Date:	28 October 2021	PANDORA Pan-African Network For Rapid Research, Response and Preparedness for Infectious Diseases Epidemics	UCL
Version:	1.1	Authors:	Dr Linzy Elton, Professor Neil Stoker, Dr Sylvia Rofael

Oxford Nanopore bioinformatics pipeline: from basecalling to sequence alignment

Contents

Acronyms

BAM	Binary SAM file
CIFS	Common Internet File System
CLI	Command line interface
CPU	Central processing unit
GPU	Graphics processing unit
GUI	Graphical user interface
INDEL	Insertion or deletion
NGS	Next generation sequencing
ONT	Oxford Nanopore Technologies
OS	Operating system
SAM	Sequence alignment/map
SNP	Single nucleotide polymorphism
vcf	Variant call format
WSL	Windows subsystem for Linux

Definitions

Alignment	Using a reference genome to put your sequencing files in the correct order
Assembly	Piecing your sequencing files together without the use of a reference genome as a guide
Concatenate	Combining multiple .fastq files to create one file with all of your reads in. Note that these reads are not in order as they haven't been aligned/assembled
Container	A programme used to encapsulate a software component and the corresponding dependencies. Containers are easily packaged and designed to run anywhere

Contig	A contig (from the word contiguous) is a set of overlapping DNA segments that represent a consensus region of DNA. If you do not have good coverage, an assembly/alignment programme will create contigs of correctly ordered files (but doesn't have enough data to completely create one whole genome)
Coverage	The percentage of the whole genome that has been sequenced. For instance, in the example below, the sequenced contigs cover approximately 80% of the reference genome (at the top of the image), which means you would have sequenced 80% of the bases that make up the genome. You will not be able to identify genes, SNPs etc. in parts of the genome that you do not have coverage for
Demultiplex	Separating .fastq files into barcoded folders to separate your samples e.g. barcode01, barcode02 etc.
Depth	The amount of times a base within a genome has been sequenced. The greater the depth, the greater the confidence in the identity of the sequenced base. In the image below, the complete reference genome is at the top. Below it are the sequenced contigs (a series of overlapping DNA fragments). Three bases are represented, which have varying depth of reads. The first has 5 reads, the second only one and the third has three reads
Environment	A directory/folder that contains a specific collection of packages/programmes that you have installed. For example, you may have one environment with Medaka and its dependencies. If you change one environment, your other environments are not affected. Environments can be activated or deactivated environments, by switching between them.
N50	A statistic that defines assembly quality in terms of contiguity. If you have a set of contigs, the N50 is defined as the sequence length of the shortest contig at 50% of the total genome length
Quality control	Identifying how good your sequencing files are in terms of quality scores, coverage, depth etc.
Threads	Corresponds to the number of 'cores' your computer has. How many parallel processes your computer can do at once

1 Introduction

This document accompanies the tutorial video created for the BSAC AMR:COVID-19 project, in collaboration with PANDORA-ID-NET and the Centre for Clinical Microbiology at University College London. It explains the steps for processing Oxford Nanopore sequencing data, from basecalling to alignment.

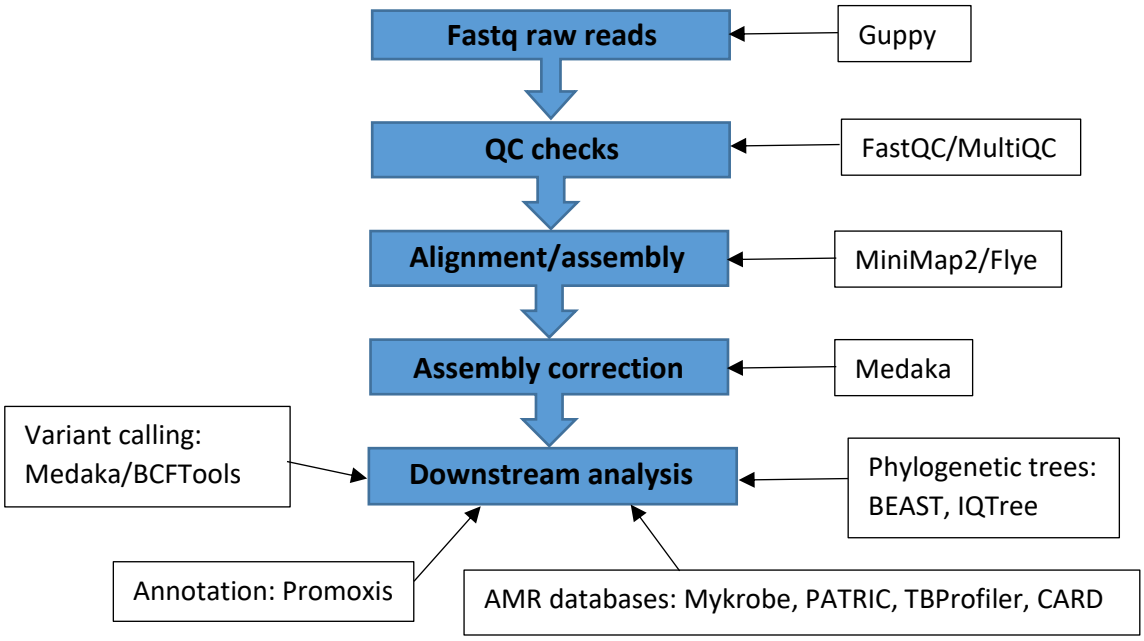
Note that some programmes mentioned within this document/tutorial are able to run on Windows, Linux and MacOS operating systems, whereas some programmes only work on certain ones. For the purposes of this training, this tutorial concentrates on Linux commands (Windows commands as an option if available).

