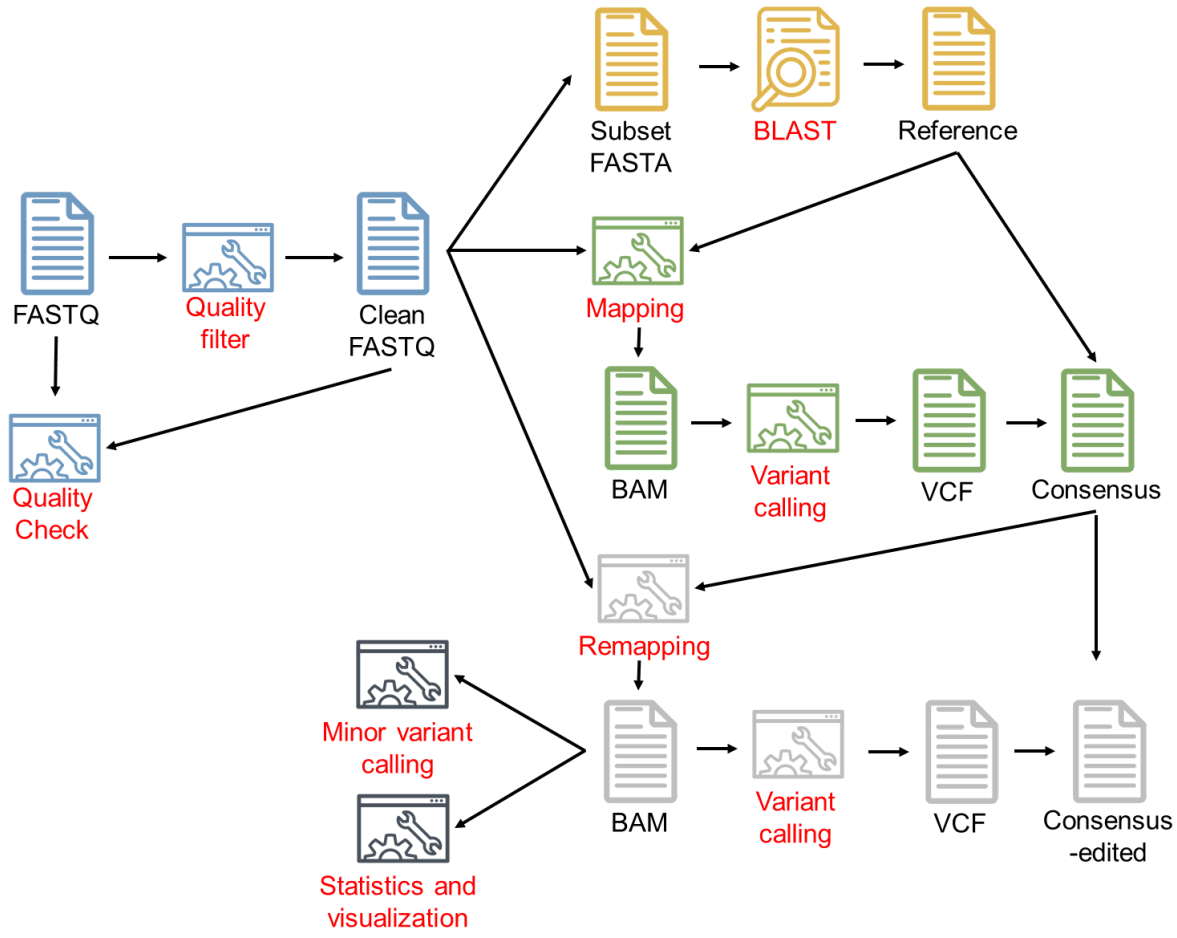
Reference assembly with long reads

Overview



Basecalling - skipped

The hardware requirement is not met. Dorado requires Graphical Processing Unit (GPU) which is unfortunately not available in pulsar.

However, if you would one day run basecalling of Nanopore data, check out epi2me workflow which utilizes nextflow and singularity/docker.

wf-basecalling documentation

Basecalling workflow This repository contains a nextflow workflow for basecalling a directory of…

https://labs.epi2me.io/workflows/wf-basecalling/

The other option would be to compile the program from their official github page https://github.com/nanoporetech/dorado.

```
wget https://cdn.oxfordnanoportal.com/software/analysis/dorado-0.4.1-linux-x64.tar.gz
tar -xvf dorado-0.4.1-linux-x64.tar.gz

#check if the compiling is successfull
./dorado-0.4.1-linux-x64/bin/dorado
```

Pre Processing

1. Quality check with NanoPlot https://github.com/wdecoster/NanoPlot

   Input for NanoPlot can be raw fastq or the summary file resulted from basecalling.
   To see the full options use `NanoPlot -h` .

   ```
   mkdir qc_raw
   NanoPlot --fastq ./training_material/fastq/sample01.fastq.gz \
   -o ./qc_raw/sample01/
   ```

   `--fastq`: location of the fastq file
   `-o` : location of the output

2. Filter out short and low quality reads with NanoFilt (no complexity filter) and fastp.

   a. Using nanofilt https://github.com/wdecoster/nanofilt

   Type `NanoFilt -h` for rull options.

