Single-cell RNA-Seq with kallisto

Vasilis Ntranos

Outline

- 10x Chromium dataset

10X GENOMICS PRODUCTS TECHNOLOGY COMPANY CAREERS

Support › Single Cell › Datasets SE

3k PBMCs from a Healthy Donor

Chromium Demonstration Dataset by Cell Ranger 1.1.0

Peripheral blood mononuclear cells (PBMCs) from a healthy donor (same donor as pbmc6k).

PBMCs are primary cells with relatively small amounts of RNA (~1pg RNA/cell).

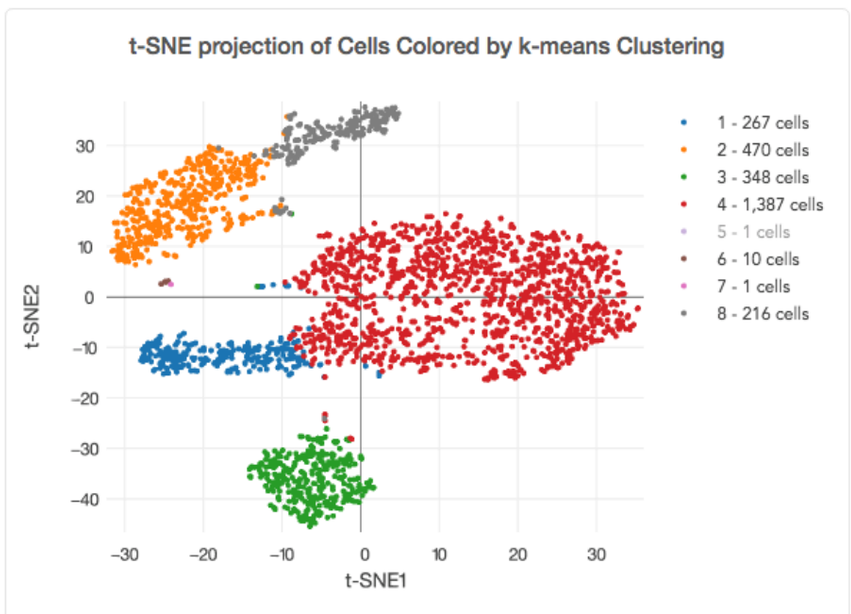
- 2,700 cells detected
- Sequenced on Illumina NextSeq 500 with ~69,000 reads per cell
- 98bp read1 (transcript), 8bp I5 sample barcode, 14bp I7 GemCode barcode and 10bp read2 (UMI)
- Analysis run with --cells=3000

Published on May 26, 2016

View Summary

Show batch download instructions

Input Files	Size	md5sum
FASTQs	17.38 GB	7999bb457af4e11c57f96bf97e4ee645



Outline

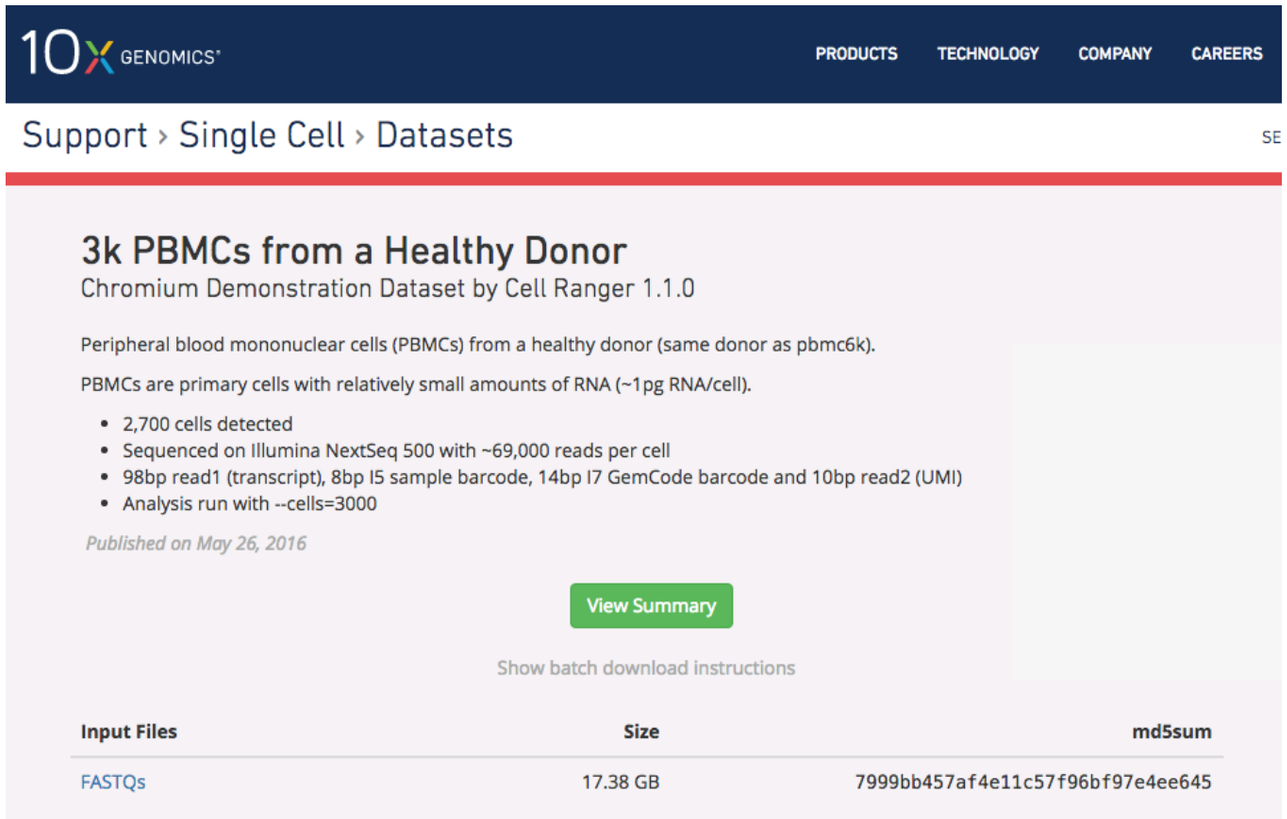
- 10x Chromium dataset

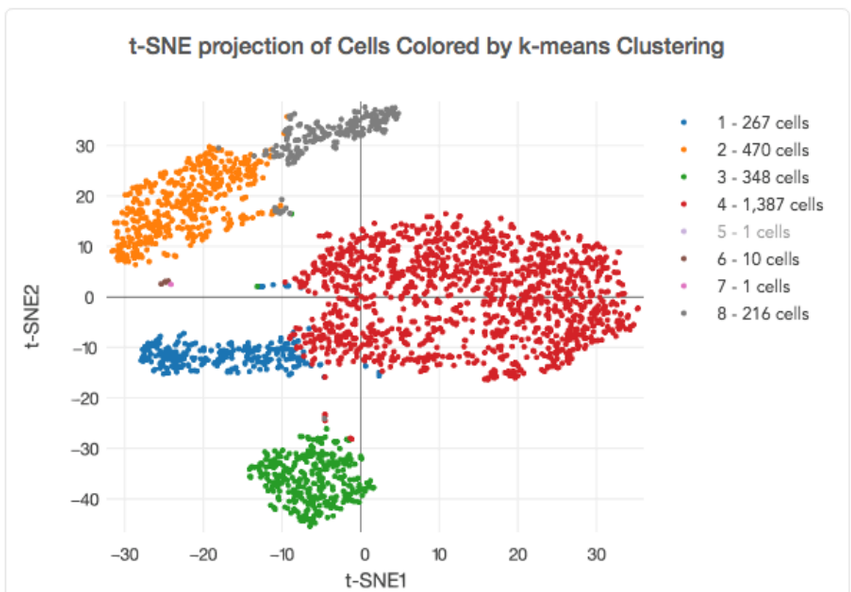
10x GENOMICS

- Single-cell workflow using kallisto

github.com/pachterlab/scRNA-Seq-TCC-prep/

jupyter notebook/python scripts for
clustering based on transcript compatibility counts





Outline

- 10x Chromium dataset

10x GENOMICS

- Single-cell workflow using kallisto

github.com/pachterlab/scRNA-Seq-TCC-prep/

jupyter notebook/python scripts for
clustering based on transcript compatibility counts

Single Cell Datasets

▼ Chromium Demonstration

- 1:1 Mixture of Fresh Frozen Human (HEK293T) and Mouse (NIH3T3) Cells
- 33k PBMCs from a Healthy Donor
- 3k PBMCs from a Healthy Donor
- 6k PBMCs from a Healthy Donor

▼ Single Cell 3′ Paper: Zheng et al. 2016 (29 datasets)

Datasets for the manuscript Zheng et al, "Massively parallel digital transcriptional profiling of single cells". We encourage you to download the data here, as the BAM files deposited in the SRA database have had the cell barcode tags removed. We are working with NCBI to resolve this issue.

- 293T Cells
- 293T and 3T3 Cell Mixture
- 50%:50% Donor B: Donor C PBMC Mixture
- 50%:50% Jurkat:293T Cell Mixture
- 90%:10% Donor B: Donor C PBMC Mixture
- 99%:1% Donor B: Donor C PBMC Mixture
- 99%:1% Jurkat:293T Cell Mixture
- AML027 Post-transplant BMMCs
- AML027 Pre-transplant BMMCs
- AML035 Post-transplant BMMCs
- AML035 Pre-transplant BMMCs
- CD14+ Monocytes
- CD19+ B Cells
- CD34+ Cells
- CD4+ Helper T Cells
- CD4+/CD25+ Regulatory T Cells
- CD4+/CD45RA+/CD25- Naive T cells
- CD4+/CD45RO+ Memory T Cells
- CD56+ Natural Killer Cells
- CD8+ Cytotoxic T cells
- CD8+/CD45RA+ Naive Cytotoxic T Cells
- ERCC (1k GEMS, 1:10 Dilution)
- Fresh 68k PBMCs (Donor A)
- Frozen BMMCs (Healthy Control 1)
- Frozen BMMCs (Healthy Control 2)
- Frozen PBMCs (Donor A)
- Frozen PBMCs (Donor B)
- Frozen PBMCs (Donor C)
- Jurkat Cells

Outline

- 10x Chromium dataset

10x GENOMICS

- Single-cell workflow using kallisto

github.com/pachterlab/scRNA-Seq-TCC-prep/

jupyter notebook/python scripts for
clustering based on transcript compatibility counts

Fluidigm C1

FLUIDIGM

Single Cell Datasets

- Chromium Demonstration
  - 1:1 Mixture of Fresh Frozen Hu… (H3T3) Cells
  - 33k PBMCs from a Healthy D…
  - 3k PBMCs from a Healthy Do…
  - 6k PBMCs from a Healthy Do…
- Single Cell 3' Paper: Zheng e…

Datasets for the manuscript Zheng… al transcriptional profiling of single cells". We encourage you to do… BAM files deposited in the SRA database have had the cell barcode tag… working with NCBI to resolve this issue.

