Omics in human diseases

Index

- Omics data and Biological databases
- NGS methods
- **NGS data analysis**
- Prediction and interpretation of pathogenic variants
- Protein-protein interaction networks

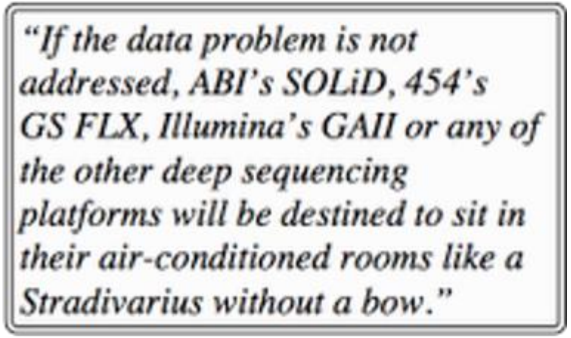
Course organization 2022/2023

Monday: Frontal lecture
Thursday: Frontal lecture/ guided practical activity

How to pass the exam: multiple choice quiz (50%) + results from practical activities (50%) + Bonus points, e.g. summary of previous lecture (up to 10%)

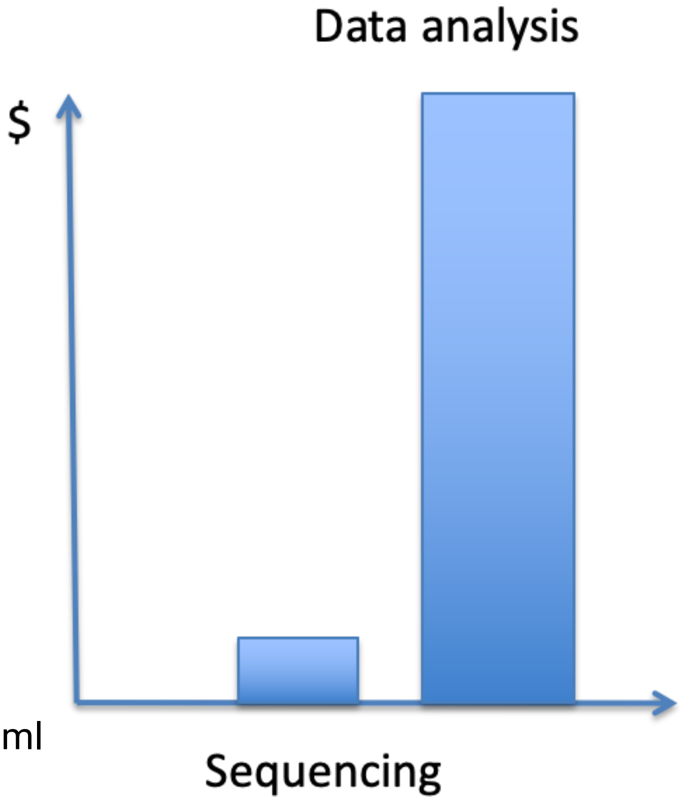
Mail: **emanuela.leonardi@unipd.it**

Main hazard - DATA ANALYSIS

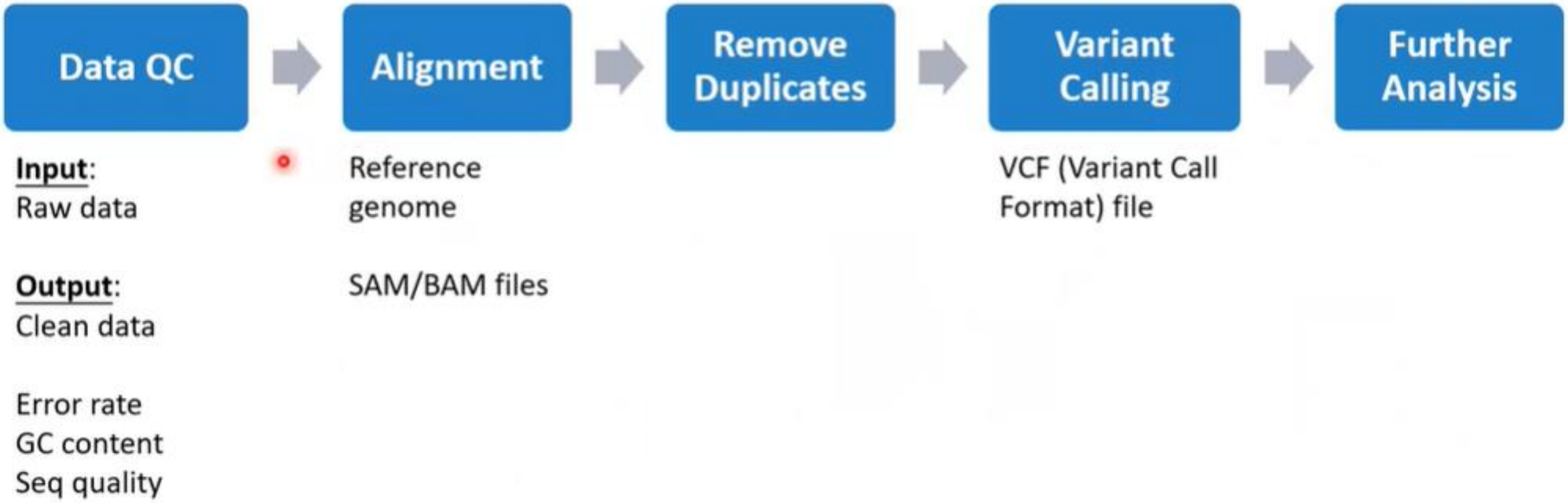


"If the data problem is not addressed, ABI's SOLiD, 454's GS FLX, Illumina's GAII or any of the other deep sequencing platforms will be destined to sit in their air-conditioned rooms like a Stradivarius without a bow."

http://finchtalk.blogspot.com/2010/09/geospiza-in-news.html



NGS analysis workflow



Data QC → Alignment → Remove Duplicates → Variant Calling → Further Analysis

Input:
Raw data

Output:
Clean data

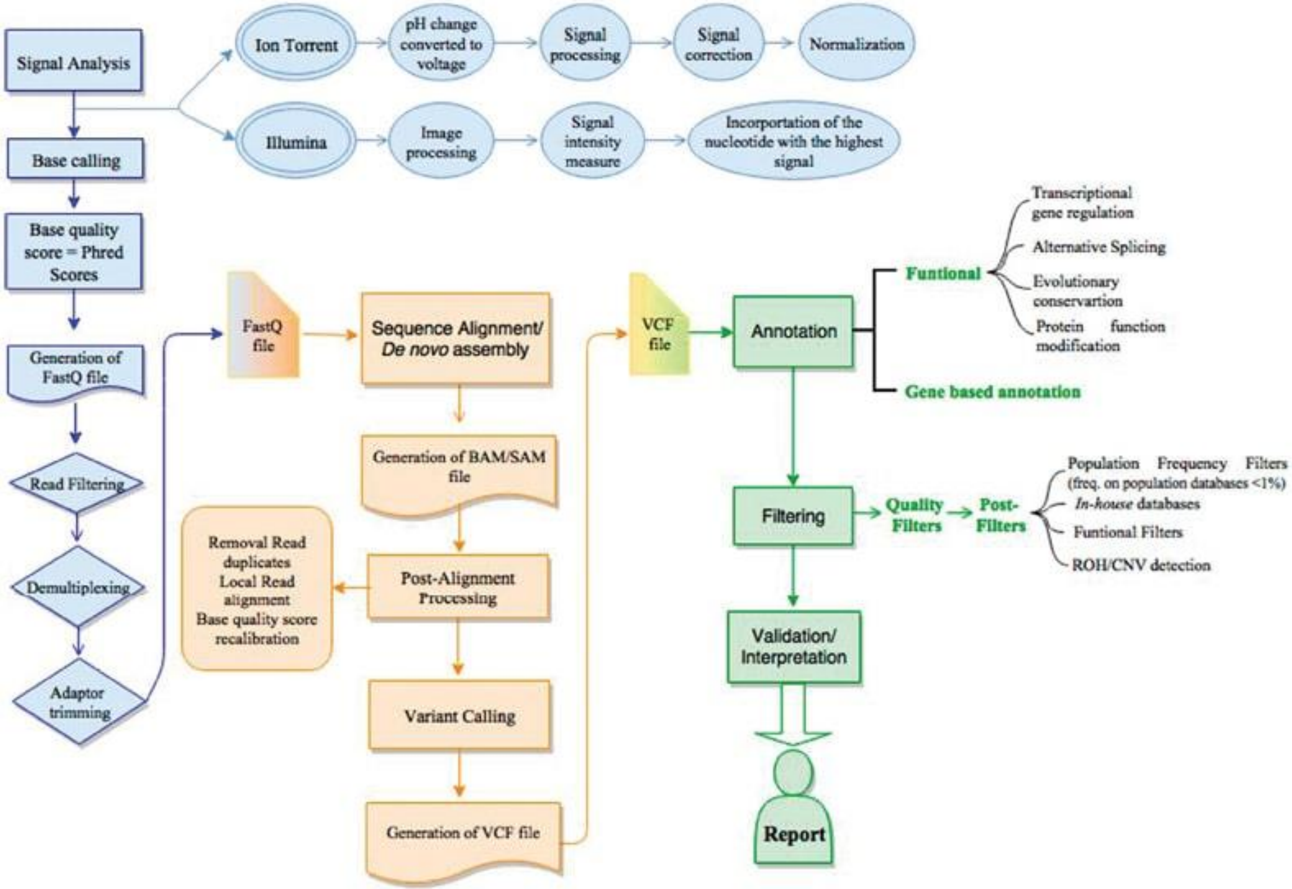
Error rate
GC content
Seq quality

Reference
genome

SAM/BAM files

VCF (Variant Call
Format) file

NGS analysis workflow



NGS analysis: output files

FILE TYPE	DESCRIPTION	WHERE IT IS USED
FASTQ	Text-based file format containing raw sequence reads and the associated quality score of each base	Storage of raw sequence data and input into sequence alignment
BED	Browser Extensible Data file is a tab-delimited text file that is used to store genomic regions as coordinates	In variant calling pipelines to direct the analysis to a genomic region
SAM	Sequence Alignment Map file, used to store text-based information for reads aligned to a reference sequence	Store information on read alignment, e.g. position and quality
BAM	Binary Alignment Map file is a compressed binary version of a SAM file. Can be opened in genome browsers to view read alignment	Used for input into variant calling pipelines
VCF	The Variant Call Format is a text file which stores **sequence variants**, each variant occupies a single row	Generated by variant calling pipelines. Used as input into variant annotation

NGS Data Analysis

1. Raw data Output
2. Sequence Alignment
3. Variant Calling
4. Additional Software and Tools

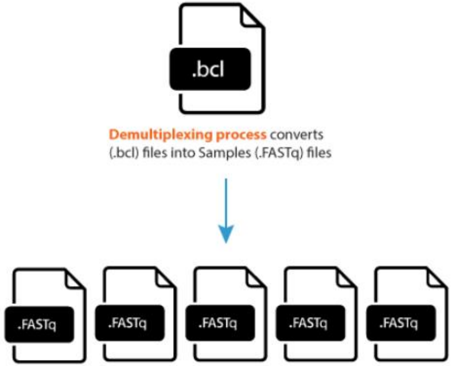
Raw data output

- .bcl format contains
  - + Base calls per cycle
  - + Quality of each call

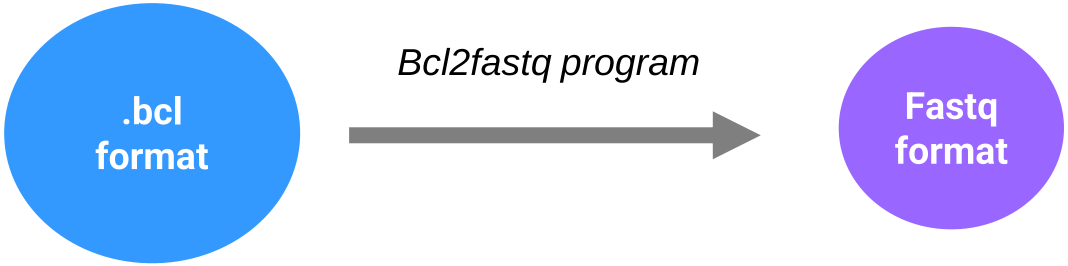
  Each base is recorded as the machine makes the call

- Demultiplexing
  - *When more samples were ran on the same sequencer, then the .bcl raw data are sorted to separate reads*

- Convert .bcl data into universally used fastq files by bcl2fastq



Demultiplexing process converts (.bcl) files into Samples (.FASTq) files



.bcl format

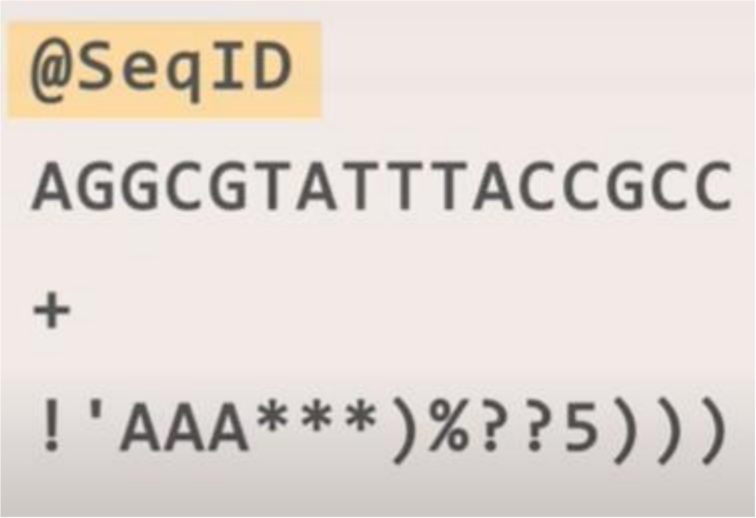
Bcl2fastq program

Fastq format

FastQ FORMAT

- Universal sequencing data file
- Consist of four lines in each reads

1. Sequence identifier (began with a @)
2. Sequence of the read
3. Spacer
4. Phred quality scores

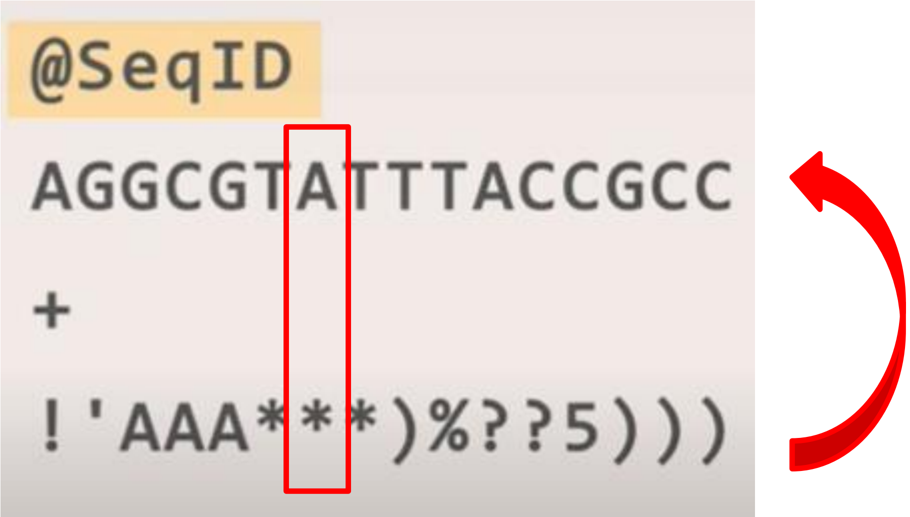


```
@SeqID
AGGCGTATTTACCGCC
+
!'AAA***)%??5)))
```

