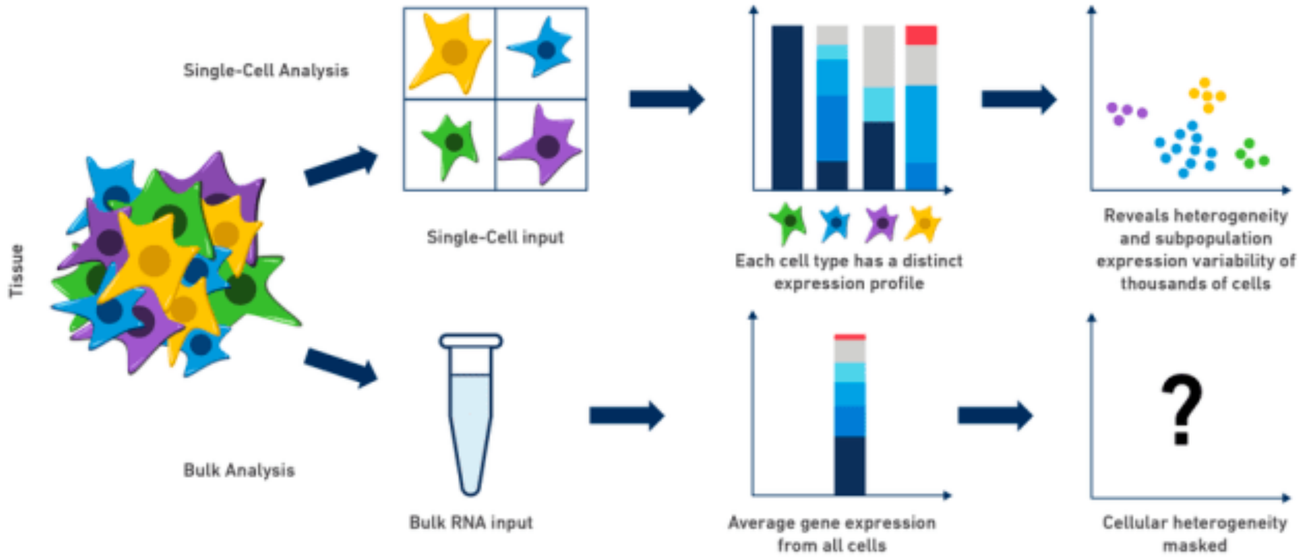
Principles of data analysis for single-cell genomics

MSc Bioinformatics

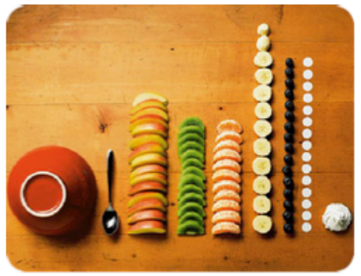
2020-31032-T1 IEO. Extracció Informació de Tecnològies OMIC

Ricard Argelaguet
31st May 2021

single-cell genomics allows for the deconvolution of homogeneous cell types from a heterogeneous tissue



bulk genomics

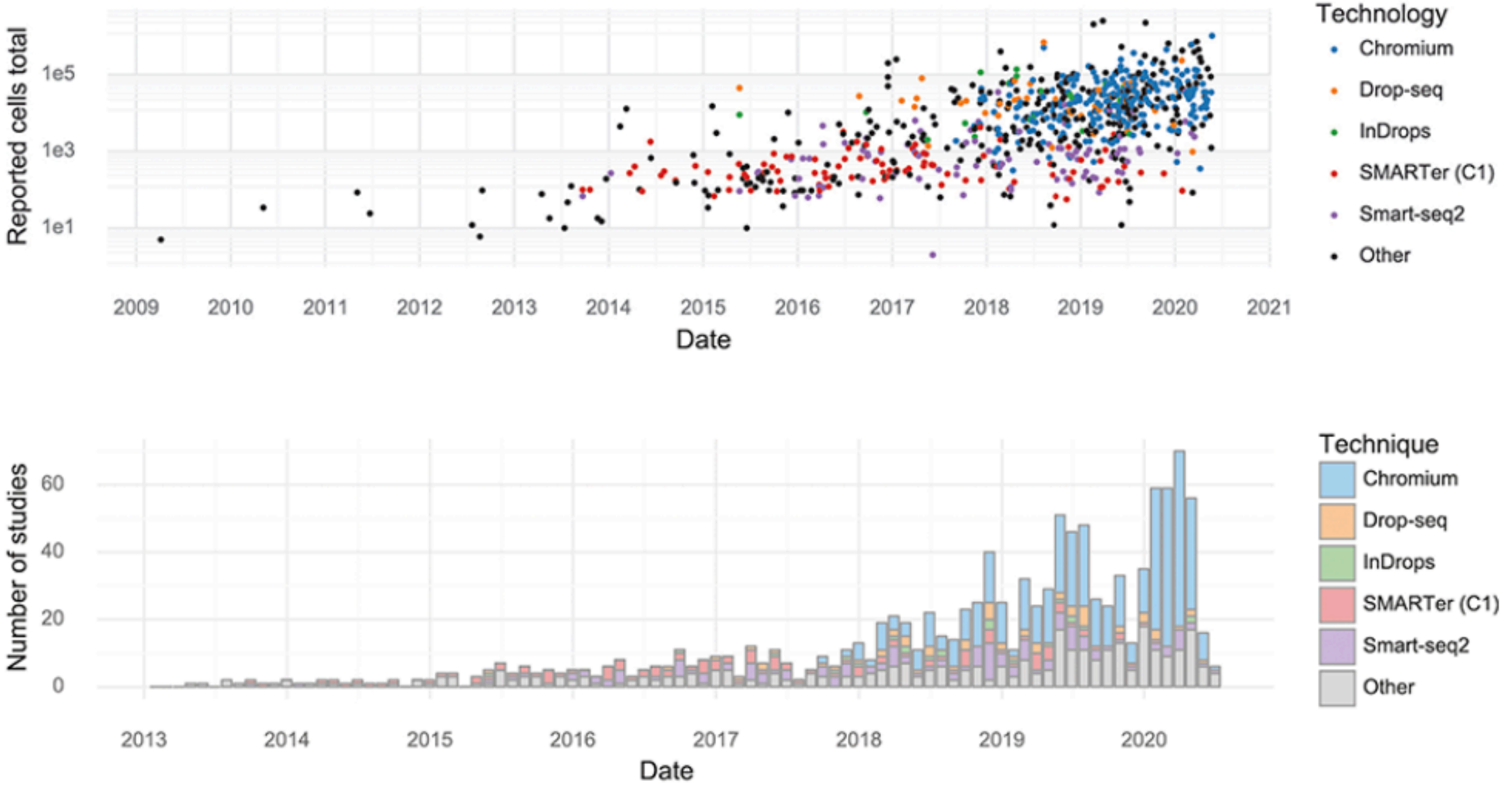


single-cell genomics

Figure extracted from 10xgenomics.com

In a single decade there has been an explosion of scRNAseq technologies

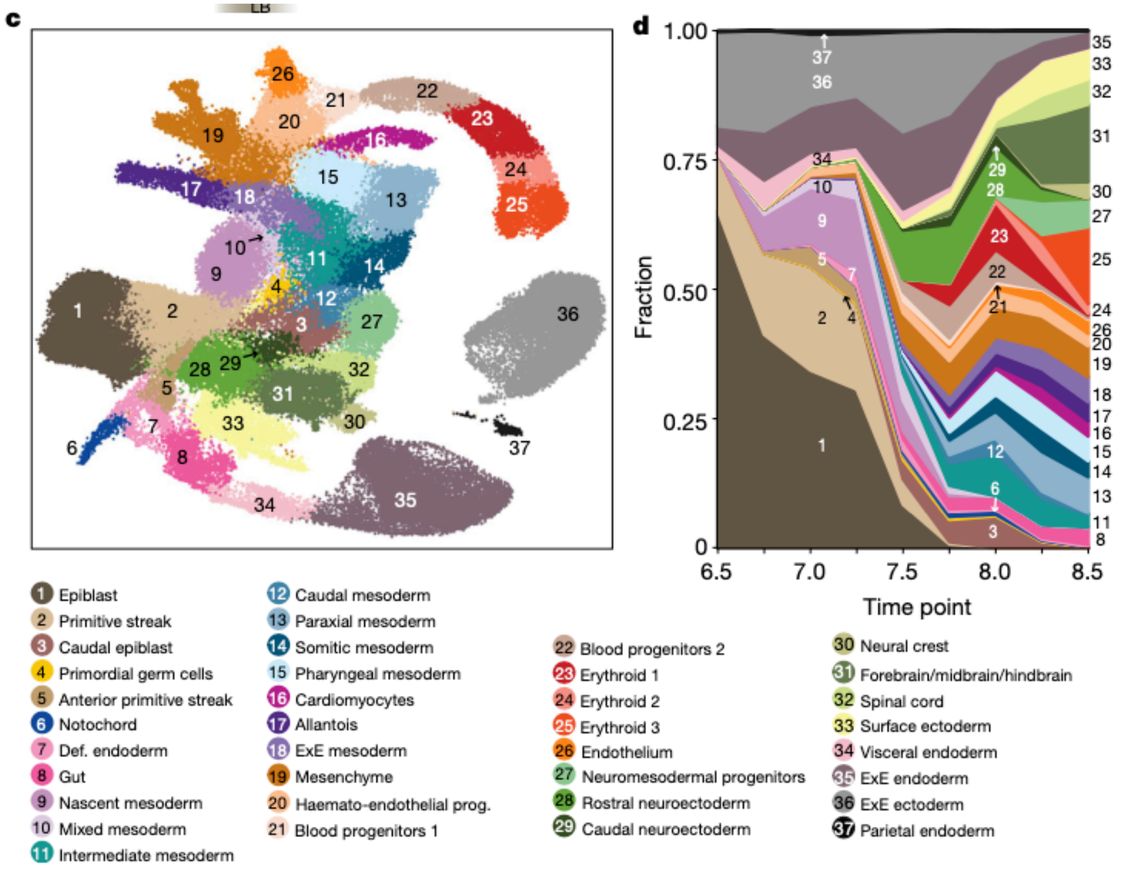
note the log scale!



Svensson et a. 2020
https://doi.org/10.1093/database/baaa073

A single-cell molecular map of mouse gastrulation and early organogenesis

Blanca Pijuan-Sala[1,2,11], Jonathan A. Griffiths[3,11], Carolina Guibentif[1,2,11], Tom W. Hiscock[3,4], Wajid Jawaid[1,2], Fernando J. Calero-Nieto[1,2], Carla Mulas[2], Ximena Ibarra-Soria[3], Richard C. V. Tyser[5], Debbie Lee Lian Ho[2], Wolf Reik[6,7,8], Shankar Srinivas[5], Benjamin D. Simons[2,4,9], Jennifer Nichols[2], John C. Marioni[3,8,10*] & Berthold Göttgens[1,2*]



E6.5

https://www.nature.com/articles/s41586-019-0933-9
https://marionilab.cruk.cam.ac.uk/MouseGastrulation2018/

A molecular single-cell lung atlas of lethal COVID-19

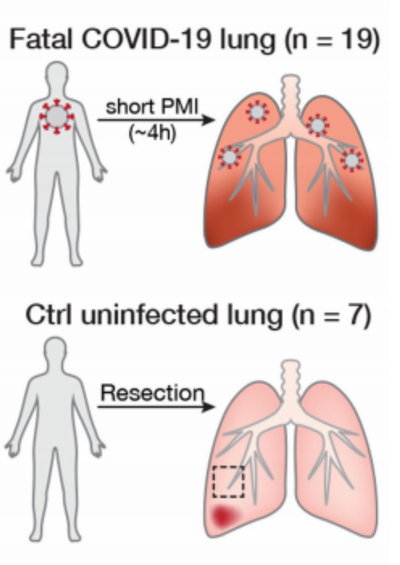
Received: 16 November 2020

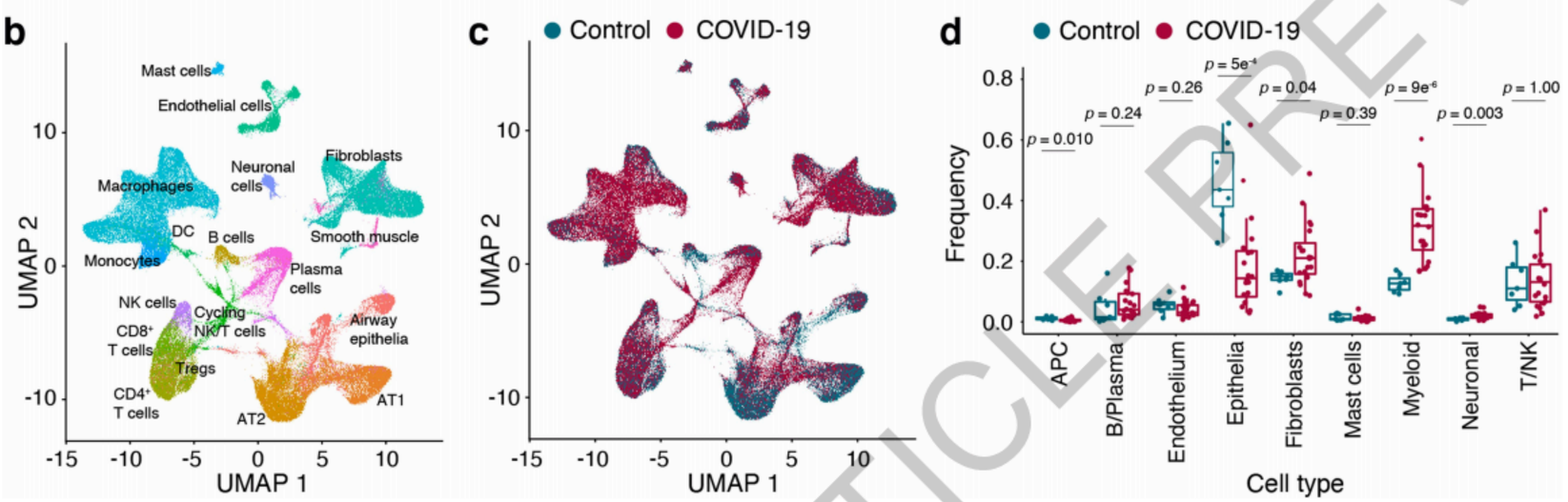
Accepted: 19 April 2021

Accelerated Article Preview Published online 29 April 2021

Cite this article as: Melms, J. C. et al. A molecular single-cell lung atlas of lethal COVID-19. *Nature* https://doi.org/10.1038/s41586-021-03569-1 (2021).

