CLINICAL SCENARIOS ........ 1
1 HCV Whole Genome Sequencing ........ 2
Metagenomic sequencing ........ 3
2 Enterovirus metagenomics ........ 4
Centrifuge ........ 4
De novo assembly ........ 4
DIAMOND ........ 5
PALADIN ........ 6
Reference Mapping ........ 6
Postscript ........ 6
3 Sequence capture or Amplicon? ........ 7

# CLINICAL SCENARIOS

This section contains three scenarios that you might be interested in looking at in the Friday session.

The first one relates to the sequencing results from the samples you processed in the laboratory, but after a sequence-capture step. The exercise looks at *de novo* assembly of genomes and subsequent mapping with **BWA** and **QuasiBAM** to find depths and minority variation, and there is a possibility that the metagenomic sequencing results will also be available, but this is not certain at the time of writing! If they are available, then they can be compared to the captured data.

The second scenario is a metagenomics exercise, looking at Enterovirus data from a surveillance programme. A number of approaches are possible here, including a direct search of the FASTQs with **centrifuge**, a *de novo* assembly of contigs followed by **centrifuge**, and **DIAMOND** and **PALADIN**, two protein-searching tools that use BLAST or a modified BWA respectively. Students will be able to look at how outputs vary across the tools, and also between databases with the same tools.

The third scenario is an extension of the variant calling practical and compares data from the same HIV sample produced by two different laboratory processes – PCR and metagenomic NGS plus sequence capture. Starting from cleaned FASTQs, contigs are generated from two pol amplicons (protease-RT and integrase) using **SPAdes**. These are used to create two reference FASTA files which can then be used as input for **BWA** mapping and **QuasiBAM** analysis. A `for` loop is used to automate the repetitive tasks.

Feel free to look at all the files and outputs during this session and see what is possible with viral NGS on clinical samples.

# 1 HCV Whole Genome Sequencing

### Sequence_capture

This directory contains paired end FASTQs of the samples you processed in the laboratory, but following sequence capture. Let's try to reconstruct some genomes. One sensible approach would be to normalise the FASTQs and attempt a *de novo* assembly.

> In **all** the command lines below, replace **xx** with the number of the sample you were testing, i.e. **01-18**.

```
normalise <FASTQ xx R1> <FASTQ xx R2>
```

Here, find the FASTQ filenames to use.

```
spades.py -t 4 -k 99,127 -1 normalised_R1.fq -2 normalised_R2.fq -o
seqcap_xx
```

```
less seqcap_xx/contigs.fasta
```

To see where each contig aligns on the HCV genome, try using the HCV Sequence Locator at the Los Alamos National Laboratory, located at hcv.lanl.gov/content/sequence/LOCATE/locate.html. If the top contig is around 9kb, then it is likely that the genome has been assembled on the first pass. Copy the contig to a new file and save it with the filename **Sample_xx.genome.fas**.

If, however, a single genome contig has not been assembled, then unfortunately, we don't have the tools on the course to stitch together contigs and close reference genome sequences. There are prepared assembled genomes in the **genomes.zip** file. Use the following command to obtain the correct FASTA:

```
unzip genomes.zip Sample_xx.genome.fas
```

Once you have a FASTA file containing the whole genome, it can be used for reference mapping. It is usually a good idea to map FASTQs to a *de novo* assembled genome, in order to refine the sequence and enable minority variant calling if necessary. In some pipelines, there is further re-mapping to the mapped sequence, to reduce doubts about accuracy.

To map, we are going to use **BWA**, an aligner that exploits the Burrows-Wheeler transform and suffix arrays. These algorithms are also used in other aligners such as **Bowtie** and **SOAP2**. Their output is a SAM file that can be processed with further downstream software such as **weeSAM**, **QuasiBAM**, etc.

Firstly, an index needs to be made from the reference FASTA (our HCV genome). Here we are using the **-p** flag to rename the index (otherwise it would be prefixed with the full FASTA filename by default).

```
bwa index Sample_xx.genome.fas -p genome
```

This produces five files with the prefix **genome** that are then used by **BWA** proper:

```
bwa mem -t 4 -M genome Sample_xx.R1.fastq Sample_xx.R2.fastq >
Sample_xx.genome.sam
```

Once we have a SAM file, we can convert it to a BAM file with **samtools**[1] and use **QuasiBAM** both to look at read depths along the reference genome, and to produce a FASTA file that can be used to look for drug resistance. Feel free to adjust the parameters of **QuasiBAM** such as the `-c` or `-d` flags. Perhaps have a look at the other options such as `-f`.

```
samtools view -SbhF 0x4 Sample_xx.genome.sam > Sample_xx.genome.bam
QuasiBAM.py Sample_xx.genome.bam Sample_xx.genome.fas -d 10
```

The FASTA output from **QuasiBAM** can be submitted to an online resistance interpretation tool:

- HCV-GLUE (hcv-glue.cvr.gla.ac.uk/#/hcvFastaAnalysis)
- HCV geno2pheno (hcv.geno2pheno.org/)

Is there any resistance in your sample?