FastQ FORMAT https://maq.sourceforge.net/fastq.shtml

- Universal sequencing data file
- Consist of four lines in each reads

1. Sequence identifier
2. Sequence of the read
3. Spacer
4. Phred quality scores

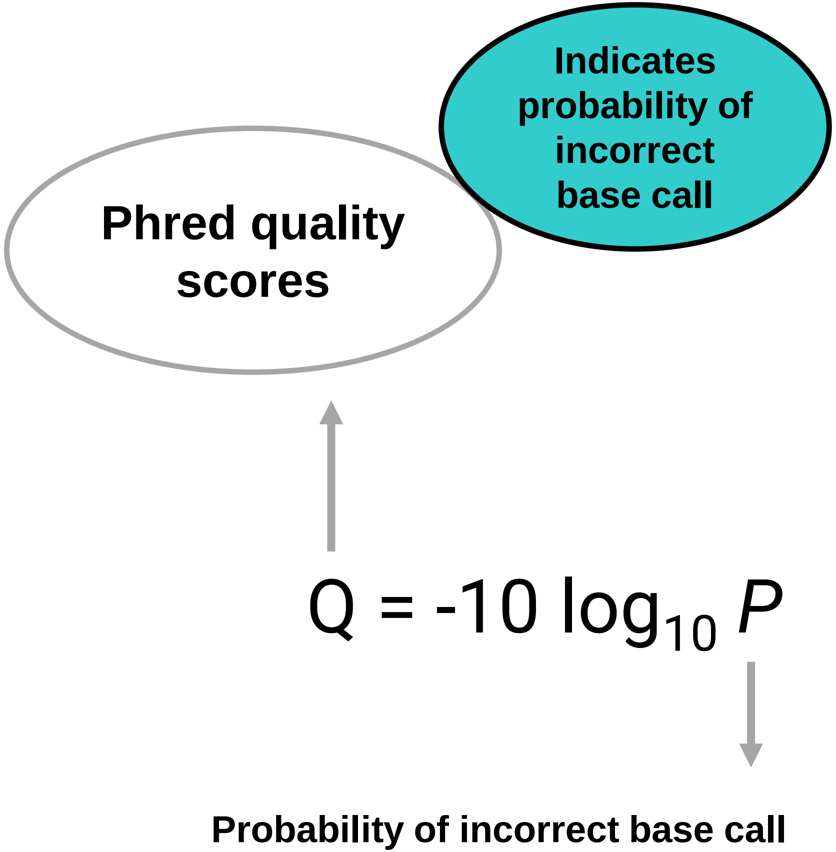


```
@SeqID
AGGCGTATTTACCGCC
+
!'AAA***)%??5)))
```

Each base and its corresponding quality score are coded using a single ASCII character.
Quality scores ranging from 0 to 93 can be encoded (not all ASCII character are printable)

Phred quality scores

https://www.illumina.com/documents/products/technotes/technote_Q-Scores.pdf



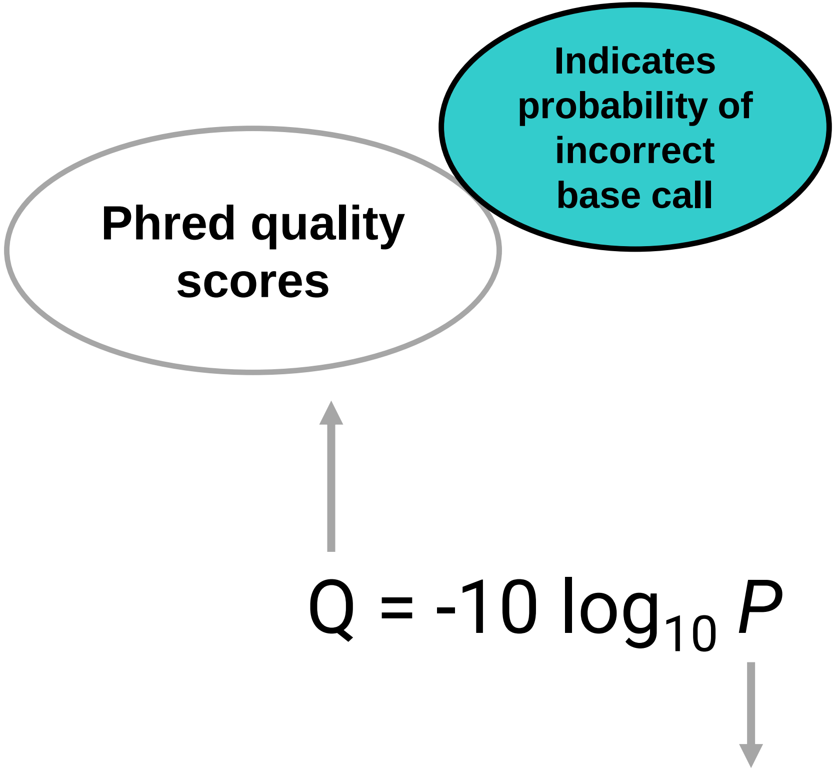
Indicates probability of incorrect base call

Phred quality scores

$$Q = -10 \log_{10} P$$

Probability of incorrect base call

Phred quality scores

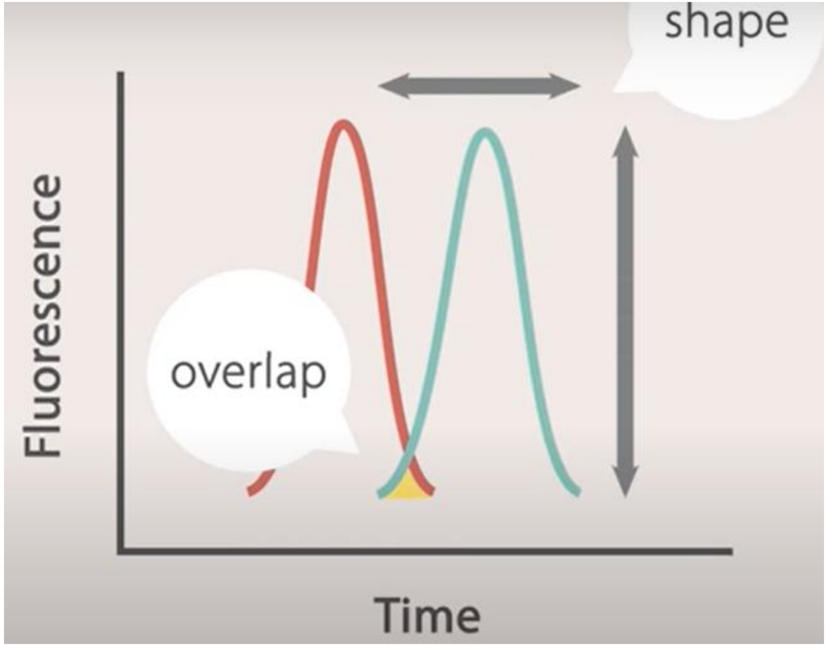


Indicates probability of incorrect base call

Phred quality scores

$$Q = -10 \log_{10} P$$

Probability of incorrect base call

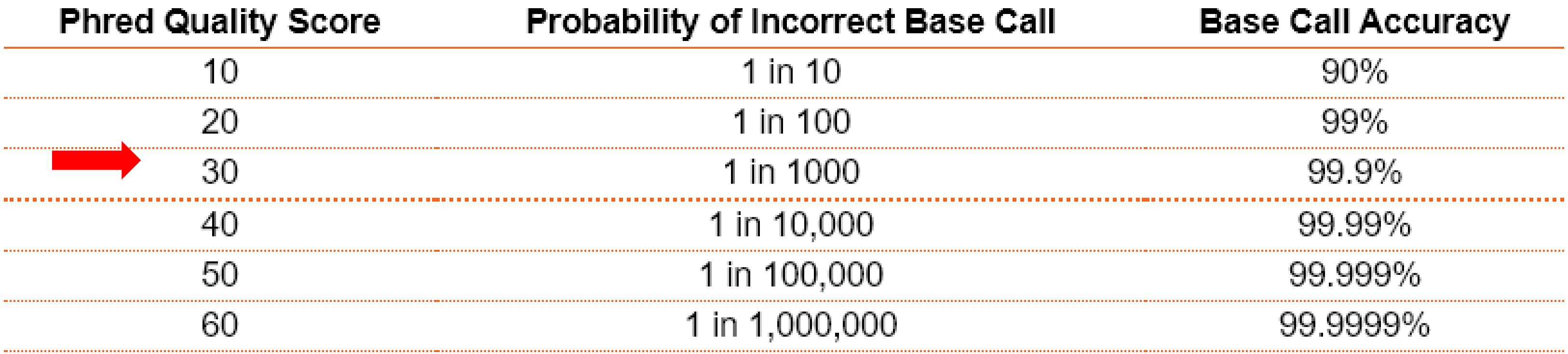


Probabilities are calculated by the machine by determining **fluorescence peak shape**, **resolution** and any potential **overlap** at every base

Phred quality scores

$$Q = -10 \log_{10} P$$

Metric used to assess the accuracy of a sequencing platform

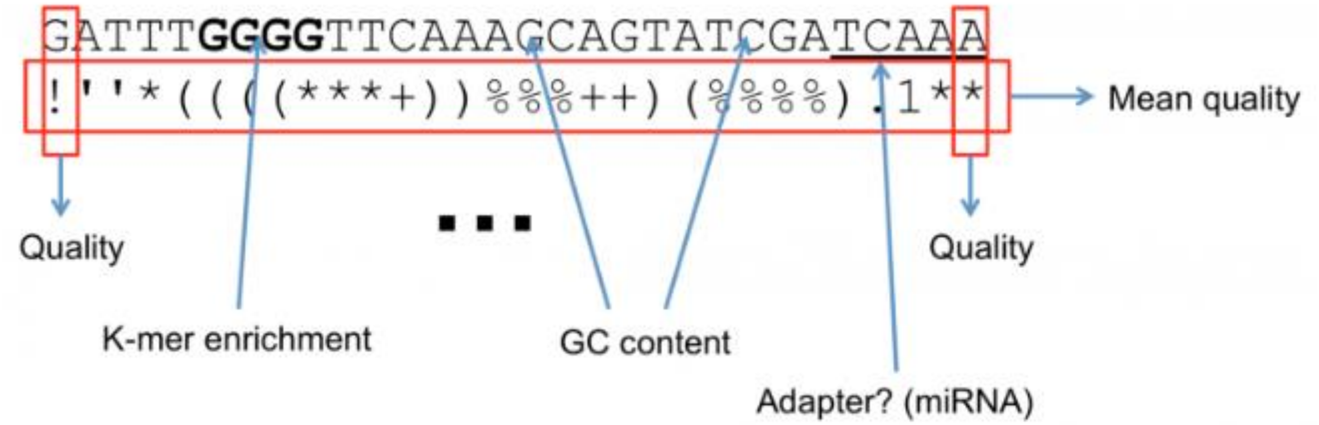


Phred Quality Score	Probability of Incorrect Base Call	Base Call Accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10,000	99.99%
50	1 in 100,000	99.999%
60	1 in 1,000,000	99.9999%

Figure 3 – Phred quality score chart

These scores tend to **drop near the end of reads**, because fluorescent overlap due to **incomplete dye cleavage** becomes a bigger factor the longer the read is.

Phred quality scores



GATTT**GGGG**TTCAAAGCAGTATCGATCAAA

!''*((((***+))%%%++)(%%%%).1**

Mean quality

Quality

...

Quality

K-mer enrichment

GC content

Adapter? (miRNA)

Common uses are to filter bases or entire reads if a particular quality threshold isn’t met

FastQC

- FastQC is an application which reads raw sequence data from high throughput sequencers and runs a set of **quality checks** to produce a **report** which allows you to quickly assess the **overall quality** of your **run** and to spot any potential **problems or biases**.

The main functions of FastQC are

- **Import of data** from BAM, SAM or FastQ files (any variant)
- Providing a quick overview to tell you in which areas there may be **problems**
- Summary **graphs and tables** to quickly assess your data
- Export of results to an HTML based permanent **report**
- **Offline operation** to allow automated generation of reports without running the interactive application

FastQC Practical 2. OHD NGS data analysis (a.y. 2022-2023)

- FastQC is available at *https://www.bioinformatics.babraham.ac.uk/projects/fastqc/*

- FastQC is a **java application**. You need to download and install a suitable 64-bit JRE and make sure that the java application is in your path

Upload Java for free from: https://adoptium.net/temurin/releases/?version=11

```
Ubuntu: sudo apt install default-jre
java -version
```

- FastQC can be run either as an **interactive graphical application** or in a non-interactive way (say as **part of a pipeline**) which will generate an HTML report for each file you process.

FastQC Practical 2. OHD NGS data analysis (a.y. 2022-2023)

- You can find Sample Fastq files at:

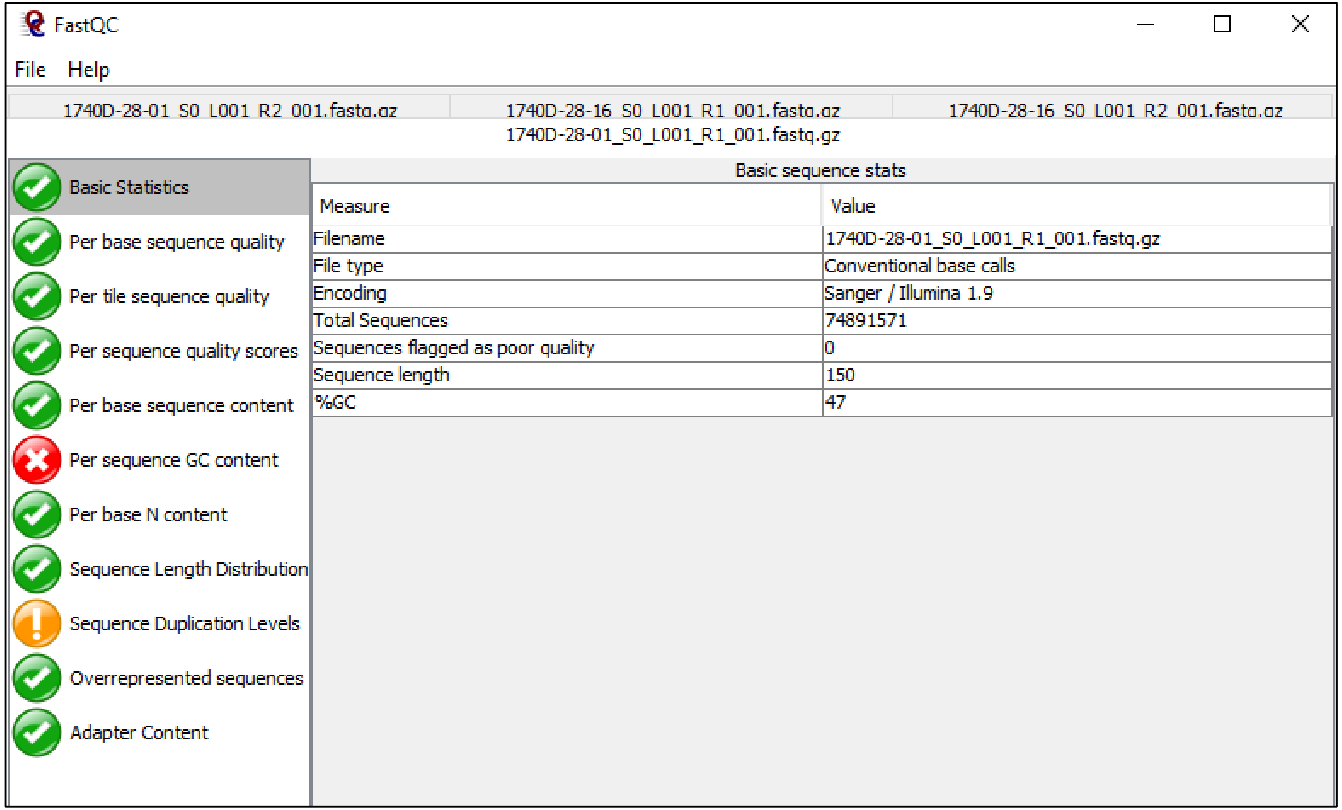
https://www.applied-maths.com/download/fastq-
or
fileshttps://zenodo.org/record/3736457#.Y3NBbHbMLIU

- *Answer the questions at:*

https://docs.google.com/forms/d/e/1FAIpQLScXl-BSdUOVc8DxLZwLNKRABiM6nRjneGW3_89Zjhd2W7lFeg/viewform?usp=pp_url

- *Deadline 29 November 2022. You can complete the test in class!*

FastQC: basic operations



1740D-28-01_S0_L001_R2_001.fastq.gz | 1740D-28-16_S0_L001_R1_001.fastq.gz | 1740D-28-16_S0_L001_R2_001.fastq.gz

1740D-28-01_S0_L001_R1_001.fastq.gz

- ✓ Basic Statistics
- ✓ Per base sequence quality
- ✓ Per tile sequence quality
- ✓ Per sequence quality scores
- ✓ Per base sequence content
- ✗ Per sequence GC content
- ✓ Per base N content
- ✓ Sequence Length Distribution
- ! Sequence Duplication Levels
- ✓ Overrepresented sequences
- ✓ Adapter Content

Basic sequence stats

Measure	Value
Filename	1740D-28-01_S0_L001_R1_001.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	74891571
Sequences flagged as poor quality	0
Sequence length	150
%GC	47

