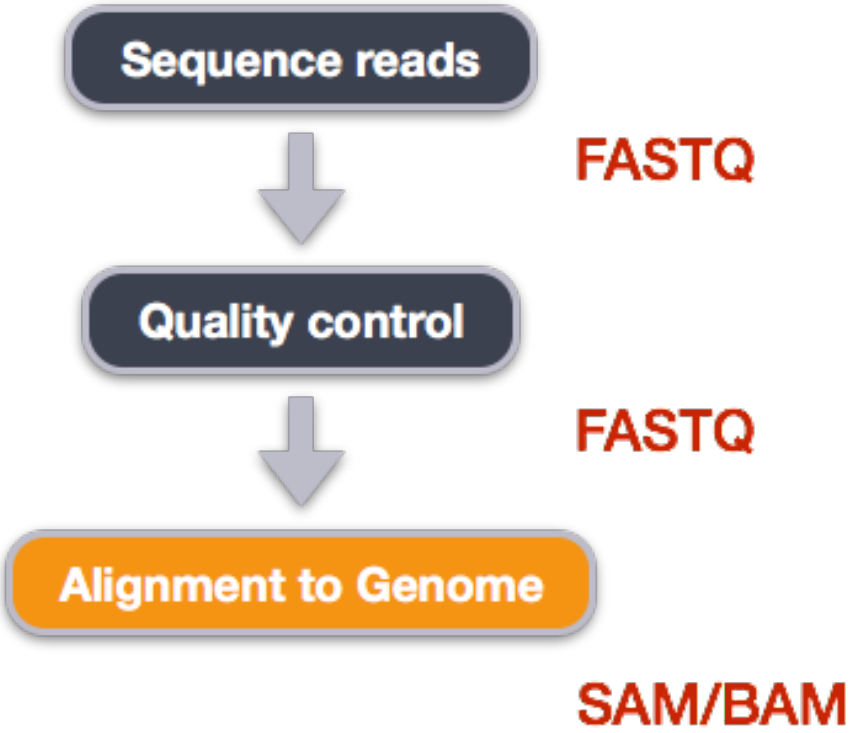
FASTQC


Sequence reads
FASTQ
Quality control
FASTQ
Alignment to Genome
SAM/BAM

Overview for today's lecture

- Job Submission recap
  - SLURM directives
  - .out files

- FASTQC

Why submit a job?

- It will be annoying to run each sample, one by one:

```
htseq-count -f bam -s yes -i gene_name -m union sample1.bam annotation.gtf > sample1.count.txt
```

WAIT FOR IT TO RUN

```
htseq-count -f bam -s yes -i gene_name -m union sample2.bam annotation.gtf > sample2.count.txt
```

WAIT FOR IT TO RUN

- Also, some programs…. take a very long time to run!

Submitting a job

- Submitting a job to an HPC machine (such as Bluemoon) is done using the batch system.

As of March 2022, the VACC provides three Clusters:

- BlackDiamond
- Bluemoon
- DeepGreen

We will primarily use the **Bluemoon** cluster for any downstream analysis.

Submitting a job

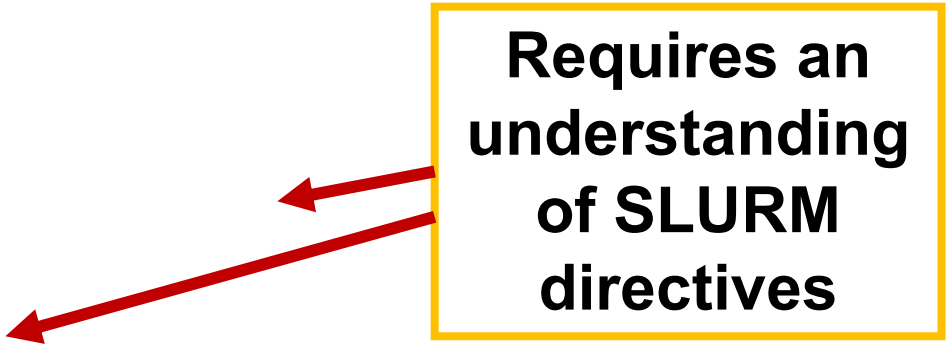
- The batch system allows users to submit jobs requesting the resources (nodes, processors, memory, GPUs) that they need.
- The jobs are queued and then run as resources become available.
- All jobs (scripts with file extension .sh) are submitted to a batch system called SLURM

BASIC STEPS:

1. Log in to VACC
2. Write job script
3. Submit Job
4. Monitor job and wait for it to run
5. Retrieve your output

BASIC STEPS:

1. Log in to VACC
2. **Write job script**
3. **Submit Job**
4. Monitor job and wait for it to run!
5. Retrieve your output



Quick look inside sra_fqdump.sh

```
nano sra_fqdump.sh
```

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=fastq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out

#while there are lines in the list of SRRs file
while read p
do
#call the bash script that does the fastq dump, passing it the SRR number next $
sbatch inner_script.sh $p
done <list_of_SRRs.txt
```

1.
SLURM DIRECTIVES

2.
CODE

What are SLURM Directives?

- At the top of the job script will always be several lines that start with **#SBATCH**.
- The SLURM directives provide the job setup information used by Slurm.
- This information is then followed by the commands to be executed in the script.

Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```

Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoor
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```

shebang = used to tell the linux OS which interpreter to use

Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```

partition = default is bluemoon if not specified

Other partitions are available. This is important to know as some jobs will take longer to run!

Other Partitions:

Partition	Intended Use	Max Runtime
bluemoon	General computing – default partition	30 hours
short	General computing with short runtime	3 hours
week	General computing with longer runtime	7 days
bigmem	Large memory requirements computing	30 hours
bigmemwk	Large memory requirements with longer runtime	7 days

You can check partition usage using the following command:

```
sinfo -p partition_name
sinfo -p bluemoon
```

Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```

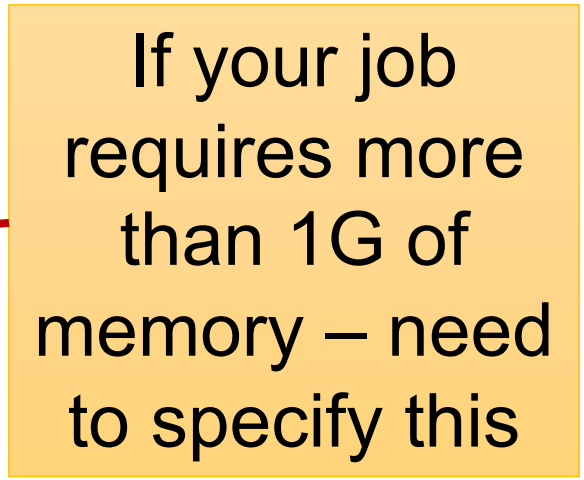
Node: A “node” is a server in the cluster. Each node has is configured with a certain number of cores (CPUs).

Task: A “task” is a process sent to a core. By default, 1 core is assigned per 1 task

Recommend that you begin with 1 node and 2 processes

Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```



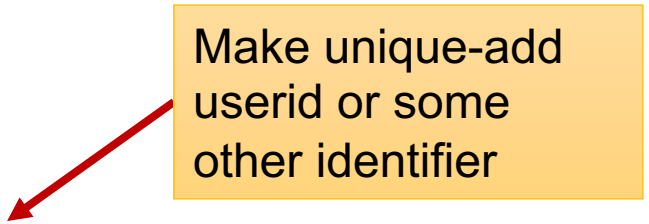
Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```

Walltime is the maximum amount of time your job will run.

Common slurm directives

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mem=50G
#SBATCH --time=30:00:00
#SBATCH --job-name=RNAseq
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```



To run script use sbatch command:

```
sbatch sra_fqdump.sh
```

When you submit your job, Slurm will respond with the job ID. For example, where the job ID Slurm assigns is “123456,” Slurm will respond:

