samtools view -q 30 out.bam*

or using shell commands:

*grep -v "^@" out.bam | awk 'BEGIN { read = 0; } { if ($5 >= 30) { read = read + 1; printf("%s\n", $0); } } END { printf("num. read (mapQ >= 30) = %i\n", read); }'*

And of course several filtering options may be applied at a time.

**Q33: Get reads mapped to the reverse strand and with mapping quality >= 30.**

*samtools view -f 16 -q 30 out.bam*

**Q34: How many reads meet the previous contidition?**

*samtools view -c -f 16 -q 30 out.bam*

or

*samtools view -f 16 -q 30 out.bam | wc -l*

## Simple stats

SAMtools computes simple statistics of SAM/BAM files.

**Q35: Compute simple stats and interpret the result.**

*samtools flagstat out.bam*

## Sort and Index

Generally after alignment, the reads in the SAM/BAM files have a random order.
You can see that in our example data getting just the fourth column of the file

*samtools view -F 4 out.bam | cut  -f 4*

**Q36. What does the '-F 4' option do?**

The program samtools sort allow us sorting the alignments according to their chromosomal position.

**Q37. Sort the out.bam file.**

*samtools sort out.bam out_sorted*

**Q38. Check that the new BAM file is ordered.**

*samtools view out_sorted.bam | cut -f 4*

**Q39. Remember the unmapped reads had a 0 position in their fourth column. What does the samtools sort option does to them?**

You can use cut to extract several fields or columns of a matrix.

**Q40. What is the result of the following command: samtools view out_sorted.bam | cut -f 2,4 | head -n 20**

Sorting the reads is convenient for many purposes. One of the main ones is that a sorted file can be **indexed** so that access to it is sped up.

SAMtools have their **own indexing tool** samtools index.

Once our BAM file is sorted, we can index it by using the command samtools index. This will create the index file with the .bai extension.

**Q41. Index the previously sorted file and check the .bai file.**

*samtools index out_sorted.bam*

**Q40. Can you index the unsorted file?**

**Q42. Can you sort SAM files?**

Once we have a sorted and indexed BAM file, we can, for instance, **access to some alignments by chromosomal position**

Paired end mapping

**Q43: Compute the mean insert size.**

grep -v "^@" out.sam | awk 'BEGIN { read = 0; acc = 0; } { if ($5 > 0 && $9 > 0) { read = read + 1; acc = acc + $9; } } END { printf("num. reads = %i, acc = %i, insert mean: %0.2f\n", read, acc, (acc / read)); }'

Explore the optional fields of the alignment section

**Q44: What does mean the optional field NM?**

**Q45: Compute the mean num. mismatches per read with mapping quality > 20.**

grep -v "^@" out.sam | awk 'BEGIN { read = 0; acc = 0; } { if ($5 > 20) { read = read + 1; n=split($12, arr, ":"); printf("%s -> %i\n", $12, arr[3]); acc = acc + arr[3]; } } END { printf("read = %i, acc = %i, mismatches mean: %0.2f\n", read, acc, (acc / read)); }'

**Visualization**

Some other tools (not just SAMtools) rely up on the sorted and indexed BAM files to work.
It is the case of IGV which can be used to visualize read alignment to the genome.

**Q46: Open IGV and load the out_sorted.bam.**

**Q47: Find the reads of the file and see how they are displayed.**

Remember in our example all reads are in one single chromosome.

Find the **base position** of the read alignments and use the IGV search box to find them in the genome.

**Q48: View reads as pairs.**

**Q49: Find some reads with soft clipping, interpret them and check them in the IGV.**

*grep -v "^@" out.sam | awk '{ if ($6 ~ /S/) { printf("%i\n", $4); } }' | head*

**Q50: Find the 10 locations of the insertion of the virus (adeno.fa) into the mitochondrial DNA sequence (mt.fa) from the paired reads (mt_adeno_infect_1.fq and mt_adeno_infect_2.fq)**

grep -v "^@" out.sam | awk '{ if ($6 ~ /S/) { printf("%i\n", $4); } }' | sort -n | uniq -c | awk -v m=5 '{ if ($1 > m) { printf("%s\n", $0); } }'

Check your locations found:
```
-------------------------------------------
Number of insertion locations:
10
-------------------------------------------
Locations:
[  1] -->        31
[  2] -->       626
[  3] -->      2411
```

```
[  4] -->      2661
[  5] -->      2693
[  6] -->      6796
[  7] -->     11517
[  8] -->     12597
[  9] -->     12686
[ 10] -->     13228
-----------------------------------------------
```