- 293T Cells
- 293T and 3T3 Cell Mixture
- 50%:50% Donor B: Donor C PBMC Mixture
- 50%:50% Jurkat:293T Cell Mixture
- 90%:10% Donor B: Donor C PBMC Mixture
- 99%:1% Donor B: Donor C PBMC Mixture
- 99%:1% Jurkat:293T Cell Mixture
- AML027 Post-transplant BMMCs
- AML027 Pre-transplant BMMCs
- AML035 Post-transplant BMMCs
- AML035 Pre-transplant BMMCs
- CD14+ Monocytes
- CD19+ B Cells
- CD34+ Cells
- CD4+ Helper T Cells
- CD4+/CD25+ Regulatory T Cells
- CD4+/CD45RA+/CD25- Naive T cells
- CD4+/CD45RO+ Memory T Cells
- CD56+ Natural Killer Cells
- CD8+ Cytotoxic T cells
- CD8+/CD45RA+ Naive Cytotoxic T Cells
- ERCC (1k GEMS, 1:10 Dilution)
- Fresh 68k PBMCs (Donor A)
- Frozen BMMCs (Healthy Control 1)
- Frozen BMMCs (Healthy Control 2)
- Frozen PBMCs (Donor A)
- Frozen PBMCs (Donor B)
- Frozen PBMCs (Donor C)
- Jurkat Cells

Outline

- 10x Chromium dataset

10x GENOMICS

Fluidigm C1

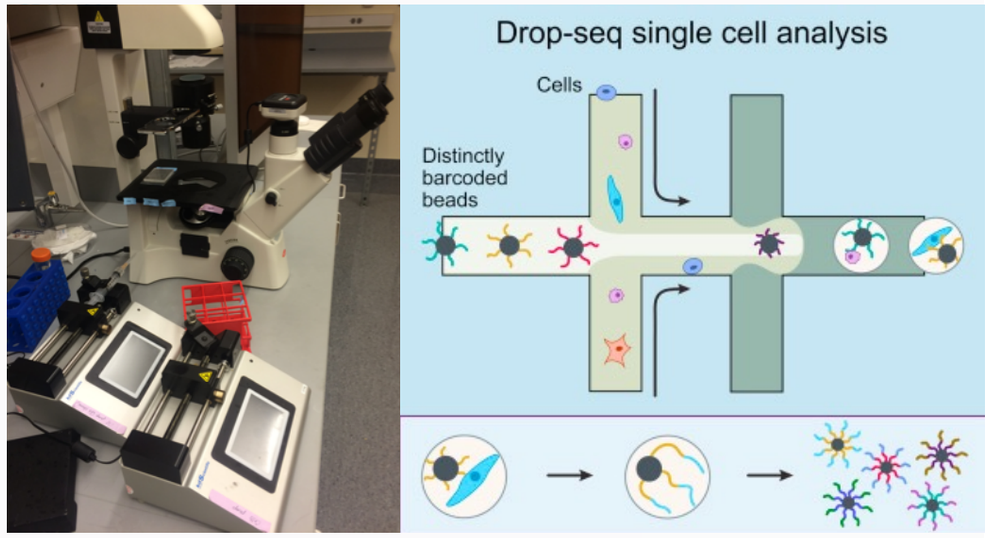
FLUIDIGM

- Single-cell workflow using kallisto

github.com/pachterlab/scRNA-Seq-TCC-prep/

jupyter notebook/python scripts for
clustering based on transcript compatibility counts

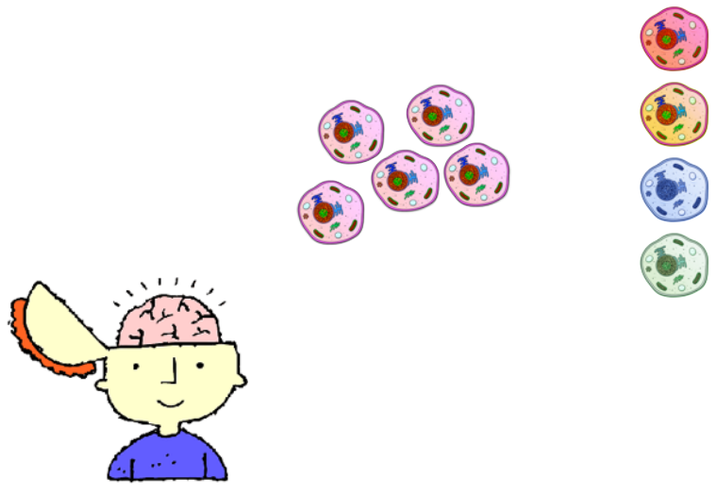
Drop-seq

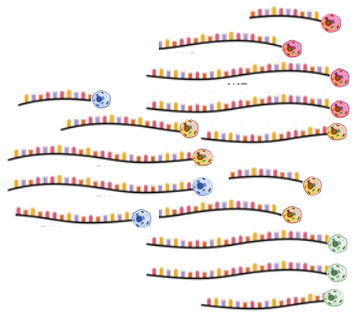


Macosko et al. 2015

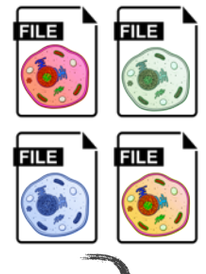
Introduction — Single-Cell RNA-Seq

Standard workflow:

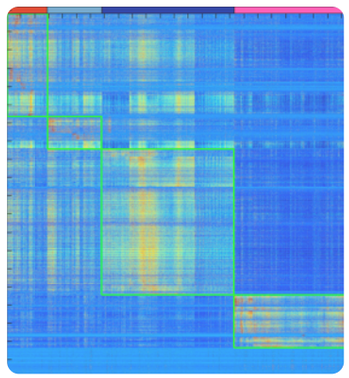




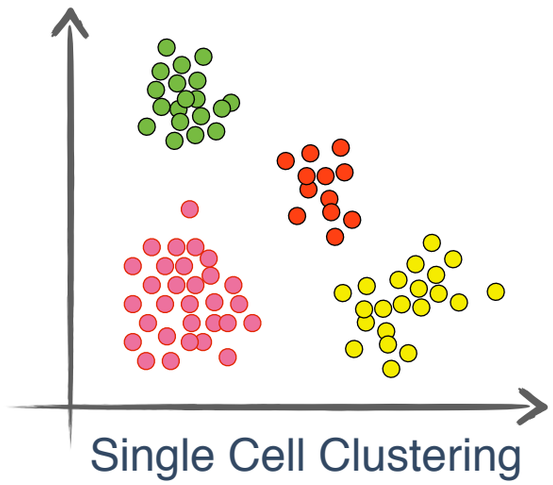
Single Cell Reads



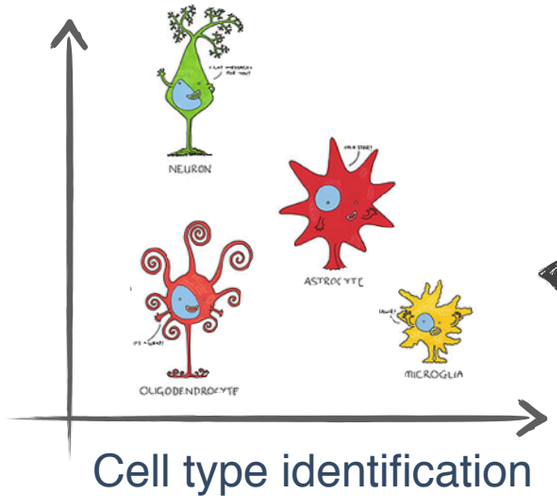

```
>
read_alignment
quantification
```



Gene expression levels



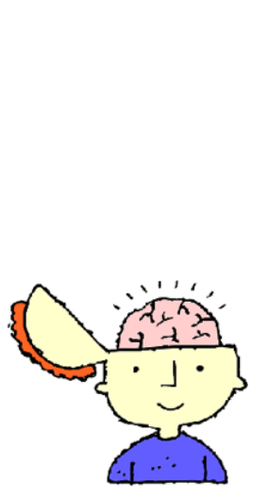
Single Cell Clustering

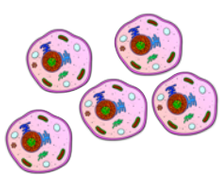


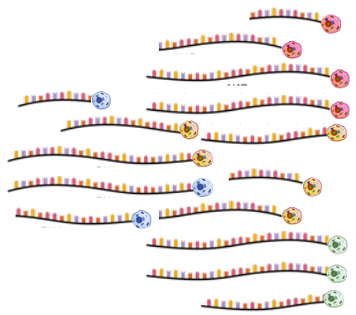
Cell type identification

Introduction — Single-Cell RNA-Seq

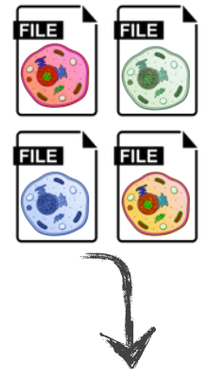
Standard workflow:







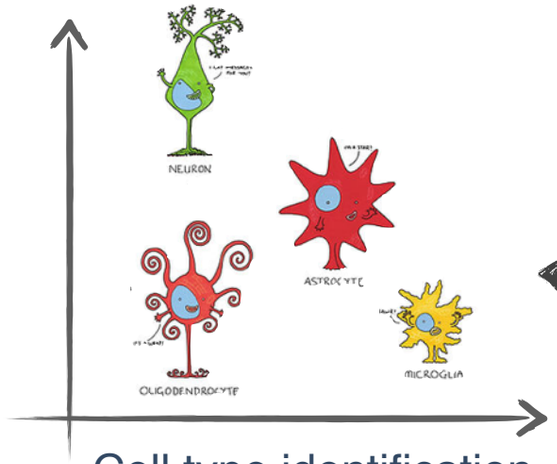
Single Cell Reads



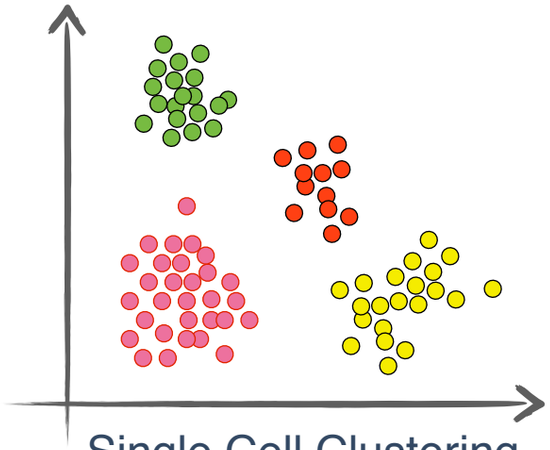

```
>
read_alignment
quantification
```

The goal is to scale up..

