CONSENSUS & VARIANT CALLING PRACTICAL

Introduction

In this session, you will learn about how to generate a consensus and look at minority variants within viral NGS datasets. For the latter, essentially, nucleotide frequency variation is determined across a region or regions of interest, and these variants are interpreted with downstream algorithms. Some tools are able to report linkage, also known as 'phasing', that is, to what degree mutations are found on the same reads (and implicitly viral genome molecules). This is clearly limited not only by depth of coverage, but also by the length of the sequenced library molecules. This is usually ~300-500bp for Illumina sequencing of viral genomes but can be much longer when looking at bacterial and eukaryotic genomes using e.g. mate-pair analysis techniques.

Variant calling starts with reads being mapped to a reference sequence, generating an alignment file, almost universally in SAM or BAM format (BAM files are simply SAM files that have been compressed to save disk space). More information about how alignment information is tabulated and stored in SAM format is found on the **samtools** github page:

https://samtools.github.io/hts-specs/SAMv1.pdf

The practical is based around looking at a variant/consensus calling tool and their outputs on two data files derived from the same FASTQ reads. Students should discover that the choice of reference for mapping is of critical importance, together with the limitations of minority variant calling when confidence in the data is low.

Command prompt

Throughout these practical notes, lines to be typed into the Unix shell are prefixed with **GCV2024:~$** in green. In the practical itself, '**GCV2024**' will be extended by text specific to the training shell, possibly including the current directory. This will not affect the running of the practical.

Results files

If time is running low, and completing all the analysis is looking unlikely, then there are pre-made results files available. In the practical directory, there is a zipped directory called **results.zip**. Within these are outputs generated by running the commands on the data for that section. You are encourage to run the commands as much as possible however!

To access the pre-run data, whilst in the directory containing the **zip** file, run the following command:

```
GCV2024:~$ unzip results.zip
```

All the outputs of the tools used in this practical will be extracted into the directory

Variant calling with QuasiBAM

For the first exercise, we will take a pair of SAM file-reference FASTA pairs and use **QuasiBAM** to explore differences in outputs caused by different references. The SAM files were prepared by firstly generating FASTQs from an amplicon covering part of the HIV pol domain important for drug susceptibility testing, running **trimmomatic** to clean the low-quality ends of reads, and then mapping them to two references using **bwa mem**:

1 HXB2
A subtype B virus dating from 1983. For three decades or so, co-ordinates of nucleotides and coding domains, roots of phylogenetic trees, and many laboratory experiments have all used HXB2 as the reference strain and it continues to serve this function.

2 de_novo_contig
This has been generated by *de novo* assembly of the trimmed FASTQs after being **normalise**-d (tomorrow's *de novo* practical will expand upon this process)

The following tools will be used:

samtools

A widely used package that contains a large number of very useful tools for manipulating the SAM and BAM files generated by mappers such as BWA.

Li H, *et al*. Bioinformatics (2009) 25(16):2078-9

QuasiBAM & QB_reanalyse

Originally written by Richard Myers at UKHSA for the express purpose of analysing variant frequencies in viral genomes. They were originally written in **C++** and **perl**, respectively, but refactored versions in **Python** are available for use in this practical. However, they are still in pre-production.

Penedos A, *et al*. PLoS One (2015) 10(11): e0143081

A: Using samtools to convert SAM to BAM

The practical starts in **/variant_calling/files**, which contains two SAM files, two reference files, and and two FASTQ files:

```
HXB2_pol.sam & contig_pol.sam,
HXB2_ref.fas & contig_ref.fas, and
FASTQ_R1.fastq & FASTQ_R2.fastq
```

The first step here is to convert each SAM file (large tabular text file that is readily viewable in e.g. *Excel* or *Calc* – try using **less** to look inside) into a BAM file (a SAM file that has been through binary compression to save space – not readily viewable, using **less** here is unhelpful). This is achieved using **samtools view** – a function within the versatile and extremely useful **samtools** package – that both converts and/or filters SAM and BAM files[1].

```
GCV2024:~$ samtools view -Sbh HXB2_pol.sam > HXB2_pol.bam
```