```
Submitted batch job 123456
```

Other important commands:

Command	What It Does
sbatch	Submits a job, e.g., sbatch myjob
scontrol show job	Detailed information about a particular job, e.g., scontrol show job 123456
squeue	Checks status of all jobs in scheduling queue
squeue -u	Checks status of all jobs belonging to the named user, e.g., squeue -u usr1234
squeue –start -j	Estimates earliest start time of a particular job, e.g., squeue –start -j 123456
squeue –start -u	Estimates earliest start time of all jobs belonging to the named user, e.g., squeue –start -u usr1234
scancel -u	Deletes/cancels all jobs belonging to the named user, e.g., scancel -u usr1234
scancel	Deletes/cancels a particular job, e.g., scancel 123456

Fastq-dump --gunzip SRR#1 -> .out file #1

Fastq-dump --gunzip SRR#2 -> .out file #2

Fastq-dump --gunzip SRR#3 -> .out file #3

Recording the output of programs

- Three data streams exist for all Linux programs
  - STDIN (Standard Input - a way to send data into the program)
  - STDOUT (Standard Output - a way to send expected data out of the program)
  - STDERR (Standard Error - a way to send errors or warnings out of the program)

- By default STDOUT and STDERR are connected to your shell, so when you see text coming from a program, it's coming from these streams.

These will be redirected to .out files

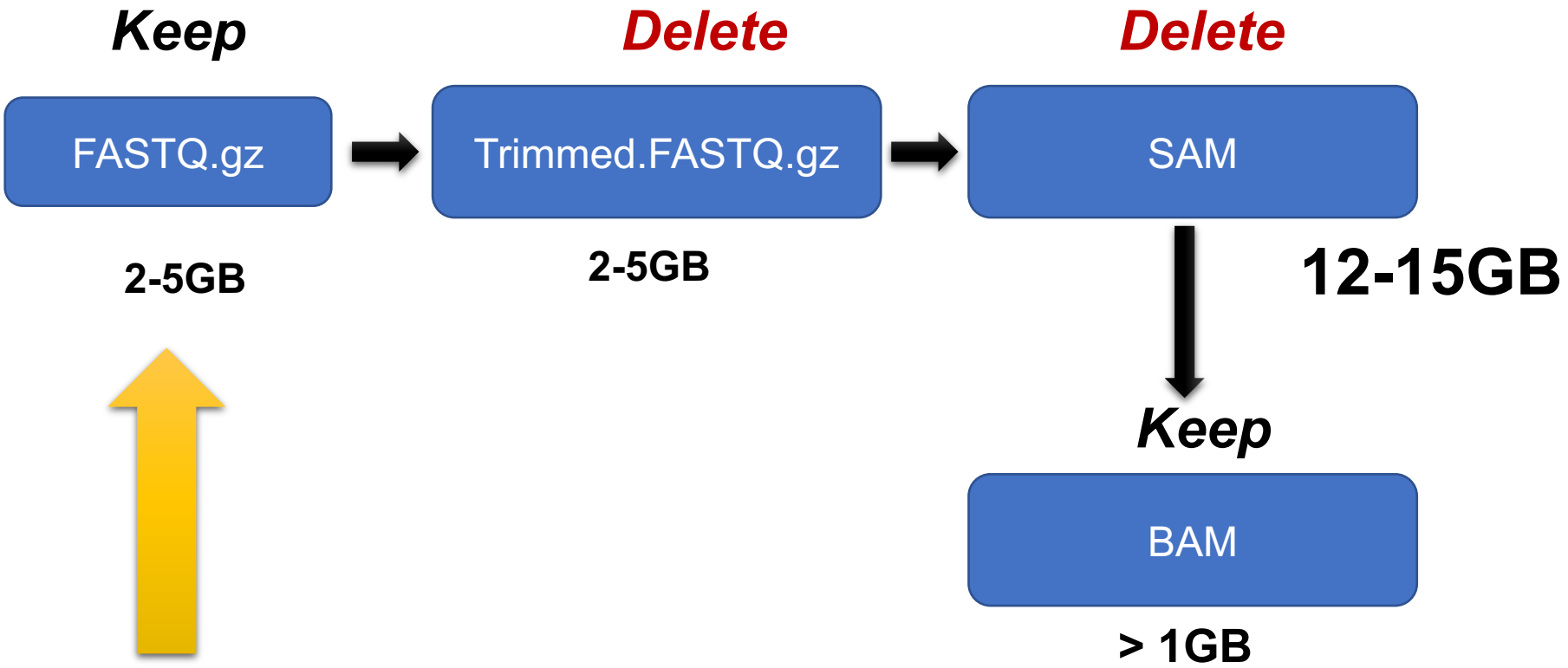
```
[pdrodrig@vacc-user1 GSE164713_Tcf1]$ ls
fastq_6008331.out  fastq_6426350.out     SRR13422709.fastq.gz
fastq_6366635.out  fastq_6426351.out     SRR13422710.fastq.gz
fastq_6366636.out  inner_script.sh       SRR13422711.fastq.gz
fastq_6366637.out  list_of_SRRs.txt      SRR13422712.fastq.gz
fastq_6366638.out  list.txt              SRR13422713.fastq.gz
fastq_6366639.out  sra_download.sh       SRR13423162.fastq.gz
fastq_6366640.out  sra_fqdump.sh         SRR13423163.fastq.gz
fastq_6366641.out  SRR13416485.fastq.gz  SRR13423164.fastq.gz
fastq_6426340.out  SRR13416486.fastq.gz  SRR13423165.fastq.gz
fastq_6426341.out  SRR13422702.fastq.gz  SRR13423166.fastq.gz
fastq_6426342.out  SRR13422703.fastq.gz  SRR13423167.fastq.gz
fastq_6426343.out  SRR13422704.fastq.gz  SRR17379677.fastq.gz
fastq_6426344.out  SRR13422705.fastq.gz  SRR17379678.fastq.gz
fastq_6426347.out  SRR13422706.fastq.gz  SRR17379679.fastq.gz
fastq_6426348.out  SRR13422707.fastq.gz  SRR17379680.fastq.gz
fastq_6426349.out  SRR13422708.fastq.gz
```

Any errors will be recorded in these .out files

```
hg38_bam/917_EC90_2hr_R1B_S8_unmapped_hg38_sorted.bam
Traceback (most recent call last):
  File "/users/p/d/pdrodrig/anaconda3/bin/htseq-count", line 5, in <module>
    HTSeq.scripts.count.main()
  File "/users/p/d/pdrodrig/anaconda3/lib/python3.8/site-packages/HTSeq/scripts/count.py",$
