

# Assignment report, Advanced sequencing and genome assembly

MSc Applied Bioinformatics,

Marie Schmit

# Assembly

## Quality control and genome size

Illumina reads

A control quality is performed on the raw reads using FastQC. Read1 has 2 475 000 reads and a sequence length of 101. The mean quality of it’s reads is very poor, with a value of 17. No sequence is flagged as poor quality.

Chart

Description automatically generated

Figure Quality score across bases of first Illumina read

Its per sequence quality score is also too low, with a mean sequence quality of 17 (more than 1% error rate). The per base content of the first read is good, with little to no differences between bases: no bias was introduced in the reads.

The GC contents presents a warning as its distribution is slightly shifted, which could indicate a bias that does not depend on base position.

Chart, histogram

Description automatically generated

Figure GC content of first Illumina read

All the other metrics (per base N content, sequence length distribution, overrepresented sequence and adapter content) are good.

The results are the same for Illumina second reads. The quality per base sequence are problematic. Indeed, they are all the same and very low. Thought, no sequence is flagged as poor quality. A problem might have occurred with the sequencing. I will neither trim nor filter those data to improve their general quality, since all the values are the same and are poor. No data would be left after trimming or filtering.

A k-mer analysis is then performed with Jellyfisch, from MaSURCA on both short reads. 2 threads, 1G of elements hash are used. Different values of k-mer are tested: 19; 25; 30. The number of kmer is calculation with “count” function and a histogram is computed using the function “histo”. The results are plot with R.

|  |  |  |
| --- | --- | --- |
| 19mer, R1 | 25mer, R1 | 30mer, R1 |
|  |  |  |
|  |  |  |

The results are the same for the second reads.

The first major peak of the histogram, at count 1, is the error component. Errors boost the number of kmers appearing once. Apart from this error peak, the distribution presents one major peak, which is characteristic of a rather good data quality.

A correction is applied to try to decrease the error peak, with KmerFreq\_AR and Corrector\_AR functions (parameters: 1 thread, 33 ASCII shift). The k parameter for the correction is 15 for lighter computation.

|  |  |  |
| --- | --- | --- |
| 19mer, R1 after correction | 25mer, R1 after correction | 30mer, R1 after correction |
|  |  |  |

The number of reads went from 2 475 000 to after 2 378 197 reads correction. 100 000 reads were lost during the correction, but 96% of the reads were kept: the data was not too messy and a large number of reads could be corrected.

For 19 kmer, the error peak is still very high. Its density value has decreased, going from 5e7 before correction to 8e5 after correction, but the amelioration is small (only a difference of e2). The improvement is poorer with 25mer: the density went from 6e7 to 1e7. The same goes for 30mer, with a density going from 6e7 to 2e7 after correction.

The rest of the analysis will still be done with the correction, since the control quality with FastQC gave poor quality results. The correction process is repeated for R2. The number of reads goes from 2 475 000 before correction to 2 377 838 after.

The resulting k-spectrum has two peaks, one at half the abscise of the otherThe first, the hetero-peak, has a frequency of almost 15 000, while the homo-peak has a frequency of around 250 000. It indicates a low level of heterozygoty. This might correspond to heterozygotic alleles in diploid cells.

Chart

Description automatically generated

Figure In orange, a similar case of hetero-peak due to heterozygotic alleles in diploid cells (Nam Jin-Wu, 2016)

The genome will be considered as highly homozygote, so the second peak is used to calculate the genome size. Statistics on the peak are calculated with R, like the height of the peak or the number of kmers in it. The genome size was calculated with the following formula: G = N/C1 = R\*L\*((L-k+1)/(L\*C2)) where N is the number of base sequenced, L is the length of the reads, C1 is the base coverage and C2 is the kmer coverage.

|  |  |  |
| --- | --- | --- |
| 19mer spectrum for both reads (corrected) | 25mer spectrum for both reads (corrected) | 30mer spectrum for both reads (corrected) |
|  |  |  |
|  |  |  |
| Total of kmers in the peak: 338 767 481  Maximum value: 211 332 at position 69  Genome size = 338 767 481 / 69 = 4.91M  Single copy portion of the genome: 4 562 644[[1]](#footnote-1) (92.93%) | Total of kmers in the peak: 307 463 847  Maximum value: 223 641 at position 63  Genome size = 4.88M  Single copy portion of the genome: 4 678 372[[2]](#footnote-2) (95.86%) | Total of kmers in the peak: 308 190 218  Maximum value: 232 832 at position 58  Genome size = 5.3M  Single copy portion of the genome: 4 738 324[[3]](#footnote-3) (89. 17%) |