What about if you set the threshold for detection at 2% rather than 20%? (Remember, this can be achieved either by running **QuasiBAM** or **QB_reanalyse** with `-c` set to the new value, but don't forget to retain the `-d 10` flag too.)

Open the **QuasiBAM** tabular file in *Calc* or *Excel*. Plot the depth (column 3) by genome position. What do you see? Is it a flat line indicating even coverage, or does it appear that some regions have better depth of coverage than others? Why might this be? Consider the impact of this upon the qPCRs used in the laboratory QC stages.

## Metagenomic sequencing

If the metagenomic sequencing data from your laboratory training is available, then try mapping the FASTQs to the reference genome, generate a new BAM file and then look at the depths in the **QuasiBAM** tabular output file. Compare these to the captured depths. Inevitably, they will be lower due to the lack of enrichment, but is the ratio between the two depths consistent throughout the length of the genome. If not, what might be affecting it? [Hint – what is used to effect the capture?].

[1] The `-F 0x4` flag filters out unmapped reads during the conversion from SAM to BAM. This makes the BAM file smaller, considerably so when there are many unmapped reads. If the metagenomic data is available, the size differential between SAM and BAM will be even larger.

## 2 Enterovirus metagenomics

**~/Clinical/Enterovirus**

Routine national enterovirus surveillance has historically utilised Sanger sequence of a structural protein (VP1). However, many new enteroviruses have been detected in recent years (D68, A71, C106) as well as variants of existing viruses with altered pathogenicity (e.g. A6), and it is feared that exisitng tools are not capable of detecting or characterising emerging strains. Attempts are being made to improve this service in order to enable characterisation of a wider range of target viruses independent of their underlying sequence, using a WGS approach.

In this practical, there is a FASTQ dataset from an enterovirus-positive respiratory sample, as determined by low-resolution qualitative PCR detection in a peripheral hospital pathology lab. Your task is to ascertain the viral contents of the sample using the tools at your disposal.

The FASTQs have been trimmed and had almost all of their human sequences removed:

```
EV_meta_R1.fastq
EV_meta_R2.fastq
```

You may like to try:

### Centrifuge

There are two databases available, the one from Cristina's practical, found at:

```
~/Cristina/Metagenomics/Centrifuge_Db/Centrifuge-viral_db
```

and another one found at:

```
~/DBs/centrifuge/v
```

Remember, the command is as follows (substitute either of the two database locations above for ***location_of_database***, and name your own ***output_file***):

```
centrifuge -p2 -x location_of_database -1 EV_meta_R1.fastq -2
EV_meta_R2.fastq -S output_file
```

Look at the tables produced in each ***output_file***. How well can you identify the viral contents of this respiratory sample?

### De novo assembly

This can be run with normalisation. See the *de novo* assembly practical for more information. These two steps will take a little time, as the files are larger than in the original practicals.

```
Normalise.py EV_meta_R1.fastq EV_meta_R2.fastq
spades.py -t 4 -k 21,33,55,77,99,127 -1 normalised_R1.fastq -2
normalised_R2.fastq -o EV_norm &> /dev/null
```

The contigs will be found in `EV_norm/contigs.fasta` as expected. Try using BLAST to look for matches. Or maybe try to use **centrifuge** as above, but this time add the `-f` flag to signal that you are using the FASTA sequences rather than FASTQ:

```
centrifuge -p2 -f -x location_of_database EV_norm/contigs.fasta -S
output_file
```

Is the ***output_file*** any more useful this time?

## DIAMOND

This is a mapping tool that aligns nucleotide inputs to protein sequences. It exploits much of the **blastx** functionality. **RAPSearch2** and **PALADIN** (below) are two other protein-mapping tools used in similar fashion. Unfortunately, it does not take into account paired-end reads, so either process them one at a time, or concatenate them into a single file (`cat` ***file1 file2*** `>` ***bigfile***). It is possible to attempt to merge paired end reads where they overlap, but this is beyond the scope of this exercise. One example tool is **abyss-mergepairs**, but there are others.

