2 - FastQC and MultiQC

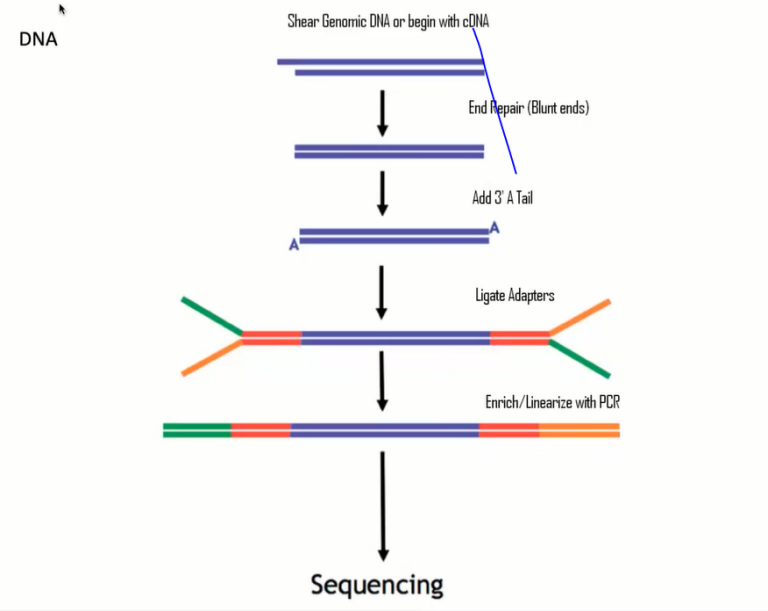
TOPICS	ASCII BulkRNA Seq Fastq files Phred score bcl2fastq quality control
DATE	@March 16, 2022 2:00 PM
LAST EDITED BY	
LAST EDITED TIME	@April 10, 2022 11:01 PM
MADE BY	
PROF	Calogero
Recording	lesson2.mp4
STATUS	☑

GitHub is an instrument to deposit workflow and instruments used in publications.

Data analysis of Bulk RNA-sequencing

This course is going to focus on the analysis of data, rather than how these are generated. The professor assumes that the students have a basic knowledge of how the data are acquired, i.e. RNA sequencing techniques.

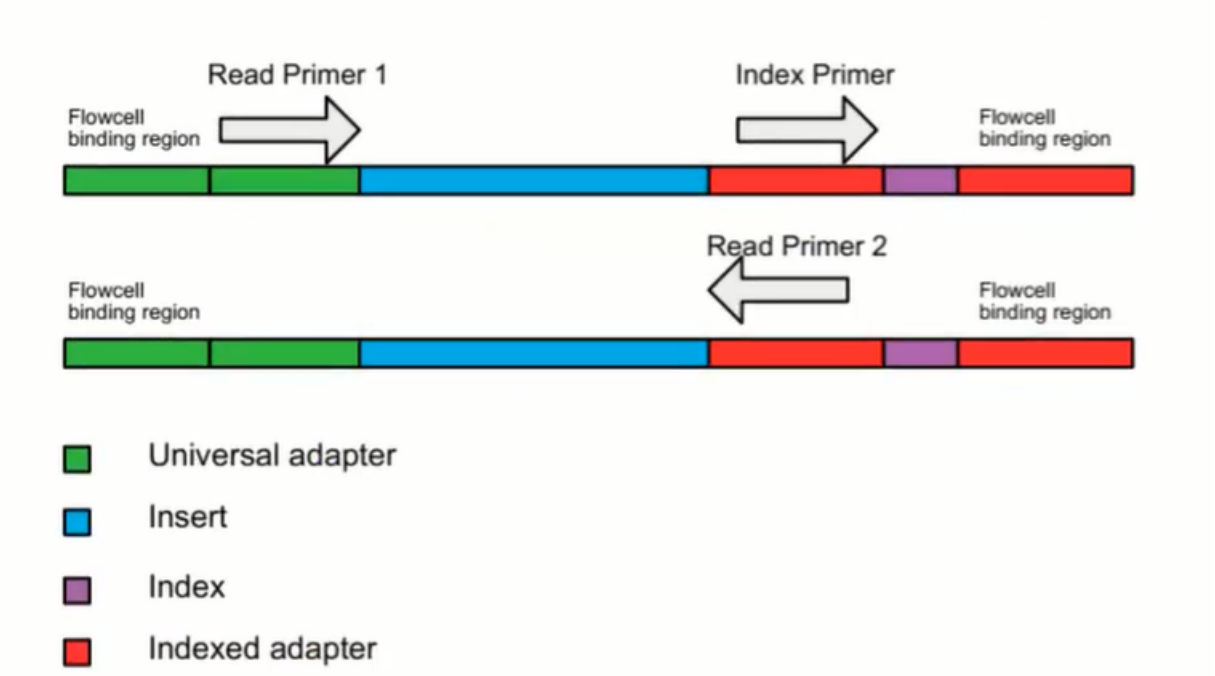
The minimum information required to understand data coming from bulk RNA sequencing experiments is summarised in the following picture. Bulk analysis refers to the simultaneous sequencing of a mixture of cells; instead, single cell sequencing is performed after the cells are separated from the bulk.



From the technical point of view, represented in the picture, a piece of cDNA is blunted, repaired and attached with primers through adapters (Y structured). Then, the cDNA strands are amplified with multiple reaction cycles to build a library of cDNA ready to be sequenced.

The blue cDNA sequence is unknown, derives from RNA and is enclosed in known sequences.

One important notion to remember is that if the RNA is sequenced with short read sequencing, the longest sequence obtainable is about 150-500 nucleotides long, depending on the fragmentation technique used. Indeed, the methods based on second generation sequencing require the fragmentation directly of the RNA or immediately after the cDNA sequencing; this means that there is no way to sequence a full transcript with these technologies.



Paired-end sequencing scheme

The structure of the library obtained has the **Read Primer 1**, which enables to amplify the first part of unknown region (in blue), while the **Read Primer 2** is used to read “upside down” the cDNA sequence.

The oligos are made in a way that Read1 is the primer for the forward read, whereas Read2 is used for the reverse read; this entails that one of the options of the sequencer is to choose between single end or double end/paired-end sequencing. The second option is preferred in DNA sequencing because in those analysis there is the need for sequence coverage, i.e. the need to know the exact content of the sequence.

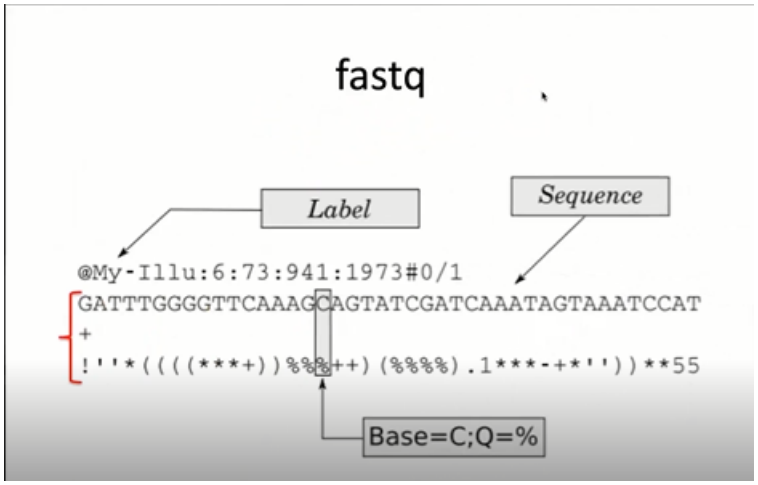
Today, the sequencing technique improved in sensitivity and single-end sequencing is more than enough to do gene level analysis; in this case, single end sequencing refers to around 75 nucleotide sequenced on one side of the unknown sequence. Normally, on the new platforms, around 150 nucleotides for one direction are sequenced with this method.

