Q&A

# Reads quality control

* **What is the structure of a FASTQ file?**

A FASTQ file is composed by multiple entries, each of which consists of four lines. The first one is a sequence identifier, with information about the cluster and sequencing run (instrument, run number, flowcell ID, lane, tile...). The second line contains the sequence (nucleotide bases). Afterwards, there’s a + sign that works as a separator. Finally, the last line has Illumina Quality Scores.

* **How is the quality of the data stored in the FASTQ files? How are paired reads identified?**

Data quality is stored in the quality scores, there is one character per nucleotide which indicates how likely it is that the base assigned is actually correct. Paired reads can be identified because of numbered file names. There are two files that can be found for every compressed file (\*.fastqc.gz), and they are numbered. Also, they can be identified by information in their headers.

* **How is the quality of your data?**

Data quality was not optimal, for both DNA and RNA raw data. In the case of DNA, it was even necessary to indicate the software to ignore poor data quality when running the command (as stated in the Student Manual). RNA reads did improve quite a lot after trimming, which is the main reason why RNA trimmed reads were used.

* **What can generate the issues you observe in your data? Can these cause any problems during subsequent analyses?**

It is difficult to determine what caused the poor quality of the DNA data, it could have been due to experimental issues with sample preparation or processing. In the case of RNA, it could be due to the primers used or to contamination. While these problems are not enough to make our analyses unreliable, they could compromise some of our conclusions about mapping of reads to the genome or differential expression.

# Trimming / Reads preprocessing

* **How many reads have been discarded after trimming?**
* **How can this affect your future analyses and results?**

Since not too many reads were removed, trimming poor quality reads should improve the reliability of this analysis. Mapping of the reads and expression analyses should improve as a consequence of trimming.

* **How is the quality of your data after trimming?**

After trimming ERR2036629, the quality of the data

* **What do the LEADING, TRAILING and SLIDINGWINDOW options do?**

The LEADING option compares the start of a read with a user-defined minimum quality score. If it is below the threshold, a number of bases (set by user) will be removed. TRAILING does the same at the end of the read. SLIDINGWINDOW, as its name suggests, passes a dynamic panel of a particular length through the read, and trims whatever is below the quality requirements.

# Genome assembly

* **What information can you get from the plots and reports given by the assembler (if you get any)?**

Canu is an assembler that provides a lot of information to the user, both in the form of plots and reports. In its report, it informs the user of how many reads were corrected and trimmed, the amount of contigs and unitigs found and several quality characteristics of them. For example, 4 contigs were produced, no bubbles were found and 3946 sequences were unassembled. Plots for both contigs and unitigs are also generated.

* **What intermediate steps generate informative output about the assembly?**

Canu generates a lot of information about the intermediate steps it performs. Its output produces, for example, a correction and trimming folder with logs about corrected reads. It also generates a folder, called canu-logs, with information about steps performed by this assembler.

* **How many contigs do you expect? How many do you obtain?**

The amount of contigs obtained was expected to be somewhat variable. As Max, one of the teaching assistants of this course, explained in Slack, the assembly is a heuristic process and does not always produce the same output. The authors of the paper obtained two of them, while Canu produced four in my case. This was to be expected.

* **What is the difference between a ‘contig’ and a ‘unitig’?**

A unitig is a verified, or high-confidence contig. Contigs are sets of reads that are put together to cover a continuous, contiguous sequence. Unitigs are an unique assembly of a subset of fragments, there are no other possible choices for how that region could be assembled (ignoring the existence of repeats).

* **What is the difference between a ‘contig’ and a ‘scaffold’?**

A scaffold is a combination of ordered contigs that are connected together, a set that can also include gaps between said contigs. Contigs are, by definition, continuous.

