

Lecture 2

Intro to Data processing:

From bcl to count matrix

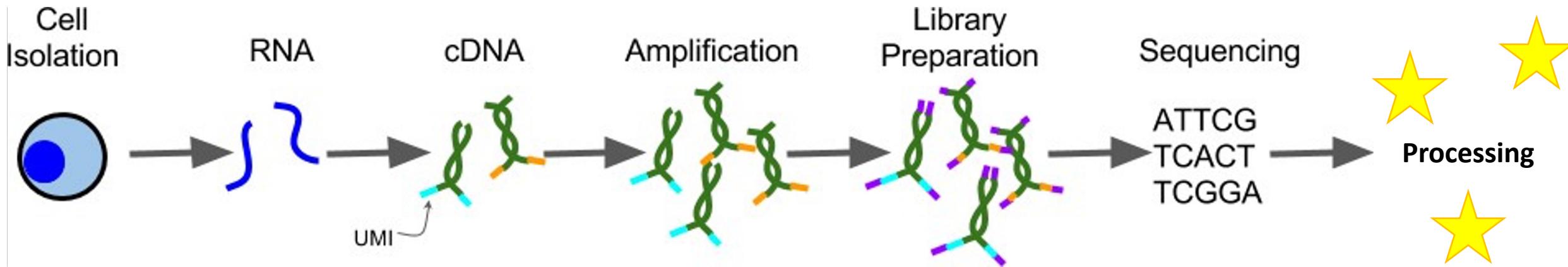
Physalia course 2025

—

Single-cell RNA-seq with R/Bioconductor

Instructors: Orr Ashenberg, Jacques Serizay, Fabrício Almeida-Silva

Experimental pipeline

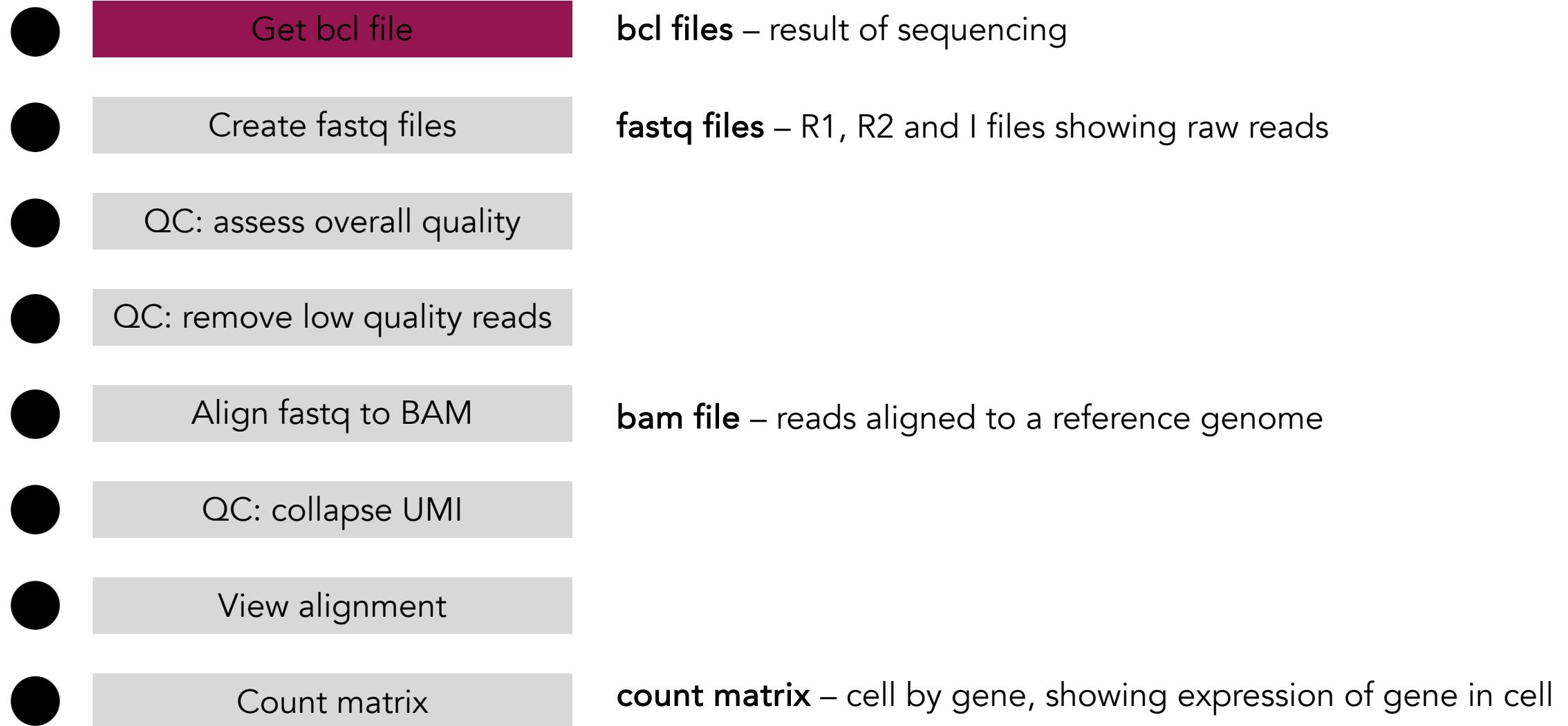


Regardless of who is going to do the experimental steps, **DISCUSS WITH THEM !!!**

Experimental design is crucial for the success of a single-cell project !!!

Systematic comparison of single-cell and single-nucleus RNA-sequencing methods, Ding et al., Nat. Biotech. 2020

Flowchart: from .bcl to count matrix



Flowchart: from .bcl to count matrix



Get bcl file



Create fastq files



QC: assess overall quality



QC: remove low quality reads



Align fastq to BAM



QC: collapse UMI



View alignment



Count matrix

.bcl files

.bcl:

- **Raw data** output of a sequencing run
- Binary, non-human-readable file
- Contains the base calling and quality score per cluster per sequencing lane

Flowchart: from .bcl to count matrix



Get bcl file

That is the role of the sequencing machine!



Create fastq files



QC: assess overall quality



QC: remove low quality reads



Align fastq to BAM



QC: collapse UMI



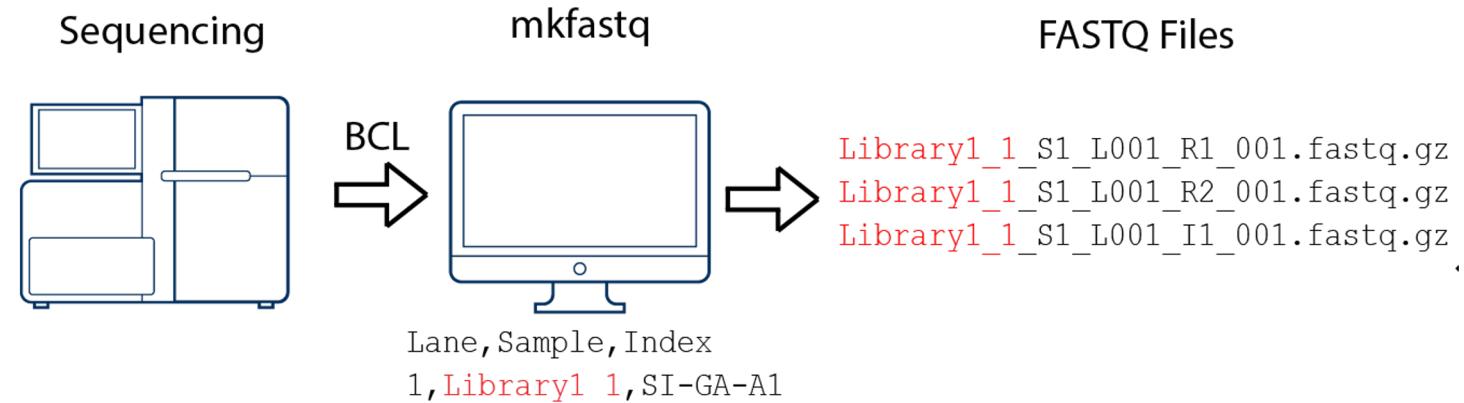
View alignment



Count matrix

Command:

```
bcl2fastq --run-folder-dir <bcl_files_folder> -p 12 --output-dir <fastq_files_folder>
```

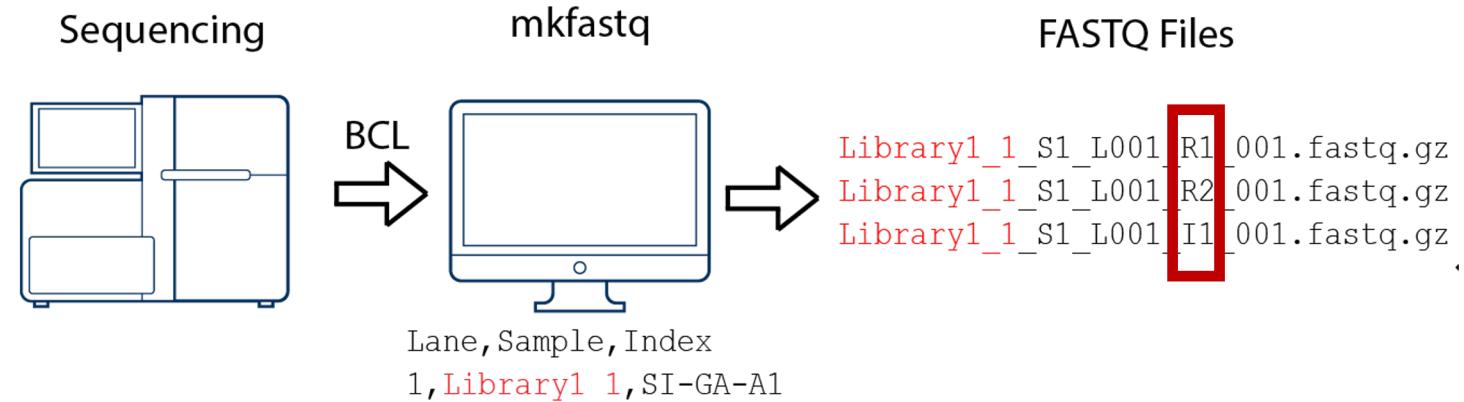


User guide:

https://support.illumina.com/content/dam/illumina-support/documents/documentation/software_documentation/bcl2fastq/bcl2fastq_letterbooklet_15038058brpmi.pdf

Command:

```
bcl2fastq --run-folder-dir <bcl_files_folder> -p 12 --output-dir <fastq_files_folder>
```



User guide:

https://support.illumina.com/content/dam/illumina-support/documents/documentation/software_documentation/bcl2fastq/bcl2fastq_letterbooklet_15038058_brpmi.pdf

How is a .fastq organized?