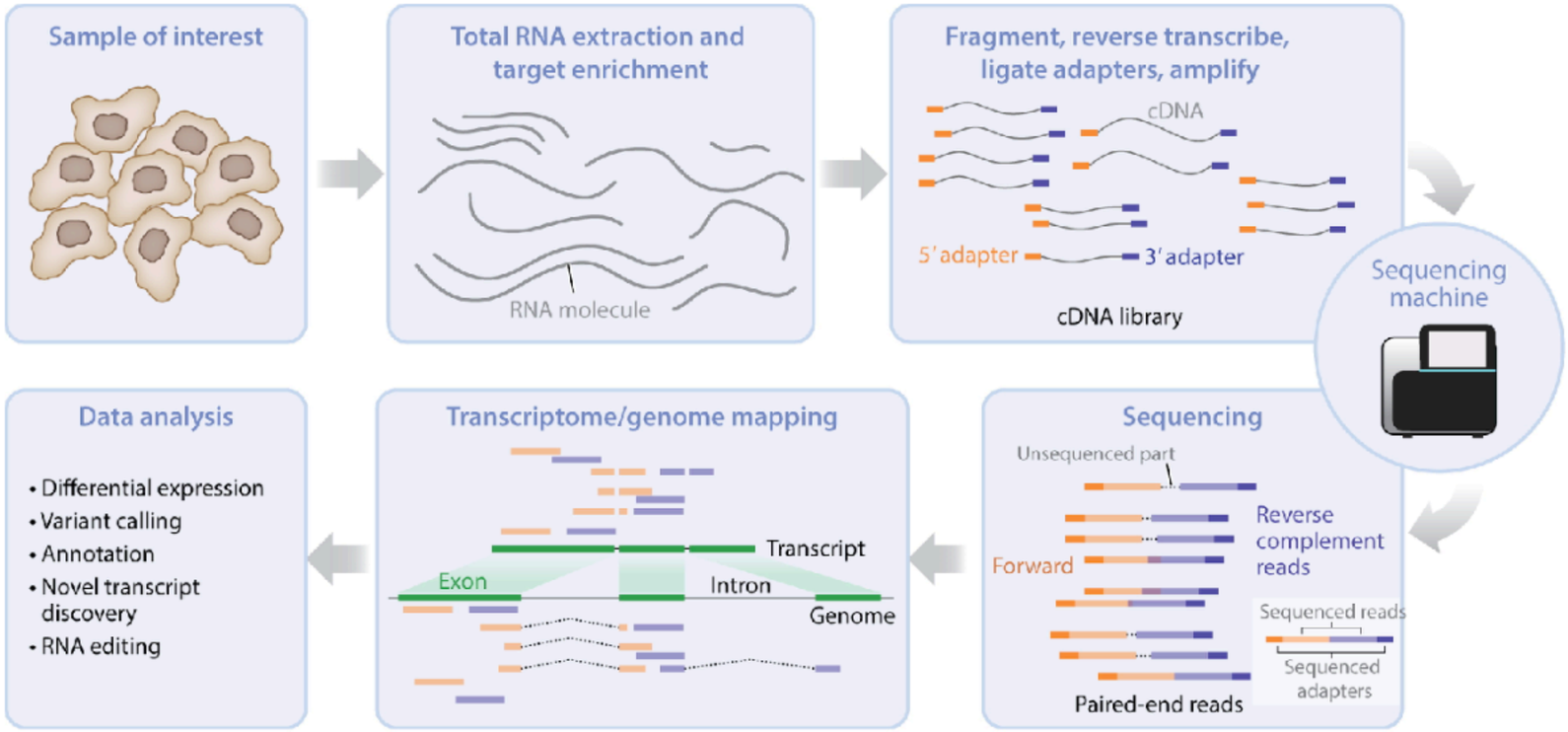
Johannes C. Melms, Jana Biermann, Huachao Huang, Yiping Wang, Ajay Nair, Somnath Tagore, Igor Katsyv, André F. Rendeiro, Amit Dipak Amin, Denis Schapiro, Chris J. Frangieh, Adrienne M. Luoma, Aveline Filliol, Yinshan Fang, Hiranmayi Ravichandran, Mariano G. Clausi, George A. Alba, Meri Rogava, Sean W. Chen, Patricia Ho, Daniel T. Montoro, Adam E. Kornberg, Arnold S. Han, Mathieu F. Bakhoum, Niroshana Anandasabapathy, Mayte Suárez-Fariñas, Samuel F. Bakhoum, Yaron Bram, Alain Borczuk, Xinzheng V. Guo, Jay H. Lefkowitch, Charles Marboe, Stephen M. Lagana, Armando Del Portillo, Emmanuel Zorn, Glen S. Markowitz, Robert F. Schwabe, Robert E. Schwartz, Olivier Elemento, Anjali Saqi, Hanina Hibshoosh, Jianwen Que & Benjamin Izar





Basic scRNA-seq pipeline

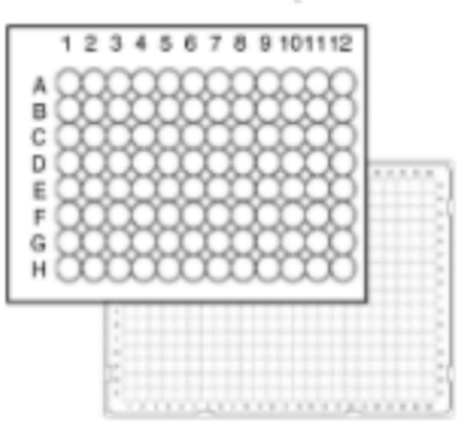
(before exploratory data analysis)



Van den Berge K, et al. 2019.
Annu. Rev. Biomed. Data Sci. 2:139–73

What are the key parameters that define technologies?

Plate-based

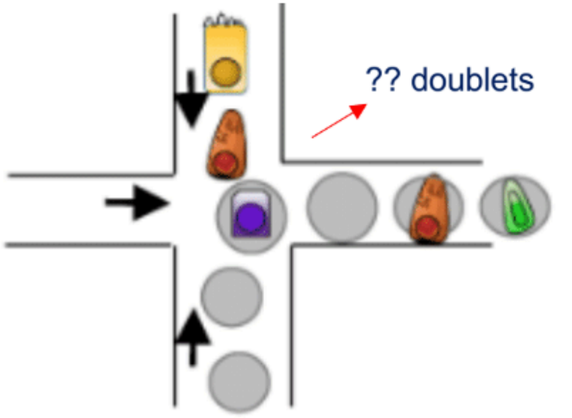


- Scalability ↓
- Sensitivity ↑
- Cost ↑

Sensitivity (genes/cell):
Cell lines: ~ 7k-10k
Primary tissues: ~2k-6k

Scalability (N cell/run):
1 cell at a time (it can be automated)

droplet-based



- Scalability ↑
- Sensitivity ↓
- Cost ↓

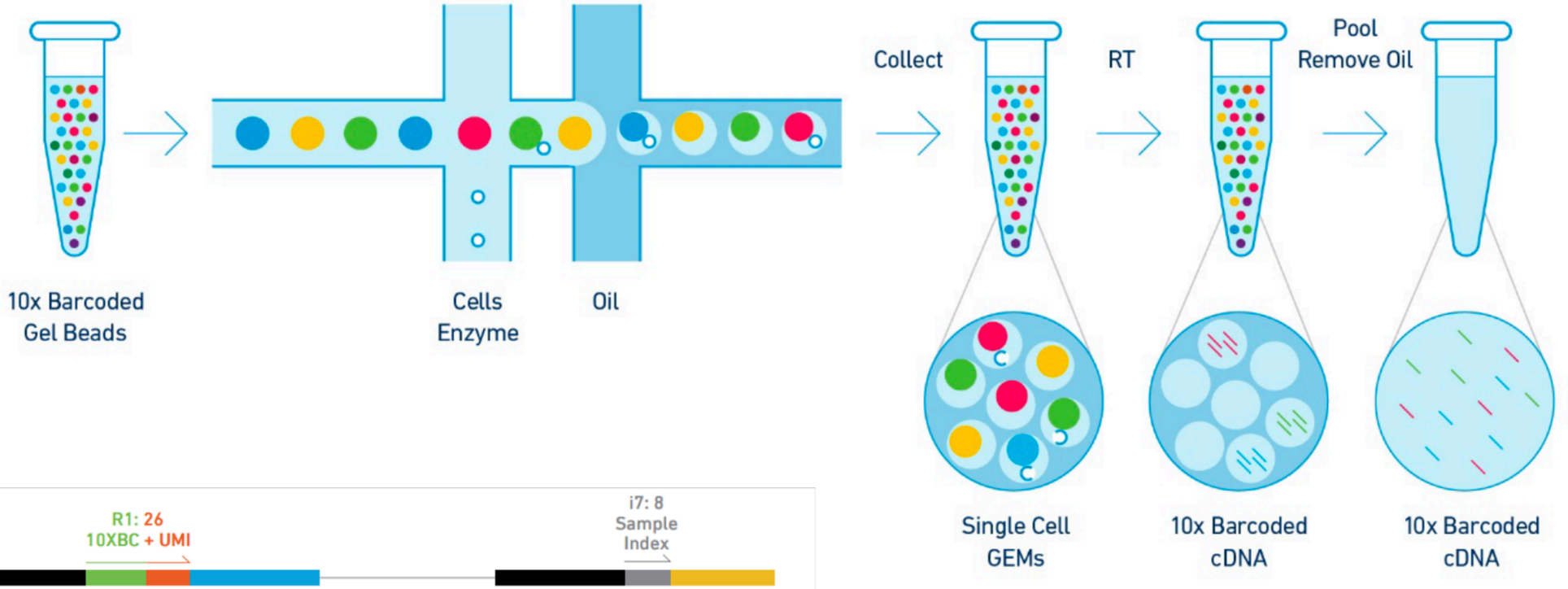
Sensitivity (genes/cell):
Cell lines: ~ 5k
Primary tissues: ~1k-3k

Scalability (N cell/run):
~ 50,000

From Satija

https://www.youtube.com/watch?v=vL7ptq2Dcf0 2:10

Overview of library preparation for scRNA-seq (10x Genomics)



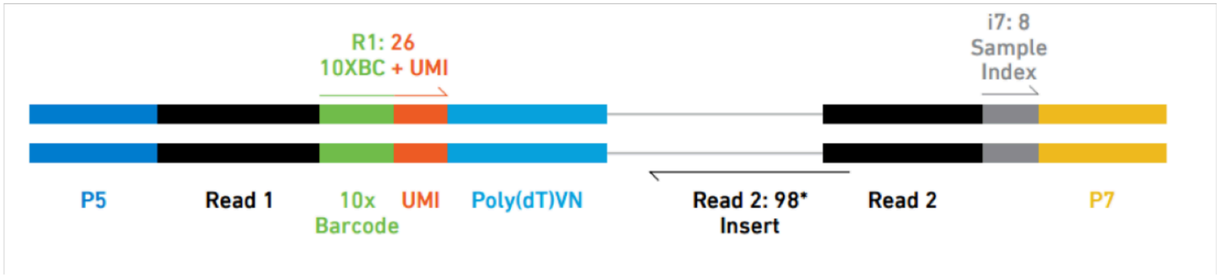
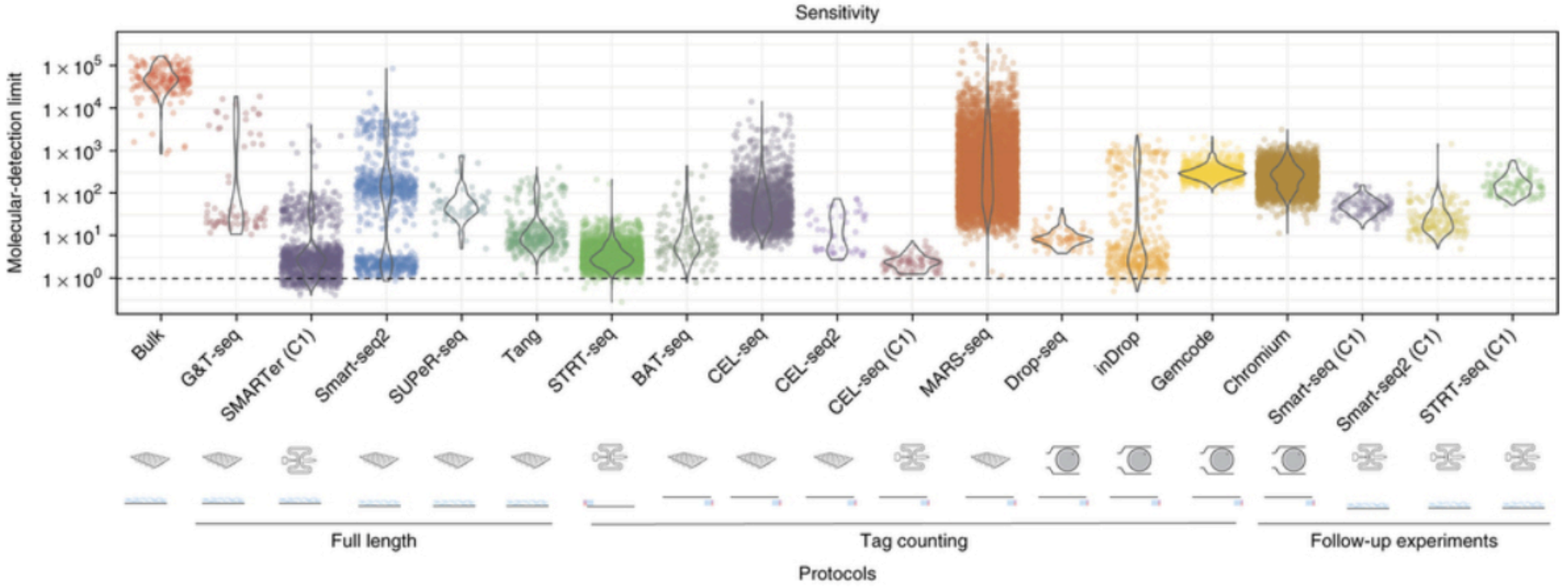


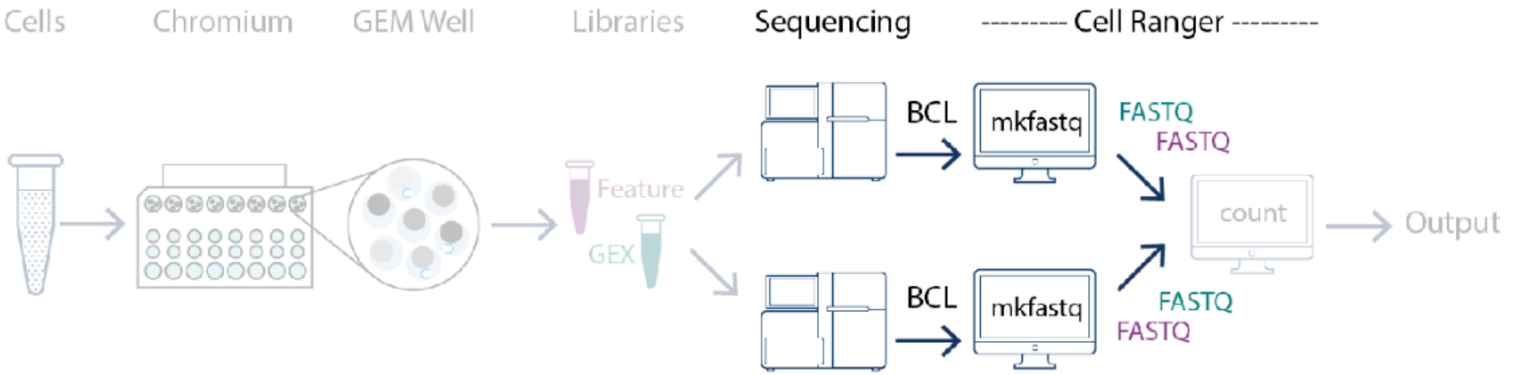
Fig. 2. Schematic of a fragment from a final Chromium™ Single Cell 3' v2 library. *Can be adjusted.