Use **ll** to see how the BAM files are much smaller than the SAM file. Try reversing the process to see how all the information is retained (note the absence of **Sb** input/output flags in this instance):

```
GCV2024:~$ samtools view -h HXB2_pol.bam > HXB2_pol_clone.sam
```

We can use **diff** to compare the two files. However, there will be no output at all, as both files are exactly the same!

```
GCV2024:~$ diff HXB2_pol.sam HXB2_pol_clone.sam
```

Depending upon the version of **samtools**, either nothing will be output (see the use of the **-s** flag, below), or you will get three lines that can be interpreted as follows:

> The first line (“2a3,4”) of the output tells us that the second file has two additional lines after line 2, and the second two lines contain the two commands used above – there is no difference in the main body data of the two SAM files, but it is interesting to note how each SAM file can contain some of the history of its production.

Hence, to store a mapping result, BAM files are much more space-efficient than SAMs. Even the unmapped read data is retained, so the FASTQ files used to generate the SAM file can be discarded! This can be demonstrated by using another of **samtools**’ functions – **fastq**. This tool generates FASTQ files from BAM files:

```
GCV2024:~$ samtools fastq -1 test1.fq -2 test2.fq HXB2_pol.bam
```

Try **diff** again to look for differences. Try it with and without the **-s** flag[2]:

```
GCV2024:~$ diff -s FASTQ_R1.fq test1.fq
GCV2024:~$ diff FASTQ_R1.fq test1.fq
```

[1] The set of flags applied here are **-S**, **-b**, and **-h** (collated into **-Sbh** for convenience):
-S declares the *input* to be in SAM format.
-b dictates that the *output* should be in BAM format – it can be omitted if the output file has the **.bam** suffix, i.e. **samtools** detects what is needed.
-h flag tells **samtools view** that the header information is to be retained, which is essential for many if not most downstream BAM-using applications, including **QuasiBAM**.

[2] The **-s** flag tells **diff** to report when the two files are the same rather than outputting nothing.

B: Using QuasiBAM – nucleotide frequency tables

QuasiBAM takes as input a BAM file and a FASTA file containing relevant reference sequence(s). In order to obtain in-frame amino acid variant information, one or more of the reference sequences can each have one or more sub-sequences corresponding to coding regions. These subsequences must be exactly contained within the larger reference genome, i.e. not derived from an alternative source, and must be present in the reference file *after* the primary reference sequence. All sequences must be provided in FASTA format.

Here, the additional subsequences of the protease ('PR') and reverse transcriptase ('RT') fragments have been provided – use `less` to look inside `HXB2_ref.fas`.

Run **QuasiBAM** on the first BAM-FASTA pair:

```
GCV2024:~$ QuasiBAM.py HXB2_pol.bam HXB2_ref.fas
```

For each of the reference and subsequence(s), there are 2 output files:

`<name>.tabular`

A nucleotide frequency table with one row per nucleotide position, and columns including depth, `A`/`C`/`G`/`T` frequencies, gaps, inserts and more. This can be visualised using `less`[3], but will be easier to look at in a spreadsheet application like *Excel* or *Calc*.

`<name>.fas`

A FASTA file generated from the nucleotide frequency table. There are two key parameters dictating how these are derived:

Depth: Only positions where the depth of coverage exceeds a specified value (default = 100) are reported. Positions with lower depths are reported as Ns.

Consensus frequency: At each position, the output IUPAC-IUB nucleotide code[4] reflects all bases whose frequency exceeds a specified percentage. Hence this is the threshold for minor variant reporting. The default is set here at 20.0%, the frequency generally accepted as equivalent to the sensitivity of Sanger sequencing.

[3] The `-S` flag chops off the ends of lines that extend beyond a single screen width (as opposed to wrapping the text onto the next screen line).

[4] See en.wikipedia.org/wiki/Nucleic_acid_notation#IUPAC_notation for more information on these base codes.

Questions

1. Have a look at the various FASTA outputs (e.g. using **`less HXB2_pol.<domain>.fas`**). Are the PR and RT subsequences complete?