Each fastq file contains reads, each read is composed of 4 lines:

1. A sequence identifier with information about the sequencing run
2. The sequence (the base calls; A, C, T, G and N).
3. A separator, which is simply a plus (+) sign.
4. The base call quality scores, using ASCII characters to represent the numerical quality scores.

```
►jacquesserizay@LOCAL[12:46:19]:~ $ cat SRR11575369_1.fastq.gz | zcat | head -n 8
@SRR11575369.1 1/1
ANCAACAGTGGATTGTTGATGAAAAAAATAAATTGTTCTCAAAGCAGAGTGAATGATGCAGTACGAGCTCTGCTCTGAAAACCCATCACAACTTATAATTAAATAATTAGTGAAAATTAAAAAAATAATTCTTATATT
+
F#F:FFFFF:FFFFFFFFF:F.FF.FF::F.:FF:FFFFFFFFF:FF:FF:F.FFF.F:FF:FF:FFF:FF:FFFF.:F::FFFF:FF.FF:F:FF.:F:F:::F:F:FF.::::FF:..F.....:FFFF::F:...FF
@SRR11575369.2 2/1
TNGCCAGTCATAACGCCCTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTGTGTTTTTTTTTATATAATAAAATTTTTTTTAATAAAATTTTTTTTATT
+
F#FFFFFFF::FF:FF:::FFFFFFFFFFFFFFF:FFFFFFFFF,,FFFFFFFFF:FFFFFF:FFFFFFFFF:F,FFFFF,,,,:,:,:::FFFFFFF,,::::,F,,,,,,FFFFF,F,,FF:,F,:,F,FFFFF,,,:,F::F
```

Why do we end up with so many fastq files?

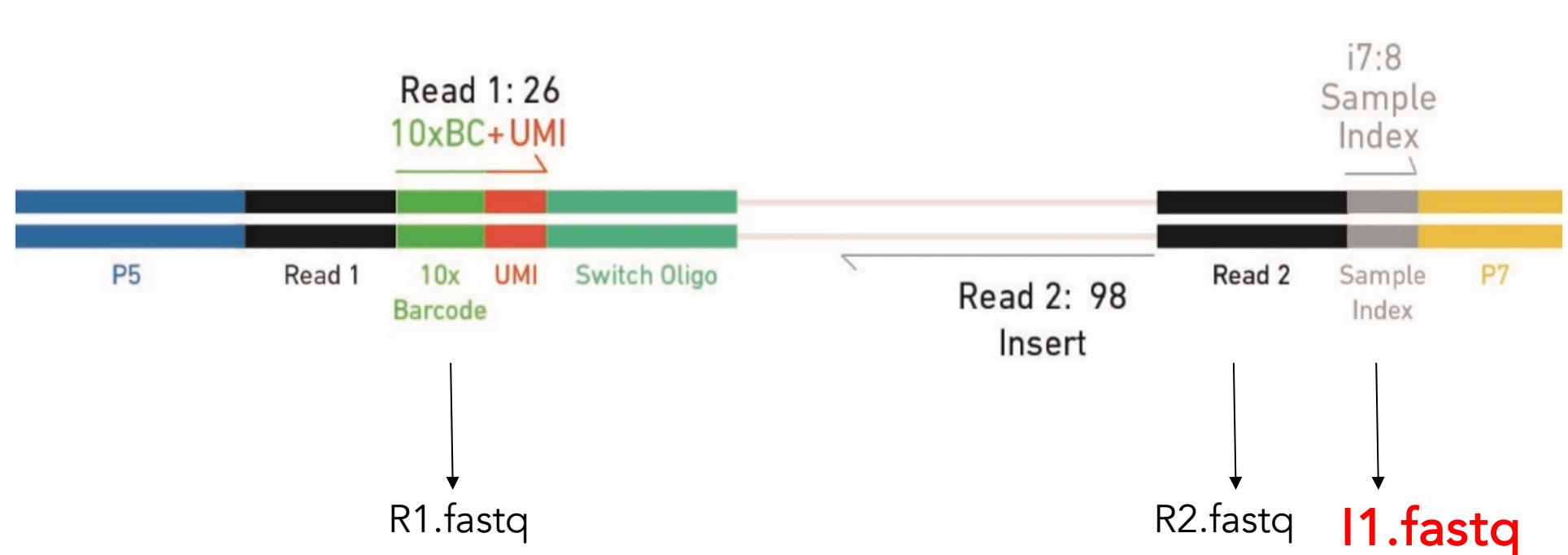
We sequence the paired ends of each DNA fragment molecule, in 3 different sequencing “runs”.



I1 is important for demultiplexing multiple samples simultaneously sequenced

Each fastq generated by **bcl2fastq** contains a different information:

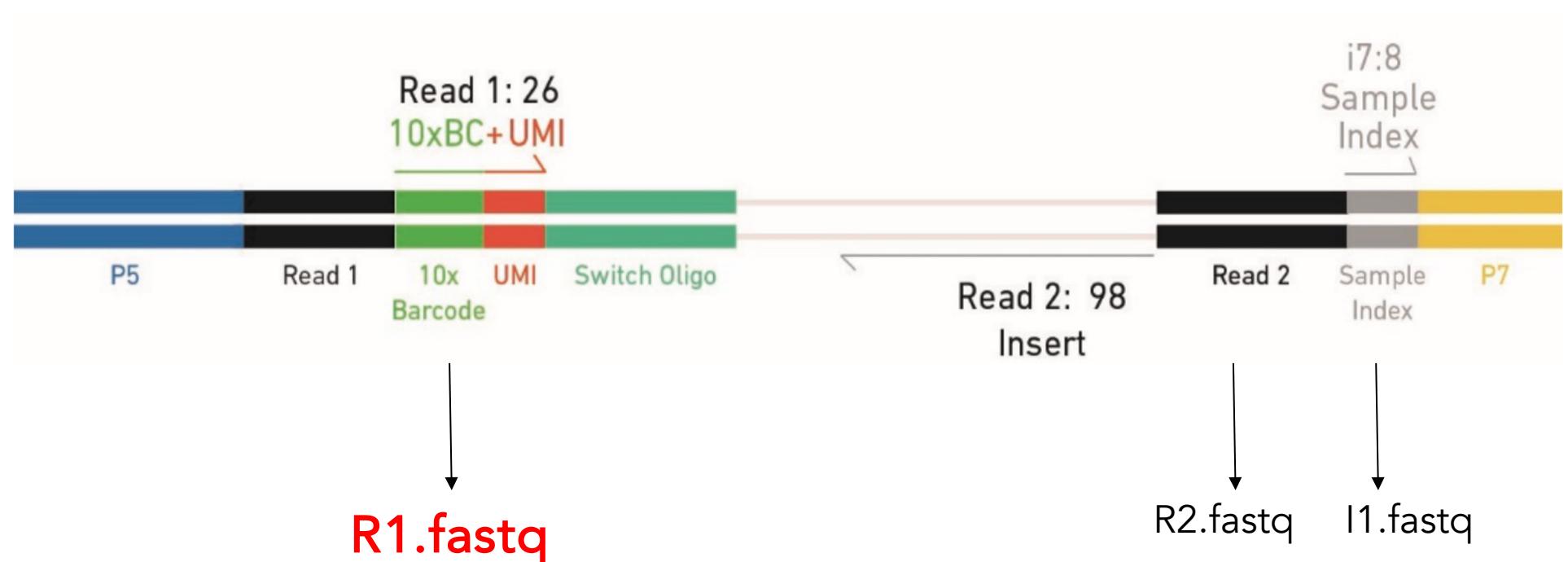
- I1.fastq contains sample index
- R1.fastq contains cell barcode + UMI
- R2.fastq contains transcript information



R1 contains information on the cell of origin (as well as a UMI)

Each fastq generated by **bcl2fastq** contains a different information:

- I1.fastq contains sample index
- R1.fastq contains cell barcode + UMI
- R2.fatsq contains transcript information



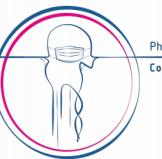
R2 contains sequence of the DNA fragment captured by sequencing

Each fastq generated by **bcl2fastq** contains a different information:

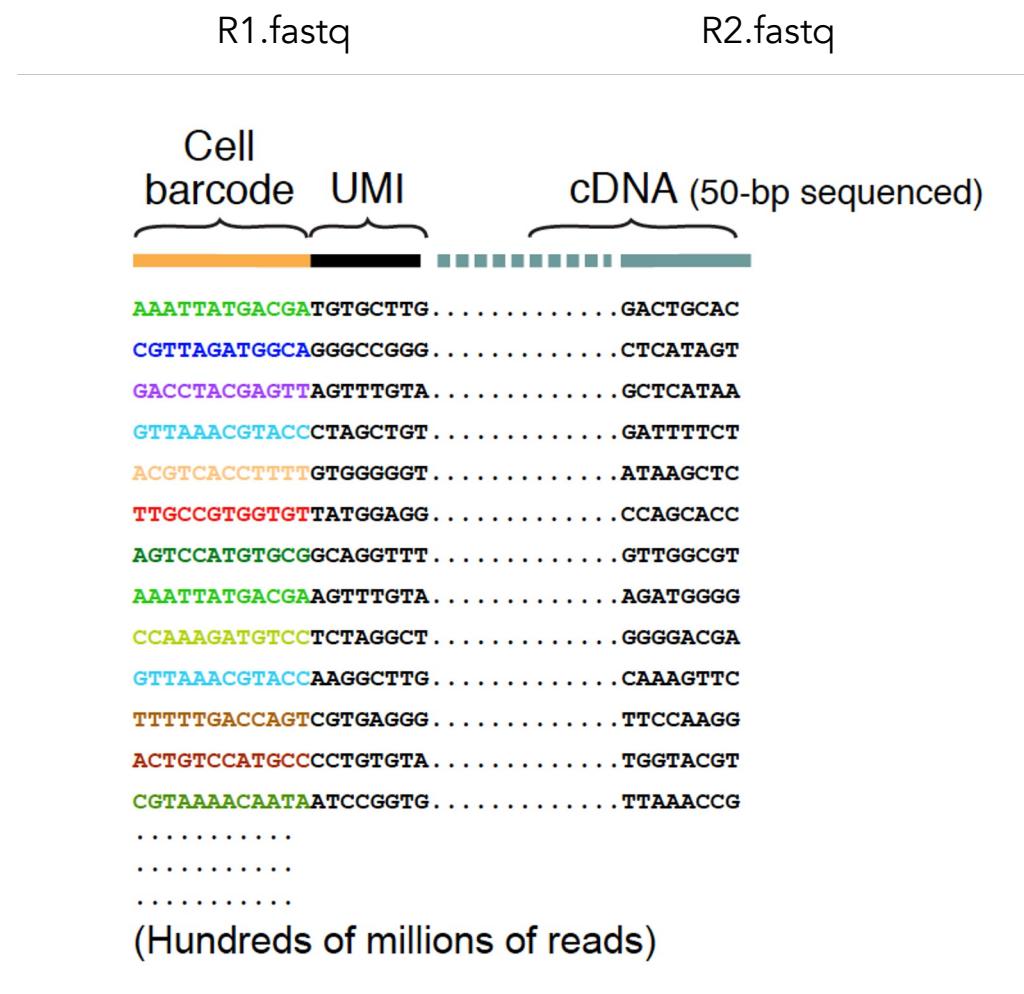
- I1.fastq contains sample index
- R1.fastq contains cell barcode + UMI
- R2.fastq contains transcript information