With 50-75 nucleotides are already enough to have an unique identification of the sequence on a unique transcript.

The **index primer** allows to sequence together more than one sample; when the samples are prepared, the adaptors ligated on the side of the index primer are chosen in a way that the sequence of the index primer is different for each sample.
Then all the sequences are analysed together, but, since there is the possibility to read the primer index, software that know which unique primer index is assigned to each sequence are capable to divide them.

With these kind of primers it is possible to sequence multiple samples all together and then divide the sequences coming from the same sample by using software. this is important for bulk sequencing

What are the sequences obtained with RNAseq?



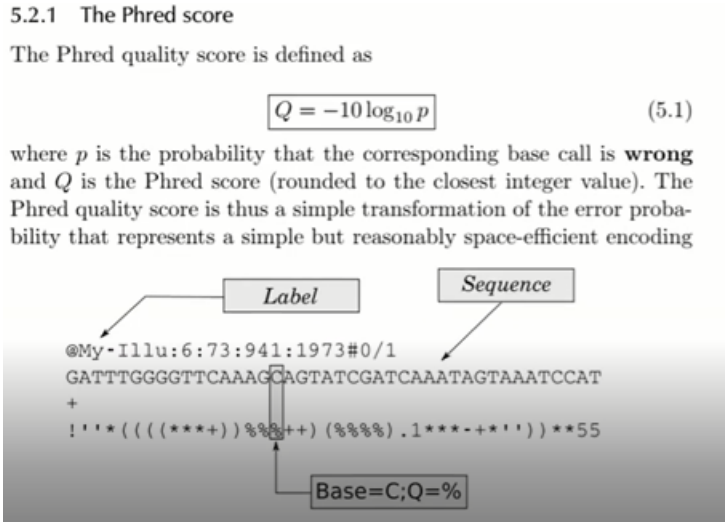
Normally these sequences are stored in FastQ files which are a modification of FastA file, which normally is a file that contains an header with some information and then the sequence.

FastQ files are used to store sequencing data; they have a header that starts with @, the sequence, a separator which normally is a “+” symbol and then quality information (Q).

The quality information are used by the aligner software to have a measure of the reliability of each nucleotide, i.e. if a position can be trusted during the alignment or not.

Quality refers to the fact that any time that a single nucleotide is sequenced, it is possible to generate the probability that the sequencing output is incorrect. The smallest the probability of error, the higher the quality.

The quality is encoded in ASCII code, which is nothing else that the characters that are present in the American typewriter machine. The ASCII code is read and used by the alignment program.



The quality score, also called the **Phred score**, is derivative from the Phred score firstly designed for sequencing the human genome with Sanger sequencing and then it was adapted to the latest sequencing approaches (NGS).

The quality score can be calculated with the following equation:

$$Phred\ score = -10\ log_{10}\ (p - Value)$$

The use of the logarithmic notation allows to shrink the information in a value with a little number of zeros enabling to have a better visualization of the number.

At the same time, representing the quality score with numbers has the issue that quality scores higher than 9 will need to use two digits to be represented, thus it won't be aligned perfectly with the corresponding nucleotide. Thus, the score is expressed in ASCII characters to maintain the co-linearity between the nucleotide sequence and the quality score.

Table 5.1. Base Quality and Accuracy.

Q_{Phred}	p	Accuracy
0	1	0%
10	10^{-1}	90%
20	10^{-2}	99%
30	10^{-3}	99.9%
40	10^{-4}	99.99%
50	10^{-5}	99.999%
60	10^{-6}	99.9999%
70	10^{-7}	99.99999%
80	10^{-8}	99.999999%
90	10^{-9}	99.9999999%
93	$10^{-9.3}$	99.99999995%

Note: Base quality Phred scores and their associated error probability (p) and base accuracy. A selection of values from the lowest (0) to the highest (93) Phred score representable in a FASTQ file is shown.

Today, the highest quality score obtainable is 40, which corresponds to an accuracy of 99.99%; nowadays, everything above 30 is considered a good quality score.

To have the co-linearity between the sequence and the Q score, ASCII Table from the typewriter are used to represent the score.

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

The first part of the typeable characters, from 0 to 32 (which is the **space**) on the first column from the left, are characters that cannot be normally written in a conventional ASCII terminal. So, if you use the line command and you type to see characters corresponding to numbers below 33, there is no output. This is because they are strange symbols that cannot be normally typed using a conventional keyboard.

In reality, very old ASCII tables, the characters between 0 and 32 are present; however, they are not typeable on a conventional line command approach.

At 33 there is the **! (exclamative point)** and at 73 there is **I**. Basically, in order to have a single character representing the data, i.e. the single base, the quality score has to be more than 33. So, the characters from 33 to 73 are representing the actual quality scores from 1 to 40, which are the ones normally assigned to a specific single nucleotide.

Normally, the first nucleotide incorporated has very low quality, then the quality is good and starts to decrease around more or less 100 nucleotides; after that the sequencing becomes less reliable.

As said before, the need for a one character score is due to the necessity to maintain the co-linearity between the nucleotide sequence and the scores.

The score is useful for the aligner to align the sequence to the genome or to the sequence of reference. The quality is used by the software to do the procedure of alignment.
For example, if the sequence matches perfectly one region of the genome, but there is one nucleotide completely wrong in the middle, before saying that the alignment is good but has one mismatch, it is necessary to look at the overall quality of the alignment.

The quality score is used by the software that associate the reads to specific positions in the genome during the alignment.

How are the Fastq sequences generated?

The sequences are generated by the **bcl2fastq (bcl-to-fastq)** software. The **bcl** are the image files that are generated by the sequencer; those files are first divided considering the index, thus divided by sample, and then for each of the different samples files that are associated are created.

```
<sample_name>_<barcode_sequence>_L<lane>\
  _R<read_number>_<set_number>.fastq.gz


NIST7035_TAAGGCGA_L001_R1_001.fastq.gz
```

Of course, the sequences obtained from the sequencing machine are random fragments coming from large number of sequences forming the bulk; thus, they are randomly positioned. The sample, in this case, is the bulk. to detect the genomic position encoding forn the specific gene
The sequencing will generated from 20 to 100 millions reads per sample.

In bulk RNAseq it is normal to work with at least 20mln reads per sample, which is enough to detect the genomic location encoding the transcripts associated to that specific gene; this is the aim of coding gene analysis. With this kind of analysis it is impossible to discriminate isoforms.

For non coding genes the number of reads have to be increased to 80-100mln, since the complexity of the non coding RNA is much greater than one of the coding RNA.
The amount of reads necessary for a particular analysis is connected to the amount of information to retrieve.

In other words, in order to get a reasonable representation of the sample, each sequence must be sampled an appropriate amount of times.

There are some kits that enable to do the analysis of coding and non coding genes together.