   ```
   mkdir clean

   gunzip ./training_material/fastq/sample01.fastq.gz

   NanoFilt -l 300 --maxlength 3000 -q 10 --headcrop 20 --tailcrop 20 \
   ./training_material/fastq/sample01.fastq > ./clean/sample01.fastq
   ```

   `-l`: minimum length filter
   `--maxlength`: maximum length filter
   `-q` : base quality threshold
   `--headcrop` : how many bases to be cut from the 5' end
   `--tailcrop`: how many bases to be cut from the 3' end

   Compressing/decompressing files

File extension	Decompressing	Compressing
zip	`unzip`	`zip`
gzip/gz	`gunzip`	`gzip`
tar.gz	`tar -xzf`	`tar -czf`

   b. Using fastp https://github.com/OpenGene/fastp

   Filter out the short reads and low complexity reads with fastp. The command below specify the minimum read length (`--length_required 300`), maximum read length (`--length_limit 3000`), and minimum mean read quality (`-e 10`).
   Use `fastp -h` to see the full options.
   Note: fastp is originally created for Illumina.

   ```
   fastp -A --length_required 300 --length_limit 3000 -e 10 \
   -i ./training_material/fastq/sample01.fastq.gz -o ./clean/sample01.fastq
   ```

   `-A` : disable the adapter trimming (this tool is build for illumina)
   `--length_required` : minimum length filter
   `--length_limit`: maximum length filter
   `-e` : base quality filter
   `-i` : input file
   `-o` : output fi

3. Quality check

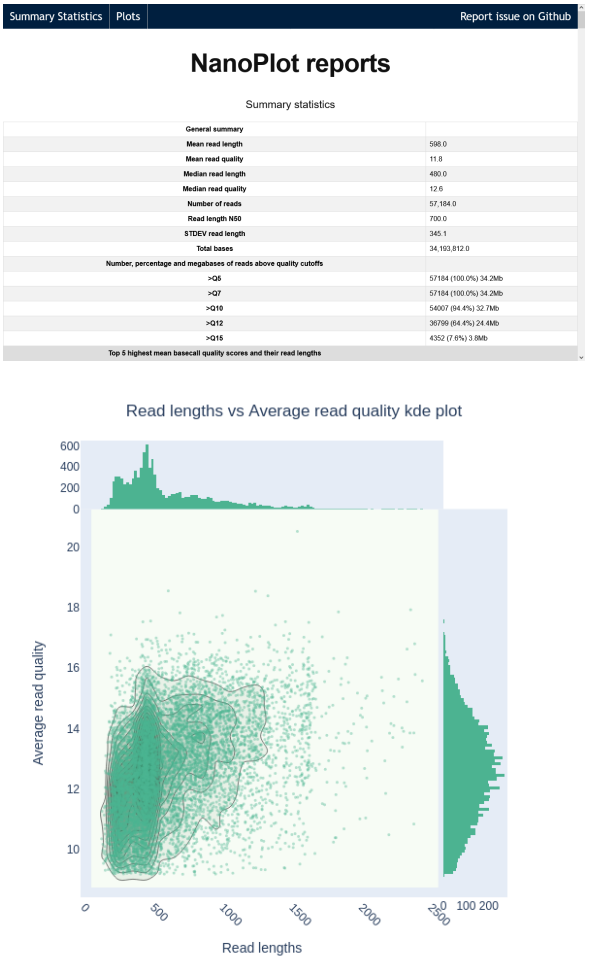
```
mkdir qc_clean

NanoPlot --fastq ./clean/sample01.fastq -o ./qc_clean/sample01/
```

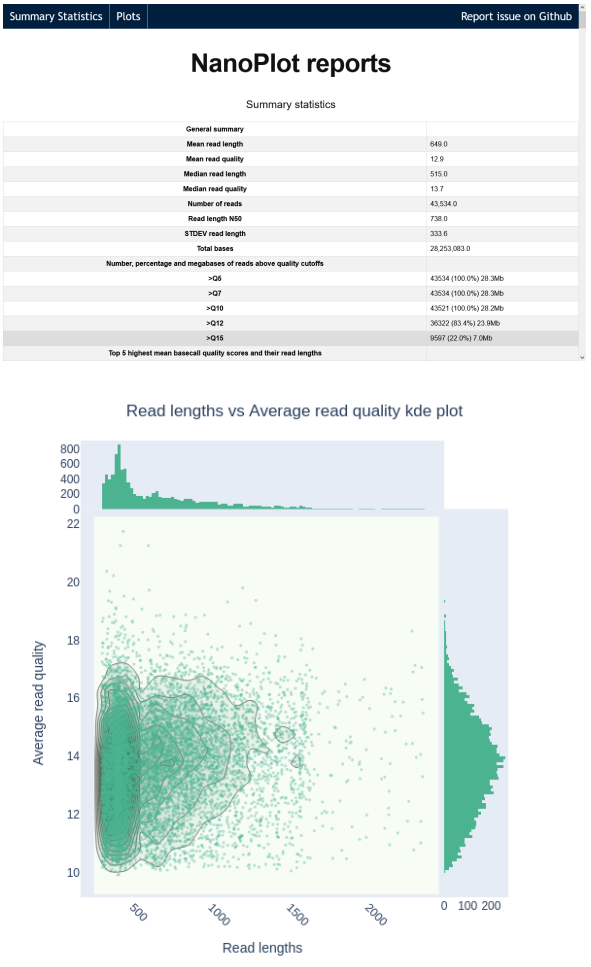
`--fastq`: location of the fastq file
`-o`: location of the output