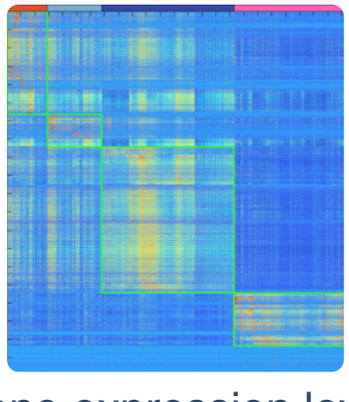
...both the **number of cells** and the **read depth**
in order to ultimately **increase single-cell resolution**



Cell type identification



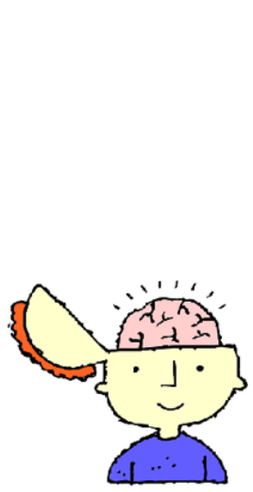
Single Cell Clustering

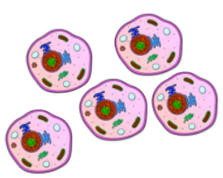


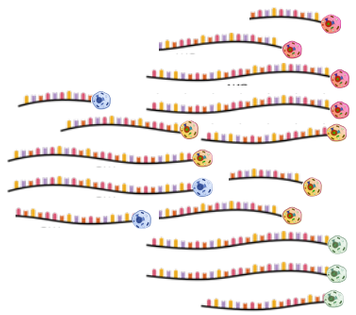
Gene expression levels

Introduction — Single-Cell RNA-Seq

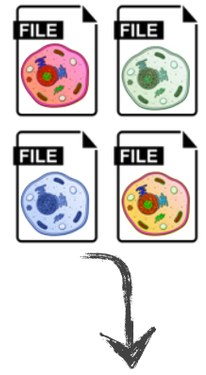
Standard workflow:







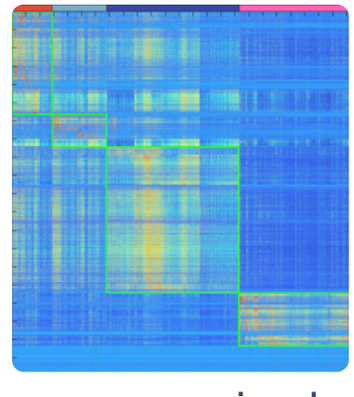
Single Cell Reads



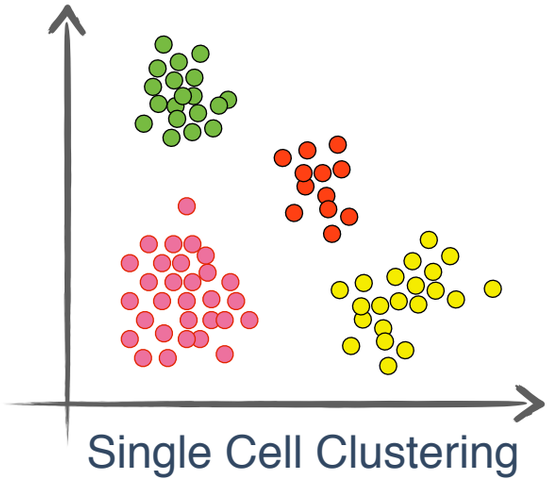

```
>
read_alignment

quantification
```

Wide variety of scRNA-Seq assays

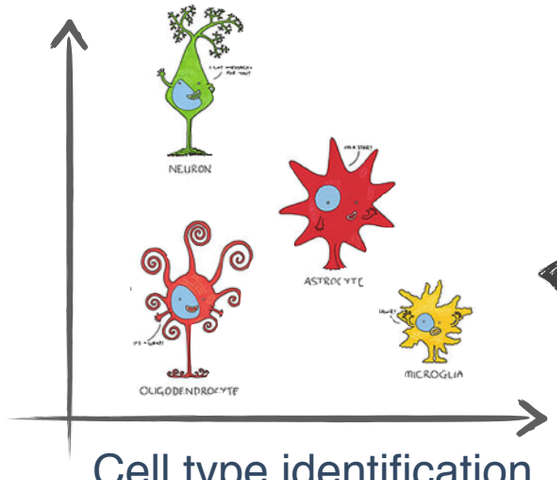
… technologies with different characteristics,
need different **models**, difficult to **compare** results...
lack of "**universality**"



Gene expression levels



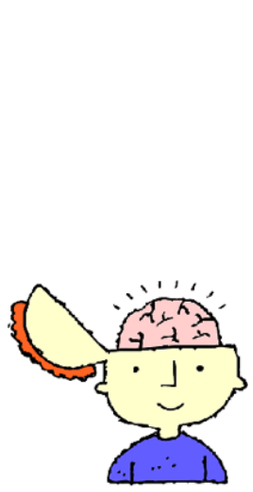
Single Cell Clustering

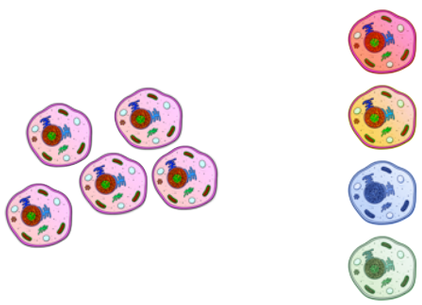


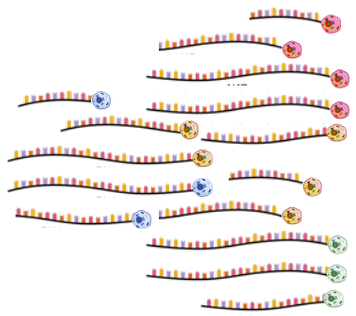
Cell type identification

Introduction — Single-Cell RNA-Seq

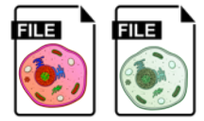
Standard workflow:







Single Cell Reads

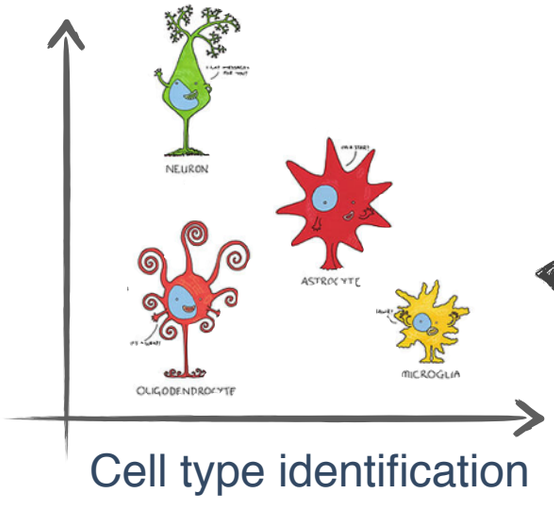



```
>
read_alignment
quantification
```

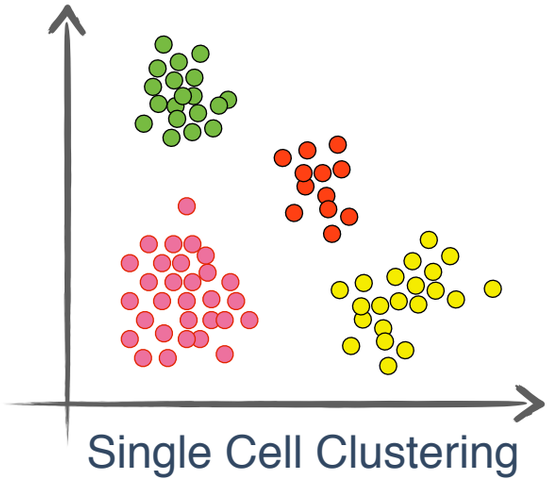
Computational Bottleneck

need to process each cell separately...

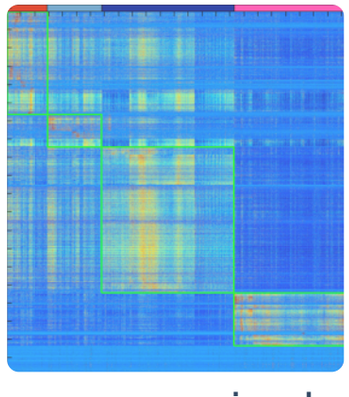
processing time scales ~ **number of cells x read depth**



Cell type identification



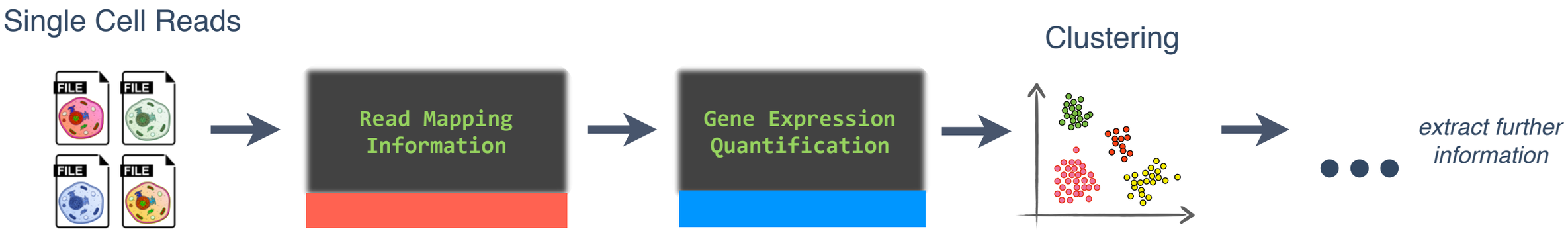
Single Cell Clustering



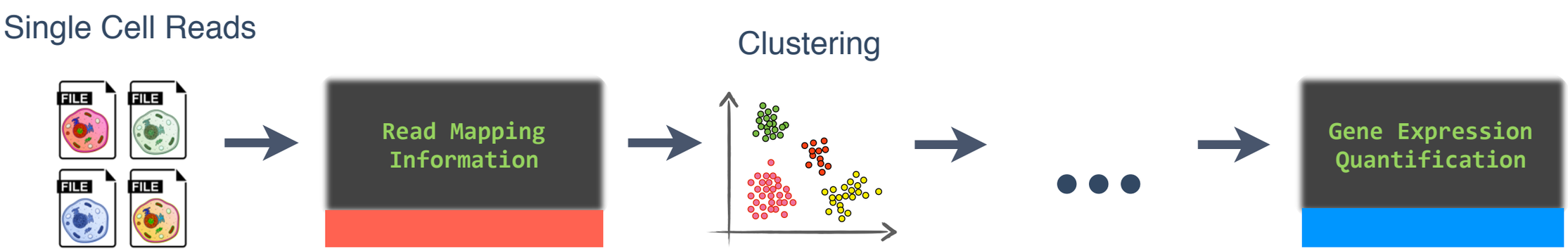
Gene expression levels

Big Picture

Standard analysis workflow:



Our proposed workflow:

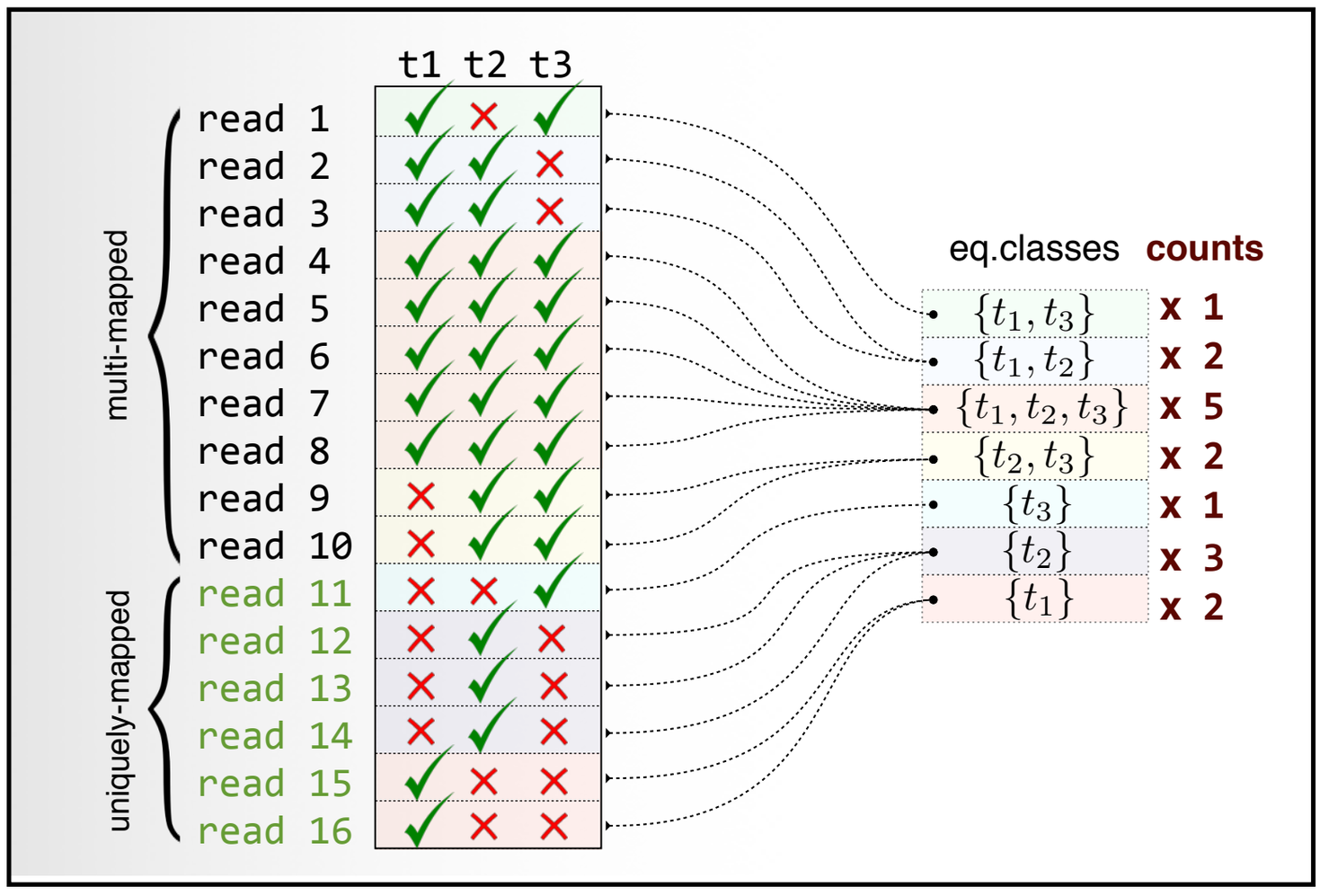


- Ntranos, V., Kamath, G.M., Zhang, J., Pachter, L., Tse, D.,N., **"Fast and accurate single-cell RNA-seq analysis by clustering of transcript-compatibility counts,"** **Genome Biology, May 2016**

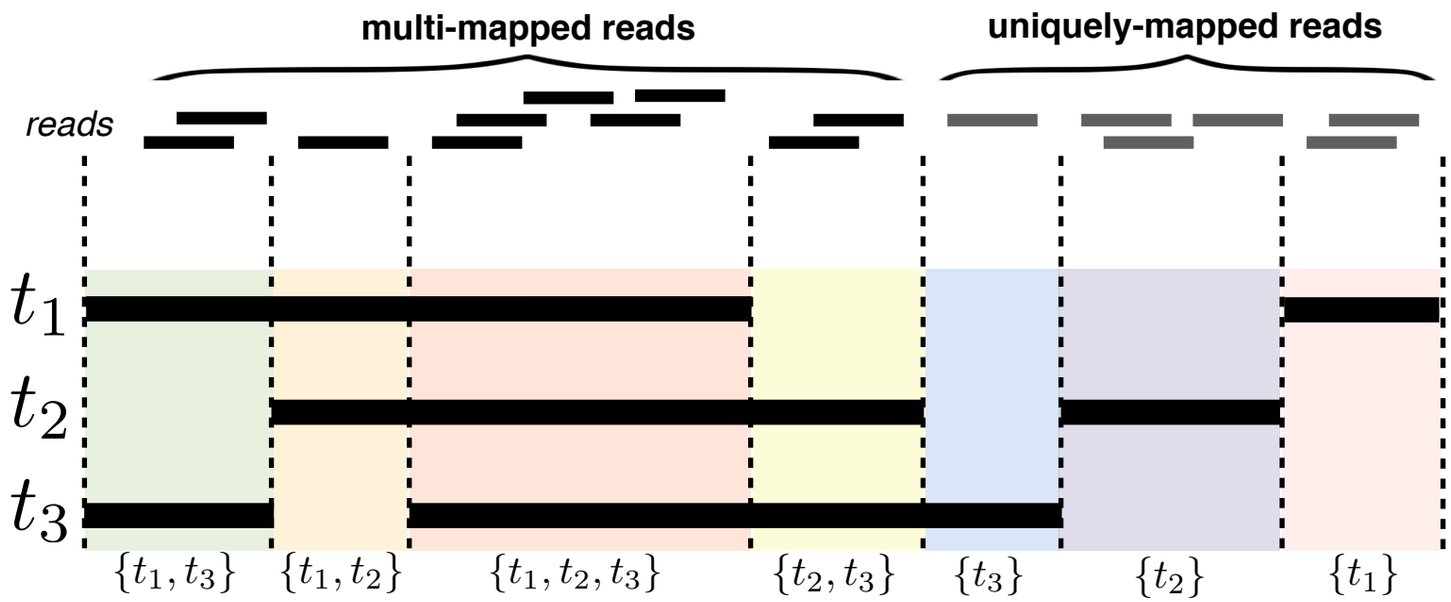
Equivalence Classes and Counts

Definitions:

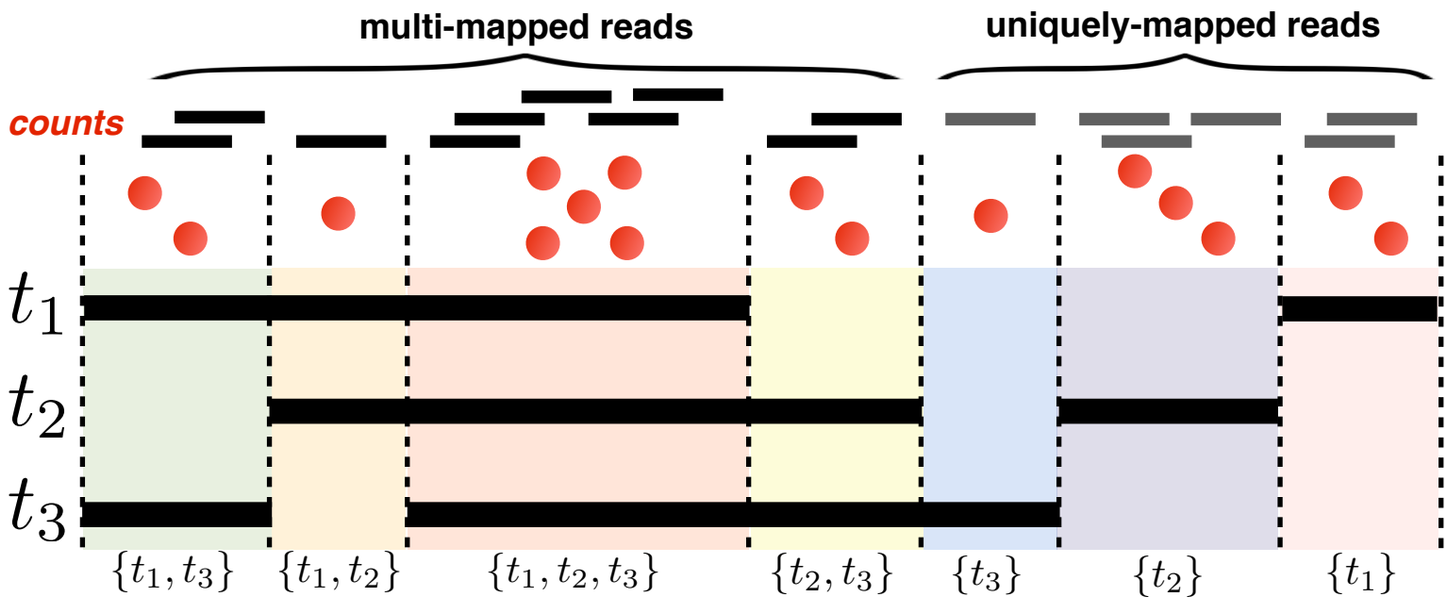
- Each read is *compatible* with one or more transcripts (i.e., the transcripts it could possibly have come from).
- An **equivalence class** is a group of reads that are *compatible* with the same set of transcripts.
- The **transcript-compatibility counts** are the *total number of reads* in each equivalence class.



