# Lab 3: Genome Assembly With Short Reads

## Learning Objective

The objective of this lab is to help you understand the basics of genome assembly, using some popular tools and algorithms. You will learn how to run an assembler on real bacterial sequencing data. More importantly, you need to learn how to assess an assembler with a few relevant quality metrics. In this lab, several techniques to obtain assemblies of reasonable quality are provided.

## 1 Background Information

You are given a bacterial sequencing dataset to assemble: 500,000 reads from *V. cholerae* (genome size: $\sim 4$ Mbp) obtaining using Illumina HiSeq/MiSeq sequencer. The insert size is 335 bp, so the pairs are overlapping (you could try merging them, but that is optional). Note that these reads are paired-end.

In this section, the goal is to obtain a quick, initial assembly (which might be poor) of the *V. cholerae* genome using the provided reads in the directory called assembly. You can use the following recommended assemblers, or you can try more. Try to guess reasonable values for the mandatory parameters (e.g. the $k$-mer size) but do not over- think it.

### Recommended assembler

#### Velvet

Velvet is probably the most popular assembler. It works well with any sort of Illumina genomic data. However, Velvet requires a lot of memory (hundreds of GB RAM for 100+ Mbp genomes). Fortunately, the bacterial genome is relatively small, thus Velvet should not consume more than 1 GB of memory.

Website: `http://www.ebi.ac.uk/~zerbino/velvet/`

Manual: `http://www.ebi.ac.uk/~zerbino/velvet/Manual.pdf`

### Minia

Minia takes as input the Illumina data set and can assemble large genomes (mammalian-sized) on a PC. Minia creates contigs, in other words it does NOT use paired-end information to build scaffolds. For the dataset provided in this lab, the insert size is small, so an assembly formed only by contigs might be of sufficient quality.

Website: `http://minia.genouest.org`
Manual: `http://minia.genouest.org/files/manual.pdf`

### Spades

Spades is an assembler designed for small genomes. It does an excellent job at assembling bacteria, either multi-cell or single-cell data, and small metagenomes. It takes generally longer time and memory than other assemblers. Now the new version also supports IonTorrent and PacBio.

Website: `http://bioinf.spbau.ru/en/spades`
Manual: `http://spades.bioinf.spbau.ru/release2.5.1/manual.html`

### Other assemblers

Although we cannot give the complete list of the available assemblers here, all of the listed are commonly used and should work well on this data set.

**SGA**: `https://github.com/jts/sga`
**ABySS**: `http://www.bcgsc.ca/platform/bioinfo/software/abyss`
**Ray**: `http://denovoassembler.sourceforge.net/`
**SOAPdenovo 2**: `http://sourceforge.net/projects/soapdenovo2/`

## 2 Part I: Get familiar with assemblers

### Minia

(1) Install `minia` on your machine.

```
curl http://minia.genouest.org/files/minia-1.6067.tar.gz | tar xz
make -C minia-1.6067
sudo cp minia-1.6067/minia /usr/local/bin/
```

(2) Now make a temporary directory to host your data

```
mkdir minia-assembly
cd minia-assembly
```

(3) Generate the list of reads to assemble:

```
ls -1 read/trimmed/reads_?.trimmed..fastq.gz > list_reads.txt
```

(4) Now we are ready to assemble. Let's choose the following parameters for Minia: $k = 31$ and $min\_abundance = 5$, estimated genome size $4,000,000$ bp, and we will set the output prefix as *vcholerae_init*:

```
minia list_reads.txt 31 5 4000000 vcholerae_init
```

(5) If all go well, then your assembly should be contained in the file *vcholerae_init.contigs.fa*.

# 3 Part II: Evaluate the assembly with QUAST

## 3.1 Install QUAST

**Prerequisites**

- Python 2 (2.5+)
- Perl 5.6.0+
- g++
- make
- sh
- csh
- sed
- awk
- ar

If you use MAC, you need to install Xcode to make them available.

It is also highly recommended to install the Matplotlib Python library for drawing plots.

