Computational practical 4

Alignment and variant calling

Introduction
In the section we will learn how we identify variants: single nucleotide polymorphisms (SNPs) and small insertions and deletions (indels). For this purpose, we will use the sequence reads and align them to a reference genome (genome sequence of an isolate which is already well characterised and annotated). Over the years, a number of bioinformatic tools have been developed to enable mapping of sequence reads to a reference genome (https://www.frontiersin.org/articles/10.3389/fpls.2021.657240/full). Here we will be using the BWA (Burrows-Wheelar Aligner) tool followed by variant calling using samtools. For this process we will need two things: sequence reads and a reference genome. Choosing a reference genome is a critical step in this process, an ideal reference should: have a complete genome (a single contiguous assembly), belong to the same species as your sequenced isolate, well annotated and available at NCBI/ENA databases.

Set-up
For this practical we will be analysing the sequence reads of a *S. typhi* isolate ERR2093239. The sequence reads and the reference sequence are located in the folder cp4.
Now migrate to this folder by typing

```
cd /home/manager/course/cp4
```

Note: To ensure the tools are installed properly the following commands when typed in the terminal must not generate any error.

```
bwa
samtools
bcftools
```

4.1. Aligning reads to reference

4.1.1 The first step is to create an index of the reference genome sequence. Indexing allows the aligner to quickly identify target regions and helps make the process quicker. This needs to be carried out only once for a reference sequence and can be reused for mapping more isolated to the same reference.
Now type the following command in the terminal:

bwa index reference.fa

Check how many new files are created as a result using the command

ls -l

4.1.2 The next step is to carry out mapping using the BWA-MEM algorithm. A typical command looks like:

bwa mem reference.fa read1.fastq read2.fastq >output.sam

Note: Don't worry the command won't run as we have not provided the correct read files.

In the above command, "reference.fa" is the reference genome sequence and read1 and read2 are the two reads files. The alignment gets directed to a chosen file which is "output.sam" in the command above.
Now, we will be using the reads of the isolate ERR2093239 labelled as ERR2093239_1.fastq.gz and ERR2093239_2.fastq.gz to map on the reference sequence (*Salmonella typhi* CT18 accession:NC_003198.1).
Now, run the following command to initiate the alignment process:

bwa mem reference.fa ERR2093239_1.fastq.gz ERR2093239_2.fastq.gz >ERR2093239_aln.sam

Note: This process could take some time to complete, so it would be good to take a look at the course material so far or read more about BWA on (https://bio-bwa.sourceforge.net/bwa.shtml)

4.1.3 The output file generated is large in size therefore don't attempt to open it on the computer. The output file is in SAM format which is a tab-delimited file containing details about each read and its alignment to the reference. The details of SAM format can be found here (https://samtools.github.io/hts-specs/SAMv1.pdf)
For ease of handling and to reduce the size of the alignment file the sam file is converted into a compressed binary file called BAM file.
Type the following command in the terminal:

samtools view -O BAM -o ERR2093239_aln.bam ERR2093239_aln.sam

In the command the option -O specifies the output format and -o specifies the output file name.
We can check the difference of sizes between the sam and the bam file using the following command:

ls -lh

4.1.4 We will now sort the alignment (.bam) file by chromosome coordinates using the following command.

samtools sort -T temp -O bam -o ERR2093239_aln_sorted.bam ERR2093239_aln.bam

In the command option -T specifies the name of temporary files that are created when running the process, -O specifies the output file format and -o specifies the output file name.

4.1.5 Now index the sorted alignment using the following command.

samtools index ERR2093239_aln_sorted.bam

4.1.6 Once completed, we can extract the mapping statistics using "samtools stats". This provides details of how many reads mapped to the reference, how much of the reference is covered by the sequence reads and more. The statistics generated could reveal a lot of information about the choice of reference and the quality of sequence data as well. Use the following commands to gather statistics.

samtools stats ERR2093239_aln_sorted.bam > ERR2093239_bamstats.txt

grep "^SN" ERR2093239_bamstats.txt > stats.txt

Quiz
1: What is the total number of mapped reads? …….
2: What is the total number of unmapped reads? ……
3: What is the total number of mapped and properly paired reads? ……
4: What is the average insert size? ……
5: What is the percentage of reads properly paired? …..

4.2 Variant calling

Once we have successfully aligned the sequence reads to the chosen reference, we can use it to identify variants present in the isolate.

