LIÈGE université
GIGA

Data processing in genomics, hands-on

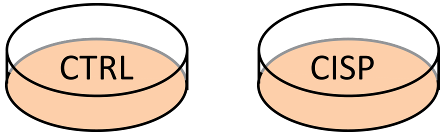
GIGA doctoral school 2021

Alice Mayer, PhD
GIGA bioinformatic team
bioinfo.giga@uliege.be

RNAseq experiment

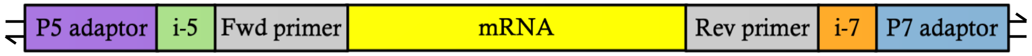
- Aim: compare gene expression between 2 conditions
- What they did in this experiment ?

1. grow cell line in presence or absence of cisplatin

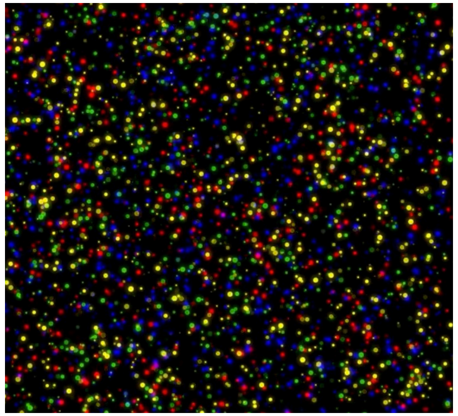


2. harvest cells and extract messenger RNAs

3. prepare libraries of cDNA from these mRNAs



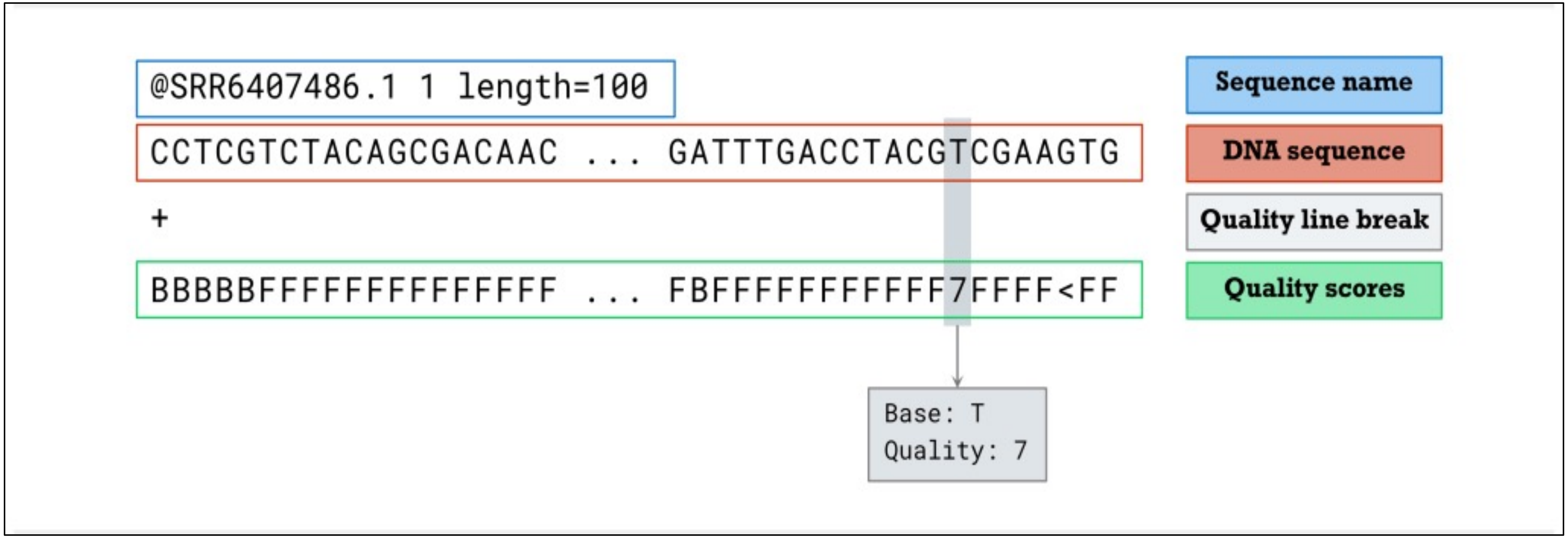
4. sequence with Illumina sequencer
   https://www.youtube.com/watch?v=fCd6B5HRaZ8



5. démultiplex sequencing run

Sequencing "raw" data

- fastq = text file, 4 lines per read



```
@SRR6407486.1 1 length=100
CCTCGTCTACAGCGACAAC ... GATTTGACCTACGTCGAAGTG
+
BBBBBFFFFFFFFFFFFFF ... FBFFFFFFFFFFF7FFFF<FF
```