* **What are the k-mers? What k-mer(s) should you use? What are the problems and benefits of choosing a small kmer? And a big k-mer?**

A k-mer is a subset of a sequence, of an assigned length k. According to Canu’s documentation, it is ‘a contiguous sequence of k bases’. Choosing a k-mer depends on the particularities of your data and what information you intend to obtain. Using a big k-mer makes it less likely to generate the exact same k-mer twice, so similar regions like repeats will be better assembled. In the other hand, it also makes it more vulnerable to sequencing errors. Small k-mers are a good idea for analysis of short reads or if there are not many repeats. It is also less computationally expensive.

* **Some assemblers can include a read-correction step before doing the assembly. What is this step doing?**

Canu is one of those assemblers. Its read-correction step consists on a quality assessment of the reads. If any segments are considered to have too poor quality to be assembled, they are trimmed from the reads. This is extremely convenient for the user, since they do not have to perform quality evaluations of the raw data before assembling.

* **How different do different assemblers perform for the same data?**

Only Canu was used for assembly in my analysis. According to the paper, authors used HGAP3 at their sequencing facility, and afterwards they used Gepard to inspect dot plots and Circlator to circularize it. Since I have not performed those steps, I cannot really make an informed comparison.

* **Can you see any other letter apart from AGTC in your assembly? If so, what are those?**

There are no other letters apart from AGTC in my assembly. For example, a possibility would have been N (any nucleotide) or W (indicating A or T, ‘weak’ nucleotides because of their two hydrogen bonds).

# Assembly evaluation

* **What do measures like N50, N90, etc. mean? How can they help you evaluate the quality of your assembly? Which measure is the best to summarize the quality of the assembly (N50, number of ORFs, completeness, total size, longest contig ...)**

NX measurements are quality parameters that give you the length of the shortest contig needed to cover X% of the genome. They are frequently used to evaluate assembly quality (especially N50) since high values would indicate that there is a long average length of contigs. However, what is considered a ‘high’ value depends of the particular case that is under study. Therefore, this parameter should not be used to compare different assemblies (of different genomes) in many cases. Assessing assembly quality can be tricky, and therefore, it is most useful to take into account different parameters and use reference literature of similar organisms.

* **How does your assembly compare with the reference assembly? What can have caused the differences?**

While reading Quast’s report, it was brought to my attention that my assembly contained one misassembly of a very large size (2563357 bp), which was the same as the size of the largest contig. However, other parameters seem to be fine, so it was unlikely that any big mistakes were committed during the assembly. While this can seem confusing at first, it is most likely a consequence of this genome being circular.

* **Why do you think your assembly is better/worse than the public one?**

I do not think that there are extremely significant differences between my assembly and the public one. Most of Quast’s plots show a good overlap between this assembly and the reference. The output from nucmer does not show any big issues either. However, it is true that this assembly could have been improved if I had more expertise.

# Annotation

* **What types of features are detected by the software? Which ones are more reliable a priori?**

Prokka detects genes, CDS (protein coding genes), rRNA (ribosomal RNA), signal peptides, tmRNA (transfer-messenger RNA), tRNA (transfer RNA), repeat regions and other ‘miscellaneous’ RNA. Prokka was designed for prokaryotic genome annotation and uses several highly reliable databases. Therefore, there should be no problem with the reported information. However, rRNA is very conserved, so it could be considered the most reliable feature reported.

* **How many features of each kind are detected in your contigs? Do you detect the same number of features as the authors? How do they differ?**

In my contigs, I detected 2572 CDS, 2635 genes, 6 rRNAs, 112 signal peptides, 1 tmRNA, 48 tRNAs, 1 repeat region and 8 miscellaneous RNA. The authors detected different numbers: 2541 total genes, 2486 CDS and 277 signal peptides. RNA data, however, was the same. The most likely explanation for this is that I have been performing this analysis in 2020, with more updated databases and a more updated version of Prokka than those available when this paper was published.