- Open a file

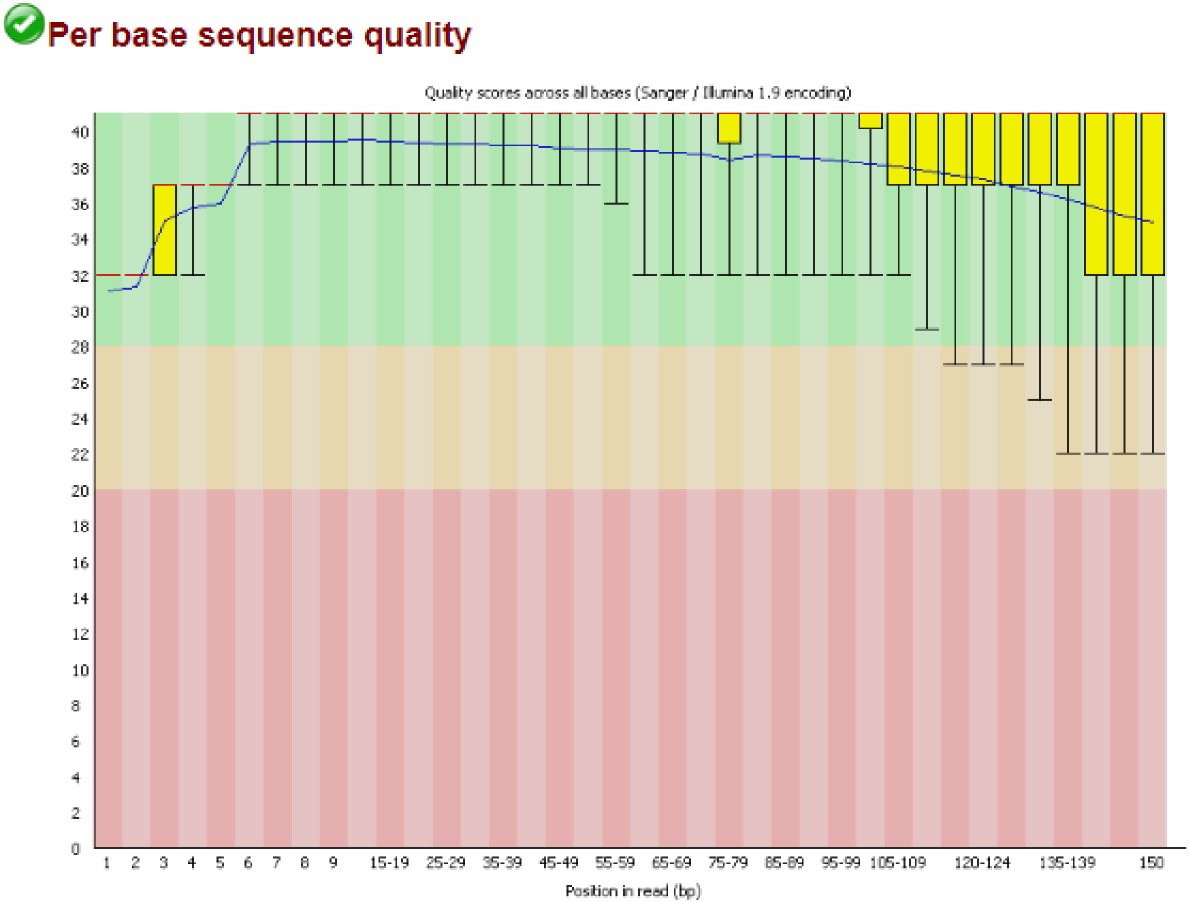
- Evaluating Results

Each test is flagged as a pass, warning or fail depending on how far it departs from what you'd expect from a normal large dataset with no significant biases.

✓ ✗ ⚠

- Save a report

FastQC module: per base sequence quality



Overview of the range of **quality values** across all bases at **each position** in the FastQ file

•The central **red line** is the median value

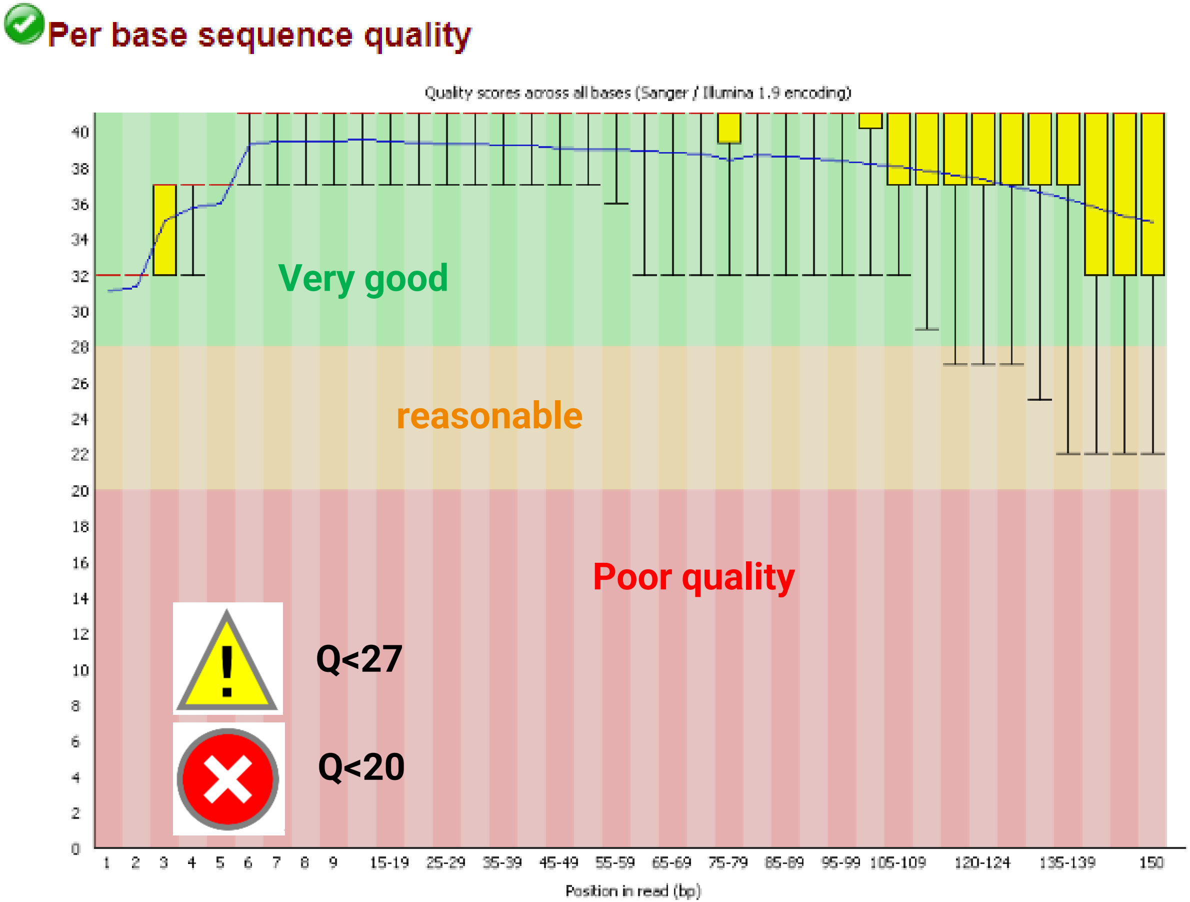
•The **yellow box** represents the inter-quartile range (25-75%)

•The upper and lower **whiskers** represent the 10% and 90% points

•The **blue line** represents the mean quality

The quality of calls on most platforms will degrade as the run progresses

FastQC module: per base sequence quality



Overview of the range of **quality values** across all bases at **each position** in the FastQ file

•The central **red line** is the median value

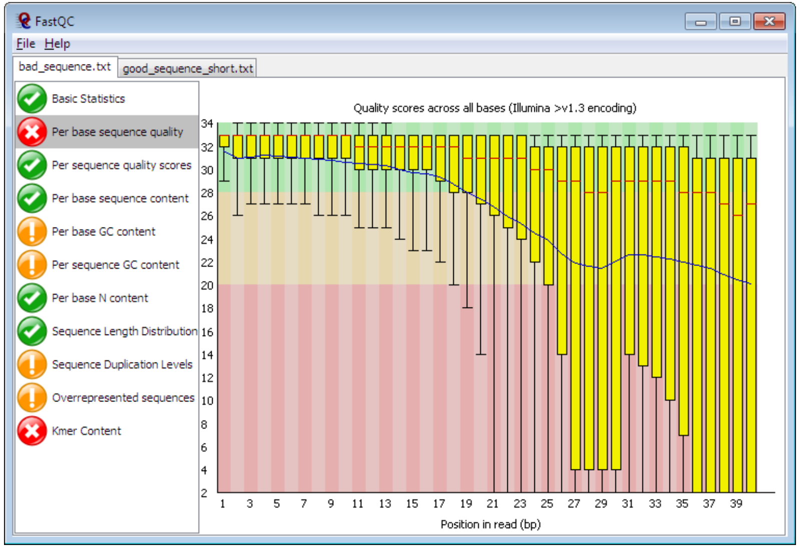
•The **yellow box** represents the inter-quartile range (25-75%)

•The upper and lower **whiskers** represent the 10% and 90% points

•The **blue line** represents the mean quality

The quality of calls on most platforms will degrade as the run progresses

FastQC module: per base sequence quality



⚠ Q<25

✖ Q<20

A bad per base sequence graph

- General degradation of quality over the duration of long runs

>>> perform quality trimming where reads are truncated based on their average quality

- Short loss of quality earlier in the run, which then recovers to produce later good quality (sequence transient problem with the run e.g bubbles passing through a flowcell)

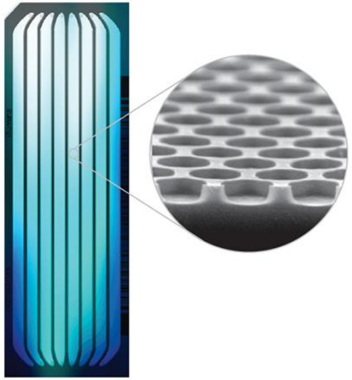
>>> masking bases during subsequent mapping or assembly

- very low coverage for a given base range

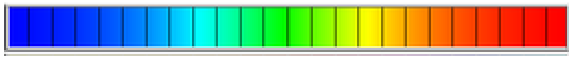
>>> check how many sequences were responsible for triggering an error

(look at the length distribution module)

FastQ: per tile sequence quality

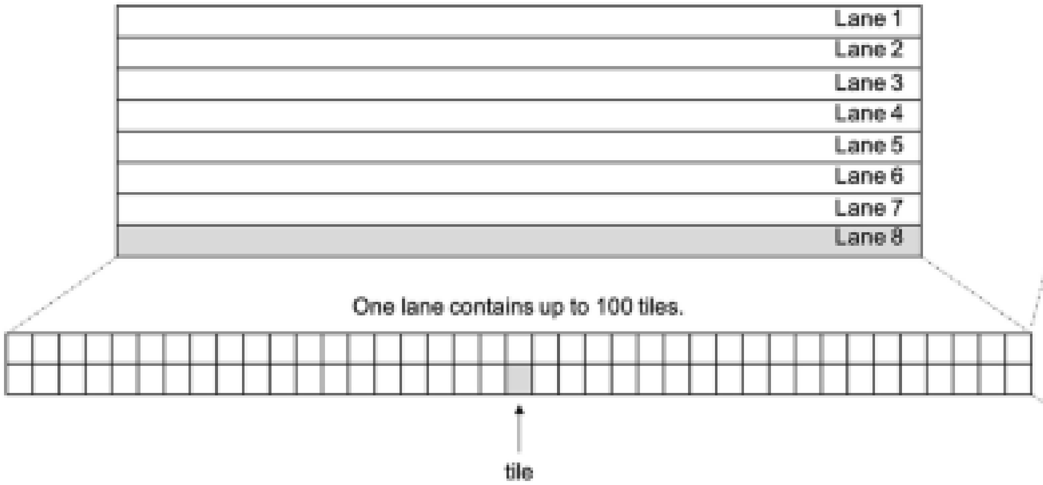


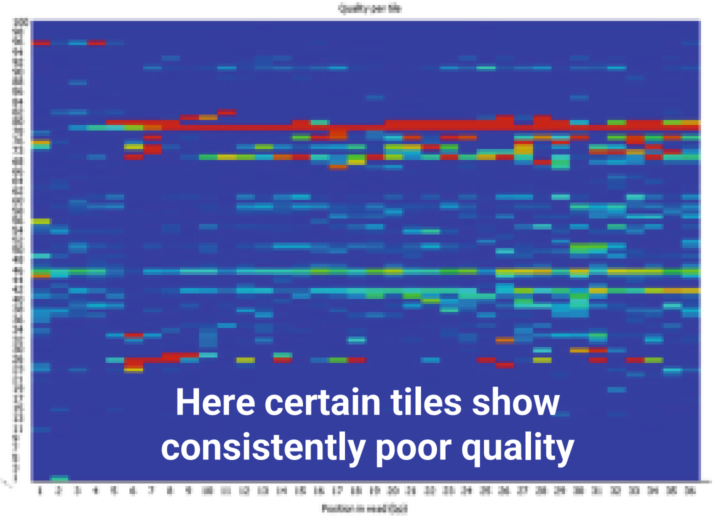
Look at the **quality scores** from **each tile** across all of **your bases** to see if there was a loss in quality associated with only one part of the **flowcell**.



>= average quality

< average quality





Here certain tiles show consistently poor quality

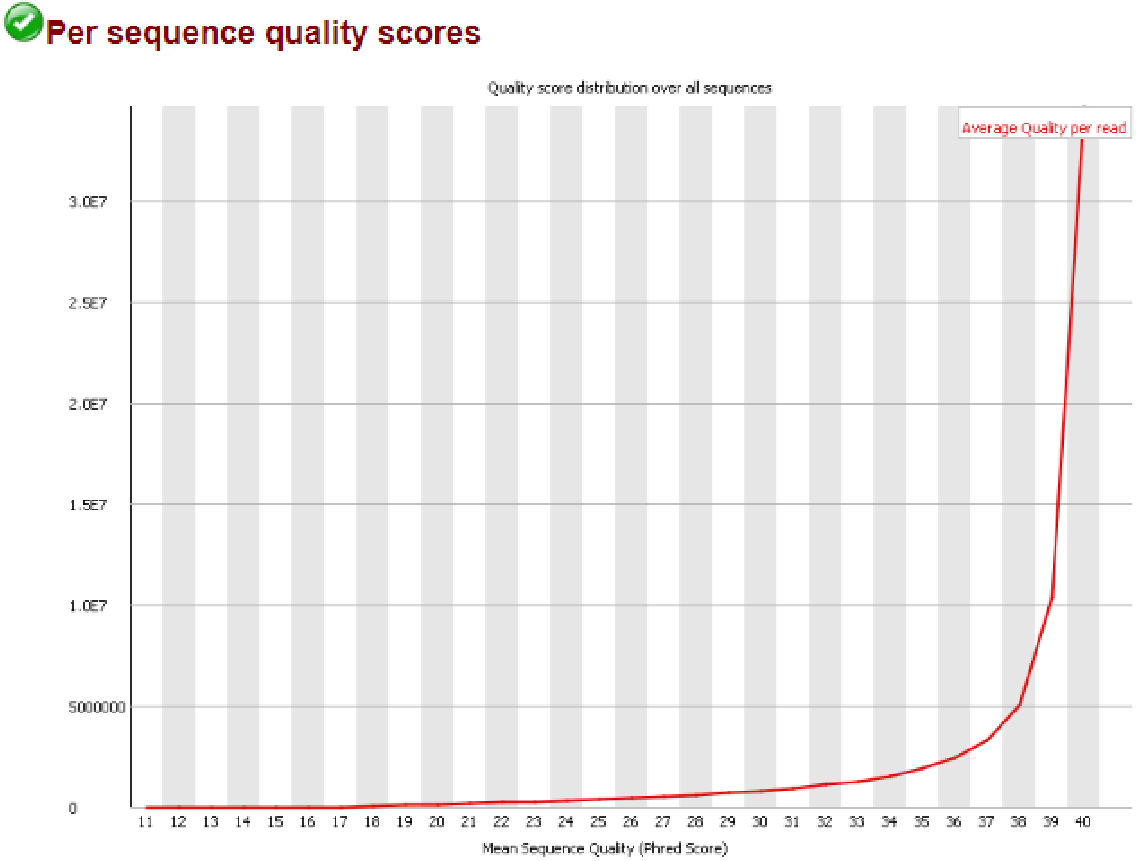
FastQ: per tile sequence quality ⚠ ❌

- **Smudges** on the flowcell or **debris** inside the flowcell lane
- A flowcell is **overloaded** (In this case events appear all over the flowcell rather than being confined to a specific area or range of cycles)

We would generally

- Ignore errors which mildly affected a small number of tiles for only 1 or 2 cycles
- but would pursue larger effects which showed **high deviation** in scores, or which **persisted for several cycles**.

