TECHNICAL REPORTS – AN INTRODUCTION TO GENOME ASSEMBLY

Disusun untuk memenuhi tugas UTS mata kuliah Bioinformatika Lanjut

Dosen : Dr. Risman Adnan, S.Si., M.Si.

Muhammad Ridho (2206130776)

PROGRAM MAGISTER MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS INDONESIA

DEPOK

2023

Contents

1. Introduction

Velvet is one of a number of *de novo* assemblers that use short read sets as input (*e.g.* Illumina Reads). The assembly method is based on the manipulation of de Bruijn graphs, via the removal of errors and the simplification of repeated regions.

For this tutorial, we have a set of reads from an imaginary *Staphylococcus aureus* bacterium with a miniature genome (197,394 bp). Our mutant strain read set was sequenced with the whole genome shotgun method, using an Illumina DNA sequencing instrument. From these reads, we would like to rebuild our imaginary *Staphylococcus aureus* bacterium via a *de novo* assembly of a short read set using the Velvet assembler.

We also have a sequence for a reference genome that we will use later in the tutorial.

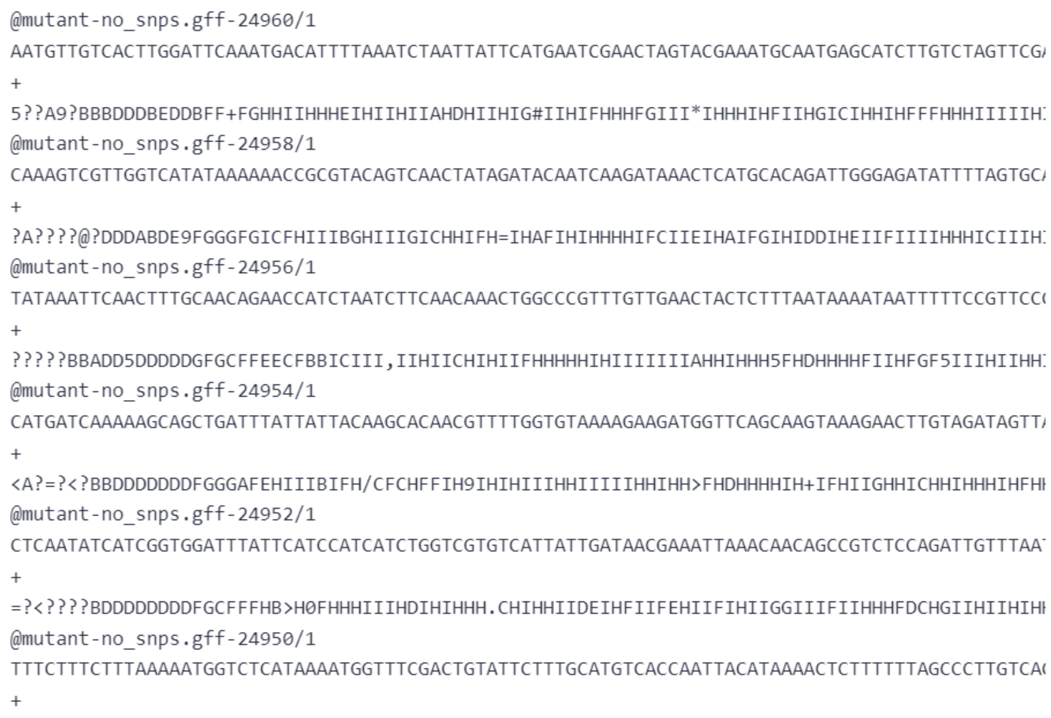
2. Get the Data

Hands-on: Getting the Data

1. Create and name a new history for this tutorial.
   Click the + icon at the top of the history panel.
   If the + is missing:
   - Click on the ⚙ icon (**History options**) on the top of the history panel.
   - Select the option **Create New** from the menu.
2. Import from Zenodo or from the data library the files:
   https://zenodo.org/record/582600/files/mutant_R1.fastq
   https://zenodo.org/record/582600/files/mutant_R2.fastq
   https://zenodo.org/record/582600/files/wildtype.fna
   - Copy the link location.
   - Click **Upload Data** at the top of the tool panel.
   - Select **Paste/Fetch Data**.
   - Paste the link(s) into the text field
   - Press Start
   - Close the window
3. Change the name of the files to **mutant_R1**, **mutant_R2**, and **wildtype.fna.**
   - Click on the **pencil icon** for the dataset to edit its attributes
   - In the central panel, change the **Name** field
   - Click the **Save** button