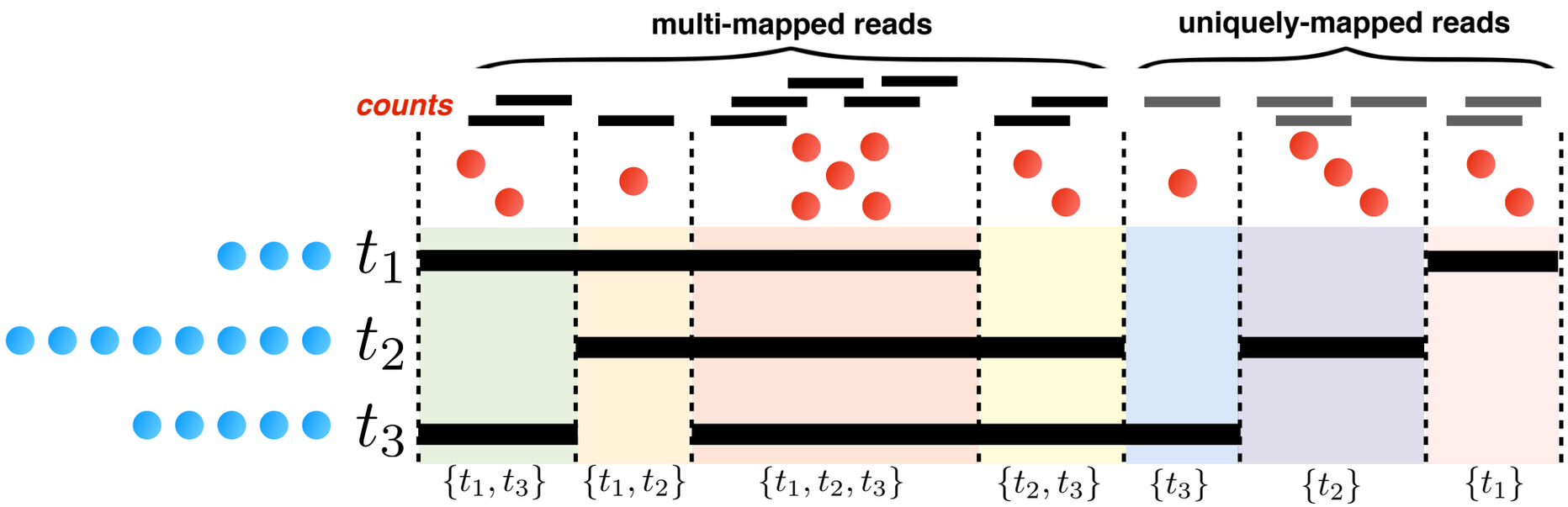
Pseudoalignment



Pseudoalignment



TCC Model



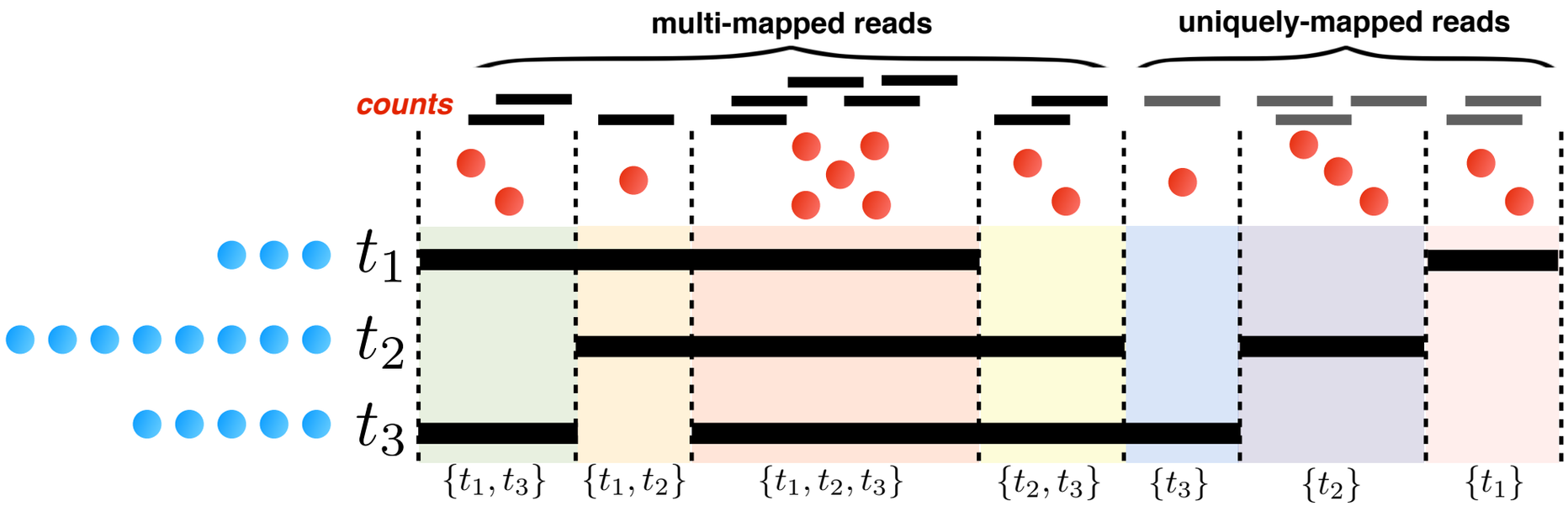
Underlying Model (ground truth):

$$p^*_t = \quad \text{transcripts}$$

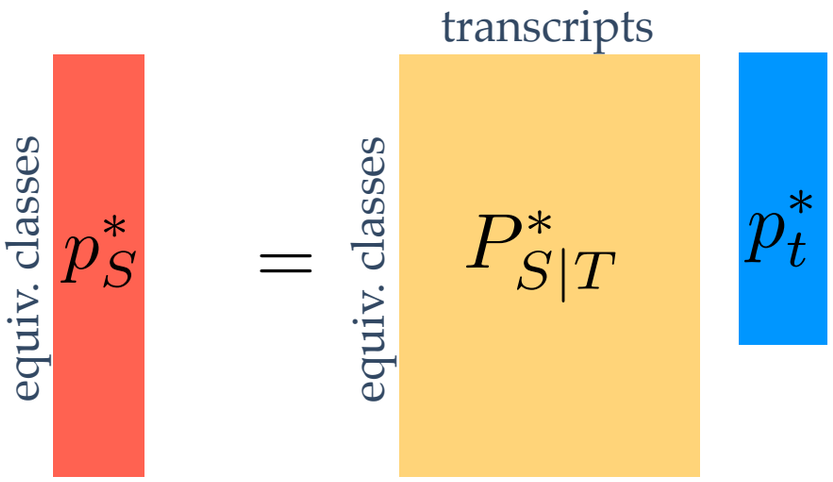
$$P^*_{S|T} = \quad \text{equiv. classes} \times \text{transcripts}$$

$$p^*_S = \quad \text{equiv. classes}$$

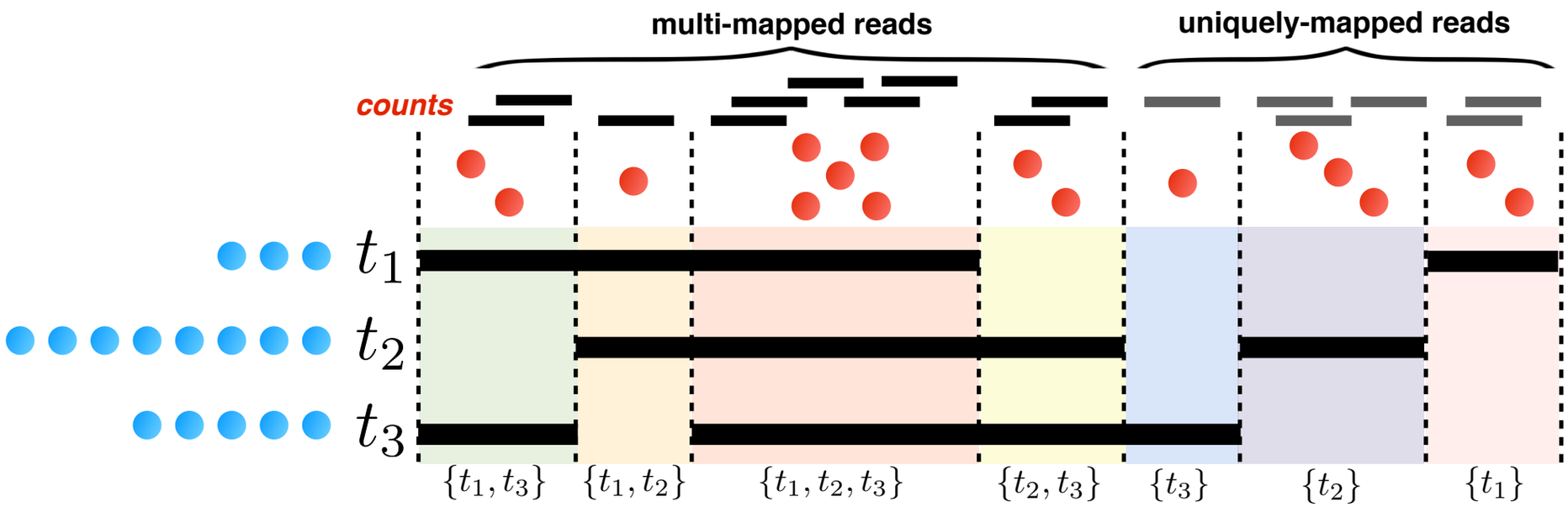
TCC Model



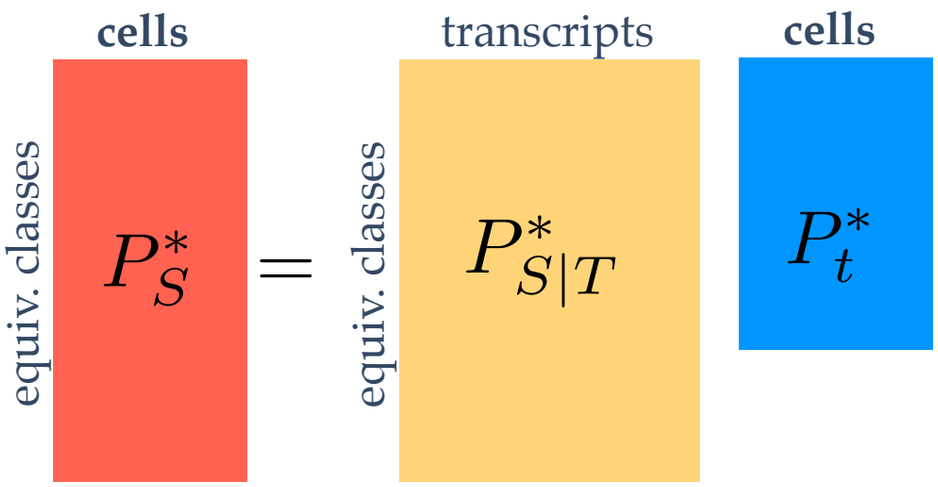
Underlying Model (ground truth):



TCC Model

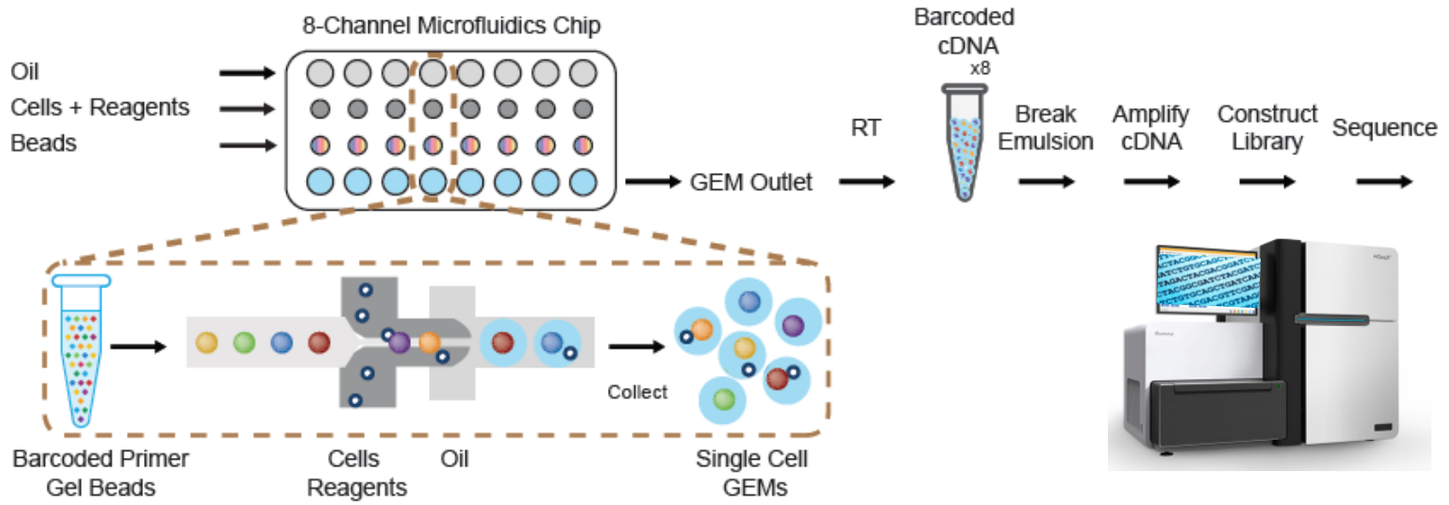


Underlying Model (ground truth):



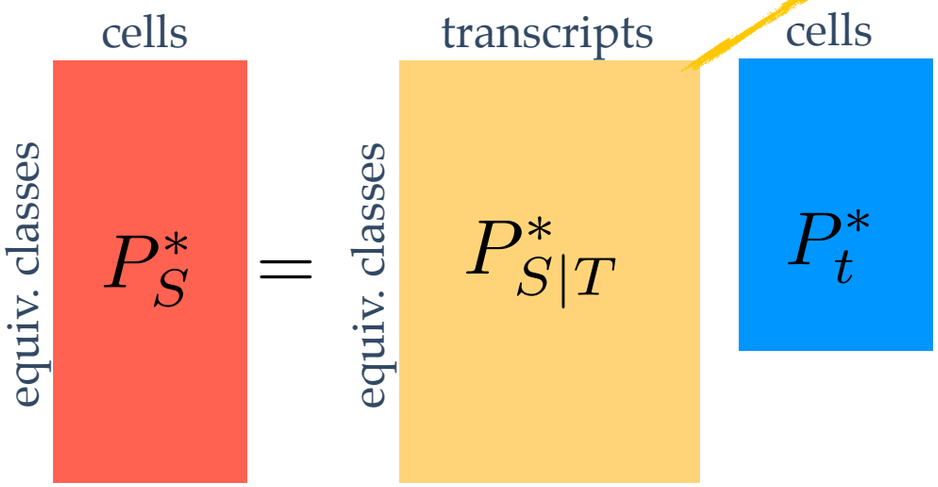
TCC Model

for example:



10x Chromium

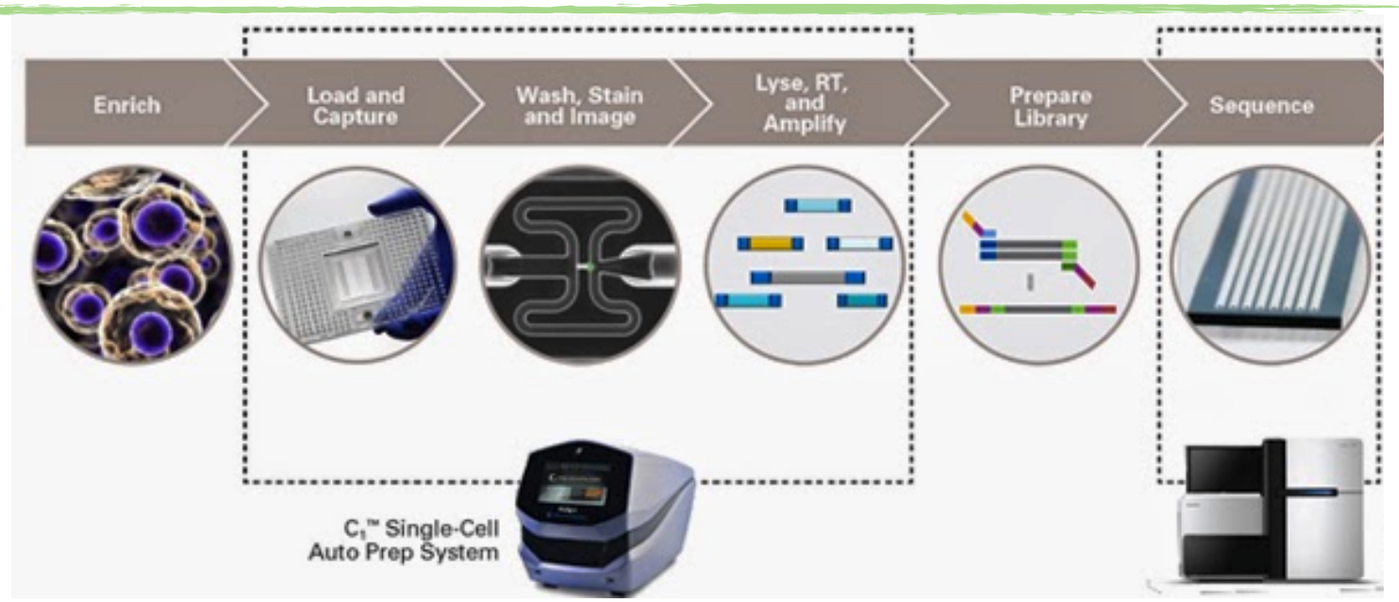
Underlying Model (ground truth):



TCC Model

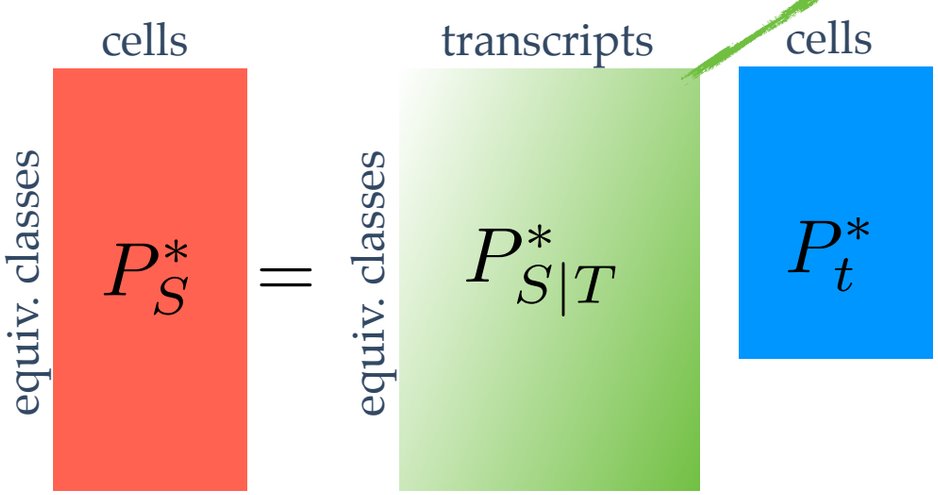
for example:

FLUIDIGM



Fluidigm C1

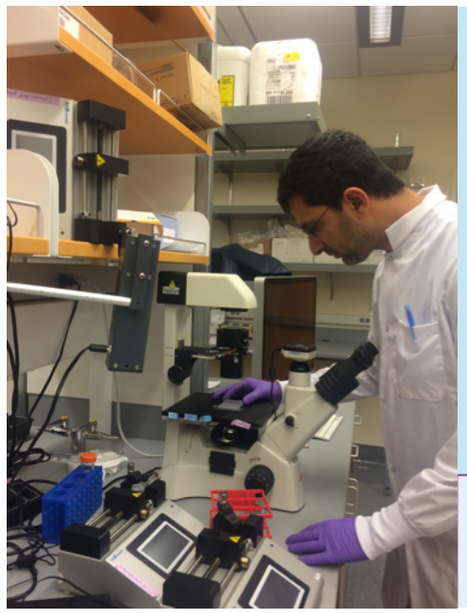
Underlying Model (ground truth):

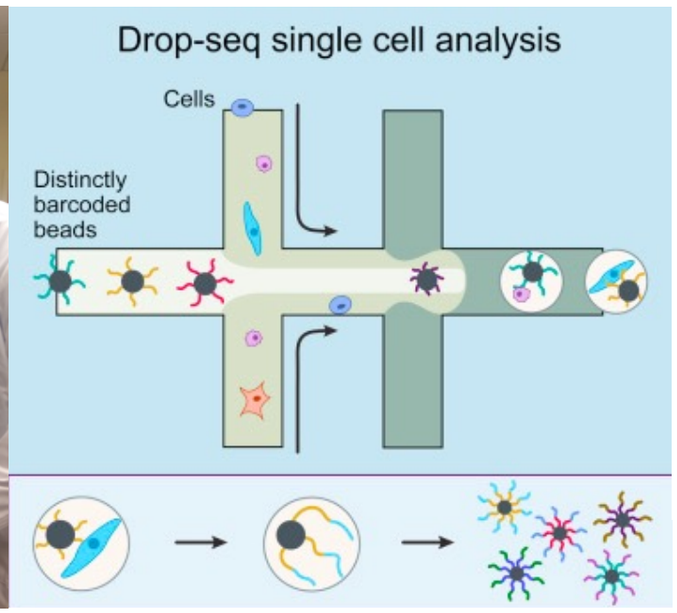


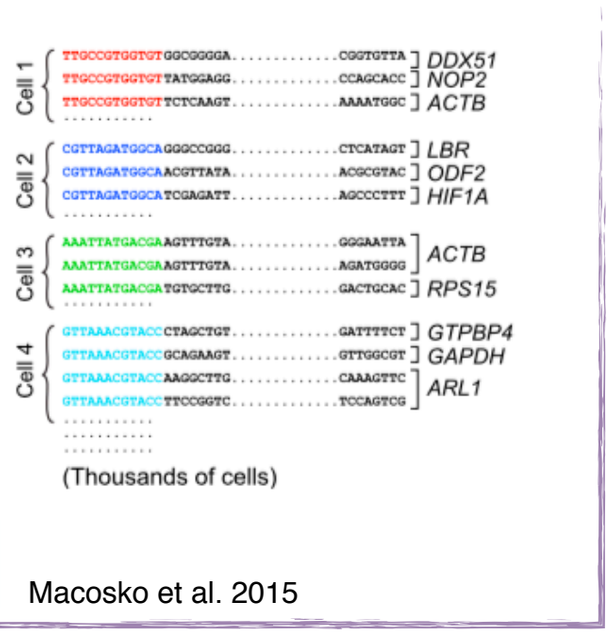
TCC Model*

for example:

Drop-seq







Macosko et al. 2015

Drop-Seq

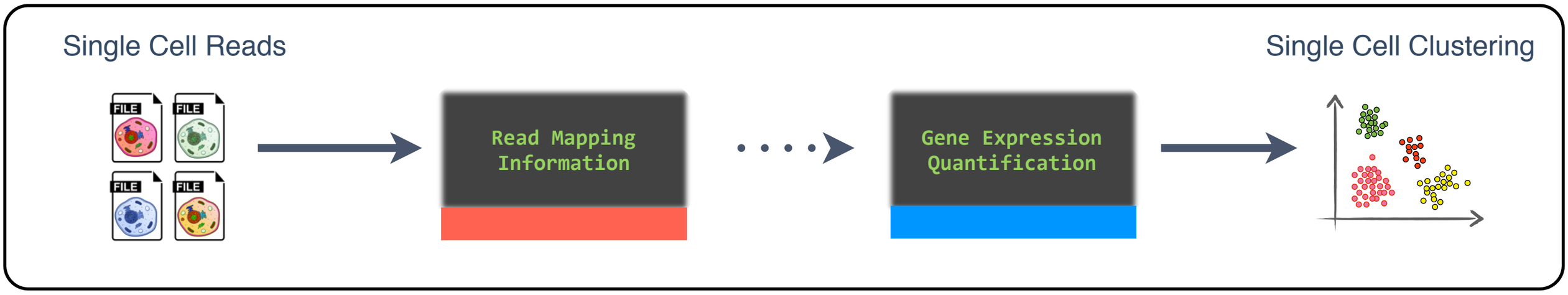
Underlying Model (ground truth):

$$P_S^* = P_{S|T}^* P_t^*$$

equiv. classes × cells = equiv. classes × transcripts × cells

*Disclaimer: All characters and events in this slide, even those based on real people, are entirely fictional.