2. Are all bases present?

3. How many are missing at the 5' end of PR?

4. Examine the nucleotide frequency table **`HXB2_pol.PR.tabular`**. The third column gives the depths. Can these values be correlated with the bases identified in the previous question?

5. If you have access to *Calc* or *Excel* can the missing positions in RT be correlated with data in its nucleotide frequency table?

C: Re-analysing the QuasiBAM output

The problem with the initial QuasiBAM output is that there are regions where the depth is below 100 – the FASTA file has Ns at these loci. To generate a FASTA file with a lower depth, we can run **`QB_reanalyse`**. This is a tool that uses the **`tabular`** files generated by QuasiBAM to generate new FASTAs with new parameters. By entering the following command, the usage is displayed:

By entering the following command, the usage is displayed:

```
GCV2024:~$ QB_reanalyse.py
```

The values in square brackets at the end of each line are the defaults, i.e. *the value that is used if no other value is specified*. Looking at the usage, we need to provide a **`tabular`** input file as the last argument. For this exercise, we also need to enter the new depth using **`-d`**:

```
GCV2024:~$ QB_reanalyse.py -d 30 HXB2_pol.tabular
```

The script is quicker than a QuasiBAM re-run as it only reads the tabular file rather than the BAM file. The output filename (**`HXB2_ref.c20.0.d30.fas`**) incorporates the new consensus and depth parameters.

It must be noted that QB_reanalyse cannot ‘see’ frequencies that have not been reported in the tabular file. If a lower frequency than the reporting frequency is needed, QuasiBAM will need to be re-run. The default reporting frequency is 1%; this can be set using the **`-f`** flag.

Questions

1. Look in the output FASTA file for the “HXB2” data. Has the the complete sequence has been recovered by changing the minimum reportable depth to 30?

2. Have a look at the top of the **`HXB2_pol.tabular`** file. How might you rewrite the **`QB_reanalyse.py`** command line to make sure that the first five bases of protease (‘PR’) were reported?

D: The importance of the reference sequence when using mapping data

Repeat the analysis on the second BAM/FASTA filepair, replacing **`HXB2`** with **`contig`** in the filenames.

Questions

1. Take a look at the contig-derived output FASTA file with the default depth of 100. How is it different to the one using HXB2 as a reference?

2. What is the default *consensus* frequency for **`QB_reanalyse.py`**?

3. Think about whether you might have any concerns about using this frequency at depths below 100? What about at lower frequencies, e.g. 10%, 5% or 2%?

4. Type **`QuasiBAM.py`** into your shell. Can depth and consensus frequency be controlled at this stage?

6. Why might there be Ns at the beginning and ends of the reference sequences in both FASTAs (hint: think about where the data came from)?

E: Minority variants (interpretation)

Now let's look at minority variants. For this, we are going to use the 'RT' subsequences of the two reference files. QuasiBAM has already generated the tabular files for both PR and RT, so we can use `QB_reanalyse.py` directly on the RT files.

Firstly, generate two new consensus files, each with a depth threshold of 30 and a consensus threshold of 2 (i.e. we are going to look at all nucleotides present at ≥2%)

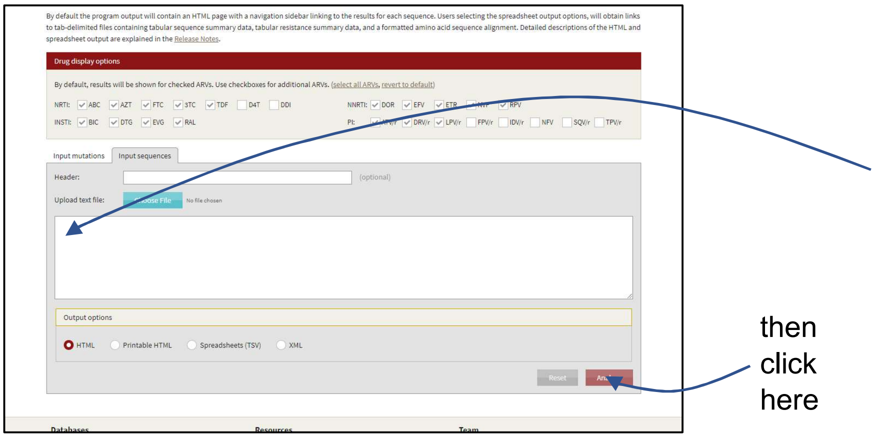
```
GCV2024:~$ QB_reanalyse.py -d 30 -c 2 HXB2_pol.RT.tabular
```