If, for example, 5mln reads are obtained from an experiment that analyses coding and non-coding genes, the information obtained will be related only to the more expressed genes. To get a reasonable representation of the analysed material, i.e. to have enough sampling of the sample, a minimal number of reads are required.

This issue is a coverage issue, meaning that a **minimal coverage** is necessary to have a significant result; in the last years, multiple studies have determined the minimum number of reads needed for different kind of analysis.

Another important point about RNAseq is that the absolute evaluation cannot be done, but instead the RNAseq analysis is based on comparative evaluation. depends on the efficiency of retro tramscript transbrine
This is due to the differential efficiency in retrotranscription in cDNA sequences for each mRNA transcript, which is dependent on the efficacy of the RT enzyme and of the type of fragment itself.

Indeed, it is impossible to tell that in the same sample GAPDH is more expressed than p53; this is because the fragments that have been generated have different sequences and thus different RT efficiency.

However, it is possible to say that the gene expression changes from sample1 to sample2 (same gene, same fragment, two different experiments). With this technology only the difference between two different samples/experiments can be told.

Sequencer output

oyu keep track on the lane at whihc is done the sequencing and you know where the problem is

When the file it's produced by the sequencer, the output contains:

1. <sample name> → e.g. NIST7035.
2. <barcode sequence> → e.g. TAAGGCA , it reports the index associated to the name.
3. L<XXX> → indicates in which lane of the sequencer the sequence was created; indeed, the sequences are analysed in different flow cells of the sequencer. Normally this parameter is used for quality control of the samples, i.e. to know from which lane the bad sequence comes from, and doesn't effect the downstream analysis. The surface of the sequencer is divided in multiple lanes.
4. R<X> → read number, which indicates if the sequence comes from **read1** or from **read2** (fwd and rv respectively). With R1 the mRNA is read from 5' to 3', while R2 is the reverse complement of the sequence of interest and is on the opposite strand of R1.
5. <set_number>

the two reverse complement

not for the user

For each piece of sequence generated, called **read**, there is a head that indicates the name of the read, followed by several characters that are used by the quality control instrument. The last character of the read label can be 1 or 2, for read1 and read2 respectively. The label starts with an @.

Read1 and Read2 of the same sequence have the same length and the same name, except for the last character. By doing so, the reads of the same sequence will be paired in the output file.

the most important law in Bioinformatics: GIGO -> garbage in , garbage out

Quality Control (QC) of raw data

In bioinformatics, it's essential to remember that is the data/sample is bad to start with, no bioinformatic analysis will be able to improve it. Analysing bad samples is a waste of resources.

One of the main criticalities is the experimental design. Informaticians typically do not take part in the design of the biological experiments, their role is to analyse the output data; however, only the informaticians know the limitations of the analytic tools and thus know what to avoid in order to produce data that can actually be analysed informatically.
In reality, they would like to be involved in the design process, before the experiment is carried out.

Basically, it is necessary to avoid the **fishing expeditions,** meaning building a dataset without following a specific biological question. For example, taking a bunch of patients and sequencing all of their genome: this is purposeless and only a waste of money.
Sequencing is not wrong per se, the error is not having a question to answer to.

The fishing expeditions are always a failure: every experiment must start with a precise question and the experiments have to be well designed for a specific purpose. Sometimes a lot of patients are sequenced because their genome is relevant for a specific diagnosis; but the analysis will be carried out by other researchers that will use those sequenced genomes as a database to answer a specific biological question.

Moreover, it is also possible that the dataset built to answer a specific question could be useful to answer other biological questions. But this requires reorganisation of the dataset.

Another important issue is homogeneity; it is important for the sample to be homogeneous, i.e. the number of control and treated should be similar; otherwise, the analysis will be biased. Balancing the experiment is important.

In the DNA data analysis the most important parameters: base quality, nucleotide distribution, GC content distribution, duplication rate, adapter sequence contamination

Also, the number of replicates in an experiment depends on the noise of the experiment; for example, if a cell line is studied, e.g. multiple experiments of cells treated or not after 5 passages, the noise will depend on the technical procedures (different positions in the incubator, different user...) and details of each experiment. However, since the cell line is kept the same, the noise should be very little.

This is not true when analysing patients; indeed, cell lines experiments have little noise compared to human experiment and in the latter it is impossible to know a priori the number of replicates necessary.
Normally. a pilot experiment is build, around 6-7 samples per condition is good for mice and less good for humans, then the statistical power of this experiment is estimated. Only then it is possible to determine if the cohort of samples should be increased or not.

Cell lines and humans experiments are at the two ends of the spectrum of number of replicates necessary, which always depends on the biological question. For example, if a behavioural diseases like autism is studied, a lot more patients should be used to have a statistically significant information compared to a antitumoral drug clinical trial.

An exception can be given with rare diseases, since it is difficult to have human replicates; however, it is possible to immortalize cells coming from the patients, thus replicates can be obtained. Those are not perfect replicates, but are a way to go around the issue.

The number of replicates also depends on the type of experiment and samples.

Lastly, in the experimental design it's important to remember that control and treatment samples should be treated in the same moment to ensure that the timeframe is not a confounding parameter during the analysis. Controls and treatments should be coupled together. experiment in the same day, together

✅ Be sure that the experiment is designed in a way that the data produced are not confounded by the noise that is around.

To evaluate the quality of the sequence, stored in the FastQ file, the software FastQC can be used. This is a generalistic Java software that works with RNA and DNA sequence.
Nowadays, the sequences are produced by hardcore facilities that are pretty reliable; nevertheless, this type of QC has to be done to check if everything is correct.

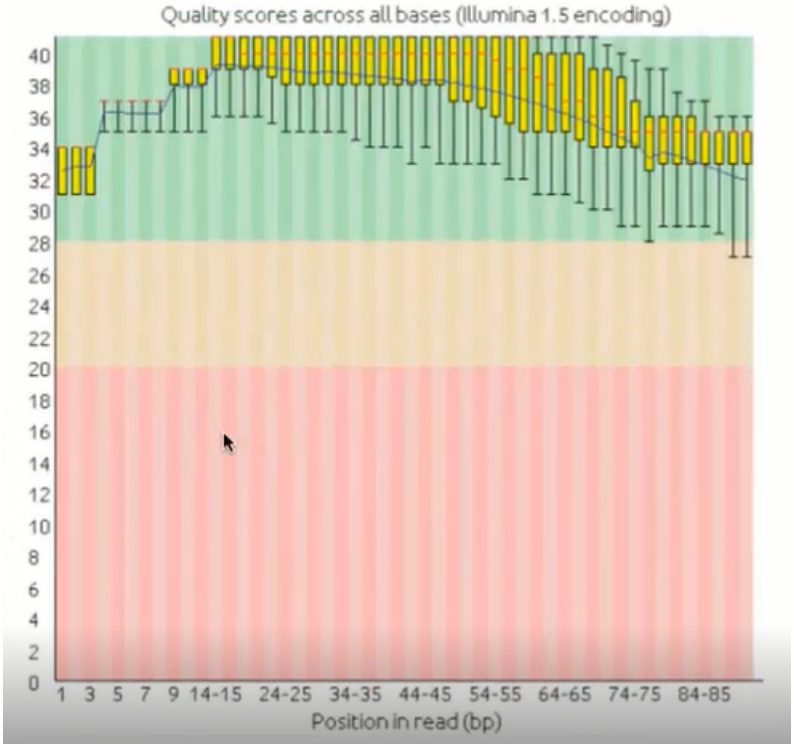
FastQC analyses the quality of the samples (RNA or DNA) through multiple parameters; however, the way the interpretation of the FastQC analysis is done depends a lot on the experiment considered.