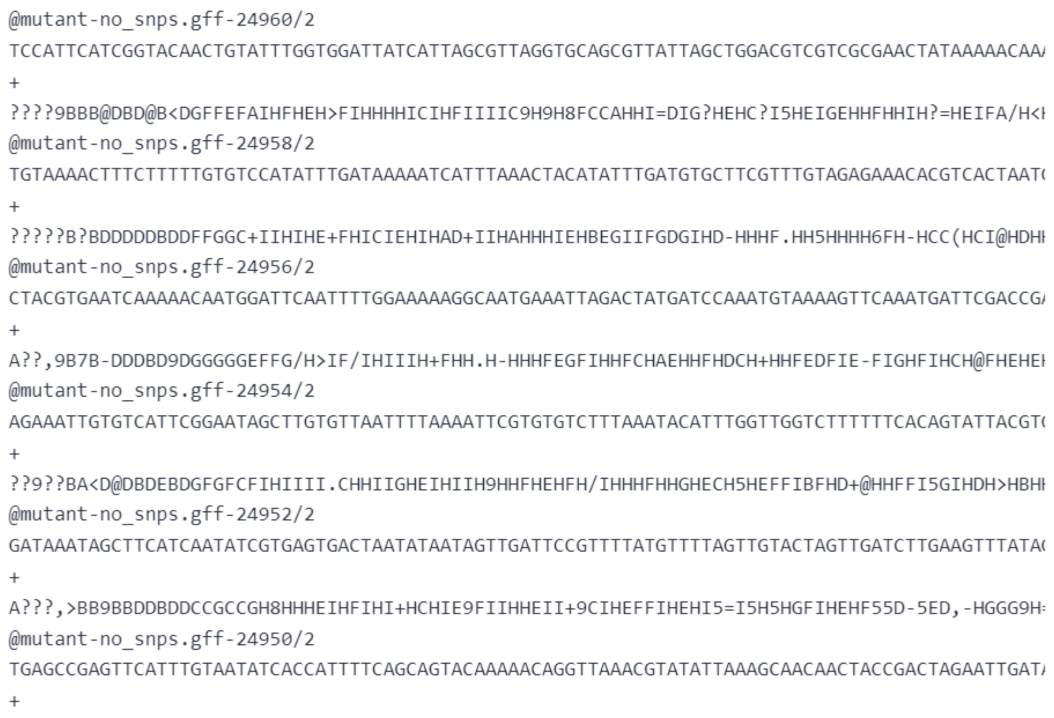
   As a default, Galaxy uses the link as the name of the new dataset. It also does not link the dataset to a database or a reference genome.
4. Inspect the content of a file.
   - Click on the **(eye) icon** next to the relevant history entry
   - View the content of the file in the central panel

