Preprocesamiento FUM-AMON

Karla Zarco

2024-05-14

Fastqc

To do some quality control checks on raw sequence data coming from high throughput sequencing pipelines https://www.bioinformatics.babraham.ac.uk/projects/fastqc/.

```
# fastqc filename.fastq
```

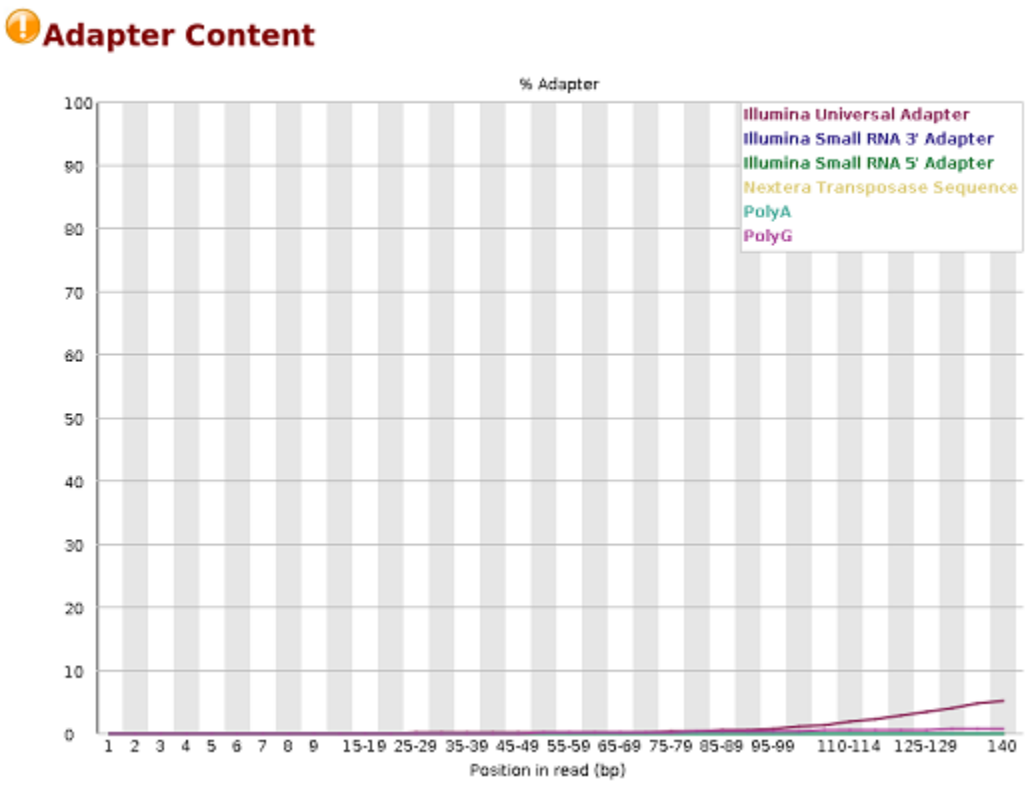

Figure 1: The fastq report shows that there are adapters

To remove adapters we will use Trimmomatic http://www.usadellab.org/cms/?page=trimmomatic

TRIMMOMATIC

Trimmomatic performs a variety of useful trimming tasks for illumina paired-end and single ended data.The selection of trimming steps and their associated parameters are supplied on the command line.

The current trimming steps are:

```
ILLUMINACLIP: Cut adapter and other illumina-specific sequences from the read.
SLIDINGWINDOW: Perform a sliding window trimming, cutting once the average
quality within the window falls below a threshold.
LEADING: Cut bases off the start of a read, if below a threshold quality
TRAILING: Cut bases off the end of a read, if below a threshold quality
CROP: Cut the read to a specified length
HEADCROP: Cut the specified number of bases from the start of the read
MINLEN: Drop the read if it is below a specified length
TOPHRED33: Convert quality scores to Phred-33
TOPHRED64: Convert quality scores to Phred-64
```