Code highlighted in grey is for Linux platforms
Code highlighted in pink is for EPI2ME labs
Code highlighted in yellow is for Windows command prompt

Note that the programmes mentioned in this document are examples, there may be others that you could use (and that you may feel more comfortable with). The same is true of the code itself, whilst the commands mentioned here should work, there may be other versions that work too.

This is a fairly comprehensive list of sequencing programmes that you might also want to try.

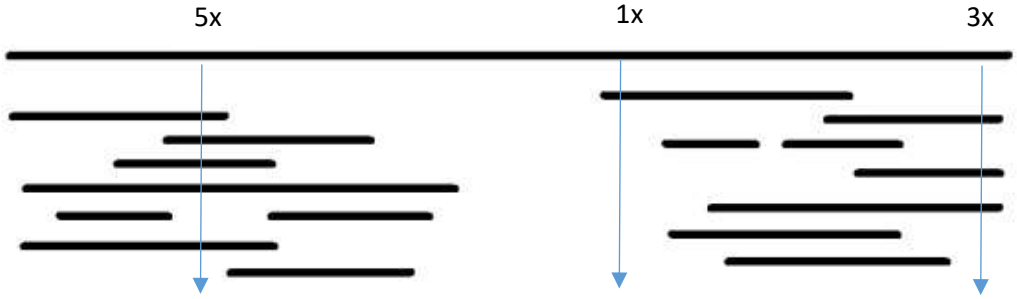
The Nanopore community bioinformatics page has lots of really useful information specifically for ONT sequencing data analysis.



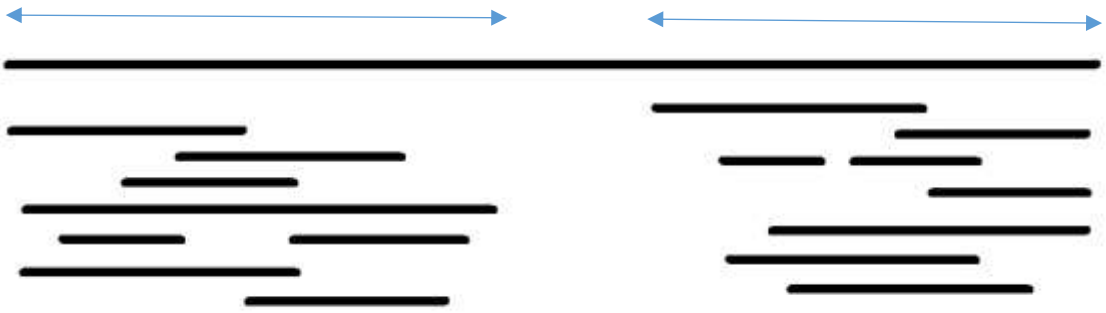
1.1 Depth vs coverage

Depth and coverage are both very important when it comes to sequencing, but they mean different things.

Depth: this is the amount of times a base within a genome has been sequenced. The greater the depth, the greater the confidence in the identity of the sequenced base. In the image below, the complete reference genome is at the top. Below it are the sequenced contigs (a series of overlapping DNA fragments). Three bases are represented, which have varying depth of reads. The first has 5 reads, the second only one and the third has three reads.