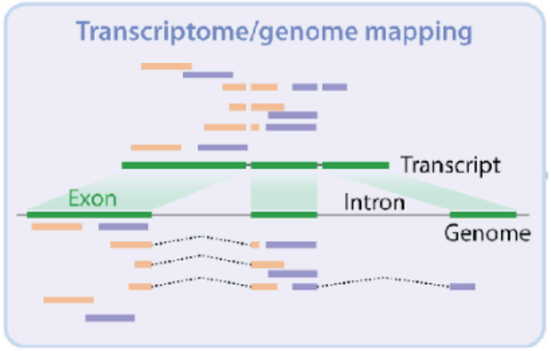
scRNA-seq technologies vary in sensitivity and scalability



scRNA-seq data processing

Step 1: sequencing and mapping





FASTQ file

```
@SRR014849.1 EIXKN4201CFU84 length=93
GGGGGGGGGGGGGGGGCTTTTTTTGTTTGGAACCGAAAGG
GTTTTGAATTTCAAACCCTTTTCGGTTTCCAACCTTCCAA
AGCAATGCCAATA
+SRR014849.1 EIXKN4201CFU84 length=93
3+&$#""""""""""""7F@71,'";C?,B;?6B;:EA1EA
1EA5'9B:?:#9EA0D@2EA5':>5?:%A;A8A;?9B;D@
/=<?7=9<2A8==
```

@title and optional description
sequence line(s)
+optional repeat of title line
quality line(s)

Van den Berge K, et al. 2019.
Annu. Rev. Biomed. Data Sci. 2:139–73

scRNA-seq data processing

Step 2: transcript quantification

Each column is a sample

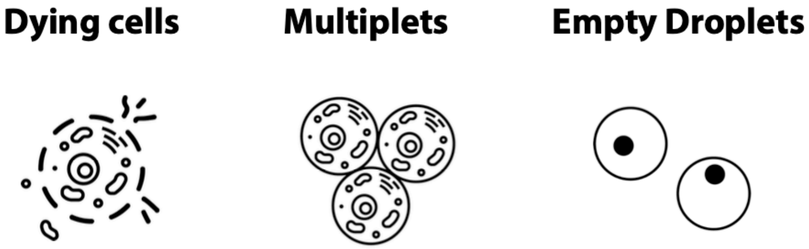
Each row is a gene

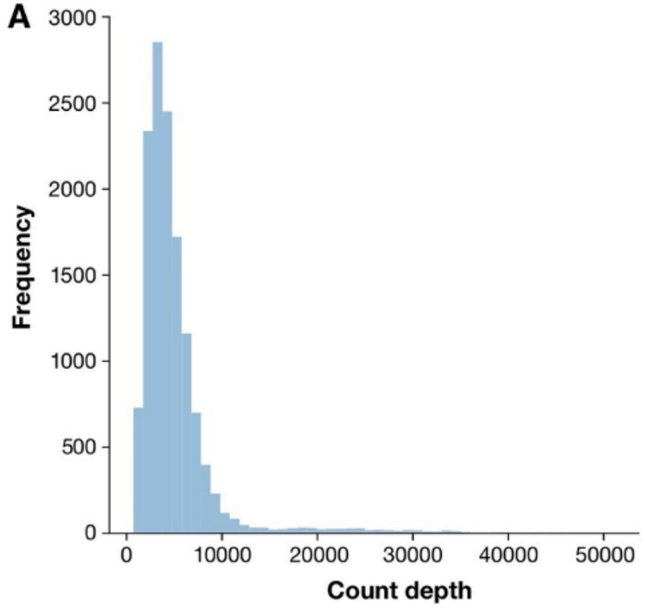
GENE ID	KD.2	KD.3	OE.1	OE.2	OE.3	IR.1	IR.2	IR.3
1/2-SBSRNA4	57	41	64	55	38	45	31	39
A1BG	71	40	100	81	41	77	58	40
A1BG-AS1	256	177	220	189	107	213	172	126
A1CF	0	1	1	0	0	0	0	0
A2LD1	146	81	138	125	52	91	80	50
A2M	10	9	2	5	2	9	8	4
A2ML1	3	2	6	5	2	2	1	0
A2MP1	0	0	2	1	3	0	2	1
A4GALT	56	37	107	118	65	49	52	37
A4GNT	0	0	0	0	1	0	0	0
AA06	0	0	0	0	0	0	0	0
AAA1	0	0	1	0	0	0	0	0
AAAS	2288	1363	1753	1727	835	1672	1389	1121
AACS	1586	923	951	967	484	938	771	635
AACSP1	1	1	3	0	1	1	1	3
AADAC	0	0	0	0	0	0	0	0
AADACL2	0	0	0	0	0	0	0	0
AADACL3	0	0	0	0	0	0	0	0
AADACL4	0	0	1	1	0	0	0	0
AADAT	856	539	593	576	359	567	521	416
AAGAB	4648	2550	2648	2356	1481	3265	2790	2118
AAK1	2310	1384	1869	1602	980	1675	1614	1108
AAMP	5198	3081	3179	3137	1721	4061	3304	2623
AANAT	7	7	12	12	4	6	2	7
AARS	5570	3323	4782	4580	2473	3953	3339	2666

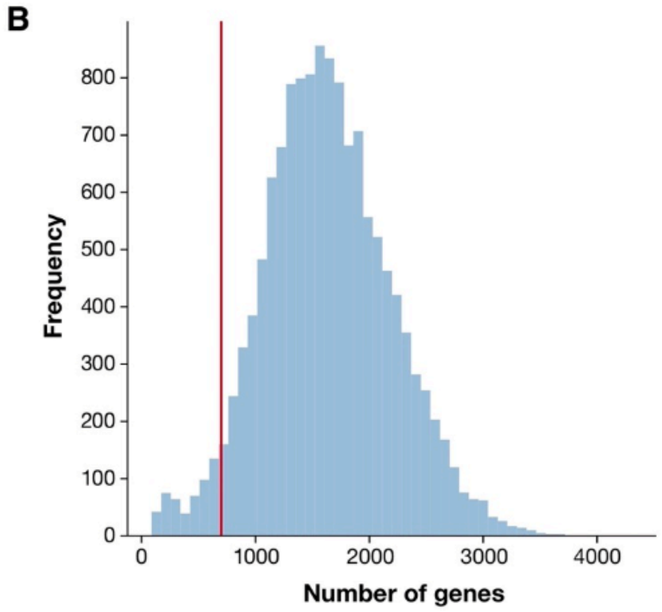
https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/05_counting_reads.html

scRNA-seq data processing

Step 3: quality control







scRNA-seq data processing

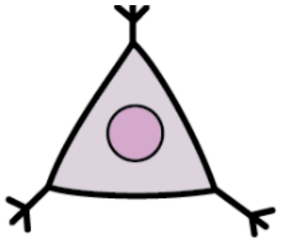
Step 4: normalisation

10% Capture Efficiency

Gene	Cell A
X	10
Y	20
Z	70

20% CE

Gene	Cell B
X	20
Y	40
Z	140



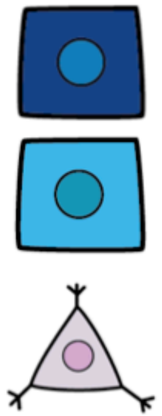
20% CE

Gene	Cell C
X	20
Y	0
Z	80

scRNA-seq data processing

Step 4: normalisation

Raw counts



	X	Y	Z
A	10	20	70
B	20	40	140
C	20	0	80

Pairwise distances

dist(A,B) = 71.4

dist(A,C) = 24.5

dist(B,C) = 67.1

$$\text{dist(A,B)} = \sqrt{(x_A - x_B)^2 + (y_a - y_b)^2 + (z_a - z_b)^2}$$

scRNA-seq data processing

Step 4: normalisation

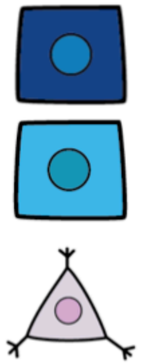
Raw counts

	X	Y	Z	Library Size
A	10	20	70	100
B	20	40	140	200
C	20	0	80	100

scRNA-seq data processing

Step 4: normalisation

Normalized counts



	X	Y	Z	Library Size
A	0.1	0.2	0.7	100
B	0.1	0.2	0.7	200
C	0.2	0	0.8	100

Pairwise distances

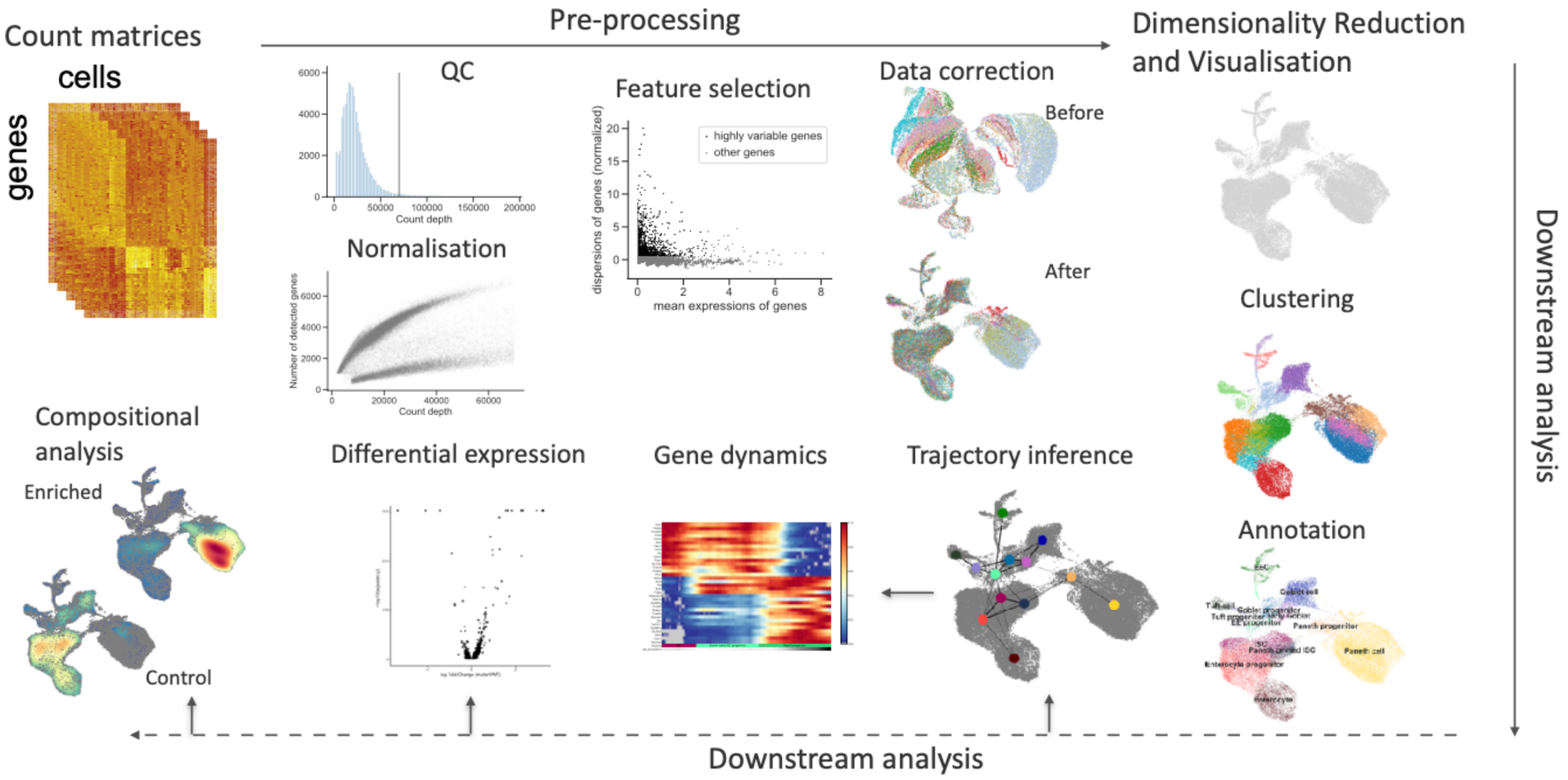
dist(A,B) = 0

dist(A,C) = 0.25

dist(B,C) = 0.25

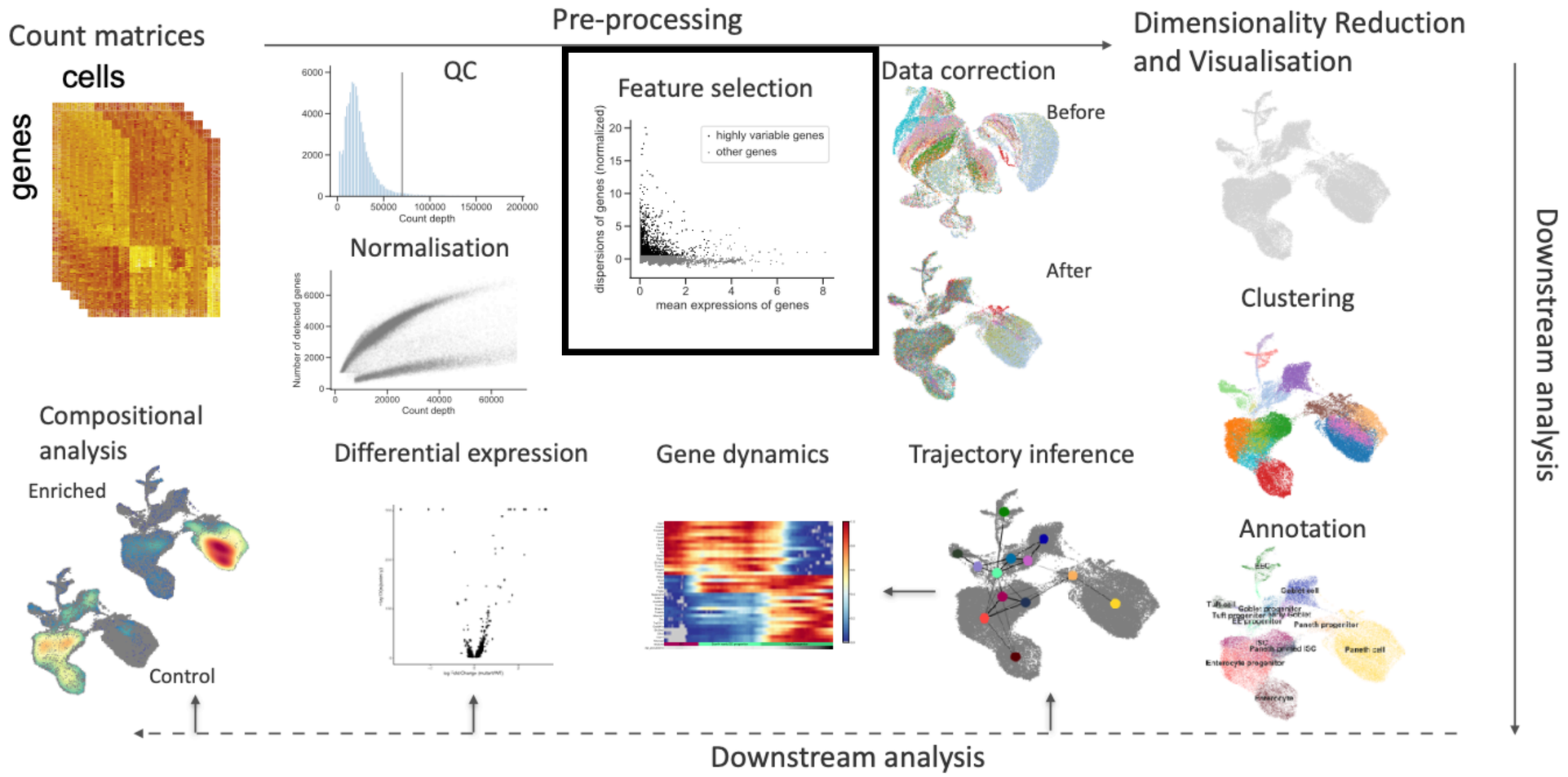
$$\text{dist(A,B)} = \sqrt{(x_A - x_B)^2 + (y_a - y_b)^2 + (z_a - z_b)^2}$$

Overview of scRNA-seq data analysis



scRNA-seq downstream analysis

Feature selection



scRNA-seq downstream analysis

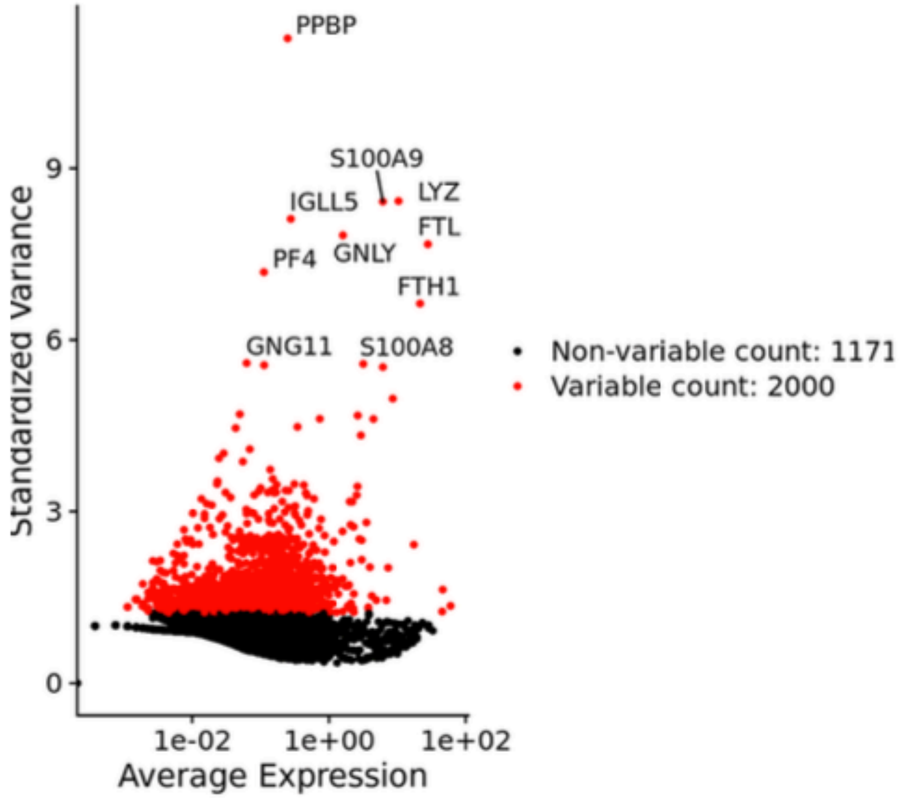
Feature selection

Why select features?