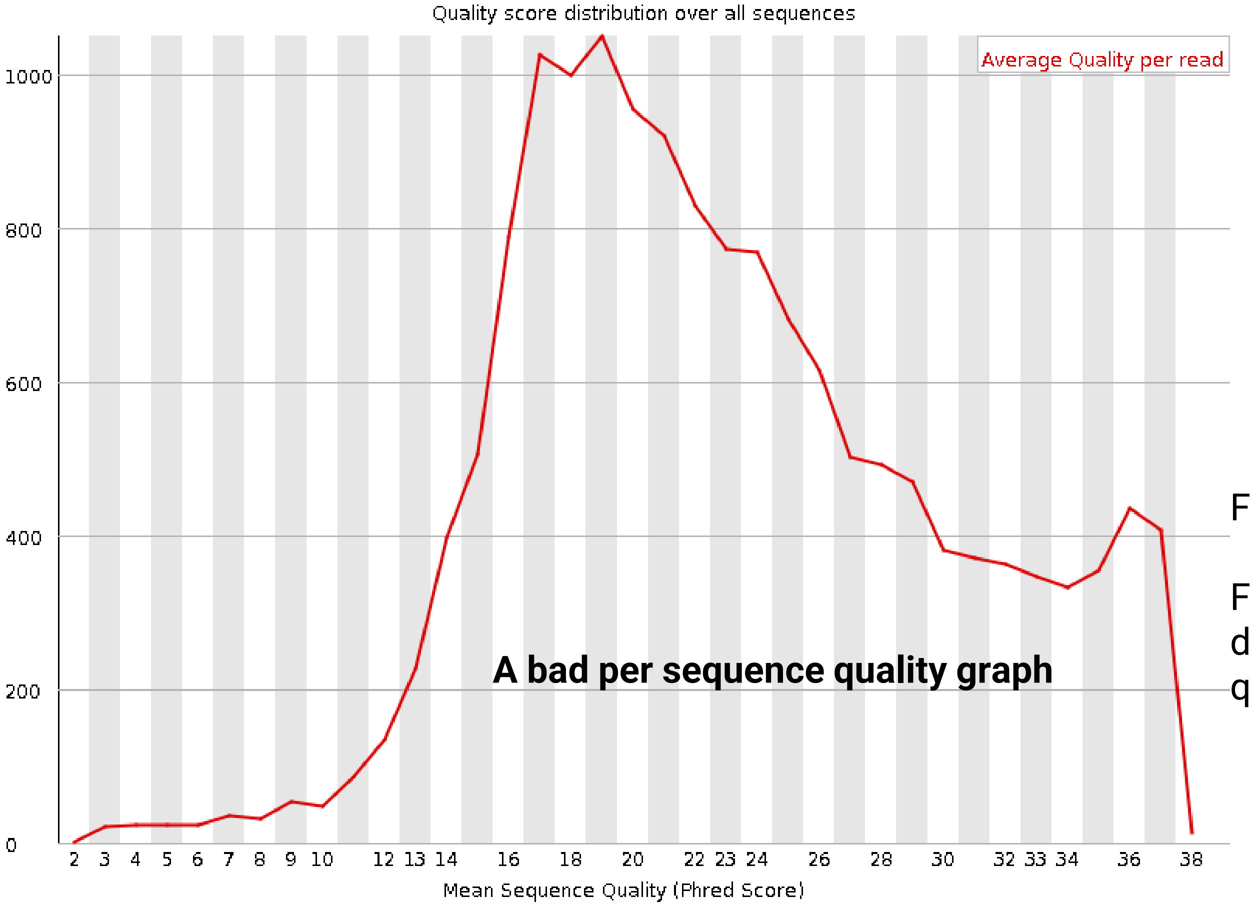
FastQC module: per sequence quality score



See if a **subset** of your sequences have universally **low quality** values (e.g. poorly imaged on the edge of the field of view)
Indicates a systematic problem - possibly with just part of the run (for example one end of a flowcell)

✓

FastQC: per sequence quality score



A bad per sequence quality graph

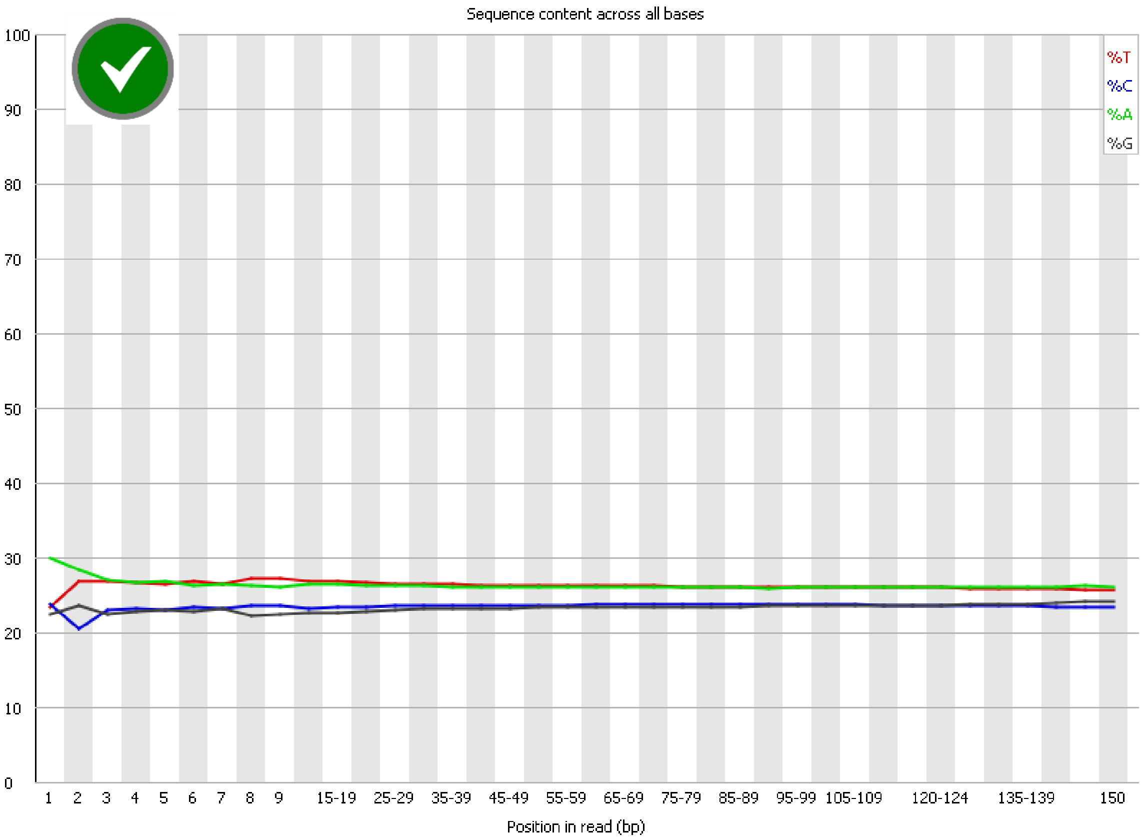
⚠ **Q<27,** 0.2% error rate

❌ **Q<20,** 1% error rate

For long runs>> quality trimming

For bimodal or complex distribution>> evaluate per tile quality

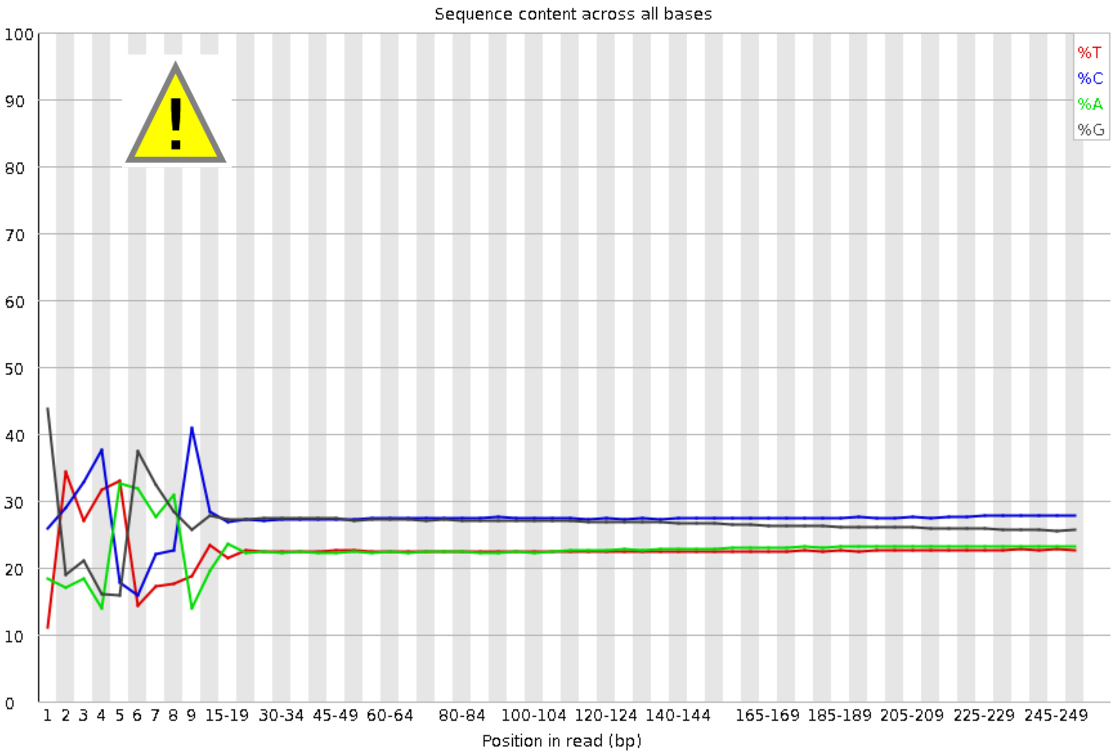
FastQC module: **per base sequence content**



Proportion of each base position in a file for which each of the four normal DNA bases has been called

DNA sequencing:
remain relatively constant over the length of the read with %A=%T and %G=%C
The lines in this plot should run **parallel** with each other

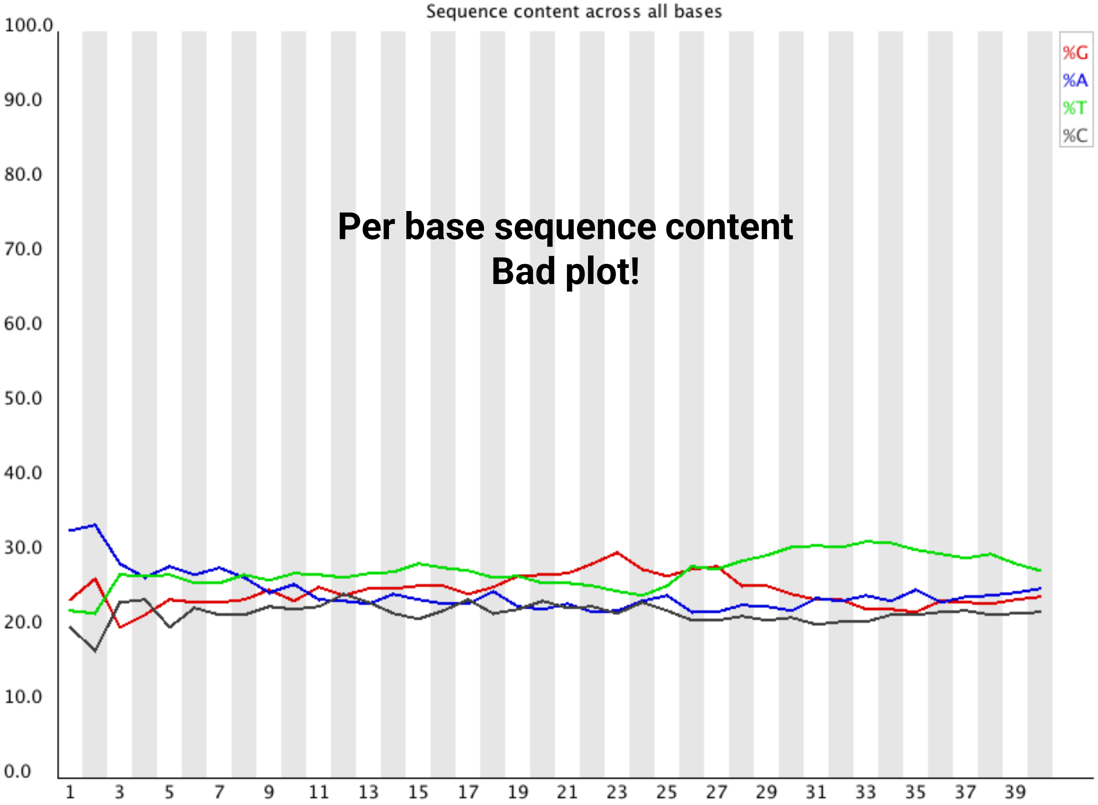
FastQC module: per base sequence content



RNA sequencing
non-uniform distribution of bases for the first 10-15 nucleotides

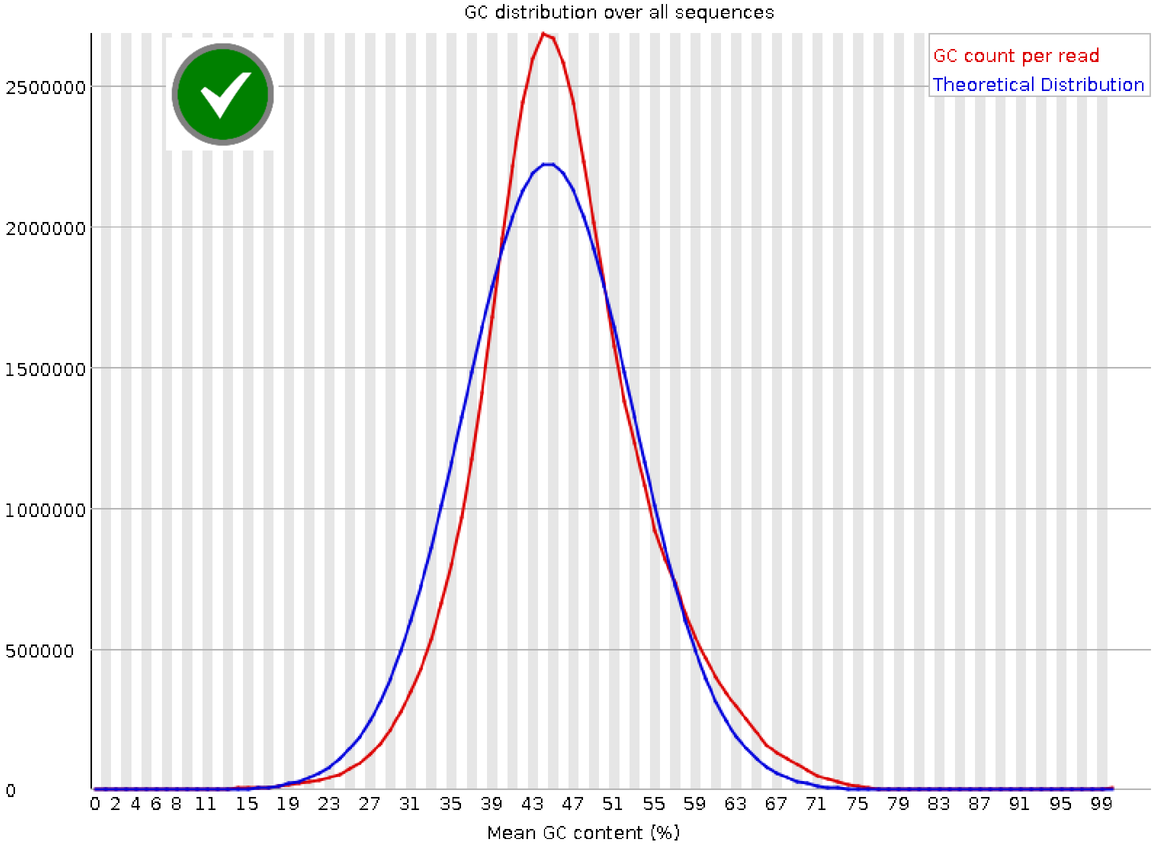
The sequence is good!

FastQC module: per base sequence content



Per base sequence content
Bad plot!