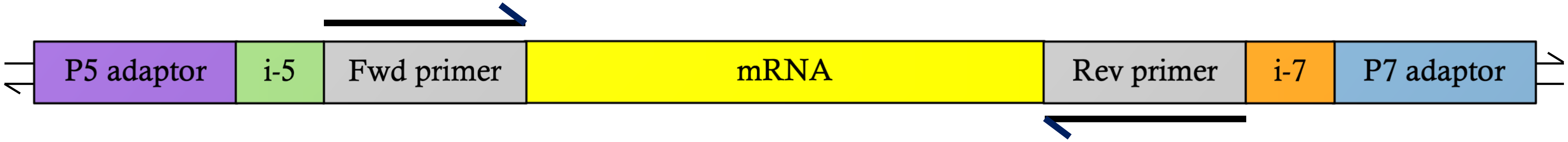
from https://gencoded.com/index.php/2020/05/20/fastq-format-an-overview/

Sequencing "raw" data

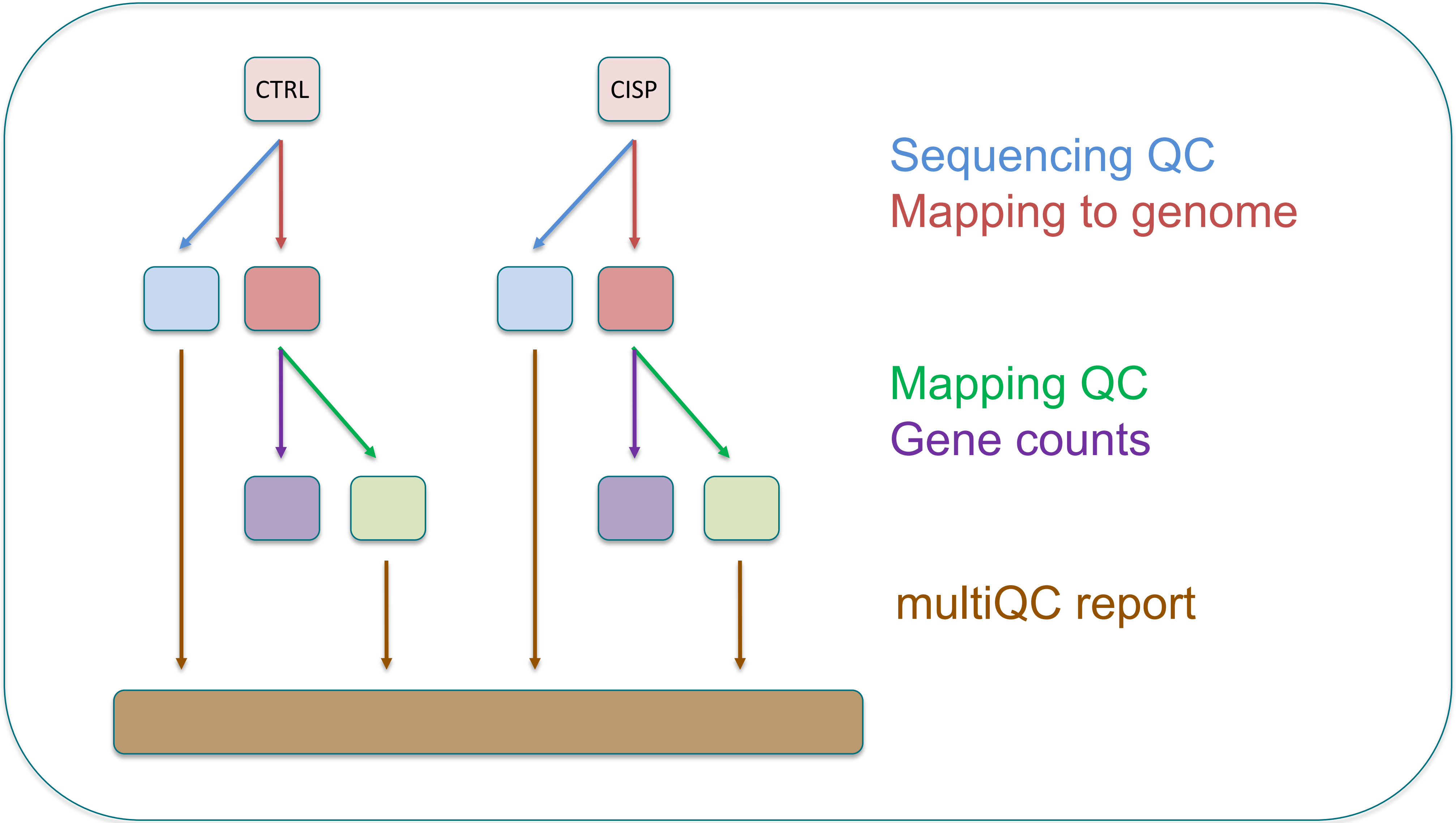
- downsampled to 3M reads (130Mb) by selecting all reads mapping to chromosome 2 + 120,000 random reads. NB: typical RNAseq exp = 25-50M reads
- $HOME/_SHARE_/Platforms/GEN/BIOINFO/TRAINING/RNAseqAnalysis/Data

```
ssh u123456@cluster.calc.priv

DataDir=$HOME/_SHARE_/Platforms/GEN/BIOINFO/TRAINING/RNAseqAnalysis/Data

cd ${DataDir}

ls -lh

zcat CTRL_R1.fastq.gz | head
```

