De novo Assembly

In the past decade, next-generation sequencing (NGS) has become an important tool for biological research. It is a powerful tool that has revolutionised many areas of research. NGS platforms can generate large amounts of sequence data in a relatively short time, making them well-suited for studies of genomes, transcriptomes, and epigenomes.
One of the important applications of NGS is whole-genome sequencing (WGS), which can be used to obtain the complete genome sequence of an organism. One challenge with WGS is that it can produce a large amount of data, making it difficult to assemble the genome. *De novo* assembly is one approach that can be used to overcome this challenge.
De novo assembly is the process of assembling a genome from scratch, without using any reference genome. This can be a challenge, since it requires correctly assembling millions of short DNA sequences (reads) that cover the entire genome. This approach has been used to discover new viruses, characterise known viruses, and study the evolution of viruses.
There are several steps involved in viral *de novo* assembly, including, quality control and read trimming, assembly, scaffolding of contigs, and gap filling.

De novo Assembly Practical

In this practical we will be using three different *de novo* assemblers i.e. Spades, IDBA_UD and ABYSS to assemble the SARS-CoV-2 genome from Illumina reads. This data is generated from a clinical sample using amplicon sequencing strategy (https://www.ncbi.nlm.nih.gov/sra/?term=SRR21065613).
Let us first create the project directory and download the required data.

```
mkdir -p ~/DeNovo/SARS-CoV-2

cd ~/DeNovo/SARS-CoV-2

curl -o SRR21065613.fq.gz https://trace.ncbi.nlm.nih.gov/Traces/sra-reads-be/fastq?acc=SRR21065613
```

Here reads are in zipped (compressed) format, unzip them using "gunzip"

```
gunzip SRR21065613.fq.gz
```

Do a quality check and trimming using trim_galore program.

```
trim_galore --illumina --length 50 -q 30 SRR21065613.fq
```

After running trim_galore you should see a new file (SRR21065613_trimmed.fq) in your working directory.
Next, create individual directories for Spades, IDBA_UD and Abyss along with Ref and Validate for reference sequence and validation, respectively.

```
mkdir Spades Abyss IDBA_UD Ref Validate
```

** If you're copying and pasting the commands from this pdf file, please take care with single and double quotes. They are displayed in a different manner in the pdf file.

De novo assembly using Spades

First go to the Spades directory you have created above

```
cd ~/DeNovo/SARS-CoV-2/Spades
```

Run spades assembly program with different k-mer sizes.

```
spades.py --careful -k 21,45,73,101 -s ../SRR21065613_trimmed.fq -o .
```

Here

- –careful : Tries to reduce number of mismatches and indels in the assembly
- -k : different k-mer sizes
- -s: single-end fastq read file
- -o: output location. Here we are saving in the current directory(.), we can specify any name.

The program will take some time depending on the cpu power and available memory to complete the job. If everything goes without any error, the final assembled contigs will be stored in **contigs.fasta** file. In this file, contigs are saved in a sorted manner, with the longest contig first. Remember, these contigs can be positive or negative strands. We have to check the orientation comparing them with the reference sequence.
TODO: Extract all the headers of contigs from "contigs.fasta" file. What is the length of the longest contig generated?

De novo assembly using IDBA_UD

Now let us try running IDBA_UD

```
cd ~/DeNovo/SARS-CoV-2/IDBA_UD/
```

Though it is an assembly program for high throughput reads, IDBA_UD works only with fasta files(not fastq). It will not take the quality of the reads into consideration while assembling. So... we have to convert reads from fastq to fasta and submit them to IDBA_UD.

```
fq2fa --filter ../SRR21065613_trimmed.fq reads.fa
idba_ud --mink=21 --maxk=51 step=5 -r reads.fa -o .
```

Here

- fq2fa: program to convert reads from fastq to fasta
- mink: minimum k-mer size
- maxk: maximum k-mer size
- step: k-mer size increment
- -r: input reads
- -o: output directory

The final results will be stored in contig.fa file

TODO: Again, extract all the headers of contigs from “contig.fa” file. What is the length of the longest contig generated?