- Signal-to-noise ratio (Not all genes contain the same information)
- “Curse of dimensionality”
- Computational & storage efficiency

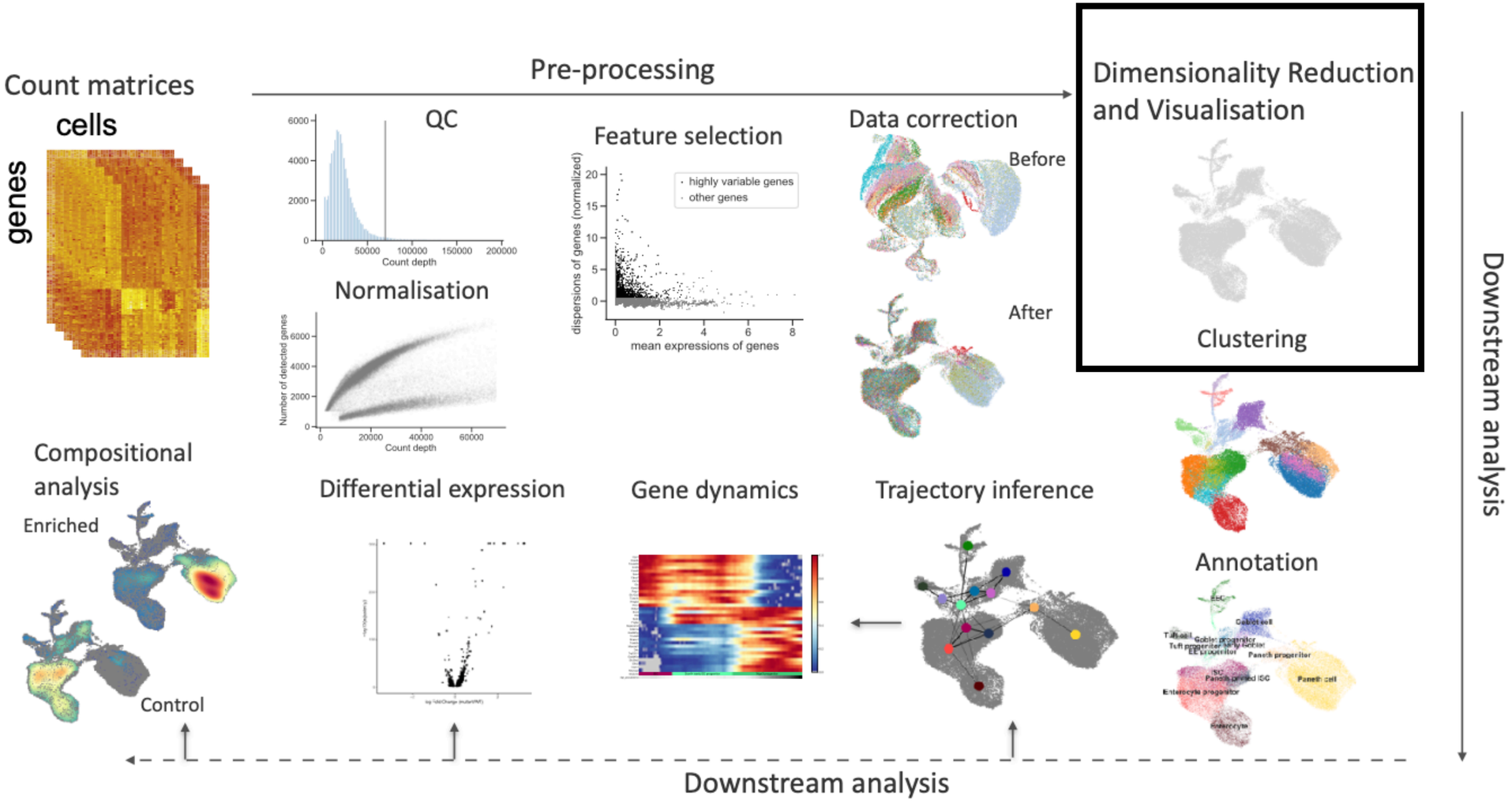
Approach: select genes with higher variance than expected from mean-variance relationship model (“Highly variable genes”)

- Relationship of gene dispersion ($\frac{\text{Var}}{\mu}$) to mean
- bin dispersions by means
- normalize dispersions (z-scores)
- select top genes based on normalized dispersions



scRNA-seq downstream analysis

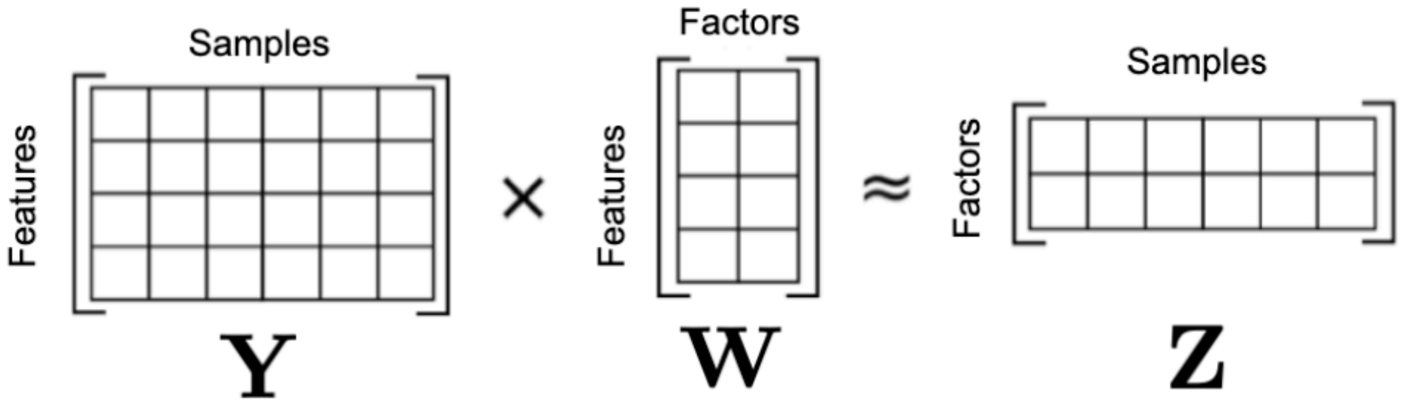
Dimensionality reduction



scRNA-seq data processing

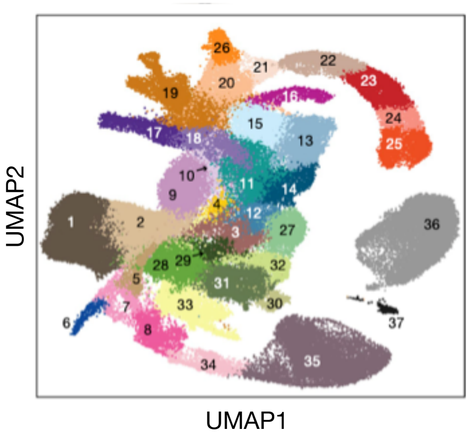
Linear dimensionality reduction: Principal Component Analysis

PCA defines $f(\mathbf{Y}|\mathbf{\Theta})$ to be a linear transformation via a matrix $\mathbf{W} \in \mathbb{R}^{D \times K}$ that maps the observations $\mathbf{Y} \in \mathbb{R}^{N \times D}$ onto the latent space $\mathbf{Z} \in \mathbb{R}^{N \times K}$.

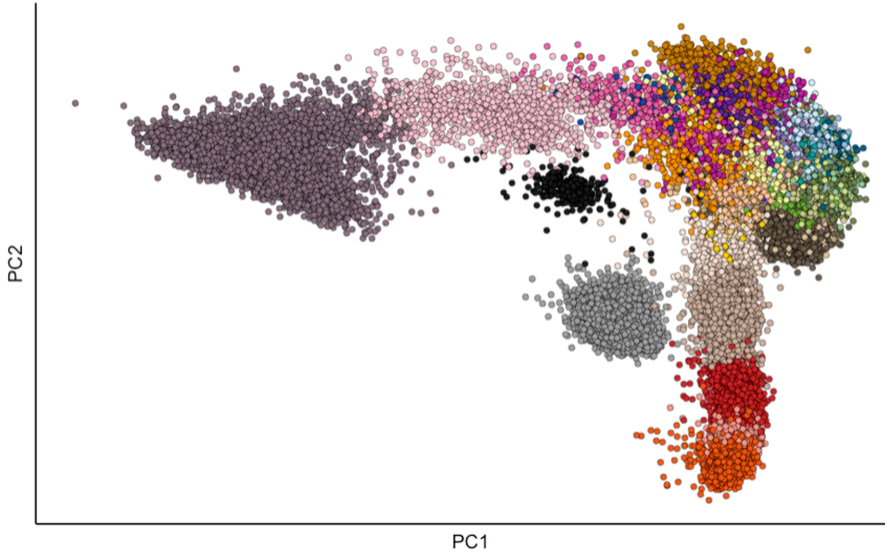


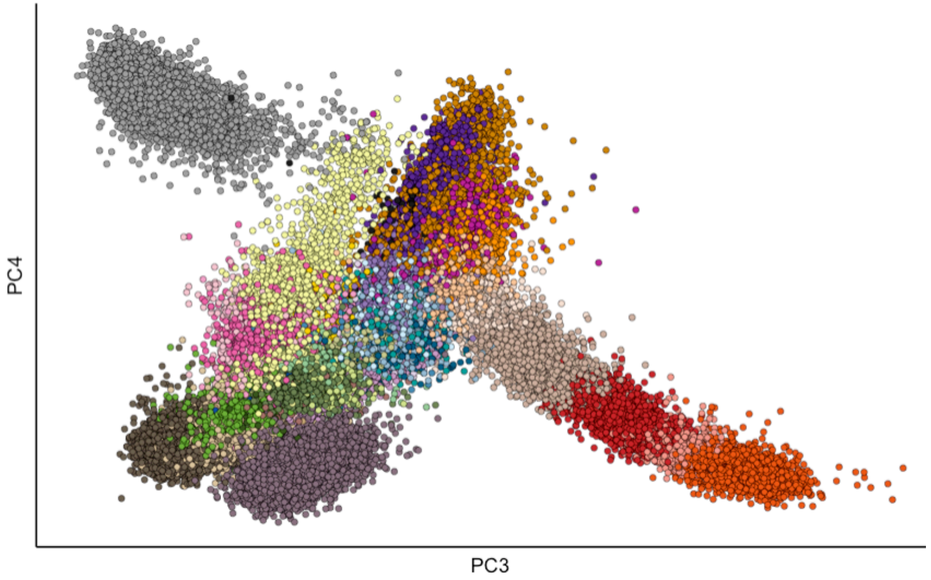
scRNA-seq data processing

Linear dimensionality reduction: Principal Component Analysis



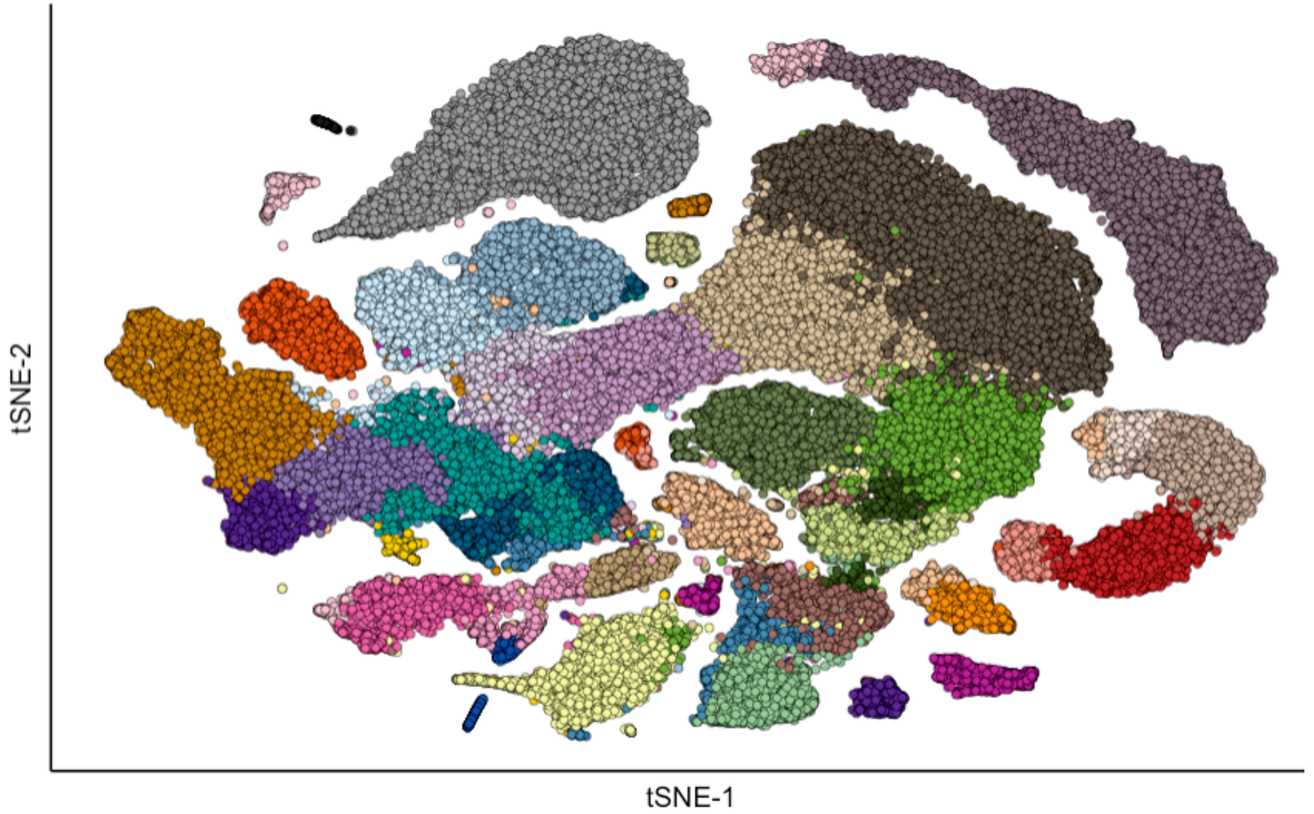
Unlike most bulk RNA-seq studies, single-cell genomics data has too many sources of variation to be easily visualised in two dimensions with PCA





scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE



scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE

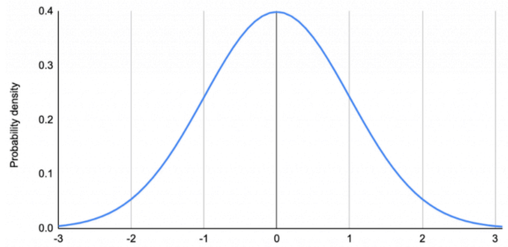
Stochastic Neighbor Embedding (SNE) starts by converting the high-dimensional Euclidean distances between datapoints into conditional probabilities that represent similarities. The similarity of datapoint *x_j* to datapoint *x_i* is the conditional probability, *p(j|i)*, that *x_i* would pick *x_j* as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at *x_i:*