- 2 files per sample (read 1 and read2 in opposite direction),



Analysis steps



Sequencing QC
Mapping to genome

Mapping QC
Gene counts

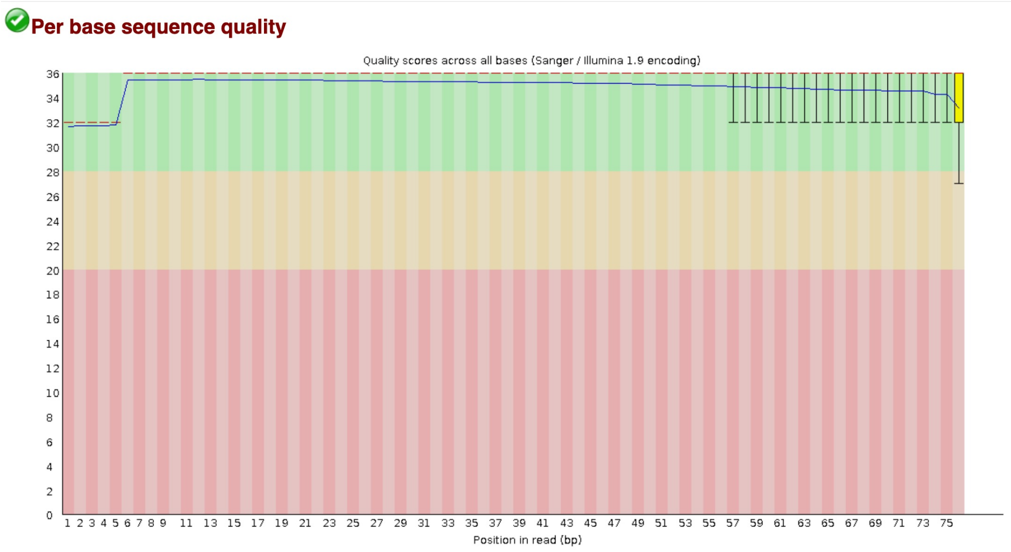
multiQC report

Sequencing QC with **fastQC**

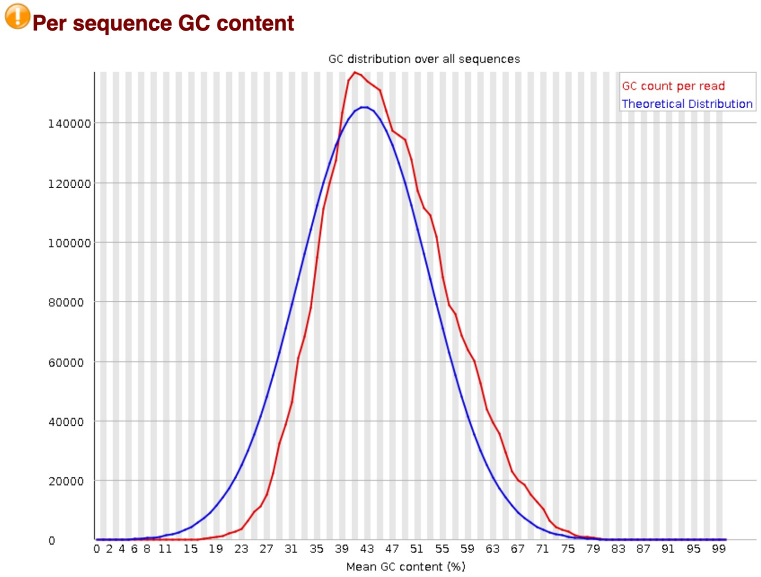
- https://www.bioinformatics.babraham.ac.uk/projects/fastqc/
- Will generate QC stats and plots for each fastq file
- Command to use for each fastq file:

```
fastqc --noextract --nogroup --outdir=${FastqcDir} ${FastqFile}
```