De novo assembly using ABYSS

Go to the Abyss directory

```
cd ~/DeNovo/SARS-CoV-2/Abyss/
```

Run Abyss assembler

```
abyss-pe k=27 n=4 se='../SRR21065613_trimmed.fq' name=Abyss-27 np=1
```

Here

- k: k-mer size
- n: number of threads to use (adjust this value as per your system capacity)
- se: single-end input reads
- name: output name

As you notice from above, abyss assembler works with one k-mer at a time. Since we do not know the optimal k-mer size for our data it is advisable to use various k-mer sizes. Please use the below for loop to run the program with different k-mers.

```
for k in $(seq 45 10 75);
do
abyss-pe k=$k n=4 se='../SRR21065613_trimmed.fq' name=Abyss-$k np=1;
done
```

This will run abyss with k-mer sizes from 45 to 75 with the increment of 10 (we are using seq command here) and save the output in corresponding files. Here you might have noticed that each k-mer size produces different output. Let us combine all the contigs...

```
cat Abyss*-unitigs.fa > Abyss-contigs.fa
```

To get the assembly stats, run

```
abyss-fac *-contigs.fa
```

Assembly validation

Check the contigs generated from all the assemblers. Did any of them give (near)full-length contig? Let us check whether the contigs belong to SARS-CoV-2, if so where they map and what is the percentage of genome covered by them. For this we will be using the "nucmer" program to compare the contigs to a reference genome. First, download the SARS-CoV-2 reference sequence (NC_045512) from GenBank

```
cd ~/DeNovo/SARS-CoV-2/Ref/

wget -O SARS-CoV-2.fa https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nuccore
\&id=NC_045512\&rettype=fasta\&retmode=text

wget -O SARS-CoV-2.gff https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nuccore
\&id=NC_045512\&rettype=gff\&retmode=text
```

Here we have used **wget** program to fetch genome and gene features files of GenBank id NC_045512 from NCBI. Now let us go to Validate directory.

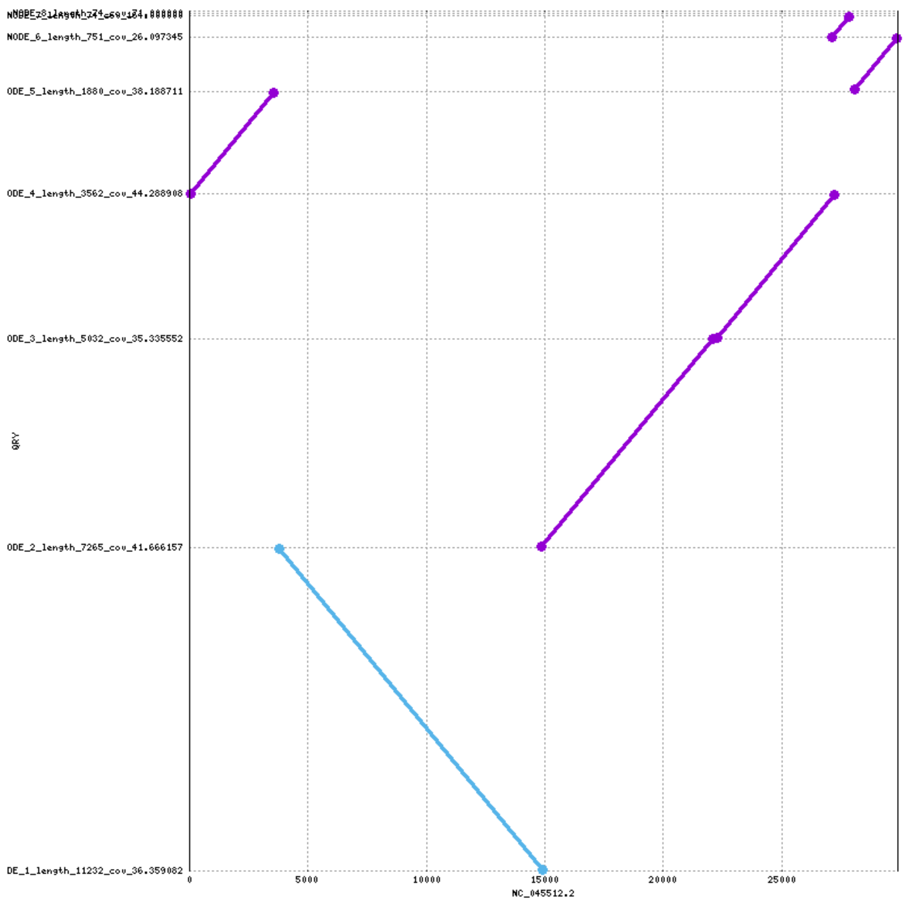
```
cd ~/DeNovo/SARS-CoV-2/Validate

nucmer ../Ref/SARS-CoV-2.fa ../Spades/contigs.fasta -p Spades

mummerplot -p Spades -t png Spades.delta

eog Spades.png
```