    args = _parse_sanitize_cmdline_arguments()
  File "/users/p/d/pdrodrig/anaconda3/lib/python3.8/site-packages/HTSeq/scripts/count.py",$
    _check_sam_files(args.samfilenames)
  File "/users/p/d/pdrodrig/anaconda3/lib/python3.8/site-packages/HTSeq/scripts/count.py",$
    with pysam.AlignmentFile(sam_filename, "r") as sf:
  File "pysam/libcalignmentfile.pyx", line 742, in pysam.libcalignmentfile.AlignmentFile._$
  File "pysam/libcalignmentfile.pyx", line 947, in pysam.libcalignmentfile.AlignmentFile._$
ValueError: file does not contain alignment data
hg38_bam/917_EC90_2hr_R2A_S22_unmapped_hg38_sorted.bam
Traceback (most recent call last):
  File "/users/p/d/pdrodrig/anaconda3/bin/htseq-count", line 5, in <module>
    HTSeq.scripts.count.main()
  File "/users/p/d/pdrodrig/anaconda3/lib/python3.8/site-packages/HTSeq/scripts/count.py",$
    args = _parse_sanitize_cmdline_arguments()
  File "/users/p/d/pdrodrig/anaconda3/lib/python3.8/site-packages/HTSeq/scripts/count.py",$
    _check_sam_files(args.samfilenames)
  File "/users/p/d/pdrodrig/anaconda3/lib/python3.8/site-packages/HTSeq/scripts/count.py",$
    with pysam.AlignmentFile(sam_filename, "r") as sf:
```

Data management planning

- What types of intermediate data files will I generate?
- Consider the **datatype and file sizes**. How much total raw data do you have?
- Consider the **type of analyses** you plan on performing.
- What is the best way to organize my **directory structure**?

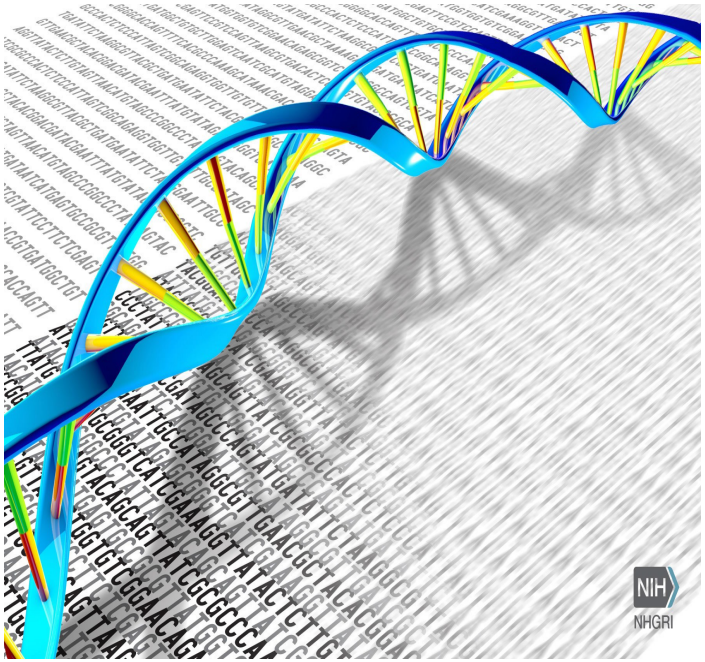


You will need to collect information about the data you are processing

Data types: Raw data

- For NGS experiments, these are the **FASTQ** files you obtain from your sequencing facility or GEO

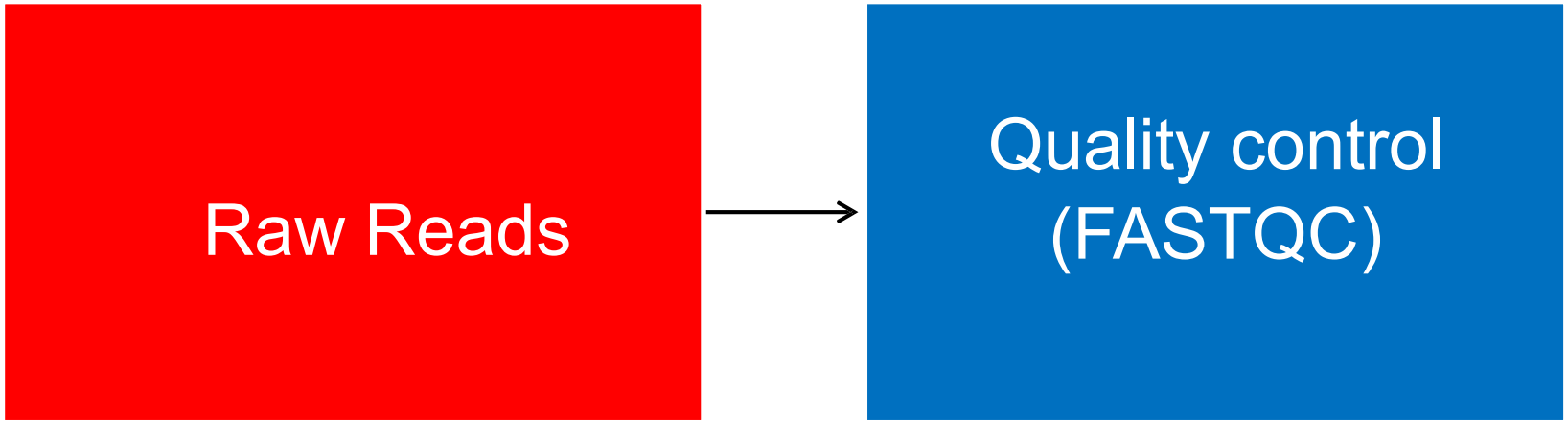
- This data should never be directly modified (i.e. always keep a copy of this stored somewhere, untouched from its original state)



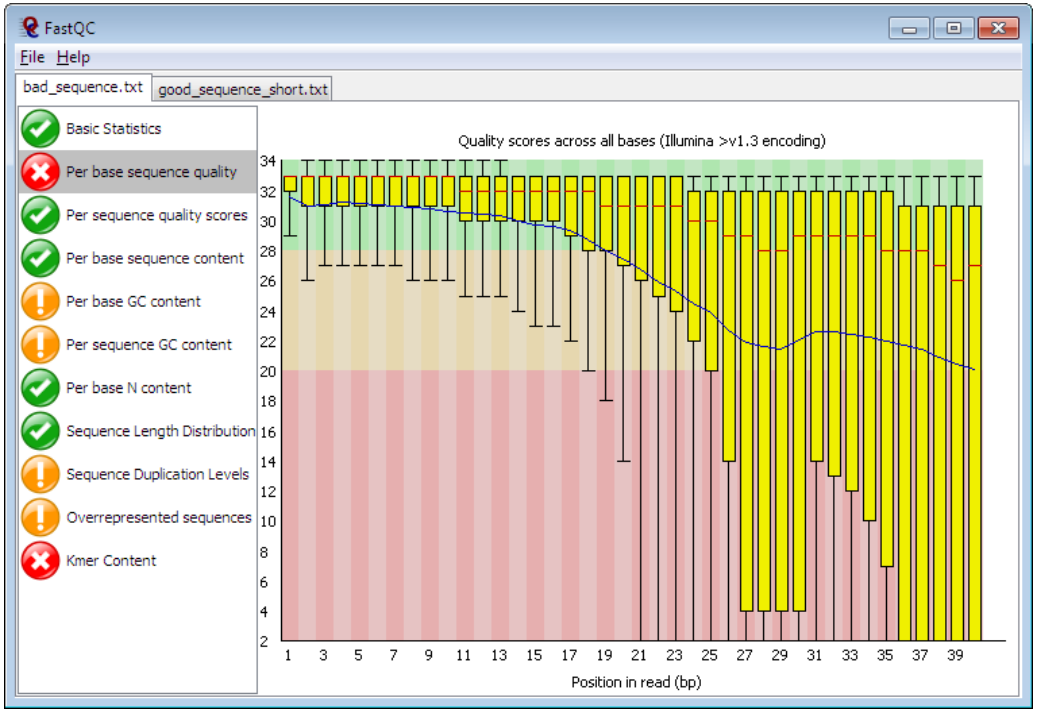
Metadata

- Metadata are data that provide information about the data you are processing
- Metadata exists to give data context
- In genomics or transcriptomics, metadata describes the sample that the DNA/RNA sequence was obtained from, the organism, the cell line, and library preparation method

Workflow



FastQC



- Reads raw fastq files as input
- Performs multiple checks
  - Pass/warn/fail
  - Compares to genomic library
- HTML Report

http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

Sequence Output Format

- FASTQ

  Line 1: Unique ID for a sequencing read

  Line 2: Sequences

  Line 3:+

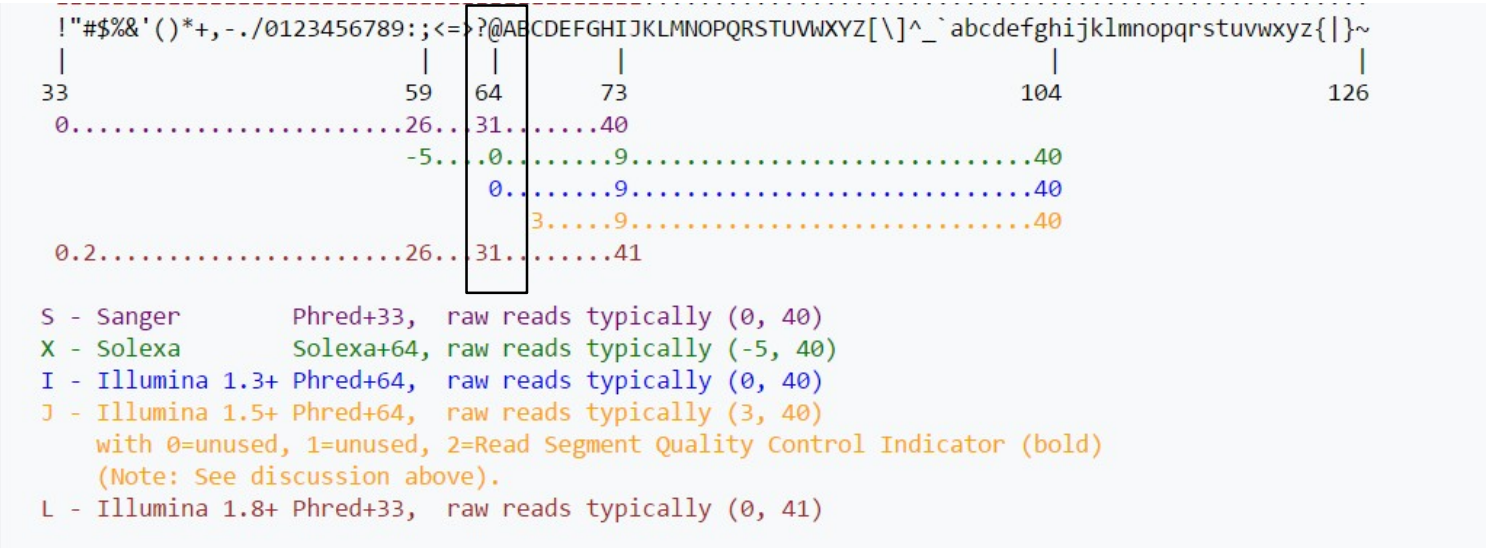
  Line 4: Base calling quality score (American Standard Code for Information Interchange *(*ASCII) value)

  Example:

```
@HISEQ:126:H14YJADXX:1:1101:1118:2101 1:N:0:ATCACG
CTCCATAGTCAGAAACTTCAGCATGACAGTACCTCATGCTGCATCAGGTGATCATGAAAAGATTACAGGCTTTCTAAAATTATCAGCAAGATATGG
+
@@?ADDDD?ADHDIIIIIIIEIIIGEFHC<?FH4C9E9BGAFIGH<DG9BD?@DGGEGHHG<DCBBCC8C>FHCGEHIGEEE>EEHEEEEC>A>;;
```

Quality Score Representation

- Quality scores are represented as ASCII characters in order to save space, so that there is one ASCII character per base.
- Converts between the ASCII value into Phred score



Base Call Qualities (Phred scores)

- For most runs, quality should be good for most reads through the whole run
- If quality deteriorates we should understand how and why
- Good (Illumina) quality is generally Phred > 30
- Concerning (Illumina) quality is Phred < 20

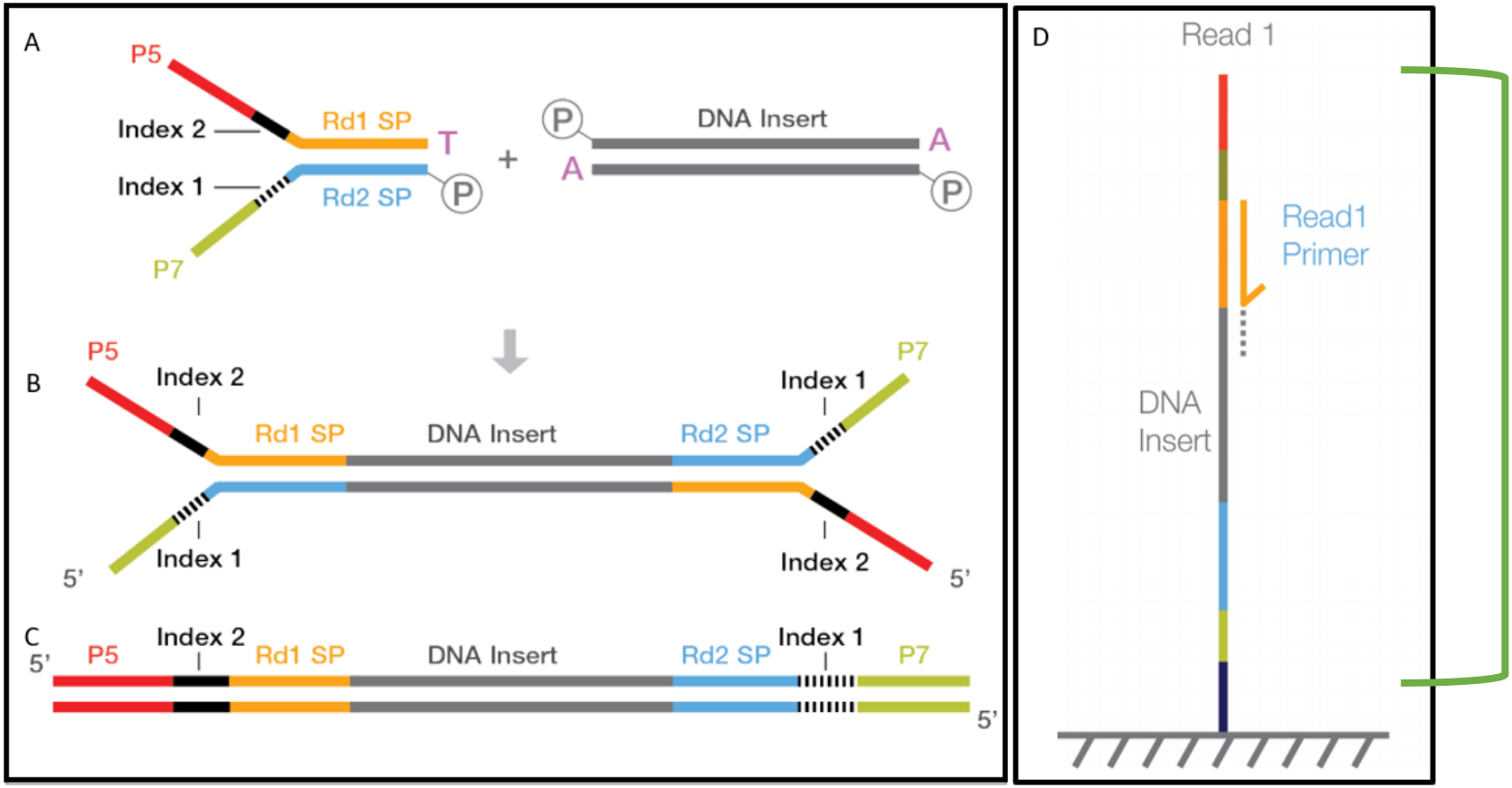
What should you look for in your libraries?

Basic Statistics

Measure	Value
Filename	Mov10_oe_1.fastq
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	39971841
Filtered Sequences	0
Sequence length	100
%GC	47

Architecture of Standard Illumina NGS library

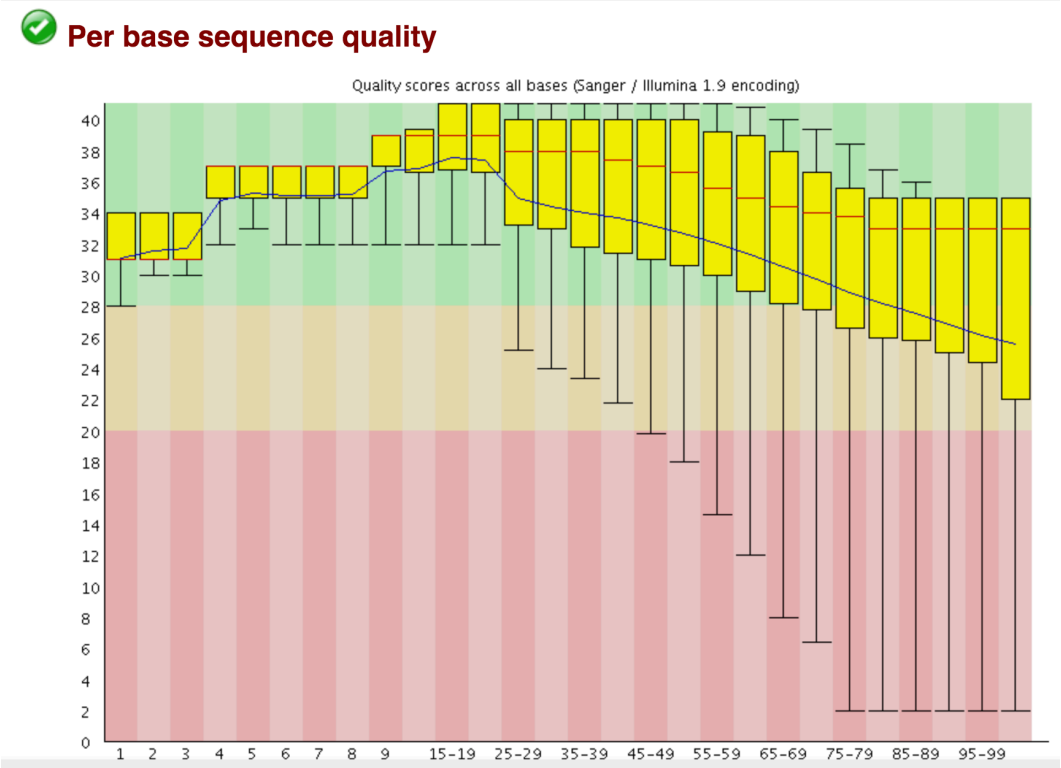
P5 and p7 sequences are required to bind the flow cell