- Resources needed: 1 CPU and 1G of memory



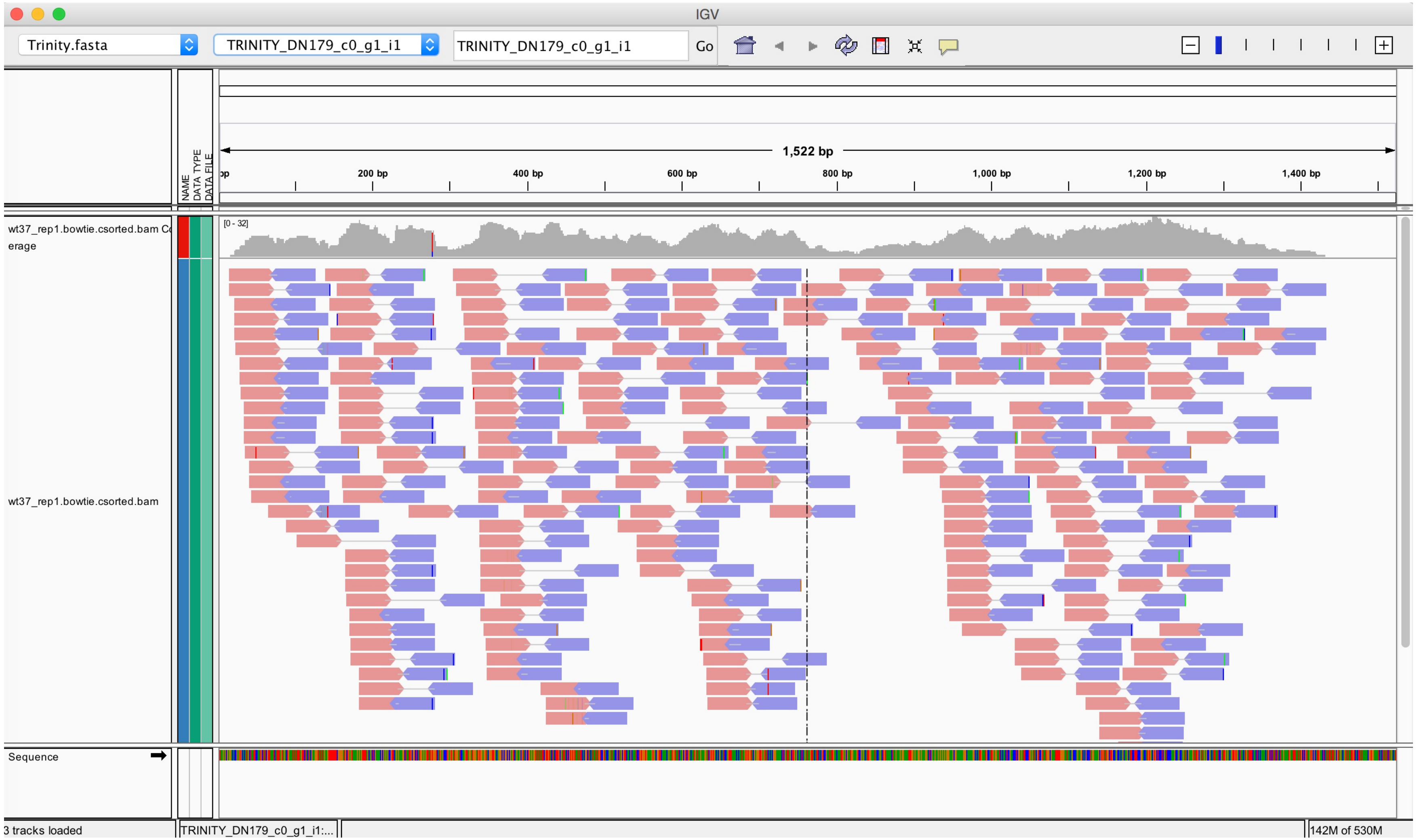
Per base sequence quality



Per sequence GC content

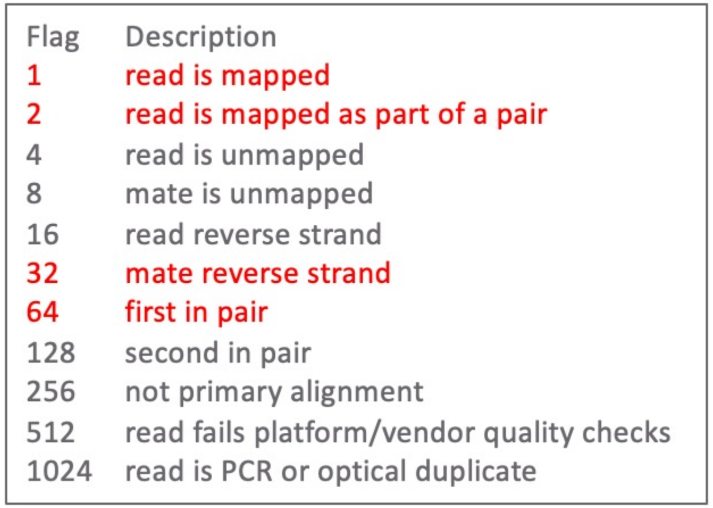
Mapping reads to genome

- Input = fastq files by pairs + genome indexes



Mapping output = alignment file (SAM/BAM)

- SAM/BAM files
  - FLAG - Information
  - RNAME - Chromosome
  - POS – Location of 1st base
  - MAPQ – Quality score
  - CIGAR - Operations



Flag	Description
1	read is mapped
2	read is mapped as part of a pair
4	read is unmapped
8	mate is unmapped
16	read reverse strand
32	mate reverse strand
64	first in pair
128	second in pair
256	not primary alignment
512	read fails platform/vendor quality checks
1024	read is PCR or optical duplicate

Paired-End

```
A00801:76:HGJCYDSXY:4:1544:20401:36699   99   1   3112677   255   150M   =   3112770   244
CTAGGAGATAGTAGGGATTGGGAAGCAACTACTGAAAGGTCTGTGTCTTCTTTGTGGATGATAAAATATTCTGGAATTATATTGTATGCTAGGCGCACAATCTTGTGACCATAGTACAGATATTCAACAGATAAATTTTGTGTGCTATGA
F:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFF:FFFFF
NH:i:1          HI:i:1          AS:i:299          nM:i:0          RG:Z:SV2-CTRL2_NGS20-O393_AHGJCYDSXY_S241_L004_R1_001
```