QC result

Before



Afte



Mapping

We will use the **sample01.fastq, sample02.fastq, and sample03.fastq** located in **./training/fastq/**.

1. BLAST search
   a. Build BLAST database with `makeblastdb`

```
#build a blast database
mkdir ./db/
mv ./training_material/db/fluadb.fa.gz ./db/
gunzip ./db/fluadb.fa.gz
makeblastdb -in ./db/fluadb.fa -dbtype nucl -out ./db/fluadb.fa
```

`-in` : input file (fasta).
`-dbtype` : type of the database, `nucl` for nucleotide and `prot` for protein.
`-out` : the output directory.

Question: how many sequences are included in the database?

b. Subset 1000 reads from the file.

```
mkdir ./sample01/
head -4000 ./clean/sample01.fastq | seqkit fq2fa -w 0 \
> ./sample01/subsetsample01.fasta
```

`head -4000` : subset 4000 lines from the beginning.
`seqkit fq2fa` : transform fastq file to fasta file.
`-w 0` : print the sequences in one line.

c. Run blast

Keep in mind that blast is a heuristic process. When specifying the command to only give one hit, this hit is not necessarily the best hit. It is simply gave the first best hit it found in the reference. That is why when you run this command you will get a warning: `Warning: [blastn] Examining 5 or more matches is recommended`.

```
blastn -query ./sample03/subsetsample03.fasta \
-db ./db/fluadb.fa \
-outfmt "6 qseqid bitscore pident length sseqid gapopen qstart qend sstart send evalue bitscore qlen slen" \
-num_threads 16 -perc_identity 90 -max_target_seqs 1 -out ./sample03/blastres03.txt &
```

`-query` : the input file (fasta).
`-outfmt` : the type of format to output the result. In this example we asked for format 6 (tabular).
`-num_threads` : number of threads to use.
`-perc_identity` : threshold for percent identity (`pident`).
`-mar_target_seqs` : number of hits for each sequence.
`-out` : output filed
`&` : tell the shell to run the command on the background.

While it run practice with `bg` , `fg` , `jobs` , and `CTRL+Z` .

Check the blast result

```
#check the result
head ./sample01/blastres01.txt
```

Should give you something like this.

```
SRR23318372.2   1718    95.041  1109    2457043|EPI_ISL_17187790|A/Uchaf/0224/2022|NA   20      92      1179    1290    18
SRR23318372.3   540     95.376  346     2599592|EPI_ISL_17806299|A/Lebanon/105/2023|MP  9       94      433     538     19
SRR23318372.4   830     96.806  501     2468511|EPI_ISL_17246596|A/England/225120420/2022|MP    7       97      590     45
SRR23318372.12  878     96.161  547     2601824|EPI_ISL_17813962|A/ABUDHABI/UAE/0004240/2022|NP 12      96      631     42
SRR23318372.13  1328    97.000  800     2316013|EPI_ISL_16680982|A/Nord_Pas_de_Calais/57131/2022|MP     16      176     96
SRR23318372.17  1683    95.199  1083    2604128|EPI_ISL_17832066|A/California/122/2022|NP       19      97      1154    15
SRR23318372.20  575     96.078  357     2541090|EPI_ISL_17607760|A/United_Kingdom/GSTT-IAV-ED63/2022|HA 6       90      44
SRR23318372.24  350     91.892  259     2356101|EPI_ISL_16822937|A/France/ARA-HCL022217909701/2022|PB1  7       84      33
SRR23318372.27  1511    94.828  986     2601508|EPI_ISL_17813906|A/ABUDHABI/UAE/0004579/2022|NP 21      82      1049    98
```

d. sort the results

Find the hit for PB2 gene with the highest bitscore.

```
grep "|PB2" ./sample01/blastres01.txt | sort -k 2n | tail -1 | awk '{print $5}' > ./sample01/ref.acc
```

In simple, this command means to find line contains the gene name, sort the result based on column number 2 (bitscore). Then get the last line (result with the highest bitscore), then print column number 5, which will give us the reference sequence's id.

`sort -k 2n`: sort the data based on column two and treat the data as numeric.
`tail -1` : print last line.
`awk '{print $5}'` : print the fifth column, can be replaced with `cut -f5`.