   mutant_R1



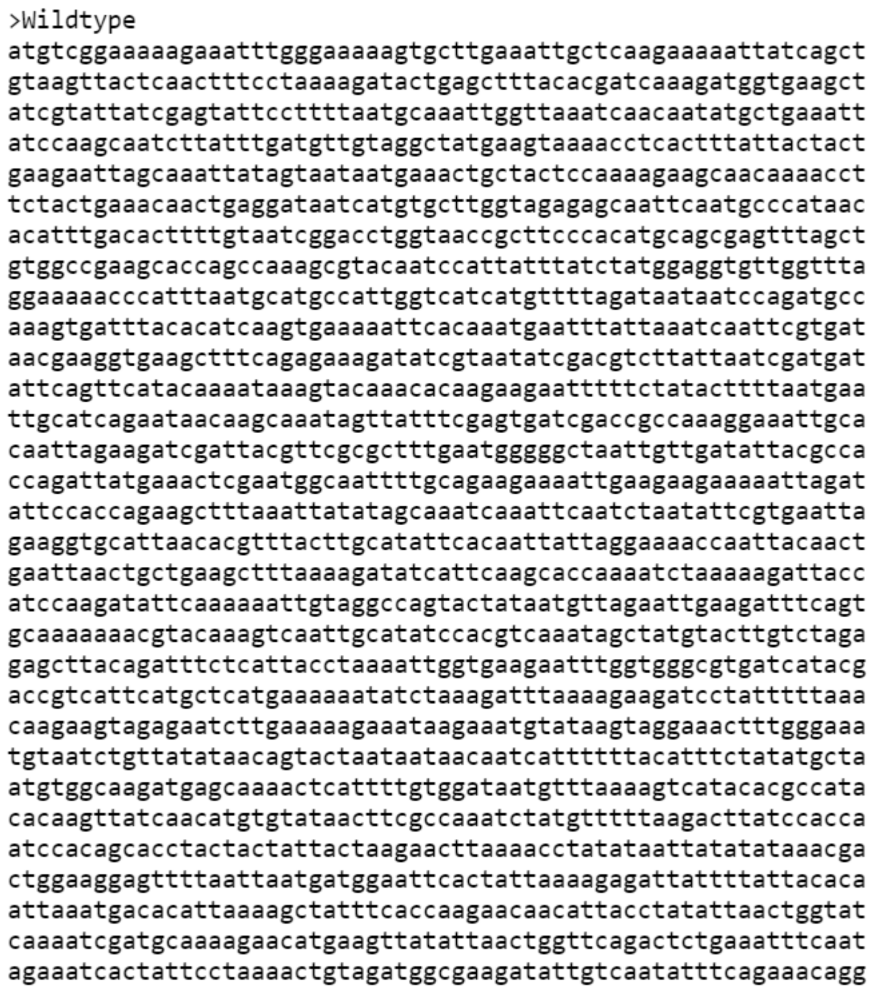
```
@mutant-no_snps.gff-24960/1
AATGTTGTCACTTGGATTCAAATGACATTTTAAATCTAATTATTCATGAATCGAACTAGTACGAAATGCAATGAGCATCTTGTCTAGTTCG
+
5??A9?BBBDDDBEDDBFF+FGHHIIHHHEIHIIHIIAHDHIIHIG#IIHIFHHHFGIII*IHHHIHFIIHGICIHHIHFFFHHHIIIIIH
@mutant-no_snps.gff-24958/1
CAAAGTCGTTGGTCATATAAAAAACCGCGTACAGTCAACTATAGATACAATCAAGATAAACTCATGCACAGATTGGGAGATATTTAGTGC
+
?A????@?DDDABDE9FGGGFGICFHIIIBGHIIIGICHHIFH=IHAFIHIHHHHIFCIIEIHAIFGIHIDDIHEIIFIIIIHHHICIIIH
@mutant-no_snps.gff-24956/1
TATAAATTCAACTTTGCAACAGAACCATCTAATCTTCAACAAACTGGCCCGTTTGTTGAACTACTCTTTAATAAAATAATTTTTCCGTTCC
+
?????BBADD5DDDDDGFGCFFEECFBBICIII,IIHIICHIHIIFHHHHHIHIIIIIIIAHHIHHH5FHDHHHHFIIHFGF5IIIHIIHH
@mutant-no_snps.gff-24954/1
CATGATCAAAAAGCAGCTGATTTATTATTACAAGCACAACGTTTTGGTGTAAAAGAAGATGGTTCAGCAAGTAAAGAACTTGTAGATAGTT
+
<A?=?<?BBDDDDDDDFGGGAFEHIIIBIFH/CFCHFFIH9IHIHIIIHHIIIIIHHIHH>FHDHHHHIH+IFHIIGHHICHHIHHHIHFH
@mutant-no_snps.gff-24952/1
CTCAATATCATCGGTGGATTTATTCATCCATCATCTGGTCGTGTCATTATTGATAACGAAATTAAACAACAGCCGTCTCCAGATTGTTTAA
+
=?<????BDDDDDDDDFGCFFFHB>H0FHHHIIIHDIHIHHH.CHIHHIIDEIHFIIFEHIIFIHIIGGIIIFIIHHHFDCHGIIHIIHIH
@mutant-no_snps.gff-24950/1
TTTCTTTCTTTAAAAATGGTCTCATAAAATGGTTTCGACTGTATTCTTTGCATGTCACCAATTACATAAAACTCTTTTTTAGCCCTTGTCA
+
```