Mapping and count with STAR

- Most recent module on cluster = version 2.5.2b
- https://raw.githubusercontent.com/alexdobin/STAR/2.5.2b/doc/STARmanual.pdf
- Input = fastq files by pairs + genome indexes
- Output = alignment file (BAM)
- Resources: 4 CPUs and 12G RAM in total (very small dataset)
- Command options:

```
STAR    --runThreadN ${NbCPUs} \
        --genomeDir ${StarIndexDir} \
        --readFilesIn ${DataDir}/${Read1} ${DataDir}/${Read2} \
        --readFilesCommand zcat \
        --outFileNamePrefix ${AlnDir}/${Prefix}_ \
        --outSAMtype BAM SortedByCoordinate \
        --quantMode GeneCounts \
        --outTmpDir ${ScratchDir}
```

BAM indexing with samtools

- https://www.htslib.org/doc/samtools-index.html
- Index BAM file for fast random access
- Input = BAM file
- Output = index file (.bai)
- Command: samtools index ${BamFile}
- Resources: 1 CPUs et 1 Gb RAM

Mapping QCs with Picard

- https://broadinstitute.github.io/picard/command-line-overview.html
- 2 commands to collect various alignment statistics

```
java -jar $Picard CollectAlignmentSummaryMetrics \
        R=${FastaRefGen} \
        I=${BamFile} \
        O=${Prefix}_Ali_Metrics.out
```

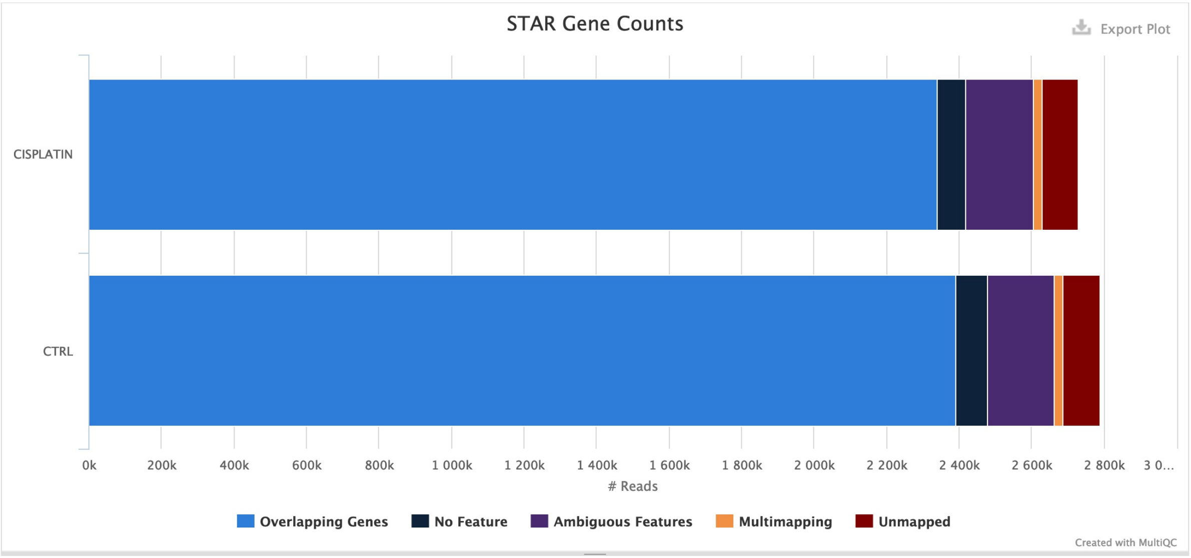
```
java -jar $Picard CollectRnaSeqMetrics \
        I=${BamFile} \
        O=${Prefix}_RNA_Metrics.out \
        REF_FLAT=${RefFlat} \
        STRAND=${Strandness}
```

Strandness = "NONE", "FIRST_READ_TRANSCRIPTION_STRAND" or" SECOND_READ_TRANSCRIPTION_STRAND", but not sure it's using it for paired end reads (?)