It works with FASTQ (using phred + 33 or phred + 64 quality scores, depending on the Illumina pipeline used), either uncompressed or gzipp'ed FASTQ. Use of gzip format is determined based on the .gz extension.

```
# trimmomatic PE NS.X0085.007.IDT_i7_103---IDT_i5_103.35C3_R1.fastq.gz
NS.X0085.007.IDT_i7_103---IDT_i5_103.35C3_R2.fastq.gz
/media/karla/Vol/FUM-AMON/TRIM/103.35C3_R1_paired.fastq
/media/karla/Vol/FUM-AMON/TRIM/103.35C3_R1_unpaired.fastq
/media/karla/Vol/FUM-AMON/TRIM/103.35C3_R2_paired.fastq
/media/karla/Vol/FUM-AMON/TRIM/103.35C3_R2_unpaired.fastq
ILLUMINACLIP:/media/karla/Vol/TruSeq3-PE-2.fa:2:30:10 LEADING:3
TRAILING:3 SLIDINGWINDOW:4:20 MINLEN:30
```

We perform the analysis again with FASTQC and

Command loop

We can perform preprocessing in loop, for that we generate a file that contains the names of our files using the ls and sed commands

```
# ls *_R1.fastq.gz| sed 's/_R1.fastq.gz//' > listanombres
```