Adaptors will serve as primer binding sites for amplification and sequencing

Indices are used to combine many samples into 1 seq. run

Phred Score



Cycles of Chemistry

For each position a BoxWhisker type plot is drawn.

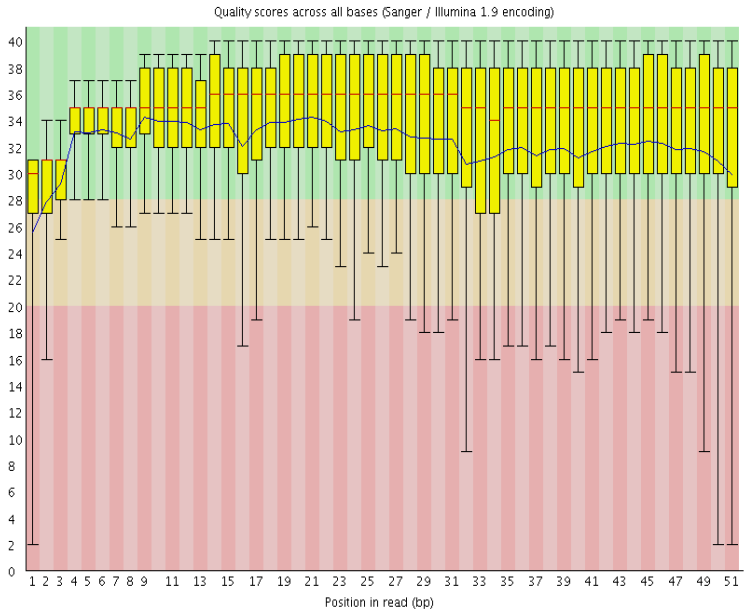
- The central red line is the median value
- The yellow box represents the inter-quartile range (25-75%)
- The upper and lower whiskers represent the 10% and 90% points
- The blue line represents the mean quality

FastQC Report

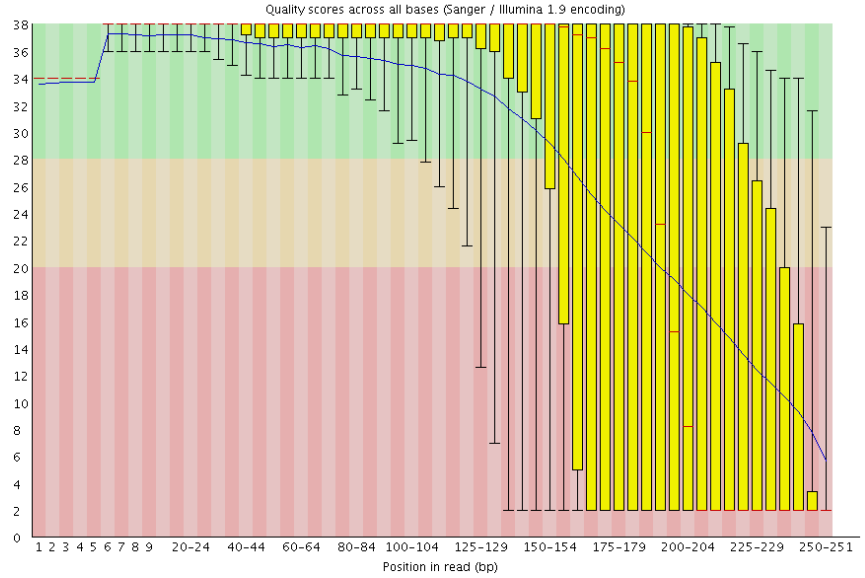
PHRED Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90 %
20	1 in 100	99 %
30	1 in 1000	99.9 %
40	1 in 10000	99.99 %
50	1 in 100000	99.999 %

Base Call Qualities – Per Cycle

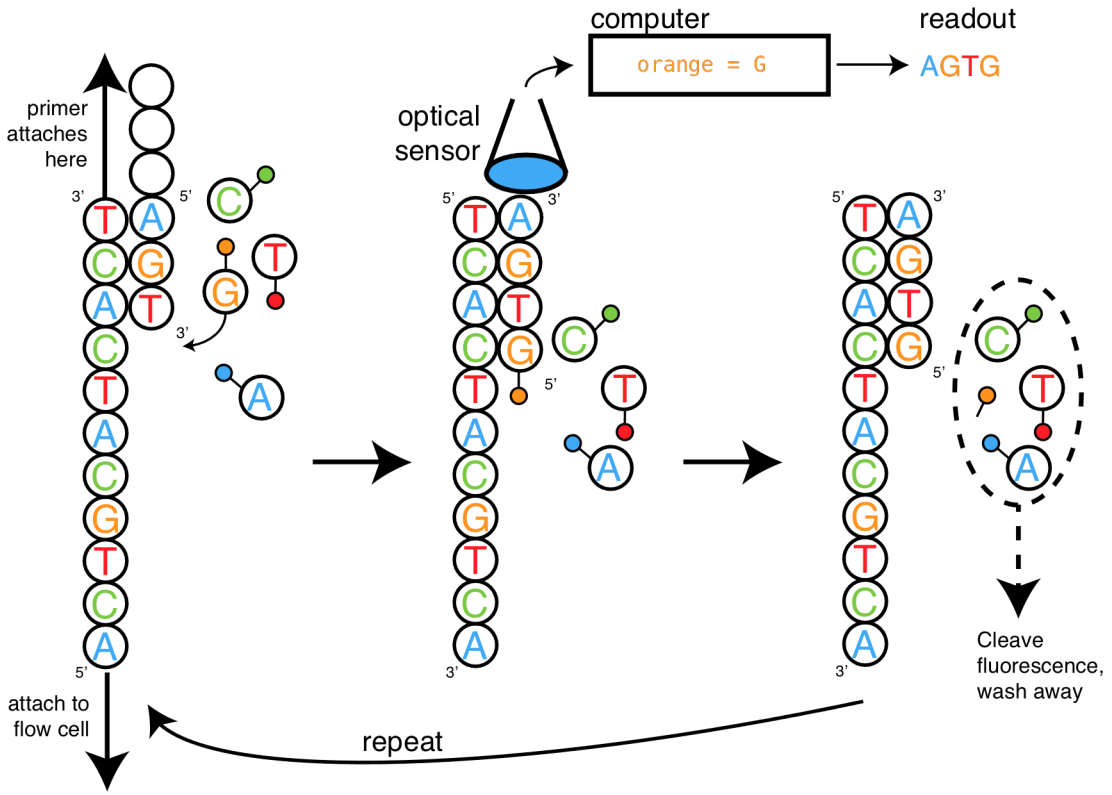
Read 1



Read 2



The quality of base call tend to degrade as the run progresses



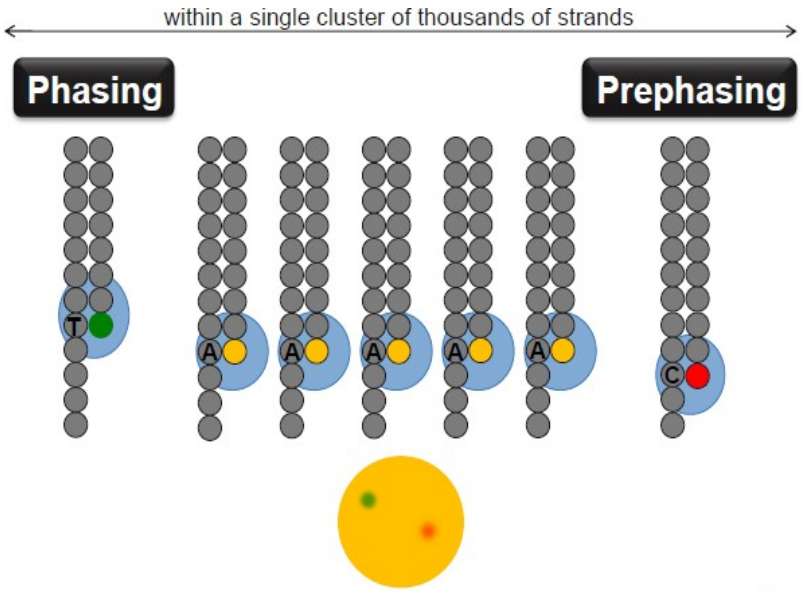
- Then sequencing begins!
  - Have fluorescent-labeled nucleotides
  - Each dNTP has a corresponding color
  - During each **cycle**, a labeled dNTP is added to the growing chain
  - An image is then taken
  - Then the fluorescent dye is cleaved to allow for the next nucleotide to be incorporated in the next **cycle**