mutant_R2



```
@mutant-no_snps.gff-24960/2
TCCATTCATCGGTACAACTGTATTTGGTGGATTATCATTAGCGTTAGGTGCAGCGTTATTAGCTGGACGTCGTCGCGAACTATAAAAACAA
+
????9BBB@DBD@B<DGFFEFAIHFHEH>FIHHHHICIHFIIIIC9H9H8FCCAHHI=DIG?HEHC?I5HEIGEHHFHHIH?=HEIFA/H<
@mutant-no_snps.gff-24958/2
TGTAAAACTTTCTTTTTGTGTCCATATTTGATAAAAATCATTTAAACTACATATTTGATGTGCTTCGTTTGTAGAGAAACACGTCACTAAT
+
?????B?BDDDDDBDDFFGGC+IIHIHE+FHICIEHIHAD+IIHAHHHIEHBEGIIFGDGIHD-HHHF.HH5HHHH6FH-HCC(HCI@HDH
@mutant-no_snps.gff-24956/2
CTACGTGAATCAAAAACAATGGATTCAATTTTGGAAAAAGGCAATGAAATTAGACTATGATCCAAATGTAAAAGTTCAAATGATTCGACCG
+
A??,9B7B-DDDBD9DGGGGGEFFG/H>IF/IHIIIH+FHH.H-HHHFEGFIHHFCHAEHHFHDCH+HHFEDFIE-FIGHFIHCH@FHEHE
@mutant-no_snps.gff-24954/2
AGAAATTGTGTCATTCGGAATAGCTTGTGTTAATTTTAAAATTCGTGTGTCTTTAAATACATTTGGTTGGTCTTTTTTCACAGTATTACGT
+
??9??BA<D@DBDEBDGFGFCFIHIIII.CHHIIGHEIHIIH9HHFHEHFH/IHHHFHHGHECH5HEFFIBFHD+@HHFFI5GIHDH>HBH
@mutant-no_snps.gff-24952/2
GATAAATAGCTTCATCAATATCGTGAGTGACTAATATAATAGTTGATTCCGTTTTATGTTTTAGTTGTACTAGTTGATCTTGAAGTTTATA
+
A???,>BB9BBDDBDDCCGCCGH8HHHEIHFIHI+HCHIE9FIIHHEII+9CIHEFFIHEHI5=I5H5HGFIHEHF55D-5ED,-HGGG9H
@mutant-no_snps.gff-24950/2
TGAGCCGAGTTCATTTGTAATATCACCATTTTCAGCAGTACAAAAACAGGTTAAACGTATATTAAAGCAACAACTACCGACTAGAATTGAT
+
```

wildtype.fna



```
>Wildtype
atgtcggaaaaagaaatttgggaaaaagtgcttgaaattgctcaagaaaaattatcagct
gtaagttactcaactttcctaaaagatactgagctttacacgatcaaagatggtgaagct
atcgtattatcgagtattccttttaatgcaaattggttaaatcaacaatatgctgaaatt
atccaagcaatcttatttgatgttgtaggctatgaagtaaaacctcactttattactact
gaagaattagcaaattatagtaataatgaaactgctactccaaaagaagcaacaaaacct
tctactgaaacaactgaggataatcatgtgcttggtagagagcaattcaatgcccataac
acatttgacacttttgtaatcggacctggtaaccgcttcccacatgcagcgagtttagct
gtggccgaagcaccagccaaagcgtacaatccattatttatctatggaggtgttggttta
ggaaaaacccatttaatgcatgccattggtcatcatgttttagataataatccagatgcc
aaagtgatttacacatcaagtgaaaaattcacaaatgaatttattaaatcaattcgtgat
aacgaaggtgaagctttcagagaaagatatcgtaatatcgacgtcttattaatcgatgat
attcagttcatacaaaataaagtacaaacacaagaagaattttttctatacttttaatgaa
ttgcatcagaataacaagcaaatagttatttcgagtgatcgaccgccaaaggaaattgca
caattagaagatcgattacgttcgcgctttgaatgggggctaattgttgatattacgcca
ccagattatgaaactcgaatggcaattttgcagaagaaaattgaagaagaaaaattagat
attccaccagaagctttaaattatatagcaaatcaaattcaatctaatattcgtgaatta
gaaggtgcattaacacgtttacttgcatattcacaattattaggaaaaccaattacaact
gaattaactgctgaagctttaaaagatatcattcaagcaccaaaatctaaaaagattacc
atccaagatattcaaaaaattgtaggccagtactataatgttagaattgaagatttcagt
gcaaaaaaacgtacaaagtcaattgcatatccacgtcaaatagctatgtacttgtctaga
gagcttacagatttctcattacctaaaattggtgaagaatttggtgggcgtgatcatacg
accgtcattcatgctcatgaaaaaatatctaaagatttaaaagaagatcctatttttaaa
caagaagtagagaatcttgaaaaagaaataagaaatgtataagtaggaaactttgggaaa
tgtaatctgttatataacagtactaataataacaatcatttttttacatttctatatgcta
atgtggcaagatgagcaaaactcattttgtggataatgtttaaaagtcatacacgccata
cacaagttatcaacatgtgtataacttcgccaaatctatgtttttaagacttatccacca
atccacagcacctactactattactaagaacttaaaacctatataattatatataaacga
ctggaaggagttttaattaatgatggaattcactattaaaagagattattttattacaca
attaaatgacacattaaaagctatttcaccaagaacaacattacctatattaactggtat
caaaatcgatgcaaaagaacatgaagttatattaactggttcagactctgaaatttcaat
agaaatcactattcctaaaactgtagatggcgaagatattgtcaatatttcagaaacagg
```

The reads have been sequenced from an imaginary Staphylococcus aureus bacterium using an Illumina DNA sequencing instrument. We obtained the 2 files we imported (**mutant_R1** and **mutant_R2**).