Coverage: this is the percentage of the whole genome that has been sequenced. For instance, in the example below, the sequenced contigs cover approximately 80% of the reference genome (at the top of the image), which means you would have sequenced 80% of the bases that make up the genome. You will not be able to identify genes, SNPs etc. in parts of the genome that you do not have coverage for.



This YouTube video explains the difference between coverage and depth well.

1.2 General useful hints

1.2.1 Creating links

Instead of typing the entire file address for programmes each time, you can link to them. There is the option to 'hard' or 'soft' link to them and you can read how to do that (for Linux) here. There are also lots of tutorials, e.g. here and here.

1.2.2 Saving and annotating your code

It's really useful (especially for your future self) to write down and annotate any code that you use, so that you can come back to it and use it again (and understand what you did and why!) You can read about how to annotate code using Notepad++ here.

1.2.3 Threads

Note that 'threads' (required as a parameter in a number of commands (usually -t) corresponds to the number of 'cores' your computer has. To identify how many cores your computer has follow these instructions.

Find out what is a thread (aka logical processor) is here.

How do I check how many CPU threads I have?

1.3 Using Windows Command Prompt and Linux (Windows subsystem)

1.3.1 Windows Command Prompt

Certain ONT bioinformatics programmes work with Windows, but not all of them.

Windows can only execute files with the extension .COM .EXE .BAT .CMD .VBS .VBE .JS .JSE .WSF .WSH .PSC1 so make sure they aren't zipped etc. (7 zip can remove certain extensions)

1.3.2 Ubuntu (using Windows Subsystem)

1.3.2.1 Windows Subsystem for Linux (WSL) Install

1. Open **Powershell** as an Administrator (this is different to command prompt)
2. Enable WSL

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

3. Download the chosen Linux application from Microsoft® store. This example will use Ubuntu 20.04. Other distros can be found from docs.microsoft.com/en-us/windows/wsl/install-manual

```
Invoke-WebRequest -Uri https://aka.ms/wslubuntu2004 -OutFile Ubuntu.appx -UseBasicParsing
```

Note: Powershell 6 uses basic parsing by default and will not require that parameter.

4. Install Linux Distro. Again, the example is the Ubuntu file downloaded in the previous step. Note: This will install Linux for the user account that powershell is running as (e.g. administrator). If powershell is not running as the customer, close the current session and begin another as the correct user.

```
.\ubuntu.appx
```

5. The installed distro is now available from the Windows start menu.
6. The user will be prompted to provide a username and password for their Linux install. This account will have full sudo rights.
7. Update Ubuntu Packages. This is a full blown operating system with system access.

```
sudo apt-get update
```

Notes: The next iteration of WSL is intended to offer a graphical user interface (GUI) and direct access to the graphics processing unit (GPU). Common Internet File Systems (CIFS) mounts do not work from WSL to network storage. A drive should be mapped on the host operating system (OS) and that drive mounted via WSL.

1.3.3 Linux commands

NB If you are using Linux via the WSL and you need to access your Windows directories and files, you may need to change directories to your local C: drive:

```
cd /mnt/c/
```

1.4 Installing programmes and containers

It's easier to install programmes using Conda or Docker. Note that all the config files of the programmes/tools that are installed through Conda will go into the folder where Conda is (if you are using WSL, they will go into a directory within the Linux directory system).

Docker containers act as separate environments so that there won't be conflicts in versions and dependencies between the applications running in separate containers, on the other hand this may be a problem in conda unless you create a separate environment file for the programmes to run in where you specify the dependencies.

Find out how to install Miniconda on a Windows machine here.

1.5 Introduction to EPI2ME Labs

EPI2ME Labs is a cloud-based programme, created by ONT. It combines a command line interface (CLI) with graphical user interface (GUI) and runs within Docker. It's useful as a place to start learning about bioinformatics and command line use, as each notebook shows you the commands you need to run it, but explains the reasoning too. Follow the Quick Guide to install and set up Docker and EPI2ME Labs.

NB if you want to import your own data into EPI2ME Labs, in Docker, go to settings>general and uncheck the "use the WSL 2 based engine" box. You will find that in the "resources" tab, you should now be able to set a folder that Docker and EPI2ME Labs can access.