Find the hit for PB1 gene with the highest bitscore.

```
grep "|PB1" ./sample01/blastres01.txt | sort -k 2n | tail -1 | awk '{print $5}' >> ./sample01/ref.acc
```

Repeat for other segments: PA, HA, NA, NP, MP, NS.

Hint: you can simplify this process using loop command.

e. Check if all eight segments are available

```
more ./sample01/ref.acc #or
wc -l ./sample01/ref.acc #should return 8. If it is less than 8, consider to blast more sequences.
```

The content of ref.acc file.

```
2604131|EPI_ISL_17832066|A/California/122/2022|PB2
2541076|EPI_ISL_17607760|A/United_Kingdom/GSTT-IAV-ED63/2022|PB1
2541082|EPI_ISL_17607760|A/United_Kingdom/GSTT-IAV-ED63/2022|PA
2601955|EPI_ISL_17813980|A/ABUDHABI/UAE/0004555/2022|HA
2589091|EPI_ISL_17785701|A/Bangkok/P1601/2023|NA
2604128|EPI_ISL_17832066|A/California/122/2022|NP
2604398|EPI_ISL_17832843|A/Michigan/UOM10049390076/2023|MP
2571447|EPI_ISL_17699895|A/Serbia/7001/2022|NS
```

f. Extract the reference sequence from list of sequence used for blast database

```
#we will use loop for this
while read line; \
do grep -A1 $line ./db/fluadb.fa; \
done < ./sample01/ref.acc > ./sample01/sample01ref.fasta
```

This is a loop command. It basically tells the computer to read each line in file.acc and find 1) line containing the string and 2) one line after the match in the file fluadb.fa. The result STDOUT is then redicrect to sample01ref.fasta.

Check the reference file

```
cat ./sample01/sample01ref.fasta
```

2. Mapping

a. Mapping with `minimap2`

```
minimap2 -ax map-ont ./sample01/sample01ref.fasta \
./clean/sample01.fastq > ./sample01/sample01.sam
```

`-a:` ouput as sam
`-x`: preset, in this case use `map-ont` preset.

Check the sam file with `more` command:

```
more ./sample01/sample01.sam
```

b. SAM to BAM

1. Convert SAM to BAM.

```
samtools view -b ./sample01/sample01.sam > ./sample01/sample01_unfilt.bam
```

`samtools view`: convert a sam/bam/cram file into a sam/bam/cram file.
`-b`: output as a bam file.

2. Filter the BAM file using the `-F` and `-f` options.

```
samtools view -b -F 2052 ./sample01/sample01_unfilt.bam > ./sample01/sample01_filt.bam
```

`samtools view`: convert a sam/bam/cram file into a sam/bam/cram file.
`-b`: output as a bam file.
`-F` or `-f`: flag to exclude/keep.

What does SAM flag 2052 stands for? Check in Explain SAM Flags (broadinstitute.github.io)

3. Sort BAM files

`samtools sort` id used to sort the BAM file based on position.

```
samtools sort ./sample01/sample01_filt.bam > ./sample01/sample01_filt_sort.bam
samtools sort ./sample01/sample01_unfilt.bam > ./sample01/sample01_unfilt_sort.bam
```

The three commands above can be merged as:

```
minimap2 -ax map-ont ./sample01/sample01ref.fasta ./clean/sample01.fastq | \
samtools view -bS -F 2052 | samtools sort > ./sample01/sample01_filt_sort.bam
```

`-b`: output in BAM format, `-S:` input is SAM format, `-F`: exclude alignment containing certain flags.

Check the mapping result using `samtools idxstats`.

```
samtools index ./sample01/sample01_filt_sort.bam
samtools idxstats ./sample01/sample01_filt_sort.bam
```

It should show something like this:

```
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      2316    639     0
2541076|EPI_ISL_17607760|A/United_Kingdom/GSTT-IAV-ED63/2022|PB1        2274    2403    0
2541082|EPI_ISL_17607760|A/United_Kingdom/GSTT-IAV-ED63/2022|PA 2151    1696    0
2601955|EPI_ISL_17813980|A/ABUDHABI/UAE/0004555/2022|HA 1737    3123    0
2589091|EPI_ISL_17785701|A/Bangkok/P1601/2023|NA        1441    1696    0
2604128|EPI_ISL_17832066|A/California/122/2022|NP       1541    5630    0
2604398|EPI_ISL_17832843|A/Michigan/UOM10049390076/2023|MP      982     7760    0
```

```
2571447|EPI_ISL_17699895|A/Serbia/7001/2022|NS  865     415     0
*       0       0       0
```

Compare the result with the unfiltered bam file.

