NGS DATA FORMATS & QUALITY CHECK

Blessy M Baby
Ph.D. Scholar
ICAR - IISR

DATA FORMATS

READ FORMATS

- ✓ **SFF**

  Standard Flowgram Format, to hold the "trace" data for 454 reads

- ✓ **SRF**

  Sequence Read Format . Applied Biosystems SRF Conversion Tool (solid2srf) converts SOLiD™ system reads into SRF format.

- ✓ **SCARF**

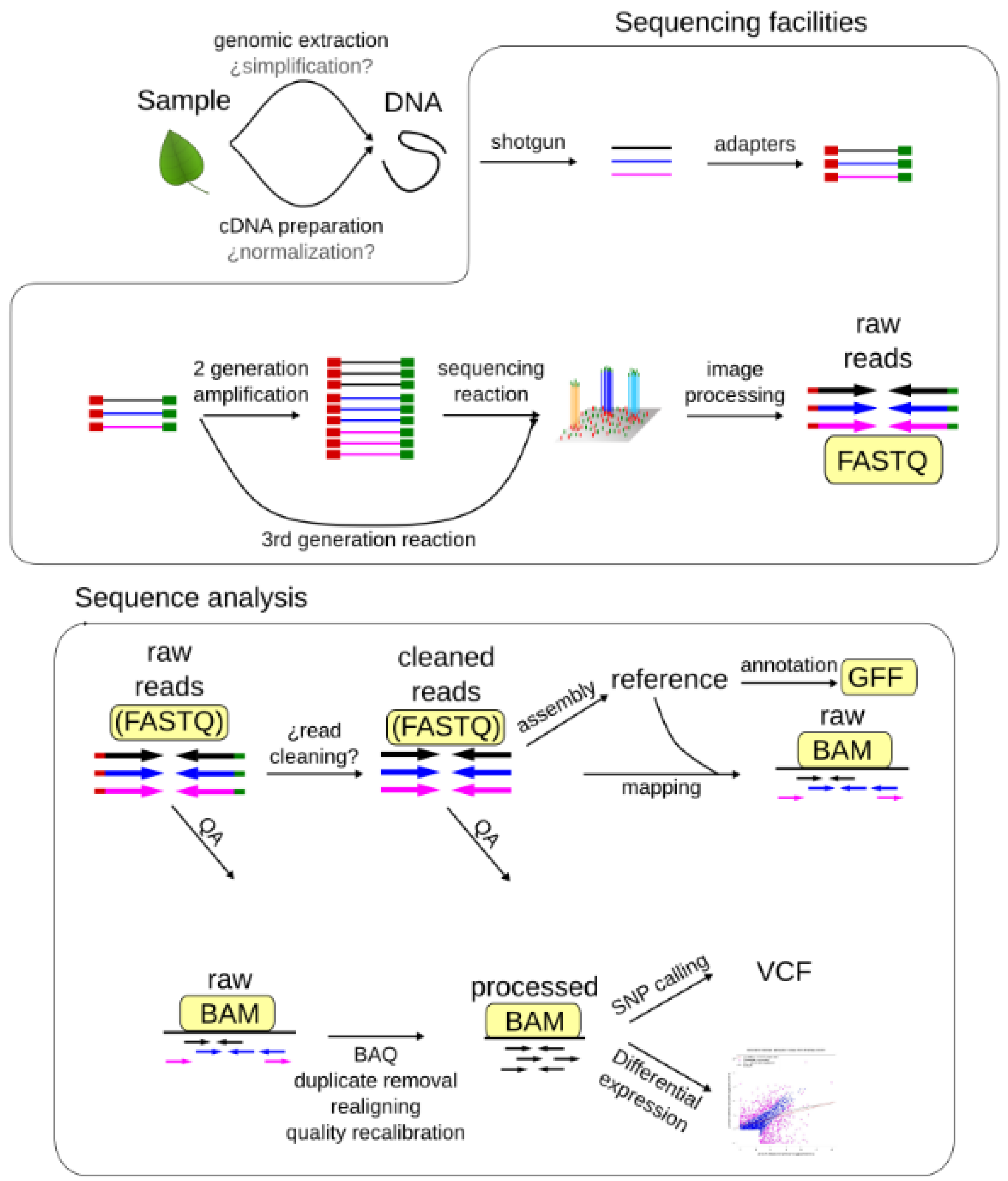
  Solexa Compact ASCII Read Format. This format contains all information for one read in a single line.

- ✓ **SCF**

  first version was described in 1992, since then it has undergone several important changes such as a major reorganization of the ordering of the data items in the file and also in the way they are represented

- ✓ **FASTQ**

  a common format for short reads with quality scores. It is supported in EMBOSS 6.1.0 as a sequence format. Quality scores are also used if the format is more explicitly named in EMBOSS: fastq sanger or fastq illumina



ASSEMBLY FORMATS

- ✓ MAQ

  a compressed binary file format designed for short read alignment

- ✓ MAF, MIRA

  MIRA Assembly Format

- ✓ AMOS A

  A Modular Open-Source Assembler assembly format, used by velvet

- ✓ SAM/BAM

  Sequence Alignment/Map format is a generic format for storing large nucleotide sequence alignments

GENOME ANNOTATION & VARIATION FORMATS

- ✓ GFF format

  latest version GFF3. It's a tabular plain-text format for genome or sequence annotation, can contain also the sequences, alignments, dependencies between features

- ✓ BioXSD

  a new set of structured, “object-oriented” formats for exchange of sequence data, any kind of sequence/genome annotation, and related

- ✓ VCF

  A standard file format for storing variation data. VCF is a preferred format because it is unambiguous, scalable and flexible, allowing extra information to be added to the info field.

Reads	FASTQ-SFF (BAM)
↓	
Alignments	SAM/BAM
↓	
Variations	VCF
Genome	FASTA
Genome annotations	GFF/BED

FASTA

- The fasta format is based on a simple text. Each sequence starts with a “>” followed by the sequence name, an space and, optionally, the description

- Two components of a FASTA sequence

  Header line - begins with a > character, followed by a sequence name and an optional description

  Sequence line – the sequence data either all on a single line or spread across multiple lines

```
>seq_1 description
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
>seq_2
ATCGTAGTCTAGTCTATGCTAGTGCGATGCTAGTGCTAGTCGTATGCATGGCTATGTGTG
```

FASTQ

- Text based format for storing both DNA sequence and its corresponding quality scores
- Illumina FASTQ Format

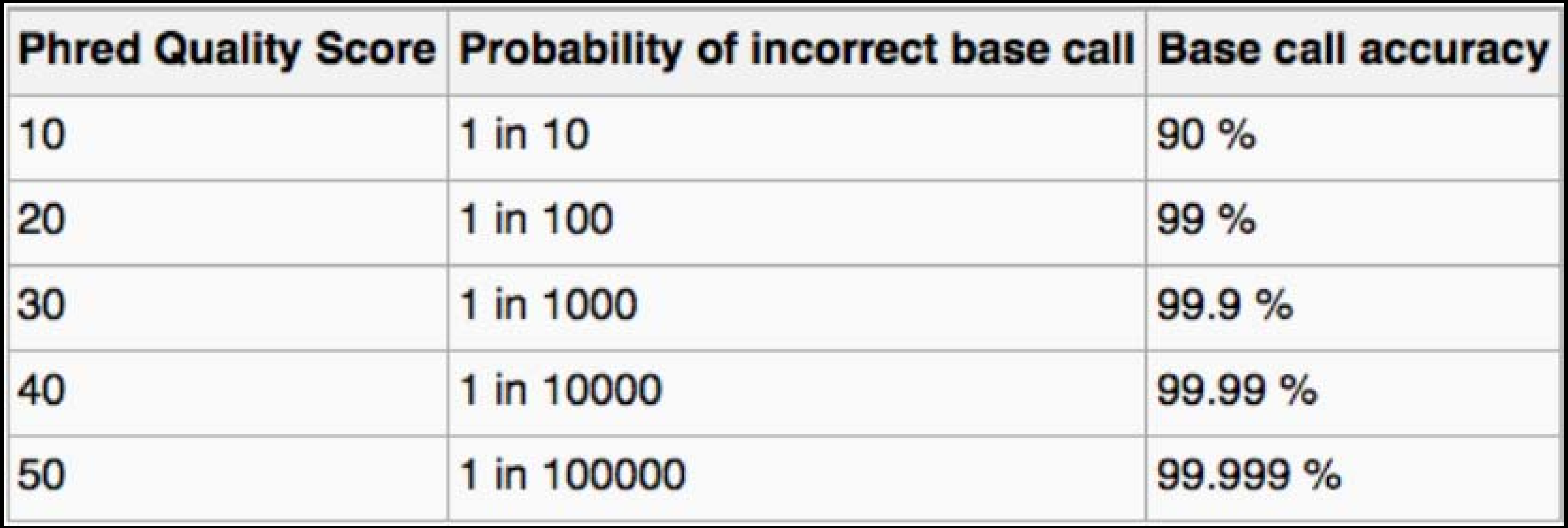
```
@HWIIST375_119:5:2308:19782:7202/1
AGATCACCTGATCTTGACGAAATTTCCAACTTATAACTATTA
+
7@;DD>DDHD>FABGH<F>F<G=H4A?+4CFE4BDA@CFC
```

@HWIIST375_119:5:2305:20392:171561/1	Sequence Identifier
CTTAATGTTTATATACTATGTTGCTATTATGAACTGAACTTAT	Sequence read
+	Sequence quality header
@@@FDFFFGHDHFFGGIEHHEAFEHEHIHBHHIFHEHIGI	Sequence quality score

```
@HWIIST375_119:5:2105:14681:149697/1
GAGAAGGTATTGAAAAATGATAGGTTTTTAATTAAAGATG
+
B@CFFFFDHHHHHJJJJJJIJJJJHJJJJJIJJIJJJIJJJJJJJJJJFIIJJIJJJJ
```

FASTQ

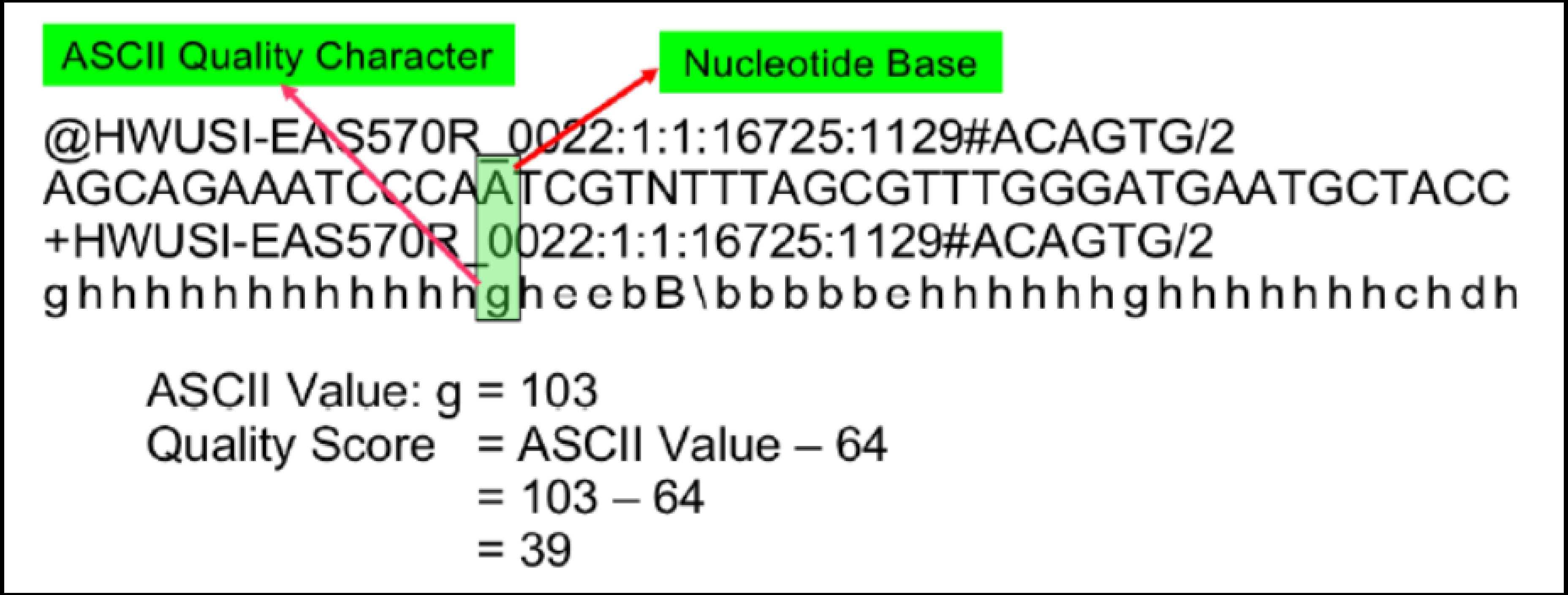
- Phred quality score Q(score), is defined as a property that is logarithmically related to the base calling error probabilities (P) : $Q = -10 \log_{10} P$
- Historically used to determine Sanger sequencing accuracy, this method proved to be highly accurate across a range of sequencing chemistries and instruments, making it the quality scoring standard for commercial sequencing technologies