The k parameter providing the best percentage of estimation of the genome is 25. This value will be kept for the assembly: 25mers seem a good compromise between long and short k values . Indeed, long kmers help remove repetitions in the reads but with a decrease of coverage, while small kmers can remove errors and provide a better assembly when the coverage is low (Luo R, 2012).

The number of bases in reads 1, before correction, is 249.975.000. For a genome of 5M, the coverage is approximately 50, which is a very high coverage value. After correction, the number of reads is 218 052 444, so the coverage is approximatively 43, still a very high value. The genome estimation might be too low, since the coverage values are excellent. Or the sequencing depth was important.

PacBio reads

For this file, the number of sequences is 33 413. Their minimal length is 167 and the maximal is 246451. No sequences are flagged as poor quality and the percentage of GC content is 37.

The per base sequence quality is very poor, with a mean of 8.5. The quality is the same for almost all the reads, with a very light drop of 0.5 at the end. Since the quality is certainly bad but constant, no filtering nor trimming will be applied: all the values are bad.

Chart, bar chart

Description automatically generated

Figure Per base sequence quality, PacBio reads

The per sequence quality score is also flagged as failed. The mean quality value is around 8, and the maximal value is 10. It confirms the per base sequence quality: both are problematic.Chart, line chart

Description automatically generated

Figure Average quality per read (Quality score distribution over all sequences), PacBio reads

The per base content is also flagged as a fail. The percentage of adenine is for instance 30% higher than the percentage of guanine for the last bases (between 23500 and 24000).