Within QC, the first thing done is the evaluation of the quality of the sequence. To do that, there are a certain number of elements that the software utilises.

The quality of the reads, which is one of the most important parameters analysed, is estimated and the results are represented in the following plot:

there are experimetn gving strange results, you have to know if this come fromt he quality of your sample/examination
Maybe RNA quality is low or also the fastq need to be evaluated in this term
FasrQC aims to provide a simple way to do some quality control checks on raw sequence data coming from high throughput sequencing pipelines. It pro ides a modular set of analyses which can be use to give a quick impression of whether data has any problems of which user should be aware before doing any further analysis.
Main functions: -import data from FastQ files; - provide a auick overview to tell you in whihc areas there may be problems - Export of results to HTML report -Offline operations to generate reports

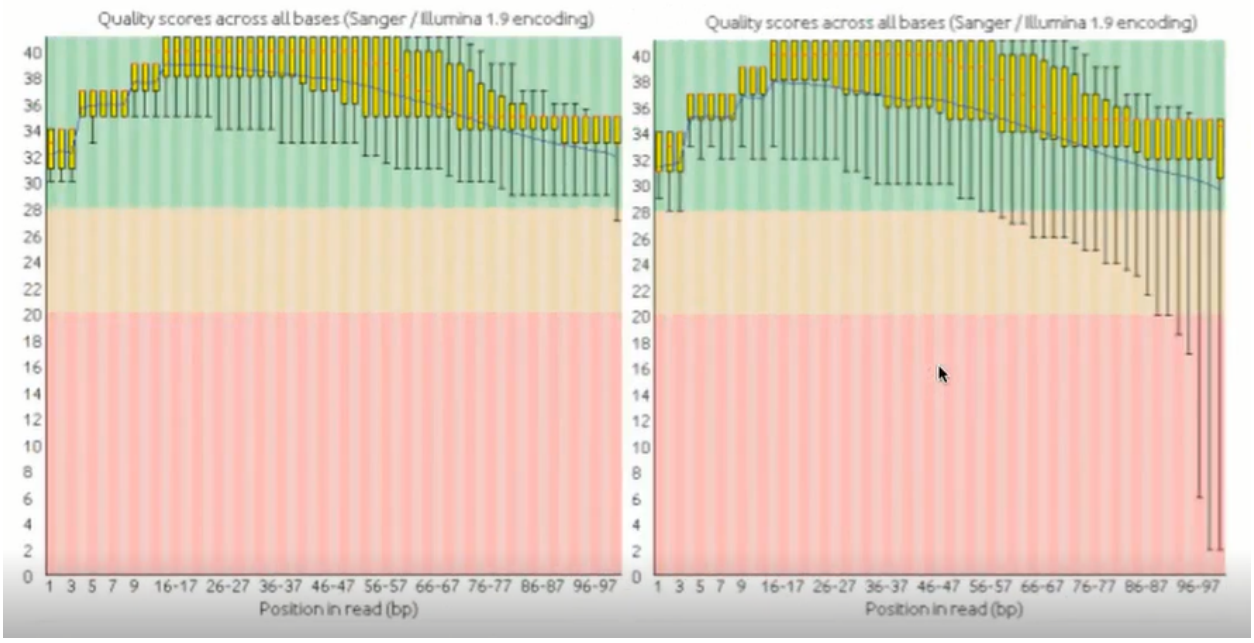
each info per base represents a sum of all the base of all the reads in that position



As seen in the plot, the quality score for each nucleotide addition for all the reads should always fall in the green region, meaning with a score above 28. Below 20, the sequence has some issues.

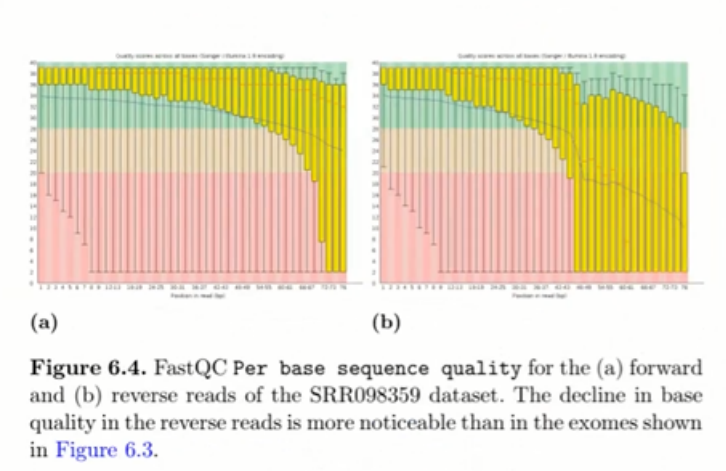
If 40mln reads have been sequenced, the plot will represent the dispersion of the 40mln first nucleotides incorporated during the sequencing.

baffi The whiskers coming out of the yellow boxes (which represent the quality distribution of the nucleotides added in that step) are not so critical; the important thing is that the mean stays above. in the green



forward reverse

In the plot above, the graph on the left represents the quality of the read1, while the right one refers to the quality of the read2. As seen, the quality of the second run starts to decrease slightly; however the decrease, due to the presence of outliers, is not so dramatic since the mean is always above 28.



(a) (b)

Figure 6.4. FastQC `Per base sequence quality` for the (a) forward and (b) reverse reads of the SRR098359 dataset. The decline in base quality in the reverse reads is more noticeable than in the exomes shown in Figure 6.3.

In the picture above, an experiment with poor results is reported, thus the experiment is limited by the quality of the sequence. However, there is a way to manipulate the data and remove all the reads of low quality; for example all the ones with a quality below 20. By setting a quality cut-off, some sequences may be lost, but the analysis is cleaned up.
Software can help to filter out the very low quality reads.

Another possibility is to analyse only a fraction of the sequence, e.g. stop the analysis at nucleotide 60. This procedure will only change the length of the sequences but not their overall number.

The choice between the two approaches is dependent on the experiment: if the aim of the analysis is gene detection, the length is not so critical, 50 nucleotides are more than enough. This is especially true for the R2 sequence, which normally has a lower quality than the R1; indeed, if R2 quality is low, it can just be ignored.

It’s important to consider also R2 if the interest is studying splicing events.

Overall, for gene analysis it is sufficient to look only at R1 with a sequence of around 50 nucleotides.