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)},$$

For the low-dimensional counterparts *y_i* and *y_j* of the high-dimensional datapoints *x_i* and *x_j*, we define a similar conditional probability, which we denote by:

$$q_{j|i} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|y_i - y_k\|^2\right)}.$$

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



van der Maaten & Hinton et al. JMLR. 2018

scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE

Kullback-Leibler (KL) divergence measures “distance” between two probability distributions

$$\sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}},$$

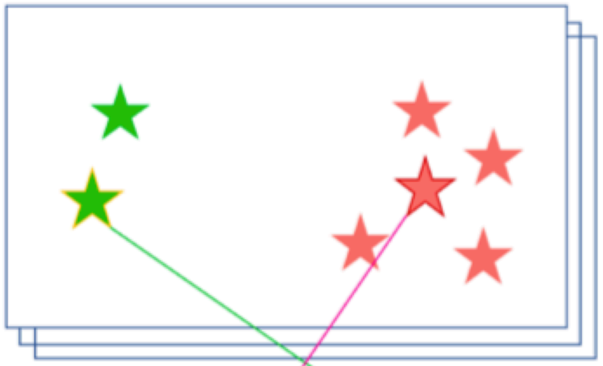
t-SNE minimizes the sum of Kullback-Leibler (KL) divergences over all datapoints using a gradient descent method. The gradient with respect to *y_i* is:

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE

High dimension



$$p_{ij} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)}{\sum_k \sum_{l \neq k} \exp(-\|\mathbf{x}_k - \mathbf{x}_l\|^2/2\sigma^2)}$$

Compute pairwise similarities between samples with normalized Gaussian kernel

Low dimension

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}$$

Measure normalized Student-t similarities in the t-SNE map

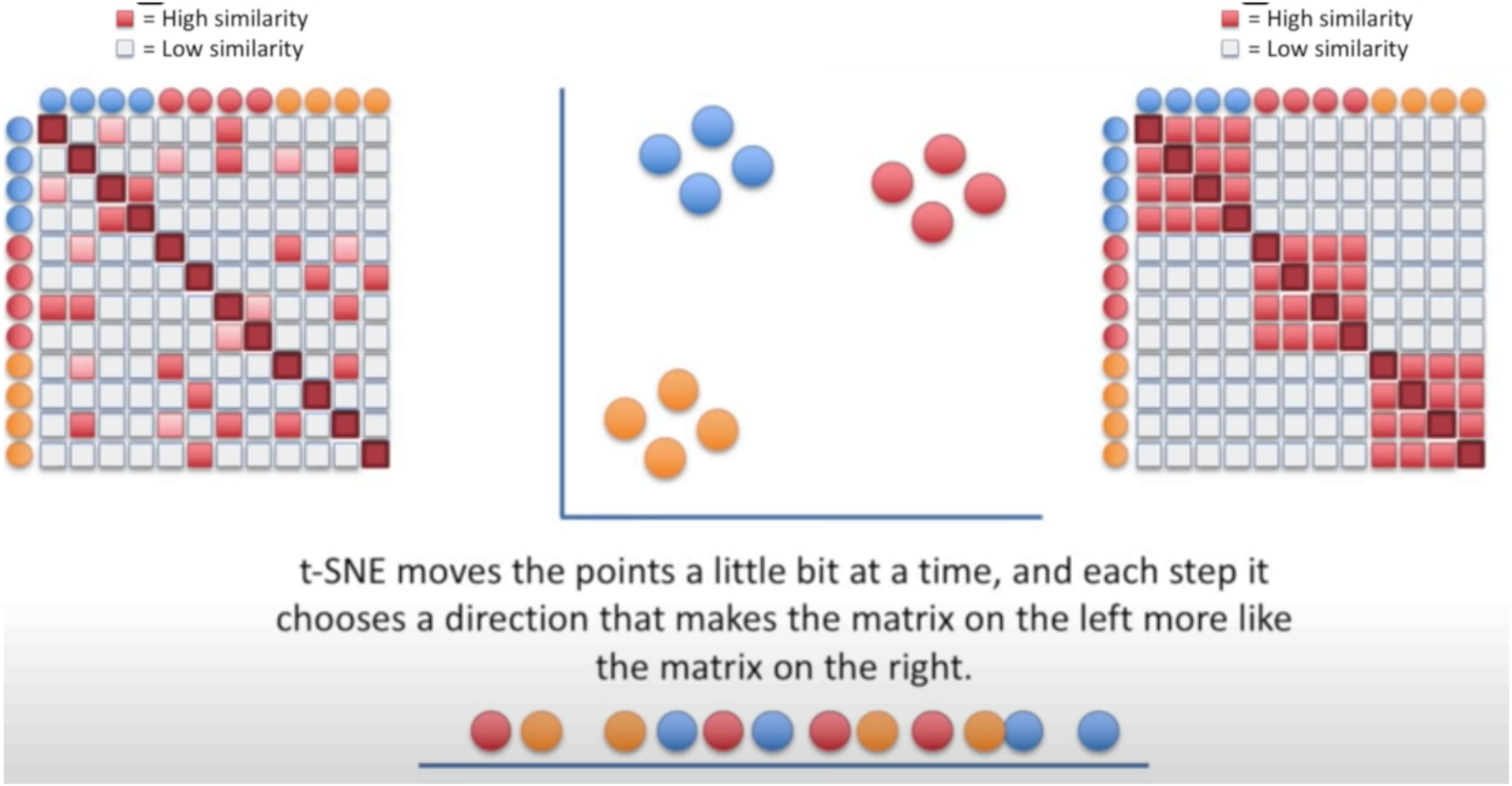
Low dimension

$$KL(P||Q) = \sum_i \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Minimize the divergence between both distributions

scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE



https://www.youtube.com/watch?v=NEaUSP4YerM

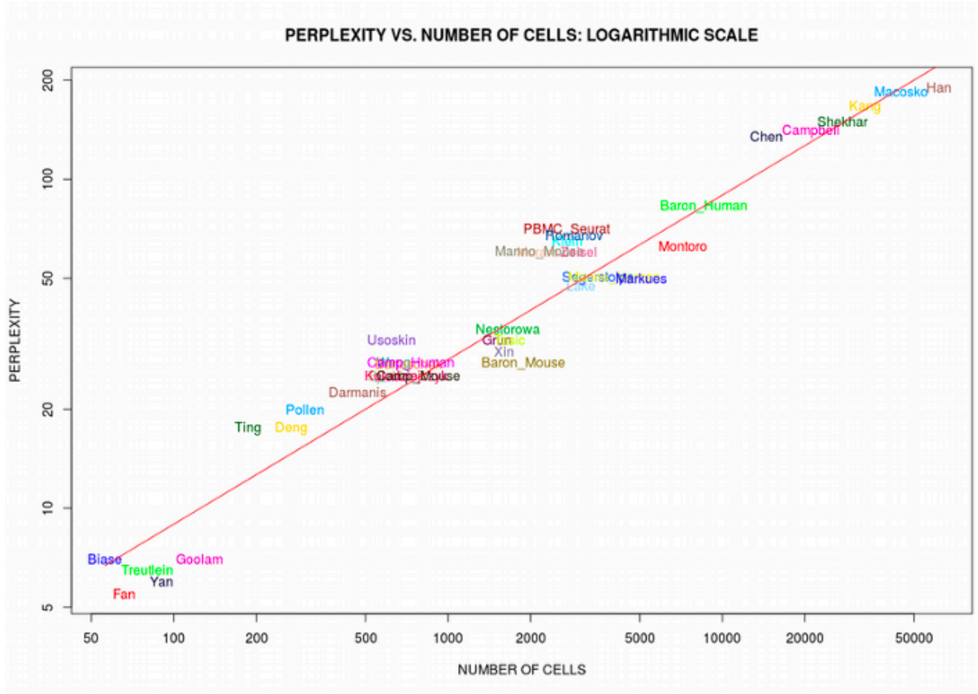
scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE

t-SNE has one key hyperparameter, which is the variance of the normal distribution centered at each data point. However, in the t-SNE implementation they reformulate some equations and instead of providing the variance you need to provide the **perplexity**, which can be interpreted as the effective number of neighbors each data point can “see”

How to select the optimal perplexity:

- Effective perplexity values are usually between 10 an 50. Just “try and see”
- Larger perplexities will take more global structure into account
- The larger the data set, the higher the “optimal” perplexity



https://towardsdatascience.com/how-to-tune-hyperparameters-of-tsne-7c0596a18868

scRNA-seq data processing

Non-linear dimensionality reduction: t-SNE

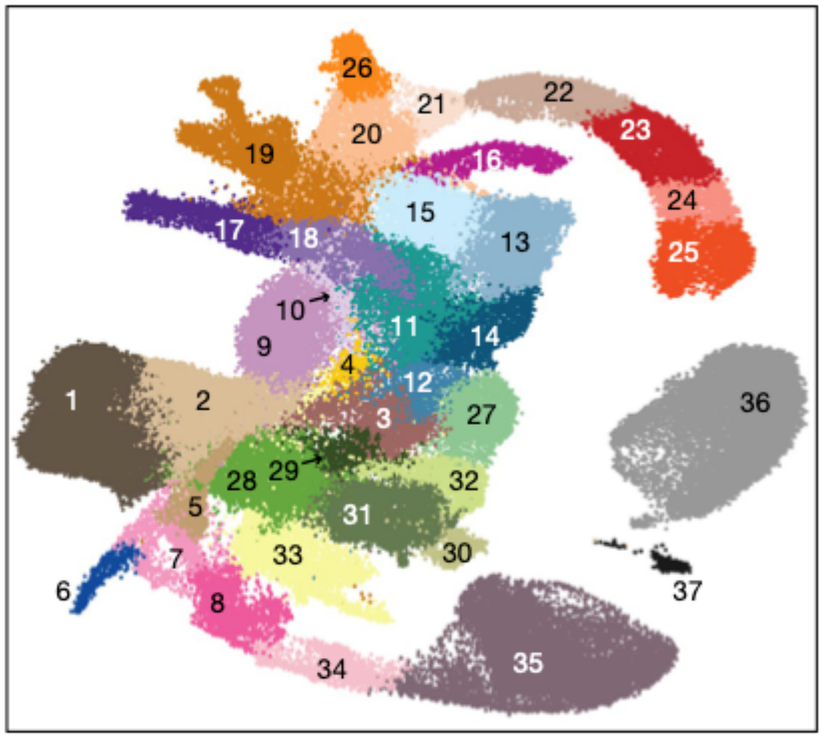
Practical hints when using t-SNE:

- Since t-SNE is a stochastic algorithm, the method might yield different solutions at different runs
- Use the PCA representation as input, not the gene expression matrix
- within-cluster distances are more meaningful than between-cluster distances

scRNA-seq data processing

Non-linear dimensionality reduction: UMAP

UMAP is constructed from a theoretical framework based in Riemannian geometry and algebraic topology. It betters preserves the global structure with superior run time performance than t-SNE.



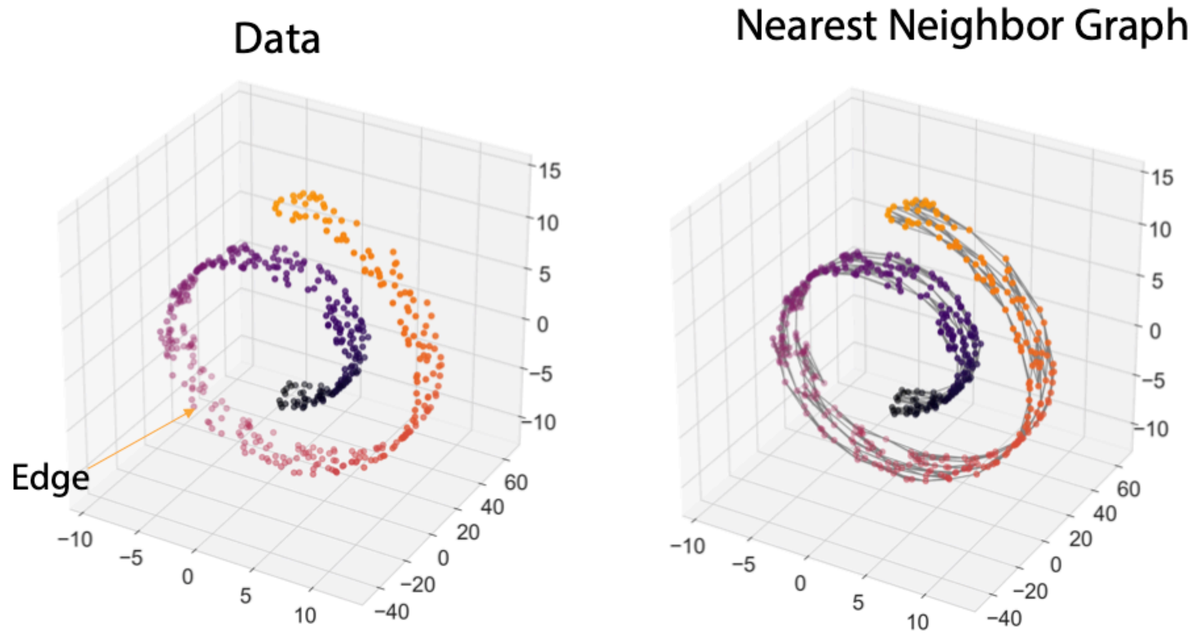
1 Epiblast
2 Primitive streak
3 Caudal epiblast
4 Primordial germ cells
5 Anterior primitive streak
6 Notochord
7 Def. endoderm
8 Gut
9 Nascent mesoderm
10 Mixed mesoderm
11 Intermediate mesoderm
12 Caudal mesoderm
13 Paraxial mesoderm
14 Somitic mesoderm
15 Pharyngeal mesoderm
16 Cardiomyocytes
17 Allantois
18 ExE mesoderm
19 Mesenchyme
20 Haemato-endothelial prog.
21 Blood progenitors 1
22 Blood progenitors 2
23 Erythroid 1
24 Erythroid 2
25 Erythroid 3
26 Endothelium
27 Neuromesodermal progenitors
28 Rostral neuroectoderm
29 Caudal neuroectoderm
30 Neural crest
31 Forebrain/midbrain/hindbrain
32 Spinal cord
33 Surface ectoderm
34 Visceral endoderm
35 ExE endoderm
36 ExE ectoderm
37 Parietal endoderm

PijuanSala et al., *Nature*. 2019

scRNA-seq data processing

Non-linear dimensionality reduction: UMAP

The first step is to construct a graph representation of the data, which they call a "fuzzy simplicial complex". Each node is a cell and edges connect cells based on the similiarities of their RNA expression profiles



scRNA-seq data processing

Non-linear dimensionality reduction: UMAP

The low-dimensional embedding is found by searching for a low dimensional projection of the data that has the closest possible equivalent fuzzy topological structure.

Three assumptions are required to derive the mathematical framework (which we will ignore):

- The data is uniformly distributed on Riemannian manifold
- The Riemannian metric is locally constant (or can be approximated as such)
- The manifold is locally connected.