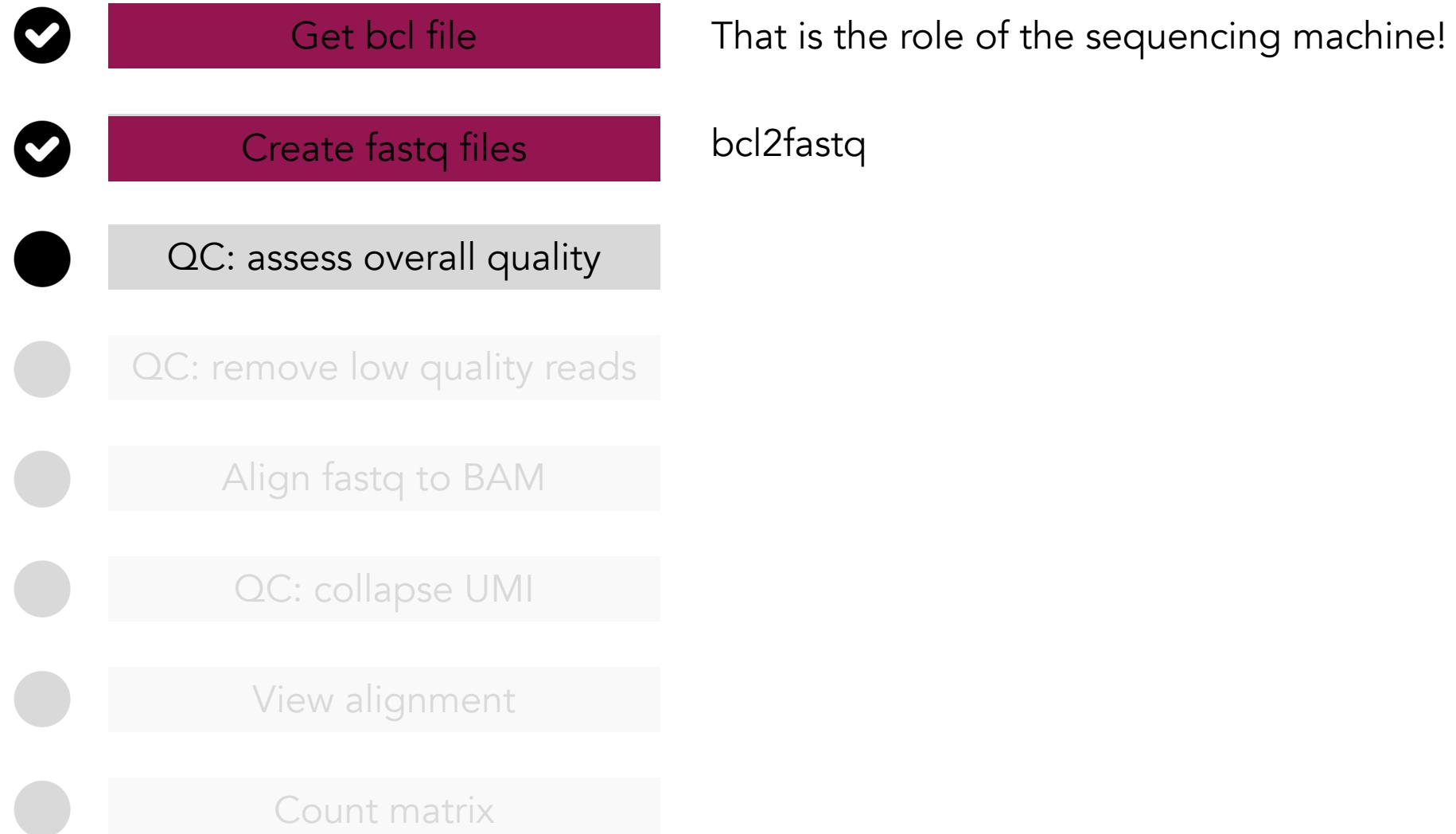
Reconstructing actual molecules loaded in the sequencer



After demultiplexing, here is what you have:



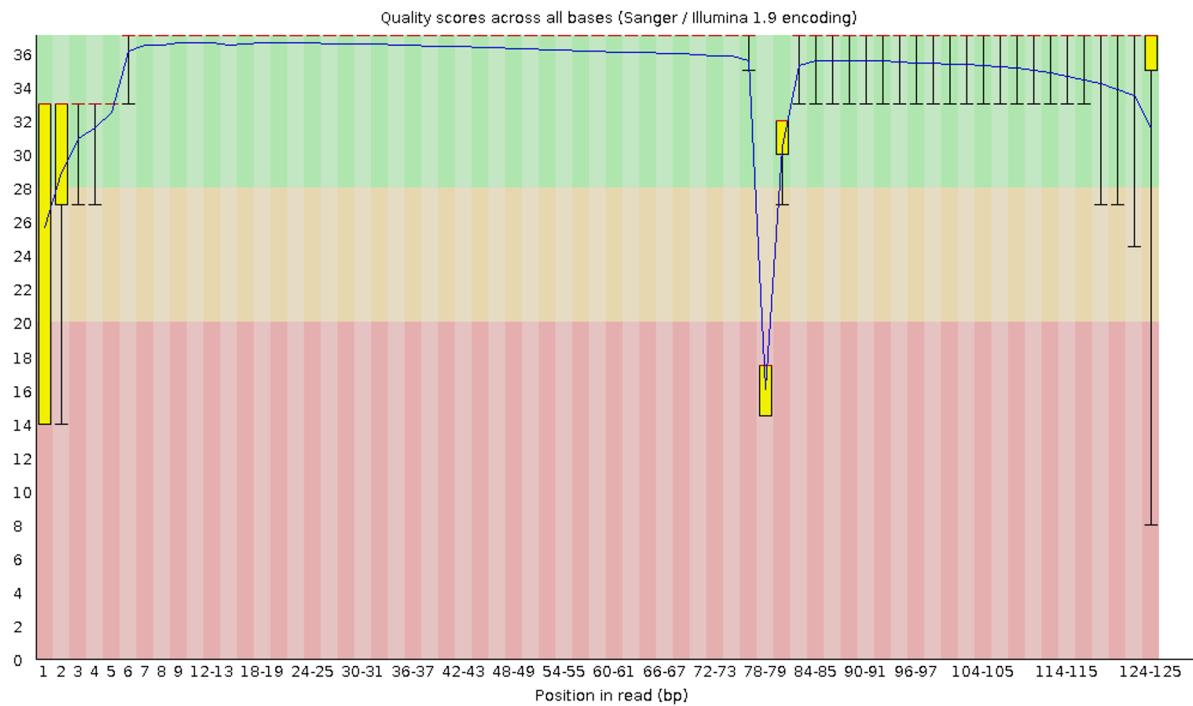
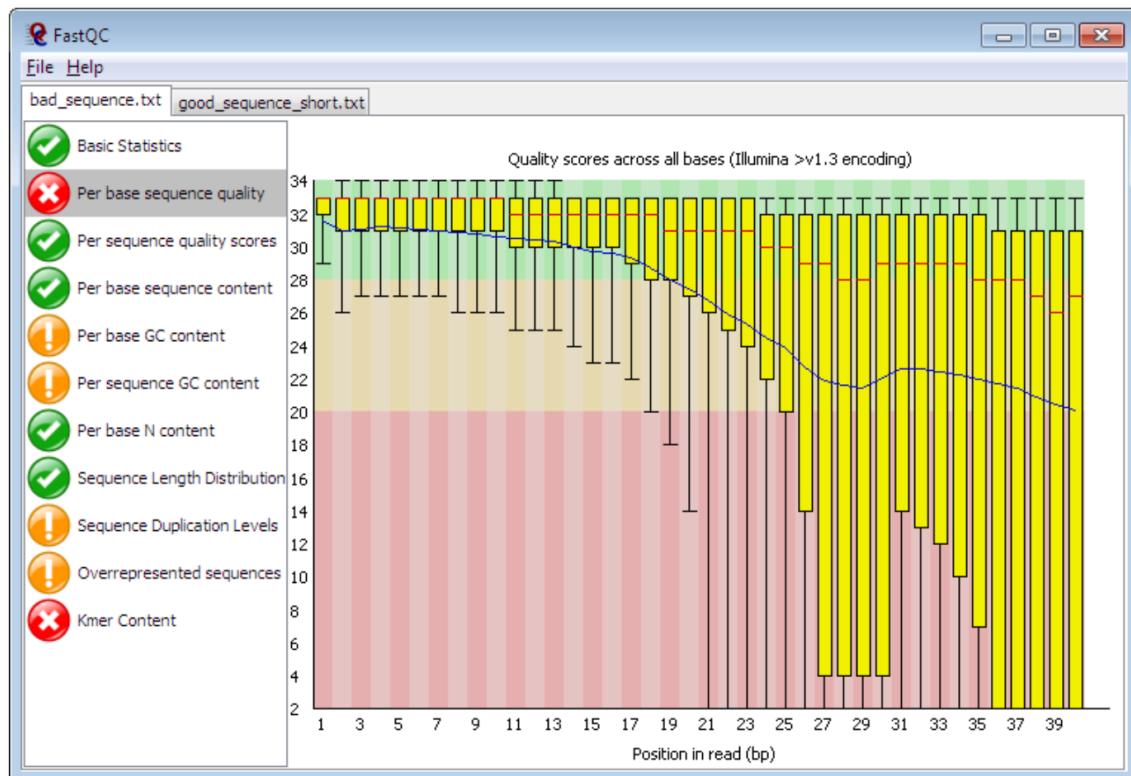
Flowchart: from .bcl to count matrix



Raw sequencing QC is typically done with Fastqc

FastQC

FastQC is a program designed to spot potential problems in high throughput sequencing datasets. It runs a set of analyses on one or more raw sequence files in fastq or bam format and produces a report which summarises the results.



FastQC will highlight any areas where this library looks unusual and where you should take a closer look. The program is not tied to any specific type of sequencing technique and can be used to look at libraries coming from a large number of different experiment types (Genomic Sequencing, ChIP-Seq, RNA-Seq, BS-Seq etc etc).

Flowchart: from .bcl to count matrix



Get bcl file

That is the role of the sequencing machine!