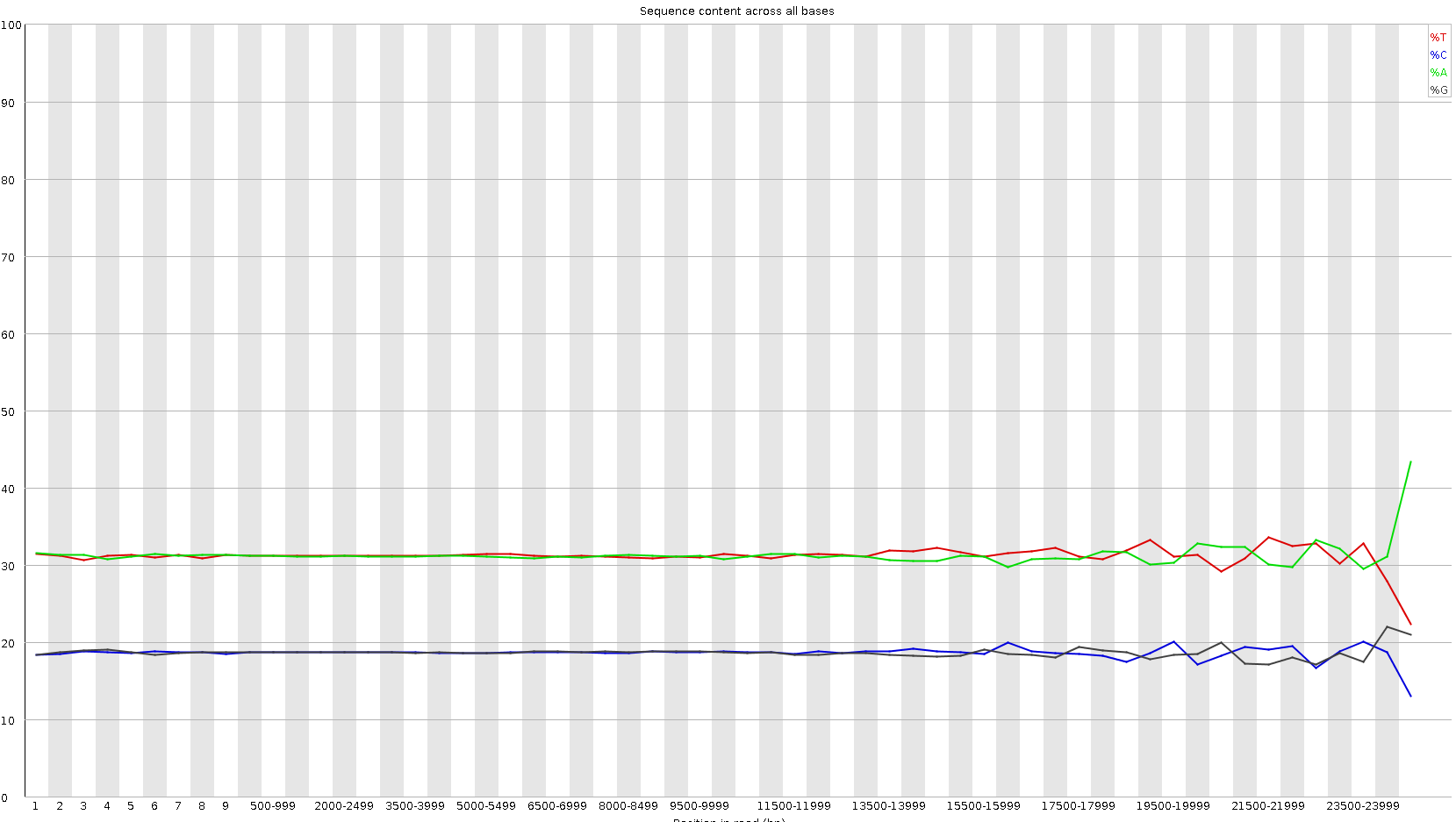


Figure Sequence content across all bases

As for the short reads, the per sequence GC content has a warning. The same goes for the sequence length distribution. The length of the sequences indeed vary, with a majority of sequences (5000) having a length between 1500 and 2500 bp.

Background pattern

Description automatically generated

Figure Sequence length distribution over all sequences for PacBio reads

The adapter and per base N contents, the sequence duplication level are good and there is no overrepresented sequences.

The same kmer analysis and spectrum are obtained for the long reads, with the different values of k. No other peak than the error one is visible.

|  |  |  |
| --- | --- | --- |
| 19mer, PacBio reads | 25mer, PacBio reads | 30mer, PacBio reads |
|  |  |  |

A correction is applied as previously on the Illumina short reads. The results are much better, with the apparition of, again, two peaks. Again, no peak appears. This might be due to the large number of errors that often occurs with a PacBio analysis. The distribution that is mostly represented would be the error one. This hypothesis seems confirmed, since "in PacBio, assuming a typical k-mer size of 16, only ~5% of the distintc k-mers from the reads are error free” ( (Carvalho AB, 2016)). The error peak thus contains more than 90% of the kmers, explaining the observed k spectrum.

## De-novo assemblies

Multiples assemblies will be tested, with three different values of kmer. Short and long reads assembler will be used, then hybrid assemblers. The best results are expected from the hybrid assemblies, since they combine the pros of both the short (more precise) and long reads.

### Short reads assembly using SOAP

Assembly

For the first assembly, only the short reads are used. The de novo assembler used is SOAPdenovo2, which uses De Bruijin graphs, and the performed steps are the following: pregraph and contig, since I do not have a long jump library. Pregraph calculates for instance the frequency of kmers, the number of vertex, of edges, etc. Contig uses those information to create the graph, by calculating for instance information about contigs.

SOAPdenovo-63mer is used to avoid using too much memory. In the configuration file, the insert size is 350bp, the reverse\_seq is 0 because the sequence does not need to be complementarily reversed, the asm\_flags is 1 for a contig assembly, a rank of 1 since only one library is used. 2 CPU are allocated. The corrected reads are used. The genome size that is used is 4.9M for 19 and 25mers, and 5.3M for 30mers.

Quality statistics

Gnx-tool is computed to display statistics about the assembly.

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs | 19mers | 25mers | 30mer |
| Total number of sequences | 125.392 | 95.827 | 67.916 |
| Total length of sequences | 7.978.233 bp | 7.974.697 bp | 7.424.920 bp |
| Shortest sequence length | 20 bp | 26 bp | 30 bp |
| Longest sequence length | 11.346 bp | 17.573 bp | 21.479 bp |
| Total number of Ns in sequences | 0 | 0 | 0 |
| N50 | 481 | 950 | 1.764 |

For the three kmers, the total length of the sequences is higher than the estimated size of the genome (approximately 7 or 8M bp). Also, the N50 is very low: there is probably a large amount of little fragmented contig. The total number of sequences is also rather high. 30mers is better because it has the highest N50 and the smallest number of sequences. The length of its sequences is also smaller, but still closer to the estimated size of the genome. However, those statistics are in overall bad.