The database used in the first instance is sourced from the NCBI ftp site and comprises the set of viral proteins and the set of human proteins. These databases are regularly updated and it is important for this type of work that local copies are kept up-to-date. These databases were created from downloads dated May 2019 so are already somewhat out of date.

The command line is as follows – substitute your own file names for ***input_FASTQ***, ***raw_output*** and ***output_SAM***:

```
diamond blastx -p 2 -f 101 -k 1 -e 1e-10 -d vh.protein.dmnd -q
input_FASTQ > raw_output
```

It should take no longer than about a minute or so with this database. By specifying `-f 101`, we set the output format to be SAM. Unfortunately, the reference names from the database are Genbank accession numbers. Luckily (!), there is a short tabular file in this directory ("***subset.local***") that matches the accession numbers to the NCBI taxonomic identifier ("taxid"). This is a tiny part of a huge tabular file available for download from the NCBI ftp site. By using a short **awk** script, the accession numbers can be replaced with taxids from this local table. Additionally, the `$3!~/^0$/` in the script ensures that non-matching reads are not written to ***output_SAM***.

```
awk 'NR==FNR{c[$1]=$2;next}c[$3]{$3=c[$3];$0=$0;print}' subset.local
raw_output >output_SAM
```

Another short **awk** script can count the instances of each taxid in ***output_SAM*** – unfortunately, **awk** is too basic to sort the outputs without extra lines but the `"\t\t"` at least makes the screen output readable!

```
awk '$0!~/^@/{c[$3]++}END{for(i in c)if(c[i]>1000)print i"\t"c[i]}'
output_SAM
```

Check out the four top hitting taxids – what are their taxa? Find out by entering the numbers into the box at the top of www.ncbi.nlm.nih.gov/taxonomy. It's also worth looking into the identity of taxid 9606, for future reference.

The same process can be followed with an enterovirus-specific database comprised of protein sequences obtained by running a query of the online NCBI protein database. This database – `ev.protein.dmnd` – is about 10% of the size of the virus/human one, but takes about five times longer to search. This is down to **DIAMOND** iterating over a series of search patterns until it finds a match; many reads will not match anything in this database even after all the search patterns have been explored.

How do the top taxa look with the EV-specific database search?

If you feel it is simply taking too long and you want to get both of the raw SAM files for this section, then type the command `unzip results.zip` to get the outputs. The files are named `EV_vhdmnd.sam` and `EV_evdmnd.sam` for the vh & ev databases, respectively. The **awk** processes still need to be run on all of these.

## PALADIN

This tool is very similar to **DIAMOND** in that it employs a protein-aligning algorithm to find matches between FASTQ reads and a database of protein sequences. Instead of **blastx**, it uses a modification of **BWA**. To speed up the process, the 21 amino acid codes (including stop) are compressed into a smaller alphabet such that similar amino acids are grouped, allowing greater matching flexibility. **DIAMOND** uses a similarly constricted alphabet in its search tool.

With **PALADIN**, we have only provided a single database, the enterovirus-specific one. This is because they are much larger for a given input reference protein FASTA file than the parallel **DIAMOND** one (compare the total sizes of all files starting with `ev.protein.faa.gz` to the single DIAMOND database file `ev.protein.dmnd`), and the virus/human one is simply too large to conveniently use!

Again, PALADIN takes a single FASTQ file, so use the same one(s) as for the **DIAMOND** run. The command line looks as follows – don't forget to substitute ***input_FASTQ*** & ***raw_output***:

```
paladin align -T 20 -t 2 ev.protein.faa.gz input_FASTQ > raw_output
```

The program should take around two to three minutes to run. The same two **awk** scripts from the **DIAMOND** exercise can be applied to the output of **PALADIN** (SAM file format is the default).

How do the outputs compare with the **DIAMOND** outputs? Given that the same source protein data was used to generate the EV-specific databases for both tools, how might these results be interpreted?

## Reference Mapping

This might involve obtaining a genome sequence from the internet and running a mapping exercise on it, but why might this not be a good initial strategy[2]?

However, after having run **centrifuge**, **DIAMOND** or **PALADIN**, might it be the time to try reference mapping? Which sequence(s) would you download from Genbank and why?

## Postscript

Enterovirus diversity is not limited to the primary genetic sequence. Intra-group recombination is not uncommon, and is particularly widespread in Enterovirus B. Inter-group recombination also occurs, with breakpoints often between the 5' untranslated region and the rest of the genome. Furthermore, there are often multiple enteroviruses present in respiratory and gastrointestinal samples. This is much less common in other sample types such as CSF. Differentiating co-infection from recombinant viruses can be quite difficult.