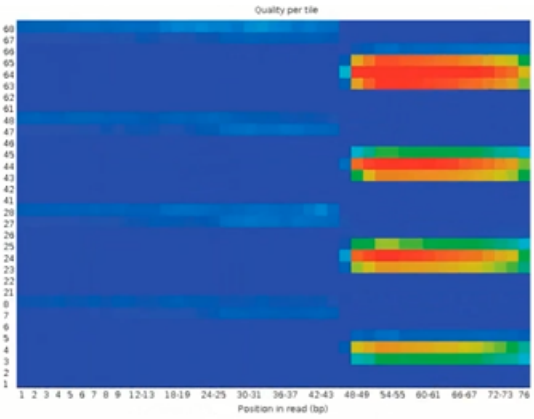


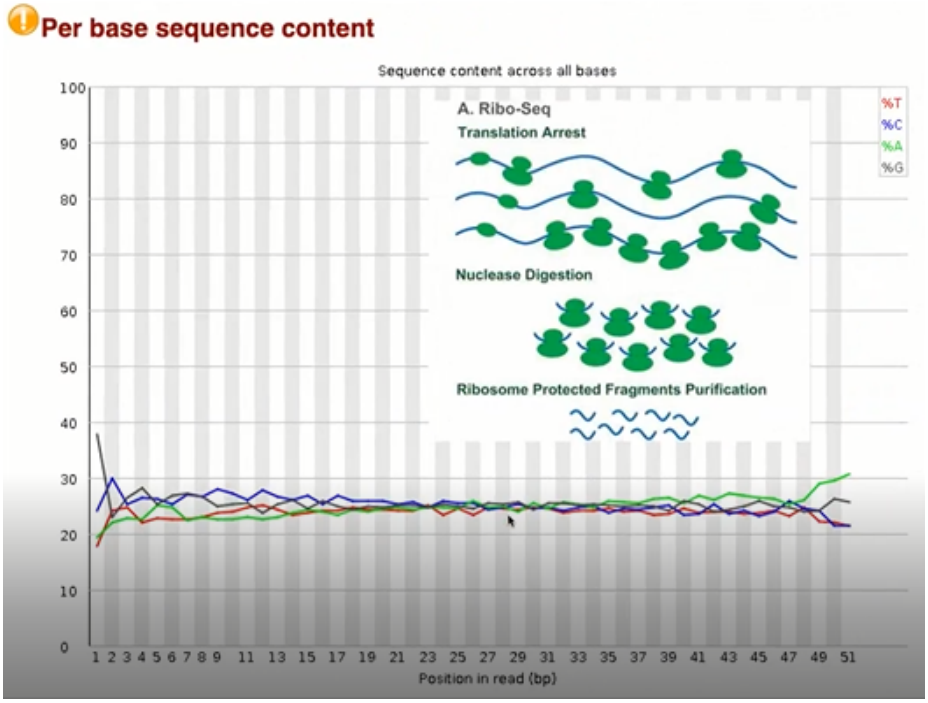
Figure 6.10. Per tile quality. FastQC `Per tile sequence quality` plot for the reverse read of sample SRR098359. The plot shows the deviation from the average quality for each tile, with hotter colors showing that a tile had worse qualities than other tiles for that base. A good plot should be blue all over. In this example, it is apparent that some tiles display poor overall quality.

Another graph created by the FastQC analysis is the **Per tile plot** which represents the change of the quality with the respect to the mean quality of the experiment; if the experiment has a good quality, the plot should be all blue, which indicates a low variation from the mean.
The red bands in the plot indicate that some of the lanes of the sequencer are behaving in a bad way. These results show that something happened during the sequencing related to the chemical reaction rather than the quality of the RNA sample; thus the problem is in the instrument itself. In this case you can complain with the provider of the flow cell.

Per base sequence content across all bases

Another output of the QC analysis is the **per base sequence content ratio** in the incorporation of the various nucleotides into the various sequencing points.
If we have a nucleotide distribution like the human one, where each nucleotide is equally represented, we should expect every nucleotide to have a distribution around the 25%. Despite this, some differences might arise due to technical properties of the sequencing kit we are using.
For example, the graph below is a sequence that comes from a procedure that is called Ribo-sequencing. To do this, you purify the polysomes, you cut away the RNA not covered by the ribosomes and you purify and analyse only the piece of sequence that was interacting with the ribosome. Those sequences are converted in DNA and then sequenced.

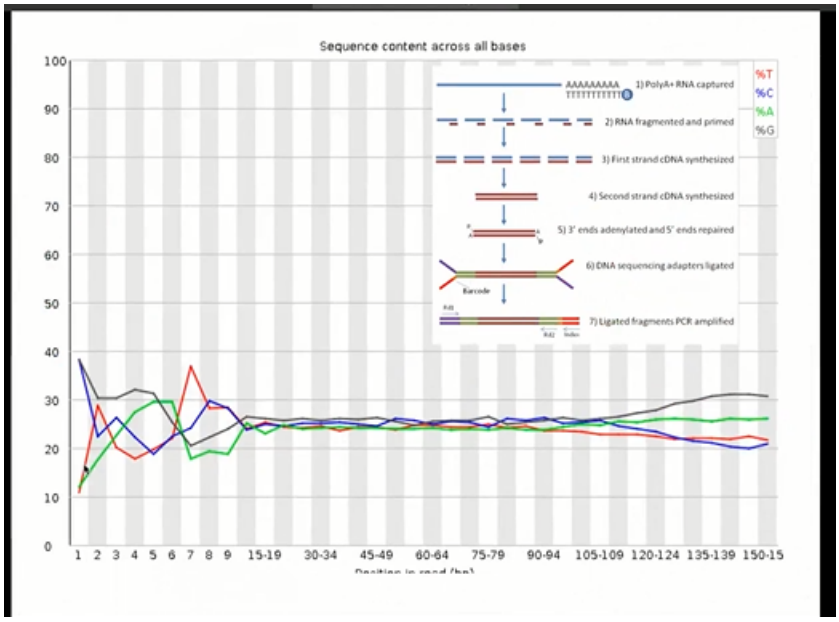
In the graph you can see that the overall distribution of the nucleotides, except for the first ones, is close to 25%.



Considering the RNA sequenced with a standard model kit using random priming represented below, there is a discrepancy in the distribution of the nucleotides in the first portions of the read.

This is due to the fact that not all the primer sequences (which are random hexamers) have the same efficiency in pairing to the target DNA.

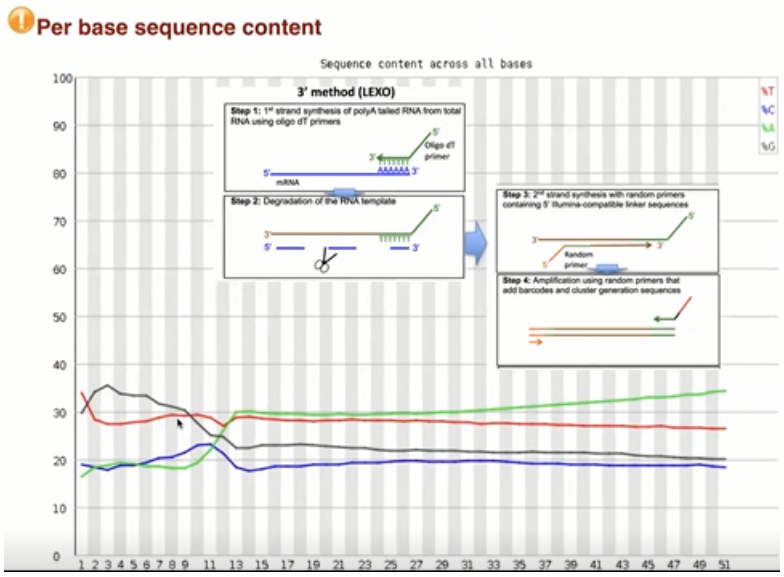
This results in the fact that some specific sequences will not be detected in a uniform way since they do not pair with primers in a strong way during the annealing.
If there is incoherence in the first part of the sequencing is not due to a biological problem and it’s not so critical; the only cause is that some random hexamers have a melting temperature very different from the annealing temperature used in the sequencer.



