SOP for sequencing data QC

1. Load your dataset (normally raw\_sequences.fastq or raw\_sequences.fastq.gz)

Need to create raw\_sequence\_folder, QC\_raw\_folder, trimmed\_sequences\_folder, QC\_trimmed\_folder

1. FastQC and MultiQC

fastqc \*.fastq.gz -o QC\_raw/ # fast QC multiple samples in the current folder and save the results in QC\_raw folder

multiqc QC\_raw/ -o QC\_summary/

fastqc sample\_R1.fastq.gz sample\_R2.fastq.gz # fast QC one sample and show the result

Key metrics to assess:

| **Metric** | **Threshold** | **Notes** |
| --- | --- | --- |
| Per-base sequence quality | > Q30 | Consistently high across read length |
| GC content | ±5–10% expected | Check for contamination |
| Adapter content | < 5% | High adapter presence → trimming required |
| Sequence duplication | < 50% | High duplication may indicate PCR bias |
| Overrepresented sequences | < 1% | Investigate contamination or adaptors |

1. Adapter and quality trimming using cutadapt
2. QC and multiQC after trimming

fastqc trimmed/\*.fastq.gz -o QC\_trimmed/

multiqc QC\_trimmed/ -o QC\_summary\_trimmed/

1. Check the sequence for one sample

# This will show the first 5 lines of this sample. Each FASTQ record is 4 lines (header, sequence, plus, quality)

head -n 5 filename.fastq # fastq file format

zcat sample\_R1.fastq.gz | head -n 5 # for compressed file, .fastq.gz

# To see **first 2 sequences**, show 8 lines:

zcat sample\_R1.fastq.gz | head -n 8

1. To check file structure quickly

zcat sample\_R1.fastq.gz | less

#use Space to scroll

#Type q to quit

1. Check the file structure without open the file, check how many lines of the file

zcat sample\_R1.fastq.gz | wc -l

1. Count total reads in a file

zcat sample\_R1.fastq.gz | awk 'END {print NR/4}' #or

zcat sample\_R1.fastq.gz | echo $((`wc -l`/4))

## Run on both R1 and R2 to confirm **equal read counts** (important after trimming).

1. Check the size and compression

ls -lh sample\_R\*.fastq.gz # Roughly similar sizes between R1 and R2 means pairs are likely intact.

1. Read length distribution (average length)

zcat sample\_R1.fastq.gz | awk 'NR%4==2 {print length($0)}' | datamash mean 1 min 1 max 1

# Requires datamash, can use conda or pip to install

1. Verify read pairing after trimming

grep "Pairs written" cutadapt.log

# or count reads:

zcat trimmed\_R1.fastq.gz | awk 'END {print NR/4}'

zcat trimmed\_R2.fastq.gz | awk 'END {print NR/4}'

# Counts should match — if not, one mate was discarded (likely due to too short or orphan reads).

| **Tool** | **Purpose** |
| --- | --- |
| **fastp** | Fast all-in-one QC + trimming + report (fastp -i R1 -I R2 -o out1 -O out2 -h fastp.html) |
| **bbduk.sh** (BBTools) | Detect and trim adapters (bbduk.sh in1=R1 in2=R2 out1=... ref=adapters.fa ktrim=r k=23 mink=11) |
| **seqtk seq -A** | Convert FASTQ → FASTA to inspect raw reads quickly |

Cutadapt example:

This script uses the SLURM job scheduler to automate processing of multiple samples:

#!/bin/bash

#SBATCH --time=01:00:00

#SBATCH --nodes=1

#SBATCH --ntasks=1

#SBATCH --cpus-per-task=12

#SBATCH --mem=32GB

#SBATCH --job-name=cutadapter

#SBATCH --account=st-jeffrich-1

#SBATCH -o /scratch/st-jeffrich-1/cutadpat/cutadapt.out

#SBATCH -e /scratch/st-jeffrich-1/cutadpat/cutadapt.err

# Load the conda environment

module load miniconda3

source activate /arc/project/st-jeffrich-1/arc/project/st-jeffrich-1/miniconda3/envs/ven\_cutadapt

# Change to working directory

cd $SLURM\_SUBMIT\_DIR

# Cutadapt variables Use NEB Next Adaptors

adaptor1="AGATCGGAAGAGCACACGTCTGAACTCCAGTCA" # remove adapter from read1

adaptor2="AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT" # remove adapter from read2

qcutoff=20

#Determine if the read file includes the R1 character.

#If it does, extract the sample name and R2 file path.

if [[ "$R1" == \*"R1"\* ]]; then

sample=$(basename $R1 \_R1.fastq.gz)

R2=$(echo $R1 | sed "s/\_R1/\_R2/")

else

echo "ERROR: File path does not appear to contain 'R1'" > /scratch/st-jeffrich-1/${initials}/logs/cutadapt/${sample}.log

exit 1

fi

# Running cutadapt with arguments

cutadapt --action trim \

--quality-cutoff ${qcutoff} \

-a ${adaptor1} \

-A ${adaptor2} \

--cores 12 \

-m 20\

-o /scratch/st-jeffrich-1/${initials}/clean/${sample}\_R1.fastq.gz \

-p /scratch/st-jeffrich-1/${initials}/clean/${sample}\_R2.fastq.gz \

${R1} \

${R2} > /scratch/st-jeffrich-1/${initials}/logs/cutadapt/${sample}.log

When you submit the job:

for R1 in /arc/project/st-jeffrich-1/RNA\_seq\_sculpin/cluster/RNA\_PS\_raw\_reads/\*\_R1.fastq.gz; do

sbatch --export=R1=${R1},initials=${projectName} cutadapt.sh;

done