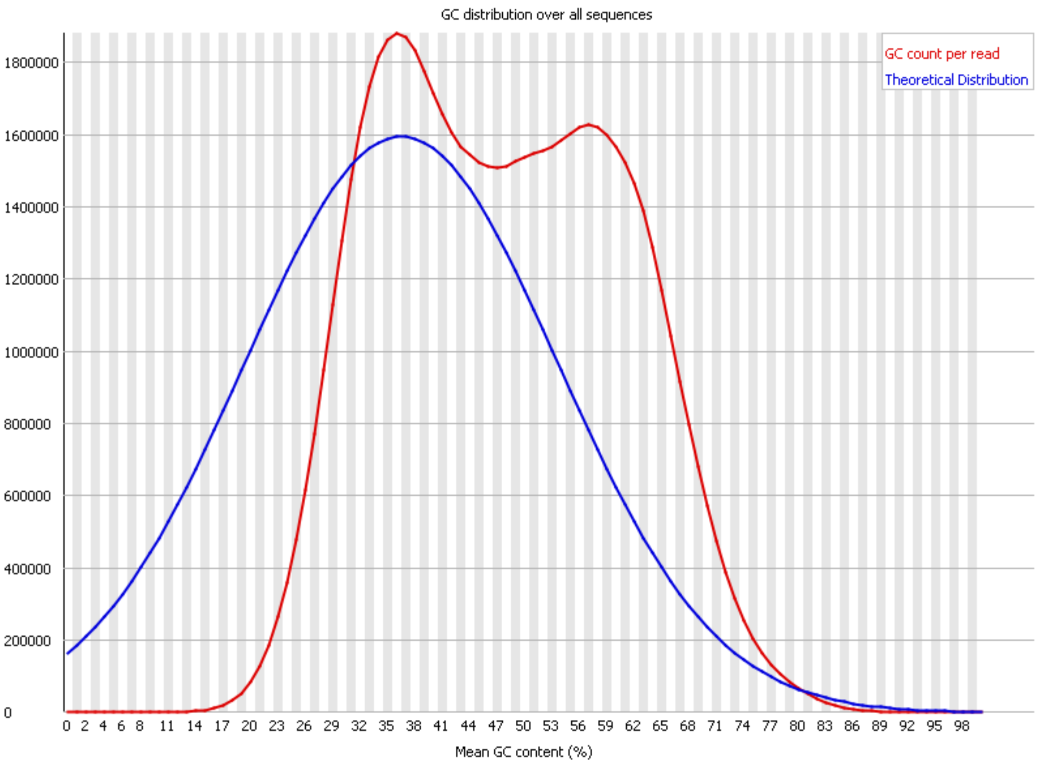
FastQC: per sequence GC content



GC content across the whole length of each sequence in a file and compares it to a modelled **normal distribution** of GC content

The central peak corresponds to the overall GC content of the underlying genome

FastQC: per sequence GC content

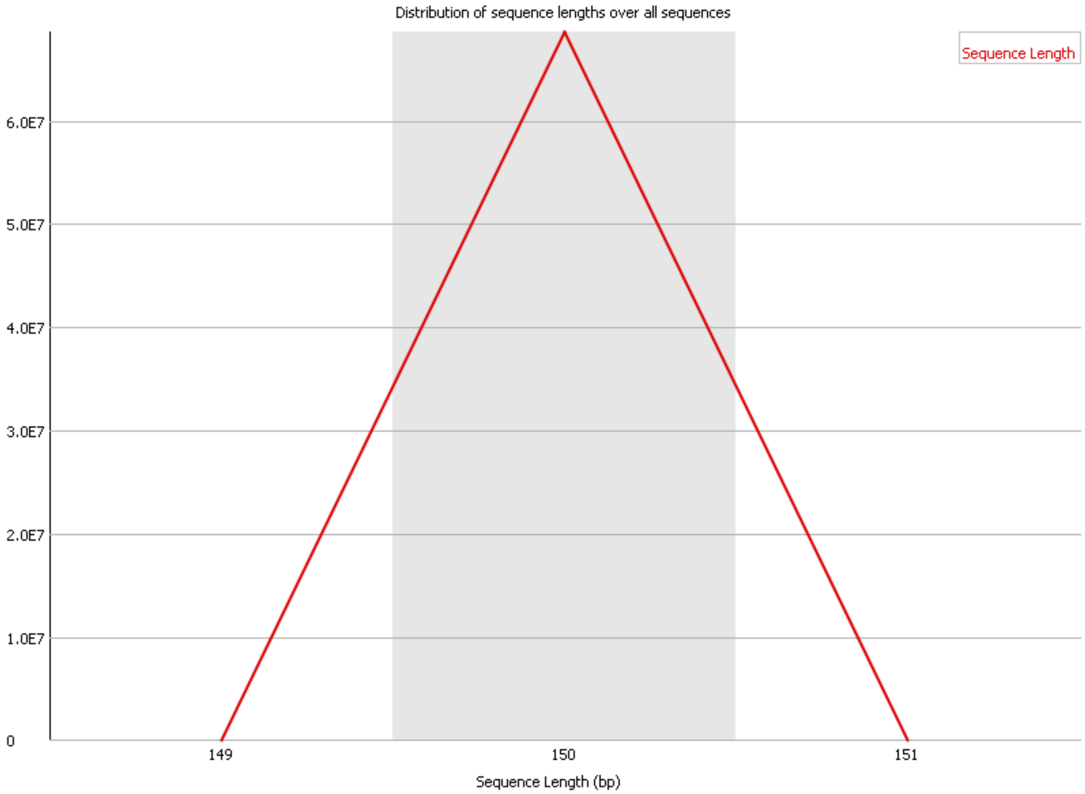


⚠

❌

An unusually shaped distribution could indicate a contaminated library or some other kinds of biased subset

FastQC: per length distribution



Some high throughput sequencers generate sequence fragments of **uniform length**, but others can contain reads of wildly varying lengths.

Even within uniform length libraries some pipelines will trim sequences to remove poor quality base calls from the end

⚠ Not uniform

✖ Sequences with zero length

For some sequencing platforms it is entirely normal to have different read lengths so warnings here can be ignored

BioComputing

FastQ: Duplicate sequences

Duplicates: Not biological copies but results of technical issues:

- In a diverse library most sequences will occur only once in the final set.
- A **low level of duplication** may indicate a very **high level of coverage** of the target sequence
- **A High level of duplication** is more likely to indicate some kind of **enrichment bias** (eg PCR over amplification)
- Same read was detected twice (**borders of tiles**)

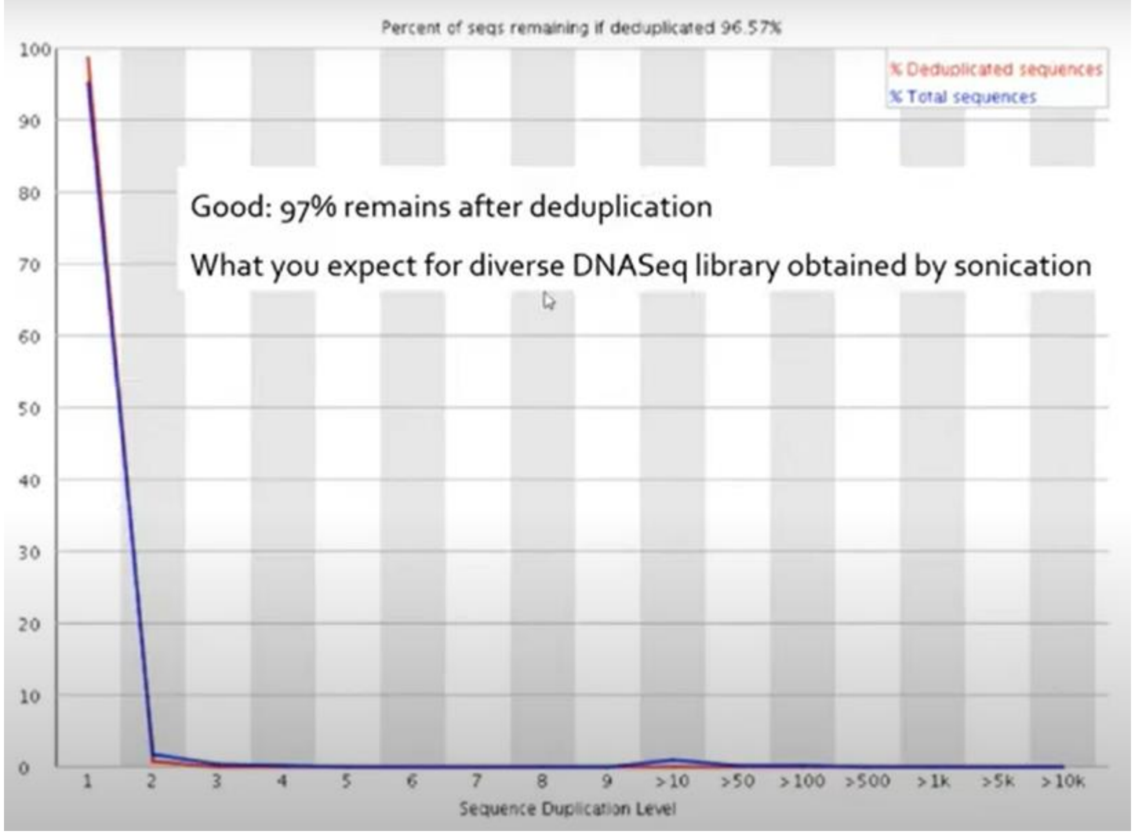
>>> remove because duplicates will distort results

The plot tell you what extent you are **wasting the sequencing capacity** you have paid for by simply resequencing the exact same sequences over and over again.

RNA-seq data: normal (expressed transcripts of a few genes)

>>> Do not remove!

FastQ: Duplicate sequences



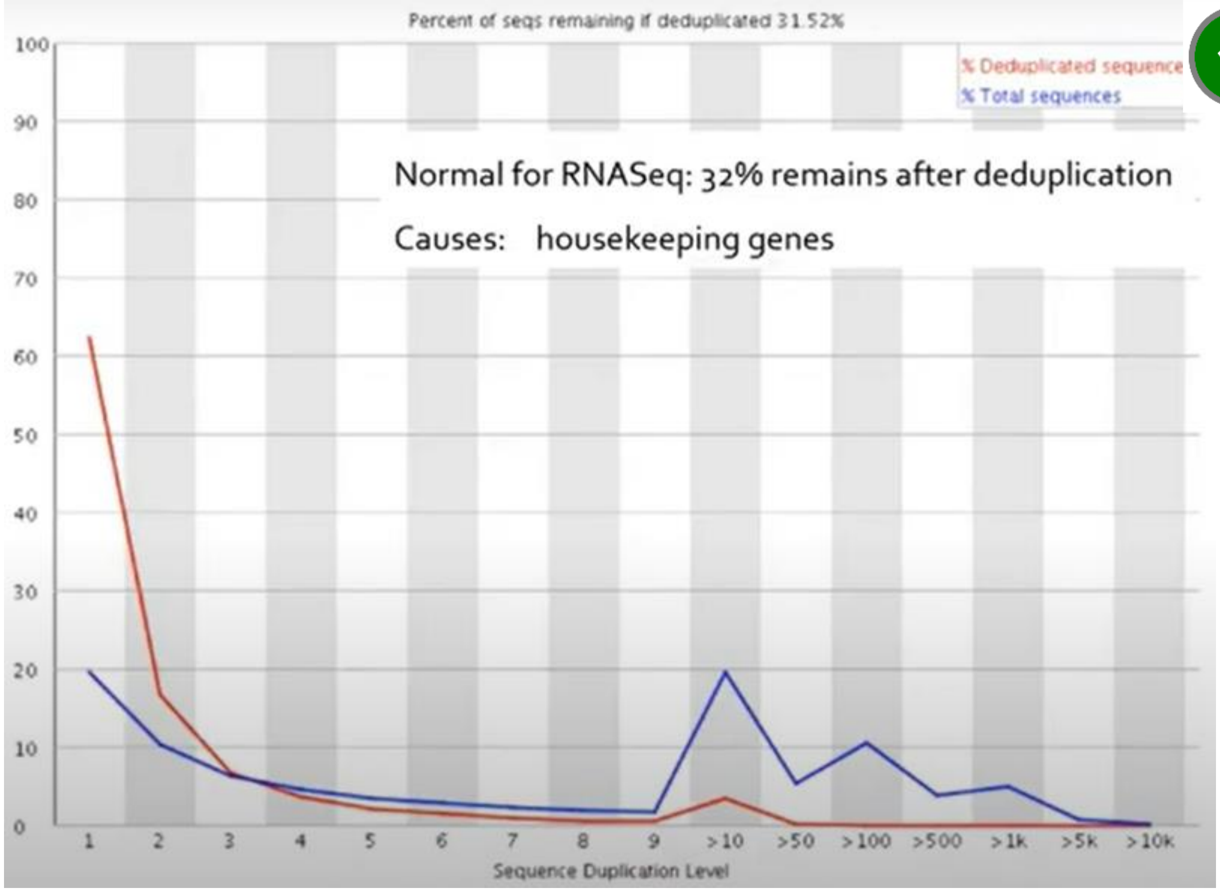
Good: 97% remains after deduplication

What you expect for diverse DNASeq library obtained by sonication

✓

The analysis occurs only for the first 100,000 different sequences seen

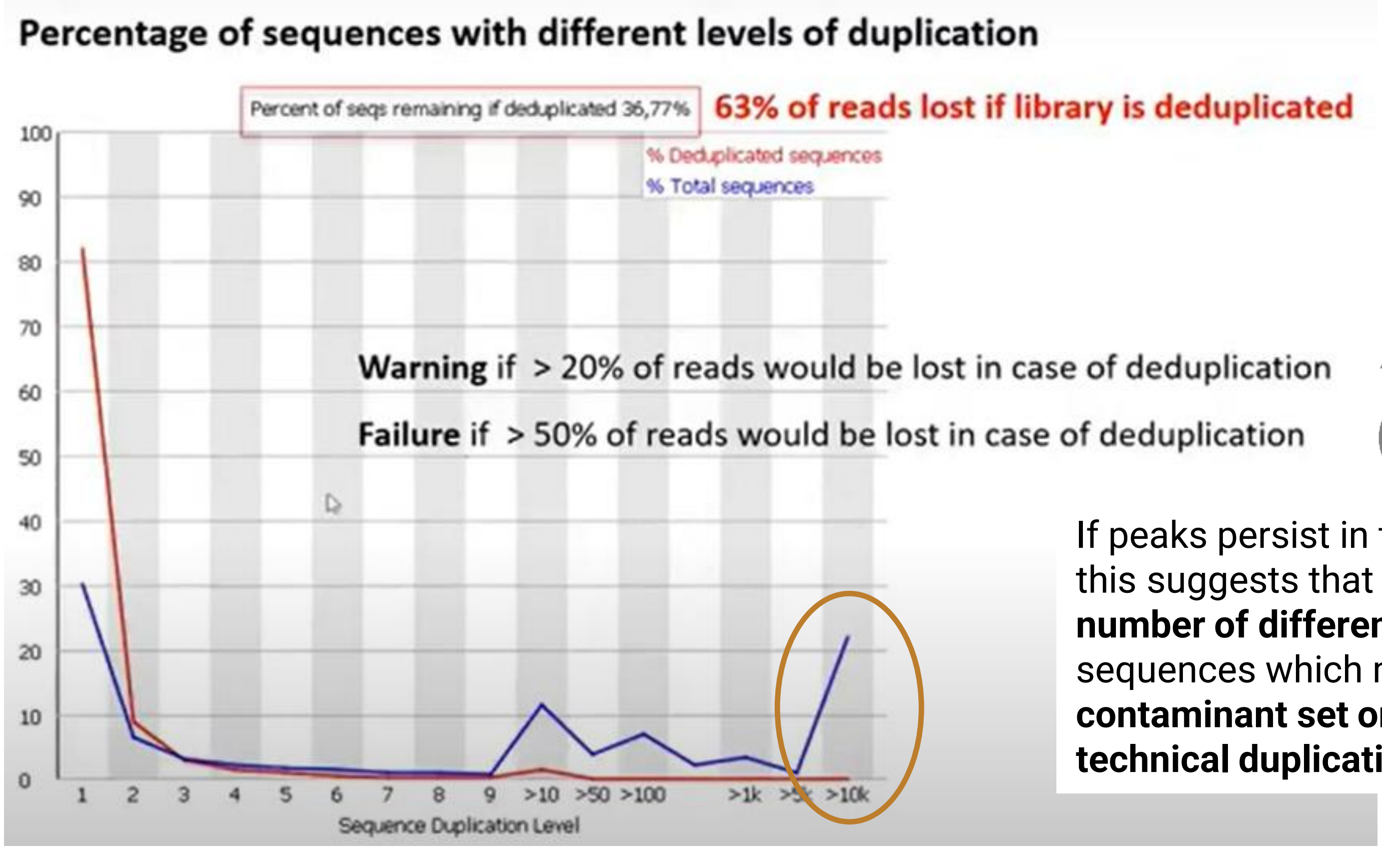
FastQ: Duplicate sequences



Normal for RNASeq: 32% remains after deduplication

Causes: housekeeping genes

FastQ: Duplicate sequences



Percentage of sequences with different levels of duplication

Percent of seqs remaining if deduplicated 36,77%

63% of reads lost if library is deduplicated

Warning if > 20% of reads would be lost in case of deduplication ⚠

Failure if > 50% of reads would be lost in case of deduplication ❌

If peaks persist in the blue trace then this suggests that there are a **large number of different highly duplicated** sequences which might indicate either a **contaminant set or a very severe technical duplication**.