3. Evaluate the Input Reads

Before doing any assembly, the first questions you should ask about your input reads include:

- What is the coverage of my genome?
- How good is my read set?
- Do I need to ask for a new sequencing run?
- Is it suitable for the analysis I need to do?

We will evaluate the input reads using the FastQC tool. This tool runs a standard series of tests on your read set and returns a relatively easy-to-interpret report. We will use it to evaluate the quality of our FASTQ files and combine the results with MultiQC.

Hands-on: FastQC on a fastq file

1. **FastQC** (Galaxy version 0.73 + galaxy 0) with the following parameters:

- *"Raw read data from your current history"*: **mutant_R1.fastq** and **mutant_R2.fastq**

2. **MultiQC** (Galaxy version 1.11 + galaxy1) with the following parameters:
   - "*Results: Which tool was used to generate logs?*": **FastQC**
   - *Click "Insert FastQC output"*
   - *"Type of FastQC output?":* **multiple datasets, select the raw data files from FastQC**

MultiQC generates a webpage combining reports for FastQC on both datasets. It includes these graphs and tables:

- General statistics
  We need to know about the data for our analysis. In particular, we need to know the read lengths as it is important in setting the maximum k-mer size for an assembly.
- Sequence Quality Histograms
  Dips in quality near the beginning, middle or end of the reads may determine the trimming/cleanup methods and parameters to be used, or may indicate technical problems with the sequencing process/machine run.

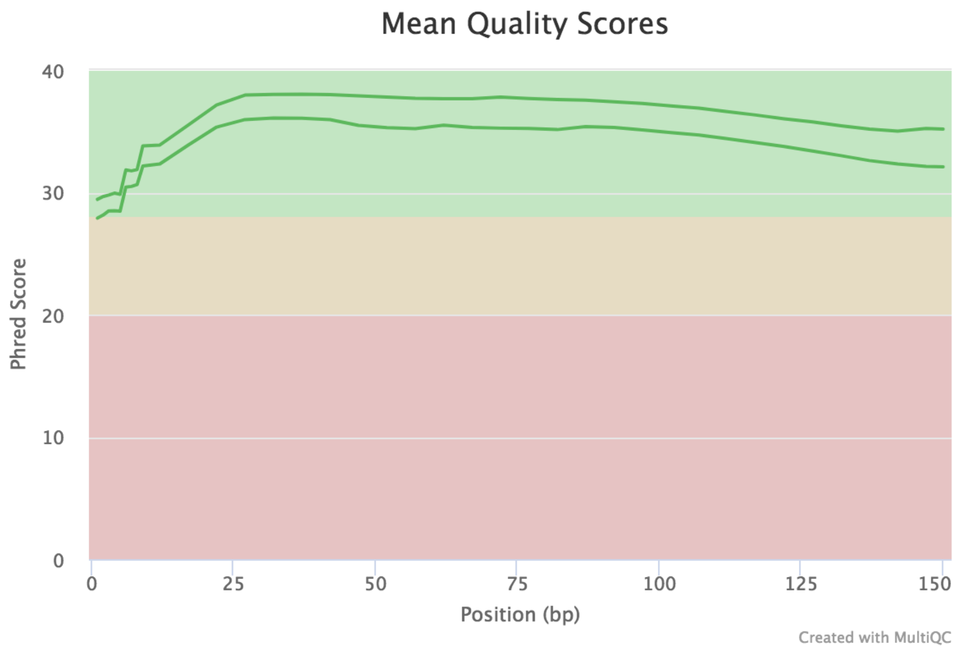


Figure 1: The mean quality value across each base position in the read

- Per Sequence GC Content
  High GC organisms tend not to assemble well and may have an uneven read coverage distribution.
- Per Base N Content
  The presence of large numbers of Ns in reads may point to a poor quality sequencing run. You will need to trim these reads to remove Ns.
- k-mer content

The presence of highly recurring k-mers may point to contamination of reads with barcodes or adapter sequences.