* **Why is it more difficult to do the functional annotation in eukaryotic genomes?**

Although it is not necessarily so, eukaryotic genomes have a tendency to be larger, and therefore, there is more room for error (statistically speaking). It is also common for eukaryotic genomes to contain many repeats, which complicates maintaining assembly quality, and therefore, annotation too. However, the most relevant difference in comparison to prokaryotic genomes is gene splicing, post-transcriptional modifications in which a single gene can code for multiple proteins.

* **How many genes are annotated as ‘hypothetical protein’? Why is that so? How would you tackle that problem?**

Prokka reported X hypothetic proteins. The reason why is because it is unable to find a function for them, since there are no similar genes reported. To solve this problem,

* **How can you evaluate the quality of the obtained functional annotation?**

A way of doing this, although not very practical, would be to manually revise the functional annotation and compare it to published information from databases/reported annotated genomes from similar organisms. For example, gene function could also be studied in wet lab experiments.

* **How comparable are the results obtained from two different structural annotation softwares?**

I did only use Prokka for annotation, since I did not manage to run eggNOGmapper successfully. However, if we consider a scenario in which two different structural annotation softwares were used, it would be important to evaluate if the numbers reported for different features were similar. If possible, annotations could be manually studied for consistency.

# Homology search

* **How relevant is the output format that you choose?**

There are several output formats available, depending on your particular needs or the software you will use afterwards. For example, I opted for the tabular format, since I needed it in order to use ACT (other formats were not recognized by it).

* **How do the resulting hits vary when you change the minimum e-value?**

The minimum e-value allows users to ‘filter out’ any non-relevant results, since BLAST could include in its output poorly aligned hits. The lower the e-value, the more significant the result is. In my case, I did not set the minimum e-value manually, since I blasted against a similar genome *(L. ferrooxidans).* However, it is important to keep in mind that when blasting against an entire database, e-value is extremely important to obtain quality results.

* **How is the alignment score calculated?**

The alignment score is calculated considering substitution and gap scores. BLAST uses BLOSUM matrices by default, although PAM is also an available option. Said matrices are used to compare one amino acid in one sequence to another, and determine how ‘likely’ it is that the substitution took place (it indicates divergence or similarity). Gap scores are determined by opening gaps in the alignment, and the penalty is extended by extending the gaps. The larger the gaps in the alignment, the worse the score will be. The combination of these two factors determines the alignment score.

* **How important is the number of threads when you blast against a database, or against a particular sequence?**

As in any computational process, a higher number of threads allow BLAST to use more resources and to work quicker. When blasting against a small local database or with small sequences, fewer threads are necessary.

# Mapping

* **What percentage of your reads map back to your contigs? Why do you think that is?**

This information was obtained from the \*.stats files, by calculating how many reads were mapped in comparison to the total reads obtained. There is a clear difference between continuous and batch samples. For continuous samples, around a 99.9% of the reads were mapped to the contigs, while it was around 70% for batch samples. There seems to be a clear indicator that treatment has affected the quality of the reads. It is possible that bioleaching (batch) samples were compromised due to contamination.

|  |  |
| --- | --- |
| **Sample** | **% reads map to contigs** |
| ERR2936629 | 99.811 |
| ERR2936630 | 99.844 |
| ERR2936631 | 69.706 |
| ERR2936632 | 72.087 |
| ERR2936633 | 99.861 |

* **What potential issues can cause mRNA reads not to map properly to genes in the chromosome? Do you expect this to differ between prokaryotic and eukaryotic projects?**

Most of the problems related to mRNA mapping would be expected to be more frequent in eukaryotic than in prokaryotic projects. The existence of introns could make reads of a same gene map in different positions of the genome, this could also happen if a gene is found twice. Gene splicing could make this issue even more complicated, since reads of different proteins could be mapped in the same place, confounding our results.

* What percentage of reads map to genes?
* How many reads do not map to genes? What does that mean? How does that relate to the type of sequencing data you are mapping?
* What do you interpret from your read coverage differences across the genome?
* Do you see big differences between replicates?