Problem

- As the chain grows, signal quality deteriorates
- At each cycle, accumulates noise

Solution

- Sequence only (up to 300bp)
- Enables robust base-calling across the genome including in repetitive sequence regions

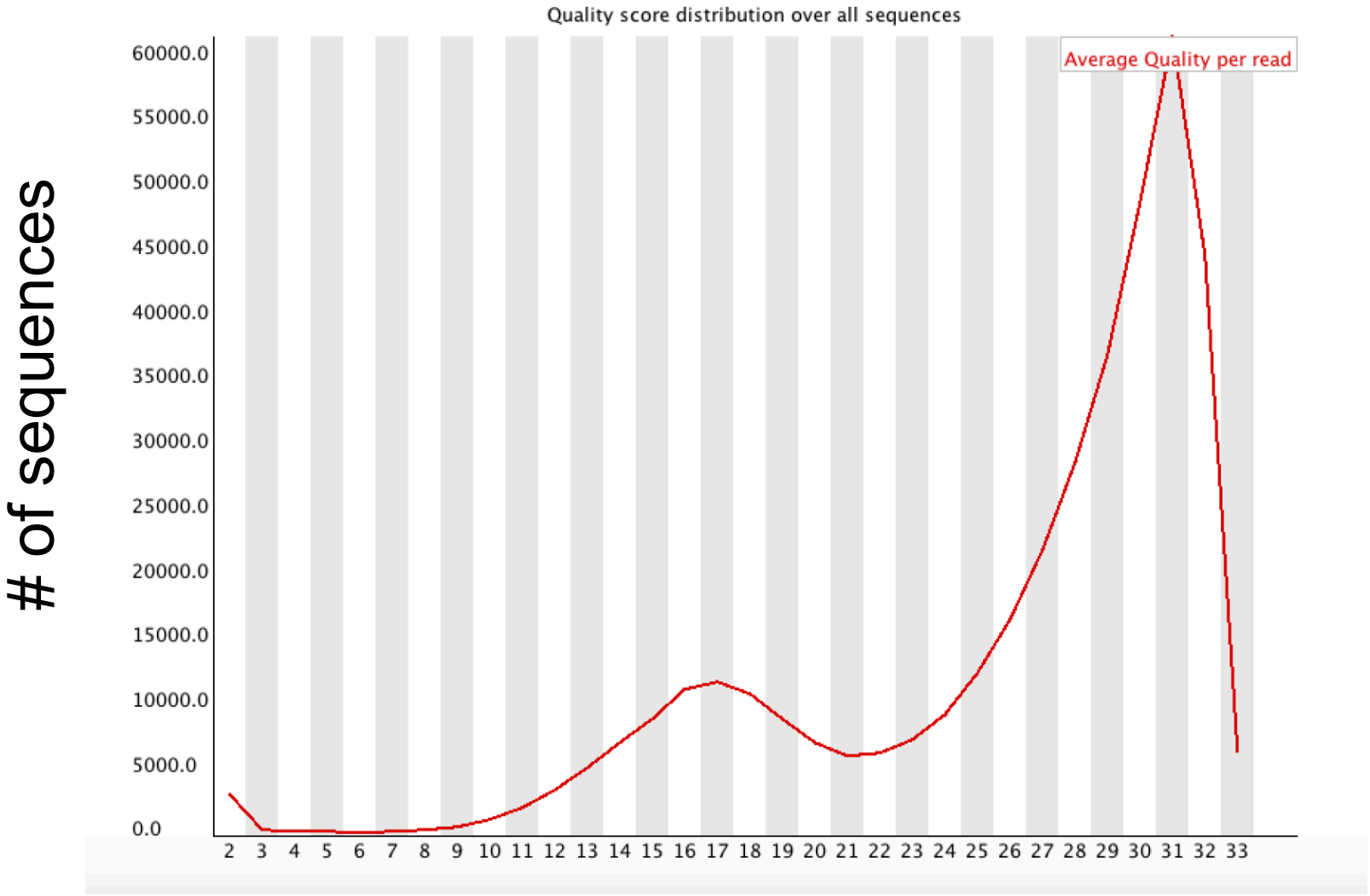


Trimming

- If the quality of the library falls to a low level then the most common remedy is to perform quality trimming where reads are truncated based on their average quality.
- Trimming must be performed on all samples - Before committing to any action, check in

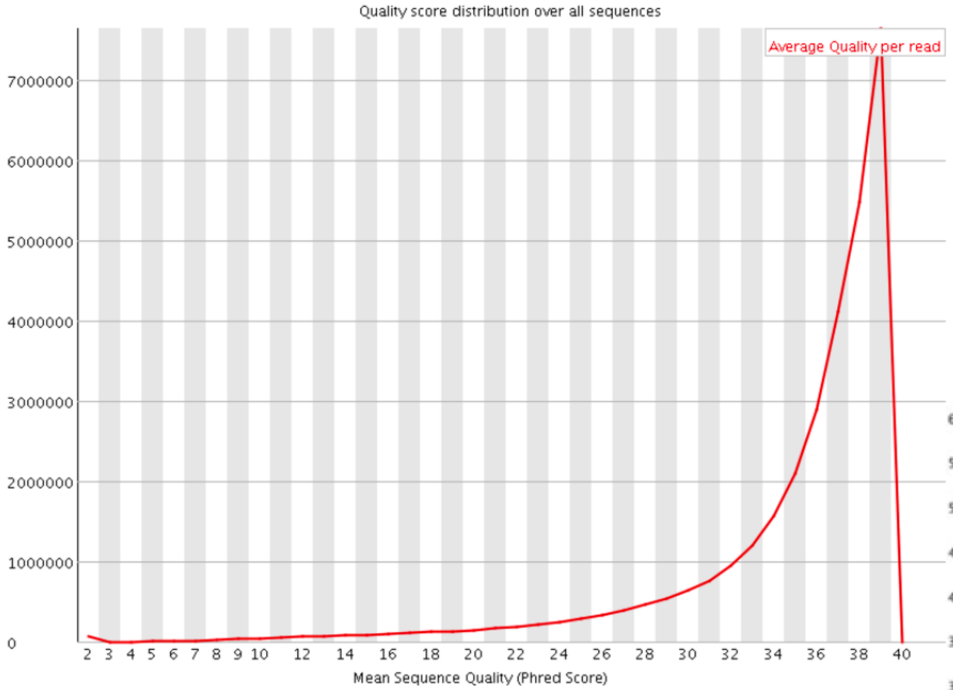
Per Sequence Quality Score

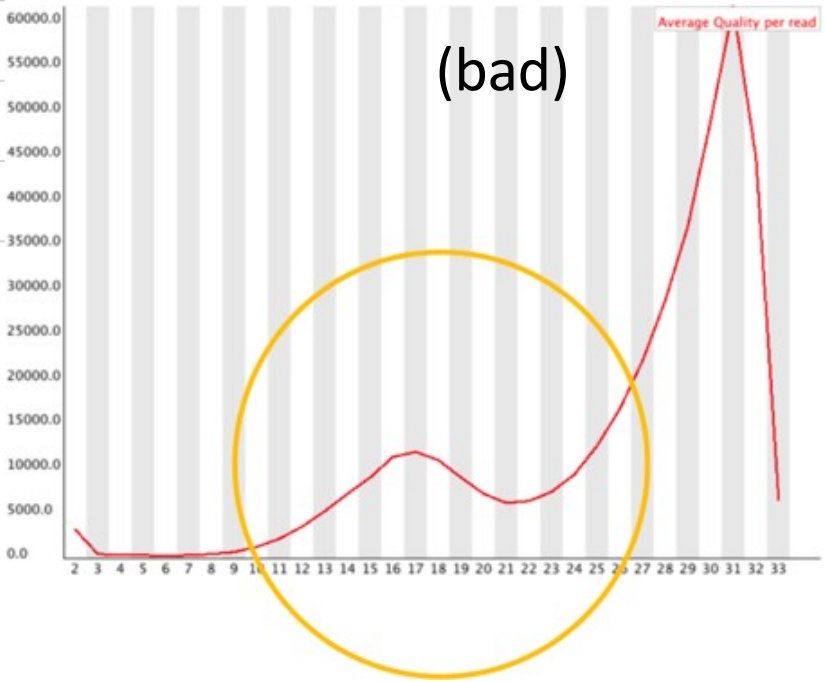
Is it a subset of the sequences that have low scores or is it the majority of them?



FastQC: per-read mean base qualities

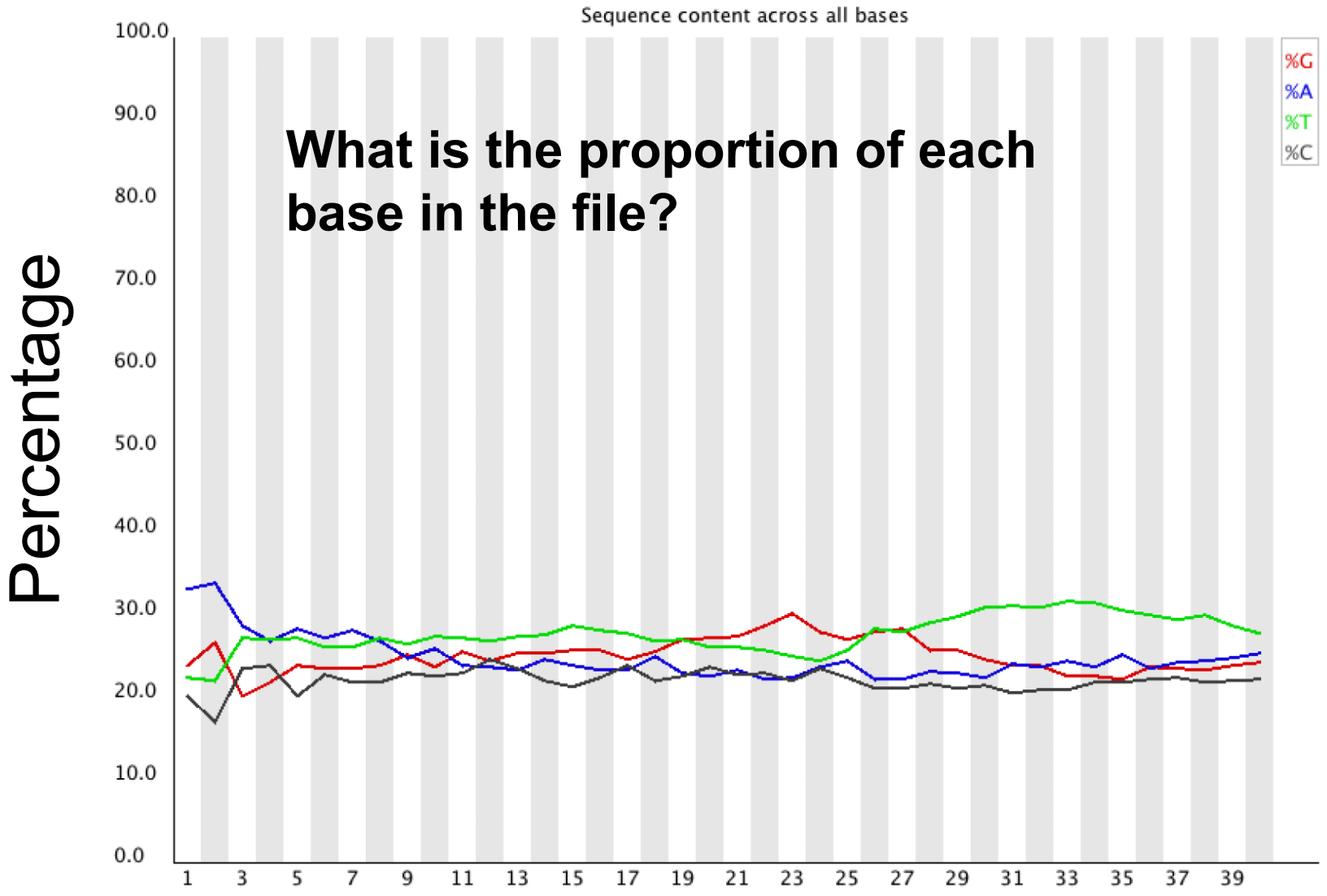
Per sequence quality scores





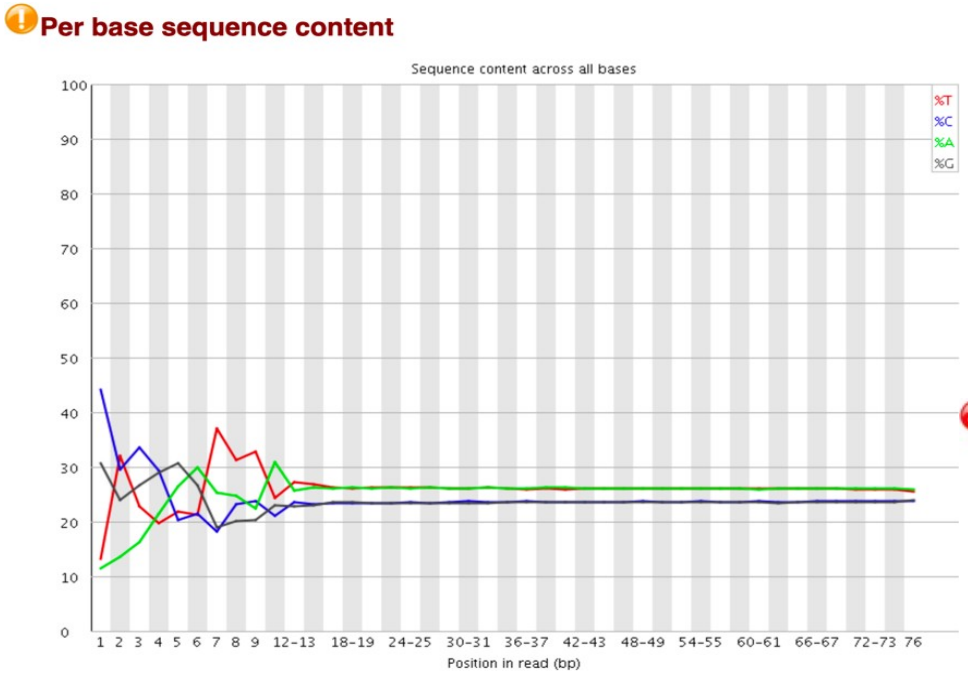
(bad)

Per Base Sequence content



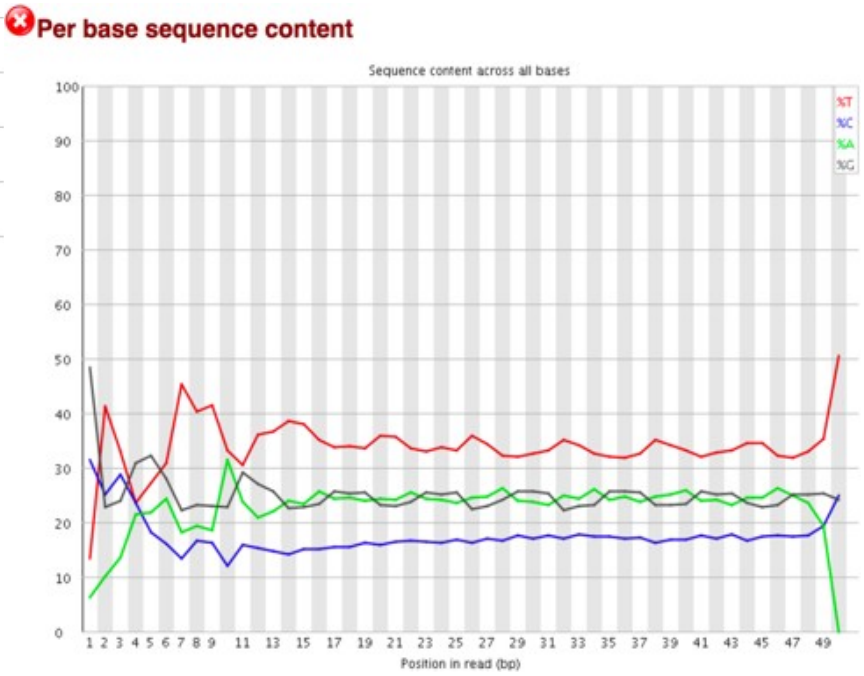
If there are 4 bases out of 100% what is the gold standard?

FastQC: %ACGT over read length

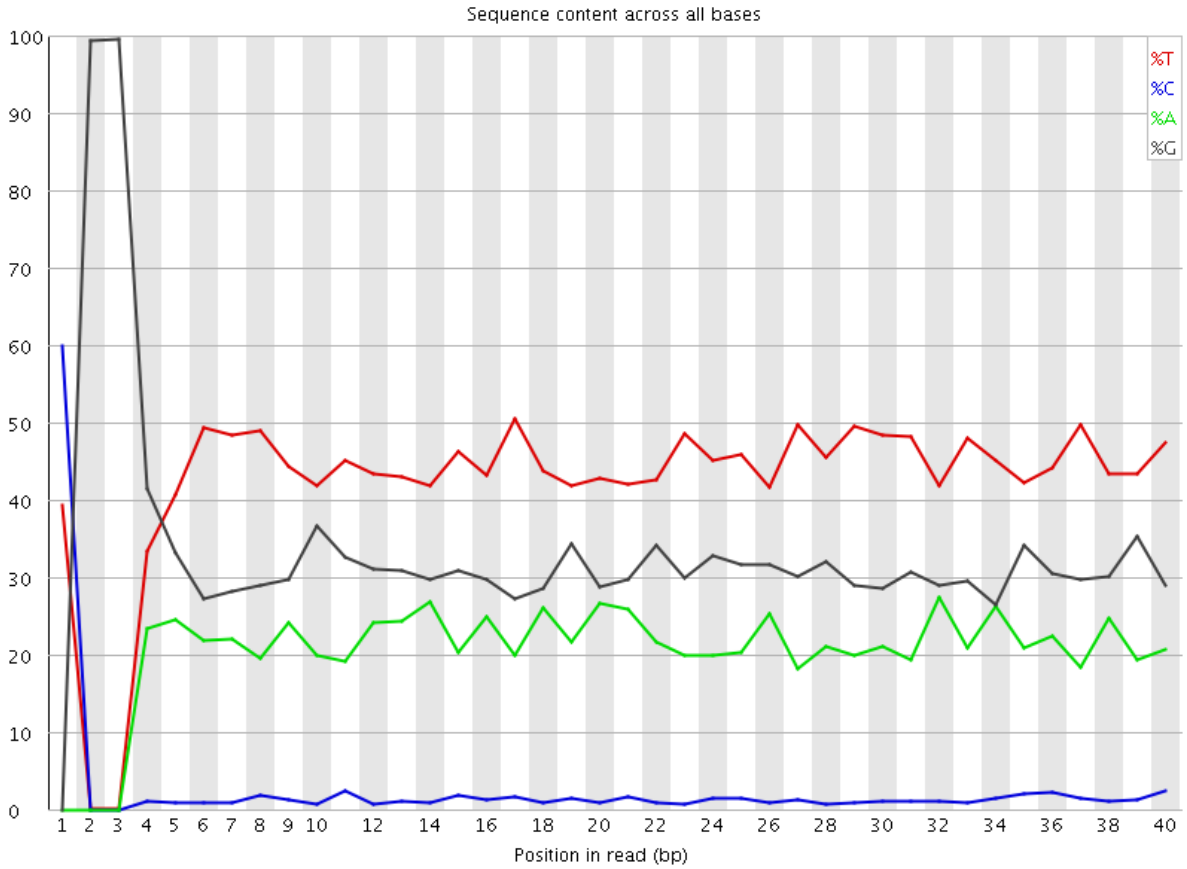