### Short reads assembly using velvet

Velvet is also a de novo assembler that uses De Bruijin graphs. The function velveth is used for graph creation, velvetg for graph assembly. The parameters are shortPaired and separate, since short paired Illumina reads in two separate files are used. The short reads files used are non-corrected, because the corrected are indicated as comporting too few sequences by velvet. After correction, the number of reads in read2 is indeed smaller than in read1.

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs | 19mers | 25mers | 30mer |
| Total number of sequences | 354.059 | 235.638 | 200.766 |
| Total length of sequences | 14.707.282 bp | 14.044.733 bp | 14.223.431 bp |
| Shortest sequence length | 37 bp | 49 bp | 57 bp |
| Longest sequence length | 128 bp | 205 bp | 282 bp |
| Total number of Ns in sequences | 0 | 0 | 0 |
| N50 | 37 | 57 | 70 |

For this analysis, the N50 is again extremely small, while the length of the sequences is very high. The short read assembly is very bad for all the kmer.

### Short read using platanus

Platanus de novo assembly is ideal for highly heterozygous diploid genomes (Rei Kajitani, 2014). This assembler might help to deal with the hetero-peak present in the kmer spectrum, since it presented very good results for various levels of heterozygosity in de novo assembly contest Assemblathon2 (Rei Kajitani, 2014).

First, the option “assemble” of platanus is used, to create contigs. The parameters are 19 for 19mers, and the corrected fastq pair end short reads files. Then, the contigs are assembled into scaffhold with the function “assemble”. The average insert size parameter is 350bp. The reads are entered as inward, since they are paired-end reads (Kajitani). The gap closing is performed with the function gap\_close and the same parameters.

For this analysis, only 25mers, chosen at the first after the kmer analysis, will be tested (see other k values: 1). Also, 30mers have a better N50 (around 3000 after scaffolding) but it comes with 300 NS. Thus, 25mers are kept for this assembly.

|  |  |
| --- | --- |
| Contigs | 25mers, assembly |
| Total number of sequences | 2.980 |
| Total length of sequences | 5.141.223bp |
| Shortest sequence length | 64 bp |
| Longest sequence length | 77.759 bp |
| Total number of Ns in sequences | 0 |
| N50 | 19.664 |

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs without correction | 25mers, assembly | 25mers, scaffolding | 25mers, gap filling |
| Total number of sequences | 7.326 | 4.851 | 4.851 |
| Total length of sequences | 5.348.653 | 5.069.939 | 5.069.939 |
| Shortest sequence length | 58 bp | 100 bp | 100 bp |
| Longest sequence length | 9.906 bp | 9.856 bp | 9.856 bp |
| Total number of Ns in sequences | 0 | 0 | 0 |
| N50 | 2.093 | 2.133 | 2.133 |

The N50 is here better than with the other two assemblies. The total length of the sequences is also closer to the estimated genome size, and the total number of sequences is lower than with soap and velvet. The longest sequence length is approximatively five times longer, which is better. There is not improvement after gap closing, but scaffolding makes the sequences longer and the N50 larger. The correction applied to the file has considerably improved the statistics: without it, the N50 value was 2.093, the longest sequence 9.906 bp and the total number of sequences 7.326. However, scaffolding could not be done after it, since the corrected number of reads were no Platanus has the best results for short reads assembly. However, it’s N50 is still very low.

**Platanus is the best assembly tool for short reads, with 25mers.** This tool is the best candidate to be used later for hybrid assemblies.

### Long reads assembly using canu

The assembly is then done with PacBio long reads. Cd The first tested assembler is canu. Canu is tested with 25mers, also with the non-corrected reads, because the read coverage of the corrected pacbio reads is too low. The coverage with canu is too low to run canu, which may be due to an incorrect genomeSize or a poor quality of reads that could not be enough corrected.