- Installation can be done by pip-installer:

  ```
  pip install matplotlib
  ```

- Or with the easy_intall Python module:

  ```
  easy_install matplotlib
  ```

- Or on Ubuntu just type:

  ```
  sudo apt-get install python-matplotlib
  ```

### 3.2 Install QUAST

```
wget https://download.sourceforge.net/project/quast/quast-3.1.tar.gz
tar xzvf quast-3.1.tar.gz
cd quast-3.1
./install.sh
./install_full.sh
```

### 3.3 Run QUAST

Run Quast on your output file as below. You need to specify a descriptive name for the output directory where QUAST will deposit a large number of output files. Use the parameter `--scaffolds` if your assembler produce scaffolds (Velvet and Spades do, but Minia doesn't).

```
quast.py -o output_directory assembly_file [--scaffolds]
```

Move into your QUAST output directory and examine the *report.txt* file. You may also take a look at the HTML report. A description of the output can be found at `http://quast.bioinf.spbau.ru/manual.html`.

Fill in the spreadsheet to compare your assembly with your peers. It will ask you for the name of the assembler you used (Velvet, Minia, etc.), and some statistics that can be found in the QUAST report.

`http://cbb.sjtu.edu.cn/course/final/assembly_stat.xlsx`

Take a look at the assembly quantities from your classmates. At this point, you may be tempted to re-run your assembly with better parameters. This is in fact what we do later.

**Questions**

- Assemblies start to be "acceptable" whne they have a contig $N50$ of 10+ kbp. Is it the case with your initial try?
- Based on the comparison, is there a single metric which reflects the overall quality of an assembly?

### 3.4 Optional: Assess the accuracy of your assembly

You may download the reference genome of *V. cholerae* from NCBI: `ftp://ftp.ncbi.nih.gov/genomes/Bacteria/Vibrio_cholerae_O1_biovar_El_Tor_N16961_uid57623`.

Concatenate the two *.fna* files to create the reference genome. Give this reference as an input to *QUAST* using the option `-R reference_file.fna`.

**Question**

- How many large/small misassemblies were made by the assembler of your choice?
- What is the actual genome coverage? Compare it to your initial guess. Why does this happen?

# 4 Part III: Improve the initial assembly

The goal of this part is to generate a much better assembly of *V. cholerae* given the reads that you have. Feel free to take a look at the parameters reported by your classmates in order to achieve the goal. Below, we highlight several essential steps.

## 4.1 Change a different assembler

With genome assembly, and maybe transcriptome assembly, running an alternative assembler is recommended for any serious project. Assemblers are designed very differently, and there isn't a single best assembler for all.

## 4.2 Use better parameters

Most assemblers, except for Spades, do not have "default" parameters, thus they cannot be executed with zero parameters. You might need to specify a $k$ value and probably other values.

For instance, Velvet generally works better if you execute *velvetg* with the parameters `-cov_cutoff auto -exp_cov auto` rather than without. Using a specific value of `-cov_cutoff` might even improve upon *auto* (often under the case you are assembling data which might contain 2 or more populations of DNA molecules). The value of $k$ also has a great influence on the result of *Velvet and Minia*. There exist some tools that can help you find optimal parameters.

**VelvetOptimizer** (included in velvet): `http://bioinformatics.net.au/software.velvetoptimiser.shtml`

VelvetOptimizer executes Velvet for many values of $k$ and picks the best result. It also optimizes the `cov_cutoff` parameter. It may take a long time to go.

```
/where/is/VelvetOptimiser.pl -s SmallestKmer -e LargestKmer \
-f '-fastq -shortPaired -separate s_1_1.fastq s_1_2.fastq'
```

**KmerGenie**: `http://kmergenie.bx.psu.edu`

Kmergenie examines the read files and returns what it believes is the best $k$ and *coverage cutoff* (`cov_cutoff` for Velvet, and `min_abundance` for Minia).

For our data set, both fastq files should be input to Kmergenie. To do this, create a text file with one read path per line. Then run Kmergenie with the name of the text file as input.

- Take a look at the generated HTML report, which contains a plot of the predicted genome size vs. $k$. The predicted best $k$ is the maximum value in this plot.
- Make sure that the plot is smooth and concave.
- Sometimes the curve is not smooth and has some small variations. Then it may be good idea to try a higher $k$ than the predicted one.
- Typically, if the estimated number of genomic $k$ is relatively stable for a range of $k$ and then drops, then use the highest $k$ as candidate.

```
mkdir kmergenie
ls -1 reads/trimmed/reads_?.trimmed.fastq > reads_list.txt
/where/is/kmergenie reads_list.txt -o kmergenie
```

### 4.3 Use more data

You were given only 500,000 reads, because that was sufficient to obtain a decent assembly of this organism. But the directory contains the full dataset. You may want to assembly it, to see whether adding more reads improves the assembly.

How many reads need to be sequenced to get the best possible assembly is a recurring question, and this is often difficult or impossible to predict prior to the actual assembly.