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 %

FASTQ

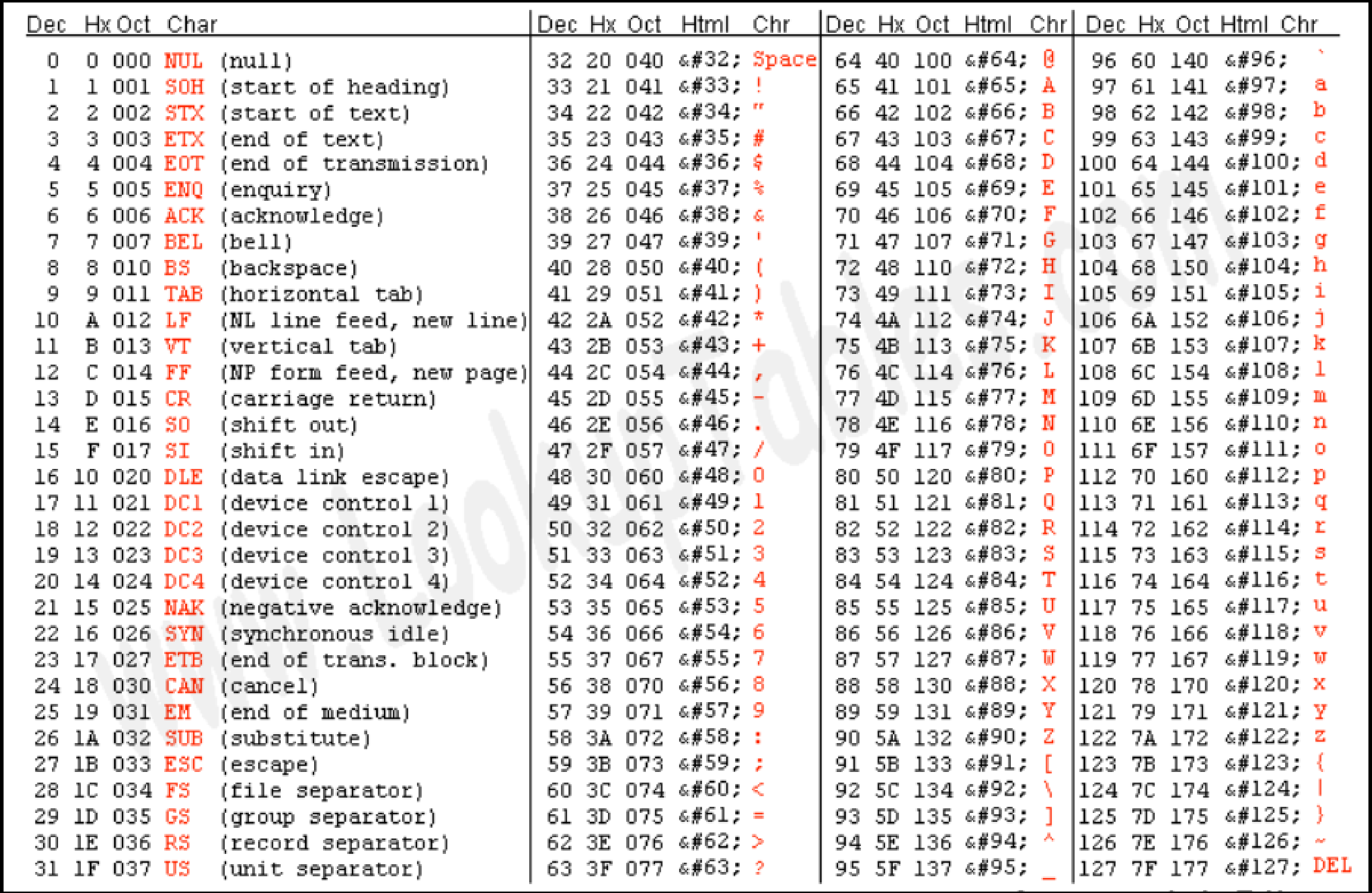
- In FASTQ format, ASCII characters are used to represent Q score for individual bases
- ASCII encryptions are done by adding 33 or 64 to the Q score and converting to ASCII
- Q-score 20 is accepted as a good quality score for a base, while Q 30 is more stringent and should be used in case of more abundance of raw data



```
@HWUSI-EAS570R_0022:1:1:16725:1129#ACAGTG/2
AGCAGAAATCCCAATCGTNTTTAGCGTTTGGGATGAATGCTACC
+HWUSI-EAS570R_0022:1:1:16725:1129#ACAGTG/2
ghhhhhhhhhhhghcebB\bbbbbehhhhhhghhhhhhhchdh
```

ASCII Value: g = 103
Quality Score = ASCII Value – 64
= 103 – 64
= 39

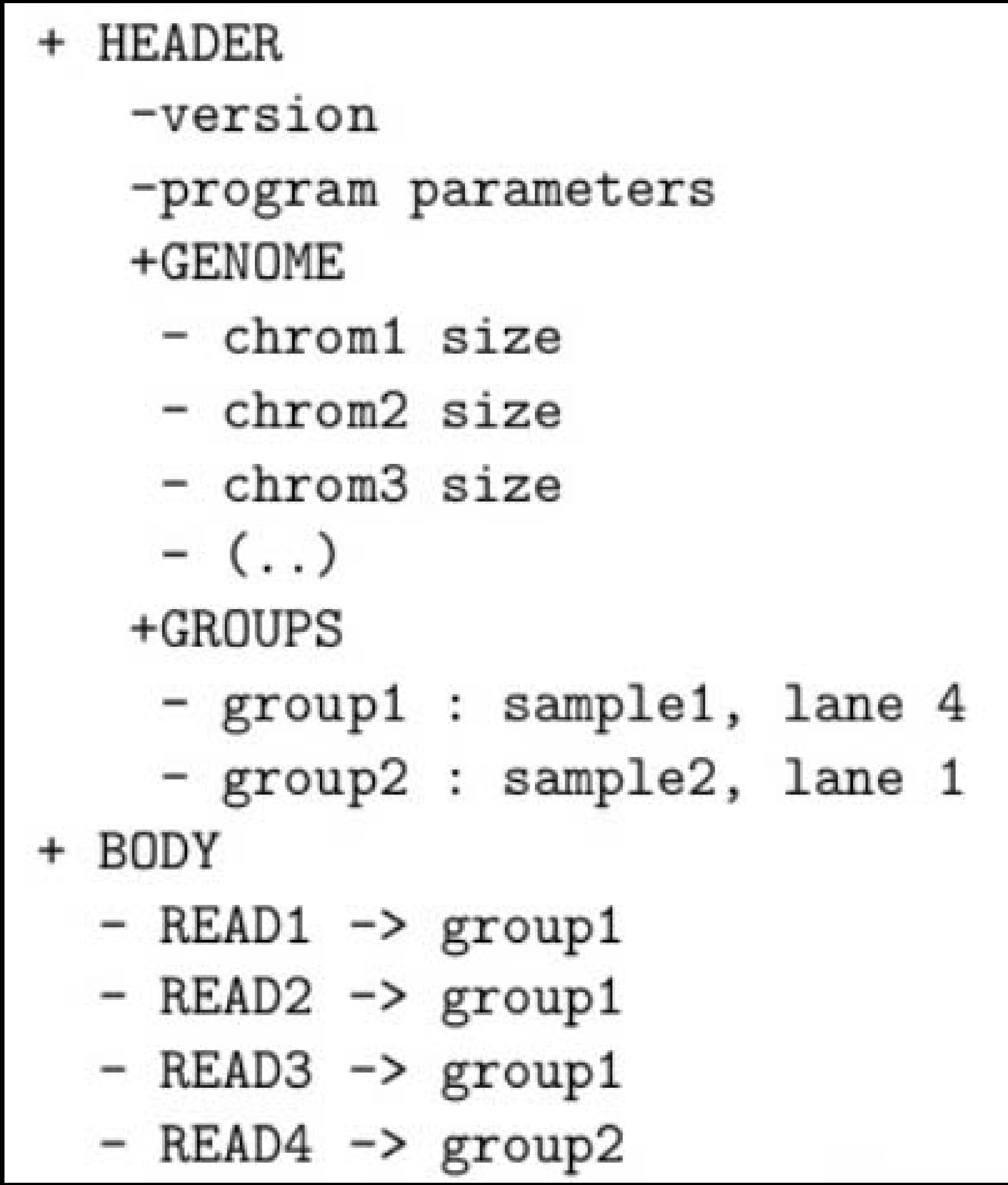
ASCII Conversion Table



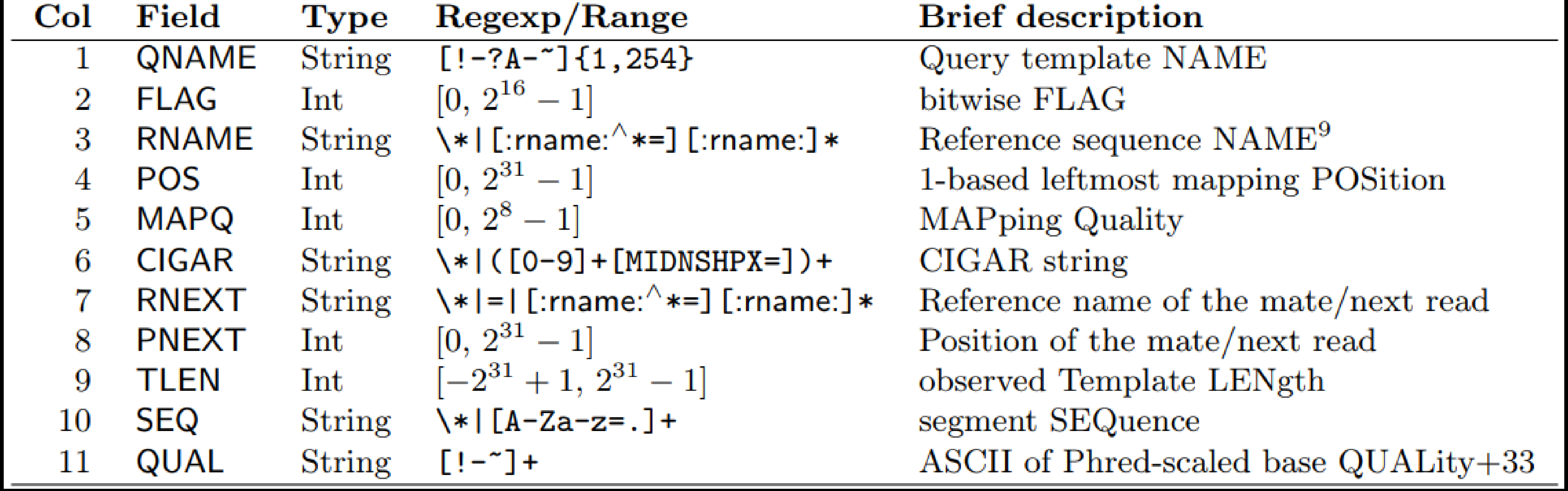
Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

SAM format

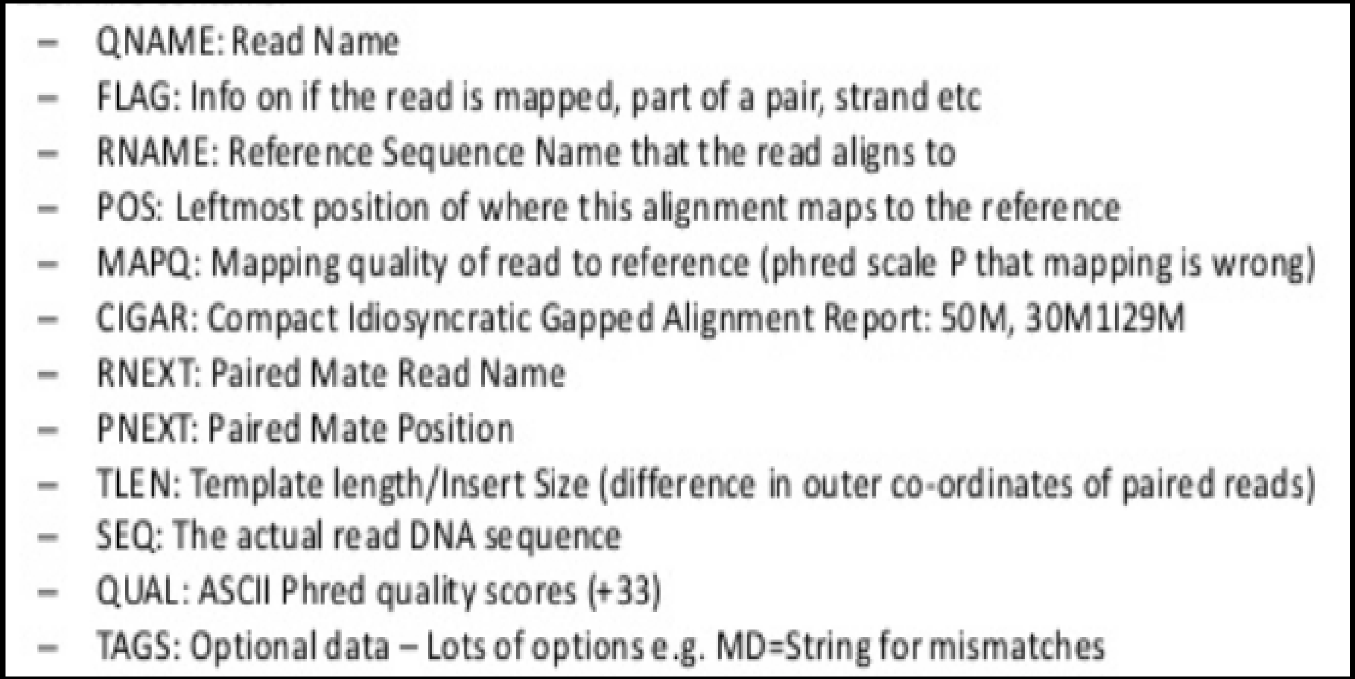
- SAM – Sequence Alignment Map format is a generic format for storing large nucleotide sequence alignments
- SAM data file are output from aligners that read FASTQ file and assign sequence/reads to a position with respect to a known reference genome.
- Its simple, flexible and compact enough to store all the alignment information generated by various alignment programs
- Allows most of the operations on the alignment to work on a stream without the whole alignment into memory
- Allows the file to be indexed by genomic position to efficiently retrieve all reads aligning to a locus



```
+ HEADER
    -version
    -program parameters
    +GENOME
     - chrom1 size
     - chrom2 size
     - chrom3 size
     - (..)
    +GROUPS
     - group1 : sample1, lane 4
     - group2 : sample2, lane 1
+ BODY
   - READ1 -> group1
   - READ2 -> group1
   - READ3 -> group1
   - READ4 -> group2
```