# Post-mapping analyses

* **What is the structure of a SAM file, and how does it relate to a BAM file?**

SAM files have a header (started by @) and an alignment section (with the linear alignment of the read and a quality score in Phred scale). The alignment section has eleven mandatory fields. BAM are compressed binary versions of the same data as SAM files, so the only difference is that they are less problematic for storage capacity.

# Read counting

* What is the distribution of the counts per gene? Are most genes expressed? How many counts would indicate that a gene is expressed?

# Expression analyses

* **If your expression results differ from those in the published article, why could it be?**
* **How do the different samples and replicates cluster together?**

According to the PCA plot obtained, there are two differentiated clusters: one of continuous samples and another one of batch samples. There is also some inter-variance in said groups, but it is very small in comparison to the overall difference between the groups.

* **What effect and implications has the p-value selection in the expression results?**

The p-value indicates an approximate percentage of fake positives in our tests (how many are significant just by chance). When running summary(res) in R, we can see that the adjusted p-value is of 0.1, so 1% of our tests will be fake positives. Since we are analysing a big amount of genes, this is not really acceptable (more in next question).

* **What is the q-value and how does it differ from the p-value? Which one should you use to determine if the result is statistically significant?**

A q-value is a modified p-value which has been corrected for multiple testing. Multiple testing will confoundingly increase significance of our results and produce false positives, as in the case of this analysis. Therefore, it is important to use the q-value instead of the p-value, which is not corrected for the False Discovery Rate (FDR).

* **Do you need a normalization step? What would you normalize against? Does DESeq do it?**

I did not use a normalization step, as the DESeq2 vignette specifies. If I were to do so, I would have to normalize reads against the overall reads per sample (our library), to make sure that there are no samples with more reads than others (leading to biased results). DESeq2 performs this step internally, as it indicates when printing “estimating size factors”.

* **What would you do to increase the statistical power of your expression analysis?**

The best way of increasing the statistical power of the expression analysis would be to analyse more biological samples. Even if they are most difficult and costly to produce, biological replicates can produce more significant data. It is also possible to use technical replicates instead, which is usually less expensive and less powerful. However, it could be very useful for this particular analysis, since our RNA data had not very high quality to begin with.

Daily log update

# 08/05/2020

Since I have not been able to register daily logs of my progress for some time, I have decided instead to provide a short summary of what has changed so far. I apologize for the inconvenience. Current state of the project:

1. Prokka was run succesfully, thanks to the indications from the TA from last scheduled session. More genes were found than in the paper, which was to be expected due to the time gap between analyses (databases would be more updated now, in 2020, than in 2017). In order to try to obtain even more information, I also used the online version of eggNOGmapper to annotate the genome. This is currently running.

2. The FastQC output seems quite difficult to understand, since it produces many files for the same samples, making comparison with trimmed reads a not-immediate task. I plan on asking the TA about this during today's scheduled lab session.

3. I have been looking forward to perform the homology/synteny comparison analysis. As it was stated in the paper, there are some known organisms in the \_Leptospirillum\_ genus, such as \_L. ferrooxidans\_ or \_L. rubarum\_. I have been able to download the complete genome from \_L. ferrooxidans\_. However, I also wanted to perform an extra analysis with information from \_L. rubarum\_. I do not seem to find any good genomes from this organism.

4. I would like to ask the TA some questions about using Artemis or IGV for figures.

Lab session with the TA:

1. Question about FastQC was solved. He gave me some solution to try once UPPMAX is back, so I can obtain one single FastQC plot that is easier to look at and analyse.

2. Question about \_L. rubarum\_ was solved. There are no 100% complete \_L. rubarum\_ assemblies available. Therefore, I will only perform the synteny comparison with \_L. ferroxidans\_ once UPPMAX is back.

3. Questions about ACT were also solved.

To be done:

\* Run BWA batch job.

\* Download ACT to generate figures.

\* Run BLAST for homology analysis.