We can check the content of 'listanombres' file

```
# cat listanombres
```

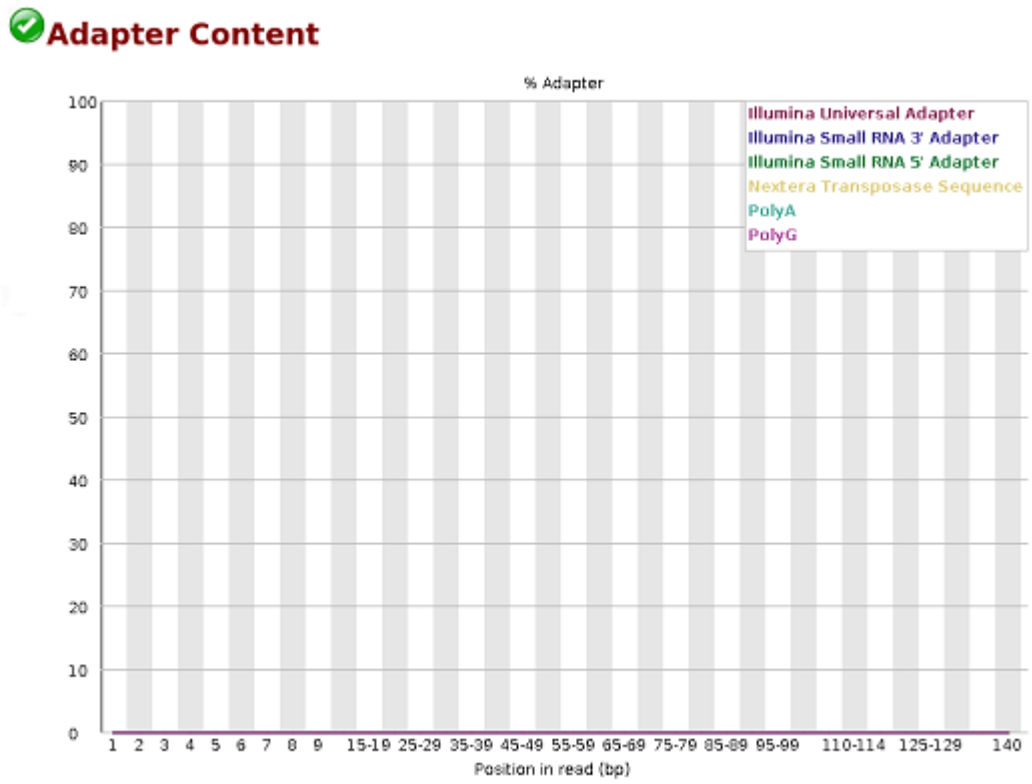

Figure 2: Fastqc report showed that adapters were removed

```
(base) karla@karla-System-Product-Name:/media/karla/Expansion/Doctorado/FUM-AMONIO/raw_s
equences$ cat listanombres
NS.X0085.007.IDT_i7_103---IDT_i5_103.35C3
NS.X0085.007.IDT_i7_104---IDT_i5_104.11N14
NS.X0085.007.IDT_i7_105---IDT_i5_105.25C0
NS.X0085.007.IDT_i7_106---IDT_i5_106.35F3
NS.X0085.007.IDT_i7_107---IDT_i5_107.21N3
NS.X0085.007.IDT_i7_115---IDT_i5_115.11C14
NS.X0085.007.IDT_i7_116---IDT_i5_116.15C0
NS.X0085.007.IDT_i7_117---IDT_i5_117.25F3
NS.X0085.007.IDT_i7_118---IDT_i5_118.35C14
NS.X0085.007.IDT_i7_119---IDT_i5_119.21F14
NS.X0085.007.IDT_i7_127---IDT_i5_127.15C14
NS.X0085.007.IDT_i7_128---IDT_i5_128.15F3
NS.X0085.007.IDT_i7_129---IDT_i5_129.25C14
NS.X0085.007.IDT_i7_130---IDT_i5_130.35N14
NS.X0085.007.IDT_i7_131---IDT_i5_131.25F14
NS.X0085.007.IDT_i7_139---IDT_i5_139.15N3
NS.X0085.007.IDT_i7_140---IDT_i5_140.15N14
NS.X0085.007.IDT_i7_141---IDT_i5_141.31C0
NS.X0085.007.IDT_i7_142---IDT_i5_142.11C3
NS.X0085.007.IDT_i7_143---IDT_i5_143.31N3
```

```
# for n in $(cat listanombres); do
trimmomatic PE $n\_R1.fastq.gz $n\_R2.fastq.gz
/media/karla/Vol/FUM-AMON/TRIM/$n\_R1_paired.fastq
/media/karla/Vol/FUM-AMON/TRIM/$n\_R1_unpaired.fastq
/media/karla/Vol/FUM-AMON/TRIM/$n\_R2_paired.fastq
/media/karla/Vol/FUM-AMON/TRIM/$n\_R2_unpaired.fastq
ILLUMINACLIP:/media/karla/Vol/TruSeq3-PE-2.fa:2:30:10 LEADING:3
TRAILING:3 SLIDINGWINDOW:4:20 MINLEN:30; done
```

Remove human genome contaminants from metagenomic datasets

Bowtie2 - qiime2

Create a bowtie2 database

A bowtie2 index is generated externally using the bowtie2-build command (https://github.com/BenLangmead/bowtie2). This will generate six files (where basename is a name defined by the user): basename.1.bt2 basename.2.bt2 basename.3.bt2 basename.4.bt2 basename.rev.1.bt2 basename.rev.2.bt2

```
# bowtie2-build GCF_000001405.40_GRCh38.p14_genomic.fna.gz GRCh38.p14.fa
```

Put these files into a single directory that does not contain any other files; then import that directory using QIIME 2. For example, if these files are in the bowtie-db directory, we would use the following command to import our database into a single artifact:

Importing the index

```
# conda activate qiime2-2023.9
```

```
# qiime tools import \
--input-path /media/karla/Vol/DATABASES/bowtie-db-hg38/ \
--output-path bt2-database.hg38.qza \
--type Bowtie2Index
```

Import the sequences

```
# qiime tools import \
--type 'SampleData[PairedEndSequencesWithQuality]' \
--input-path Manifest.txt \
--output-path paired-end-demux.qza \
--input-format PairedEndFastqManifestPhred33V2
```

https://docs.qiime2.org/2023.9/plugins/available/quality-control/filter-reads/

Usage: qiime quality-control filter-reads [OPTIONS] Filter out (or keep) demultiplexed single- or paired-end sequences that align to a reference database, using bowtie2 and samtools. This method can be used to filter out human DNA sequences and other contaminants in any FASTQ sequence data (e.g., shotgun genome or amplicon sequence data), or alternatively (when exclude_seqs is False) to only keep sequences that do align to the reference.

Filter

```
# qiime quality-control filter-reads \
--i-demultiplexed-sequences paired-end-demux.qza \
--i-database /media/karla/Expansion/Doctorado/Reference/bt2-database.hg38.qza \
--p-n-threads 10 \
--p-mode local \
--p-sensitivity sensitive \
--o-filtered-sequences demux_filtered_human.qza
```