Consensus calling

- Consensus calling using BCFtools

1. Generate pileup file with `bcftools mpileup`.

```
#generate pileup
bcftools mpileup -Ou -B -Q5 --max-BQ 30 --max-depth 10000 --max-idepth 10000 \
-f ./sample01/sample01ref.fasta ./sample01/sample01_filt_sort.bam > ./sample01/info01.bcl &
```

`-Ou`: output as standard format, `-B`: do not recalculate the base alignment quality (BAQ), `-Q:` lower threshold BAQ, `--max-BQ`: upper threshold for BAQ, `--max-depth`: max coverage being considered when running pileup. Type `bcftools mpileup -h` for explanation of the full command, or go to https://samtools.github.io/bcftools/bcftools.html#mpileup.

Should the BAQ be recalculated? BAQ is the Phred-scale probability of a read base being misaligned. Recalculating the BAQ might improve the SNP detection, but it can take long time to finish calculating. It is important to put in mind that BAQ is not the same as QUAL in sam files. It would probably better to compare both approach for your data.

2. Call the variants and index the file using `bcftools call` and use the multiallelic caller (`-m`) and specifying the ploidy.

```
bcftools call -c -Oz --ploidy 1 ./sample01/info01.bcl \
> ./sample01/calls01.vcf.gz

bcftools index ./sample01/calls01.vcf.gz
```

`-c`: classic consensus caller, `--ploidy`: the ploidy of the organism, `-p`: p-value threshold.

Lets quickly take a look at the vcf file

```
zmore ./sample01/calls01.vcf.gz
```

```
#CHROM  POS     ID      REF     ALT     QUAL    FILTER  INFO    FORMAT  ./sample01/sample01_filt_sort.bam
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      1       .       T       .       44.5868 .       DP=1;MQ0F=0;AN=1;DP4=1,0,0
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      2       .       C       .       44.5868 .       DP=1;MQ0F=0;AN=1;DP4=1,0,0
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      3       .       A       .       50.5868 .       DP=1;MQ0F=0;AN=1;DP4=1,0,0
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      4       .       A       .       49.5868 .       DP=1;MQ0F=0;AN=1;DP4=1,0,0
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      5       .       T       .       50.5868 .       DP=1;MQ0F=0;AN=1;DP4=1,0,0
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      6       .       T       .       54.5868 .       DP=1;MQ0F=0;AN=1;DP4=1,0,0
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      7       .       A       .       213.589 .       DP=24;MQ0F=0;AN=1;DP4=1,23
2604131|EPI_ISL_17832066|A/California/122/2022|PB2      8       .       T       .       221.589 .       DP=79;MQ0F=0;AN=1;DP4=1,78
```

3. Normalize the indels - left align the variants and removing extra bases.

```
bcftools norm -f ./sample01/sample01ref.fasta ./sample01/calls01.vcf.gz \
-Ob -o ./sample01/calls01.norm.bcf
```

`-f`: reference fasta file, `-Ob`: output file as bcf file; `-o`: output directory/file.

4. Filter out the indels and sites with depth less than 10.

```
#filter variants - because ONT is known to have errors skewed towards indel, we will ignore the indel.
bcftools filter --IndelGap 5 -e 'TYPE="indel" || DP<10' \
./sample01/calls01.norm.bcf -Ov -o ./sample01/calls01.flt.vcf.gz

bcftools index ./sample01/calls01.flt.vcf.gz
```

5. Apply variants

```
#apply variants to create consensus sequence
bcftools consensus -f ./sample01/sample01ref.fasta --mark-del '-' -a "N" -H 1 \
./sample01/calls01.flt.vcf.gz > ./sample01/consensus01_temp.fa
```

`-f` : reference fasta file
`--mark-del` : mark deletion with certain character.
`-a` : mark missing base with certain character.
`-H 1` : use only the first allele, because our organism is haploid.

6. Replace the fasta header

```
#generate the new header
grep ">" ./sample01/consensus01_temp.fa | cut -d'|' -f4 | sed 's/^/sample01_/' > ./sample01/newheader01
```

In short this command is to find line containing ">", but parse the string after the 4th "|" sign, then add "sample01_" at the beginning of the line.

Creating a tab separated file containing the old and new headers.

```
#create a new file contain the new header and old header
paste ./sample01/ref.acc ./sample01/newheader01 > ./sample01/header01.txt
more ./sample01/header01.txt
```

Use seqkit replace to replace the fasta header

```
#replace the fasta header using seqkit
seqkit replace -p "(.+)" -r '{kv}' -w 0 -k ./sample01/header01.txt \
./sample01/consensus01_temp.fa > ./sample01/consensus01.fasta
```

Remapping, and annotation

First, lets remapped the reads to the consensus that we just generated.

```
minimap2 -ax map-ont ./sample01/consensus01.fasta \
./clean/sample01.fastq > ./sample01/sample01_remapping.sam
```

```
#SAM to BAM
samtools view -bS -F 2052 ./sample01/sample01_remapping.sam | \
samtools sort > ./sample01/sample01_remapping.bam
samtools index ./sample01/sample01_remapping.bam
```

```
freebayes -f ./sample01/consensus01.fasta -p 1 -P 0.01 \
```

```
./sample01/sample01_remapping.bam -Oz ./sample01/freebayes.vcf.gz
```

Cross-checking with `freebayes` found these additional two errors which located in homopolymer region.

```
#CHROM  POS ID  REF ALT QUAL
sample01_PB2  579 . GAAAAAAGAAGAACTCCGA GAAAAAGAAGAACTCCGA  13.4024
sample01_NA 137 . TCCCCCCCAAAT  TCCCCCAAAT 842.896
```

1nt deletion in position 150 of NA gene - in a homopolymer region.