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	`\*|[:rname:^*=][:rname:]*`	Reference sequence NAME[9]
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	`\*|=|[:rname:^*=][:rname:]*`	Reference 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



- QNAME: Read Name
- FLAG: Info on if the read is mapped, part of a pair, strand etc
- RNAME: Reference Sequence Name that the read aligns to
- POS: Leftmost position of where this alignment maps to the reference
- MAPQ: Mapping quality of read to reference (phred scale P that mapping is wrong)
- CIGAR: Compact Idiosyncratic Gapped Alignment Report: 50M, 30M1I29M
- RNEXT: Paired Mate Read Name
- PNEXT: Paired Mate Position
- TLEN: Template length/Insert Size (difference in outer co-ordinates of paired reads)
- SEQ: The actual read DNA sequence
- QUAL: ASCII Phred quality scores (+33)
- TAGS: Optional data – Lots of options e.g. MD=String for mismatches

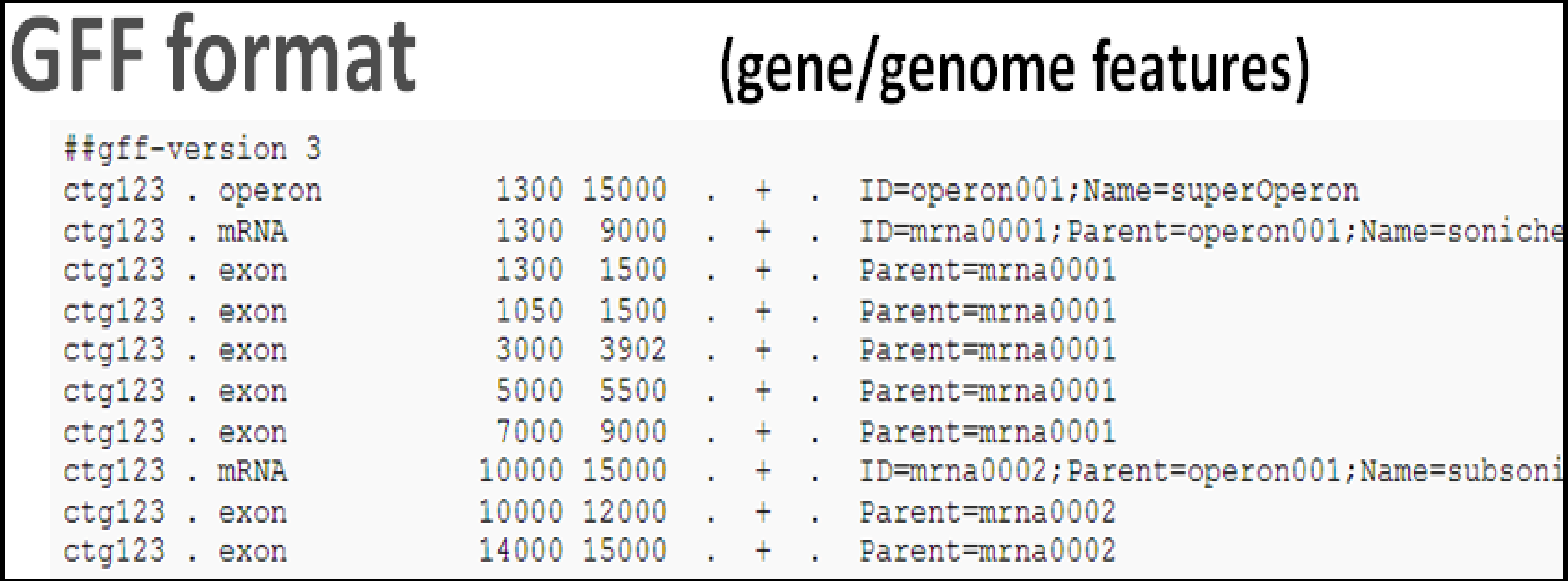
- **SAM files can be analyzed and edited with the software SAMtools.**
- **The header section must be prior to the alignment section if it is present.**
- **Headings begin with the '@' symbol, which distinguishes them from the alignment section.**
- **Alignment sections have 11 mandatory fields, as well as a variable number of optional fields.**

Genome & sequence annotation formats

- The GFF (General Feature Format) format consists of one line per feature, each containing 9 columns of data, plus optional track definition lines.
- The first line of a GFF3 file must be a comment that identifies the version, e.g.

  ##gff-version 3

- Fields must be tab-separated. Also, all but the final field in each feature line must contain a value, "empty" columns should be denoted with a ‘ . ‘



GFF format (gene/genome features)

```
##gff-version 3
ctg123 . operon      1300 15000  .  +  .  ID=operon001;Name=superOperon
ctg123 . mRNA        1300  9000  .  +  .  ID=mrna0001;Parent=operon001;Name=soniche
ctg123 . exon        1300  1500  .  +  .  Parent=mrna0001
ctg123 . exon        1050  1500  .  +  .  Parent=mrna0001
ctg123 . exon        3000  3902  .  +  .  Parent=mrna0001
ctg123 . exon        5000  5500  .  +  .  Parent=mrna0001
ctg123 . exon        7000  9000  .  +  .  Parent=mrna0001
ctg123 . mRNA       10000 15000  .  +  .  ID=mrna0002;Parent=operon001;Name=subsoni
ctg123 . exon       10000 12000  .  +  .  Parent=mrna0002
ctg123 . exon       14000 15000  .  +  .  Parent=mrna0002
```

Position	Position name	Description
1	sequence	The name of the sequence where the feature is located.
2	source	Keyword identifying the source of the feature, like a program (e.g. Augustus or RepeatMasker) or an organization (like TAIR).
3	feature	The feature type name, like "gene" or "exon". In a well structured GFF file, all the children features always follow their parents in a single block (so all exons of a transcript are put after their parent "transcript" feature line and before any other parent transcript line).
4	start	Genomic start of the feature, with a 1-base offset. This is in contrast with other 0-offset half-open sequence formats, like BED files.
5	end	Genomic end of the feature, with a 1-base offset. This is the same end coordinate as it is in 0-offset half-open sequence formats, like BED files.[citation needed]
6	score	Numeric value that generally indicates the confidence of the source on the annotated feature. A value of "." (a dot) is used to define a null value.
7	strand	Single character that indicates the Sense (molecular biology) strand of the feature; it can assume the values of " " (positive, or 5'->3'), "-", (negative, or 3'->5'), "." (undetermined).
8	phase	phase of CDS features; it can be either one of 0, 1, 2 (for CDS features) or "." (for everything else).
9	Attributes.	All the other information pertaining to this feature. The format, structure and content of this field is the one which varies the most between the three competing file formats.

QUALITY CHECK

- NGS technologies provide a high-throughput means to generate large amount of sequence data.
- Quality Check (QC) of sequence data generated from the technologies is extremely important for meaningful downstream analysis.
- Highly efficient and fast processing tools are required to handle the large volume of datasets.
- QC aims to get a quality report which can spot problems which originate either in the sequencer or in the starting library material.
- Quality check and primary analysis of raw sequence data is vital step prior to the in-depth analysis.

FastQC

- FastQC aims to provide a QC report which can spot problems which originate either in the sequencer or in the starting library material.
- A java based tool, aims to provide a simple way to perform quality control checks on raw sequence data
- Accepts data from Illumina, Roche 454 platforms and other long read sequencing platforms
- Provides a modular set of analyses, which can give a quick impression of whether the data has any problems, needs to be solved before doing any further analysis.
- Provides quality report for *Per_base_quality, Per_sequence_quality, Per_base_sequence_content, Per_sequence_GC_content, Sequence_Duplication_Levels, etc*
- Provides summary graphs and tables to quickly assess the raw data

FastQC

- FasQC can be run in one of two modes.
  - It can either run as a standalone interactive application for the immediate analysis of small numbers of FastQ files.
  - It can be run in a non-interactive mode where it would be suitable for integrating into a larger analysis pipeline for the systematic processing of large numbers of files.
- FastQC supports files in the following formats
  - FastQ (all quality encoding variants)
  - CasavaFastQ files
  - ColorspaceFastQ
  - GZip compressed FastQ
  - SAM
  - BAM
  - SAM/BAM Mapped only (normally used for colour space data)