Making QC report with multiQC

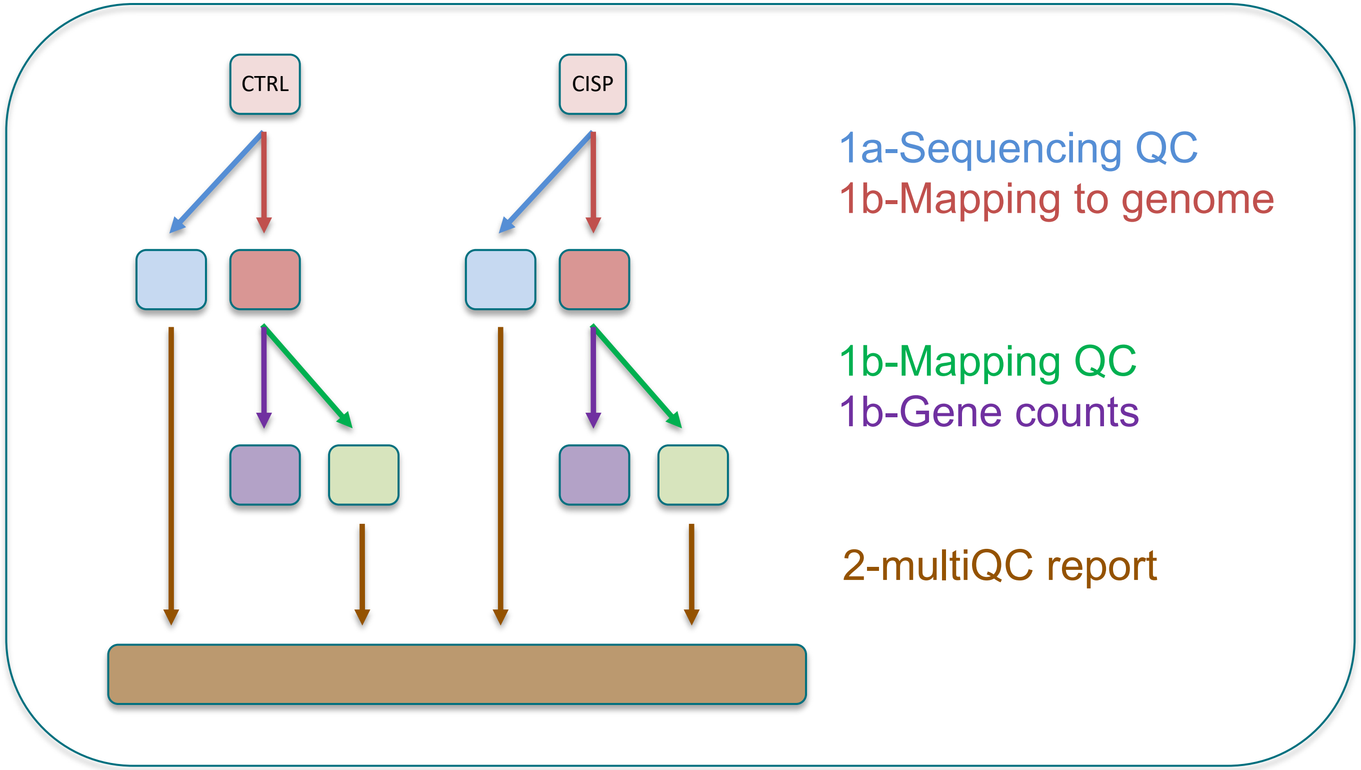
- https://multiqc.info/
- Input = stats from fastQC, Picard and STAR
- Output = html report
- Resources: 1 CPU and 1G RAM

```
cd $MultiqcDir
multiqc --force -n FastQC ${FastqcDir}
multiqc --force -n MappingQCs ${AlnDir}
```



Analysis Tools

- Sequencing QC
  - fastQC (input = fastq)
- Mapping and counts
  - STAR (inputs = fastq + genome index)
- Mapping QC
  - index BAM with samtools
  - Picard Alignment stats
  - Picard RNA metrics
- Merge QCs into html report



TO DO

- Decide how to organise outputs/scripts/logs and create folders

```
(ssh u123456@cluster.calc.priv)

cd =$HOME/_SHARE_/Platforms/GEN/BIOINFO/TRAINING/RNAseqAnalysis

ls -lh

mkdir Logs; mkdir Scripts; mkdir QC; mkdir Aln
```

- Create scripts files
  - 1a-fastQC.sh
  - 1b-mappingandcount.sh
  - 2-multiQC.sh

TO DO: write script

1. First line = #!/usr/bin/env bash
2. copy the main command(s) to the script
3. Define needed variables, examples :

```
TrainingDir=$HOME/_SHARE_/Platforms/GEN/BIOINFO/TRAINING

AnalysisDir=${TrainingDir}/RNAseqAnalysis

DataDir=${AnalysisDir}/Data

GenomeDir=${TrainingDir}/Genome/Homo_sapiens_chrom2/Ensembl/GRCh38/release_104
```

NB: check variables with command ls, ex:
ls -lh $GenomeDir

TO DO: more complex variables

```
cd ${DataDir}

Read1_list=( *_R1.fastq.gz )

i=$((SLURM_ARRAY_TASK_ID-1))

Read1=${Read1_list[${i}]}

Read2=${Read1/R1/R2}
```

NB: check variables with command echo, ex:
echo $ Read1

TO DO: write script

4. find modules for each tool
5. optional: test the variables and commands in srun session
6. write slurm headers according to resources needed
7. add few statements to monitor what's happening

```
echo "************** This job is the ${SLURM_ARRAY_TASK_ID} th ******************"
echo "###### run star on fastq ${Read1} and ${Read2}"
```