1nt deletion in position 580 of the PB2 gene - in a homopolymer region.

```
bcftools index ./sample01/freebayes.vcf.gz
bcftools consensus -f ./sample01/consensus01.fasta --mark-del 'N' -H 1 \
./sample01/freebayes.vcf.gz > ./sample01/consensus01_edited.fa
```

Get depth and plot statistics

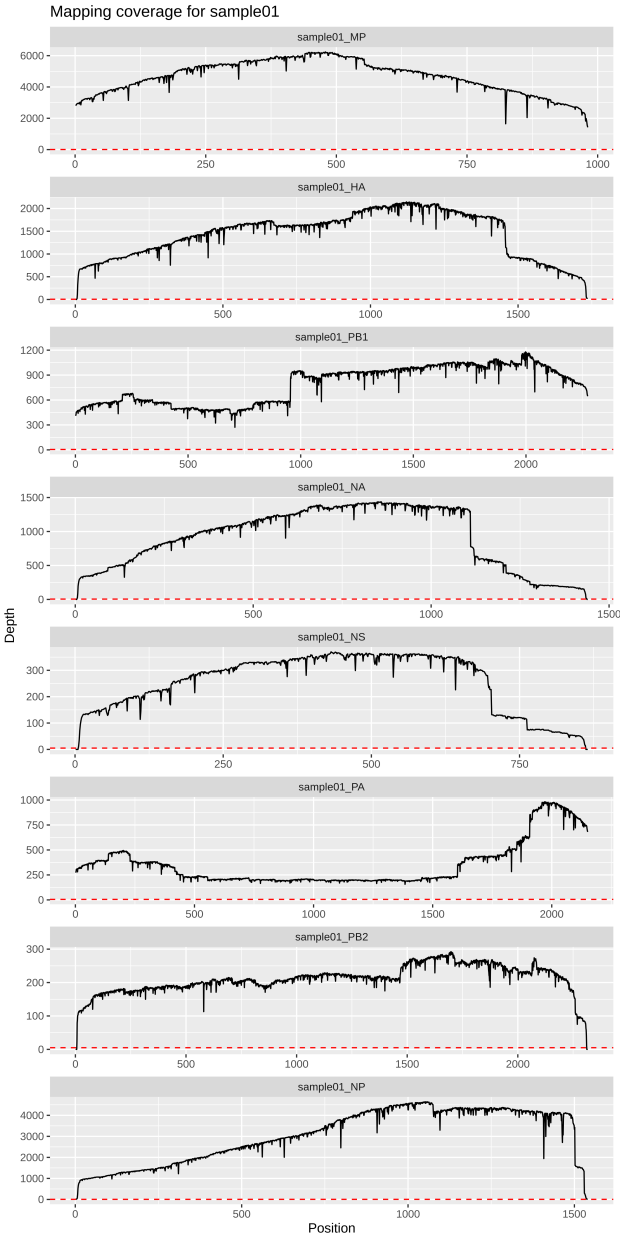
```
samtools depth -a -d 1000000 ./sample01/sample01_remapping.bam > \
./sample01/depth01.txt
```

`-a`: output all position, `-d`: maximum depth.

Use R

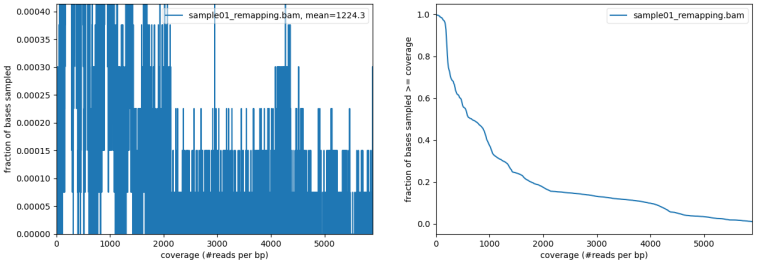
```
#make plot using ggplot (R)
#usage Rscript plotcov.rscript <depth information> <prefix> <output directory>
Rscript ./training_material/script/plotcov.rscript ./sample01/depth01.txt sample01 \
./sample01/coveragesample01.png
```

This produces a line graph with genomic position on the x axis and the coverage depth on the y axis.



Use deeptools

```
#use deepTools
plotCoverage -o ./sample01/deeptools01.png -b ./sample01/sample01_remapping.bam
```



The first graph generated by deeptools represents the found read coverage frequencies and also the mean coverage. The plot on the right will help you answer the question of how much of the genome fraction is covered by x reads?