Lastly, below there is the output of one kit that only sequences the 3' end. This is mainly used in single-cell sequencing, but we will get into more detail later. In eukaryotes, though, the 3' end of transcripts is often bearing non-coding exons characterized by a different base distribution compared to the coding regions.

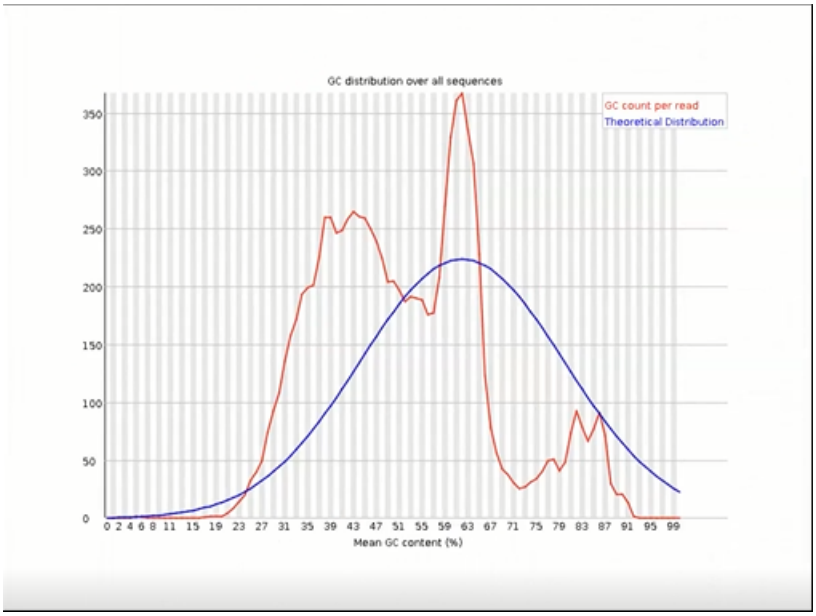
We have a lot of G and T at the beginning, more A's later on. This distribution is very different to the expected ratio; this is because we are looking at an uncoding sequence and not to a normal coding one.

because the sequencing has this beahviour in that region

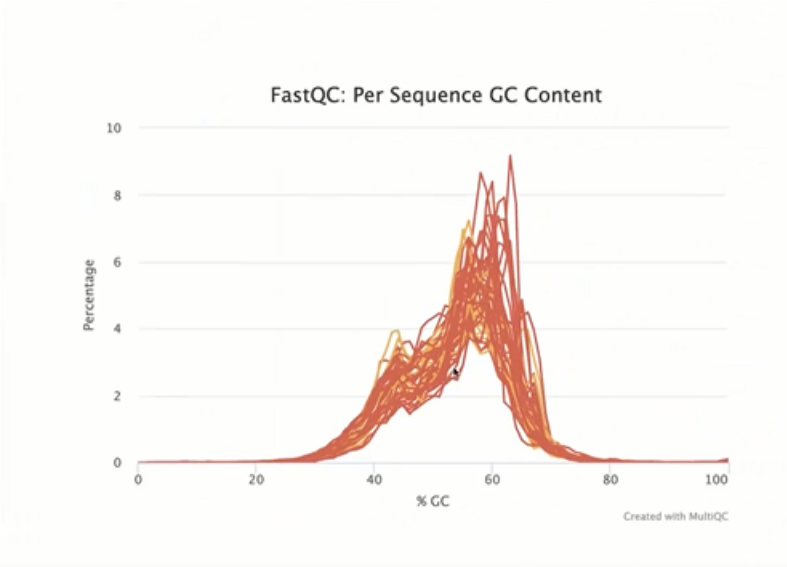


FastQC also provides the **distribution of expected GC content**; this usually not interesting in RNA sequencing since this calculation is done on genomic DNA. Whereas, in RNA sequencing, we are looking at a subpopulation of sequences that may therefore differ from what is expected (blue line).

Multiple peaks are not important in this case; however, this parameter is important with multiple experiment.



This is a technical type of information rather than a biological one: when running multiple experiments, though, it is important to have a look at the plot shape to make sure it is roughly the same among all samples, as a sort of homogeneity/coherence check.



✅ Remember that you have always to interpret the results considering the input data.

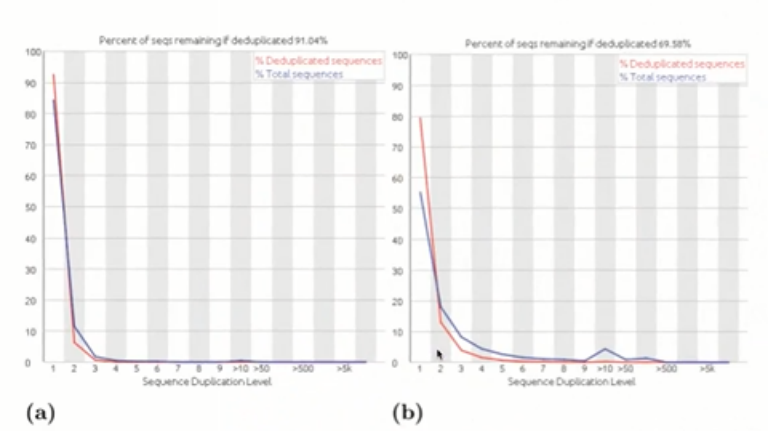
Sequence duplications

Another parameter used in FastQ is linked to sequence duplication.

Before sequencing, some PCR steps are done to get enough starting material.

between 12 to 15

Normally, around 12-15 cycles of amplification are suggested; out of this range, it is difficult to stay in the linear amplification range and maintain a kind of homogeneity among amplified and non amplified sequences. Out of the range, you lose the relative concentration of the various transcripts.



(a) (b)

Figure 6.7. Duplication level. FastQC plot of `Sequence Duplication Levels` in (a) the Corpas exome and (b) the reverse reads of the GIAB NA12878 exome dataset.

and that are not due to the representation of your gene

The problem is that some sequences can be artifactually overrepresented as a result of PCR, and this is of course an issue.

However, when we do bulk RNA seq, we don't have to pay attention to this phenomenon if the duplication structure/distribution is the same across all samples of an experiment. In this case, the technical error is equally spread across all the samples.

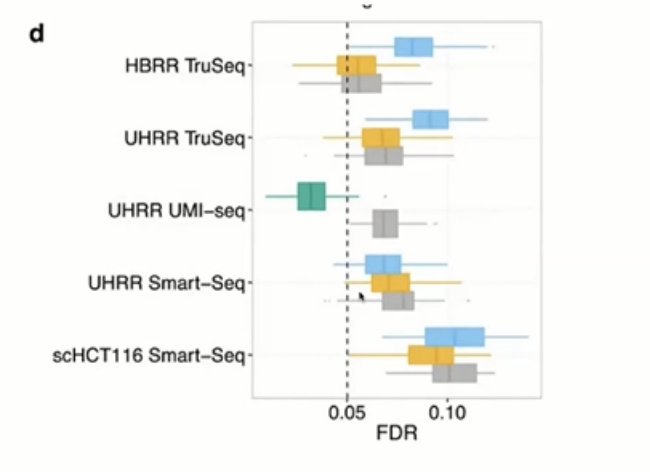
The only situation in which duplication must be removed in a way to not affect the quality of the experiment is when we do scRNA-seq. The removal of PCR artefacts is essential in the single cell RNA sequencing.