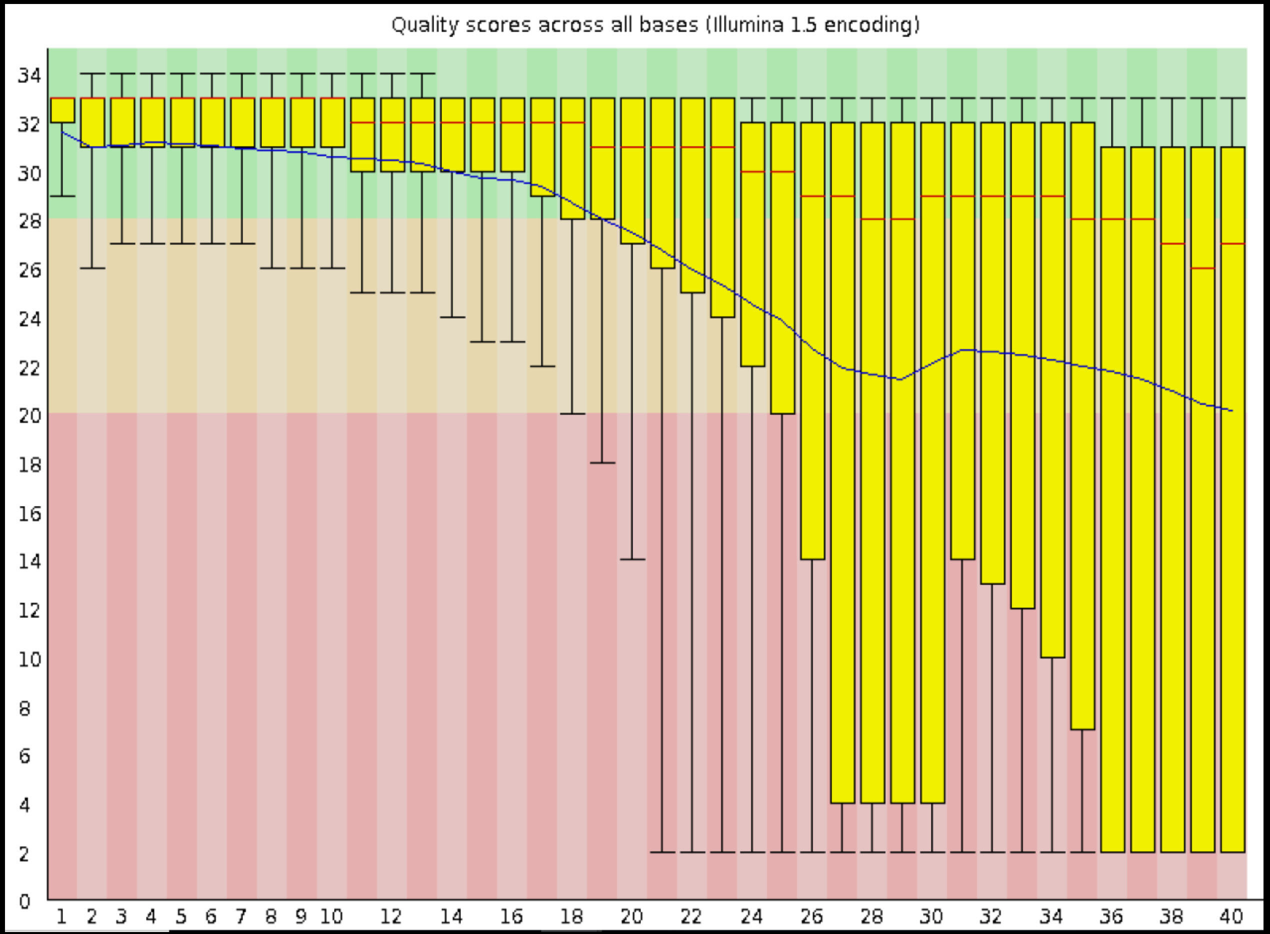
FastQC

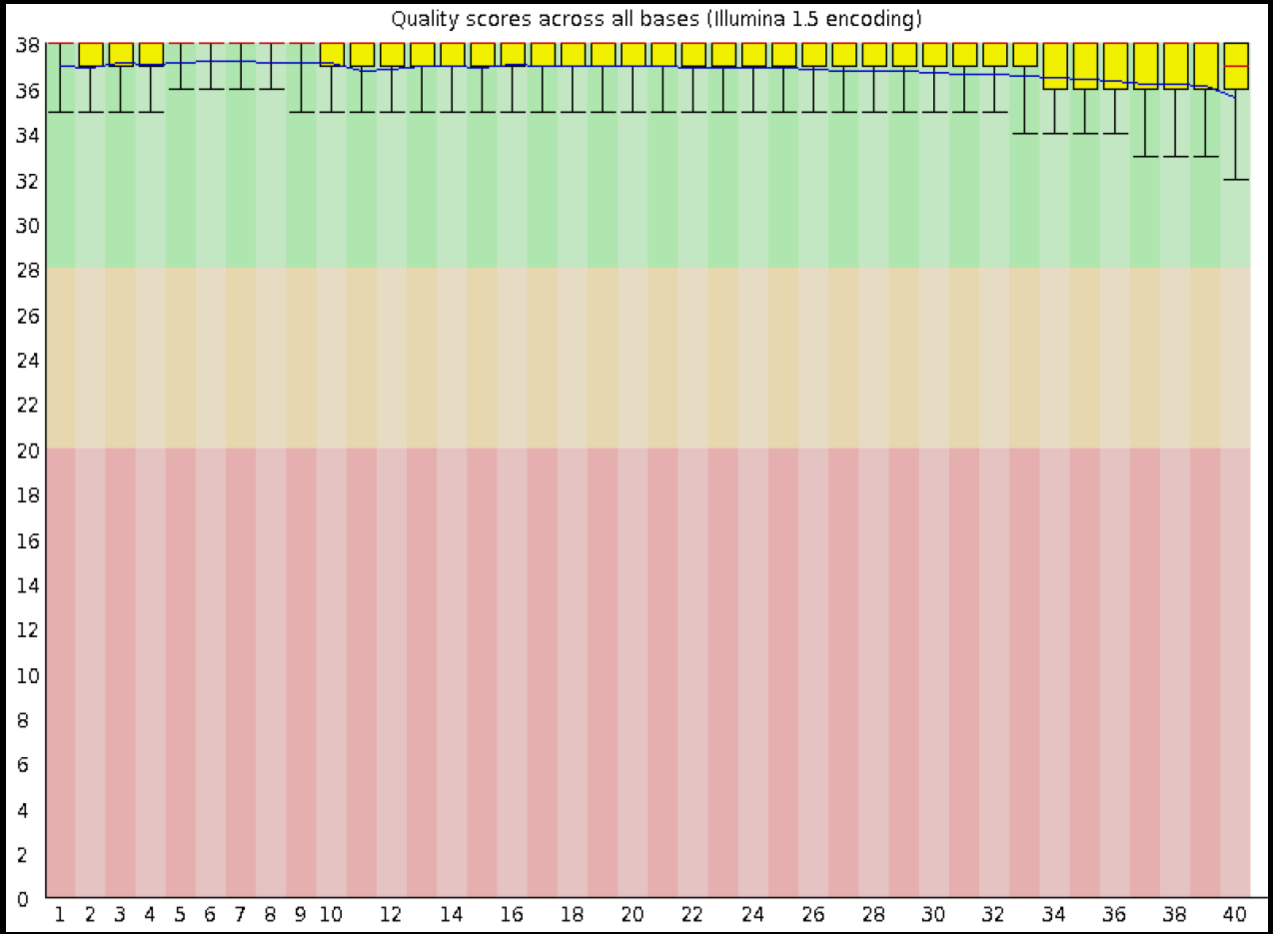
BASIC STATISTICS

- Filename: The original filename of the file which was analyzed
- File type: whether the file appeared to contain actual base calls or color space data which had to be converted to base calls
- Encoding: which ASCII encoding of quality values was found in this file.
- Total Sequences: A count of the total number of sequences processed. There are two values reported, actual and estimated.
- Filtered Sequences: If running in Casava mode sequences flagged to be filtered will be removed from all analyses. The number of such sequences removed will be reported here. The total sequences count above will not include these filtered sequences and the number of sequences actually used for the rest of the analysis.
- Sequence Length: Provides the length of the shortest and longest sequence in the set. If all sequences are the same length only one value is reported. %GC: The overall %GC of all bases in all sequence

Per Base Sequence Quality

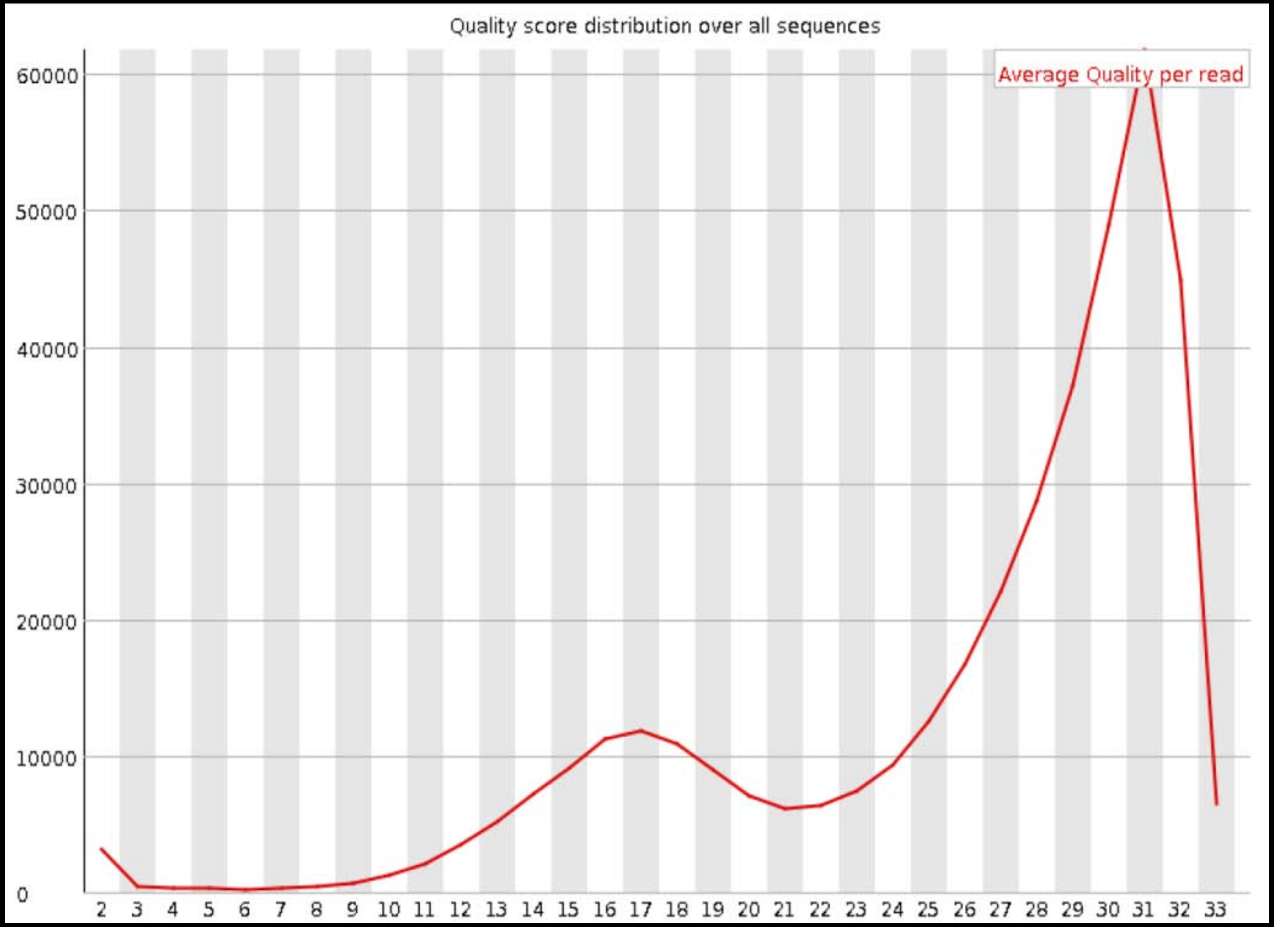
- The y-axis on the graph shows the quality scores.
- The higher the score the better the base call.

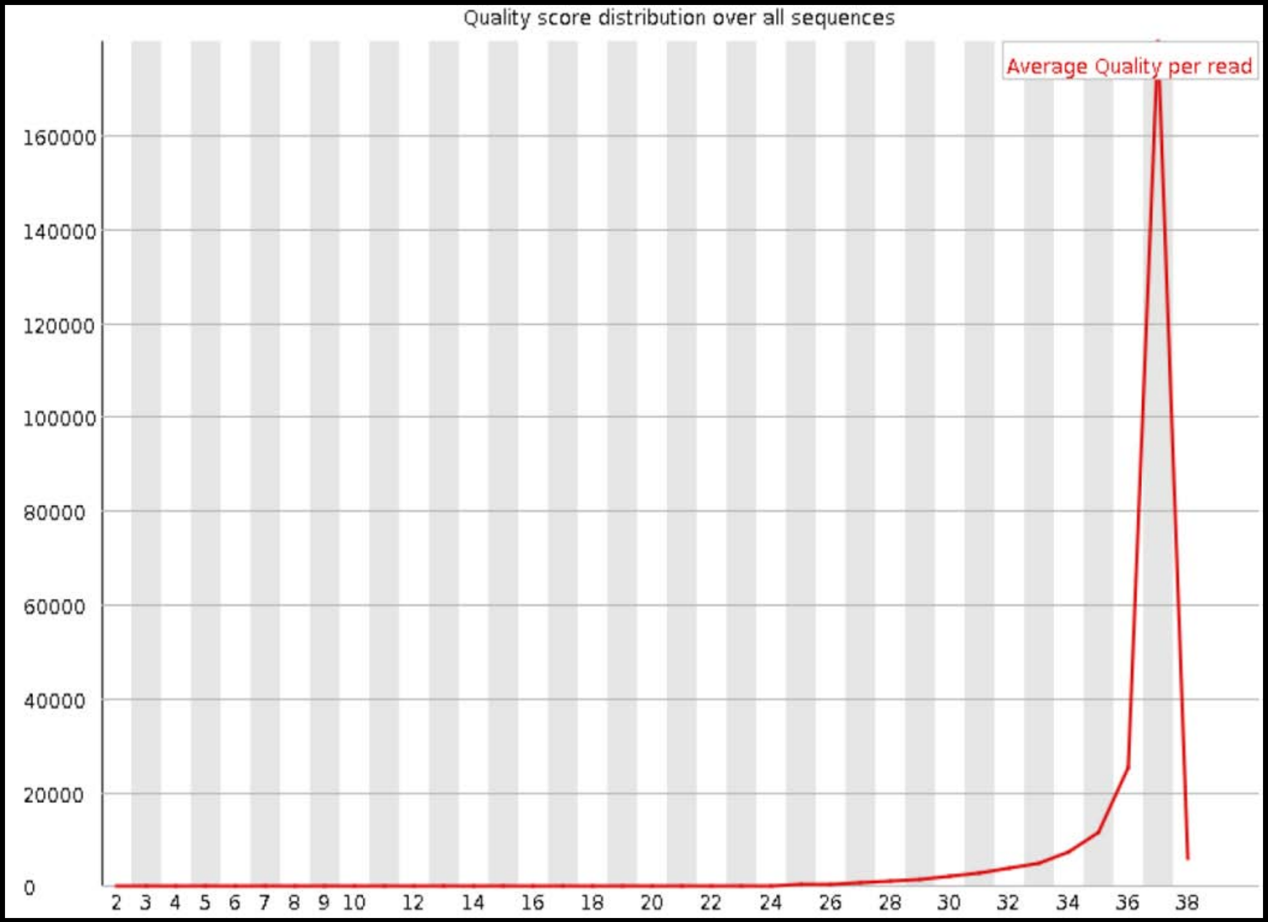




Per Sequence Quality Scores

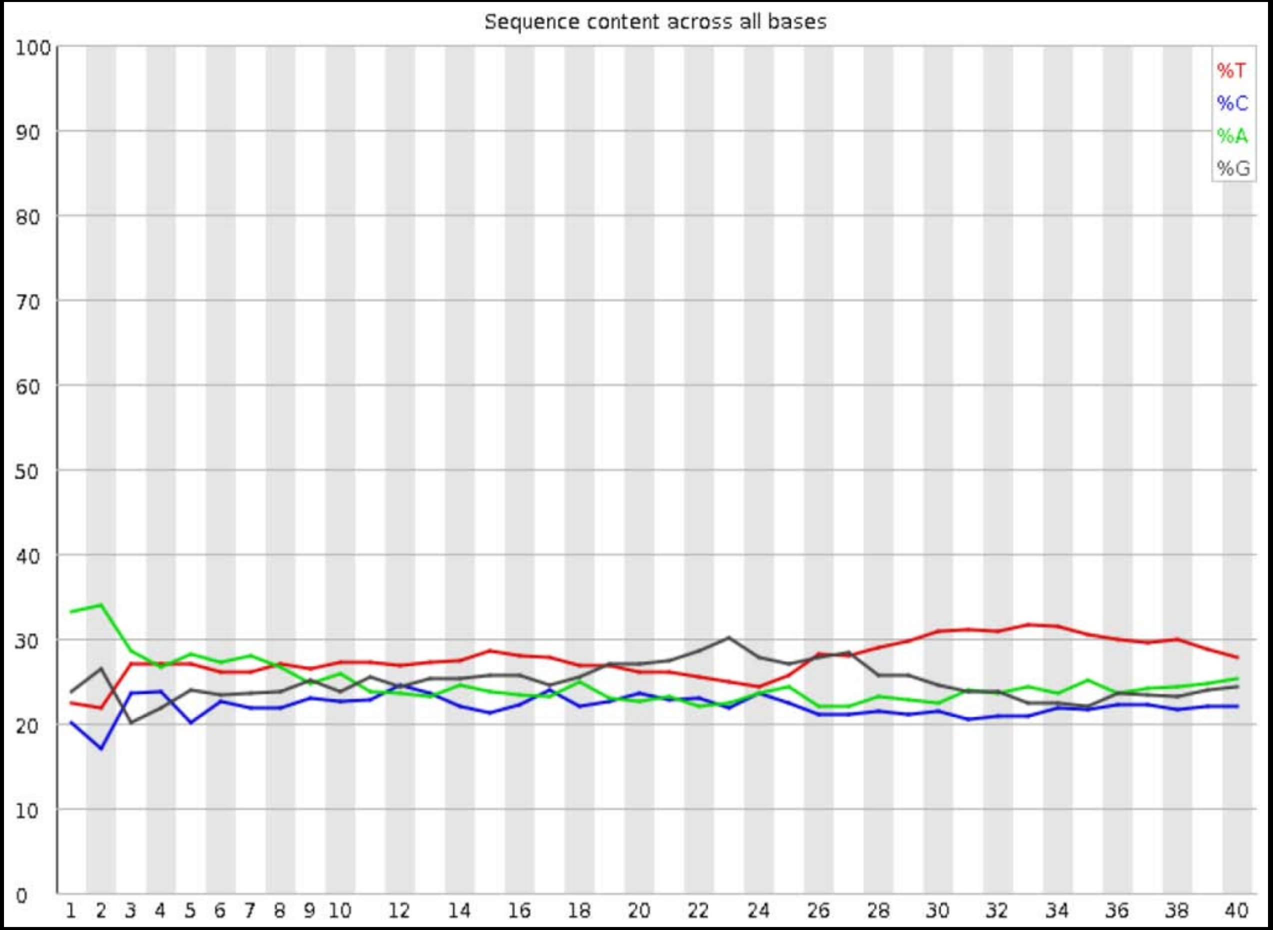
- The per sequence quality score report allows you to see if a subset of your sequences have universally low quality values