NB. Windows 10 Home can only run Docker in WSL 2 mode because Windows 10 Home can't run Hyper-V. You don't need the file sharing options because all your files are already available to WSL 2.

2 Processing fastq files for downstream applications

ONT sequencing creates FAST5 (squiggle) files, but most programmes require fastq or fasta (bases) file formats. Read about different sequencing file formats here.

2.1 Introduction to Guppy

Guppy is a line command based programme created by ONT. It can be downloaded from their software downloads page (navigate down to Guppy). This will install all the needed folders and .exe programmes.

How to: Nanopore Guppy GPU basecalling on Windows using WSL2.

How to: Install Guppy on Linux

Guppy has a number of useful run programmes:

guppy_basecaller

- This will complete any basecalling that was not completed by MinKNOW during/after the sequencing.

- Often MinKNOW basecalling is very slow, so it's often better to stop it (allow files to save into the fast5_skip folder first!!) and then basecall manually using Guppy.

guppy_barcoder

- This can demultiplex (e.g. put fastq files into folders based on their attached barcodes)
- It can also de-barcode fastq files (when aligning etc. you don't want the barcode sequences to be part of it)

guppy_aligner

- Still under development, most people use minimap2

2.1.1 To finish basecalling

Programme to use: **Guppy**
Nanopore community basecalling protocol (login required)

Basic basecalling using windows command prompt (Windows):

```
/File_path/guppy_basecaller.exe
      --num_callers 2
      -i /file_path/fast5_folder
      -s /file_path/fastq_output_folder
      -c dna_r9.4.1_450bps_fast.cfg
```

Notes

- Need the folder address for the guppy_basecaller programme (or you can link to it)
- Need the number of parallel basecallers you want to create. Usually 2
- Need the input folder address
- Need the output folder address
- Need the config file or details of the kit and flow cell

More advanced Guppy basecalling command (Linux):

```
guppy_basecaller \
      --save_path  output/ \
      --config guppy/data/dna_r9.4.1_450bps_fast.cfg \
      --compress_fastq \
      --recursive \
      --barcode_kits "EXP-NBD104 EXP-NBD114" \
      --trim_barcodes \
      --min_score 60.000000 \
      --cpu_threads_per_caller 4 \
      --num_callers 1 \
      --qscore_filtering
```

2.1.2 To separate fastq files into barcode folders (demultiplexing)

Sometimes Guppy will put all of your fastq files into the same folder, and if you had multiple samples (e.g. you barcoded them), you will need to separate them before you process them further (as you need to remove the barcodes further down the pipeline).
Programme to use: **Guppy**

Nanopore community barcoding protocol (login required)

Demultiplexing using windows command prompt:

```
"c:\Program Files\OxfordNanopore\ont-guppy-cpu\bin\guppy_barcoder.exe"
      --I c:\home\user\Documents\Sequencing\test_data
      -s c:\home\user\Documents\Sequencing\test_data\Barcodes"
      -c configuration.cfg
      --barcode_kits SQK-RBK004
```

Notes

- If you don't include the kit used, you will output into all 96 available barcodes
- NB Nanopore says that the basic configuration.cfg file will work for most demultiplexing requests

2.1.3 To trim barcodes from reads

Your fastq files will still have the barcode and adapter sequences attached to them (from when you prepared the library) so you will need to trim them off, so that you are left with only the sequence from your original sample.

Programme to use: **Guppy**

Nanopore community trimming protocol (login required)

Barcode trimming using windows command prompt:

```
"c:\Program Files\OxfordNanopore\ont-guppy-cpu\bin\guppy_barcoder.exe"
      --I c:\home\user\Documents\Sequencing\test_data\fastq_pass
      -s
      c:\home\user\Documents\Sequencing\test_data\fastq_pass\trimmed
      "
      -r
      -c configuration.cfg
      --trim_barcodes
      --barcode_kits SQK-RBK004
```

Notes:

- If you don't include the kit used, you will output into all 96 available barcodes
- Barcode trimming is part of the guppy_barcoder programme. You need to add the argument –trim_barcodes to get it to trim

2.2 Quality control

Before you can use your data for post-analysis, you should check to see if your fastq files pass quality control checks. Programmes that can be used include **FastQC** (for individual sequenced

samples/isolates) and **MultiQC** (compile FastQC reports of multiple samples into one report ONT sequencing is more prone to errors than Illumina and so you have to be more careful. The cut off for depth should be at least 30, but 50-60 is a good minimum. You can get away with lower coverage, but the genome becomes more noisy. This noise can affect downstream applications e.g. variant calls (true SNPs or sequencing error) and phylogenetic trees (more noise = more things filtered out and a less accurate tree).