** Please see the end of the section to install missing program (gnuplot)



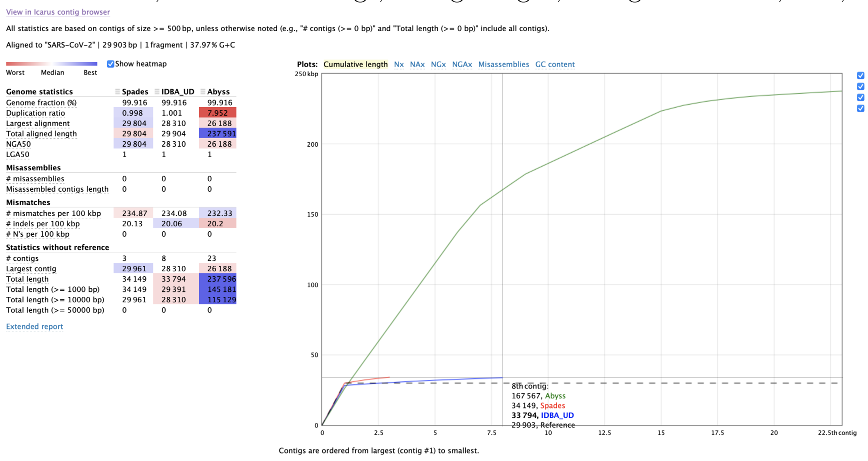
This will open a graph of nucmer mapping of all the spades generated contigs, their mapping location and orientation. Here X-axis is the reference genome location and Y-axis is the contigs mapped positions. If a contig is shown in ascending manner it is mapping to reference sequence positive strand. If it is shown in descending manner it is mapping to the negative strand of the reference sequence.
Next, run the nucmer mapping with Abyss and IDBA_UD contigs and check their orientation and coverage.
Now let us use quast program to validate the assemblies. (below command is a **single line** command)

```
quast.py  -l "Spades, IDBA_UD, Abyss" ../Spades/contigs.fasta ../IDBA_UD/contig.fa\
../Abyss/Abyss-contigs.fa -R ../Ref/SARS-CoV-2.fa --genes ../Ref/SARS-CoV-2.gff\
--single ../SRR21065613_trimmed.fq -t 1
```

This will create a directory called "quast_results" and saves the results.

```
cd quast_results/latest
firefox report.html
```

This will open the quast report in the firefox browser. Here you can see a detailed summery of assembly statistics like, number of contigs, contigs length, contig orientation, N50, L50 values etc.


View in Icarus contig browser

All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).