We won't be doing anything to these data to clean it up as there isn't much need. Therefore we will get on with the assembly!

4. Assemble Reads with Velvet

Now, we want to assemble our reads to find the sequence of our imaginary *Staphylococcus aureus* bacterium. We will perform a *de novo* assembly of the reads into long contiguous sequences using the Velvet short read assembler.

The first step of the assembler is to build a de Bruijn graph. For that, it will break our reads into k-mers, *i.e.* fragments of length *k*. Velvet requires the user to input a value of *k* (k-mer size) for the assembly process. Small k-mers will give greater connectivity, but large k-mers will give better specificity.

Hands-on: Assemble the reads

1. **FASTQ interlacer** (Galaxy version 1.2.0.1 + galaxy 0) with the following parameters:
   - *"Type of paired-end datasets":* **2 separate datasets**
   - *"Left-hand mates":* **mutant_R1.fastq**
   - *"Right-hand mates":* **mutant_R2.fastq**

   Currently our paired-end reads are in 2 files (one with the forward reads and one with the reverse reads), but Velvet requires only one file, where each read is next to its mate read. In other words, if the reads are indexed from 0, then reads 0 and 1 are paired, 2 and 3, 4 and 5, etc. Before doing the assembly per se, we need to prepare the files by combining them.
2. **velveth** (Galaxy version 1.2.10.3) with the following parameters:
   - *"Hash Length":* **29**
   - *"Input Files"*
     - Click on + *"Input Files"*
     - In "1: Input Files"
       - *"Choose the input type":* **interleaved paired end**
       - *"read type":* **shortPaired reads**
       - *"Dataset":* pairs output of **FASTQ interlacer**

   The tool takes our reads and break them into k-mers.
3. **velvetg** (Galaxy version 1.2.10.2) with the following parameters:
   - *"Velvet Dataset":* outputs of **velveth**
   - *"Using Paired Reads":* **Yes**

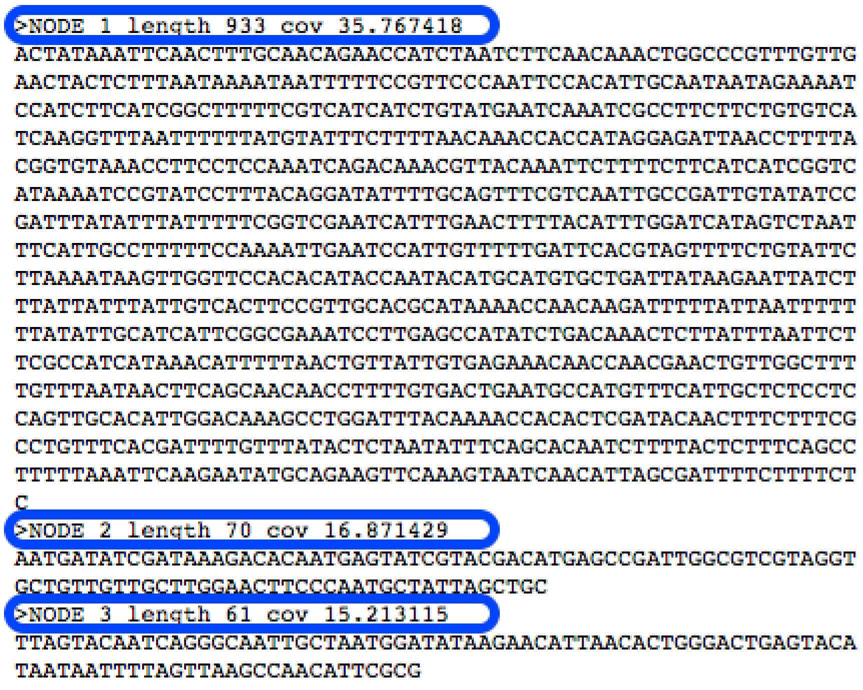
   This last tool actually does the assembly.