The five-quality control (QC) metrics every next generation sequencing (NGS) user should know.

Things to look for:

- How many reads are there?
- Percentage ATGC count
- N content (ambiguous bases that haven't been identified)
- GC content – compare the count/reads and theoretical normal distribution to ensure they are similar
- MultiQC shows QC from multiple samples to easily evaluate the QC of the sequencing data of a batch

There are a number of notebooks within EPI2ME Labs that can help you learn how to quality check your sequencing data.

2.2.1 Identifying read and base numbers

MinKNOW will identify how many reads and bases each of your barcodes have sequenced (so it's worth making a note of these from the screens). You can also identify these, as well as other quality control factors using the sequencing_summary.txt output file that should be in your sequencing run folder. The EPI2ME Labs **Basic_QC_Tutorial** and **Introduction_to_fastq_file** notebooks are useful for exploring your reads.

Before executing the following commands, you will need to navigate to the folder where your sequencing_summary.txt file is. Note that these are not the only ways to achieve this, there are other (possibly quicker) ways, but these are a few simple options.

2.2.1.1 Displaying the number of reads from your sequencing run

To display the number of reads for total sequencing run (printing the number of lines in a file). Each line in the file is a different read.

```
wc -l <file name>
```

wc = word count and -l = number of lines

2.2.1.2 Displaying the number of reads per barcode

To display the number of reads for each barcode (or print number of lines that contain X within a file). You will need to change the number of the barcode depending on which one you are searching for.

```
grep -c barcode01 <file name>
```

grep is a really useful searching tool. You can find out more about it and its arguments here.

-c = count

2.2.1.3 Displaying the number of bases per barcode

Useful for checking depth. Remember that some kits use PCR based reads (therefore they are all likely to be similar lengths) and others use native (e.g. straight from extraction) nucleic acid (which might range a lot in size, from a few hundred to hundreds of thousands of bp).

It's worth checking your sequencing.summary.txt file to make sure the columns mentioned here do correspond to the numbers. To display number of bases for total sequencing run:

```
awk '{sums[$23] += $14} END { for (i in sums) printf("%s %s\n", i, sums[i])}' <input file> | sort
```

$23 = column 23 which is alias (e.g. barcode number)
$14 = column 14 which is sequence_length_template (e.g. number of bases). This is the column we need to multiply to get total number of bases per barcode
Awk is another useful tool for manipulating data. You can read more about it here.

2.2.2 FastQC

FastQC aims to provide a simple way to do some quality control checks on raw sequence data coming from high throughput sequencing pipelines. It provides a modular set of analyses which you can use to give a quick impression of whether your data has any problems of which you should be aware before doing any further analysis. The main functions of FastQC are:

- Import of data from BAM, SAM or fastq files (any variant)
- Providing a quick overview to tell you in which areas there may be problems
- Summary graphs and tables to quickly assess your data
- Export of results to an HTML based permanent report
- Offline operation to allow automated generation of reports without running the interactive application

Find the FastQC manual here.
Find the FastQC web page here.
To reference FastQC: Andrews, S. (2010). FastQC: a quality control tool for high throughput sequence data. Available online here.
A helpful FastQC tutorial YouTube video can be found here.

For Linux, you will need to navigate to the folder and then run:

```
sudo apt install fastqc
```

if you get an error installing it, you may need to update your cache:

```
sudo apt update
```

To execute fastqc with Linux for all fastq files in a folder (navigate to fastqc folder):

```
fastqc /mnt/c/…filepath/*.fastqc -t 2
```

Note that -t 2 is the number of threads (2) used. You may find that Java runs out of memory, in which case you may need to increase the number of threads.

Executing the FastQC tool (Windows) – note you need to navigate into the folder containing the run_fastqc.bat file:

```
run_fastqc.bat <file_name.fastq.gz>
```

(you can run multiple fastq's at once by typing the folder location and *.fastq)

This blog outlines a potential solution if you come across a memory errorv.

2.2.3 MultiQC

MultiQC searches a given directory for analysis logs and compiles a HTML report. It's a general use tool, perfect for summarising the output from numerous bioinformatics tools. You can find the MultiQC website here.