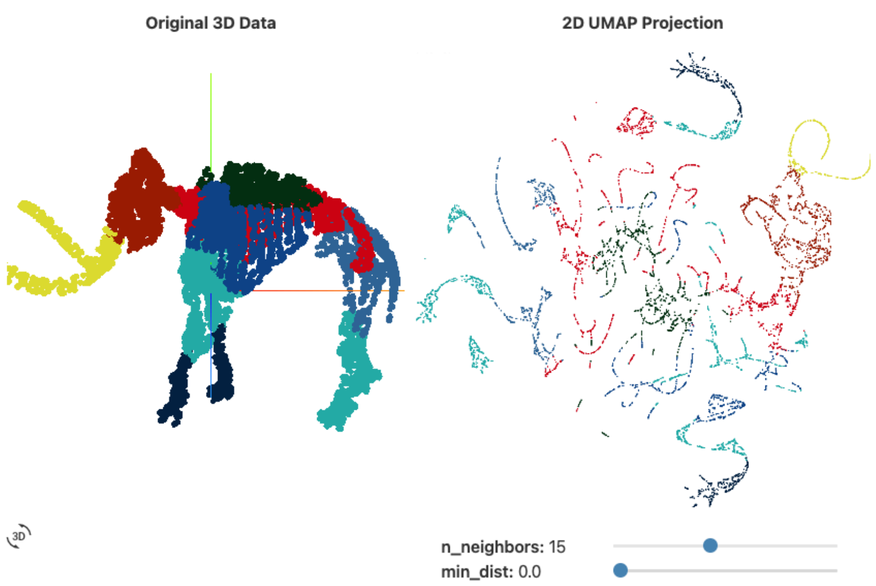
scRNA-seq data processing

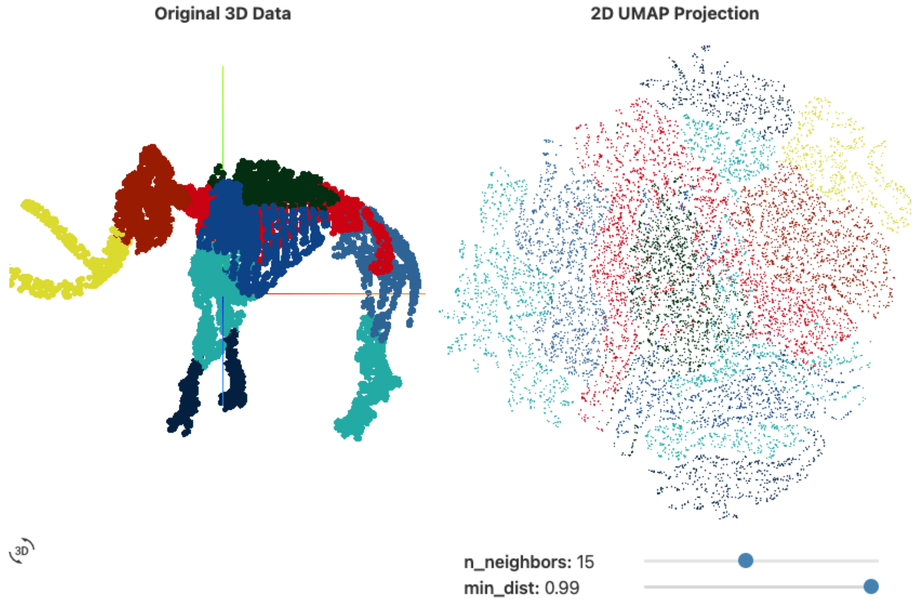
Non-linear dimensionality reduction: UMAP

UMAP has two key hyperparameters, which control the balance between local and global structure in the low-dimensional projection:

- **The number of nearest neighbours to build the graph (n_neighbors):** low values will push UMAP to focus more on local structure by constraining the number of neighboring points, while high values will push UMAP towards representing the big-picture structure while losing fine detail.

- **Minimum distance between points in low-dimensional space (min_dist):** controls how tightly UMAP clumps points together, with low values leading to more tightly packed embeddings. Larger values of min_dist will make UMAP pack points together more loosely, focusing instead on the preservation of the broad topological structure.





https://pair-code.github.io/understanding-umap/

scRNA-seq data processing

Non-linear dimensionality reduction: UMAP

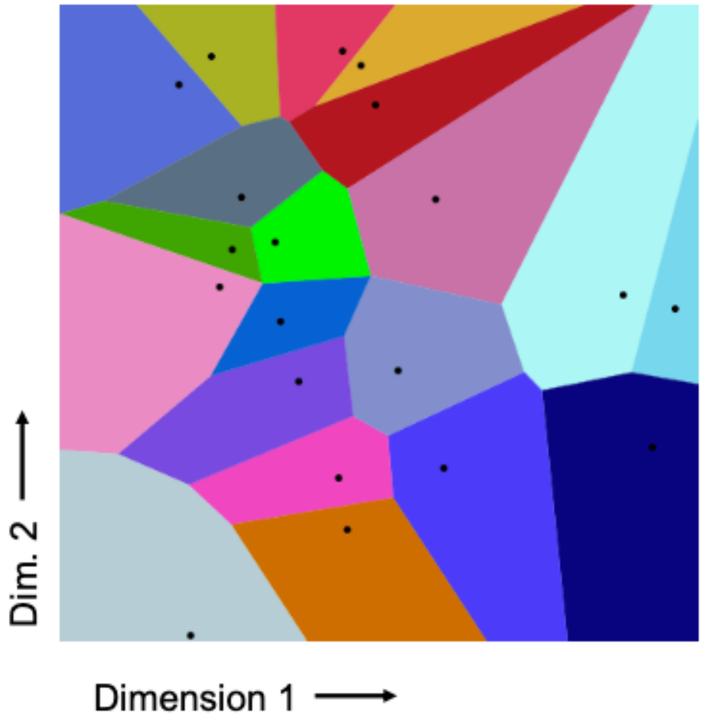
Practical hints when running UMAP:

- Since UMAP is a stochastic algorithm, the method might yield different solutions at different runs
- Use the PCA representation as input, rather than the gene expression matrix
- Hyperparameters really matter: choosing good values isn't easy, and depends on both the data and your goals (eg, how tightly packed the projection ought to be).

scRNA-seq data processing

Clustering

- Broadly speaking, the goal of clustering is to discover the natural groupings of a data set
- The ability to define cell types through unsupervised clustering on the basis of transcriptome similarity has emerged as one most powerful applications of scRNA-seq



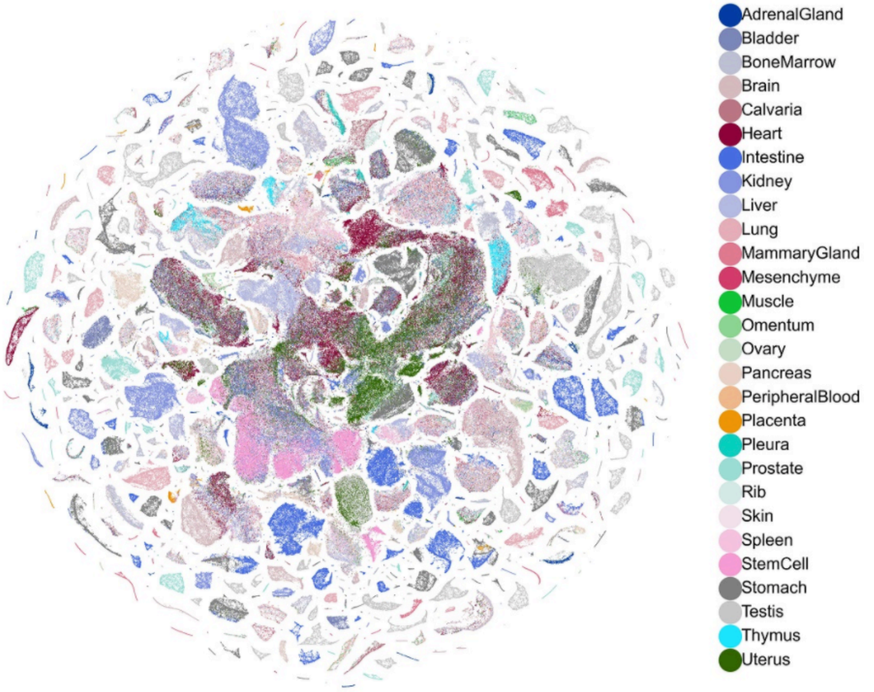
There is a multitude of “atlas” projects aim to build comprehensive references for all cell types present in an organism

Human Cell Atlas
https://www.humancellatlas.org/

HUMAN CELL ATLAS

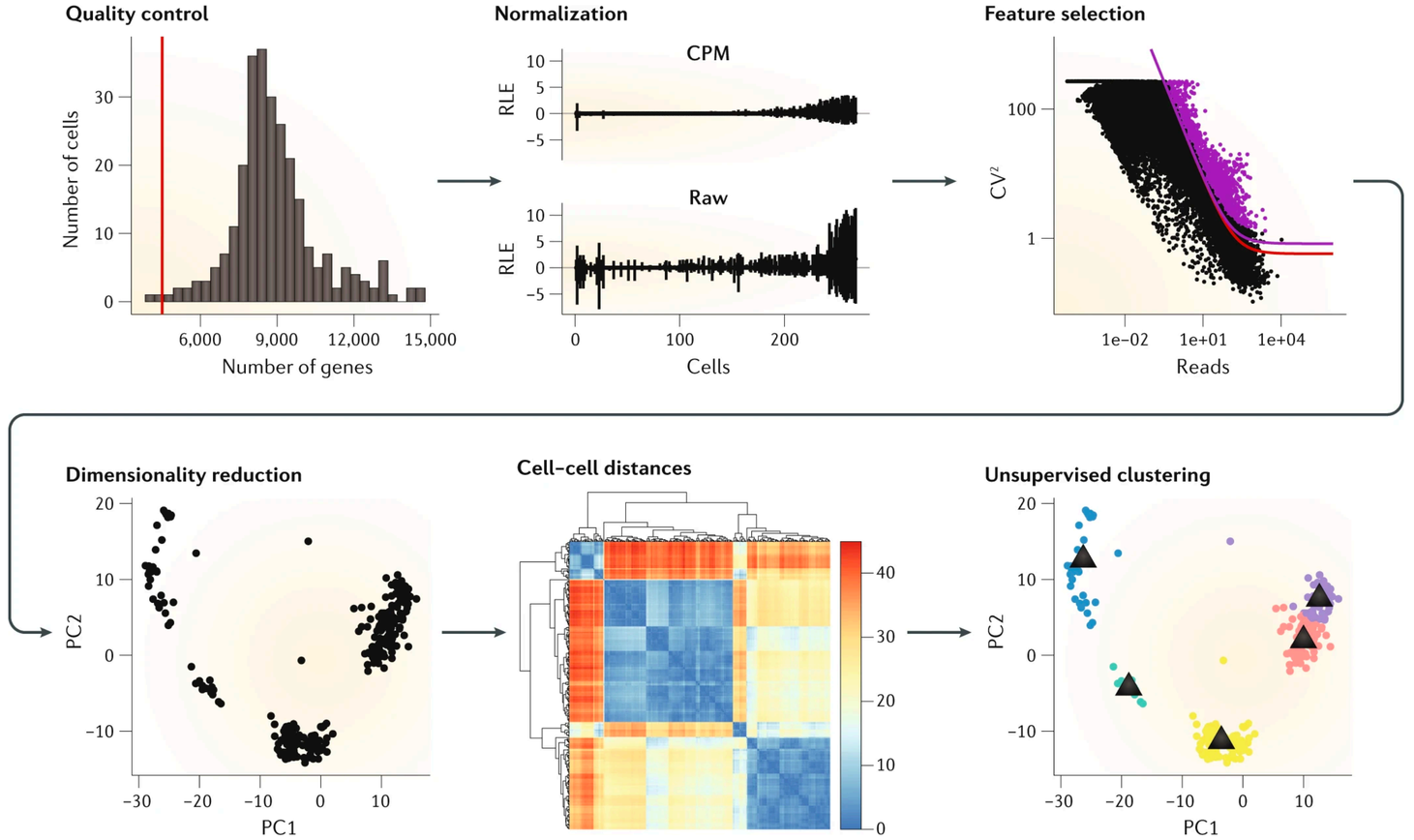
Fly Cell Atlas
https://flycellatlas.org/

Mouse cell atlas
http://bis.zju.edu.cn/MCA/



The application of **feature selection** and/or **dimensionality reduction** before clustering is important to reduce the noise and speed up the calculations.

- Feature selection involves identifying the most informative genes, for example, the ones with the highest variance
- Dimensionality reduction, for example, principal component analysis (PCA), projects data into a lower dimensional space

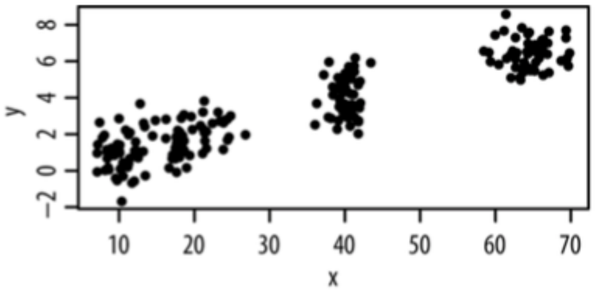


Kiselev et al. Challenges in unsupervised clustering of single-cell RNA-seq data. *Nature Review Genetics*. 2019

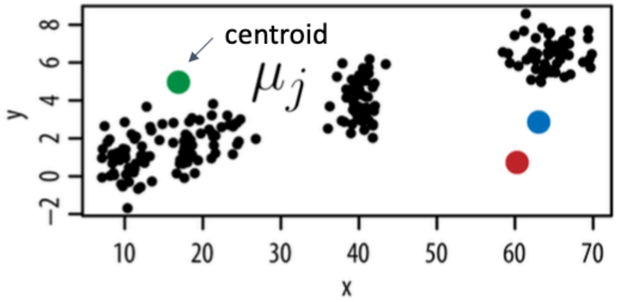
k-means clustering

The most popular generic clustering algorithm is **k-means**, which iteratively identifies *k* cluster centroids, and each cell is assigned to the closest centroid. The standard method for k-means, known as Lloyd's algorithm, has the advantage of scaling linearly with the number of points