8. launch with sbatch
9. troubleshoot if needed
10. check outputs and reports
    - fastQC and mapping QC html reports
    - alignment (BAM): visualise with IGV http://software.broadinstitute.org/software/igv/AlignmentData
    - count tables
11. document code

nf-core pipeline

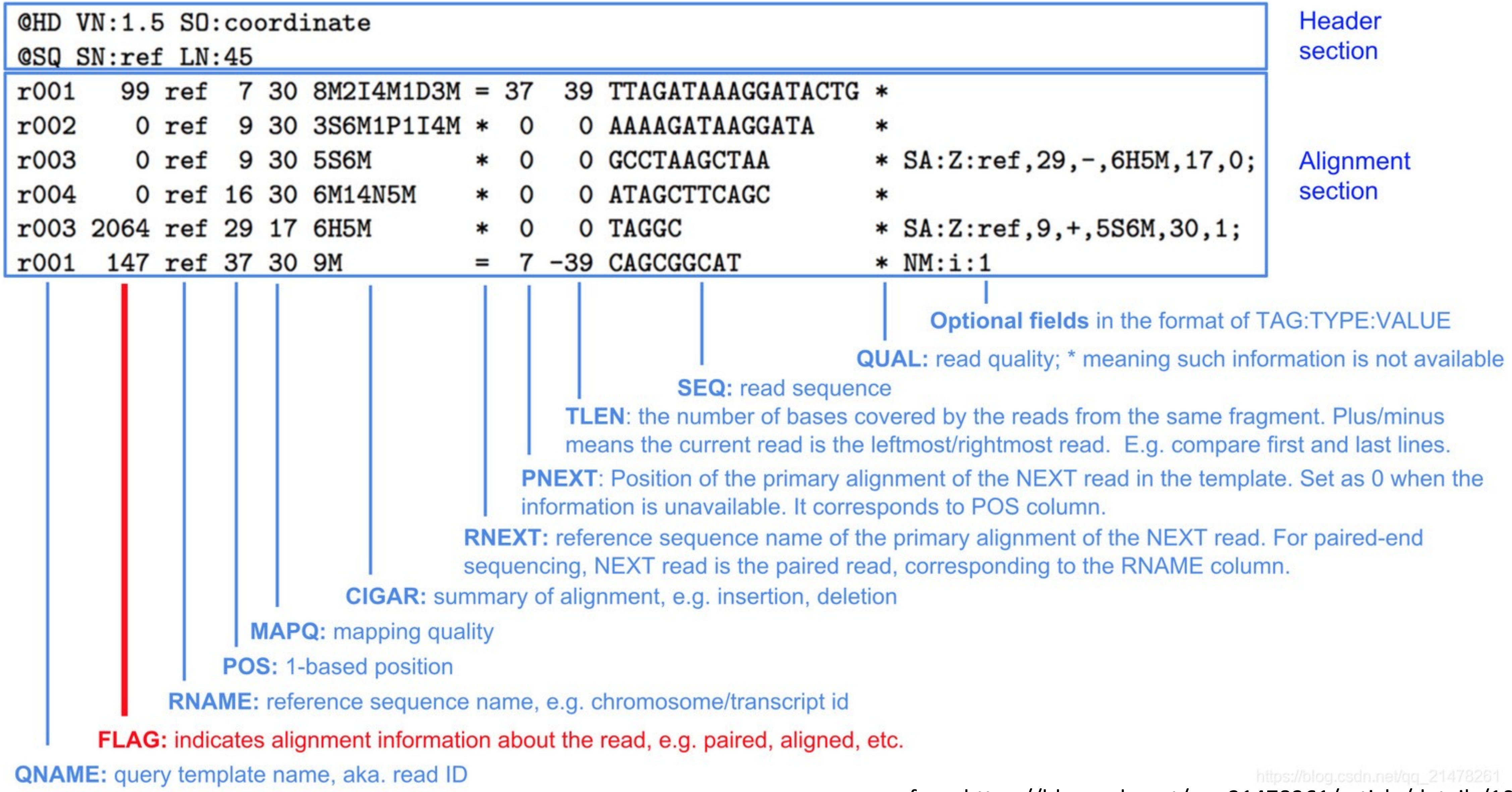
- https://nf-co.re/rnaseq
- Pipeline allowing to process several samples in parallel
- One command to launch the whole pipeline (=> write master script that will takes care of "slave" jobs)
- Lots of QCs
- Very active community

LIÈGE université
GIGA

Thank you for your attention !
Questions ?

Alice Mayer, PhD
GIGA bioinformatic team
bioinfo.giga@uliege.be

Mapping output = alignment file (SAM/BAM)



```
@HD VN:1.5 SO:coordinate
@SQ SN:ref LN:45
r001   99 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002    0 ref  9 30 3S6M1P1I4M * 0    0 AAAAGATAAGGATA    *
r003    0 ref  9 30 5S6M       * 0    0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    * 0    0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       * 0    0 TAGGC             * SA:Z:ref,9,+,5S6M,30,1;
r001  147 ref 37 30 9M         = 7  -39 CAGCGGCAT         * NM:i:1
```

Header section

Alignment section

Optional fields in the format of TAG:TYPE:VALUE

QUAL: read quality; * meaning such information is not available

SEQ: read sequence

TLEN: the number of bases covered by the reads from the same fragment. Plus/minus means the current read is the leftmost/rightmost read. E.g. compare first and last lines.

PNEXT: Position of the primary alignment of the NEXT read in the template. Set as 0 when the information is unavailable. It corresponds to POS column.

RNEXT: reference sequence name of the primary alignment of the NEXT read. For paired-end sequencing, NEXT read is the paired read, corresponding to the RNAME column.