Consensus calling with iVar

When calling consensus with bcftools for haplotype genome, the tool will only output the **majority of the base** in that position. Another program, called iVar has the function to call consensus. However, this program merged the eight genes as one long consensus, so we should use loop command to call consensus from each genes. The list of genes we used for alignment can be found in the ref.acc file.

Call for one gene:

```
samtools mpileup -aa \
-r "2604131|EPI_ISL_17832066|A/California/122/2022|PB2" --count-orphans \
--no-BAQ --max-depth 0 --min-BQ 0 \
./sample01/sample01_filt_sort.bam | \
ivar consensus -t 0.5 -q 20 -m 10 \
-n N -p sampl01_PB2
```

Call for all genes:

```
while read line1 line2; \
do samtools mpileup -aa \
-r ${line1} --count-orphans \
--no-BAQ --max-depth 0 --min-BQ 0 \
./sample01/sample01_filt_sort.bam | \
ivar consensus -t 0.5 -q 20 -m 10 \
-n N -p ${line2}; done < ./sample01/header01.txt
```

This commands generates pileup file using samtools, from all sites (`-aa`) in the genomic region given in `-r`, include anomalous reads - mapped reads with pair unmapped (`--count-orphans`), without calculating the base alignment quality (`--no-BAQ`), no limit for depth nor base quality.

The resulted pileup is then piped to `ivar consensus`. The consensus is called with this following criteria: 95% frequency, minimum quality of 20, minimum depth to call consensus is 10, and print N in region with coverage less than threshold. The consensus header will be prefixed by the string defined in `-p`.

When inputting from a TSV file, in a loop command, you can separate each column into different variables.

```
#merge all segments into one fasta file.
cat sample01*.fa >> consensus01_ivar.fasta
```

Minor variants

Calling minor variants

```
lofreq call \
-f ./sample01/consensus01.fasta -B \
--sig 0.01 --bonf dynamic \
--no-default-filter -o ./sample01/lofreqcall.vcf.gz \
./sample01/sample01_remapping.bam
```

`-f`: reference fasta file
`-B`: disable the use of BAQ;
`--sig:` P-Value cutoff;
`--bonf`: Bonferroni factor. 'dynamic' (increase per actually performed test) or INT ['dynamic'];
`-o`: output directory.

Filter

```
bcftools view -i 'QUAL>100'  ./sample01/lofreqcall.vcf.gz
```

Minor variant calling from `lofreq`.

```
#CHROM  POS     ID      REF     ALT     QUAL    FILTER  INFO
sample01_PB2    360     .       C       T       126     PASS    DP=186;AF=0.193548;SB=28;DP4=24,104,0,40
sample01_PB2    1741    .       G       A       105     PASS    DP=263;AF=0.068441;SB=3;DP4=89,153,5,14
sample01_PB1    339     .       C       T       168     PASS    DP=606;AF=0.153465;SB=144;DP4=159,337,0,98
sample01_PA     1610    .       G       C       720     PASS    DP=362;AF=0.198895;SB=8;DP4=156,114,49,23
sample01_PA     1611    .       G       T       733     PASS    DP=362;AF=0.201657;SB=7;DP4=163,120,49,25
sample01_PA     1618    .       T       C       561     PASS    DP=365;AF=0.205479;SB=8;DP4=164,123,50,25
sample01_HA     25      .       C       T       163     PASS    DP=688;AF=0.12936;SB=4;DP4=11,572,0,96
sample01_HA     79      .       G       A       570     PASS    DP=798;AF=0.142857;SB=26;DP4=40,619,0,114
sample01_HA     571     .       C       T       540     PASS    DP=1671;AF=0.14602;SB=284;DP4=347,982,2,270
sample01_NA     186     .       G       A       214     PASS    DP=695;AF=0.063309;SB=13;DP4=151,460,17,27
sample01_NA     1207    .       A       C       691     PASS    DP=510;AF=0.217647;SB=314;DP4=208,178,0,114
sample01_NP     567     .       T       C       493     PASS    DP=2734;AF=0.164594;SB=244;DP4=249,1855,0,506
sample01_NP     758     .       T       C       106     PASS    DP=3556;AF=0.06721;SB=112;DP4=292,2705,1,309
sample01_NP     775     .       C       T       148     PASS    DP=3649;AF=0.117841;SB=151;DP4=279,2769,4,540
sample01_NP     776     .       C       T       1103    PASS    DP=3653;AF=0.185327;SB=263;DP4=284,2428,4,769
sample01_NP     908     .       A       G       206     PASS    DP=4328;AF=0.020564;SB=11;DP4=315,2749,4,88
sample01_NP     1028    .       C       T       392     PASS    DP=4644;AF=0.085917;SB=100;DP4=371,3694,7,451
sample01_MP     15      .       C       T       124     PASS    DP=3182;AF=0.076996;SB=115;DP4=330,2561,1,264
sample01_MP     223     .       C       A       147     PASS    DP=5286;AF=0.083617;SB=1278;DP4=1660,2102,2,576
sample01_MP     225     .       G       A       106     PASS    DP=5295;AF=0.098017;SB=1232;DP4=1151,3118,493,162
sample01_MP     670     .       A       G       198     PASS    DP=4914;AF=0.075499;SB=508;DP4=2710,1712,402,26
sample01_MP     728     .       G       A       161     PASS    DP=4591;AF=0.055761;SB=484;DP4=2785,1488,284,1
sample01_MP     729     .       A       G       146     PASS    DP=4591;AF=0.05685;SB=436;DP4=2775,1486,289,5
sample01_MP     786     .       T       G       402     PASS    DP=4122;AF=0.093644;SB=576;DP4=2481,1123,417,3
sample01_MP     831     .       T       C       138     PASS    DP=3820;AF=0.021466;SB=11;DP4=2758,949,70,14
sample01_MP     966     .       G       A       512     PASS    DP=2639;AF=0.136036;SB=106;DP4=2081,153,384,0
sample01_NS     269     .       T       C       101     PASS    DP=316;AF=0.126582;SB=5;DP4=38,212,10,36
sample01_NS     698     .       A       T       844     PASS    DP=243;AF=0.395062;SB=112;DP4=43,93,0,96
```