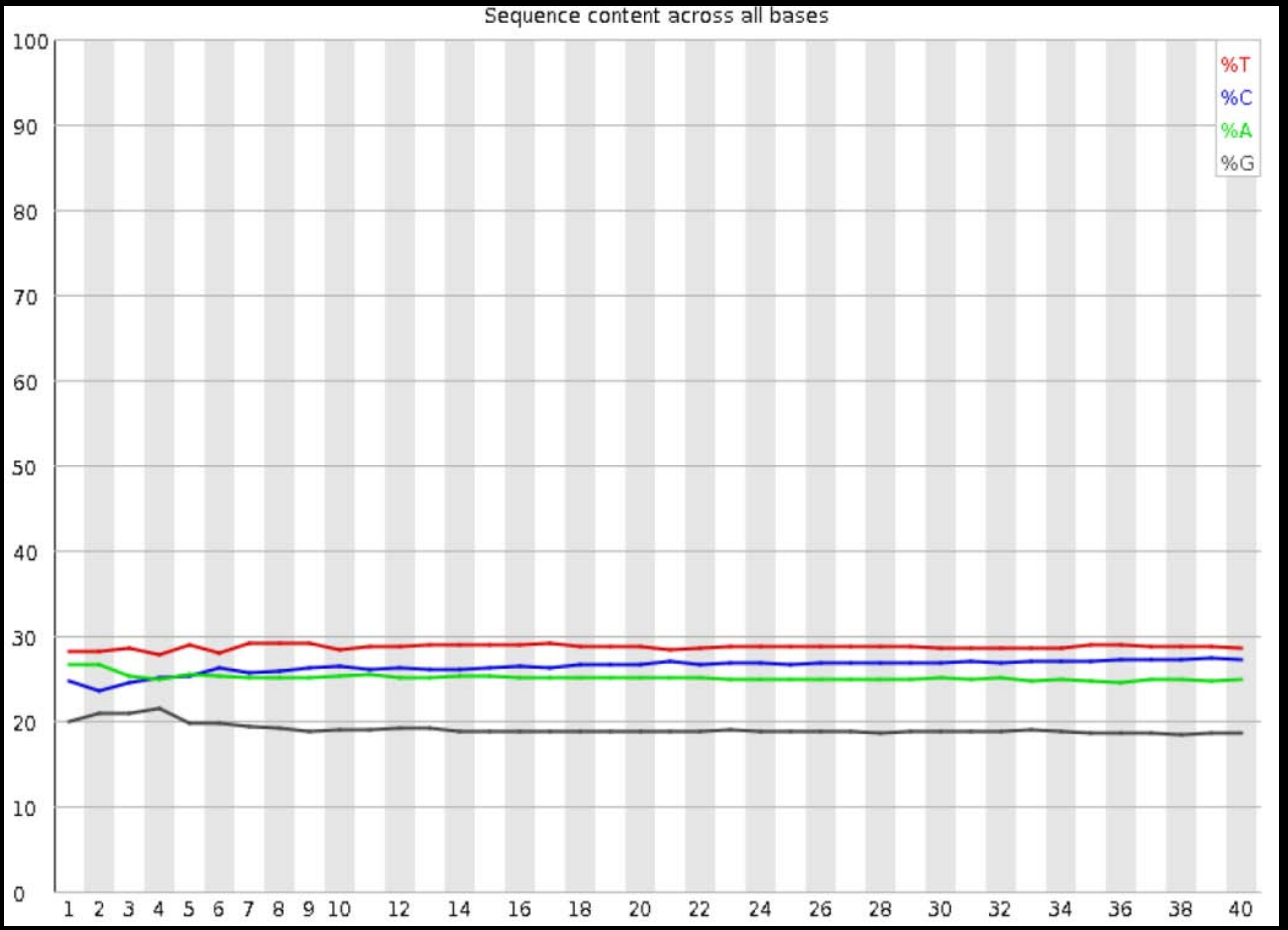




Per Base Sequence Content

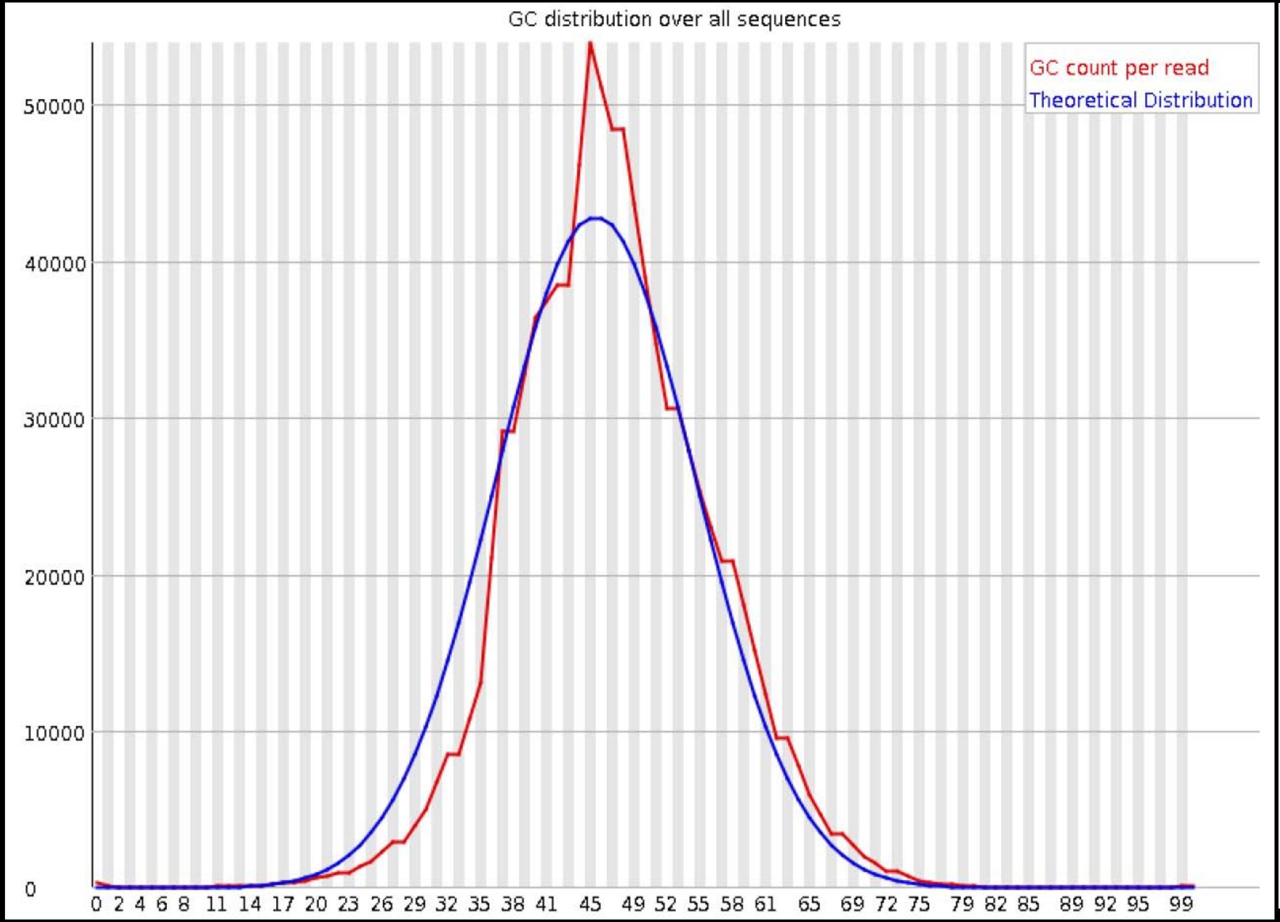
- Per Base Sequence Content plots out the proportion of each base position in a file for which each of the four normal DNA bases has been called
- In a random library you would expect that there would be little to no difference between the different bases of a sequence run, so the lines in this plot should run parallel with each other