Create fastq files

bcl2fastq



QC: assess overall quality

fastqc



QC: remove low quality reads



Align fastq to BAM



QC: collapse UMI



View alignment



Count matrix

Filtering low quality reads

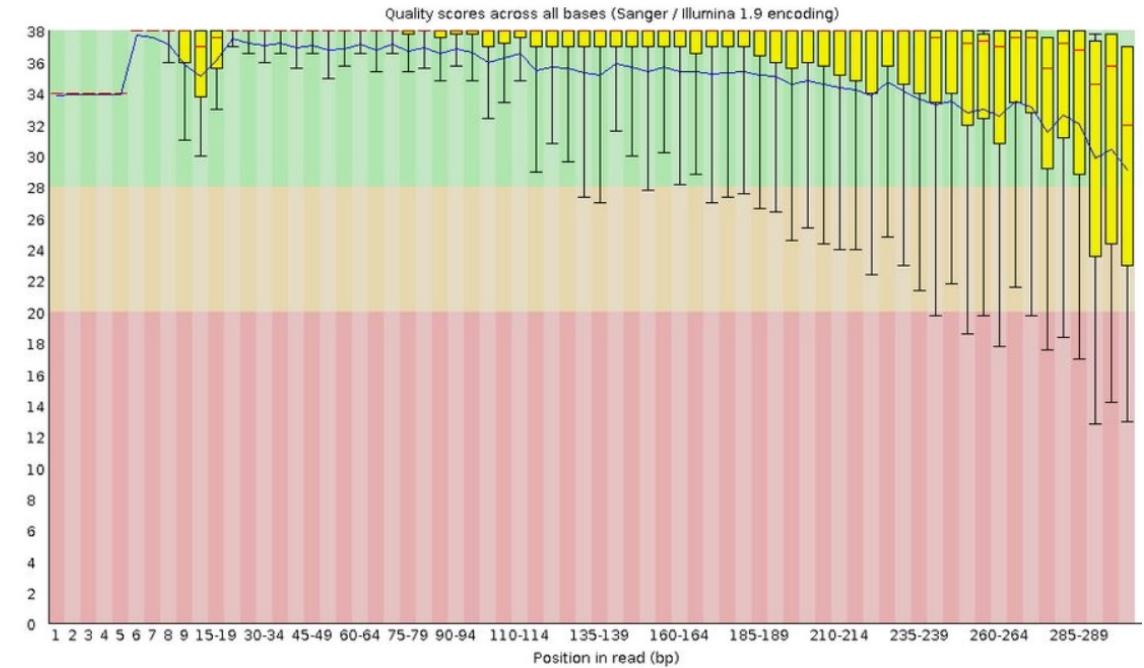
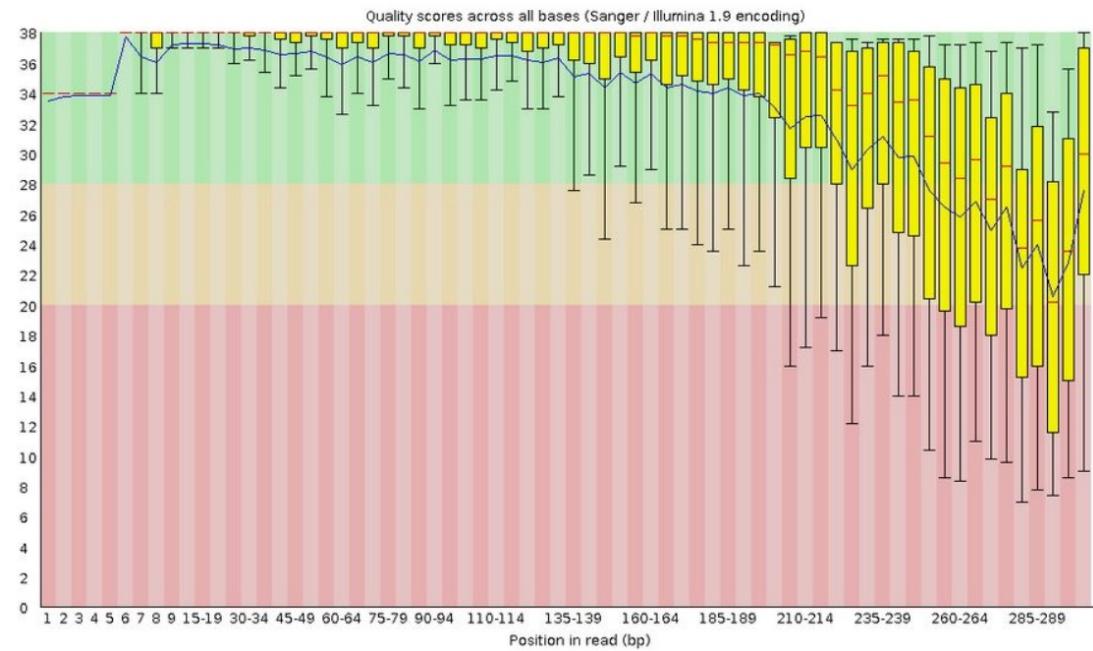
It is important to consider (and remove!) :

1. Reads with overall low quality
2. Unrecognized cell barcode

Filtering low quality reads

It is important to consider (and remove!) :

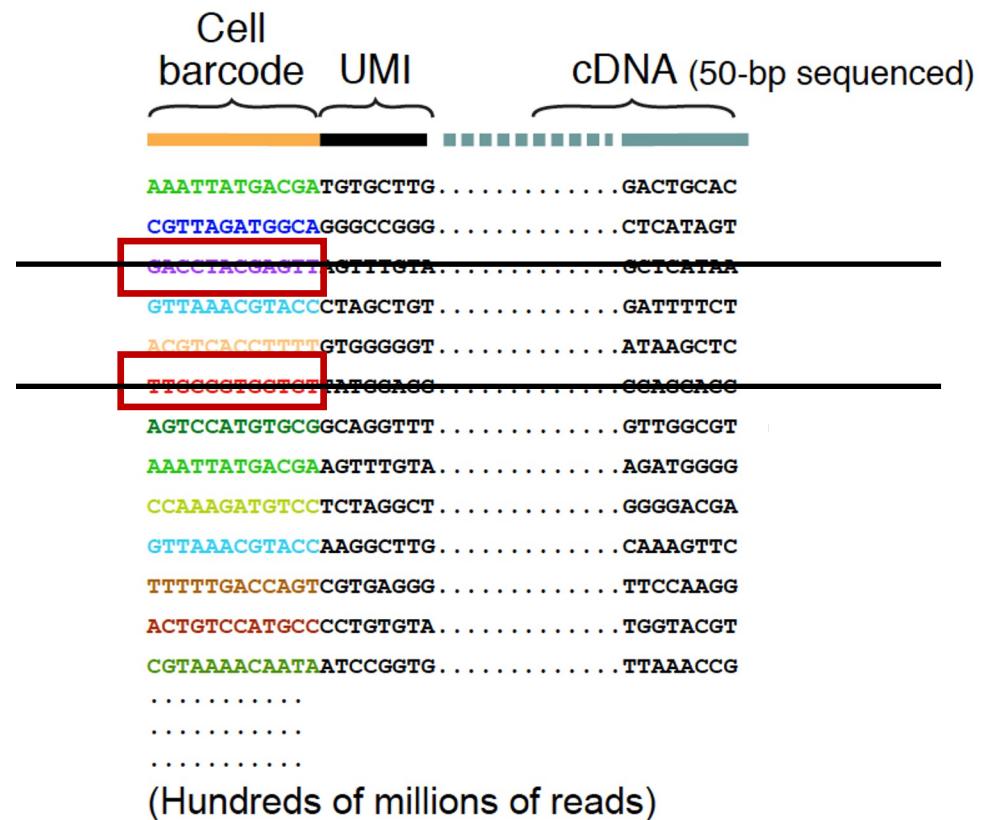
1. Reads with overall low quality
2. Unrecognized cell barcode



Filtering low quality reads

It is important to consider (and remove!) :

1. Reads with overall low quality
2. **Unrecognized cell barcode**



What is a barcode whitelist?

List of all known barcode sequences that have been included in the assay kit and are available during library preparation.

For example, there are roughly 737,000 cell barcodes in the whitelist for Cell Ranger's Single Cell 3' applications. Here are the first 10 lines of the corresponding barcode whitelist 737K-august-2016.txt:

```
AAACCTGAGAAACCAT
AAACCTGAGAAACCGC
AAACCTGAGAAACCTA
AAACCTGAGAAACGAG
AAACCTGAGAAACGCC
AAACCTGAGAAAGTGG
AAACCTGAGAACAACT
AAACCTGAGAACAAATC
AAACCTGAGAAACTCGG
AAACCTGAGAAACTGTA
```

How to correct barcode sequencing errors?

For every observed barcode in the dataset not on the whitelist, but 1-Hamming-distance away (i.e. 1 mismatch) from a whitelist barcode:

- Compute the posterior probability that the observed barcode originated from the whitelist barcode with a sequencing error at the differing base (based on the base Q score).
- Replace the observed barcode with the whitelist barcode with the highest posterior probability that exceeds 0.975.

Flowchart: from .bcl to count matrix



Get bcl file

That is the role of the sequencing machine!