Repeat the above command, replacing **`HXB2`** with **`contig`** in the tabular filename.

Open each output FASTA file and submit the new RT sequences to the online Stanford HIV drug resistance interpretation tool, found at the following URL:

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

Paste the sequences into the large text box, and click Analyze.



By default the program output will contain an HTML page with a navigation sidebar linking to the results for each sequence. Users selecting the spreadsheet output options, will obtain links to tab-delimited files containing tabular sequence summary data, tabular resistance summary data, and a formatted amino acid sequence alignment. Detailed descriptions of the HTML and spreadsheet output are explained in the Release Notes.

Drug display options

By default, results will be shown for checked ARVs. Use checkboxes for additional ARVs. (select all ARVs, revert to default)

NRTI: ☑ ABC ☑ AZT ☑ FTC ☑ 3TC ☑ TDF □ D4T □ DDI

NNRTI: ☑ DOR ☑ EFV ☑ ETR ☑ NVP ☑ RPV

INSTI: ☑ BIC ☑ DTG ☑ EVG ☑ RAL

PI: ☑ ATV/r ☑ DRV/r ☑ LPV/r □ FPV/r □ IDV/r □ NFV □ SQV/r □ TPV/r

Header: (optional)

Upload text file: No file chosen

Output options

● HTML ○ Printable HTML ○ Spreadsheets (TSV) ○ XML

Paste sequence(s) here

then click here

Notice the differences in the reports – specifically, the drug resistance interpretations and the quality information.

Question

1. What are the key differences in output between the two sequences?

F: Minority variants (a closer look at the data)

Now let's look more closely at the `<name>_ref.RT.tabular` files. It is useful to be able to view the data in a more user-friendly format, so open them both in *Calc*.

Start by looking at the amino acid positions in RT where there were resistance mutations reported in Stanford:

65 This is a key position for multiple NRTI resistance, particularly for tenofovir and 3TC, two drugs used in many first line therapeutic regimens. Under- or over-reporting this mutation would have a significant impact on patient treatment.

77 Mutations at 77 are usually found as part of a multi-resistant array of mutations centred around Q151M. Found individually, they are of uncertain significance.

103 Lysine (K) is the wild-type, and usually it is N that is seen as at this position in viruses resistant to first-generation NNRTIs such as nevirapine and efavirenz.

184 Methionine (M) usually passes through an intermediate isoleucine (I) intermediate *en route* to valine (V) when HIV acquires resistance to 3TC and FTC. This reflects the mutational bias of retroviral reverse transcriptases.

Questions

1. Is there any trace of K65R in `HXB2_pol.RT.tabular` at the amino acid or nucleotide level?

2. What is its frequency in `contig_pol.RT.tabular`?

3. Could this be significant?

4. What about F77L in `contig_pol.RT.tabular`?

5. Why was this not seen in the Stanford report?

6. What is happening at amino acid positions 63 & 64 of RT? Why might `HXB2_ref.RT.tabular` be reporting gaps here?

7. What about the same region of the `contig` dataset?

8. What are the amino acid mixes at position 93 of RT?

9. Imagine if it were established that in cases where M184I is present at 3% or above, it is considered prejudicial to treatment with 3TC and FTC. Is this sample resistant? What difficulties might be encountered when trying to validate an assay, given the reported M184I significance threshold of 3% in the previous question?

10. Where did the four cysteines (TGT & C) seen at RT position 127 in the **`codon`** and **`AA`** columns of **`contig_pol.RT.tabular`** come from, considering that there is no G reported at nucleotide position 380 (hint: look at the homologus locus in the HXB2 tabular file and consider the **`-f`** flag of **`QuasiBAM.py`**)?