A typical thing done in bulk RNA analysis is to use **heat-ion fragmentation** with Mg and 65°C which can fragment the RNA into 150-base pieces in just 5 minutes; unfortunately, the fragmentation is not random and depends on the the secondary structure of the RNA.

Regions in secondary structures can be less sensitive to fragmentation. Thus, some artefacts that can be observed in PCR are not actual artefacts but are sequences that have not been fragmented by heat. This means that some regions will be overrepresented because of the way the fragments are created.

This doesn't apply to single cell data analysis.

so practiaclly i am dealing with multiple copies of the RNA



In the graph above is represented the output of the analysis of the differential expression data coming from different sequencing approaches after removing single reads or paired-ends duplicates in bulk or single cell analysis.

When the duplication artefacts are removed in the bulk analysis, the False Discovery Rate increases; the FDR indicates the number of false positives in the analysis.

contaminants

As displayed in the plot above, the False Discovery Rate (FDR, ability of having few false positives) becomes worse (increases) when removing sequence duplications; this always happens except for when we employ a specific sequencing approach, that is the one used in single cell sequencing.

duplications if distributed in all the experiment is not a problem

Adaptor Trimming

The process of fragmentation mentioned above may produce some fragments smaller than 150 bases. When these undergo sequencing, it may happen that some of the reads end with the 3' adapter sequence.

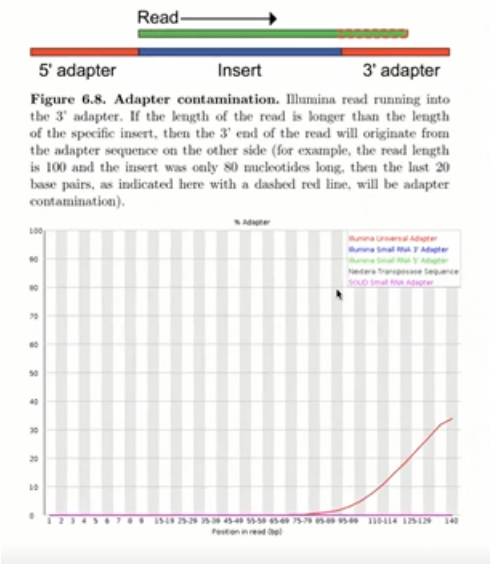
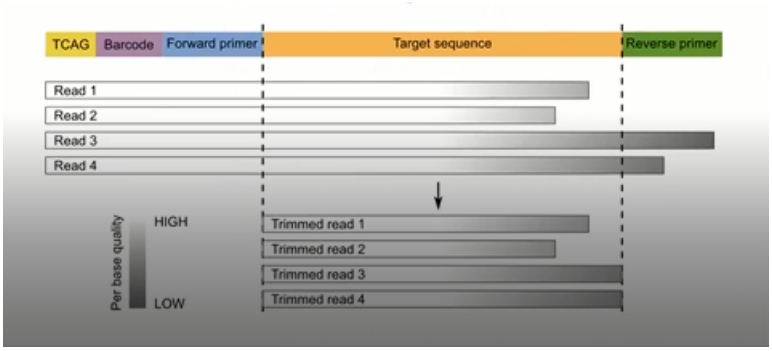


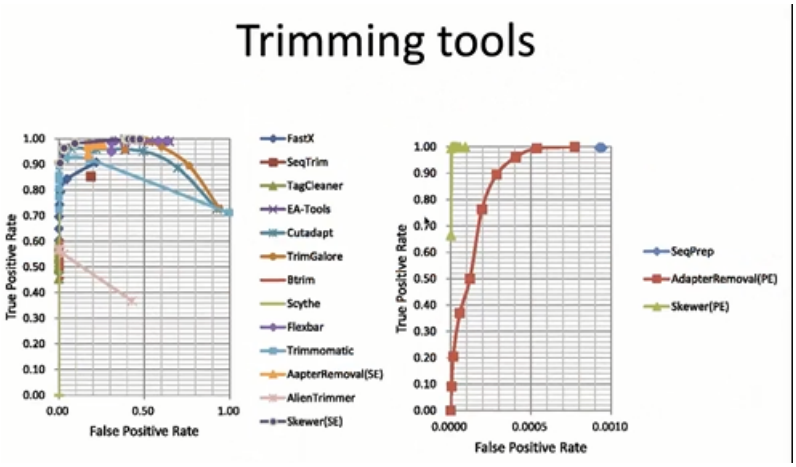
Figure 6.8. Adapter contamination. Illumina read running into the 3' adapter. If the length of the read is longer than the length of the specific insert, then the 3' end of the read will originate from the adapter sequence on the other side (for example, the read length is 100 and the insert was only 80 nucleotides long, then the last 20 base pairs, as indicated here with a dashed red line, will be adapter contamination).

The adaptor sequences present in the sequencing data can be removed with specific **trimming tools** that also allow to remove sequences with a quality score lower than a set threshold (this may also result in the

removal of low quality sequences which are not adapters).



For example, **Skewer** is a trimming tool that enables single end and also paired-end data trimming.



Bulk vs Single cell library structure

In bulk libraries, as described earlier in the lecture, the unknown sequence is in between the p7 and p5 adapters. Within the adapters, 6 nucleotide long index sequences are found, with the purpose of discriminating the sample of origin of the sequence.

Double indexing is now used (one index sequence within both adapters) since recent sequencing machines allow to simultaneously load a very large number of samples and therefore only one index sequence was not enough anymore.

When we look at the library structure for scRNAseq, the situation is a bit different. Let's consider as an example the 10x Genomics structure, since this is the main platform in this field.

10x Genomics in scRNAseq → Barcoding

Nowadays, the main player of sequencing is Illumina; 99% of the bulk sequences generated worldwide are obtained on Illumina platforms.

Read2 is the one that will sequence the unknown piece of cDNA, thus it tells you which transcript you are analysing.

tells you the RNA with which you are actually working

how does it work: if i have detected thta my gene is p53, i look at the 10x and there is the barcode for the cell “A” and then looking at the UMI , this should be always different. if there are more representation these are PCRartifcats. So 1 identifes the cell and the other the number of trnscript for each cell

Read1 is mainly used to discriminate between 2 things: *(this will be explained in far more details in the lesson number 4).*

- the 10x Barcodes that allow to discriminate the cell from which the sequences are coming.
- UMI (unique molecular identifier) is a 6-base random sequence unique for each starting transcript in PCR. It allows to tell the actual number of transcripts from a single cell.
  The UMI should be always different if it comes from different sequences (even if of the same cDNA sequence) originating from the same cell.