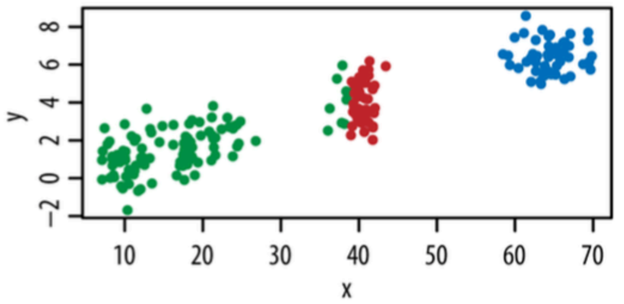
Unlabelled data



1. Random initialization

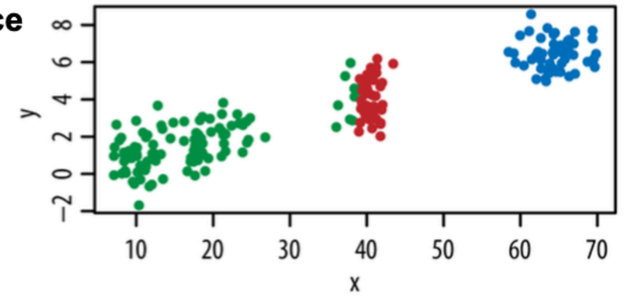


2. Assign points to nearest centroid



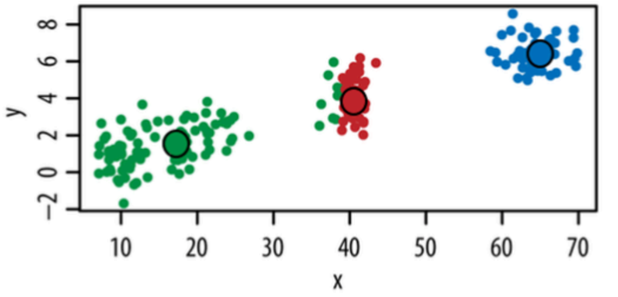
Repeat until convergence

2. Assign points to nearest centroid



3. Move centroids to middle of cluster

$$\mu_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$



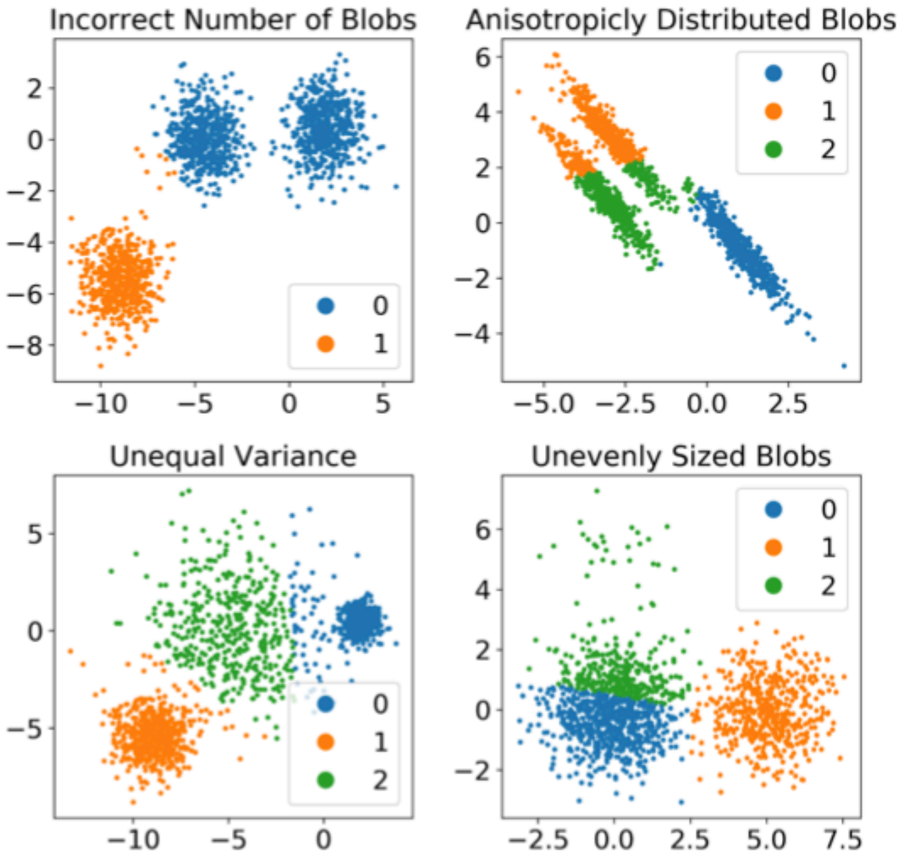
Drawbacks of k-means clustering

k-Means assumes:

1. K is chosen correctly
2. The data is distributed normally around the mean
3. All clusters have equal variance
4. All clusters have the same number of points

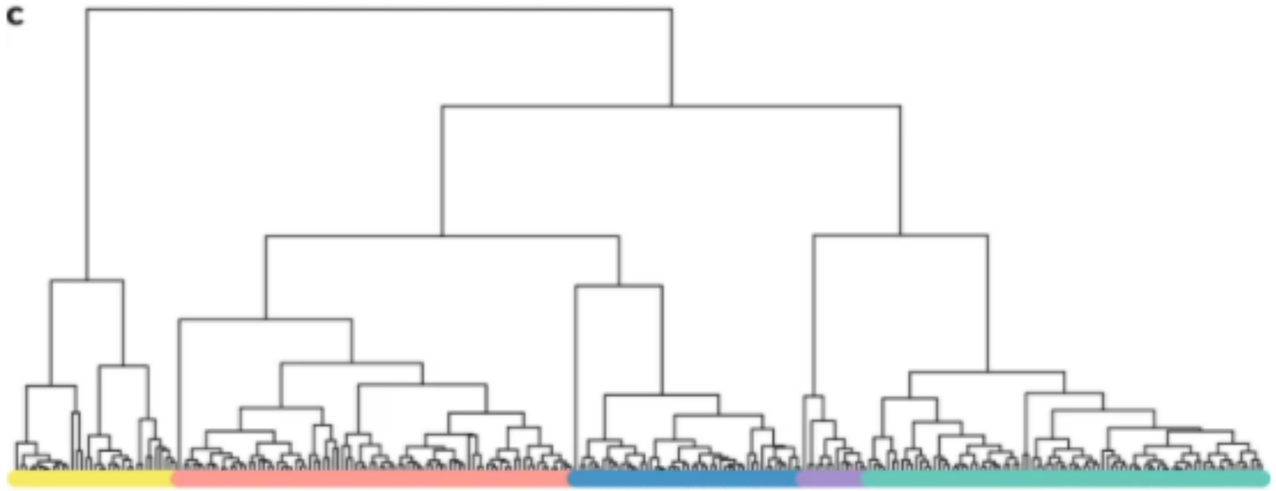
K-Means is sensitive to:

1. Initialization
2. Outliers



Hierarchical clustering

- Another widely used generic clustering algorithm frequently adapted for scRNA-seq is hierarchical clustering, which sequentially combines individual cells into larger clusters (agglomerative) or divides clusters into smaller groups (divisive).
- It captures the intrinsic hierarchy of cell taxonomies
- An important shortcoming is that both time and memory requirements scale quadratically with the number of data points, which means that it is prohibitively expensive to use for large data sets.



Graph-based community detection methods

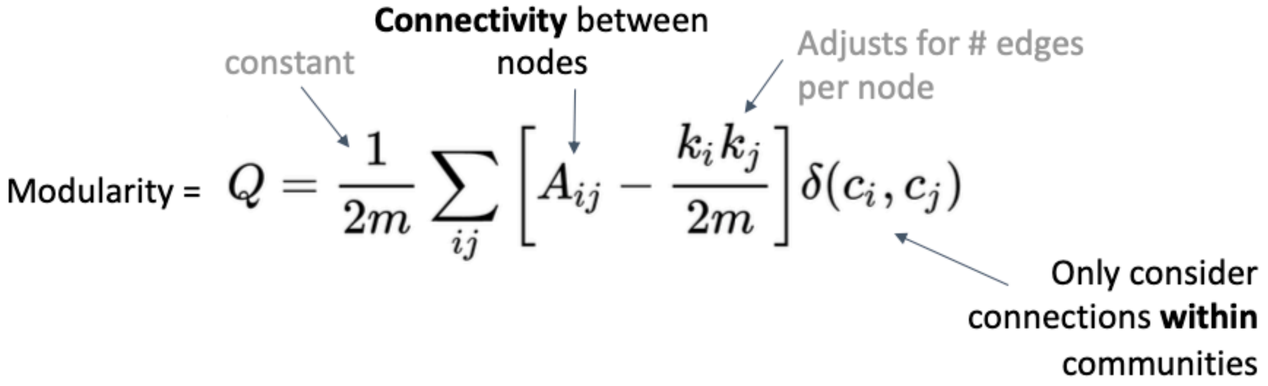
- Community detection is a variant on the idea of clustering that is specifically applied to graphs. Instead of identifying groups of points that are close together, community detection identifies groups of nodes that are densely connected.
- As some of the graph data sets available are extremely large, (i.e. social networks), graph-based algorithms have been developed with an emphasis on speed and scalability. Therefore, for large scRNA-seq data sets it has become increasingly popular to apply community-detection algorithm
- To apply community-detection algorithms to scRNA-seq data, it is necessary to construct a *k*-nearest-neighbours graph.

Graph-based community detection methods

- Community detection is a variant on the idea of clustering that is specifically applied to graphs. Instead of identifying groups of points that are close together, community detection identifies groups of nodes that are densely connected.
- As some of the graph data sets available are extremely large, (i.e. social networks), graph-based algorithms have been developed with an emphasis on speed and scalability. Therefore, for large scRNA-seq data sets it has become increasingly popular to apply community-detection algorithm
- To apply community-detection algorithms to scRNA-seq data, it is necessary to construct a *k*-nearest-neighbours graph.

Graph-based community detection methods: Louvain clustering

Louvain clustering iteratively aggregates clusters on a graph to increase the “modularity”, a metric that measures the density of links inside communities compared to links between communities. For a weighted undirected graph, modularity is defined as:

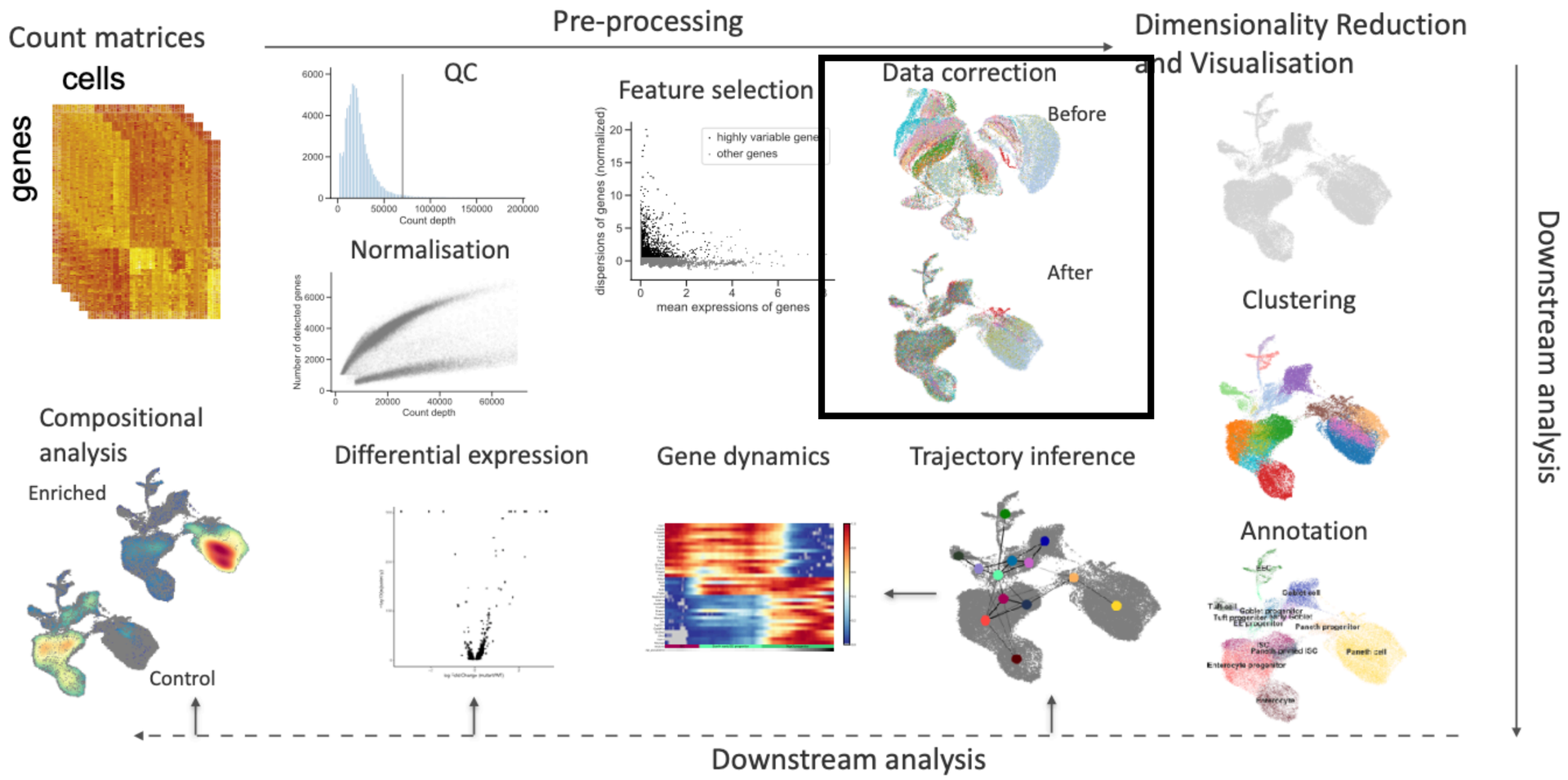


m is the sum of all edge weights in the graph
Aij is the edge weight between nodes *i* and *j*
ki is the sum of all edges around node i
δ is 1 if i and j are in the same community and 0 otherwise

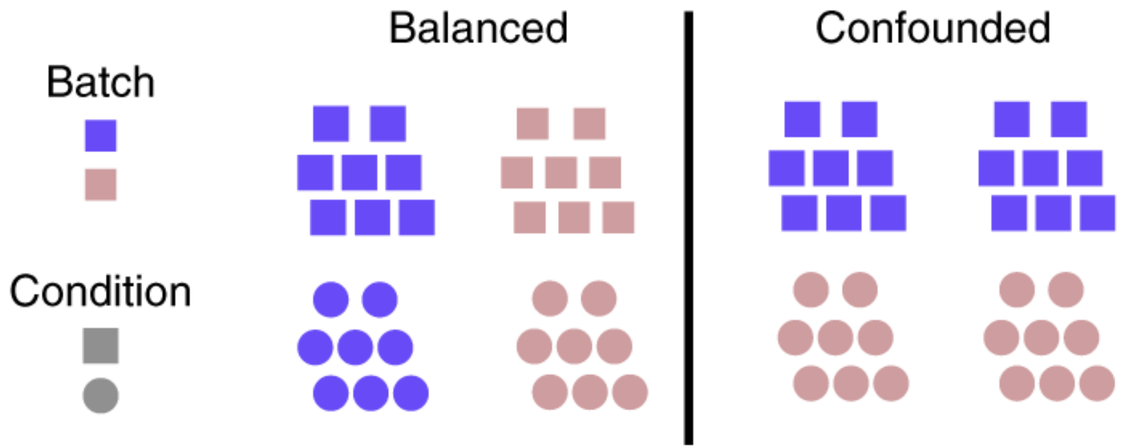
- Modularity satisfies -1 <= Q < 1
- Q > 0: the # of edges per community is higher than in a randomly wired graph
- Q < 0: the # edges per community is smaller than in a randomly weird graph
- The bigger Q, the more “significant” the communities are

scRNA-seq downstream analysis

Batch effect correction



Design of experiment is important to enable distinction between meaningful biological differences and difference caused by batches



Distinction of biological and batch effects feasible

Distinction of biological and batch effects not feasible

(Simple and linear)

Regression

- Fit regression model with batch effect covariate

Residuals (often using linear regression):

$$\hat{n}_{gc} = f_D(B_c, \dots)$$

$$r_{gc} = n_{gc} - \hat{n}_{gc} = n_{gc} - (\beta_0 + \beta_1 B_c)$$

in linear model case

Example:
sc.tl.regress_out()

Correct for fitted batch effect:

$$n_{gcb} = \alpha_g + X\beta_g + \gamma_{gb} + \delta_{gb}\epsilon_{gcb}$$

bio design matrix

additive batch effect

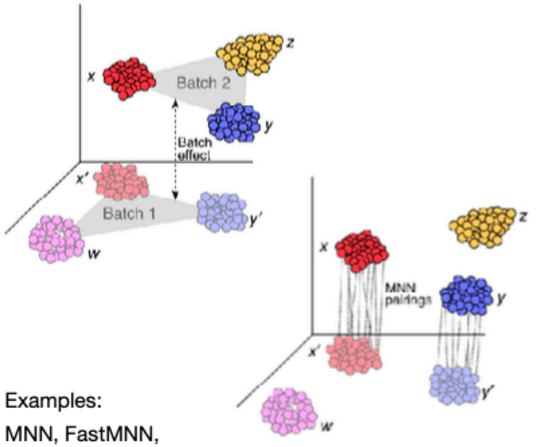
multiplicative batch effect

Example:
ComBat - ***scanpy.pp.combat()***

(Simple and non-linear)