To cite MultiQC, use this reference.
To download and install MultiQC using conda, follow this guide.

Install multiQC (Linux):

```
conda install -c bioconda -c conda-forge multiqc
```

To run multiQC, navigate to the folder you want to combine analysis reports for and run:

```
Multiqc .
```

2.3 QC using EPI2ME Labs

Notebook: Basic_QC_Tutorial

You will need to follow the 'test' tutorial first to get an understanding of how to use it. When it comes to using your own files, when it asks you to set a working directory, create a new one where you want to save all your work. Name it so that you know it corresponds to that sequencing run!

```
from epi2melabs import ping
pinger = ping.Pingu()
pinger.send_notebook_ping('start', 'basic_qc_tutorial')
```

```
# create a work directory and move into it
tutorial_name = "qc_Kerry1"
working_dir = '/epi2melabs/{}/'.format(tutorial_name)
!mkdir -p "$working_dir"
%cd "$working_dir"
```

You will then need to add the sequencing_summary.txt file that corresponds to that run into the tutorial. Make sure you are 'in' the tutorial you want to work in on EPI2ME labs (navigate to it on the left hand side) and when prompted under the "data entry" section, type the file name in and press enter.

2.4 Manipulating file formats

2.4.1 Fastq

2.4.1.1 Combining (concatenating) fastq files into one file

Assembly programmes usually require a single fastq file, so you will need to concatenate them all into one. This can be done by navigating to the folder with your fastq files in and typing the following command into the Windows Command Prompt:

```
type *.fastq > barcode01.fastq
```

This will create a new fastq file called barcode01.fastq and will have all of the combined data from the fastq files within the folder. NB might want to move the concatenated fastq files into a "combined" folder, so they don't get muddled.

3 Assembling/aligning sequencing data using command line interfaces (CLIs)

Genomes will either need to be aligned/mapped to a reference genome (using programmes such as **MiniMap2**) or assembled as a de novo genome (using programme such as **Flye**).

Read about Assembly statistics (N50) here. Imagine that you line up all the contigs in your assembly in the order of their sequence lengths. You have the longest contig first, then the second longest, and so on with the shortest ones in the end. Then you start adding up the lengths of all contigs from the beginning, so you take the longest contig + the second longest + the third longest and so on — all the way until you've reached the number that is making up 50% of your total assembly length. That length of the contig that you stopped counting at, this will be your N50 number. The higher the N50 this indicates longer contigs and therefore better assembly.

3.1 Assembly using EPI2ME labs

Notebook: assembly_tutorial

NB this tutorial uses **Flye** to assemble (another option is **Canu**)

NB. that you will need at least 16GB RAM to do this.

Click on the 'assembly tutorial' and follow the steps. You can either use the example files, or utilise your own data

3.2 Aligning/mapping to a reference genome – MiniMap2

Minimap2 is a programme that allows you to align your fastq files with a reference genome. It runs using Linux or MacOS.

To learn how to install MiniMap2 (on Linux systems), click here.
To learn how to install MiniMap2 using Conda, click here.
You can find a list of MiniMap2 commands here.
If you want to reference MiniMap2, you can find the link here.

To run fastq files against a reference genome (Linux):

```
minimap2 <reference genome file> <fastq files> > <alignment.sam> -ax map-ont
```

Output format: Sequence Alignment Map (SAM) file

To convert SAM to BAM (Linux):

```
samtools view -bS alignment.sam > alignment.bam
samtools sort <input.bam> -o <output.bam>
samtools rmdup <input.bam> <output.bam>
```

This blog (from Illumina, but still relevant to ONT) explains some ways to visualise alignments.

3.3 De Novo assembly - Flye

Flye is a de novo assembler for single molecule sequencing reads, such as those produced by PacBio and ONT. It is designed for a wide range of datasets, from small bacterial projects to large mammalian-scale assemblies. The package represents a complete pipeline: it takes raw PacBio/ONT reads as input and outputs polished contigs. Flye also has a special mode for metagenome assembly.

Find Flye on GitHub here.
If you want to cite Flye, use this reference.

To install Flye using conda (Linux):

```
conda install -c bioconda flye
```

To execute Flye assembly command (Linux):

```
<flye_address> -t <num of threads> --nano-raw <fastq_raw_reads> -g 4.3m -o <output_file.fasta>
```

Note that Flye outputs fasta files.