TCC — Single-cell processing workflow



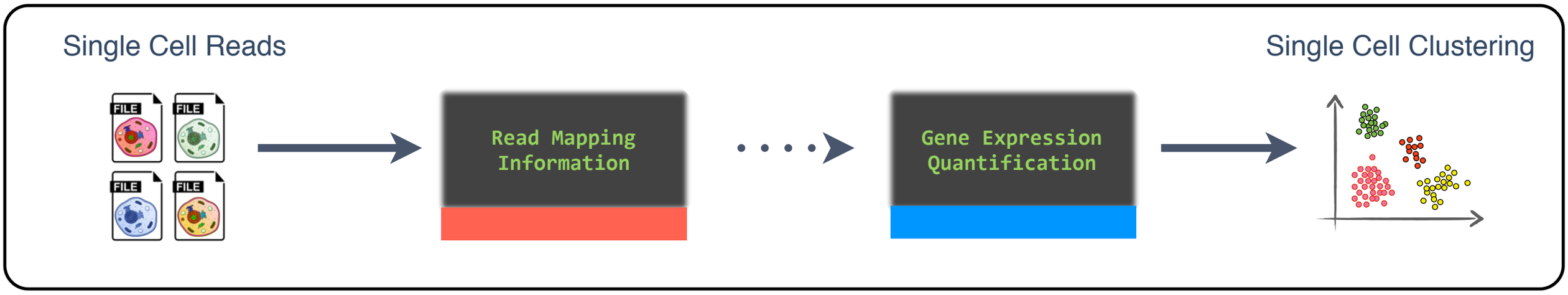
main idea

"similar" cell types should have "similar" eq.class counts

isoform / gene expression profiles

$$\underbrace{P_S^*}_{\text{cells} \times \text{equiv. classes}} = \underbrace{P_{S|T}^*}_{\text{transcripts} \times \text{equiv. classes}} \underbrace{P_t^*}_{\text{cells}}$$