2. Sequence alignment

Fastq format

Sequence Alignment

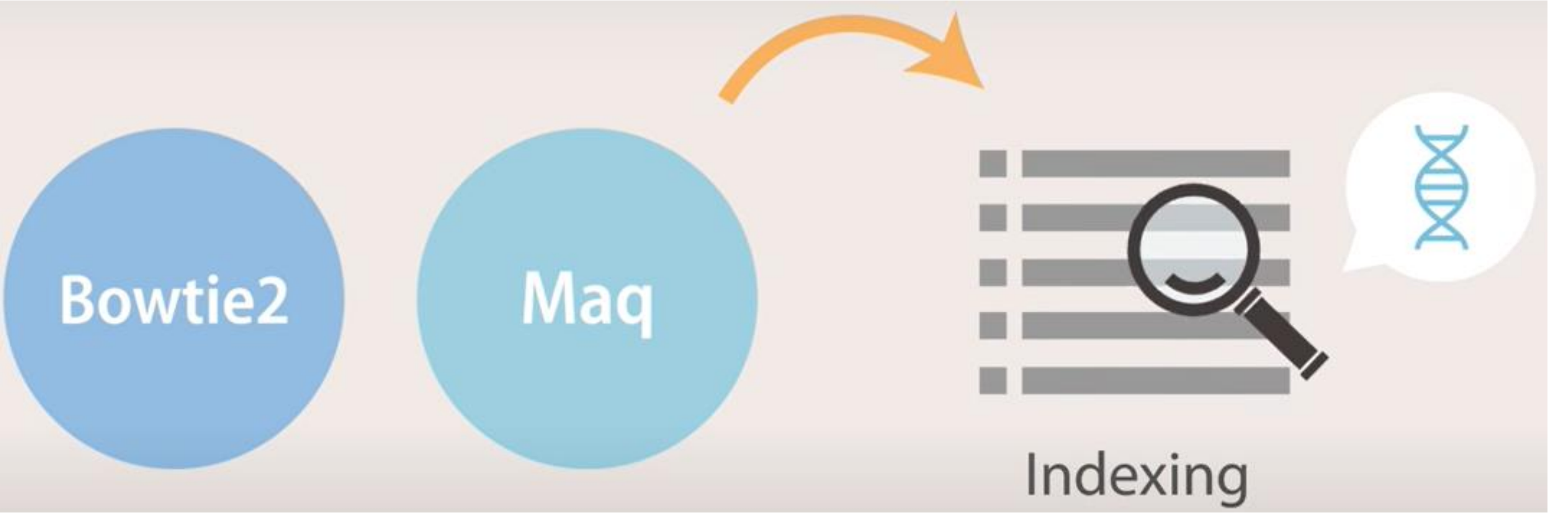
Reference Genome

Sequence Alignment programs

BWA
Bowtie2
Maq
Stampy
Novoalign

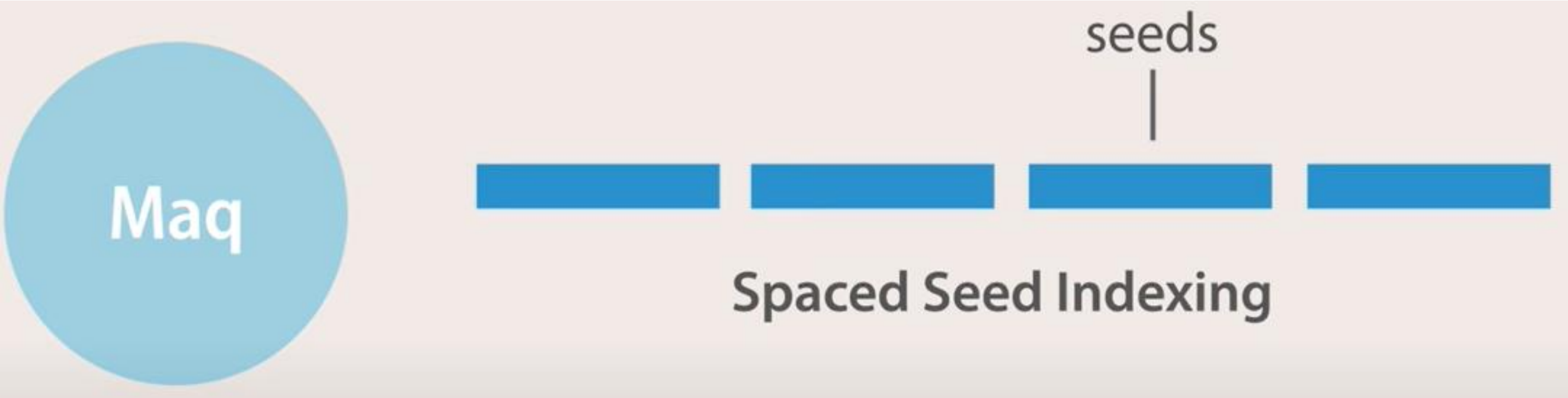
These programs are exponentially better suited than tools such as BLAST, because they **use heuristic (approximate) algorithms** to make the alignment process extremely fast and able to deal with millions or billions of reads being mapped against very large reference genomes

Sequence Alignment programs



Use a computational strategy called **indexing**, which works much like a index at the end of a book to speed up mapping algorithms that takes an index of a large DNA sequence and rapidly finding shorter sequences embedded within it

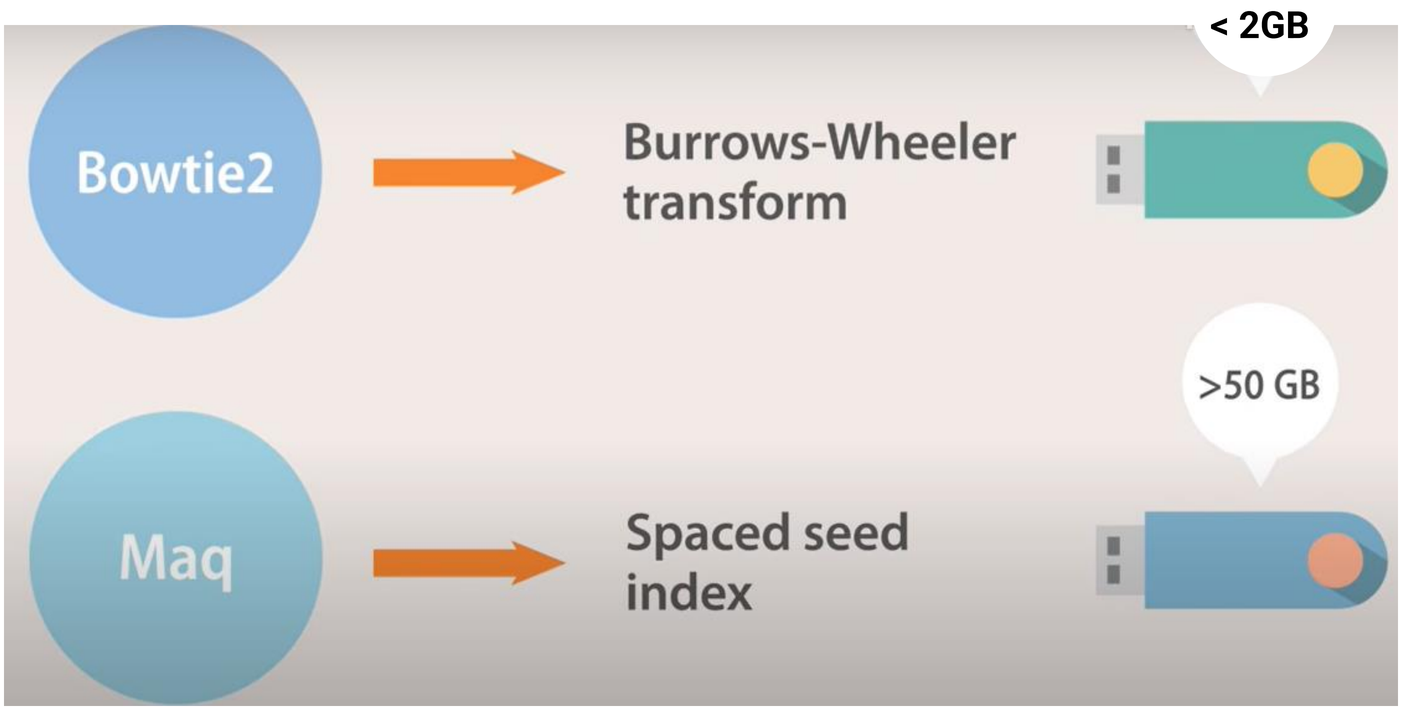
Sequence Alignment programs



Maq uses **spaced seed indexing** where a read is divided into four segments of equal length called seeds

Bowtie uses a different techniques called Burrows-Wheeler transform that can fit the entire human genome in less than two gigabytes of memory

Sequence Alignment programs



Bowtie2 → Burrows-Wheeler transform

< 2GB

Maq → Spaced seed index

>50 GB

Sequence Alignment programs

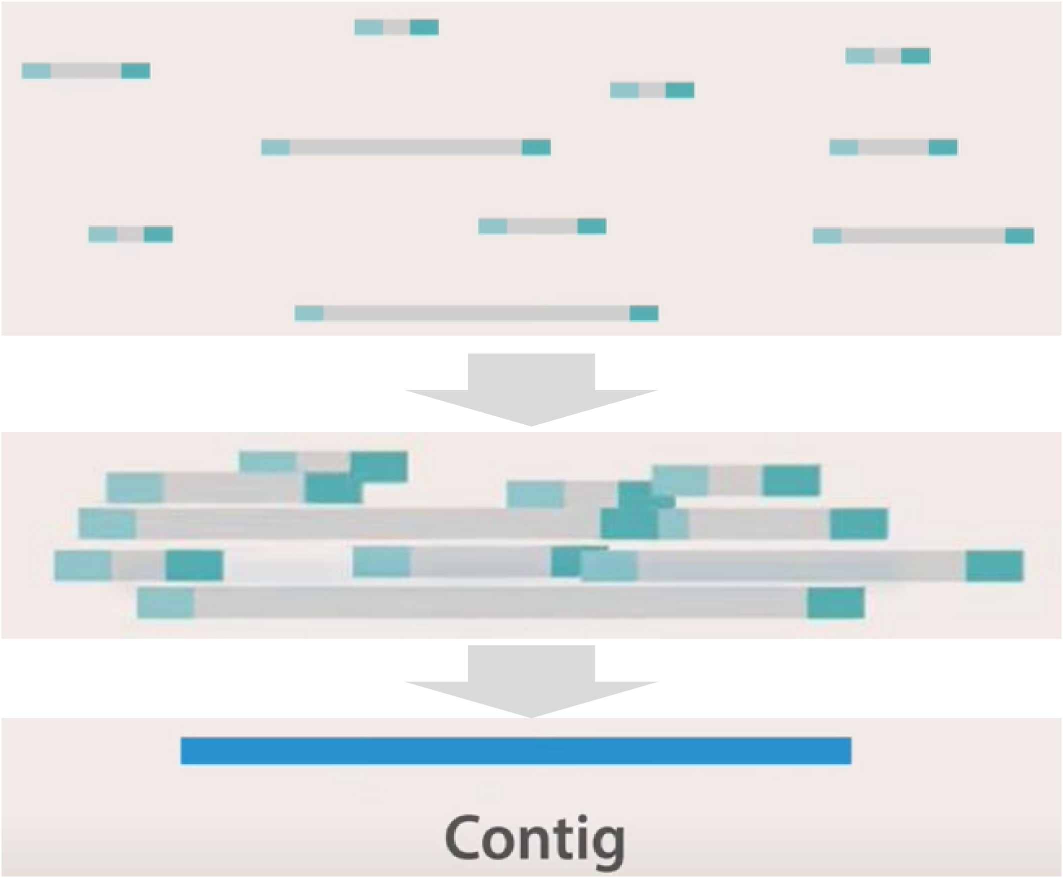
If the orgaism being sequenced does **not** have a **reference Genome** available, the reads must be aligned de novo, using programs such as **ABySS** and **SOAPdenovo**

Fastq format

Sequence Alignment

Aligned *de novo*

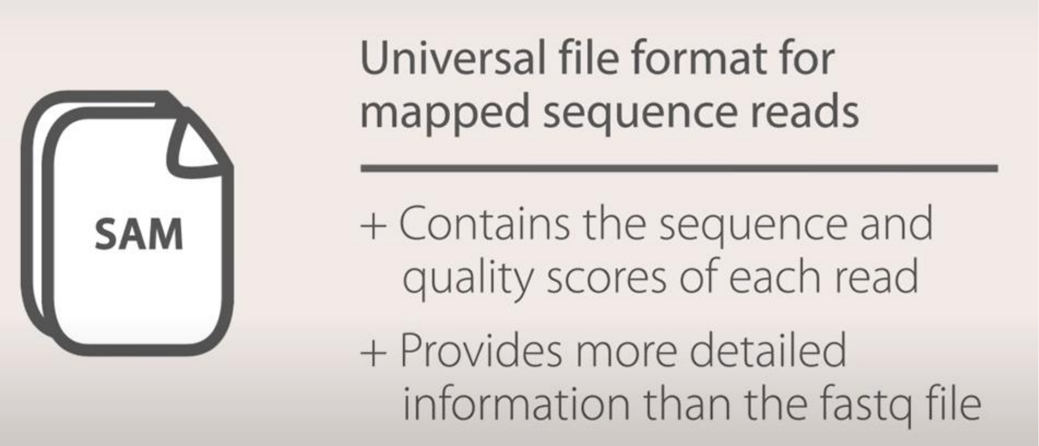
Sequence Alignment programs: de novo assembly



A contig encompasses the entire genome of the organism

Sequence Alignment: FastQ >> SAM fles

- Alignment of sequenced fastq data through either **reference** or **de novo methods** will result in the generation of a **SAM file**



Universal file format for mapped sequence reads

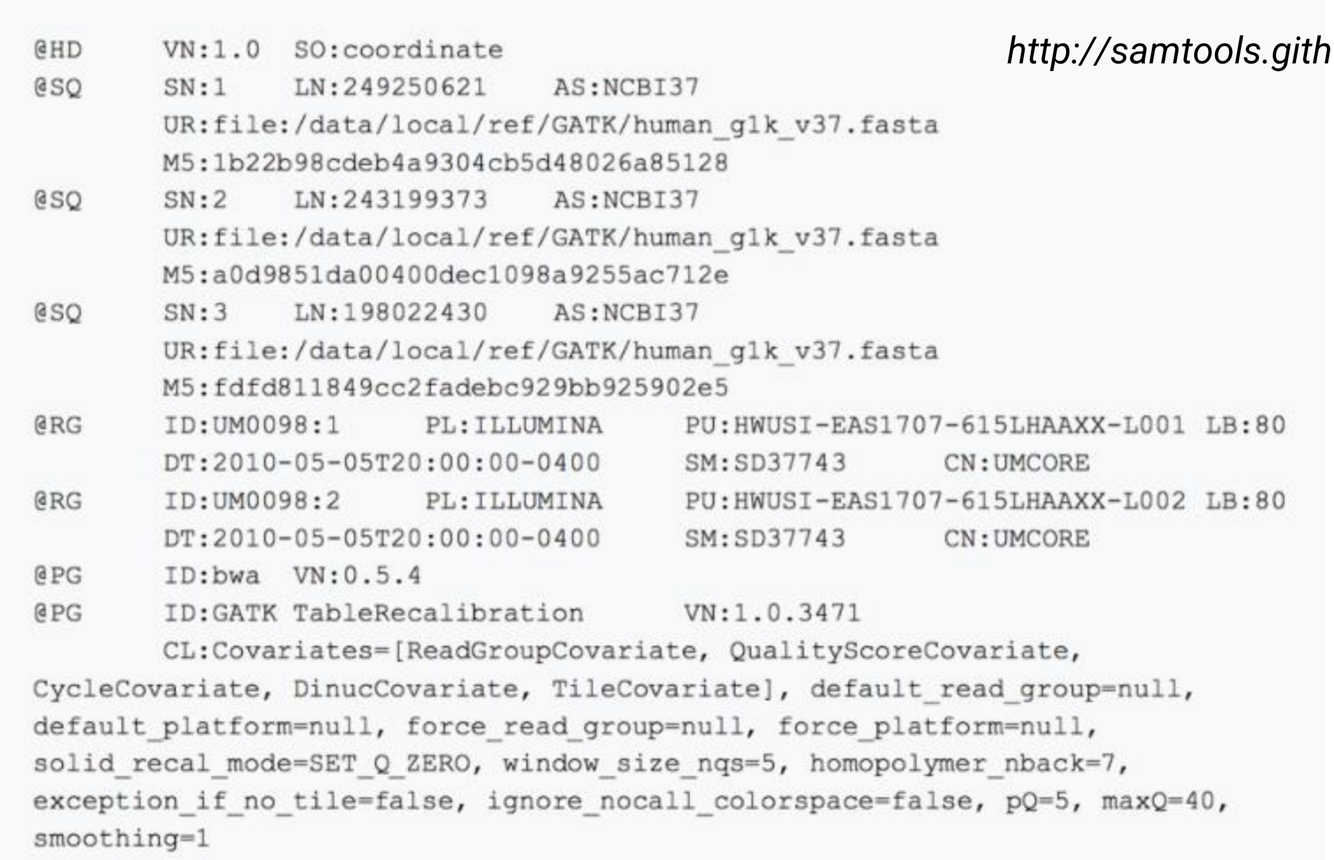
+ Contains the sequence and quality scores of each read