Five files are generated. We will look at the contigs file and the stats file:

- The **Contigs** file

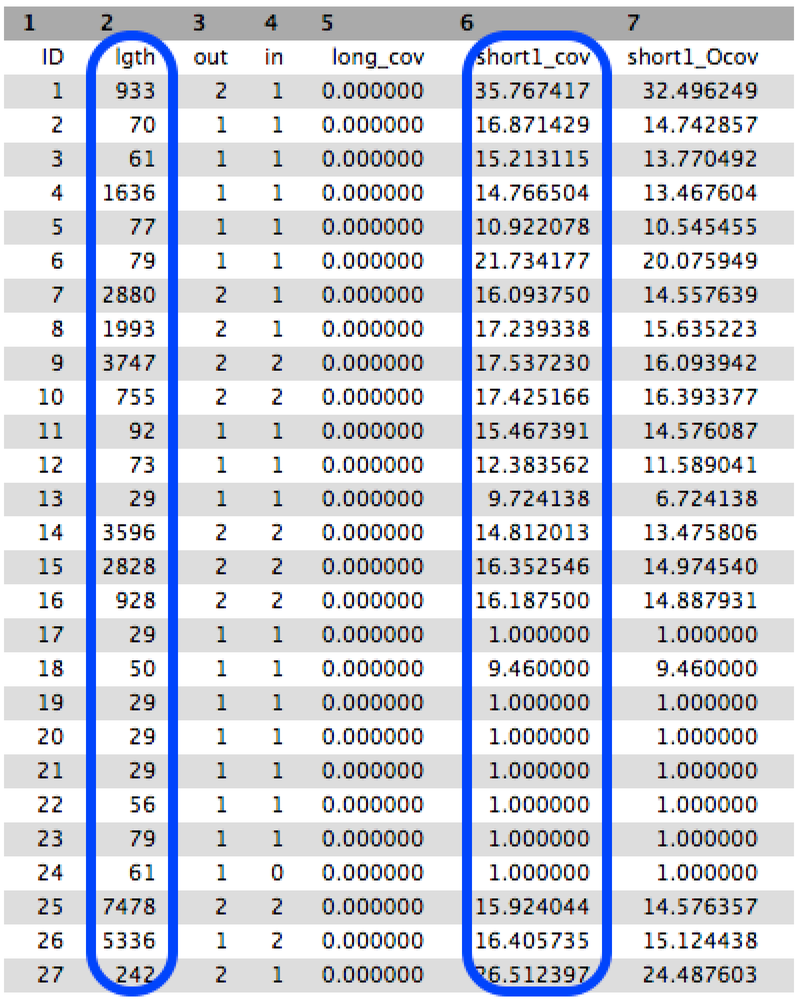
This file contains the sequences of the contigs. In the header of each contig, a bit of information is added:

- the k-mer length (called “length”): For the value of k chosen in the assembly, a measure of how many k-mers overlap (by 1 bp each overlap) to give this length.
- the k-mer coverage (called “coverage”): For the value of k chosen in the assembly, a measure of how many k-mers overlap each base position (in the assembly).
- Note that your results may look different to the example in the image below.



```
>NODE 1 length 933 cov 35.767418
ACTATAAATTCAACTTTGCAACAGAACCATCTAATCTTCAACAAACTGGCCCGTTTGTTG
AACTACTCTTTAATAAAATAATTTTTCCGTTCCCAATTCCACATTGCAATAATAGAAAAT
CCATCTTCATCGGCTTTTTCGTCATCATCTGTATGAATCAAATCGCCTTCTTCTGTGTCA
TCAAGGTTTAATTTTTTATGTATTTCTTTTAACAAACCACCATAGGAGATTAACCTTTTA
CGGTGTAAACCTTCCTCCAAATCAGACAAACGTTACAAATTCTTTTCTTCATCATCGGTC
ATAAAATCCGTATCCTTTACAGGATATTTTGCAGTTTCGTCAATTGCCGATTGTATATCC
GATTTATATTTATTTTTCGGTCGAATCATTTGAACTTTTACATTTGGATCATAGTCTAAT
TTCATTGCCTTTTTCCAAAATTGAATCCATTGTTTTTGATTCACGTAGTTTTCTGTATTC
TTAAAATAAGTTGGTTCCACACATACCAATACATGCATGTGCTGATTATAAGAATTATCT
TTATTATTTATTGTCACTTCCGTTGCACGCATAAAACCAACAAGATTTTTATTAATTTTT
TTATATTGCATCATTCGGCGAAATCCTTGAGCCATATCTGACAAACTCTTATTTAATTCT
TCGCCATCATAAACATTTTTAACTGTTATTGTGAGAAACAACCAACGAACTGTTGGCTTT
TGTTTAATAACTTCAGCAACAACCTTTTGTGACTGAATGCCATGTTTCATTGCTCTCCTC
CAGTTGCACATTGGACAAAGCCTGGATTTACAAAACCACACTCGATACAACTTTCTTTCG
CCTGTTTCACGATTTTGTTTATACTCTAATATTTCAGCACAATCTTTTACTCTTTCAGCC
TTTTTAAATTCAAGAATATGCAGAAGTTCAAAGTAATCAACATTAGCGATTTTCTTTTCT
C
>NODE 2 length 70 cov 16.871429
AATGATATCGATAAAGACACAATGAGTATCGTACGACATGAGCCGATTGGCGTCGTAGGT
GCTGTTGTTGCTTGGAACTTCCCAATGCTATTAGCTGC
>NODE 3 length 61 cov 15.213115
TTAGTACAATCAGGGCAATTGCTAATGGATATAAGAACATTAACACTGGGACTGAGTACA
TAATAATTTTAGTTAAGCCAACATTCGCG
```