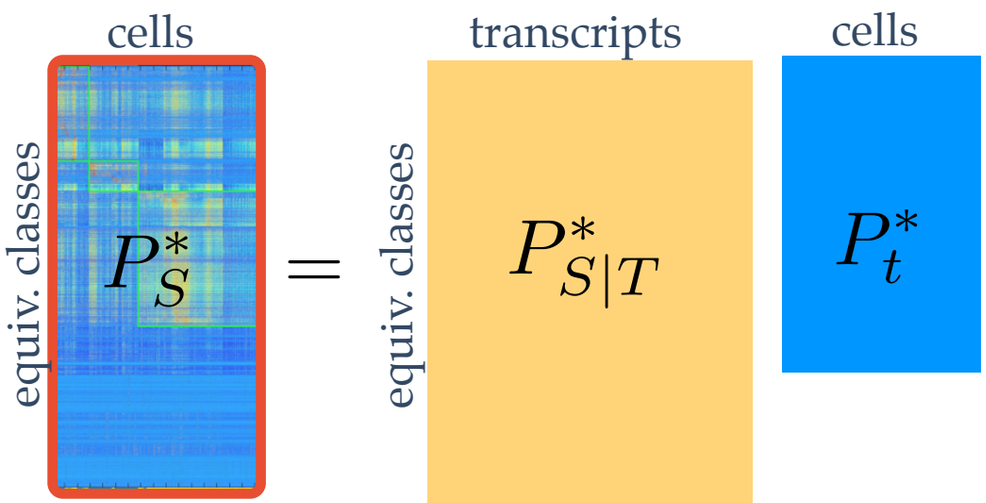
TCC — Single-cell processing workflow



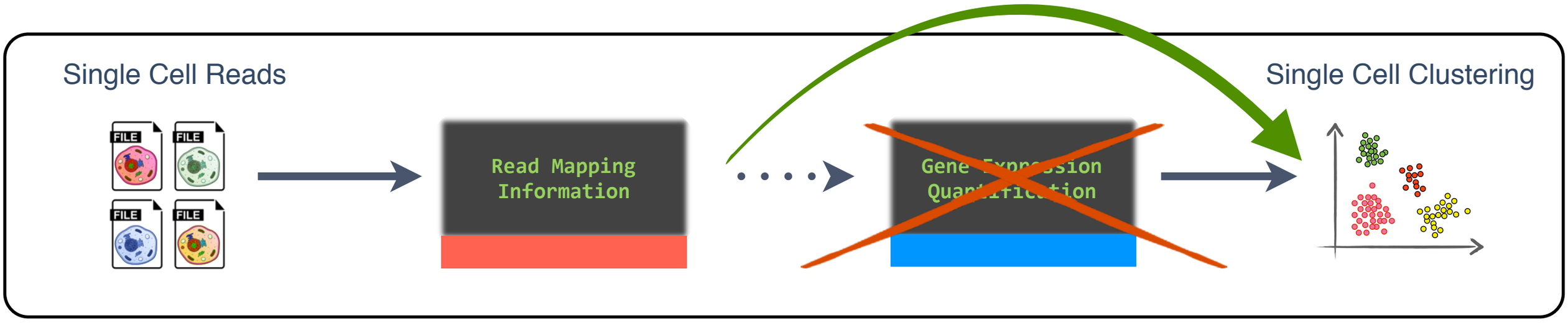
main idea

“similar” cell types should have “similar” eq.class counts

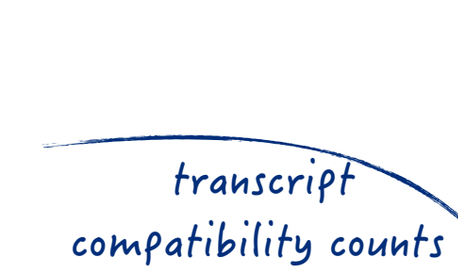
transcript compatibility counts



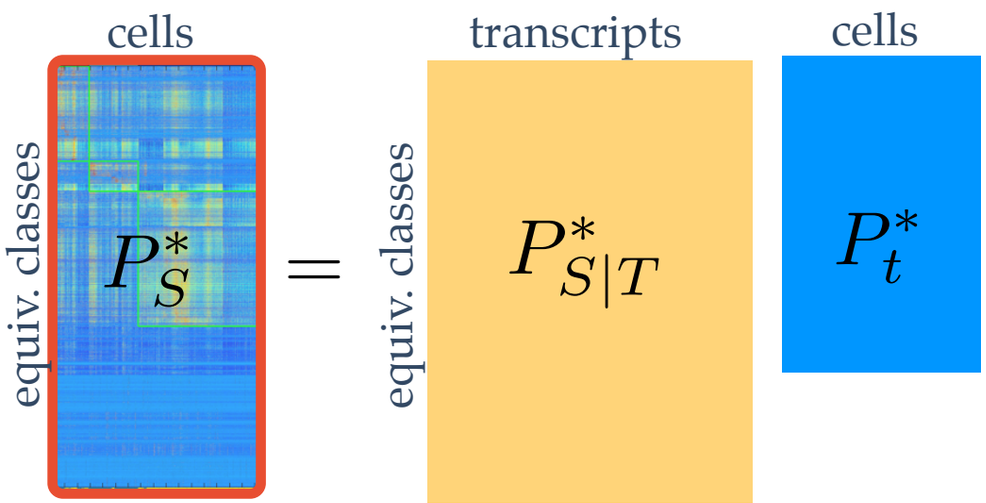
TCC — Single-cell processing workflow



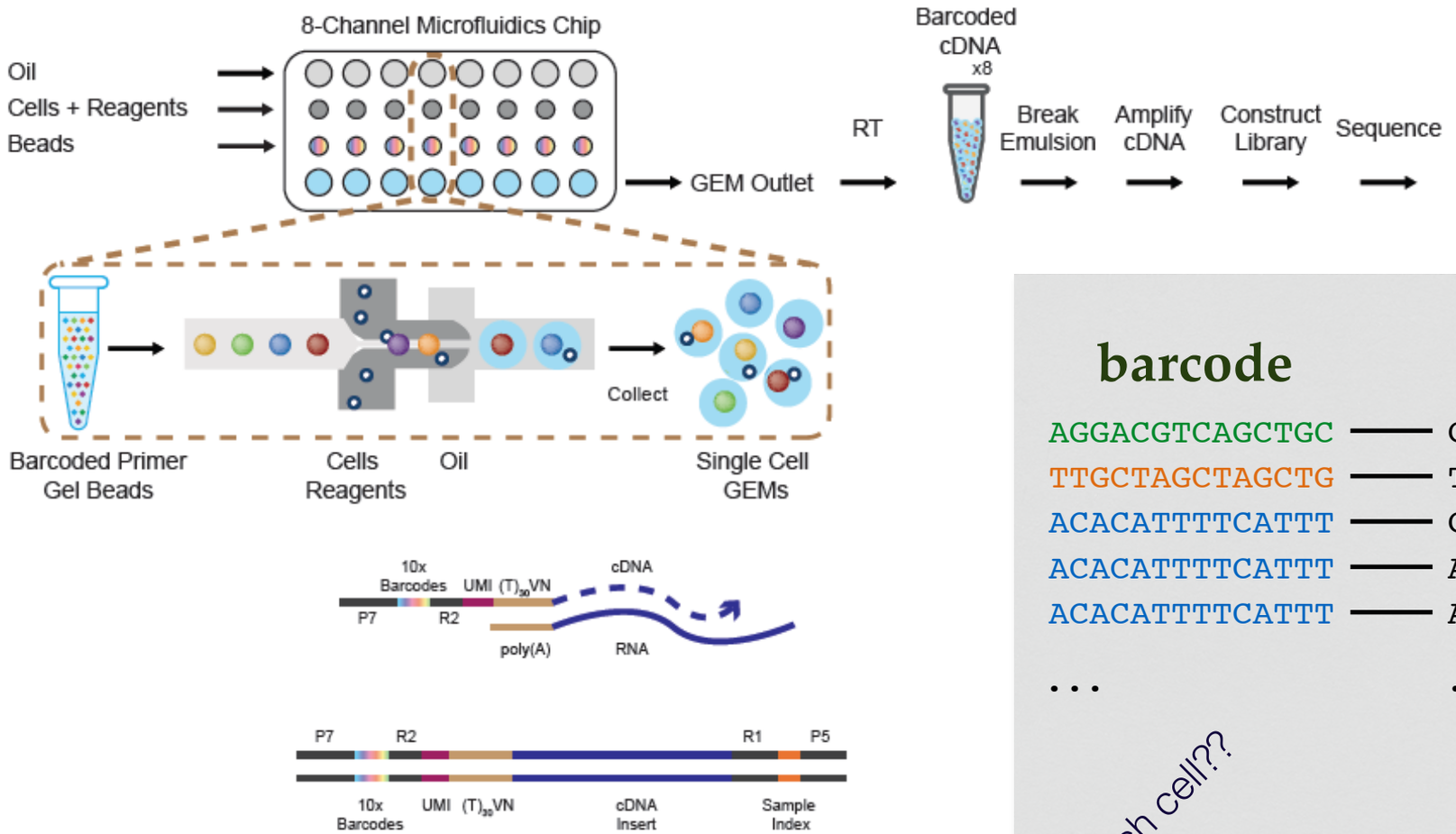
main idea



"similar" cell types should have "similar" eq.class counts



Overview — 10x Chromium data



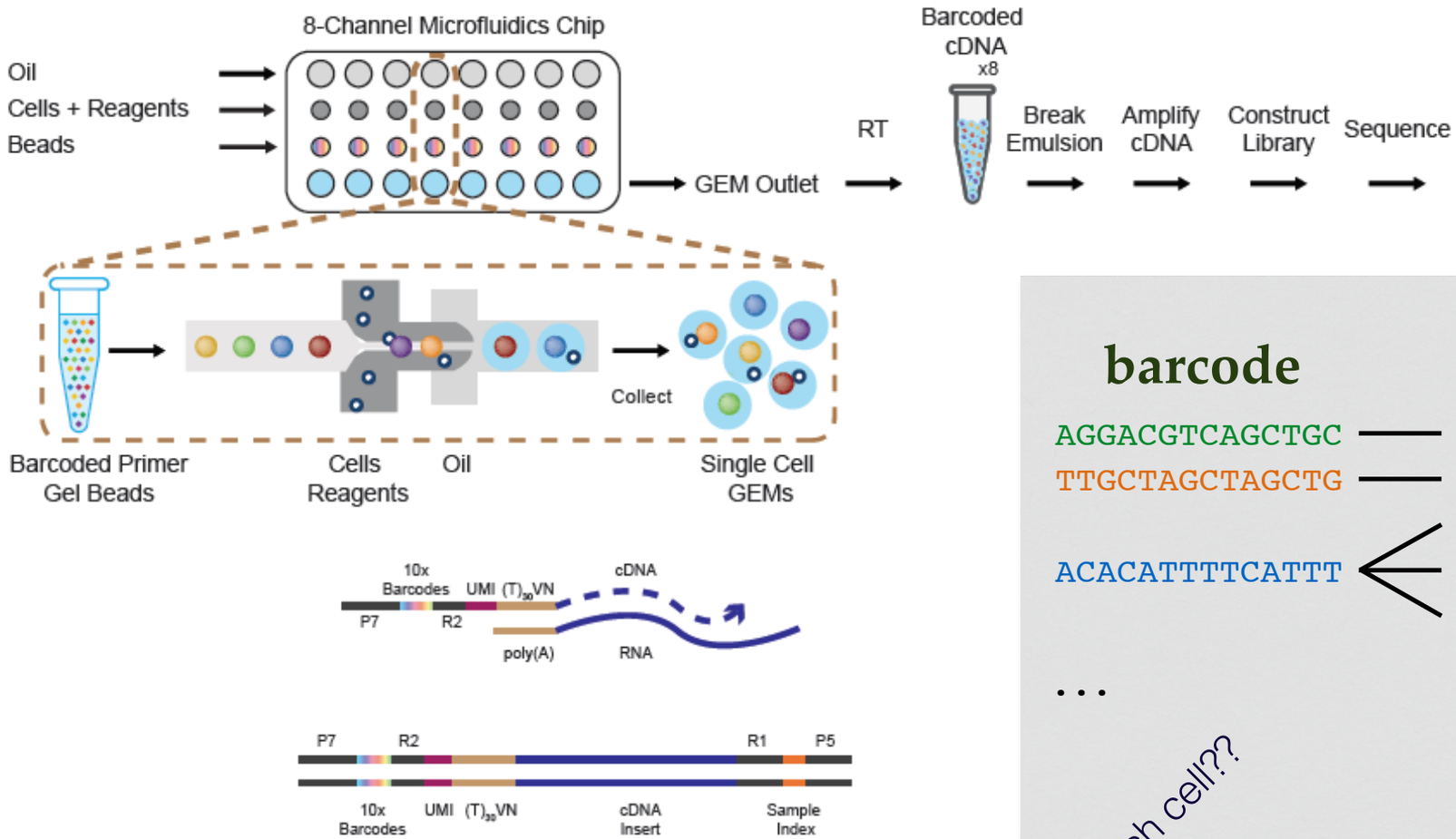
barcode	umi	read
AGGACGTCAGCTGC	GACATAGTGC	GTAAAGAGCGA.....
TTGCTAGCTAGCTG	TCATCGGCAT	AGCGGGCGGCG.....
ACACATTTTCATTT	GCAGTGCTGA	CGCGGCACCGA.....
ACACATTTTCATTT	AACGTCGCCC	ACAACACGGGG.....
ACACATTTTCATTT	AACGTCGCCC	CAACACGGGGG.....
...	...	...

which cell??

which molecule??

which transcript??

Overview — 10x Chromium data



barcode	umi	read
AGGACGTCAGCTGC	GACATAGTGC	GTAAAGAGCGA.....
TTGCTAGCTAGCTG	TCATCGGCAT	AGCGGGCGGCG.....
ACACATTTTCATTT	GCAGTGCTGA	CGCGGCACCGA.....
	AACGTCGCCC	ACAACACGGGG.....
	AACGTCGCCC	CAACACGGGGG.....
...	...	...

which cell??

which molecule??

which transcript??

TCC pipeline for 10x data

10X GENOMICS

`cellranger demux`

```
read-I1_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-I1_si-TTTCGCGA_lane-002-chunk-000.fastq.gz


read-RA_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-RA_si-TTTCGCGA_lane-002-chunk-000.fastq.gz
```

SI-3A-A10 (sample index) --> [ACAGCAAC, CGCAATTT, GAGTTGCG, TTTCGCGA]

github.com/pachterlab/scRNA-Seq-TCC-prep/

v0.2

Python scripts and jupyter notebooks to:

- Detect and Correct Cell Barcodes
- Generate single-cell fastqs
- Get distinct UMI-TCC matrix
- Prep for Clustering and Analysis

TCC pipeline for 10x data

10x GENOMICS

`cellranger demux`

```
read-I1_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-I1_si-TTTCGCGA_lane-002-chunk-000.fastq.gz


read-RA_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-RA_si-TTTCGCGA_lane-002-chunk-000.fastq.gz
```

SI-3A-A10 (sample index) --> [ACAGCAAC, CGCAATTT, GAGTTGCG, TTTCGCGA]

github.com/pachterlab/scRNA-Seq-TCC-prep/

Python scripts and jupyter notebooks to:

- Detect and Correct Cell Barcodes
- Generate single-cell Fastqs
- Get distinct UMI-TCC matrix
- Prep for Clustering and Analysis

notebooks/

10xGet_cell_barcodes.ipynb

10xResults.ipynb

source/

10xPrep_Data.py

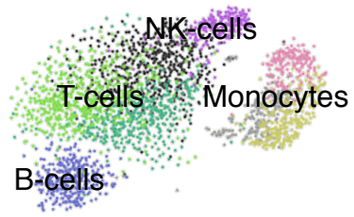
error_correct_and_split.py

compute_TCCs.py

prep_TCC_matrix.py

Step 0: Find a 10xDataset, e.g.,

Peripheral Blood Mononuclear Cells (PBMCs) from a Healthy Donor



Step 1: In your project dir, copy the **source/** files and the two notebooks and edit a **config.json** file:

```
{
    "NUM_THREADS": 8,
    "WINDOW": [500, 5000],
    "SOURCE_DIR": "/path/to/source/",
    "BASE_DIR": "/path/to/fastq_files/",
    "sample_idx": ["ATCGCTCC","CCGTACAG","GATAGGTA","TGACTAGT"],
    "SAVE_DIR": "/output/dir/for/save_data/",
    "dmin": 4,
    "BARCODE_LENGTH": 14,
    "OUTPUT_DIR": "/output/dir/for/singlecell_fastqs/",
    "kallisto":{
        "binary": "/path/to/kallisto",
        "index": "/path/to/kallisto_index.idx",
        "TCC_output" : "/output/dir/for/TCC_output/"
    }
}
```

TCC pipeline for 10x data

10X GENOMICS

cellranger demux

```
read-I1_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-I1_si-TTTCGCGA_lane-002-chunk-000.fastq.gz


read-RA_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-RA_si-TTTCGCGA_lane-002-chunk-000.fastq.gz
```

SI-3A-A10 (sample index) --> [ACAGCAAC, CGCAATTT, GAGTTGCG, TTTCGCGA]

github.com/pachterlab/scRNA-Seq-TCC-prep/

v0.2

Python scripts and jupyter notebooks to:

- Detect and Correct Cell Barcodes
- Generate single-cell Fastqs
- Get distinct UMI-TCC matrix
- Prep for Clustering and Analysis

notebooks/

10xGet_cell_barcodes.ipynb

10xResults.ipynb

source/

10xPrep_Data.py

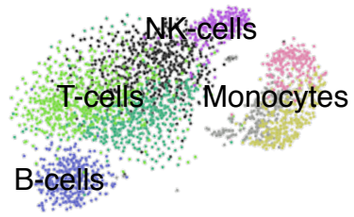
error_correct_and_split.py

compute_TCCs.py

prep_TCC_matrix.py

Step 0: Find a 10xDataset, e.g.,

Peripheral Blood Mononuclear Cells (PBMCs) from a Healthy Donor



Step 1: In your project dir, copy the **source/** files and the two notebooks and edit a **config.json** file:

TCC pipeline:

10xGet_cell_barcodes.ipynb

`python source/`**`10xPrep_Data.py`** `config.json`

10xResults.ipynb

TCC pipeline for 10x data

10X GENOMICS

```
cellranger demux
```

```
read-I1_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-I1_si-TTTCGCGA_lane-002-chunk-000.fastq.gz

read-RA_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-RA_si-TTTCGCGA_lane-002-chunk-000.fastq.gz
```

SI-3A-A10 (sample index) --> [ACAGCAAC, CGCAATTT, GAGTTGCG, TTTCGCGA]

github.com/pachterlab/scRNA-Seq-TCC-prep/

v0.2

Python scripts and jupyter notebooks to:

- Detect and Correct Cell Barcodes
- Generate single-cell Fastqs
- Get distinct UMI-TCC matrix
- Prep for Clustering and Analysis

notebooks/

10xGet_cell_barcodes.ipynb

10xResults.ipynb

source/

10xPrep_Data.py

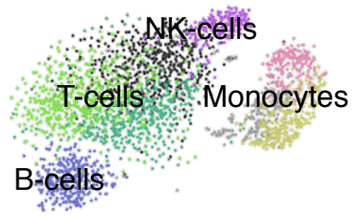
error_correct_and_split.py

compute_TCCs.py

prep_TCC_matrix.py

Step 0: Find a 10xDataset, e.g.,

Peripheral Blood Mononuclear Cells (PBMCs) from a Healthy Donor



Step 1: In your project dir, copy the **source/** files and the two notebooks and edit a **config.json** file:

TCC pipeline:

10xGet_cell_barcodes.ipynb

```
python source/error_correct_and_split.py config.json
python source/compute_TCCs.py config.json
python source/prep_TCC_matrix.py config.json
```

10xResults.ipynb

TCC pipeline for 10x data

10x GENOMICS

`cellranger demux`

```
read-I1_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-I1_si-TTTCGCGA_lane-002-chunk-000.fastq.gz

read-RA_si-ACAGCAAC_lane-001-chunk-001.fastq.gz
...
read-RA_si-TTTCGCGA_lane-002-chunk-000.fastq.gz
```

SI-3A-A10 (sample index) --> [ACAGCAAC, CGCAATTT, GAGTTGCG, TTTCGCGA]

github.com/pachterlab/scRNA-Seq-TCC-prep/

v0.2

Python scripts and jupyter notebooks to:

- Detect and Correct Cell Barcodes
- Generate single-cell Fastqs
- Get distinct UMI-TCC matrix
- Prep for Clustering and Analysis

notebooks/

10xGet_cell_barcodes.ipynb

10xResults.ipynb

source/

10xPrep_Data.py

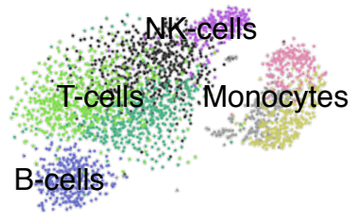
error_correct_and_split.py

compute_TCCs.py

prep_TCC_matrix.py

Step 0: Find a 10xDataset, e.g.,

Peripheral Blood Mononuclear Cells (PBMCs) from a Healthy Donor



Step 1: In your project dir, copy the **source/** files and the two notebooks and edit a **config.json** file:

TCC pipeline:

10xGet_cell_barcodes.ipynb

`python source/`**`error_correct_and_split.py`** `config.json`

generate single cell files

`python source/`**`compute_TCCs.py`** `config.json`

kallisto pseudoalignment of all cells

`python source/`**`prep_TCC_matrix.py`** `config.json`

10xResults.ipynb

pwise dist, clustering and analysis