If there is a strong preference, this indicates an overrepresented sequence

(bad)

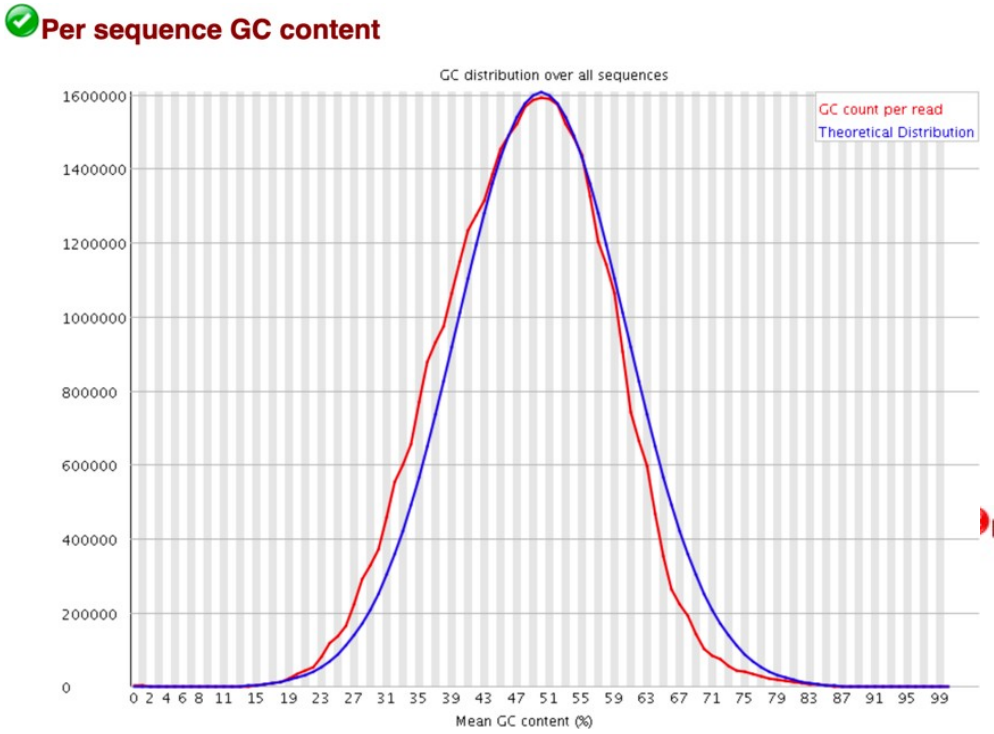


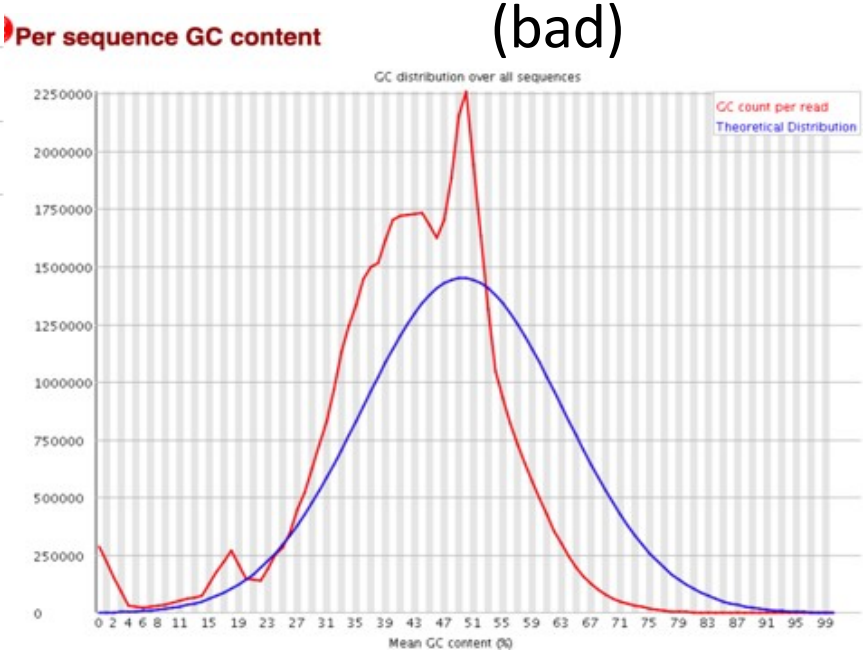
Library Base Composition



- Bisulphite treated – C is converted to T

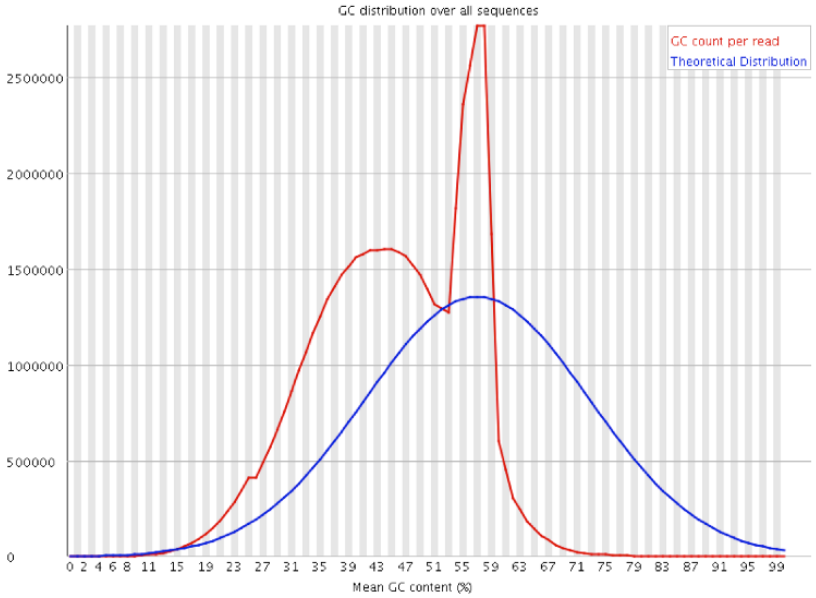
FastQC: per-read %GC



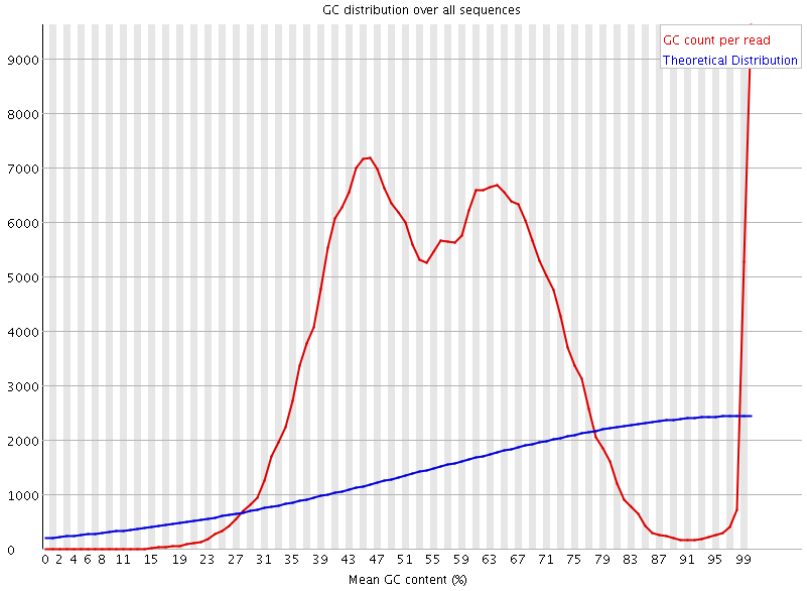


(bad)

Library GC Content

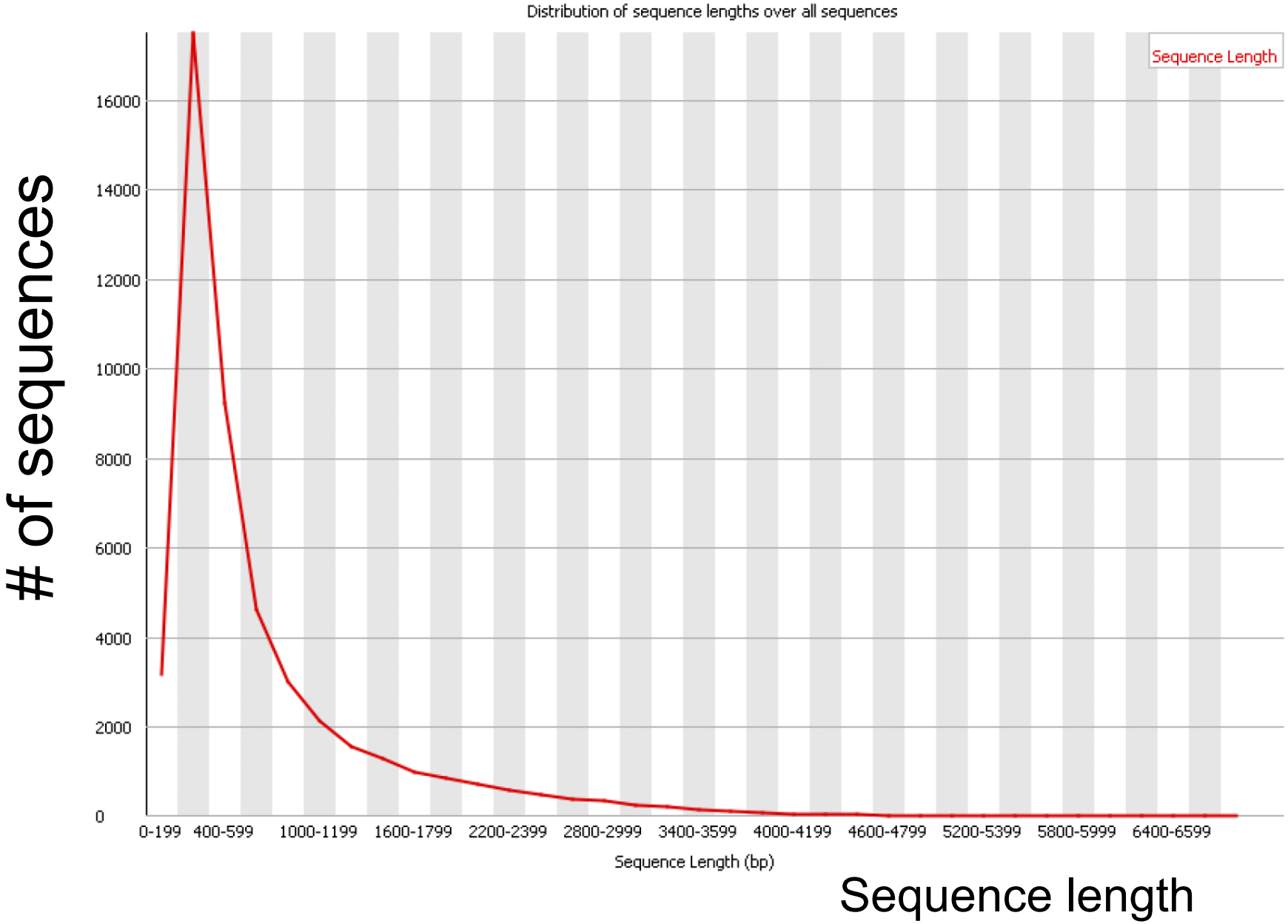


Specific Contamination

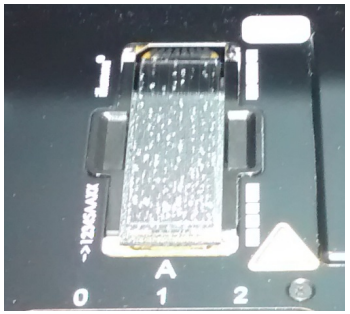


Broad Contamination

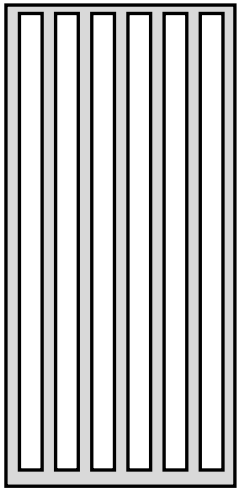
What is the average fragment length?



Positional Quality

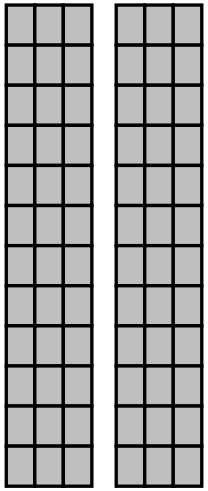


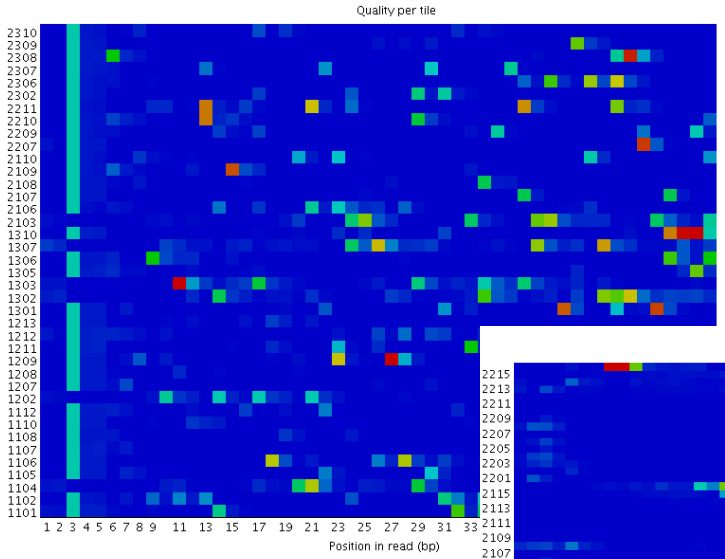
Lanes



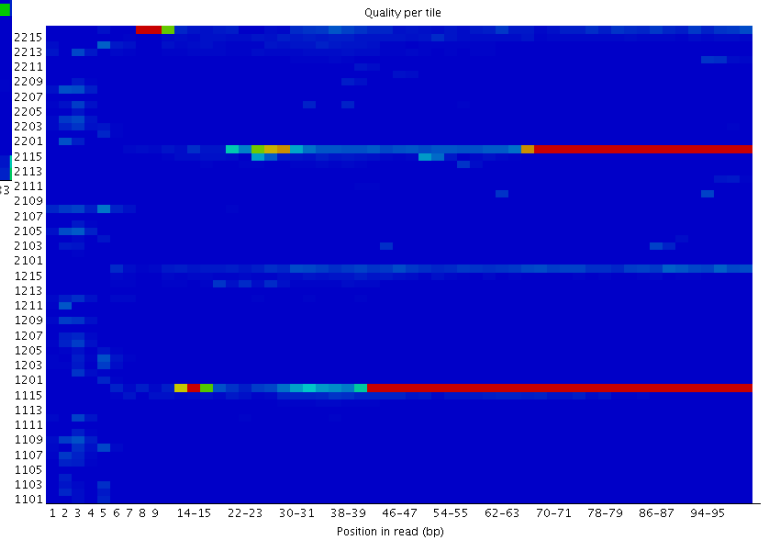
Tiles

Top Bottom





Random Patterning

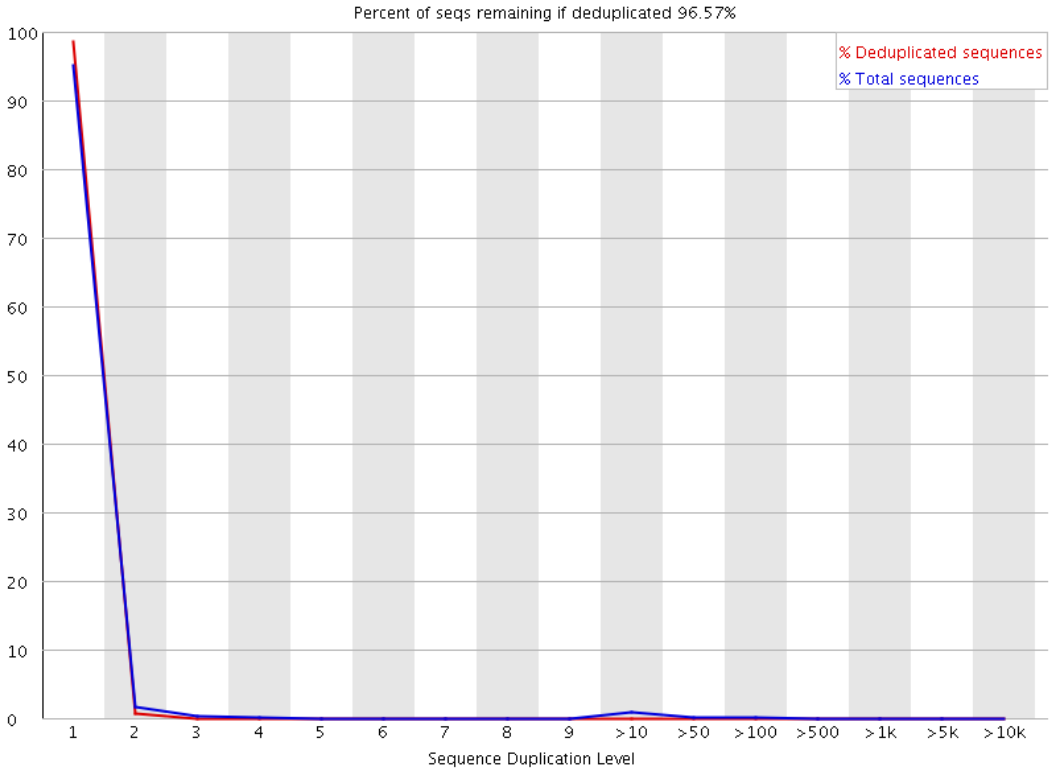


Position Specific Patterning

Duplication

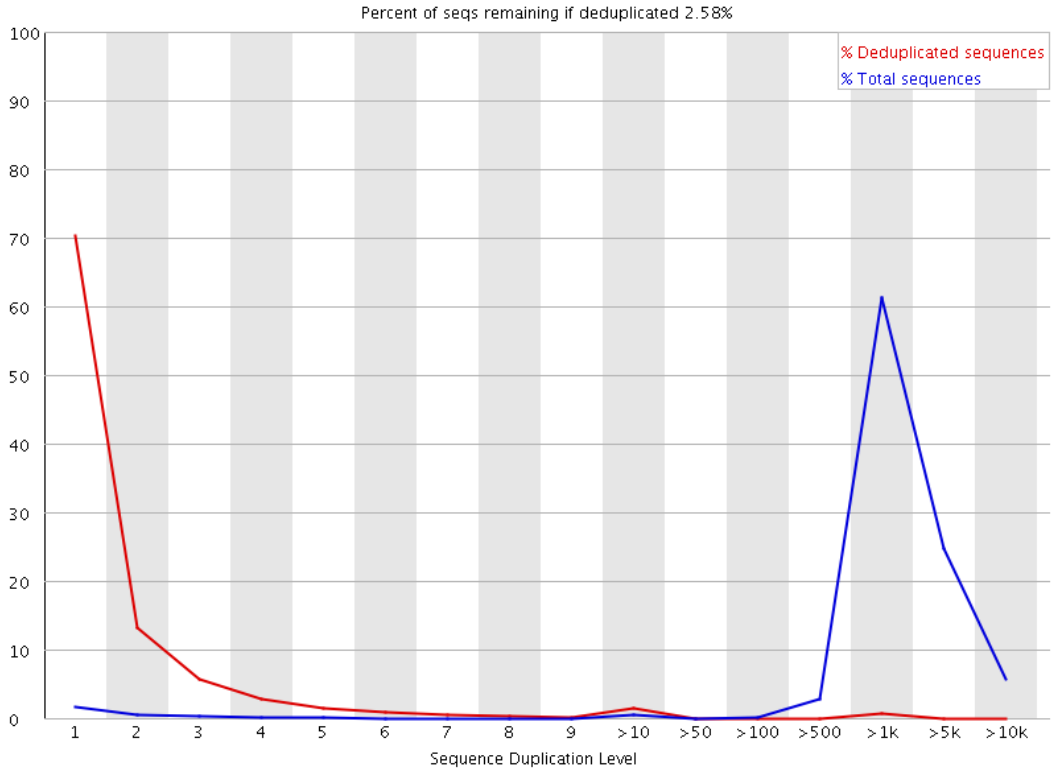
- The exact same sequence appears more than once in your library
  - The sequences come from different biological molecules and the duplication is coincidental
    - Deep sequencing
    - Highly present sequences (repeats for example)