- The **Stats** file
  This is a tabular file giving for each contig the k-mer lengths, k-mer coverages and other measures. Note that your results may look different to the example in the image below.



1	2	3	4	5	6	7
ID	lgth	out	in	long_cov	short1_cov	short1_Ocov
1	933	2	1	0.000000	35.767417	32.496249
2	70	1	1	0.000000	16.871429	14.742857
3	61	1	1	0.000000	15.213115	13.770492
4	1636	1	1	0.000000	14.766504	13.467604
5	77	1	1	0.000000	10.922078	10.545455
6	79	1	1	0.000000	21.734177	20.075949
7	2880	2	1	0.000000	16.093750	14.557639
8	1993	2	1	0.000000	17.239338	15.635223
9	3747	2	2	0.000000	17.537230	16.093942
10	755	2	2	0.000000	17.425166	16.393377
11	92	1	1	0.000000	15.467391	14.576087
12	73	1	1	0.000000	12.383562	11.589041
13	29	1	1	0.000000	9.724138	6.724138
14	3596	2	2	0.000000	14.812013	13.475806
15	2828	2	2	0.000000	16.352546	14.974540
16	928	2	2	0.000000	16.187500	14.887931
17	29	1	1	0.000000	1.000000	1.000000
18	50	1	1	0.000000	9.460000	9.460000
19	29	1	1	0.000000	1.000000	1.000000
20	29	1	1	0.000000	1.000000	1.000000
21	29	1	1	0.000000	1.000000	1.000000
22	56	1	1	0.000000	1.000000	1.000000
23	79	1	1	0.000000	1.000000	1.000000
24	61	1	0	0.000000	1.000000	1.000000
25	7478	2	2	0.000000	15.924044	14.576357
26	5336	1	2	0.000000	16.405735	15.124438
27	242	2	1	0.000000	26.512397	24.487603

5. Collect Some Statistics on the Contigs

This table is limitted, but we will now collect more basic statistics on our assembly.

Hands-on: Collect fasta statistics on our contigs

1. **Quast** (Galaxy version 5.2.0 + galaxy1) with the following parameters:
   - *"Assembly mode"*: **Individual assembly (1 contig file per sample)**
   - *"Use customized names?"*: **No**
   - *"Contigs/scaffolds file"*: contigs output of **velvetg**
   - *"Type of assembly"*: **Genome**
   - *"Use a reference genome?"*: **Yes**
   - *"Reference genome"*: **wildtype.fna**
   - *"Type of organism"*: **Prokaryotes**
   - *"Lower Threshold"*: **500**
   - *"Advanced options: Comma-separated list of contig length thresholds"*: **0,1000**

The HTML report reports many statistics computed by QUAST to assess the quality of the assembly:

- Statistics about the quality of the assembly when compared to the reference (fraction of the genome, duplication ratio, etc).
- Misassembly statistics, including the number of misassemblies.
  A misassembly is a position in the contigs (breakpoints) that satisfy one of the following criteria:
  - the left flanking sequence aligns over 1 kbp away from the right flanking sequence on the reference;
  - flanking sequences overlap on more than 1 kbp
  - flanking sequences align to different strands or different chromosomes
- Unaligned regions in the assembly.
- Mismatches compared to the reference genomes.
- Statistics about the assembly per se, such as the number of contigs and the length of the largest contig.

6. Conclusion

In conclusion, the introduction to genome assembly provided a comprehensive overview of the process, highlighting three key points. First, the use of the short read assembler Velvet was employed to assemble Illumina fastq reads into contigs. Second, the significance of the k-mer size as a crucial assembly parameter was demonstrated, showcasing its impact on the assembly outcome. Lastly, the preliminary analysis suggests the existence of exploitable patterns in the metric data relative to the chosen k-mer size. This indicates a potential avenue for optimization and fine-tuning of the assembly process. These foundational insights set the stage for a more detailed exploration of genome assembly in subsequent sections.