Create fastq files

bcl2fastq



QC: assess overall quality

fastqc



QC: remove low quality reads

Automatic in most aligners



Align fastq to BAM



QC: collapse UMI



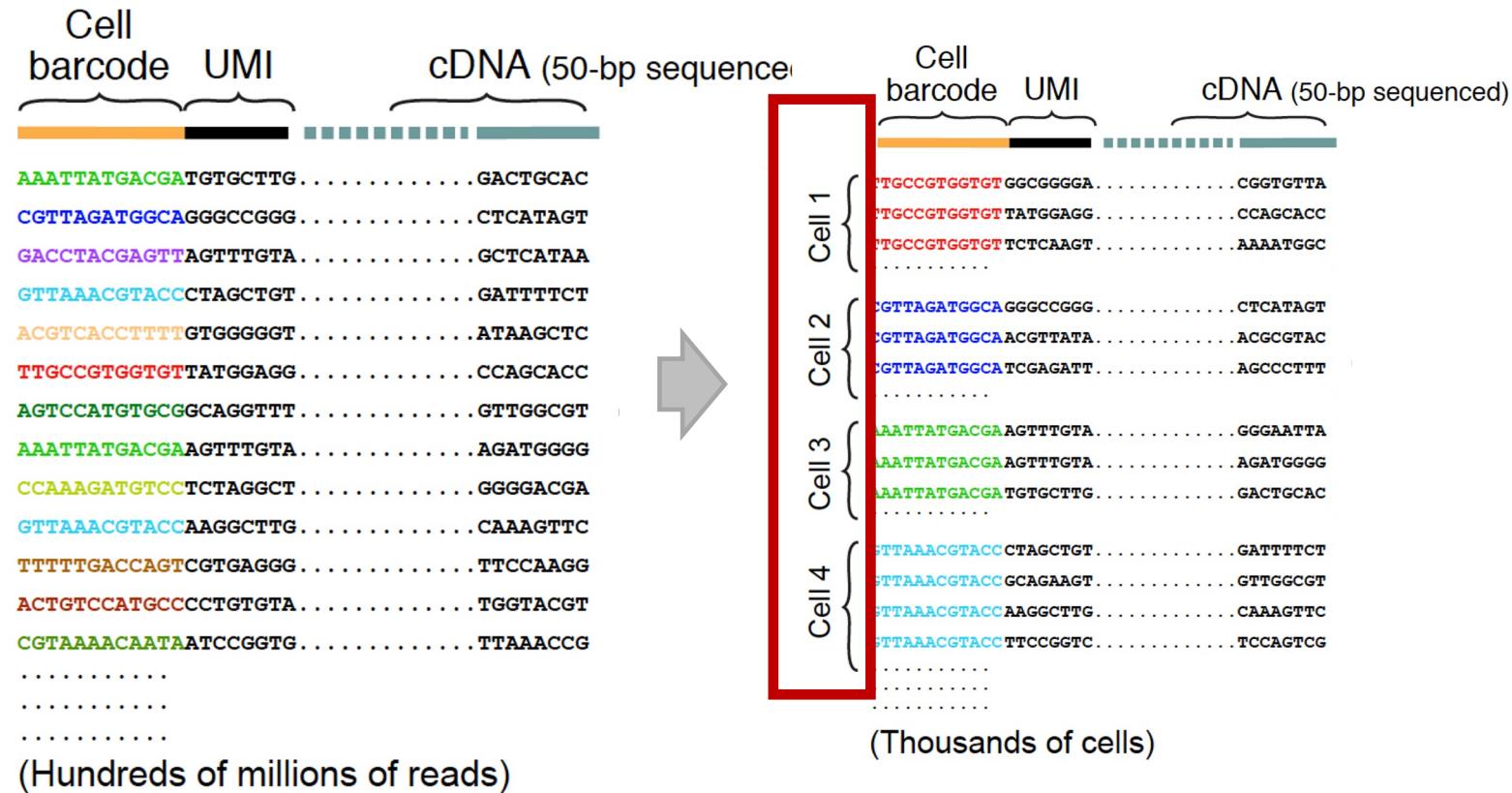
View alignment



Count matrix

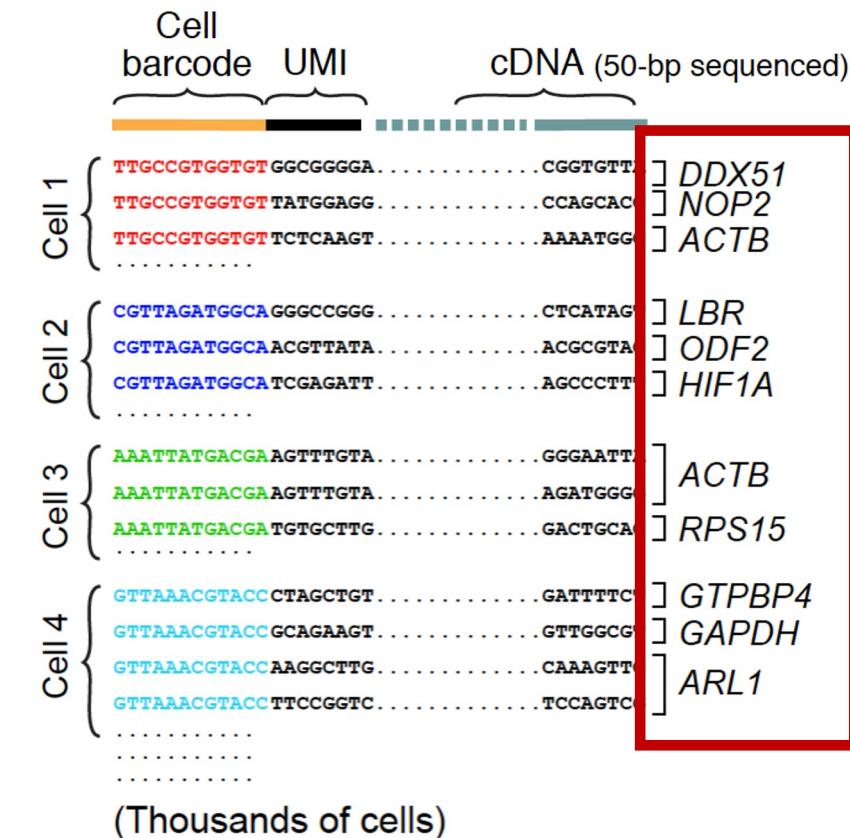
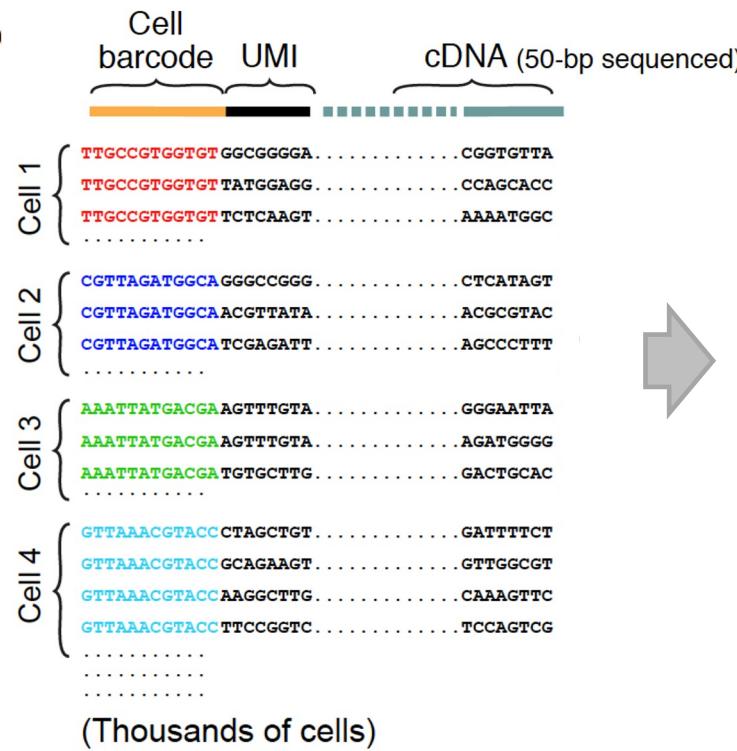
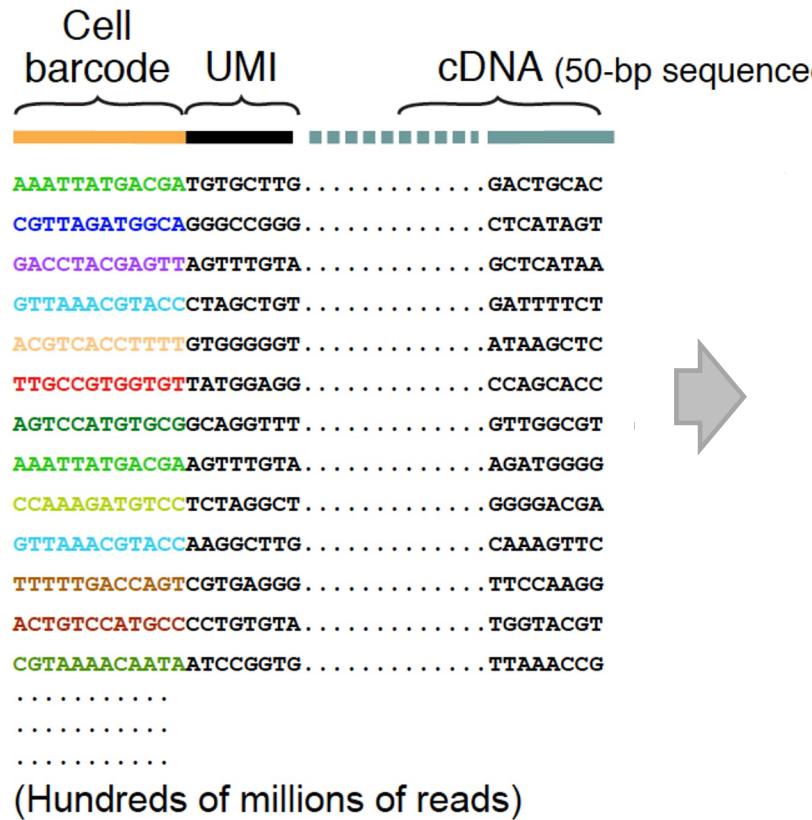
Aligning reads to a transcriptome reference

1. Group reads by cell-of-origin (using the cell barcodes)



Aligning reads to a transcriptome reference

1. Group reads by cell-of-origin (using the cell barcodes)
2. Recover which transcript the cDNA sequence aligns to



What do I need to align my reads?



Most aligners (included STAR-based cellranger) will map scRNASeq reads on a transcriptome index.

A transcriptome index consists of:

1. A genome sequence reference

FASTA example

What do I need to align my reads?

Most aligners (included STAR-based cellranger) will map scRNAseq reads on a transcriptome index.

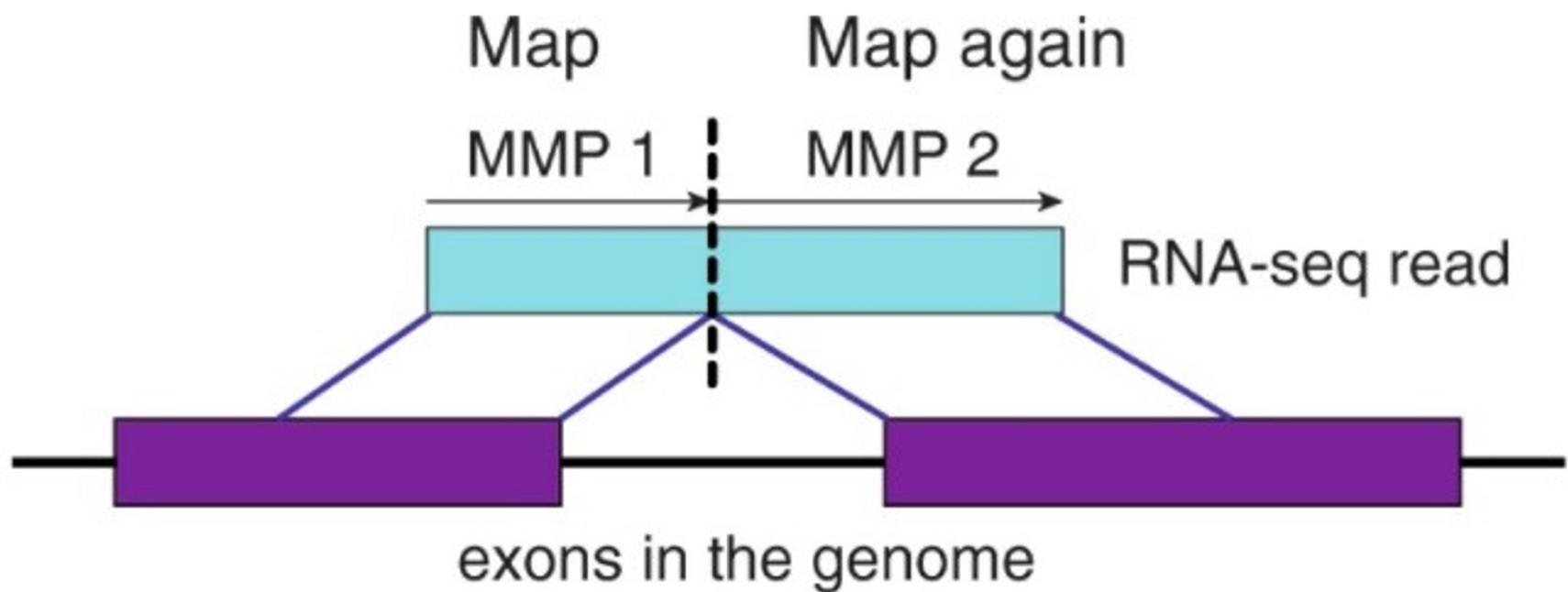
A transcriptome index consists of:

1. A genome sequence reference
2. A gene feature annotation reference

GTF example

Seqname	Source	Feature	Start	End	Score	Strand	Frame	Attributes
YHet	protein_coding	exon	311	424	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "1"; transcript_id "FBtr0113891"; exon_number "1";
YHet	protein_coding	CDS	311	424	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "1"; transcript_id "FBtr0113891"; exon_number "1";
YHet	protein_coding	exon	540	799	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "2"; transcript_id "FBtr0113891"; exon_number "2";
YHet	protein_coding	CDS	540	799	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "2"; transcript_id "FBtr0113891"; exon_number "2";
YHet	protein_coding	exon	857	1196	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "3"; transcript_id "FBtr0113891"; exon_number "3";
YHet	protein_coding	CDS	857	1196	.	+	1	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "3"; transcript_id "FBtr0113891"; exon_number "3";
YHet	protein_coding	exon	1254	1519	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "4"; transcript_id "FBtr0113891"; exon_number "4";
YHet	protein_coding	CDS	1254	1519	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "4"; transcript_id "FBtr0113891"; exon_number "4";
YHet	protein_coding	exon	1576	1729	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "5"; transcript_id "FBtr0113891"; exon_number "5";
YHet	protein_coding	CDS	1576	1729	.	+	1	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "5"; transcript_id "FBtr0113891"; exon_number "5";
YHet	protein_coding	exon	1816	2154	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "6"; transcript_id "FBtr0113891"; exon_number "6";
YHet	protein_coding	CDS	1816	2154	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "6"; transcript_id "FBtr0113891"; exon_number "6";
YHet	protein_coding	exon	2212	2324	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "7"; transcript_id "FBtr0113891"; exon_number "7";
YHet	protein_coding	CDS	2212	2324	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "7"; transcript_id "FBtr0113891"; exon_number "7";
YHet	protein_coding	exon	2376	2667	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "8"; transcript_id "FBtr0113891"; exon_number "8";
YHet	protein_coding	CDS	2376	2667	.	+	1	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "8"; transcript_id "FBtr0113891"; exon_number "8";
YHet	protein_coding	exon	2726	2879	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "9"; transcript_id "FBtr0113891"; exon_number "9";
YHet	protein_coding	CDS	2726	2879	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "9"; transcript_id "FBtr0113891"; exon_number "9";
YHet	protein_coding	exon	15564	15931	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "10"; transcript_id "FBtr0113891"; exon_number "10";
YHet	protein_coding	CDS	15564	15931	.	+	2	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "10"; transcript_id "FBtr0113891"; exon_number "10";
YHet	protein_coding	exon	16461	16907	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "11"; transcript_id "FBtr0113891"; exon_number "11";
YHet	protein_coding	CDS	16461	16907	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "11"; transcript_id "FBtr0113891"; exon_number "11";
YHet	protein_coding	exon	16954	19761	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "12"; transcript_id "FBtr0113891"; exon_number "12";
YHet	protein_coding	CDS	16954	19761	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "12"; transcript_id "FBtr0113891"; exon_number "12";
YHet	protein_coding	exon	30303	30469	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "13"; transcript_id "FBtr0113891"; exon_number "13";
YHet	protein_coding	CDS	30303	30469	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "13"; transcript_id "FBtr0113891"; exon_number "13";
YHet	protein_coding	exon	30522	31622	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "14"; transcript_id "FBtr0113891"; exon_number "14";
YHet	protein_coding	CDS	30522	31622	.	+	1	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "14"; transcript_id "FBtr0113891"; exon_number "14";
YHet	protein_coding	exon	33215	33413	.	+	.	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "15"; transcript_id "FBtr0113891"; exon_number "15";
YHet	protein_coding	CDS	33215	33410	.	+	1	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "15"; transcript_id "FBtr0113891"; exon_number "15";
YHet	protein_coding	stop_codon	33411	33413	.	+	0	gene_id "FBgn0001315"; transcript_id "FBtr0113891"; exon_number "16"; transcript_id "FBtr0113891"; exon_number "16";

STAR is a traditional aligner that works by trying to find the longest possible sequence which matches one or more sequences in the reference genome.



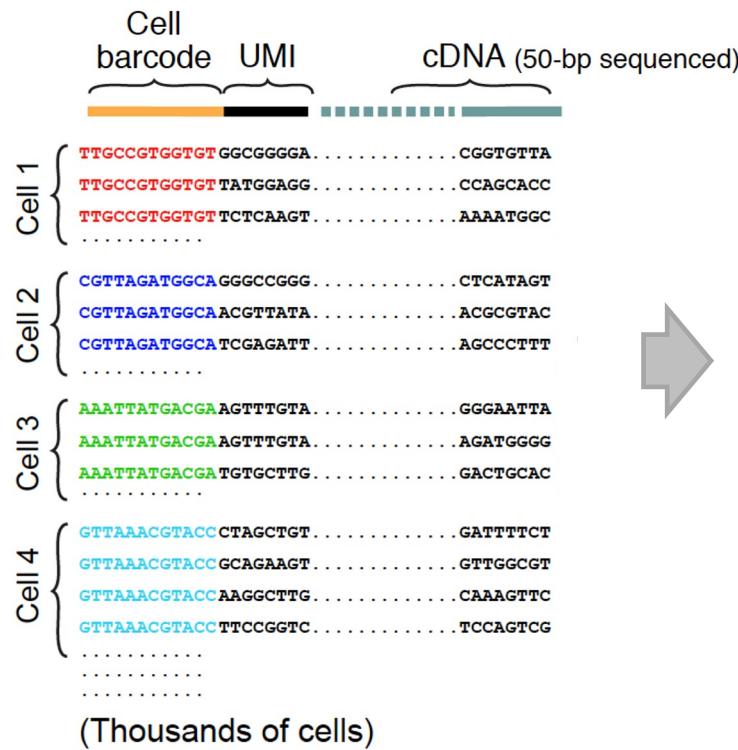
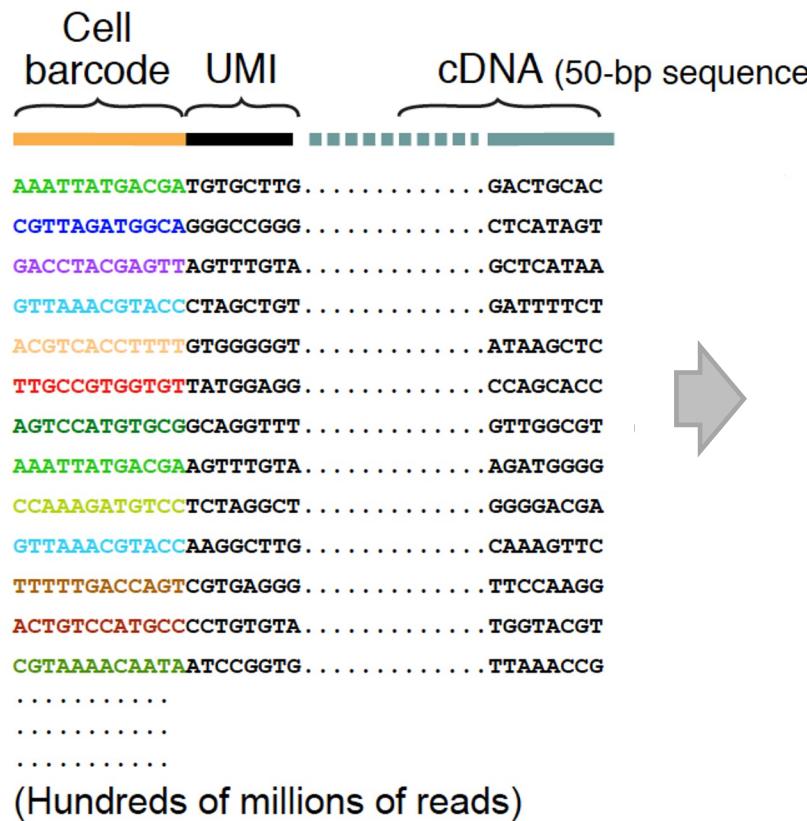
STAR is a traditional aligner that works by trying to find the longest possible sequence which matches one or more sequences in the reference genome.

- Advantages: STAR is a splice aware aligner, making it suitable if you are interested in studying alternative splicing.
- Disadvantage: STAR requires a lot of RAM.

An alternative requiring less RAM is HISAT2.

Aligning reads to a transcriptome reference

1. Group reads by cell-of-origin (using the cell barcodes)
2. Recover which transcript the cDNA sequence aligns to



	Cell1	Cell2	Cell3	Cell4
ACTB	1	0	2	0
ARL1	0	0	0	2
...
Cell 1	{ TTGCCGTGGTGT GGCGGGGA
Cell 2	{ CGTTAGATGGCA GGGCCGGG
Cell 3	{ AAATTATGACGA AGTTTGTA
Cell 4	{ GTTAACGTACC CTAGCTGT

(Thousands of cells)

Flowchart: from .bcl to count matrix



Get bcl file

That is the role of the sequencing machine!



Create fastq files

bcl2fastq



QC: assess overall quality

fastqc



QC: remove low quality reads

Automatic in most aligners



Align fastq to BAM

STAR / Kallisto / Bowtie / ...