    - Restricted diversity libraries (amplicon sequencing, restricted libraries)
  - The sequences come from the same biological and the duplication is technical
    - PCR duplicates

Duplication



- Most sequences are unique
- High diversity (or low coverage)

Duplication



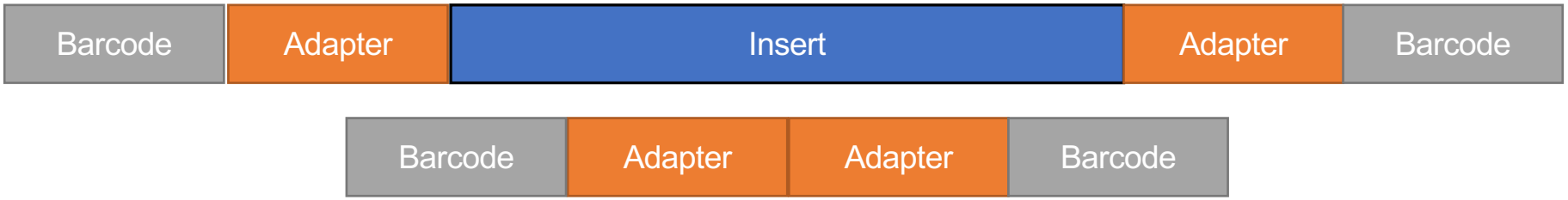
- Most sequences are present many times
- Highly duplicated (low diversity)

Overrepresented Sequences

Sequence	Count	Obs/Exp Overall	Obs/Exp Max	Max Obs/Exp Position
AAAAA	16795015	4.1748657	6.059911	2

- PolyG – Empty space in 2-colour chemistry
- PolyN – Quality too poor to make any calls
- Specific sequences – Normally Adapter Dimers

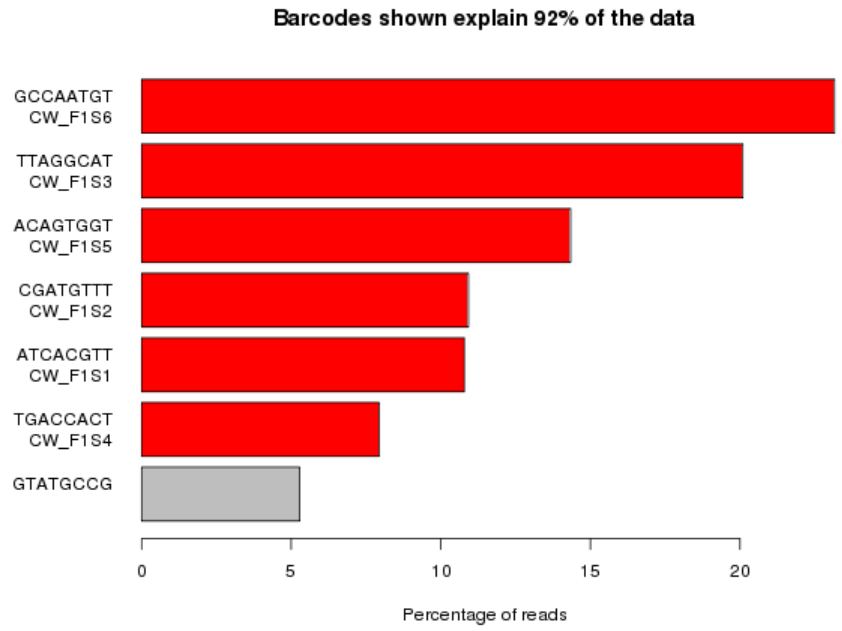
Adapter Dimers



Overrepresented sequences

Sequence	Count	Percentage	Possible Source
GATCGGAAGAGCACACGTCTGAACTCCAGTCACCTTGTAATCTCGTATGC	17957	0.14359551756800035	TruSeq Adapter, Index 12 (100% over 50bp)

Barcode Sequences



Let’s run FASTQC

How Do you run a program on the VACC?

1) Install it in your personal VACC account and then be able to call it

2) Or load it from the shared computing cluster using `module` package