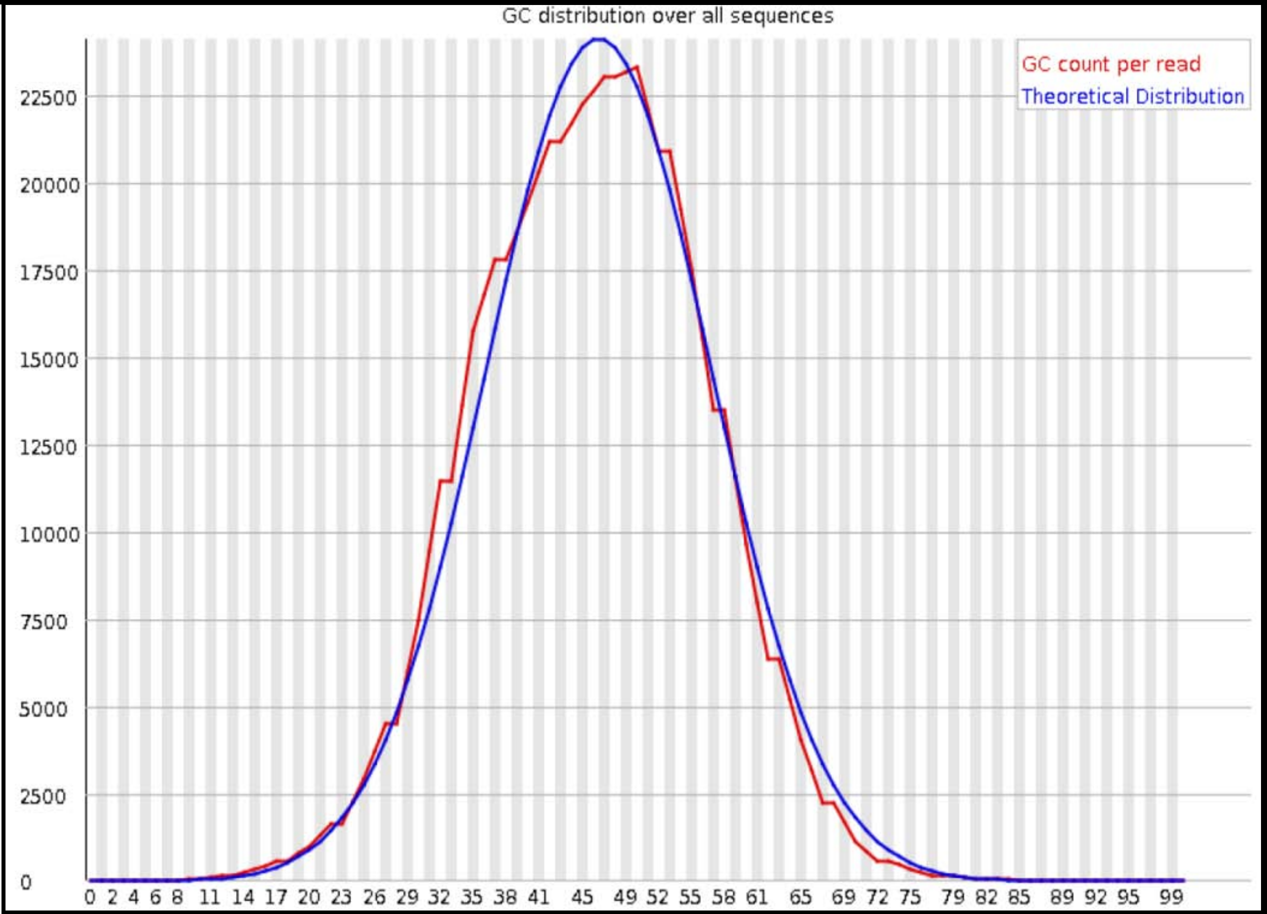




Per Sequence GC Content

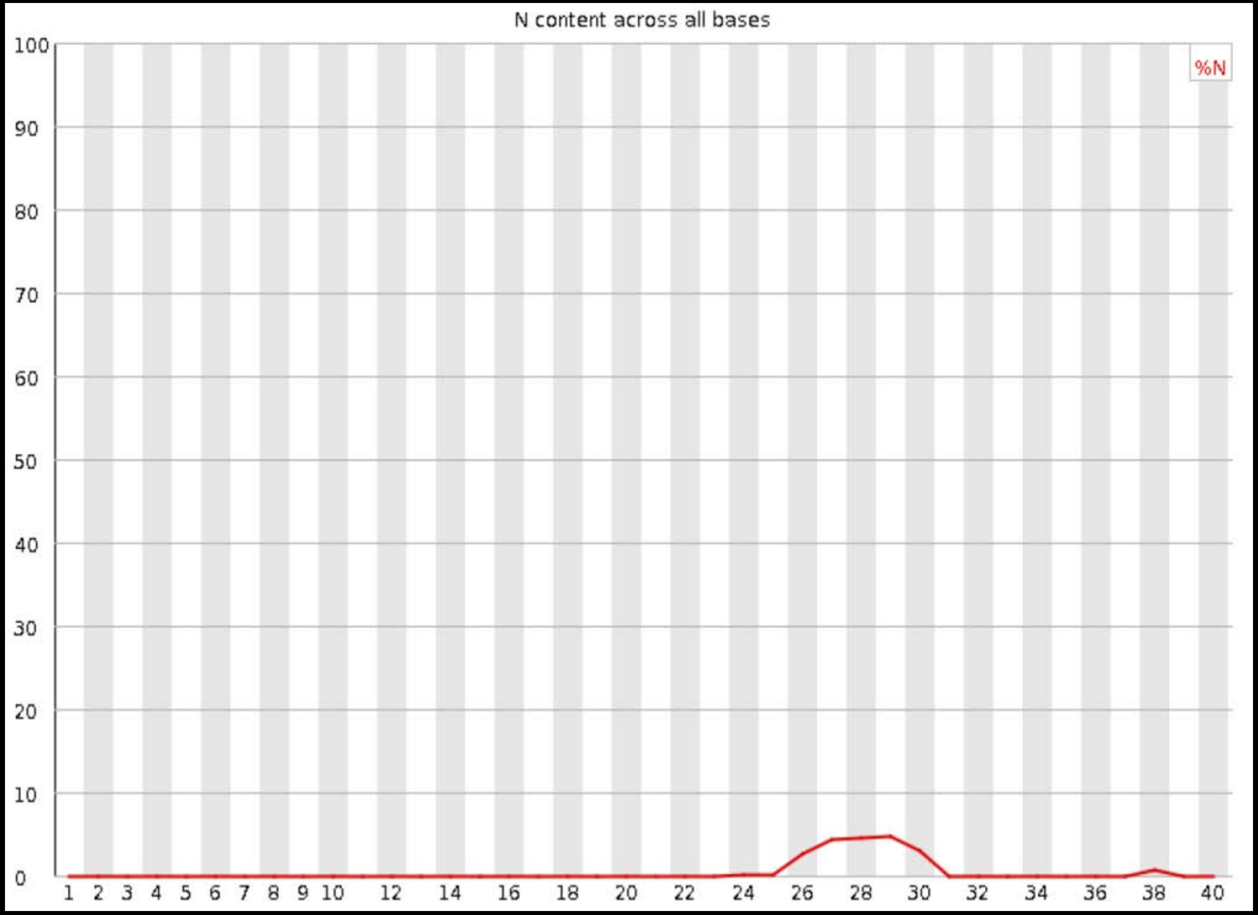
- In a normal random library you would expect to see a roughly normal distribution of GC content where the central peak corresponds to the overall GC content of the underlying genome

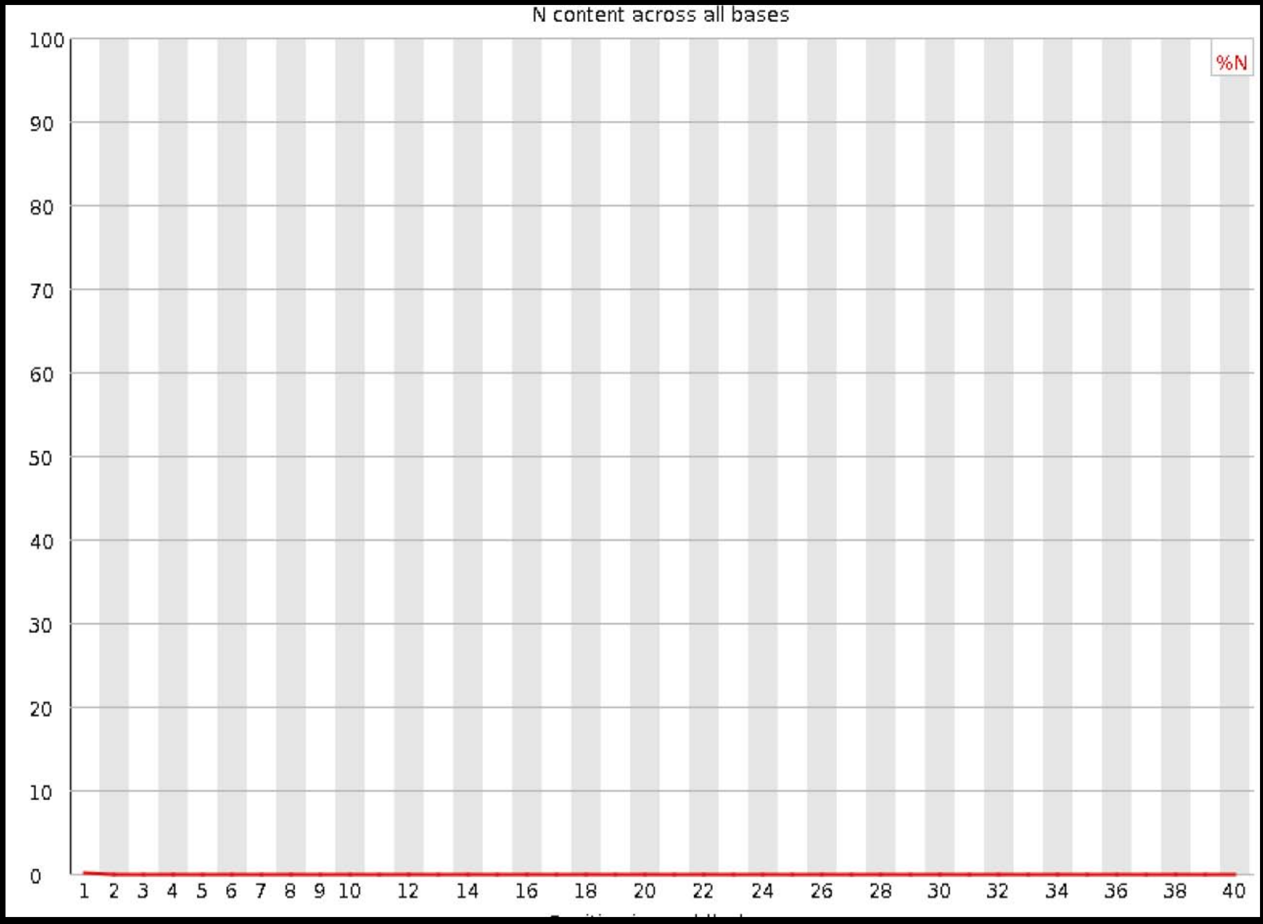




Per Base N Content

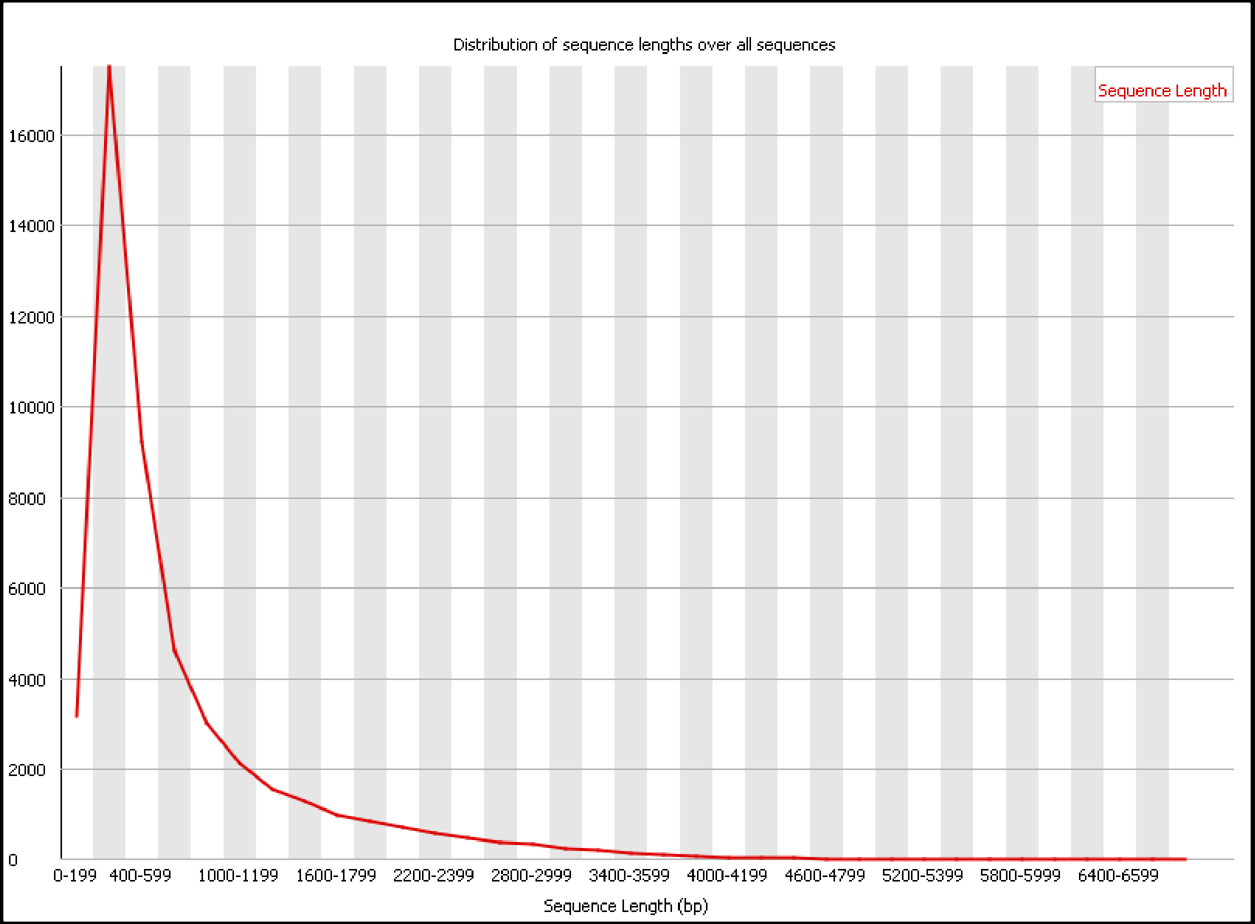
- If a sequencer is unable to make a base call with sufficient confidence then it will normally substitute an N rather than a conventional base call.
- This module plots out the percentage of base calls at each position for which an N was called.