# 11/05/2020

UPPMAX has not been available for a few days, and continues not to be by today's lab session. I have not been able to work in the project for some time as a consequence of this issue.

# 14/05/2020

During today's lab session:

1. With instructions from the TA, I ran a new FastQC batch job to generate only one file, as established in previous log.

2. I looked into downloading ACT viewer. This proved to be a more complicated task than I expected. Bioconda installation failed. I am looking into approaches for installation within WSL.

To do:

\* Install ACT.

\* Read FastQC output.

\* Run BWA job.

# 18/05/2020

During today:

1. I finally managed to install ACT locally on my computer. This is something I would have wished to have put more energy into while UPPMAX was down, but I did not expect it to be so problematic. However, I finally succeeded.

2. Created .gitignore file for better management ad updated files in GitHub.

3. For some reason, my FastQC and BLAST jobs kept on failing. I decided to re-run FastQC for a last time, but give this task less of a priority.

To do:

\* Run BLAST succesfully (required for ACT).

\* Generate synteny plots with ACT viewer.

\* Run BWA batch job.

# 20/05/2020

During today's lab session:

1. I finally managed to run my BLAST job succesfully. I found out that I had been using the wrong prokka file to perform this task, which was making it impossible to perform correctly. I executed the program with prokka\_.fna, without any annotation information. This file was renamed and put into a new folder (04\_synteny) inside analyses, to make everything easier to find.

2. With the BLAST output, I used ACT viewer to generate synteny files. \_L. ferriphilum\_ and \_L. ferrooxidans\_ seem to be quite similar in regards to genomic content, but there has been a lot of shuffling regarding positions in the genome.

3. BWA needs to generate a database with the index command before any mapping can be executed. I had quite similar issues as those with the BLAST job, since I kept using the wrong files from Prokka to generate it. I finally obtained a good database to use.

4. FastQC seems to have failed again. Therefore, I decided to ignore this secondary task for now, and revisit it later if I have time to do so in the future.

5. I ran BWA mem, as recommended in the guide, and combined it with samtools in order to generate .bam files, instead of the very heavy .sam files. I was, however, not very content with how I wrote this code, since I used a lot of repetition and could have probably done it with a loop. Since I am not very experimented with Bash and do not have lots of time either, I decided to look into it if I was to need it in the future.

To do:

\* Write HTSeq job and run it.

\* Analyse obtained data with DESeq2 package in R.

# 22/05/2020

During today's session:

1. I read the Slurm job output for BWA. I found it slightly concerning, since it kept mentioning 'skip orientation RR as there are no enough pairs'. After some research online, I found this to be expected and not an indication of any issues with BWA.

2. I read the HTSeq documentation, in order to understand how to use it. I also looked into how to write loops in Bash, since I realised I would otherwise write almost identical lines of code again. In this case, it would be very easy to commit typos, so I found it somewhat relevant. After some trial and error, I managed to write my script succesfully. Before running HTSeq, I generated a new file from the prokka\_.gff, removing all nucleotide sequences from it.

To do:

\* Read HTSeq Slurm job output and files.

\* Analyse obtained data with DESeq2 package in R.

# 23/05/2020

During today's session:

1. I read the DESeq2 documentation onto how to analyse counts from .bam files. While explained in a lot of detail, which was tremendously useful, I also found it somewhat confusing. I needed to look into [additional documentation](https://lashlock.github.io/compbio/R\_presentation.html) in order to be able to use the DESeq2 package successfully. I managed to perform a statistical analysis and obtained a relevant PCA plot.

2. Looking at the required questions and answers from the Student Manual, I realised I was lacking on information from Trimmomatic and FastQC. I had let these steps as something secondary to look into later, since I had issues performing them and more time-consuming tasks ahead of me. Therefore, I started to read into documentation to figure out what my initial issues with FastQC were and how to use Trimmomatic properly.

To do:

\* Run Trimmomatic and FastQC.

\* Update GitHub with everything required before deadline.