[2] Hint – enteroviruses, like many RNA viruses, are extremely genetically diverse.

## 3 Sequence capture or Amplicon?

```
~/Clinical/HIV_FASTQs
```

This exercise looks at parallel data for the same sample, but obtained by amplicon sequencing on the one hand, and a novel whole-genome sequence-capture technique on the other. A laboratory is weighing up whether to simply transfer the amplicons from their existing Sanger-sequencing assay onto a MiSeq instead of an ABI 3730xl, or switch to a whole new assay technology.

The question posed by the laboratory team is as follows:

**"What variation in the frequencies of minority variants in protease-RT and integrase is seen between the two datasets?"**

To approach this problem, they have produced a protease-RT amplicon, an integrase amplicon and an envelope amplicon and pooled them at roughly equimolar concentrations prior to Nextera tagmentation and MiSeq sequencing. Alongside this, the sequence capture approach using a similar assay to the laboratory HCV assay explored on Monday and Tuesday has also been sequenced:

```
amplicon.R1.fastq
amplicon.R2.fastq
```

```
seqcap.R1.fastq
seqcap.R2.fastq
```

The raw FASTQ sets have already been trimmed with trimmomatic, so the first step is to Normalise the *amplicon* dataset. This requires the *de novo* **conda** environment to be active:

```
conda activate denovo
```

```
normalise amplicon.R1.fastq amplicon.R2.fastq
```

Then use SPAdes to *de novo* assemble the protease-RT, integrase and envelope contigs and less to visualise the outputs.

```
spades.py -t 4 -k 99,127 -1 normalised_R1.fq -2 normalised_R2.fq -o
amplicon_norm
```

```
less amplicon_norm/contigs.fasta
```

In order to establish which contig corresponds to which amplicon, use either online BLAST, or the tool at www.hiv.lanl.gov/content/sequence/LOCATE/locate.html, and save the contigs from the start and end of the pol region into two fasta files – **`PRRT.fas`** and **`INT.fas`**, respectively, both within the **`HIV_FASTQs`** directory. One of the top contigs will locate into envelope (gp120). Ignore this one for the purposes of this exercise.

The next steps involve reference mapping the two original FASTQ sets to each contig and getting the QuasiBAM outputs. This requires a mapping tool (of which there are many options). Here, we are going to use **BWA**, based upon the Burrows-Wheeler transform. Prior to mapping, the reference FASTA(s) (i.e. the HIV contigs) need indexing. Then each FASTQ set can be mapped to each index, each output SAM file converted into a BAM file, and each output BAM file processed by QuasiBAM.

To speed up this process, we are going to use a pair of looping functions in our shell. Don't worry if the command lines below seem unfamiliar; however, it is worth trying to work out what each bit does when you have a moment. Type each line below exactly as written into your shell, pressing Enter after each one. Hopefully, after the **done; done** line is entered, things start to happen.

```
for i in PRRT INT; do
bwa index $i.fas
for j in seqcap amplicon; do
bwa mem -t 4 -M $i.fas $j.R1.fastq $j.R2.fastq > $i$j.sam
samtools view -Sbh $i$j.sam > $i$j.bam
quasi_bam $i$j.bam $i.fas -c 2 -d 10
done; done
```

Now there'll be a lot of sequences – 2x2 runs of QuasiBAM and three files per run. Find the two protease-RT FASTA sequences, and copy them into the Stanford website input box:

hivdb.stanford.edu/hivdb/by-sequences/

Compare the two results before doing the same with the integrase outputs.

Consider the differences between the outputs, and what might be causing them. Think back to the consensus and variant calling session yesterday morning for some ideas. Are you inclined to trust one more than the other?

To look more closely at the minority frequencies, the **QuasiBAM** tabular files can be compared line by line – this is obviously quite a boring job, but with Excel or Calc, the *second* largest frequency from columns 3-6 will give insight, particularly if they are scatter-plotted. Each line will correspond to the same line in the other tabular file as the same reference sequence was used to generate both SAMs.

Unfortunately for this exercise, it hasn't been possible to produce reference FASTAs that have the protease and RT appended separately as was done for the practical exam. One of the reasons for this that you might have spotted from the HIV sequence locator is that **SPAdes** is as likely to assemble the reads in the reverse-complement direction as in the 'correct' orientation.