QC: collapse UMI



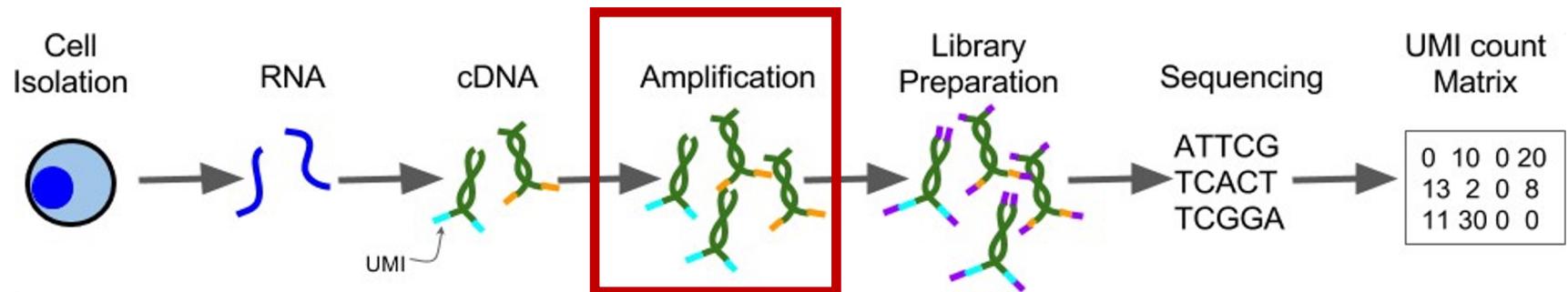
View alignment



Count matrix

PCR duplicates

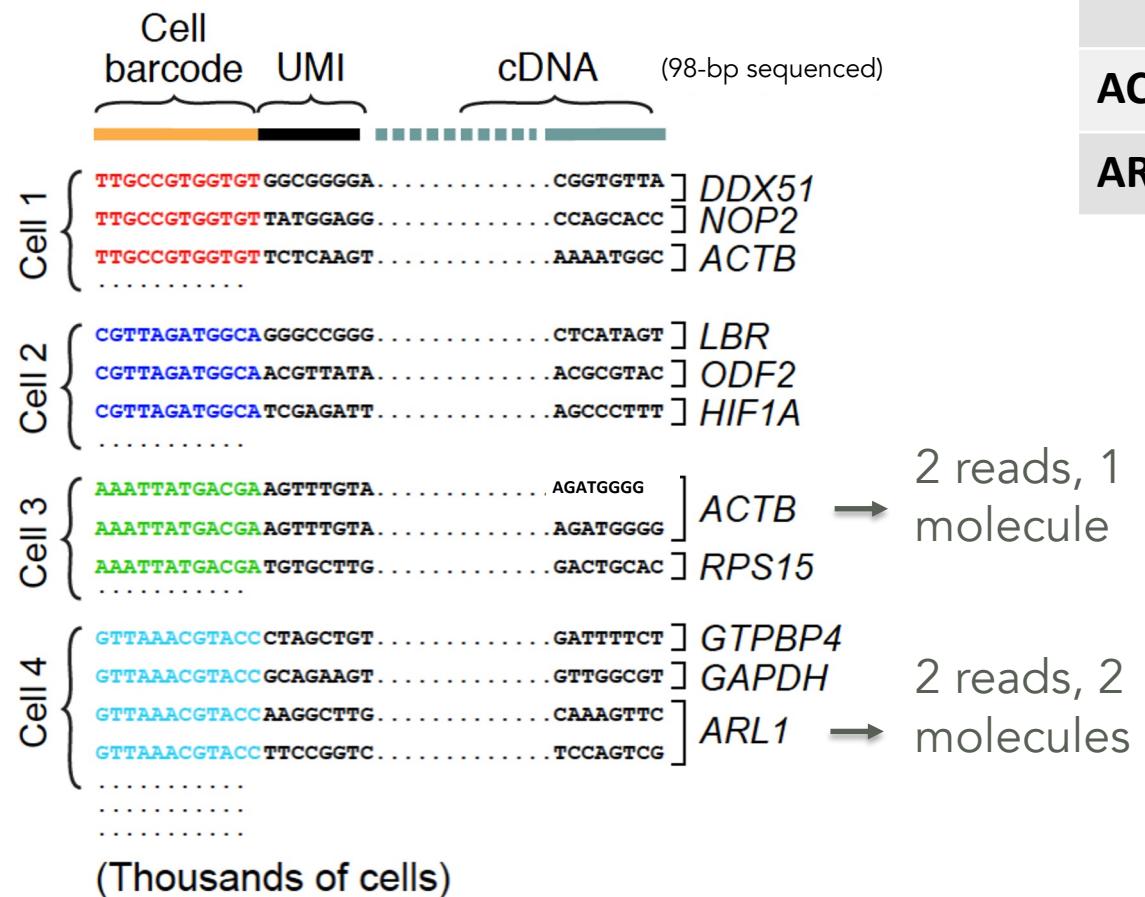
- PCR amplifications is an issue for both bulk and scRNASeq.
- During the PCR amplification process, some transcripts become over represented in the final library compared to their true abundance.
- The problem worsened when a high number of PCR cycles are used to generate the sequencing library, for example in scRNASeq due to the low amount of starting material.



Solution: using UMIs



UMIs enable sequencing reads to be assigned to individual transcript molecules and then the removal of amplification noise and biases from scRNASeq data.



	Cell1	Cell2	Cell3	Cell4
ACTB	1	0	1	0
ARL1	0	0	0	2

Columns: cells
Rows: features

"Error-correcting" almost-correct UMIs:

A UMI that is 1-Hamming-distance away from another UMI (with more reads) for the same cell barcode and gene is corrected to the UMI with more reads.

Flowchart: from .bcl to count matrix



Get bcl file

That is the role of the sequencing machine!



Create fastq files

bcl2fastq



QC: assess overall quality

fastqc



QC: remove low quality reads

Automatic in most aligners



Align fastq to BAM

STAR / Kallisto / Bowtie / ...



QC: collapse UMI

Automatic in most aligners

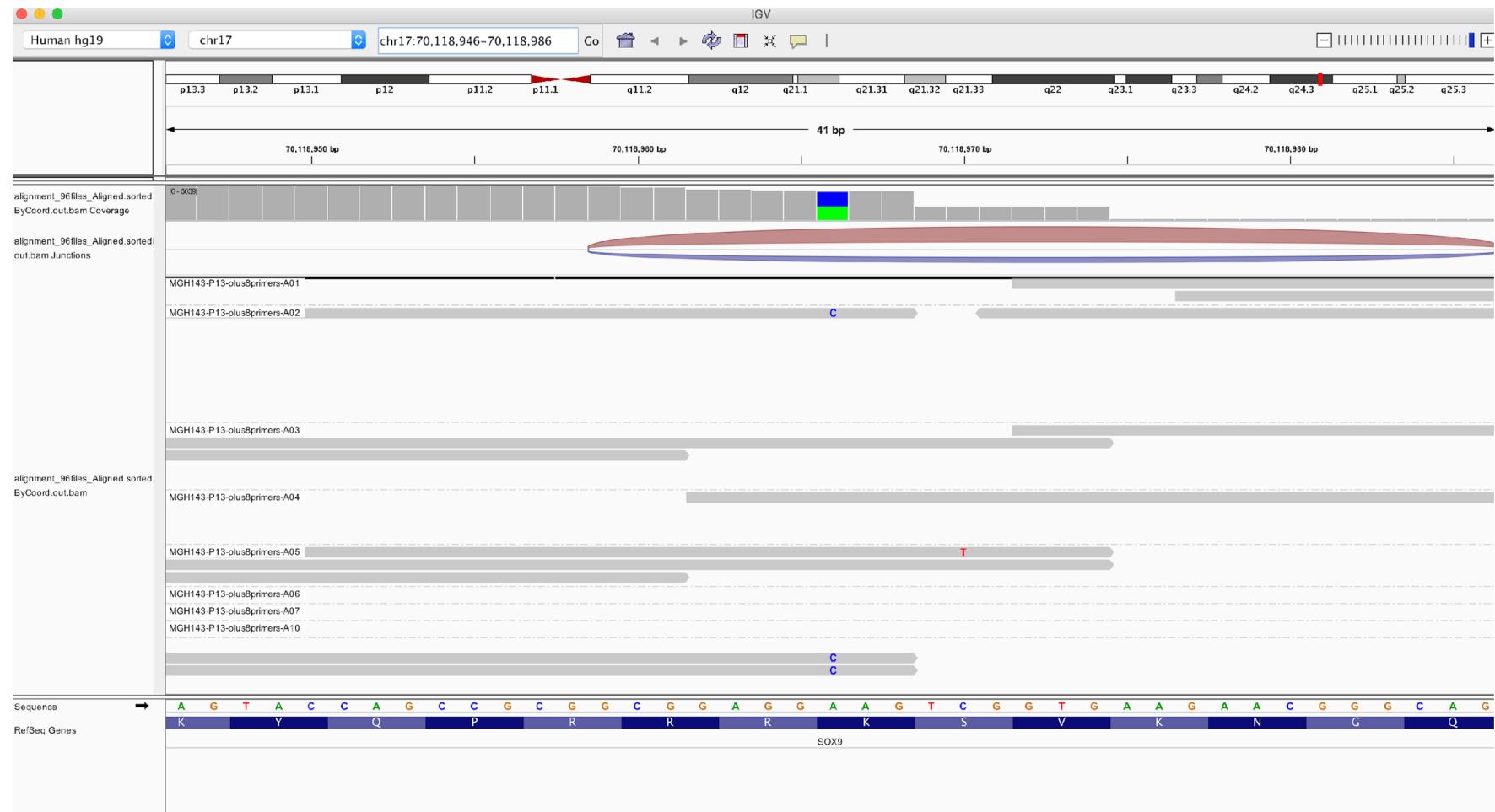


View alignment



Count matrix

Point mutation, splice junction, ...



Flowchart: from .bcl to count matrix



Get bcl file

That is the role of the sequencing machine!



Create fastq files

bcl2fastq



QC: assess overall quality

fastqc



QC: remove low quality reads

Automatic in most aligners



Align fastq to BAM

STAR / Kallisto / Bowtie / ...