CIGAR: summary of alignment, e.g. insertion, deletion

MAPQ: mapping quality

POS: 1-based position

RNAME: reference sequence name, e.g. chromosome/transcript id

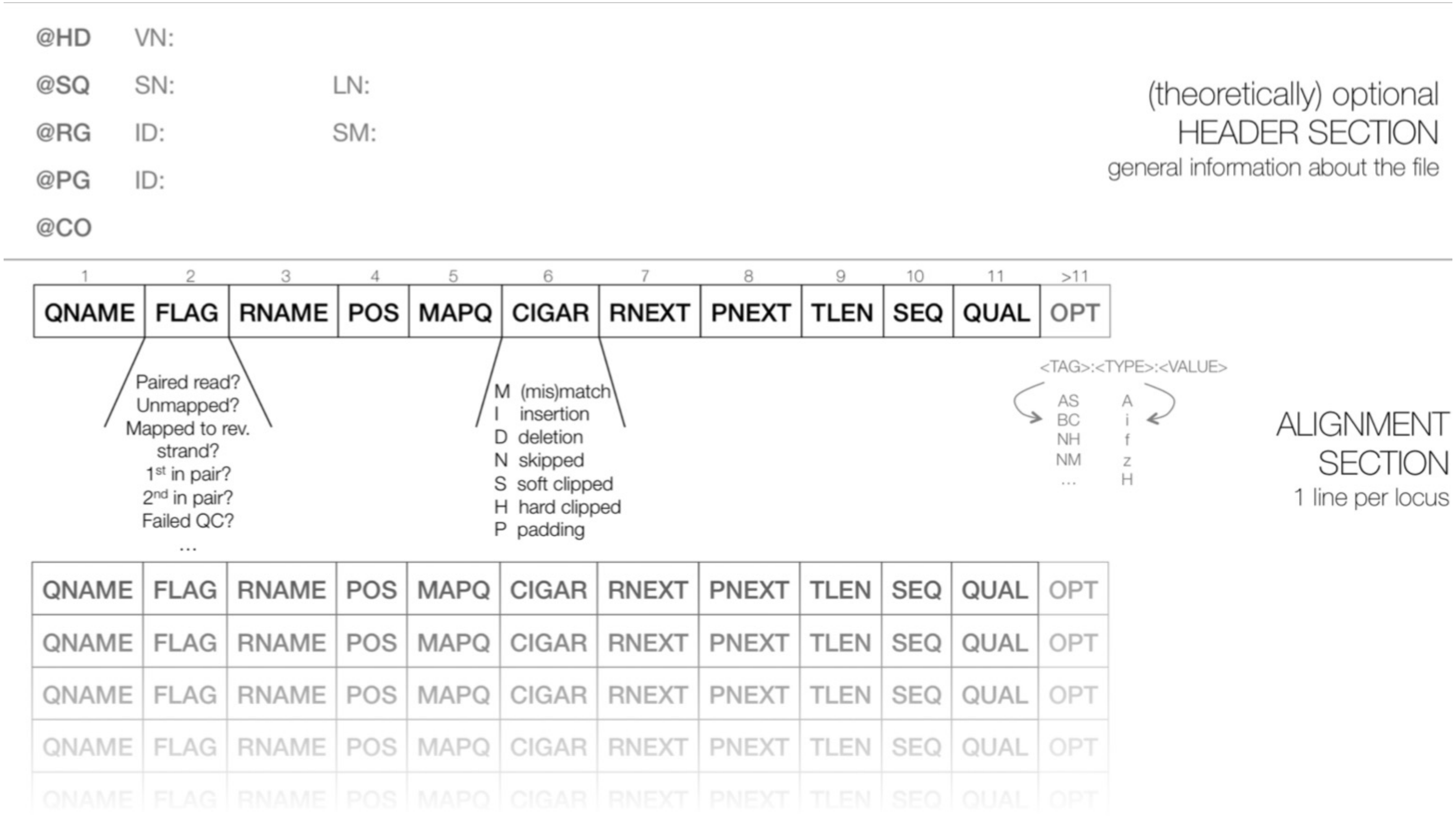
FLAG: indicates alignment information about the read, e.g. paired, aligned, etc.

QNAME: query template name, aka. read ID

https://blog.csdn.net/qq_21478261

from https://blog.csdn.net/qq_21478261/article/details/106005482

Mapping output = alignment file (SAM/BAM)



@HD	VN:	
@SQ	SN:	LN:
@RG	ID:	SM:
@PG	ID:	
@CO		

(theoretically) optional
HEADER SECTION
general information about the file

1	2	3	4	5	6	7	8	9	10	11	>11
QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPT

Paired read?
Unmapped?
Mapped to rev. strand?
1st in pair?
2nd in pair?
Failed QC?
...

M (mis)match
I insertion
D deletion
N skipped
S soft clipped
H hard clipped
P padding

<TAG>:<TYPE>:<VALUE>

AS	A
BC	i
NH	f
NM	z
...	H

QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPT
QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPT
QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPT
QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPT
QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPT

ALIGNMENT SECTION
1 line per locus

from https://www.biostars.org/p/376099/