MORE ABOUT THE QuasiBAM NUCLEOTIDE FREQUENCY TABLE

The meaning of each column is given in the following table:

#	Header	Meaning
1	**Pos**	Position (within subsequence, if present)
2	**Ref_N**	Reference nucleotide at position **Pos**
3	**Depth**	Depth of coverage at that *nucleotide* locus
4	**A**	Frequency of A
5	**C**	Frequency of C
6	**G**	Frequency of G
7	**T**	Frequency of T
8	**Gap**	Frequency of gap (i.e. deletions)
9	**Ins**	Frequency of insertions
10	**I_Desc**	Insert descriptions & frequencies
11	**AA_pos**	Amino acid position within subsequence (if present)
12	**Ref_AA**	Reference amino acid starting at position **Pos**
13	**AA_depth**	Depth of coverage of the *codon* starting at position **Pos**
14	**Cod**	Frequency distribution of codon triplets starting at position **Pos**
15	**AA**	Frequency distribution of amino acids at position **AA_pos**

I_Desc, **Cod**, and **AA** columns all contain a variable number of insertions/codons/amino acid identities. Their information is given as space-separated sets of colon-separated data points comprising the identity of the sequence phenomenon followed by its absolute frequency (i.e. depth) and its percentage frequency (of all such phenomena at that locus)

From columns 3-10 are derived the FASTA outputs, with insertions and deletions being treated according to the depth and consensus parameters. Where there are multiple insertions whose depth and frequency exceed the specified parameters, the most frequent is inserted.

The quality scores of each nucleotide are paramount in explaining the difference in depths between columns 3 and 13. Essentially, the depth reported in column 3 is that of nucleotides passing a quality threshold (default=36, this can be adjusted with the **-q** flag) – i.e. if the run quality was poor, the number of *reads* that cover that position may be considerably higher. In this situation, the (hidden) frequency of low-quality bases will influence the number of triplets in which all bases pass the quality threshold.

For example, for a given codon triplet, if the ratio of high-quality bases to low-quality reads is high, e.g. 9:1 as in Table 1 (below), then assuming the quality bases are independently distributed amongst all reads, there will be:

900 * 0.9 * 0.9 = 729

triplets with high quality bases at all positions. However, if there are a large number of hidden low-quality bases as the 50% seen in Table 2, although the nucleotide depths are identical, the number of high-quality triplets will be:

900 * 0.5 * 0.5 = 225.

In QuasiBAM outputs, the 900 high-quality bases is the number in the **Depth** column, and the number of high-quality triplets (e.g. 729 or 225) is the **AA_Dep** column figure.

Table 1:

Codon position	High-quality bases	Low-quality bases
1	900	100
2	900	100
3	900	100

Table 2

Codon position	High-quality reads	Low-quality reads
1	900	900
2	900	900
3	900	900

FURTHER READING

Lefterova, M. I., Suarez, C. J., Banaei, N. & Pinsky, B. A. Next-Generation Sequencing for Infectious Disease Diagnosis and Management: A Report of the Association for Molecular Pathology. J. Mol. Diag. 17, 623-634 (2015).

Macalalad, A. et al. Highly sensitive and specific detection of rare variants in mixed viral populations from massively parallel sequence data. PLoS Comput. Biol. 8, e1002417 (2012).

McCrone, J. T. & Lauring, A. S. Measurements of Intrahost Viral Diversity Are Extremely Sensitive to Systematic Errors in Variant Calling. J. Virol. 90, 6884–6895 (2016).

Orton, R. J. et al. Distinguishing low frequency mutations from RT-PCR and sequence errors in viral deep sequencing data. BMC Genomics 16, 229 (2015).

Schlaberg, R. et al. Validation of metagenomic next-generation sequencing tests for universal pathogen detection. Arch. Pathol. Lab. Med. 141, 776–786 (2017).