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs without correction | 25mers, assembly | 25mers, scaffolding | 25mers, gap filling |
| Total number of sequences |  |  |  |
| Total length of sequences |  |  |  |
| Shortest sequence length |  |  |  |
| Longest sequence length |  |  |  |
| Total number of Ns in sequences |  |  |  |
| N50 |  |  |  |

### 

### Hybrid assembly with DBG2OLC

First, both the short and long reads are computed with this assembler. It builds a De-Bruijn graph for the short reads with the command “SparseAssembler”. The assembly is computed on corrected genome, with 25mer. The genome size is 5000000, the false kmer threshold is 1, as well as the false edge threshold and the skip size.

For long reads assembly, the overlap layout consensus is performed with an adaptive k-mer matching threshold to filter low quality reads (AdaptiveTh) of 0.0001, a fixed kmer matching threshold of 2 (KmerCovTh) and a minimum overlap score (MinOverlap), the minimal overlap length required to make a De Bruijin graph, of 20. The PacBio long read file is converted in fasta format.

The consensus is then called to combine data from both short and long reads. The bundled split\_and\_run\_sparc.sh script used is the one given in class. Blasr, a long read aligner is used to perform the consensus.

|  |  |
| --- | --- |
| Contigs without correction | 25mers, assembly |
| Total number of sequences | 79 |
| Total length of sequences | 905.952 bp |
| Shortest sequence length | 1.804 bp |
| Longest sequence length | 35.122 bp |
| Total number of Ns in sequences | 0 |
| N50 | 12.486 |

For this hybrid assembly, the N50 is still small. The length of the contigs are however larger than with the previous assembly, and the total number of sequences is smaller. However, the length of the sequence is very small, it makes only one fifth of the genome size.

After this first analysis, the parameters are tuned to find a better combination. One parameter among the following: NodeCovTh, k, AdaptiveTh, kmerCovTh, MinOverlap; is tuned at a time (see detailed results here: 2).

# Annexes

#### Results of short reads platanus analysis

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs without correction | 19mers, assembly | 19mers, scaffolding | 19mers, gap filling |
| Total number of sequences | 6.362 | 6.362 |  |
| Total length of sequences | 4.856.045 | 4.856.045 bp |  |
| Shortest sequence length | 100 bp | 100 bp |  |
| Longest sequence length | 5.783 bp | 5.783 bp |  |
| Total number of Ns in sequences | 0 | 0 |  |
| N50 | 1.178 | 1.178 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs without correction | 25mers, assembly | 25mers, scaffolding | 25mers, gap filling |
| Total number of sequences | 7.326 | 4.851 | 4.851 |
| Total length of sequences | 5.348.653 | 5.069.939 | 5.069.939 |
| Shortest sequence length | 58 bp | 100 bp | 100 bp |
| Longest sequence length | 9.906 bp | 9.856 bp | 9.856 bp |
| Total number of Ns in sequences | 0 | 0 | 0 |
| N50 | 2.093 | 2.133 | 2.133 |

|  |  |  |  |
| --- | --- | --- | --- |
| Contigs without correction | 30mers, assembly | 30mers, scaffolding | 30mers, gap filling |
| Total number of sequences | 6.848 | 3.702 | 4.851 |
| Total length of sequences | 5.274.590 bp | 5.014.987 bp | 5.069.939 |
| Shortest sequence length | 52 bp | 100 bp | 100 bp |
| Longest sequence length | 18.345 bp | 18.308 bp | 9.856 bp |
| Total number of Ns in sequences | 0 | 320 | 320 |
| N50 | 3.229 | 3.335 | 2.133 |

#### Results of DBG2OLC hybrid assembly tuning

Per default values are:

* NodeCovTh = 1
* EdgeCovTh = 1
* K = 25
* AdaptiveTh = 0.0001
* KmerCovTh = 2
* MinOverlap = 20
* RemoveChimera = 1

Tuning 1: k = 30

Tuning 2: NodeCovTh = 2

1. sum(as.numeric(df19[45:95,1]\*df19[45:95,2]))/69 [↑](#footnote-ref-1)
2. sum(as.numeric(df25[40:90,1]\*df25[40:90,2]))/63 [↑](#footnote-ref-2)
3. sum(as.numeric(df30[35:85,1]\*df30[35:85,2]))/58 [↑](#footnote-ref-3)