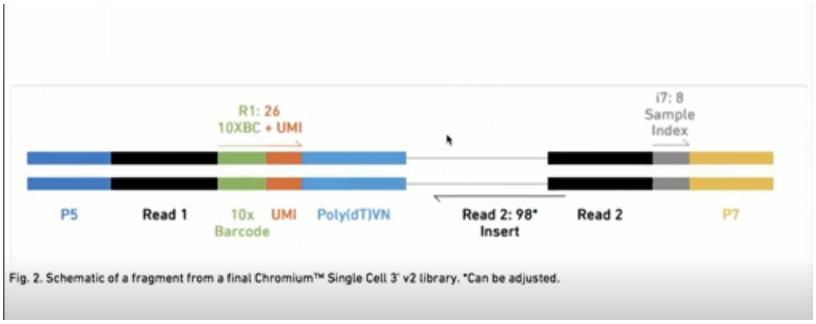


Fig. 2. Schematic of a fragment from a final Chromium™ Single Cell 3' v2 library. *Can be adjusted.

Sometimes the problem is the quality of the RNA, not the quality of the sequencing.

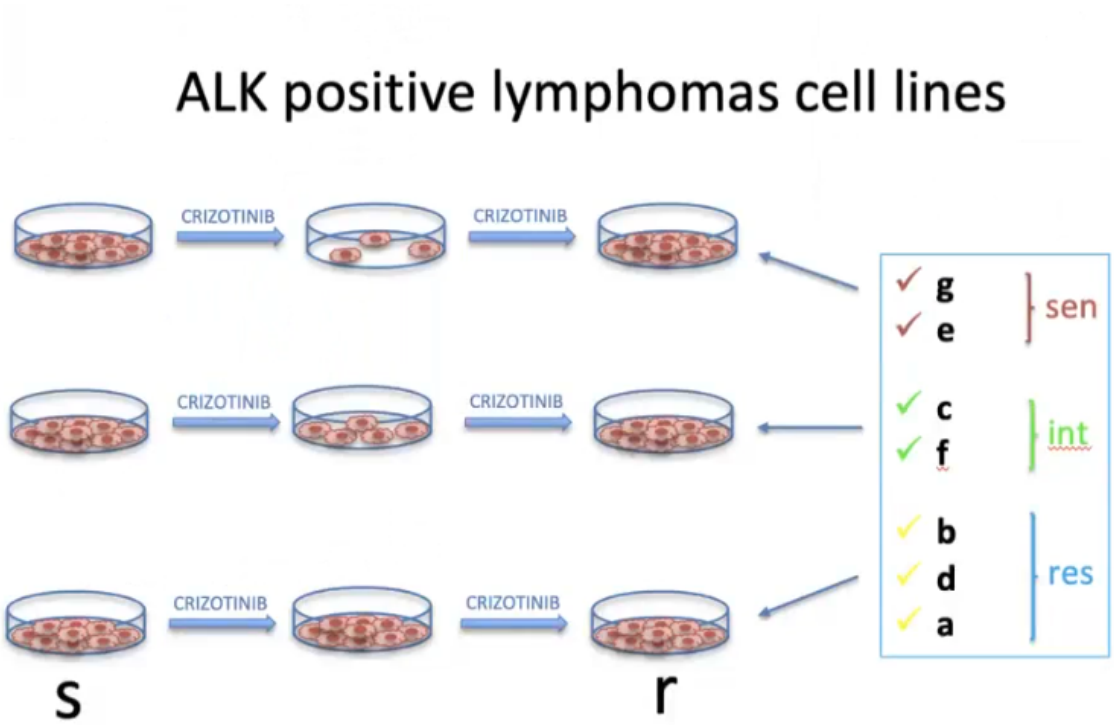
Normally, the quality of the RNA samples for the scRNA seq is not checked; this is because, the scRNAseq is trickier from a technical point of view, thus if the RNA quality is low, it will largely affect the sequencing analysis. This is not always true with bulk RNA seq.

Nevertheless, it’s always a good practice to check the quality also for thse kind of experiments.
come quello di prima con le tre fasce verde rosa rosso

EXERCISES

1. Search for FastQC on Google and download it. FastQC is a Java tool and has to be run by line command. You need to know the path to FastQC and of the files you are going to use.
2. Dataset 0 for FastQC → generates a html file that can be opened with the browser.
3. Dataset 1 is the real dataset that will be used throughout the course to perform several data analysis steps. It derives from a real experiment with 14 samples → multiQC analysis.

The experiment from which datasets have been generated is about ALK positive lymphoma cell lines. These cells, once treated with crizotinib they can be: sensitive (g,e) intermediate (g,f), resistant (b,d.a)
inhibitor of ALK



We have R and S samples, i.e. treated and untreated, for each cell line (14 samples in total). What we want to understand is **which genes are affected by crizotinib treatment**.

These are the step of the analysis:

fastQC is only a quality control but in reality when you do the analysis it is only the starting point.

1. Get a fastQ file.
2. Perform quality control with FastQC.
3. Trimming to remove the adaptors.
4. Alignment to the genome of reference. i take my sequence, i take my genome and i see where the sequence map
5. Annotating with the help of a .gtf file that has the position on the genome of each known gene.
6. Counting → how many sequences drop on a gene over the genes that are annotated on the .gtf file → we get counts for each gene for each of the samples. At the end, the counts file will be a table with:

   14 columns → each sample

   20000 rows → number of genes

so now since i know also the position of the genes i can associat e the two nformations

fastQC doen not provide all, you gain after by multiQC

Trimming and alignment quality can be checked by multiQC to provide an aggregated view of all samples.

MultiQC is a Python program and can be installed locally on the PC (not guaranteed that the Linux version is going to work on Windows) or through docker.

we are intereseted in how many sequence drop in the specific gene and not the sequence itself

Docker

Docker is a box that can work in different environments. We can use Docker to execute multiQC once is packed inside a Docker container.

Our task is to generate the multiQC results and we will look at them together next time.

Docker commands (to use in the terminal)

`docker ps` → tells us if there is any docker running at the **present time**. When a docker image is running, it's called docker container and it has a specific ID associated to it.

Anytime a docker is being run, if it's not located on the PC, it will be downloaded from the internet. So what usually happens is that we give an image to Docker and if that isn't found in the PC, the image gets downloaded from an online repository.

`docker images` → shows the docker images located on the computer

`docker ps -a` → shows the docker that have been run and the ID of the specific container. If we know the ID for the container that has been run: `docker logs` gives info about what happened during the execution of the container.

`docker logs` → we can use this command when a docker is running in background to see the logs coming from that container

`docker run` → use this to run a docker locally

`docker run -t` → makes the execution **interactive**. This means that whatever is happening is shown on the screen in real time.

`docker run -v` → creates a **connection** between our hardware and docker by mounting the folder in our PC (where the data is located) to the docker folder. Without this command, docker, which is an external entity in our computer, would not be able to see/access the folders in the computer itself.

`docker run -w` → indicates the directory in which the docker has to be executed (so where we will find the results).

Steps of the MultiQC exercise

1. `cd <name folder>` → go to the folder where the data is located
2. `pwd` → prints the working directory
3. `docker run -t -v "pwd":"pwd"` (you can write also the full path instead of pwd) what we type before the ":" is the folder in our PC containing the data. What we type after the ":" will be the name chosen for the folder **in the docker**
   `docker run -t -v "pwd":"pwd" -w "pwd"`

⇒ so now we can execute the multiQC program in the docker without having to install it locally.

On Windows PCs, in order for Linux commands (e.g., `pwd`) to work, the WSL infrastructure (Linux for Windows) has to be installed. Another option would be to install Python on Windows locally.