Verbist, B. M. P. et al. VirVarSeq: a low frequency Virus Variant detection pipeline for Illumina Sequencing using adaptive base-calling accuracy filtering. Bioinformatics 31, 94-101 (2014).

Yang, X., Charlebois, P., Macalalad, A., Henn, M. R. & Zody, M. C. V-Phaser 2: variant inference for viral populations. BMC Genomics 14, 674 (2013).

ANSWERS TO QUESTIONS

Please note that the exact numbers may vary slightly owing to changes in software versions between the setting of the questions and the course!

B: Using QuasiBAM – nucleotide frequency tables

1. No
2. No
3. 28
4. Positions with depths <100 are set to N
5. Positions with depths <100 are set to N

C: Re-analysing the QuasiBAM output

1. Not quite – 5 Ns still remain at the 5' end of protease, and over 50 bases are still missing from the 3' end of RT.
2. Set the -d flag to 13 or lower.

D: The importance of the reference sequence when using mapping data

1. Far fewer Ns, particularly in the middle of the sequence.
2. 20.0 (this is expressed as a float to allow for fractional percentages such as 17.5%)
3. How reliably do the variant frequency percentages reflect 'true' frequencies in the original material? At higher depths, the number of sampled molecules is high, mitigating stochastic variation. 5% of depth 20 would represent a single read having the variant. Think about error rates here!
4. Yes, -d and -c are flags here too.
5. The PCR product is not much larger than the reference sequence. During Nextera tagmentation, library fragments are generated by cleaving the target DNA in two places and ligating barcode adapters to both ends. For a terminal base to be included in a fragment, one of the tagmenting cleavages must be even more terminal. Thus chance dictates that fragments are more likely to contain internal sequences, and up to 50 bases at either end of a dsDNA molecule can be unsequenced.

E: Minority variants (interpretation)

1. Eight unusual RT mutations are detected in the HXB2 output, when compared to the contig output. Resistance mutation F77FL is seen in the HXB2 but not in the contig, and K65KR seen in the contig output but not HXB2. There are differences in the resistance profiles to NRTIs.

F: Minority variants (a closer look at the data)

1. No
2. 4.569%

3. Yes, it may indicate low-level resistance to a number of drugs (abacavir and tenofovir especially).
4. 1.29% (as against 3.774% in the HXB2 dataset).
5. The consensus reporting threshold was set at 2% (`-c 2`).
6. Because the HXB2 reference is genetically distant from the sample sequence, the alignment of the FASTQ reads (using BWA) has led to significant mismatch in this area.
7. In the *contig* dataset, there is a 3.38% gap frequency at nucleotide position 189. BWA aligns indel mutations wherever possible, and this position is at the start of a stretch of 6 As. These homopolymeric tracts are prime sites for indel mutations, and it is likely that this is an artefact of polymerase infidelity within both the PCR amplification and the sequencing processes.
8. RT93 is 100% G. Do not trust the HXB2 outputs where there are many purported variants, but with only single instances of each! The depth is low, so the percentages seem artificially high. In the contig tabular file, the depth is much higher (better mapping of the reads) and the frequencies of the other amino acids arising from stray reads each drop to well below 1%.
9. This is a bit of a trick question, as there are inevitably going to be large variations in variant frequency outputs between assays, owing to the laboratory methods (PCR/metagenomics/Sequence Capture) and equally importantly, the bioinformatic pipelines and the reference sequences. Converging on a precise figure that would apply in all circumstances will be effectively unattainable. Only if a single method was used universally could a cut-off be applied. It is possible that a clinical trial might employ such a strategy.
10. The frequency of G at the second codon position (row 380) in the *contig* datasets is just above 98.5%, suggesting there might be one or more nucleotides present at just under the reporting threshold of 1%. If these are in reads of relatively high base quality, then their associated codon frequencies may exceed the 1% threshold.

    In the HXB2 dataset, G is reported at just over 1%, but this corresponds to just 2 reads. As the depth is higher in the contig dataset, the four mapped reads with a G come in at under 1% whereas the corresponding codon is >1%.

    See the section "More about the QuasiBAM nucleotide frequency table" for more info.