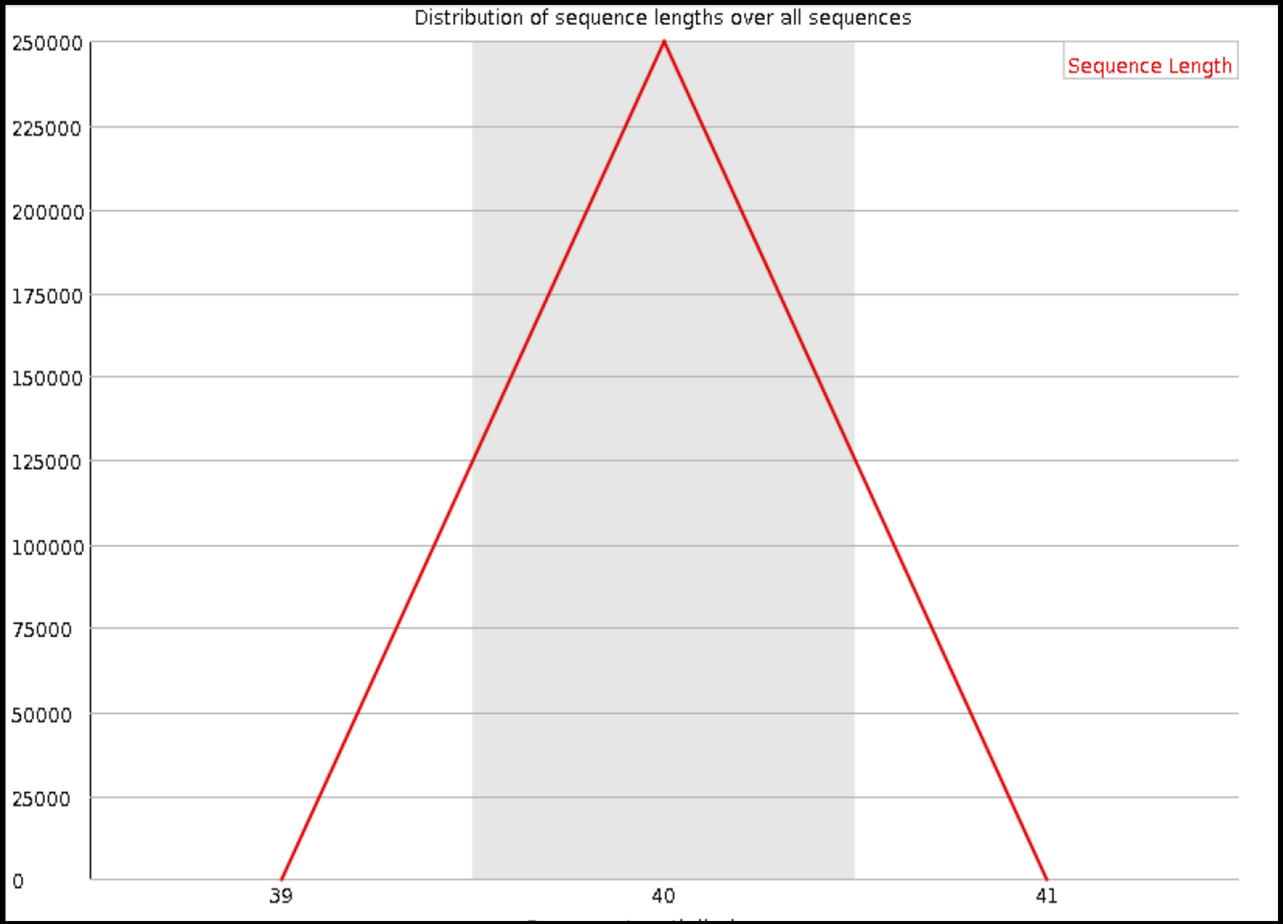




Sequence Length Distribution

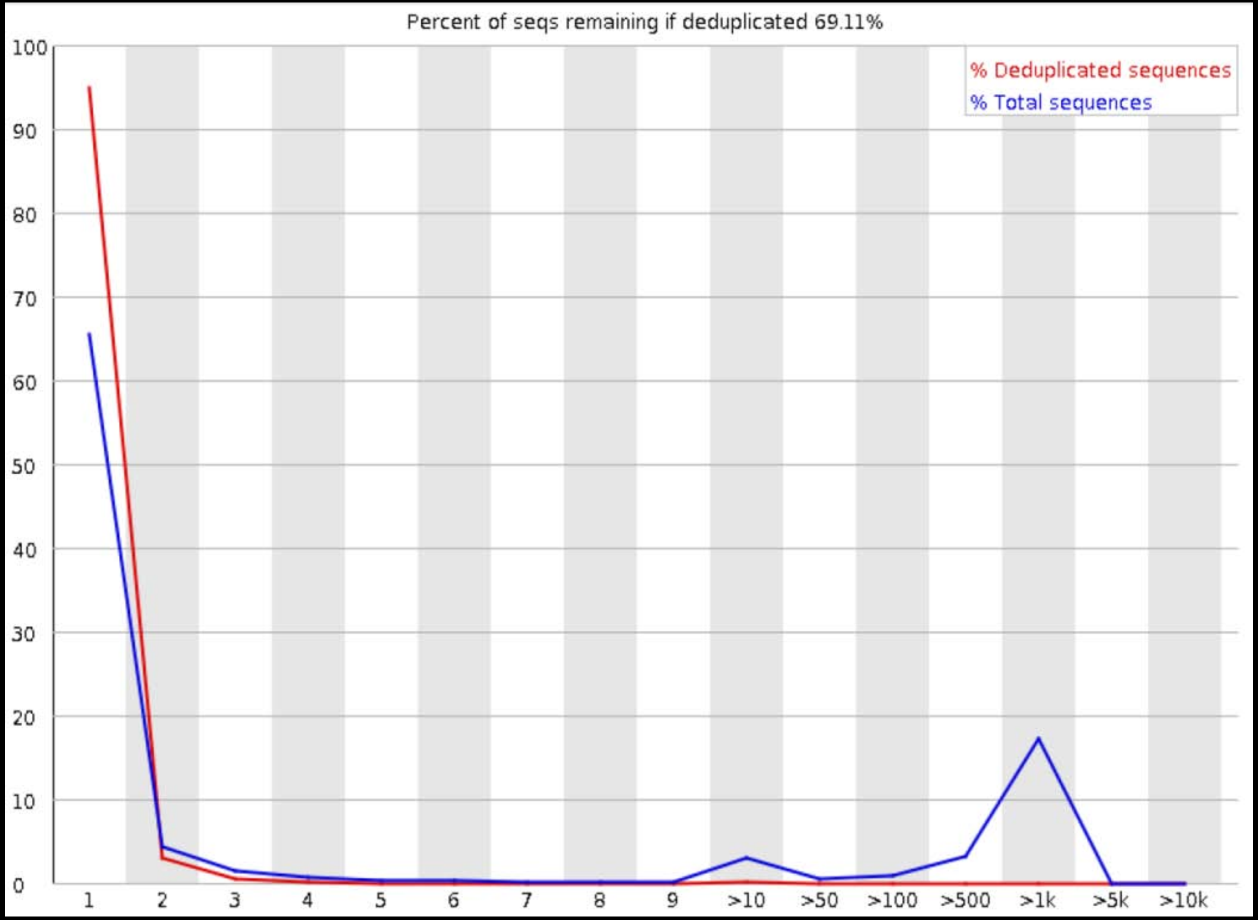
- Some high throughput sequencers generate sequence fragments of uniform length, but others can contain reads of wildly varying lengths.
- This module generates a graph showing the distribution of fragment sizes in the file which was analyzed.