4.2.1 First step is to bam alignment file to bcf (binary calling format) using the following command:

```
bcftools mpileup -Ob -m 4 -f reference.fa ERR2093239_aln_sorted.bam
>ERR2093239_variants.bcf
```

Note: You might see a few warnings, you can ignore them

In the command options -Bug specifies different run parameters which can be read here (http://www.htslib.org/doc/samtools-mpileup.html), -m specifies the minimum number of reads required for indel calling and -f specifies the reference file.

4.2.2 Now we convert the bcf file to vcf (variant calling format). This step converts the binary file into the text file containing the variants identified from the alignment.

```
bcftools call -mv -O v -o ERR2093239_variants.vcf ERR2093239_variants.bcf
```

Once finished we can see the first 10 variants of the "ERR2093239_variants.vcf" file.

```
grep -A 10 "#CHROM" ERR2093239_variants.vcf
```

4.3 Filtering the variants

At this stage the variants include both SNPs and short Indels identified from the alignment of reads. The following process describes how we can identify high quality SNPs (only) from the file applying different metrics using bcftools filter function:

4.3.1 Filter only SNPs from the "MD001_variants.vcf" file.

```
bcftools filter -i 'type="snp"' -g10 -G10 ERR2093239_variants.vcf -o
ERR2093239_SNPs.vcf
```

In the command options -g removes any SNPs that fall within the indicated bp distance (10bp) from an indel and -G removes indels that are less than the indicated bp distance (10bp).

4.3.2 Filter the SNPs with base quality (QUAL) >=50, MQ >=30 and read depth (DP) >5.

bcftools filter -i 'type="snp" && QUAL>=50 && FORMAT/DP>5 && MQ>=30' -g10 -G10 ERR2093239_variants.vcf -o ERR2093239_SNPs_try1.vcf

4.3.3 Filter all homozygous SNPs (alternate base ratio >= 0.80)

bcftools filter -i 'type="snp" && QUAL>=50 && FORMAT/DP>5 && MQ>=30 && DP4[2]/(DP4[2]+DP4[0])>=0.80 && DP4[3]/(DP4[3]+DP4[1])>=0.80' -g10 -G10 ERR2093239_variants.vcf -o ERR2093239_SNPs_filtered.vcf

4.4 Creating a pseudogenome

Now that we have filtered the SNPs that we identified in the previous steps, we can use these SNPs to generate a pseudogenome which is essentially the reference genome sequence but the nucleotides are replaced by the SNPs identified at the corresponding positions.

4.4.1 First step is to create a zipped vcf file which can be achieved by using the following command followed by indexing it.

bcftools view -O z -o ERR2093239_filtered.vcf.gz ERR2093239_SNPs_filtered.vcf

bcftools index ERR2093239_filtered.vcf.gz

4.4.2 Now we use the zipped file to generate the pseudogenome that will contain the SNPs. Type the following command in the terminal:

bcftools consensus -f reference.fa ERR2093239_filtered.vcf.gz > ERR2093239_consensus.fa

4.4.3 By default the pseudogenome created will have header similar to reference but we can replace it by using the following command:

sed 's|^>.*|>ERR2093239|' ERR2093239_consensus.fa > ERR2093239_pseudogenome.fa

Now we have created pseudogenome sequence for the isolate ERR2093239. The next exercise would be to repeat the entire process from 4.1 until 4.4.3 to generate pseduogenomes for the isolates ERR2093237, ERR2093241 and ERR2093244 using the same reference. Please create a separate folder for each isolate within the same folder.

4.5 Creating a whole genome alignment.

The pseudogenome we generated in the above exercise are of exactly the same size as reference, therefore the pseudogenomes can be concatenated to create a whole genome alignment.

cat reference.fa ERR2093237_pseudogenome.fa ERR2093241_pseudogenome.fa ERR2093244_psedudogenome.fa >concatenated_alignment.fa

4.5.1 From the whole genome alignment that we created above we can select just the variable sites (snp-sites) only, this is really helpful when running computationally intensive processes such as phylogenetic tree generation. We can use the tool snp-sites for this:

snp-sites -o snpsitesOut.fa concatenated_alignment.fa

You can determine the length of the alignment using the following bash command:

sed -n '2p' snpsitesOut.fa | wc

In the above command "-n" allows printing lines, '2p' states 2nd line to print and 'wc' counts the words in the line

Quiz
What is the length of the alignment in "snpsitesOut.fa" file?

4.5.2 Identifying the pairwise genetic distance (pairwise SNPs) among the three pseudogenomes

snp-dists snpsitesOut.fa >matrix.tsv

The output file is a tab-delimited file which displays the pairwise SNP difference.