Aligned to "SARS-CoV-2" | 29 903 bp | 1 fragment | 37.97% G+C

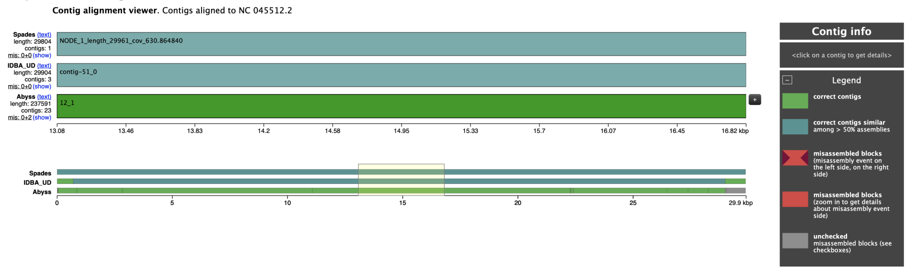
Worst Median Best ✓ Show heatmap

Genome statistics	Spades	IDBA_UD	Abyss
Genome fraction (%)	99.916	99.916	99.916
Duplication ratio	0.998	1.001	7.952
Largest alignment	29 804	28 310	26 188
Total aligned length	29 804	29 904	237 591
NGA50	29 804	28 310	26 188
LGA50	1	1	1
Misassemblies			
# misassemblies	0	0	0
Misassembled contigs length	0	0	0
Mismatches			
# mismatches per 100 kbp	234.87	234.08	232.33
# indels per 100 kbp	20.13	20.06	20.2
# N's per 100 kbp	0	0	0
Statistics without reference			
# contigs	3	8	23
Largest contig	29 961	28 310	26 188
Total length	34 149	33 794	237 596
Total length (>= 1000 bp)	34 149	29 391	145 181
Total length (>= 10000 bp)	29 961	28 310	115 129
Total length (>= 50000 bp)	0	0	0

Extended report

Contigs are ordered from largest (contig #1) to smallest.

Please click on the "View in Icarus contig browser" on top left corner of the page. It will take you to the contig mapping to the reference genome. Here you can see how the contigs are mapped and if there are any misalignmnets.


Contig alignment viewer. Contigs aligned to NC 045512.2

Spades (text) length: 29804 contigs: 1 mis: 0+0 (show)
IDBA_UD (text) length: 29904 contigs: 3 mis: 0+0 (show)
Abyss (text) length: 237591 contigs: 23 mis: 0+2 (show)

Contig info

<click on a contig to get details>

Legend

🟩	correct contigs
🟩	correct contigs similar among > 50% assemblies
🟥	misassembled blocks (misassembly event on the left side, on the right side)
🟥	misassembled blocks (zoom in to get details about misassembly event side)
⬜	unchecked misassembled blocks (see checkboxes)

Detail information of Quast output can be found here...

https://academic.oup.com/bioinformatics/article/29/8/1072/228832
https://academic.oup.com/bioinformatics/article/32/21/3321/2415080

Can you identify which assembler produced better results? and why? Do you see any misassembled blocks? Which program misassembled the contigs?
For me "NODE_1_length_29961_cov_630.864840" contig from Spades assembly seems to be covering nearly full-length of the genome however, it is in negative orientation. I am going to copy this contig and make a reverse complement to use it for mapping.

```
gedit  ~/DeNovo/SARS-CoV-2/Spades/contigs.fasta
```

Select and copy the first contig sequence and open a new file from gedit and paste this sequence. Save the file as "Denovo-Contig.fa". Run below command (**This is a single line command**) to generate a reverse complement sequence and save it in "Denovo-Contig-Complement.fa".

```
awk '{if($1!~/^>/) s=s $0; else print };END{system("echo " s "| rev|
tr '[ATGC]' '[TACG]'|fold -w 70 ");}'  Denovo-Contig.fa > Denovo-Contig-Complement.fa
```

Now this sequence (Denovo-Contig-Complement.fa) can be used with any read mapping program as a reference sequence.

Installing missing programs and fixing errors
Please execute this commands to install missing program (gnuplot) to fix problems with nucmer and mummerplot

```
sudo apt install gnuplot --fix-missing
```