QC: collapse UMI

Automatic in most aligners



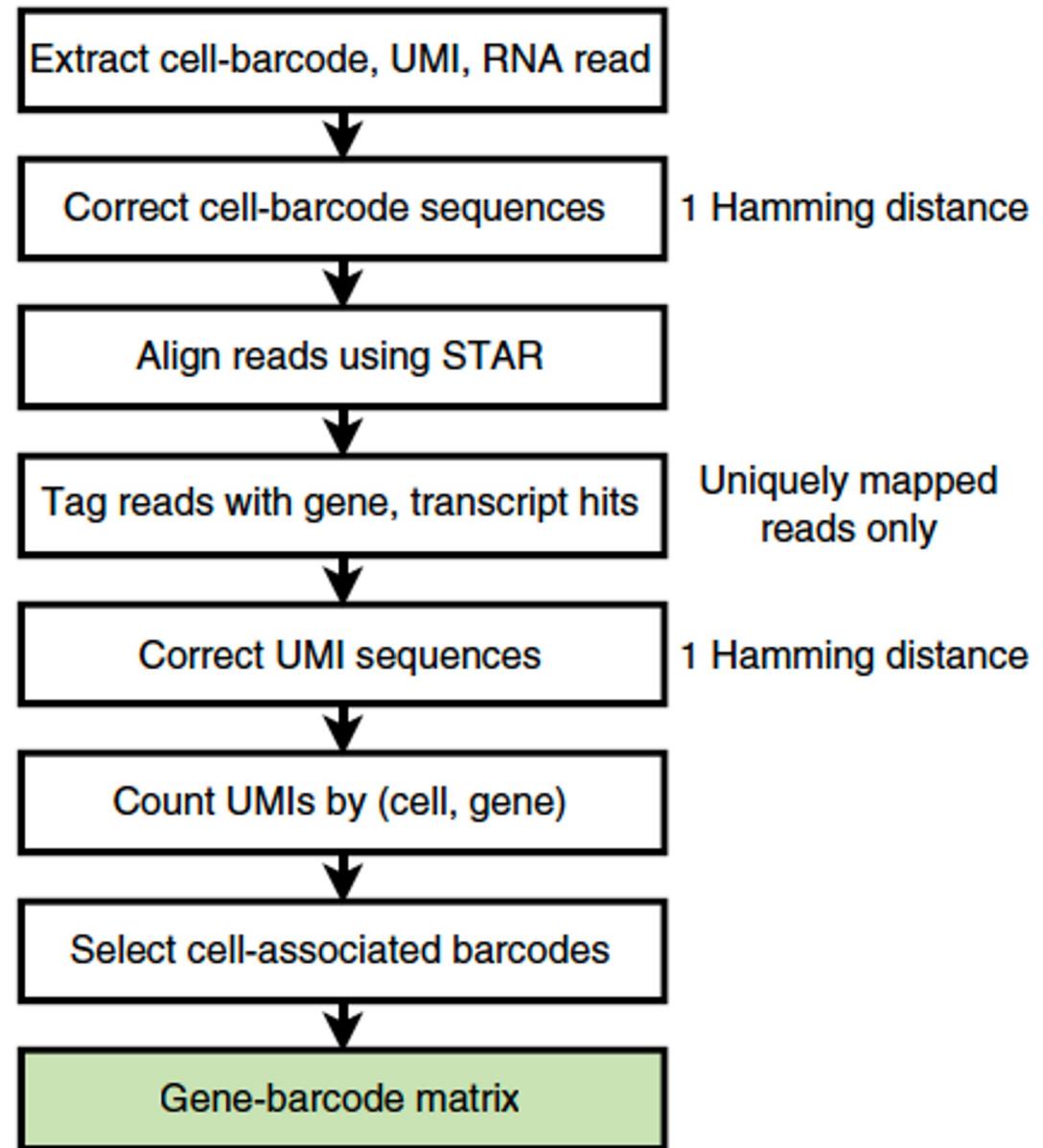
View alignment

Automatic in most aligners



Count matrix

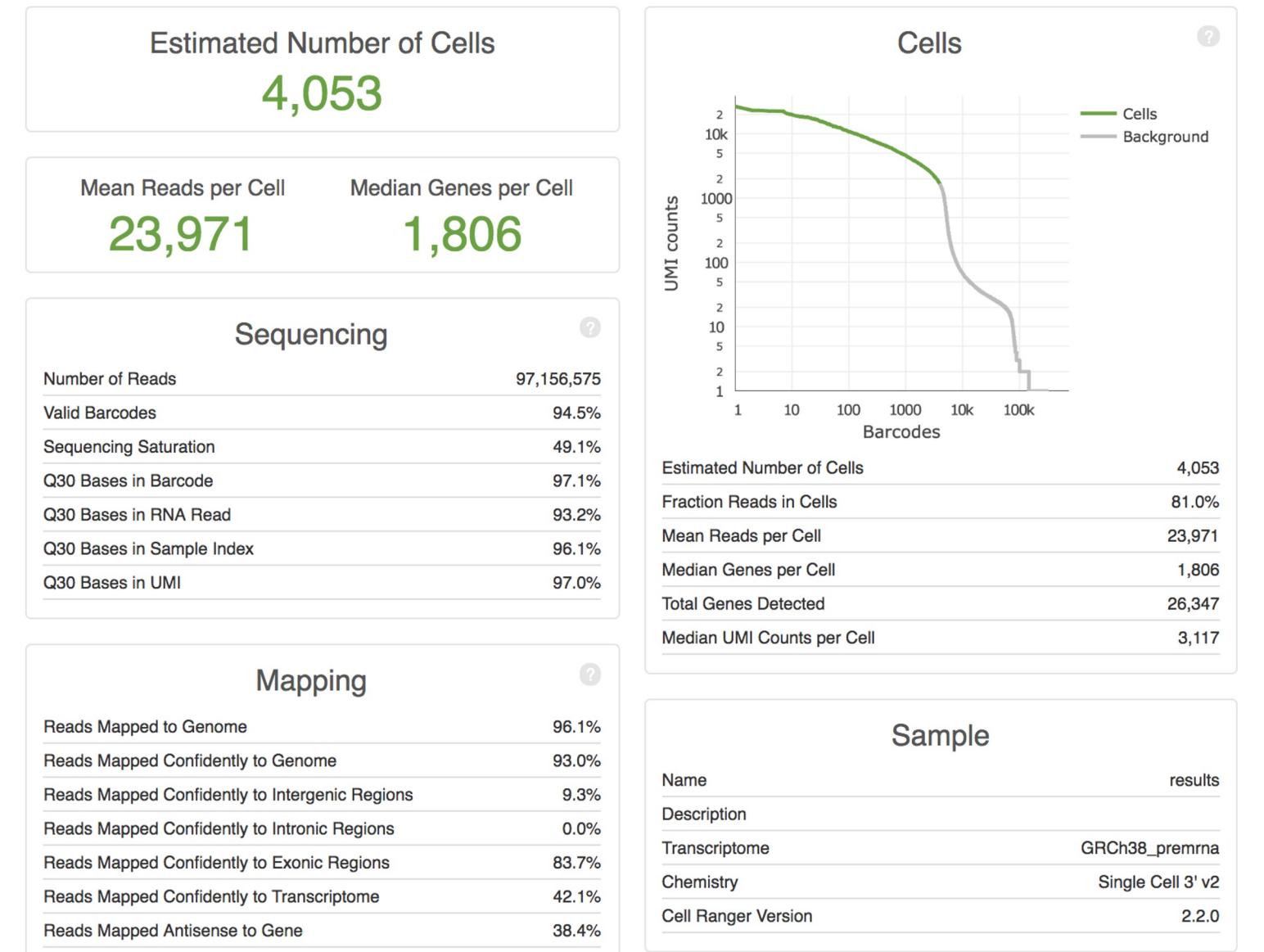
Cellranger count pipeline



Which reads are considered for UMI counting by Cell Ranger?

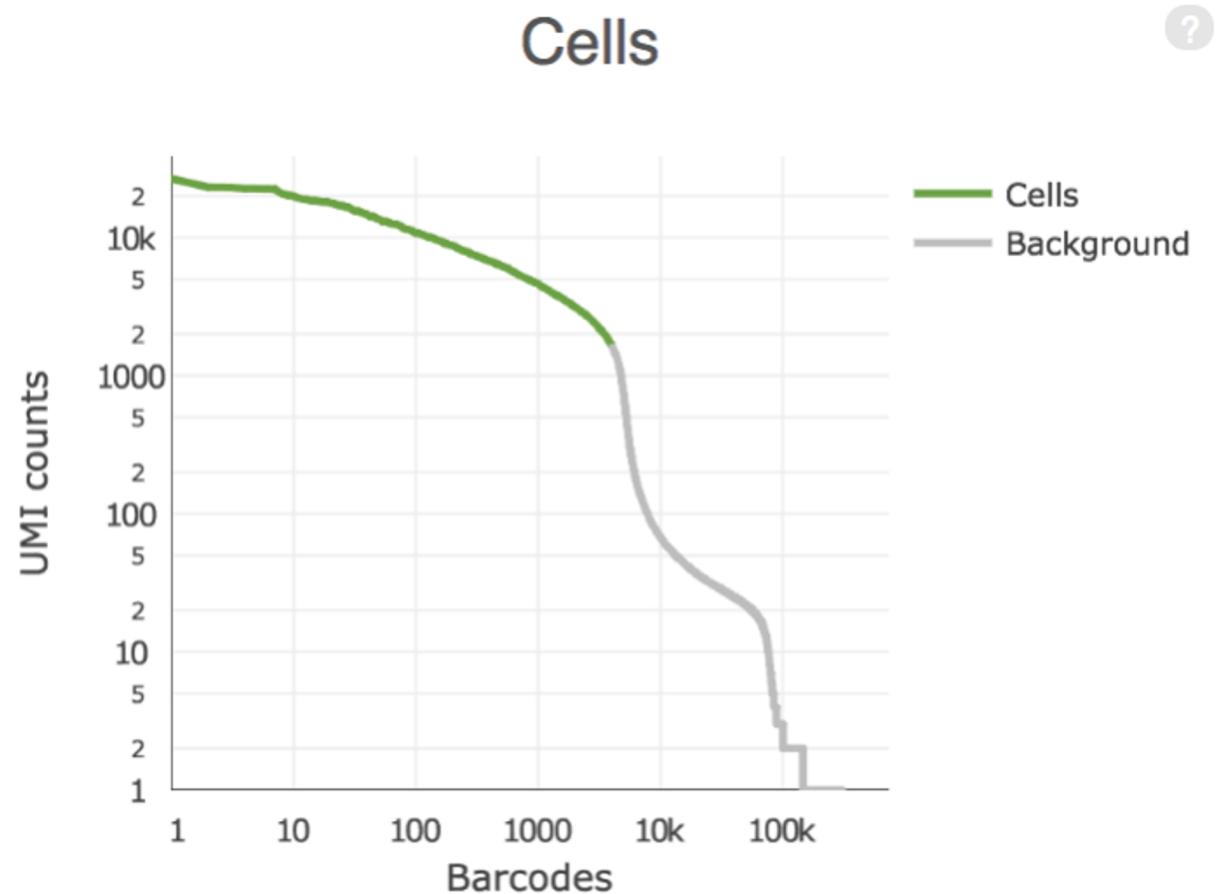
1. Only reads with a valid UMI and a valid 10x barcode.
2. No bases with base quality < 10.
3. Read maps to exactly one gene.
4. Overlaps an exon by at least 50% in a way consistent with annotated splice junctions and strand annotation.
5. Multiple reads that map to the same UMI will only count once.

Cellranger output



Cellranger output

Some cell barcodes have many UMIs, but most do not.



How deeply do I need to sequence?

- At high sequencing depth, we can detect:
 - The expression of rare splice variants.
 - Quantitative modulations in transcript abundance.

- Lower sequencing depth can still be very powerful :
 - Catalog of cell types.
 - Catalog of transcriptional programs.

Flowchart: from .bcl to count matrix

Cellranger

- ✓ Get bcl file That is the role of the sequencing machine!
- ✓ Create fastq files `bcl2fastq`
- ✓ QC: assess overall quality `fastqc`
- ✓ QC: remove low quality reads Automatic in most aligners
- ✓ Align fastq to BAM `STAR / Kallisto / Bowtie / ...`
- ✓ QC: collapse UMI Automatic in most aligners
- ✓ View alignment Automatic in most aligners
- ✓ Count matrix `Cellranger`

Any question?