+ Provides more detailed information than the fastq file

- **Flexible**
- **Simple**
- **Compact in file size**

- It specifies information about the location in the genome the reads map to and more…
- The SAM format consists of a header and an alignment section, which has 11 mandatory fields and a variable number of optional fields

SAM file: Example of header lines

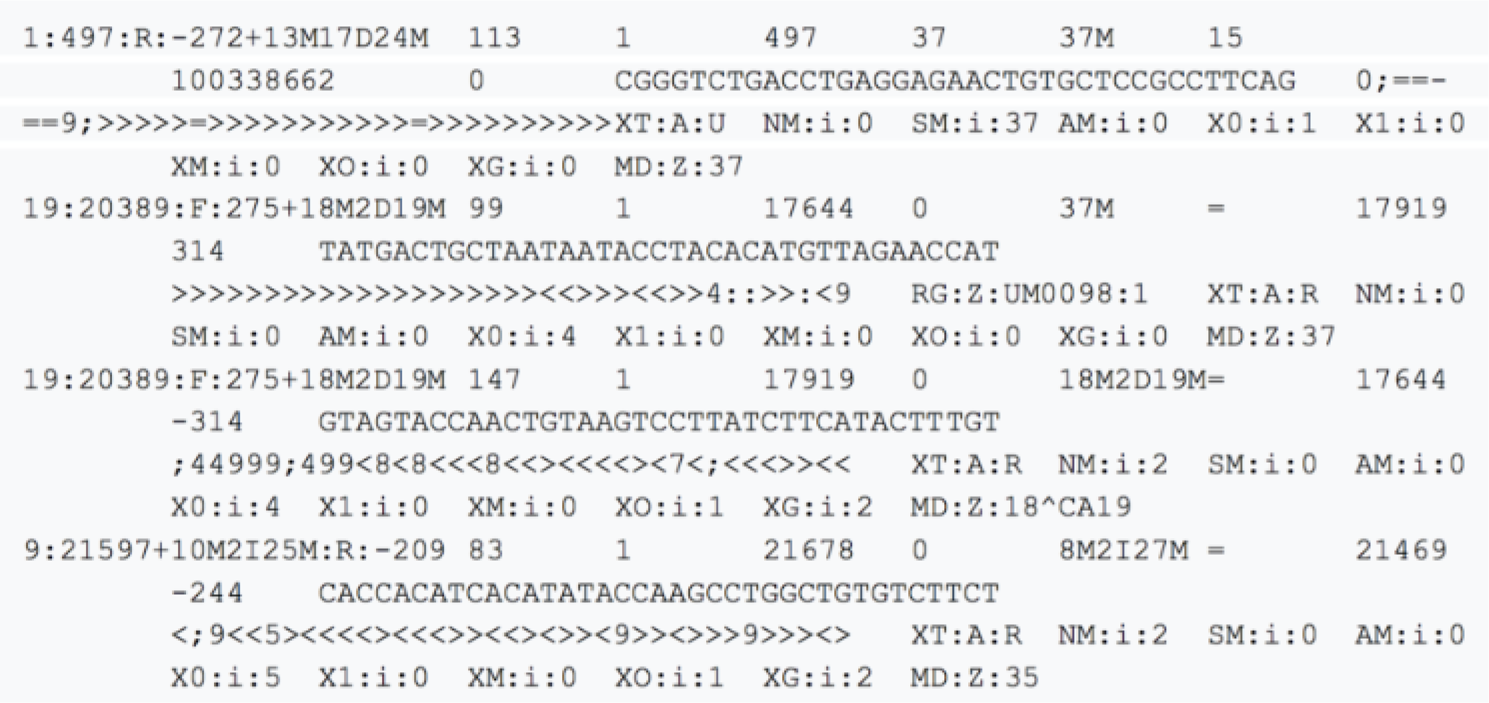


```
@HD     VN:1.0  SO:coordinate
@SQ     SN:1    LN:249250621    AS:NCBI37
        UR:file:/data/local/ref/GATK/human_g1k_v37.fasta
        M5:1b22b98cdeb4a9304cb5d48026a85128
@SQ     SN:2    LN:243199373    AS:NCBI37
        UR:file:/data/local/ref/GATK/human_g1k_v37.fasta
        M5:a0d9851da00400dec1098a9255ac712e
@SQ     SN:3    LN:198022430    AS:NCBI37
        UR:file:/data/local/ref/GATK/human_g1k_v37.fasta
        M5:fdfd811849cc2fadebc929bb925902e5
@RG     ID:UM0098:1     PL:ILLUMINA     PU:HWUSI-EAS1707-615LHAAXX-L001 LB:80
        DT:2010-05-05T20:00:00-0400     SM:SD37743      CN:UMCORE
@RG     ID:UM0098:2     PL:ILLUMINA     PU:HWUSI-EAS1707-615LHAAXX-L002 LB:80
        DT:2010-05-05T20:00:00-0400     SM:SD37743      CN:UMCORE
@PG     ID:bwa  VN:0.5.4
@PG     ID:GATK TableRecalibration      VN:1.0.3471
        CL:Covariates=[ReadGroupCovariate, QualityScoreCovariate,
CycleCovariate, DinucCovariate, TileCovariate], default_read_group=null,
default_platform=null, force_read_group=null, force_platform=null,
solid_recal_mode=SET_Q_ZERO, window_size_nqs=5, homopolymer_nback=7,
exception_if_no_tile=false, ignore_nocall_colorspace=false, pQ=5, maxQ=40,
smoothing=1
```

http://samtools.github.io/hts-specs/SAMv1.pdf

SAM file: Example of Alignment lines

http://samtools.github.io/hts-specs/SAMv1.pdf

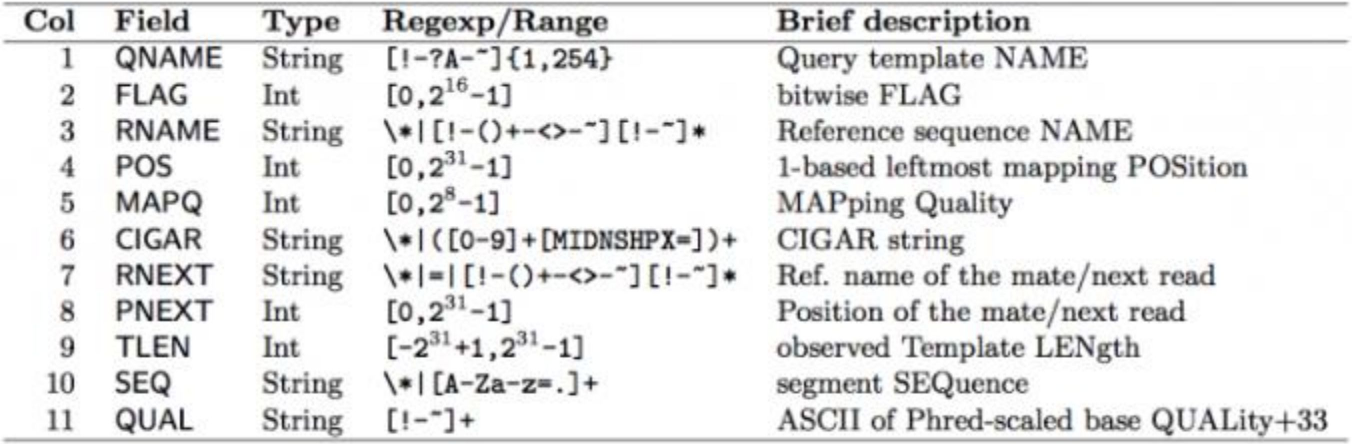


```
1:497:R:-272+13M17D24M  113     1       497     37      37M     15
        100338662       0       CGGGTCTGACCTGAGGAGAACTGTGCTCCGCCTTCAG   0;==-
==9;>>>>>=>>>>>>>>>>>=>>>>>>>>>>XT:A:U  NM:i:0  SM:i:37 AM:i:0  X0:i:1  X1:i:0
        XM:i:0  XO:i:0  XG:i:0  MD:Z:37
19:20389:F:275+18M2D19M 99      1       17644   0       37M     =       17919
        314     TATGACTGCTAATAATACCTACACATGTTAGAACCAT
        >>>>>>>>>>>>>>>>>>>><<>>><<>>4::>>:<9   RG:Z:UM0098:1   XT:A:R  NM:i:0
        SM:i:0  AM:i:0  X0:i:4  X1:i:0  XM:i:0  XO:i:0  XG:i:0  MD:Z:37
19:20389:F:275+18M2D19M 147     1       17919   0       18M2D19M=       17644
        -314    GTAGTACCAACTGTAAGTCCTTATCTTCATACTTTGT
        ;44999;499<8<8<<<8<<><<<<><7<;<<<>><<   XT:A:R  NM:i:2  SM:i:0  AM:i:0
        X0:i:4  X1:i:0  XM:i:0  XO:i:1  XG:i:2  MD:Z:18^CA19
9:21597+10M2I25M:R:-209 83      1       21678   0       8M2I27M =       21469
        -244    CACCACATCACATATACCAAGCCTGGCTGTGTCTTCT
        <;9<<5><<<<><<<><<><>><9>><>>>9>>><>    XT:A:R  NM:i:2  SM:i:0  AM:i:0
        X0:i:5  X1:i:0  XM:i:0  XO:i:1  XG:i:2  MD:Z:35
```

The **11 mandatory fields** of the alignment section include information on mapping quality, fragment position, quality control, sequence, etc.

SAM file: Example of Alignment lines

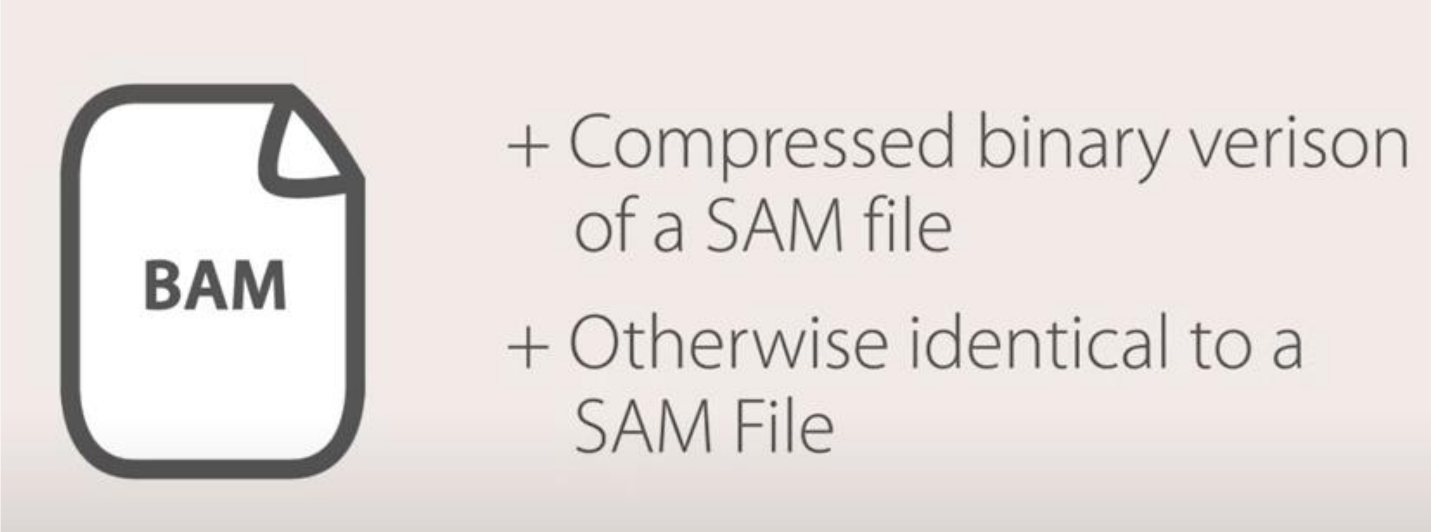
http://samtools.github.io/hts-specs/SAMv1.pdf



Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	[!-?A-~]{1,254}	Query template NAME
2	FLAG	Int	[0,2^{16}-1]	bitwise FLAG
3	RNAME	String	*|[!-()+-<>-~][!-~]*	Reference sequence NAME
4	POS	Int	[0,2^{31}-1]	1-based leftmost mapping POSition
5	MAPQ	Int	[0,2^{8}-1]	MAPping Quality
6	CIGAR	String	*|([0-9]+[MIDNSHPX=])+	CIGAR string
7	RNEXT	String	*|=|[!-()+-<>-~][!-~]*	Ref. name of the mate/next read
8	PNEXT	Int	[0,2^{31}-1]	Position of the mate/next read
9	TLEN	Int	[-2^{31}+1,2^{31}-1]	observed Template LENgth
10	SEQ	String	*|[A-Za-z=.]+	segment SEQuence
11	QUAL	String	[!-~]+	ASCII of Phred-scaled base QUALity+33

The **11 mandatory fields** of the alignment section include information on mapping quality, fragment position, quality control, sequence, etc.

Sequence Alignment: SAM >>> BAM files



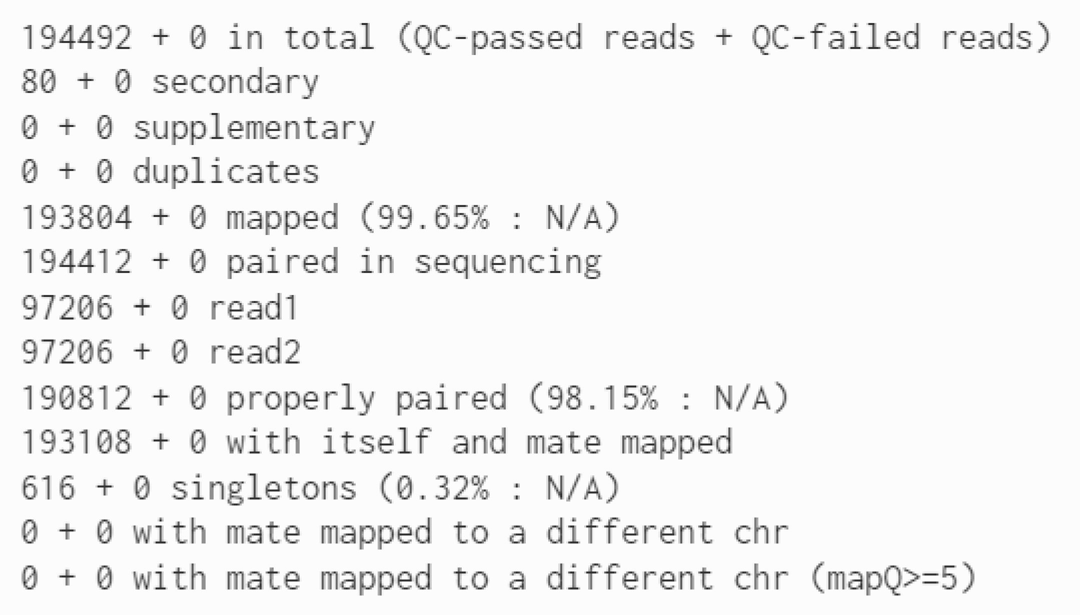
- + Compressed binary verison of a SAM file
- + Otherwise identical to a SAM File

The SAM format can be compressed to take less space in the Binary Alignment Map (BAM) format.