3.4 Assembly polishing - Medaka

Medaka is a tool to create consensus sequences and variant calls from nanopore sequencing data. This task is performed using neural networks applied a pileup of individual sequencing reads against a

draft assembly. It outperforms graph-based methods operating on basecalled data and can be competitive with state-of-the-art signal-based methods whilst being much faster.

To cite Medaka, use this reference.
To learn how to install Medaka, check out the GitHub site.

To install medaka using conda (Linux):

```
conda create -n medaka -c conda-forge -c bioconda medaka
```

As a reminder the correspondence between flowcell type and pore type is:

Flowcell	Pore type	Medaka model prefix
FLO-MIN106D	R9.4.1	r941_min
FLO-MIN111	R10.3	r103_min
FLO-PRO002	R9.4.1	r941_prom

For more information about choosing the correct model, see the medaka documentation.

To run Medaka genome polishing (EPI2ME Labs):

```
!run medaka_consensus \
    -d "$output_folder"/flye/assembly.fasta -i "$input_fastq" \
    -o "$output_folder"/medaka \
    -t 8 -m r941_min_fast_g303
```

To run Medaka in Linux:
Note that you have to activate the medaka environment first:

```
conda activate medaka
```

You will see that it changes from (base) to (medaka):

```
conda create --name medaka
```

To run medaka polishing on Linux:

```
medaka_consensus \
    -d "$output_folder"/flye/assembly.fasta -i "$input_fastq" \
    -o "$output_folder"/medaka \
    -t 8 -m r941_min_fast_g303
```

```
medaka_consensus    -d    "$output_folder"/flye/assembly.fasta    -i
"$input_fastq" -o "$output_folder"/medaka -t 2 -m r941_min_fast_g303
```

Note that:

-d	the folder location of where Flye outputted your information
-i	the original fastq file you put into Flye
-o	where you want medaka to put the output file
-t	number of threads
-m	the model to use (note make sure you are using the correct flow cell/kit etc)

It will output it to the file and location /medaka/consensus.fasta (it will tell you this at the end of the run).

3.5 Genome quality assessment - Pomoxis quality analysis

The assess_assembly program from the **pomoxis** suite in Medaka can be used to obtain an alignment-based quality analysis of a genome assembly. The basic workflow is to split assembly contigs into fixed length chunks before aligning these to a reference sequence. These alignments are then analysed for errors.

To run assess_assembly only two arguments are required, a reference sequence and the assembly. The -p option below sets an output prefix, while -t sets the number of compute threads:

EPI2ME Labs command:

```
!assess_assembly
      -r references.fasta
      -i analysis/medaka/consensus.fasta \
      -p analysis/assessment/assm
      -t 8
```

The summary Q scores section of the output gives statistics of the errors of the fixed length alignment chunks expressed in the usual logarithmic fashion (−10log10($Perr$)) with the err_ont row giving the total error rate:

$$Perr\_ont=\frac{Nins+Ndel+Nsub}{Nmatch+Nins+Ndel}$$

4 Downstream analysis programmes

4.1 Online databases

One way of bypassing some of the CLI-basesd analysis is to utilise online databases to process your sequencing data. There are lots of different ones depending on what it is you plan to do with your data. They usually utilise the same/similar CLI programmes as previously described to run the analyses, but have a user-friendly GUI that controls it.

Advantages	**Disadvantages**
GUI-based: much easier for new bioinformaticians to use	Can sometimes only upload certain sized files
Often utilises fastq files and the database will process them for you	Can only do a limited range of analysis

Useful for certain applications (e.g. tuberculosis resistance gene identification)	Be careful what you upload (human samples will need ethical approval and be anonymised)
Don't usually need a high-powered computer to run them (done on server)	

Some examples of online databases include:

Name	Uses	Link
EPI2ME	Multiple workflows	EPI2ME dashboard
CARD	Antibiotic resistance gene identification	CARD database
Mykrobe	Mycobacterium drug resistance gene identification	Mykrobe website
ResFinder	Antimicrobial resistance in total or partial DNA sequence of bacteria	RedFinder website
TB Profiler	Tuberculosis drug resistance gene finder	TB Profiler website
PANGO Lineages	Track the transmission and spread of SARS-CoV-2, including variants of concern	PANGO website
NDARO	National Database of Antibiotic Resistant Organisms	NDARO website
PATRIC	AMR identification	PATRIC website

This workshop run jointly by JPIAMR and PHA4GE provides a really good, practical introduction to analysing AMR sequencing data.