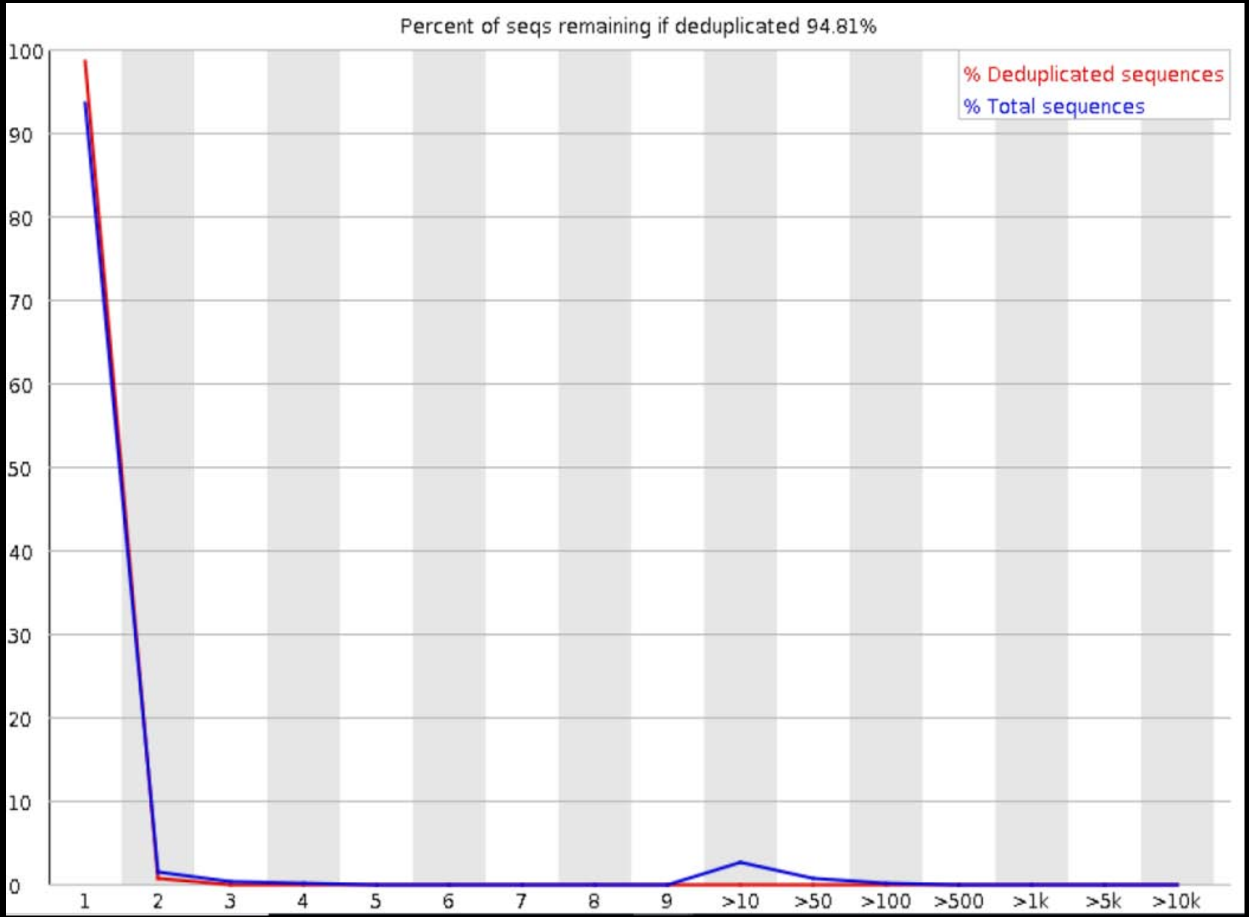




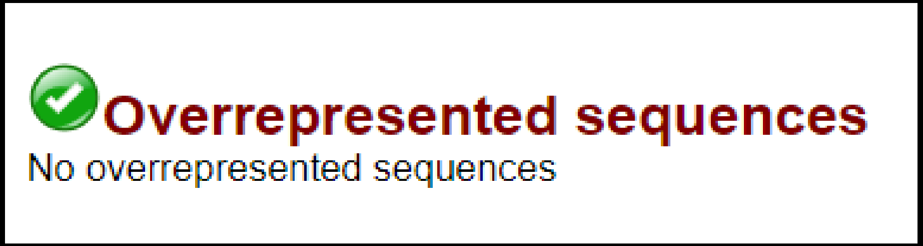
Duplicate Sequences

- 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

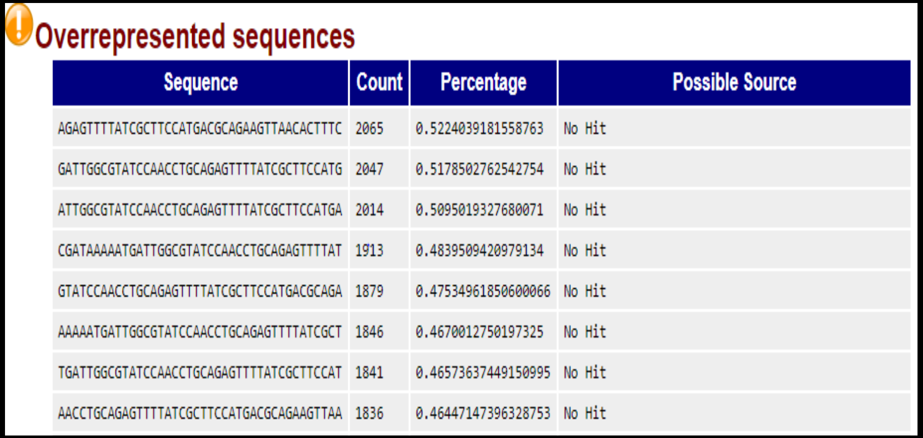




Overrepresented Sequences


Overrepresented sequences

No overrepresented sequences

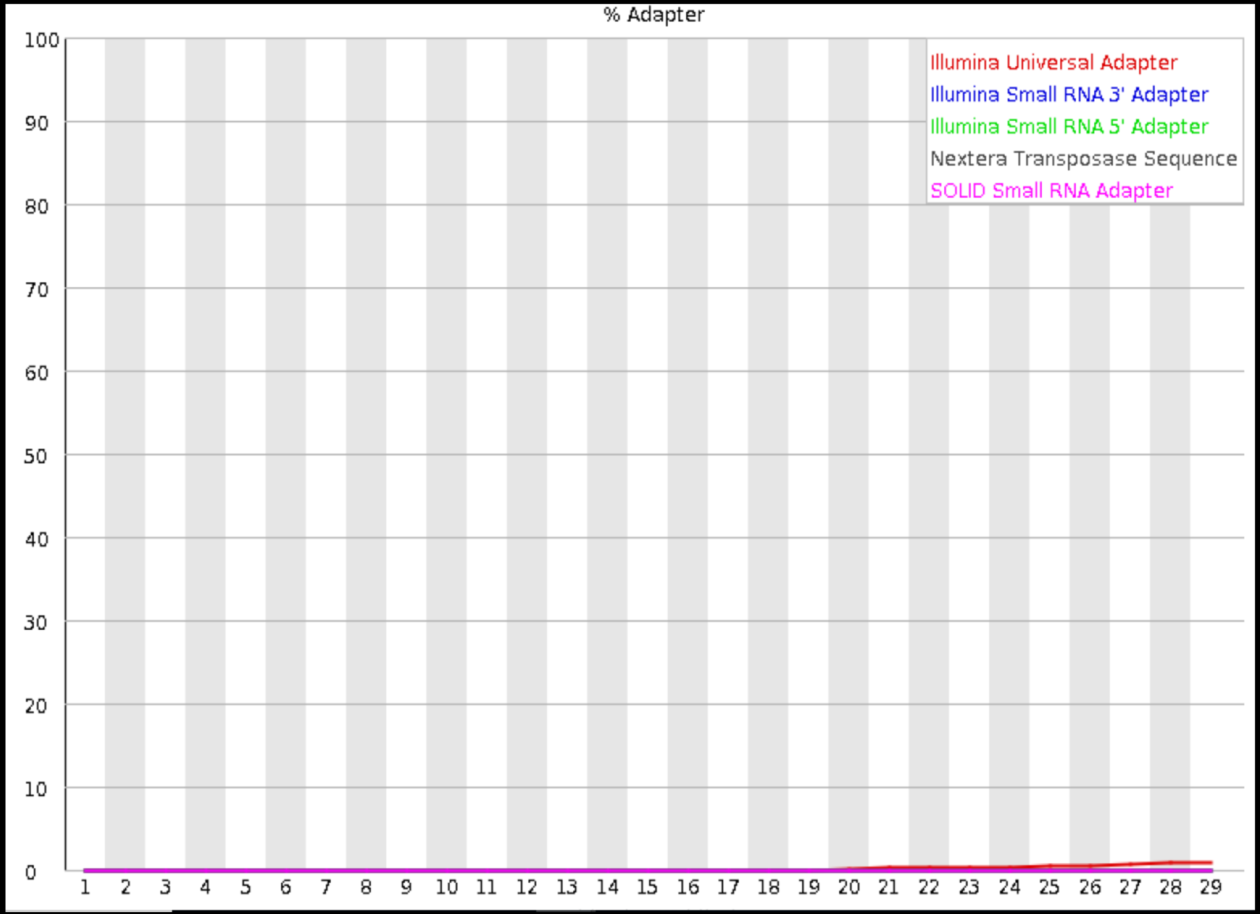

Overrepresented sequences

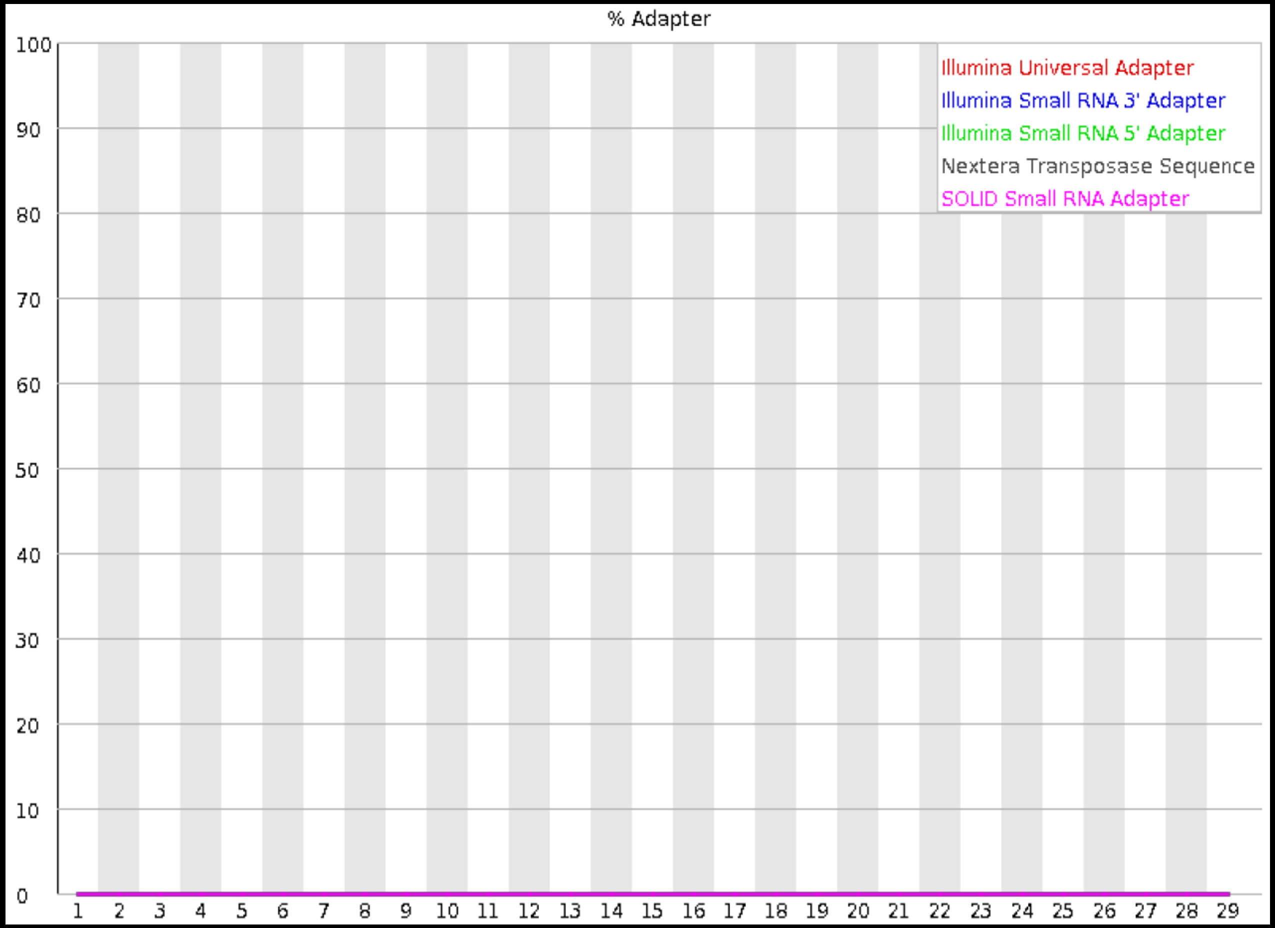
Sequence	Count	Percentage	Possible Source
AGAGTTTTATCGCTTCCATGACGCAGAAGTTAACACTTTC	2065	0.5224039181558763	No Hit
GATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATG	2047	0.5178502762542754	No Hit
ATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATGA	2014	0.5095019327680071	No Hit
CGATAAAAATGATTGGCGTATCCAACCTGCAGAGTTTTAT	1913	0.4839509420979134	No Hit
GTATCCAACCTGCAGAGTTTTATCGCTTCCATGACGCAGA	1879	0.47534961850600066	No Hit
AAAAATGATTGGCGTATCCAACCTGCAGAGTTTTATCGCT	1846	0.4670012750197325	No Hit
TGATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCAT	1841	0.46573637449150995	No Hit
AACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAA	1836	0.46447147396328753	No Hit

- Finding that a single sequence is very overrepresented in the set either means that it is highly biologically significant, or indicates that the library is contaminated, or not as diverse as you expected

Over represented Kmers

- The analysis of overrepresented sequences will spot an increase in any exactly duplicated sequences
- The Kmer module starts from the assumption that any small fragment of sequence should not have a positional bias in its apearance within a diverse library.
- This module measures the number of each 7-mer at each position in your library and then uses a binomial test to look for significant deviations from an even coverage at all positions.
- Any Kmers with positionally biased enrichment are reported.





Quality control checks on raw sequence data using FASTQC

- **Command line:**

  $FastQC_DIR/fastqc *.fq (The wildcard * is used for executing fastqc for all fastq files present in current folder)

- **Using Interface:**

  $FastQC_DIR/fastqc ---> Open ---> Browse your fastq file ----> Fastqc process ---> Analyse result

TRIMMING AND FILTERING

DATA PREPROCESSING

- Prior to doing anything with raw reads – mapping, clustering, assembly, etc – it is usually prudent to do certain preprocessing steps, many of them (like quality-trimming) are optional, so you don't have to necessarily do them.
  - Format conversion, if necessary. The simplest format for the subsequent steps is gzipped fastq, with the reads interleaved in a single file if they are paired, but that's not required. However, H5 and SRA formats are not supported, and unaligned bam should be converted to fastq first.
  - Adapter-trimming. Always recommended
  - If reads have an extra base at the end (like 2x151bp reads versus 2x150bp), it should be trimmed here with the "ftm=5" flag. That will occur before adapter-trimming.
  - Quality-trimming and/or quality-filtering. Optional; only recommended if you have very low-quality data or are doing something very sensitive to low-quality data, like calling very rare variants.
  - Deduplication. Optional; mainly for exome-capture. This is not actually part of RQCFilter because JGI does not typically do exon-capture
  - Normalization or subsampling. Optional; mainly for assembly of data with high or uneven coverage. Tool: BBNorm for normalization, Reformat for subsampling.
  - Error correction. Optional; requires adequate coverage. Paired-read merging. Optional; mainly for assembly, clustering, or insert-size calculation. Tool: BBMerge.
  - Kmer depth distribution. Optional; mainly for assembly and contamination detection.

TRIMMOMATIC

- Trimmomatic performs a variety of useful trimming tasks for illumina paired-end and single ended data.
- 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

Paired End EXAMPLE

- java -jar trimmomatic-0.30.jar PE s_1_1_sequence.txt.gz s_1_2 _sequence.txt.gz lane1_forward_paired.fq.gz lane1_forward_unpaired.fq.gz lane_reverse_paired.fq.gz lane1_reverse_unpaired.fq.gz ILLUMINACLIP:TruSeq3PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
- This will perform the following in this order
  - Remove Illumina adapters provided in the TruSeq3-PE.fa file (provided). Initially, Trimmomatic will look for seed matches (16 bases) allowing maximally 2 mismatches. These seeds will be extended and clipped if in the case of paired end reads a score of 30 is reached (about 50 bases), or in the case of single ended reads a score of 10, (about 17 bases).
  - Remove leading low quality or N bases (below quality 3)
  - Remove trailing low quality or N bases (below quality 3)
  - Scan the read with a 4-base wide sliding window, cutting when the average quality per base drops below 15
  - Drop reads which are less than 36 bases long after these steps

BBMap

- BBMap is a short read aligner, as well as various other bioinformatics tools.
- It is written in pure Java, can run on any platform, and has no dependencies other than Java being installed (compiled for Java 6 and higher).
- All tools are efficient and multithreaded.
- Some of the important tools related to quality trimming and filtering:
  - BBMap: Short read aligner for DNA and RNA-seq data. Capable of handling arbitrarily large genomes with millions of scaffolds. Handles Illumina, PacBio, 454, and other reads; very high sensitivity and tolerant of errors and numerous large indels. Very fast.
  - BBNorm: Kmer-based error-correction and normalization tool.
  - Dedupe: Simplifies assemblies by removing duplicate or contained subsequences that share a target percent identity.
  - Reformat: Reformats reads between fasta/fastq/scarf/fasta+qual/sam, interleaved/paired, and ASCII-33/64, at over 500 MB/s.
  - BBDuk: Filters, trims, or masks reads with kmer matches to an artifact/contaminant file.

BBMap-Features

- BB stands for Bestus Bioinformatics.
- Pure Java, runs on any platform; already compiled, just unzip and run.
- Fast, efficient, and multithreaded.
- Usage information displayed when running a shell script with no parameters.
- Highest sensitivity of any short-read aligner.
- Easy to install - just unzip/untar.
- Easy to use. Example: bbmap.sh ref=ecoli.fa in=reads.fq out=mapped.sam
- Handles all common formats: fasta, fastq, sam, scarf, fasta+qual, ASCII-33, ASCII-64, gzip.
- Used by the Joint Genome Institute.

Deduk

- Dedupe was written to eliminate duplicate contigs in assemblies, and later expanded to find all contained and overlapping sequences in a dataset, allowing a specified number of substitutions or edit distance.
  - Dedupe has 6 phases, most of which are optional and depend on the processing mode. They are always executed (or skipped) in the same order.
    1) Exact Matches: During this required phase, sequences are loaded into memory, and exact duplicates (including reverse-complements) are detected and discarded.
    2) Absorb Containments.
    If "absorbcontainments" is enabled (default), every read X is scanned for kmers; each kmer is looked up in a hashtable.
    3) Find Overlaps.
    If "findoverlaps" is enabled (non-default), overlaps will be sought using the same process as containment-absorbtion,
    4) Make Clusters.
    If "cluster" is enabled (non-default), clusters will be created by searching the overlap graph. Each cluster is the set of all reads reachable via transitive overlaps.
    5) Process Clusters.
    If "processclusters" is enabled (non-default), the clusters will be post processed to simplify them.
    6) Output

Bbduk

- "Duk" stands for Decontamination Using Kmers.
- BBDuk was developed to combine most common data-quality-related trimming, filtering, and masking operations into a single high-performance tool.
- It is capable of quality-trimming and filtering, adapter-trimming, contaminant-filtering via kmer matching, sequence masking, GC-filtering, length filtering, entropy-filtering, format conversion, histogram generation, subsampling, quality-score recalibration, kmer cardinality estimation, and various other operations in a single pass.
- Example:
- bbduk.sh in=reads.fq out=unmatched.fq outm=matched.fq ref=phix.fa k=31 hdist=1 stats=stats.txt
- This will remove all reads that have a 31-mer match to PhiX (a common Illumina spikein, which is included in /bbmap/resources/), allowing one mismatch

Filtering and trimming - commands

- /home/disc/NGSDAT-2019/19-03-19/bbmap/dedupe.sh in=C21_S36_L002_R1_001.fastq out=C21_S36_deduped1.fastq
- /home/disc/NGSDAT-2019/19-03-19/bbmap/dedupe.sh in=C21_S36_L002_R2_001.fastq out=C21_S36_deduped2.fastq
- /home/disc/NGSDAT-2019/19-03-19/bbmap/repair.sh in=C21_S36_deduped1.fastq in2=C21_S36_deduped2.fastq out=filter1.fastq out2=filter2.fastq outs=singleton.fastq
- /home/disc/NGSDAT-2019/19-03-19/bbmap/bbduk.sh in1=filter1.fastq in2=filter2.fastq out1=filtered1.fastq out2=filtered2.fastq k=12 ktrim=r ref=sequencing_artifacts.fa.gz,phix174_ill.ref.fa.gz
- java -jar /home/disc/NGSDAT-2019/19-03-19/Trimmomatic-0.36/trimmomatic-0.36.jar PE -phred33 filtered1.fastq filtered2.fastq output1.hq.fastq output1.unpaired.fastq output2.hq.fastq output2.unpaired.fastq LEADING:3 TRAILING:3 SLIDINGWINDOW:10:25 MINLEN:30

Thank You!