Annotation

- **Online tools from NCBI: https://www.ncbi.nlm.nih.gov/genomes/FLU/annotation/**
- ORFfinder from NCBI: https://www.ncbi.nlm.nih.gov/orffinder/
- https://github.com/JCVenterInstitute/VIGOR4
- getorf from EMBOSS

▼ Other available pipelines to try.

1. IRMA - a streamlined pipeline from CDC

> IRMA: Iterative Refinement Meta-Assembler
>
> IRMA
>
> was designed for the robust assembly, variant calling, and phasing of highly variable RNA viruses.
> Currently IRMA is deployed with modules for influenza, ebolavirus and coronavirus.
>
> https://wonder.cdc.gov/amd/flu/irma/

```
#installation
conda create -n irma
conda activate irma
micromamba install -c bioconda irma
IRMA FLU-minion ./clean/sample01.fastq sample01_IRMA
```

2. epi2me

```
#supported by nanopore
#not tested in galaxy
#including polishing
conda create -n epi2me
```

```
conda activate epi2me
micromamba install -c bioconda nextflow epi2me singularity
nextflow run epi2me-labs/wf-flu --fastq ./fastq/ --sample_sheet ./samplesheet.csv \
--medaka_consensus_model r941_min_hac_g507 -profile singularity --out_dir ./epi2me/
```

```
#A csv file containing sample name and barcode is required
#example
barcode,sample_id,type
barcode02,H1N1_strain_A-PR-8-34,test_sample
barcode03,H1N1_strain_A-Virginia-ATCC1-2009,test_sample
barcode04,H3N2_strain_A-Virginia-ATCC6-2012,test_sample
barcode08,fluB-BY-massachusetts-2-2012,test_sample
barcode09,fluB_B-taiwan-2-62,test_sample
barcode10,fluB-lee-40,test_sample
barcode31,fluB-yamagata-florida-4-2006_1,test_sample
barcode91,fluB-yamagata-florida-4-2006_2,test_sample
barcode55,fluA_H3N2_A-wisconsin-15-2009,test_sample
```

Practice:

Get the consensus from sample02 and sample03

References for sample02

```
2236758|EPI_ISL_16023452|A/Skunk/AB/FAV-0416-02/2022|PB1
2416933|EPI_ISL_17008863|A/American_Wigeon/South_Carolina/22-000345-001/2021|PA
2236678|EPI_ISL_16023371|A/Red_Fox/AB/FAV-0835-13/2022|HA
2456496|EPI_ISL_17185877|A/bald_eagle/Kansas/W22-384/2022|NA
2298214|EPI_ISL_16555204|A/eared_grebe/North_Dakota/245625/2022|NP
2298216|EPI_ISL_16555204|A/eared_grebe/North_Dakota/245625/2022|MP
2473567|EPI_ISL_17260674|A/red_fox/North_Dakota/22-017061-002-original/2022|NS
```

References for sample03

```
2526534|EPI_ISL_17514170|A/mute_swan/Austria/23003984/2023|PB2
2526391|EPI_ISL_17514128|A/mute_swan/Austria/23002053/2023|PB1
2100259|EPI_ISL_13969429|A/White-fronted_goose/England/311038/2022|PA
2576630|EPI_ISL_17716084|A/mute_swan/Austria/23011165-001/2023|HA
2526488|EPI_ISL_17514140|A/mute_swan/Austria/23015300/2023|NA
2576623|EPI_ISL_17716084|A/mute_swan/Austria/23011165-001/2023|NP
2576625|EPI_ISL_17716084|A/mute_swan/Austria/23011165-001/2023|MP
2576624|EPI_ISL_17716084|A/mute_swan/Austria/23011165-001/2023|NS
```

Resources

1. nf-core: https://nf-co.re

   Ready to use NextFlow pipeline.

2. Epi2me: https://labs.epi2me.io