4.2 Introduction to EPI2ME

EPI2ME is a cloud-based analysis programme that has multiple workflows for different pipelines. The EPI2ME platform from Metrichor allows users to perform real time biological analyses and gain actionable insights.

You will need to download a desktop agent, so that you can interact with the cloud based analysis platform. You can log into the online dashboard to look at the results of your analyses. Note that the ARMA (antimicrobial resistance mapping application) workflow utilises the CARD AMR database.

Download EPI2ME desktop agent here (Community log in required).
View the EPI2ME dashboard here.
This ONT Community pages lists common EPI2ME FAQs.
This Metrichor YouTube account has lots of useful tutorial videos.

4.2.1 Data ownership

If you are uploading your data to an online database or cloud-based processing programme, take into consideration the data ownership agreements that they have in place and check before you upload that you are happy with the regulations in place. For instance, for EPI2ME, they state that:
"Raw data remain owned by the user. Metadata remains owned by Metrichor Ltd. and its parent company Oxford Nanopore Technologies PLC and may be used for purposes including but not limited to Platform performance measurement and analysis, System performance measurement and analysis, Quality control, User support and User-facing analysis reporting".

4.3 Other downstream analysis

Once you have assembled your sequence, you can do further analysis on it. Please note that these are just examples of some of the (popular) programmes that can be used to process your sequencing data and is definitely not exhaustive.

4.3.1 Genome annotation

Your genome assemblies are still just A, T, G and C, if you want to visualise genes, introns, exons, chromosomes etc. you will need to annotate it.

Prokka is an example of command line software tool to fully annotate a draft bacterial genome in about 10 min on a typical desktop computer. You can find out more about Prokka here.

4.3.2 Variant calling

If you want to look for differences between isolates (e.g. SNPs). The output files can then be visualised in viewers.

4.3.2.1 Variant calling CLIs

Medaka, which we have already introduced earlier, can also be used to look for variations between two assemblies. You can find more information about Medaka variant calling here.

BCFTools is a more comprehensive CLI and has more options (but may also be more complicated). You can find more information on BCFTools here.

4.3.2.2 Genome viewers

Integrative Genomics Viewer (IGV) is a high-performance, easy-to-use, interactive tool for the visual exploration of genomic data. It supports flexible integration of all the common types of genomic data and metadata, investigator-generated or publicly available, loaded from local or cloud sources. You can find the IGV user guide here.

Artemis comparison tool (ACT) is a Java application for displaying pairwise comparisons between two or more DNA sequences. You can find out more about ACT here.

4.3.3 Phylogenetic trees

Genomes with genetic elements don't align well. Best to use a reference genome and align and build from around a matrix. This method speed-wise, as it filters out the sections in all sequences that are identical and so creates a tree based solely on SNPs without affecting the outcome.

As an introduction to phylogenetics read a not-so-long introduction to computational molecular evolution.

This tutorial on Phylogeny for the faint of heart is also a very good introduction.

There are many programmes and multiple approaches to building phylogenetic trees, some examples include:

4.3.3.1 Maximum likelihood

Generally maximum likelihood programmes are easier to use (with less parameters to work out).

RAxML is a commonly used programme. It's a fast maximum likelihood tree search algorithm that returns trees with good likelihood scores. You can find the step by step guide to RAxML here.

IQTree is another commonly used programme. The link to the IQTree website is here. There is also a web browser version for smaller projects.

FastTree infers approximately-maximum-likelihood phylogenetic trees from alignments of nucleotide or protein sequences. The link to the FastTree website can be found here.

4.3.3.2 Bayesian

There are a number of steps you need to go through to create a phylogenetic tree using this method.

BEAST is one example of this method (although it requires other programmes to process data (BEAUTi) and then view it afterwards (TreeAnnotator and FigTree). You can read the tutorials for BEAST here.

The Interactive Tree of Life (iTOL) website is a useful online tool for displaying and annotating trees.

4.3.4 Plasmid identification

Identifying plasmids can be useful (for instance if you are looking for AMR genes which are often found on plasmids).

Deeplasmid is one programme you could use. This paper discusses Deeplasmid, a programme that separates plasmid from chromosomal sequencing data. You can find resources and the start guide for Deeplasmid here.

MOB-suite is a set of software tools for clustering, reconstruction and typing of plasmids from draft assemblies. Information on MOB-suite's GitHub pages can be found here.