Alignment Metrics

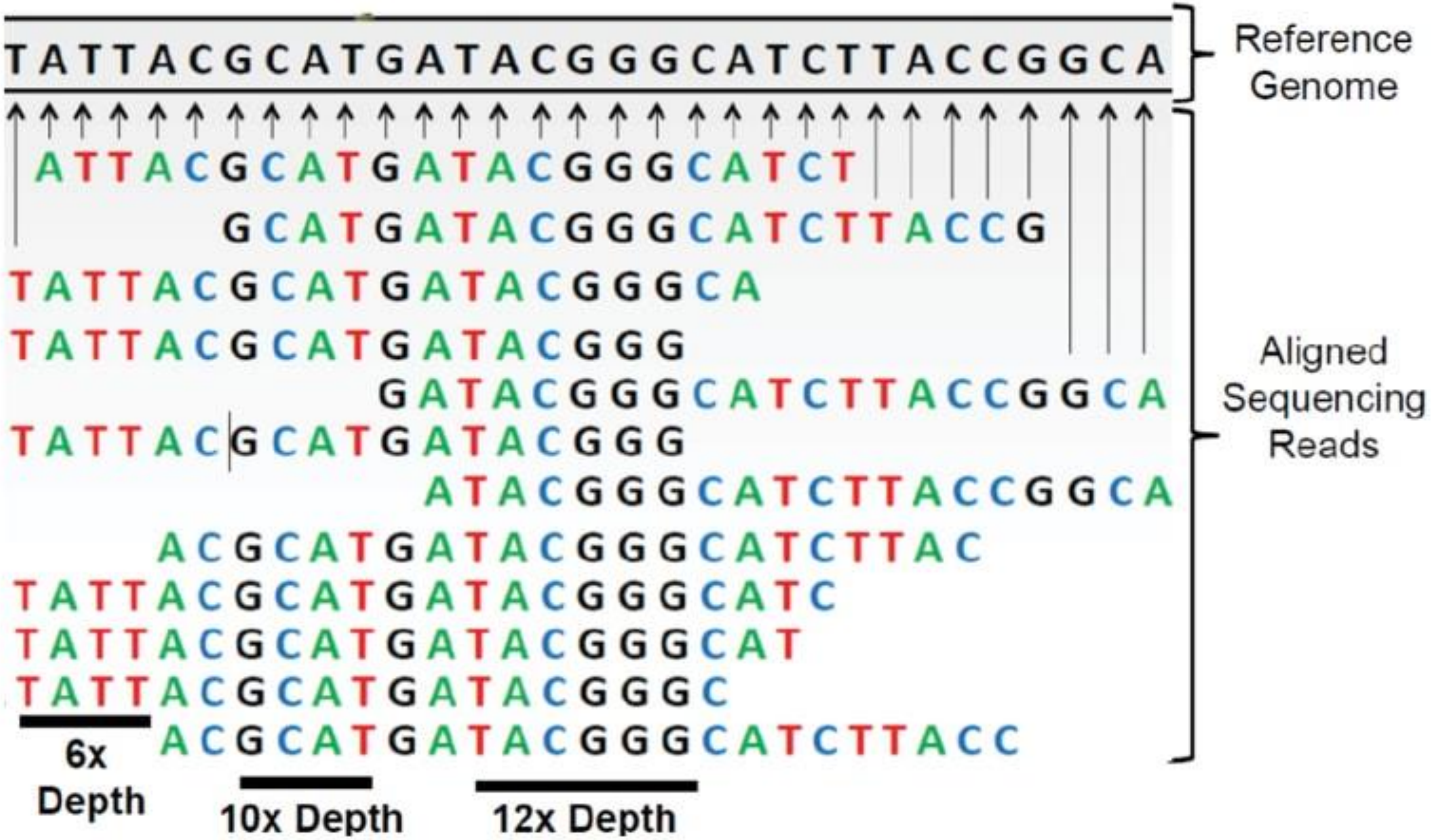
Let’s compute some statistics to see how well our reads aligned to the reference genome.
Use samtools flagstat for this.

Output:

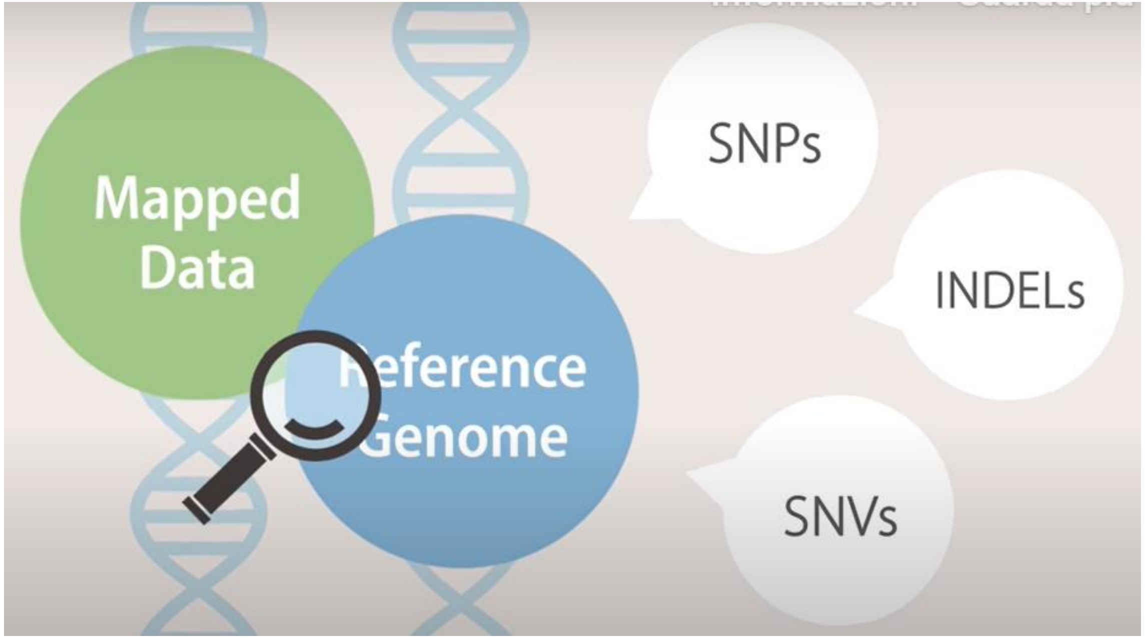


```
194492 + 0 in total (QC-passed reads + QC-failed reads)
80 + 0 secondary
0 + 0 supplementary
0 + 0 duplicates
193804 + 0 mapped (99.65% : N/A)
194412 + 0 paired in sequencing
97206 + 0 read1
97206 + 0 read2
190812 + 0 properly paired (98.15% : N/A)
193108 + 0 with itself and mate mapped
616 + 0 singletons (0.32% : N/A)
0 + 0 with mate mapped to a different chr
0 + 0 with mate mapped to a different chr (mapQ>=5)
```

Sequence Alignment



Variant calling



After alignment to a reference genome, the next step is variant calling where a program examine your mapped data and the reference side by side to determine the existence of SNPs, de novo SNVs, and INDELs.

Variant calling

- **SAMtools mpileup** and the **Genome Analyid Tollkit (GATK)** are two major variant calling programs available that use Bayesian Algorithms to compare your aligned sequence against the reference

SAMtools mpileup & GATK

Two major variant calling programs

Compares sequences using Bayesian algorithms

VCF

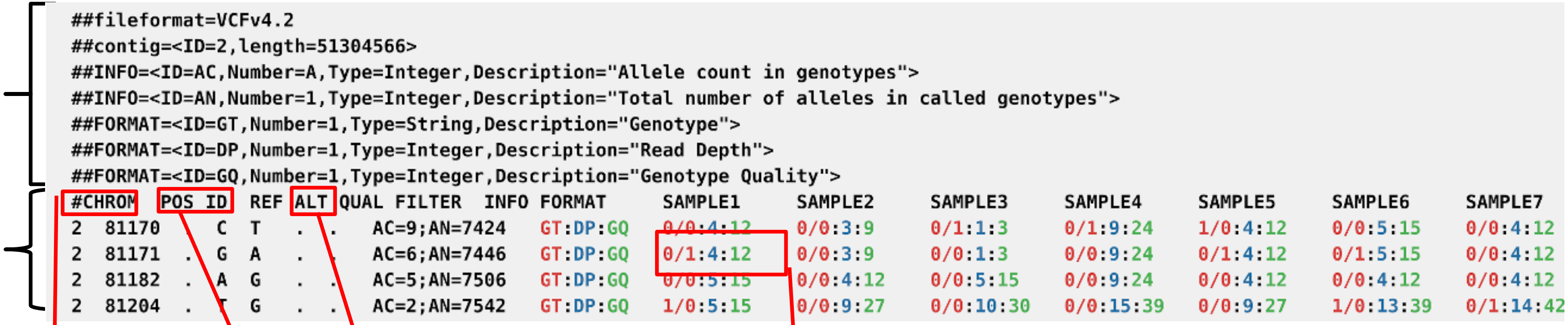
Variant Calling Format (VCF) files

- The Variant Call Format is a text file which stores sequence variants, each variant occupies a single row
- It contains meta-information lines, a header line, and then data lines each containing information about a position in the genome.
- There is an option whether to contain genotype information on samples for each position or not
- In this format, header lines start with “#”, and the body containing sequence information has 8 mandatory columns separated by tabs.

Variant Calling Format (VCF) files

VCF header

VCF body



```
##fileformat=VCFv4.2
##contig=<ID=2,length=51304566>
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes">
##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles in called genotypes">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
#CHROM  POS ID  REF ALT QUAL FILTER  INFO FORMAT     SAMPLE1     SAMPLE2     SAMPLE3     SAMPLE4     SAMPLE5     SAMPLE6     SAMPLE7
2  81170  .  C  T  .  .  AC=9;AN=7424  GT:DP:GQ  0/0:4:12    0/0:3:9     0/1:1:3     0/1:9:24    1/0:4:12    0/0:5:15    0/0:4:12
2  81171  .  G  A  .  .  AC=6;AN=7446  GT:DP:GQ  0/1:4:12    0/0:3:9     0/0:1:3     0/0:9:24    0/1:4:12    0/1:5:15    0/0:4:12
2  81182  .  A  G  .  .  AC=5;AN=7506  GT:DP:GQ  0/0:5:15    0/0:4:12    0/0:5:15    0/0:9:24    0/0:4:12    0/0:4:12    0/0:4:12
2  81204  .  T  G  .  .  AC=2;AN=7542  GT:DP:GQ  1/0:5:15    0/0:9:27    0/0:10:30   0/0:15:39   0/0:9:27    1/0:13:39   0/1:14:42
```

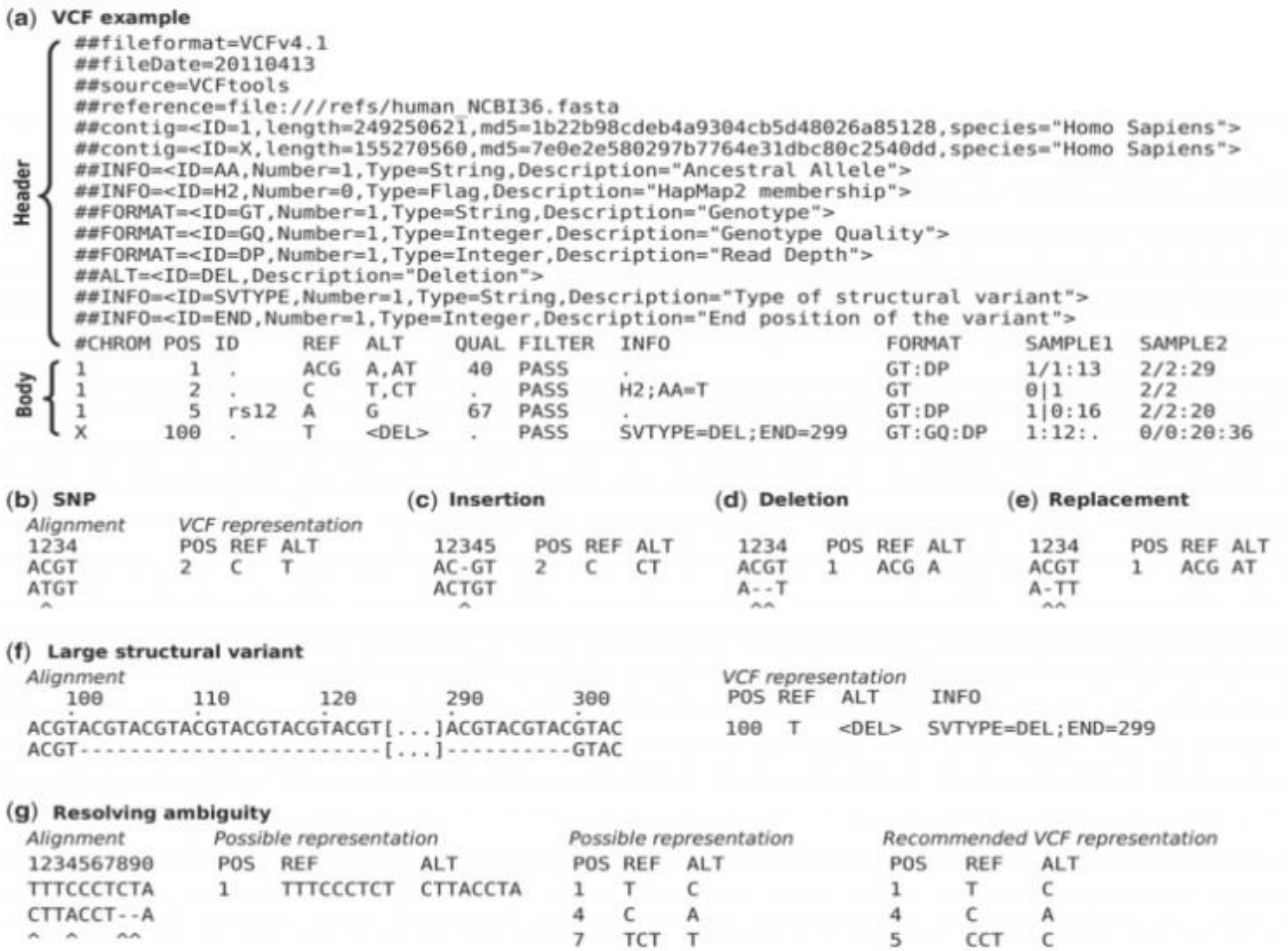
alternative base (different from the reference base)

chr number based on genome reference (*eg. Human* GRCh37 or GRCh38*)*

Reference base

GT: genotype
DP: read depth
GQ: genotype quality

VCF format



(a) **VCF example**

```
       ##fileformat=VCFv4.1
       ##fileDate=20110413
       ##source=VCFtools
       ##reference=file:///refs/human_NCBI36.fasta
       ##contig=<ID=1,length=249250621,md5=1b22b98cdeb4a9304cb5d48026a85128,species="Homo Sapiens">
       ##contig=<ID=X,length=155270560,md5=7e0e2e580297b7764e31dbc80c2540dd,species="Homo Sapiens">
       ##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
Header ##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
       ##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
       ##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
       ##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
       ##ALT=<ID=DEL,Description="Deletion">
       ##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Type of structural variant">
       ##INFO=<ID=END,Number=1,Type=Integer,Description="End position of the variant">
       #CHROM POS ID    REF ALT    QUAL FILTER INFO                 FORMAT     SAMPLE1   SAMPLE2
       1        1 .     ACG A,AT     40 PASS   .                    GT:DP      1/1:13    2/2:29
Body   1        2 .     C   T,CT      . PASS   H2;AA=T              GT         0|1       2/2
       1        5 rs12  A   G        67 PASS   .                    GT:DP      1|0:16    2/2:20
       X      100 .     T   <DEL>     . PASS   SVTYPE=DEL;END=299   GT:GQ:DP   1:12:.    0/0:20:36
```

(b) **SNP**

```
Alignment   VCF representation
1234        POS REF ALT
ACGT        2   C   T
ATGT
 ^
```

(c) **Insertion**

```
12345   POS REF ALT
AC-GT   2   C   CT
ACTGT
  ^
```

(d) **Deletion**

```
1234    POS REF ALT
ACGT    1   ACG A
A--T
 ^^
```

(e) **Replacement**

```
1234    POS REF ALT
ACGT    1   ACG AT
A-TT
 ^^
```

(f) **Large structural variant**

```
Alignment                                                        VCF representation
   100      110      120      290      300                       POS REF ALT   INFO
ACGTACGTACGTACGTACGTACGTACGT[...]ACGTACGTACGTAC                  100 T   <DEL> SVTYPE=DEL;END=299
ACGT------------------------[...]----------GTAC
```

(g) **Resolving ambiguity**

```
Alignment    Possible representation      Possible representation   Recommended VCF representation
1234567890   POS REF        ALT           POS REF ALT               POS REF ALT
TTTCCCTCTA   1   TTTCCCTCT  CTTACCTA      1   T   C                 1   T   C
CTTACCT--A                                4   C   A                 4   C   A
^  ^   ^^                                 7   TCT T                 5   CCT C
```

VCF files

- **What software use VCF?**
- Output of SNP detection tools such as GATK and Samtools
- Input for SNP feature detection like SNPeff
- VCF Tools
- Also the required format for dbSNP

- **How are these files generated?**
- SNP callers generate these files as output.
- Haplotyping software also report in this format.
- Any database holding variant information will generally have this format available for download.

Visualization of Data

- Integrative Genome Viewer (IGV)
- UCSC Genome Browser