Mutual nearest neighbours (MNN)

- Project cells into low dimensional embedding
- find most similar cells in other batch (MNNs)
- Use MNNs as anchors to calculate a correction vector

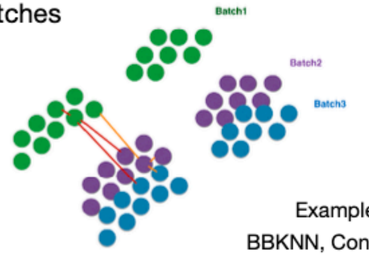


Examples:
MNN, FastMNN,
Seurat v3, Scanorama

(Complex and non-linear)

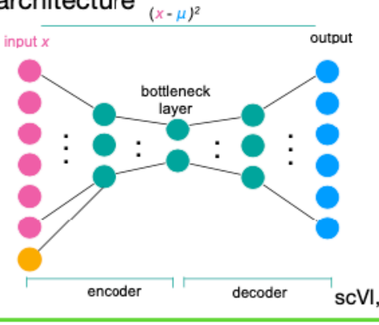
Graph methods & Deep learning

Enforce graph connections between different batches



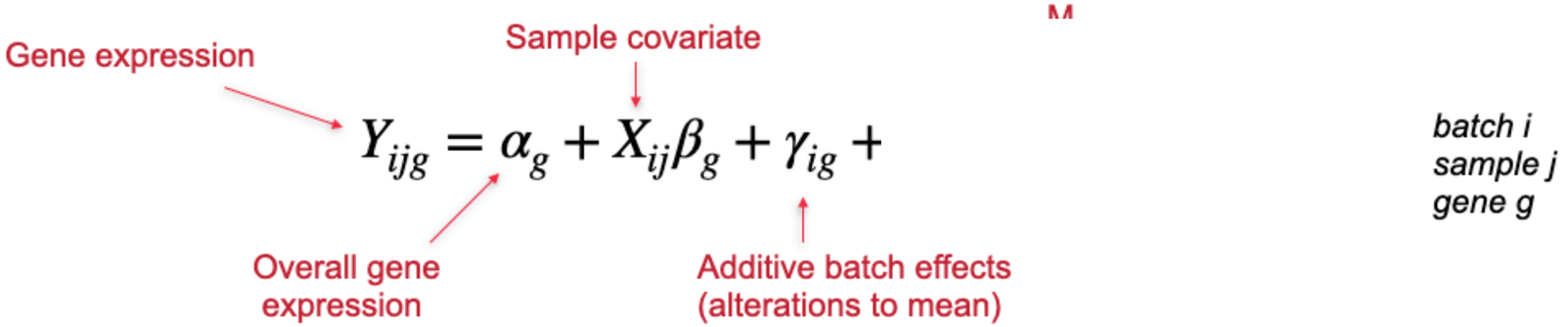
Examples:
BBKNN, Conos

Add condition node into auto-encoder architecture



Examples:
scVI, trVAE, SAUCIE

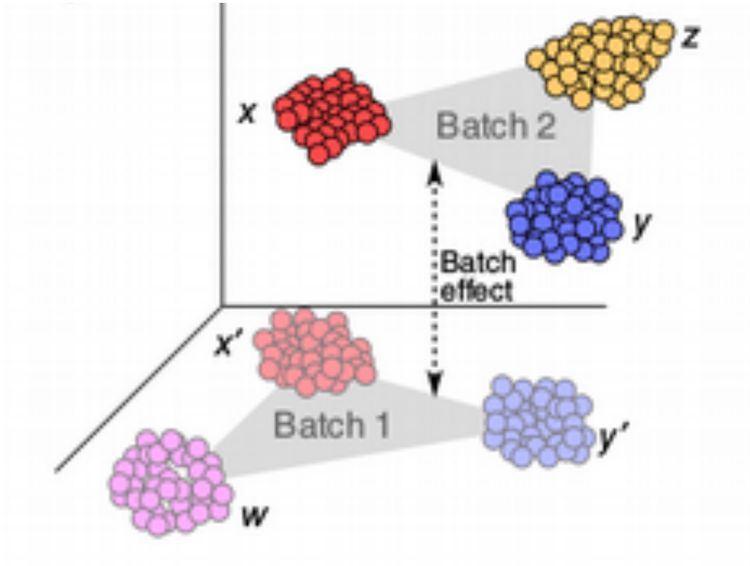
Known batch effects can be accounted for in a linear model (e.g. limma) or generalised linear models (e.g. DESeq)

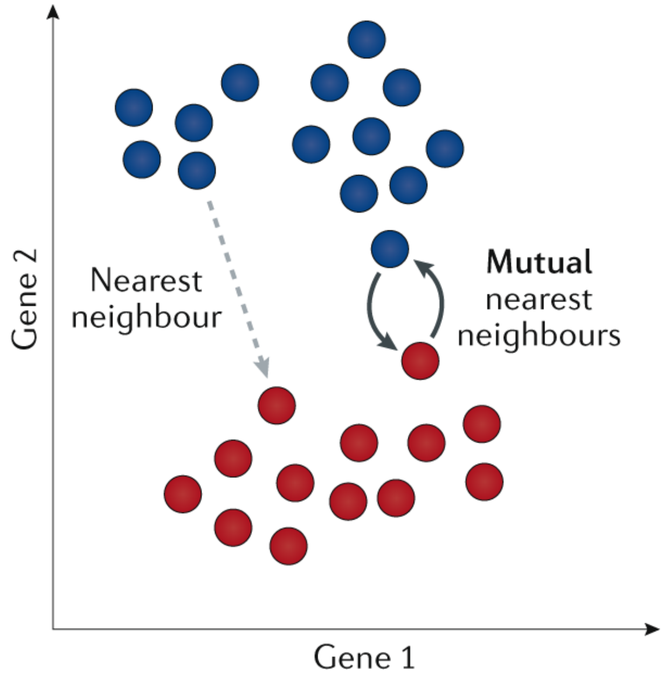


- Linear models are efficient when dealing with batches that are technical replicates generated from the same population of cells
- Linear models are not optimal when there are differences in cell type composition between batches

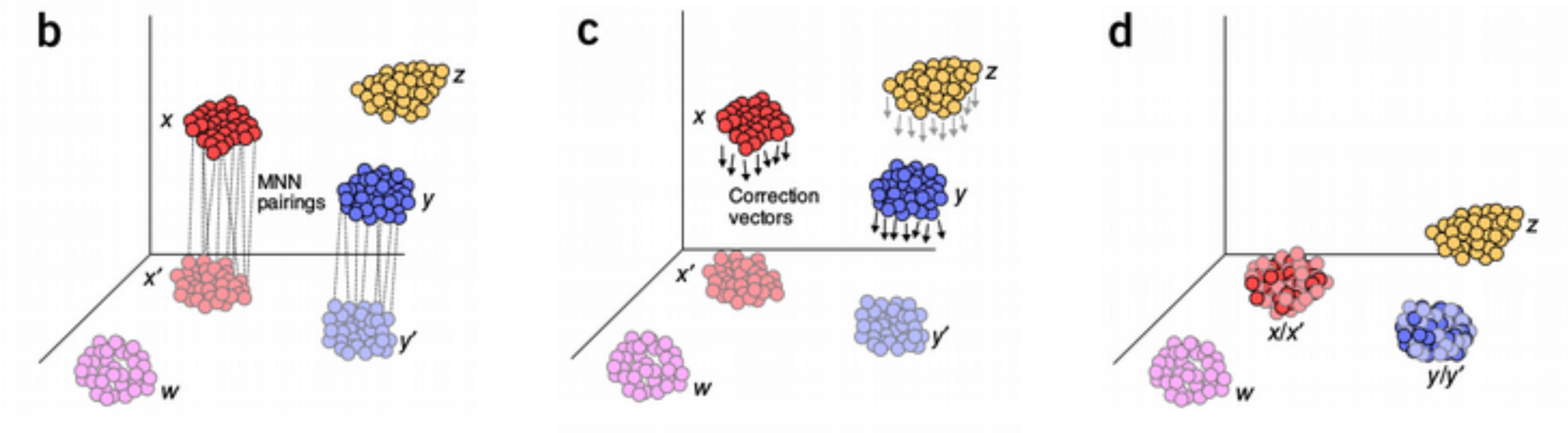
Mutual nearest neighbours (MNN) algorithm

Identify mutual nearest neighbours between batches to maintain biological differences and find technical differences.

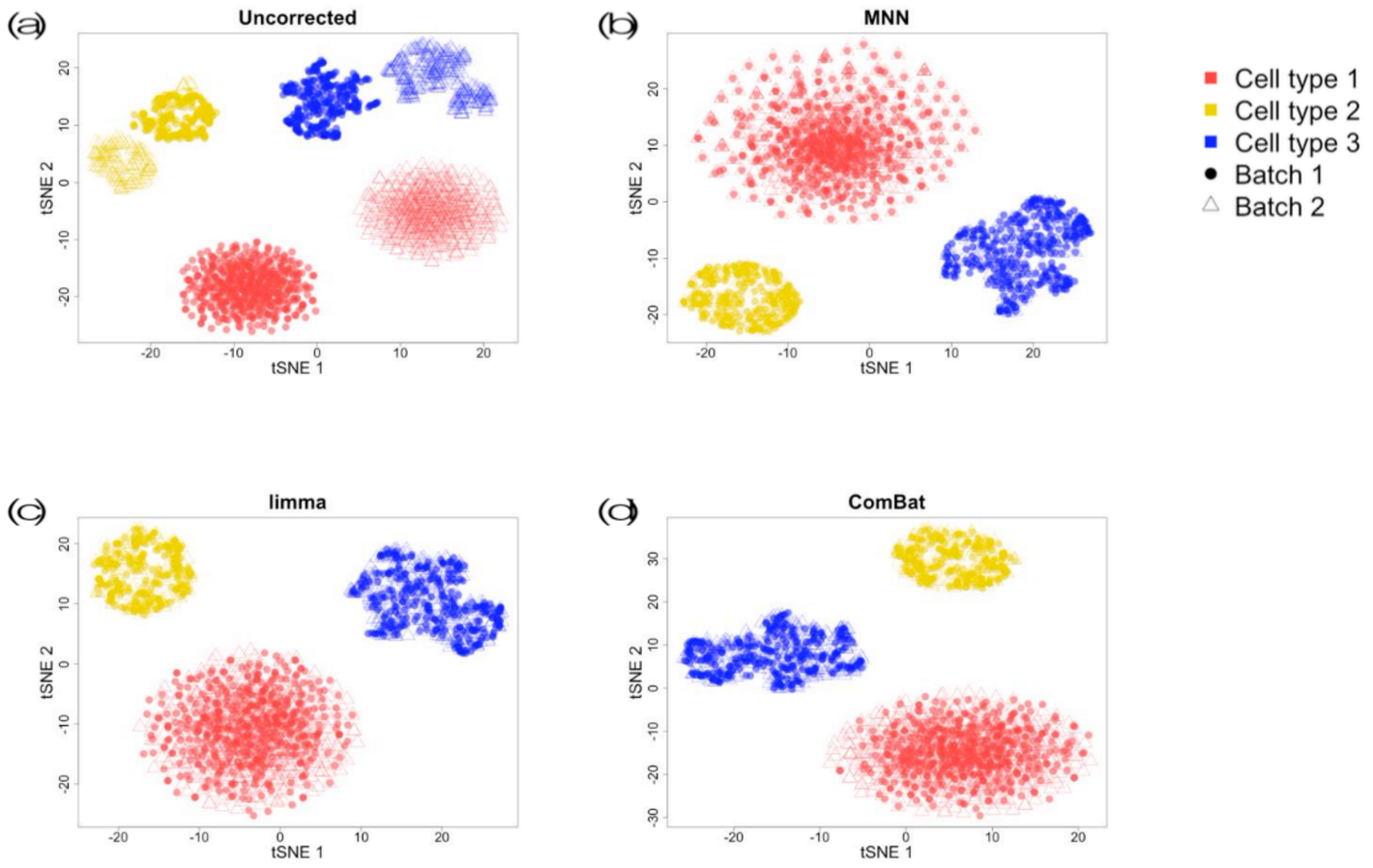




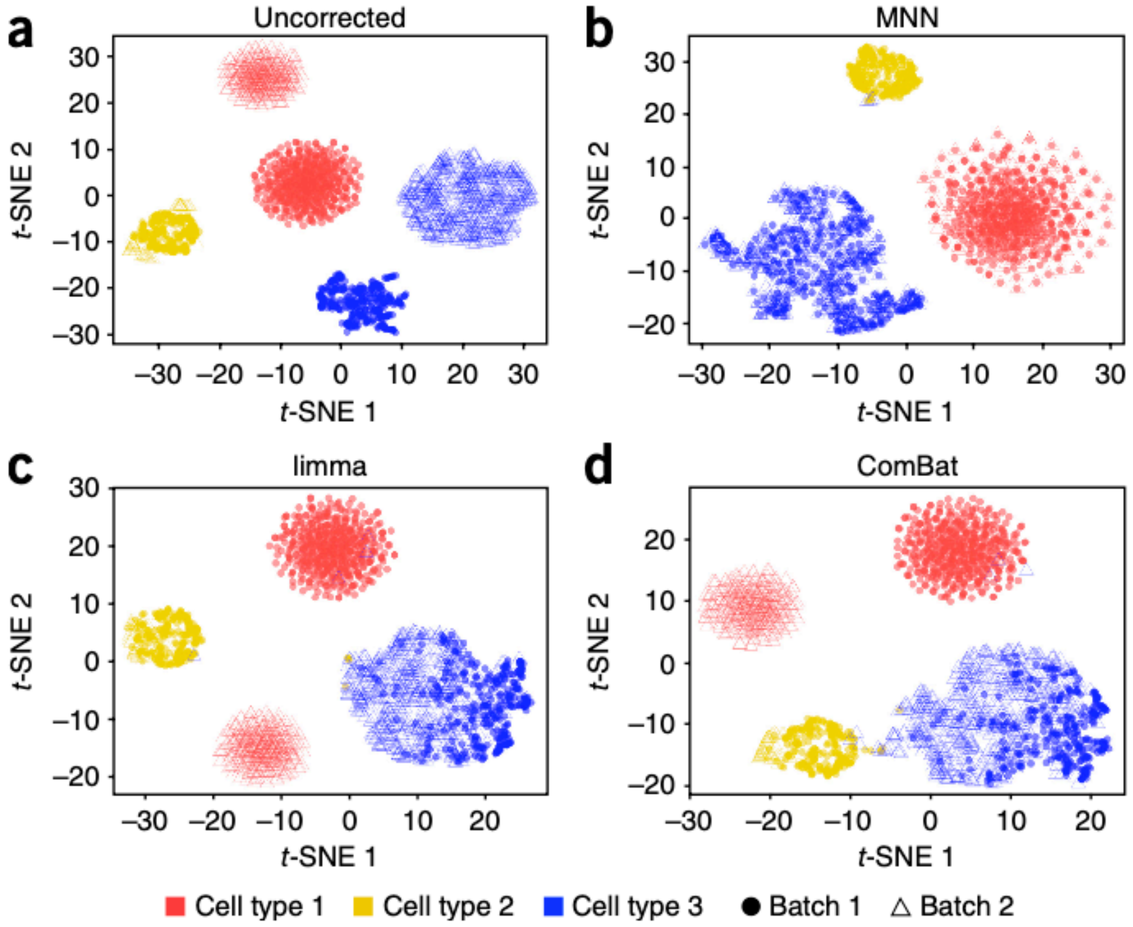
Once mutual nearest neighbours are identified, we can construct correction vectors based on their difference and subtract these to integrate one batch into the other



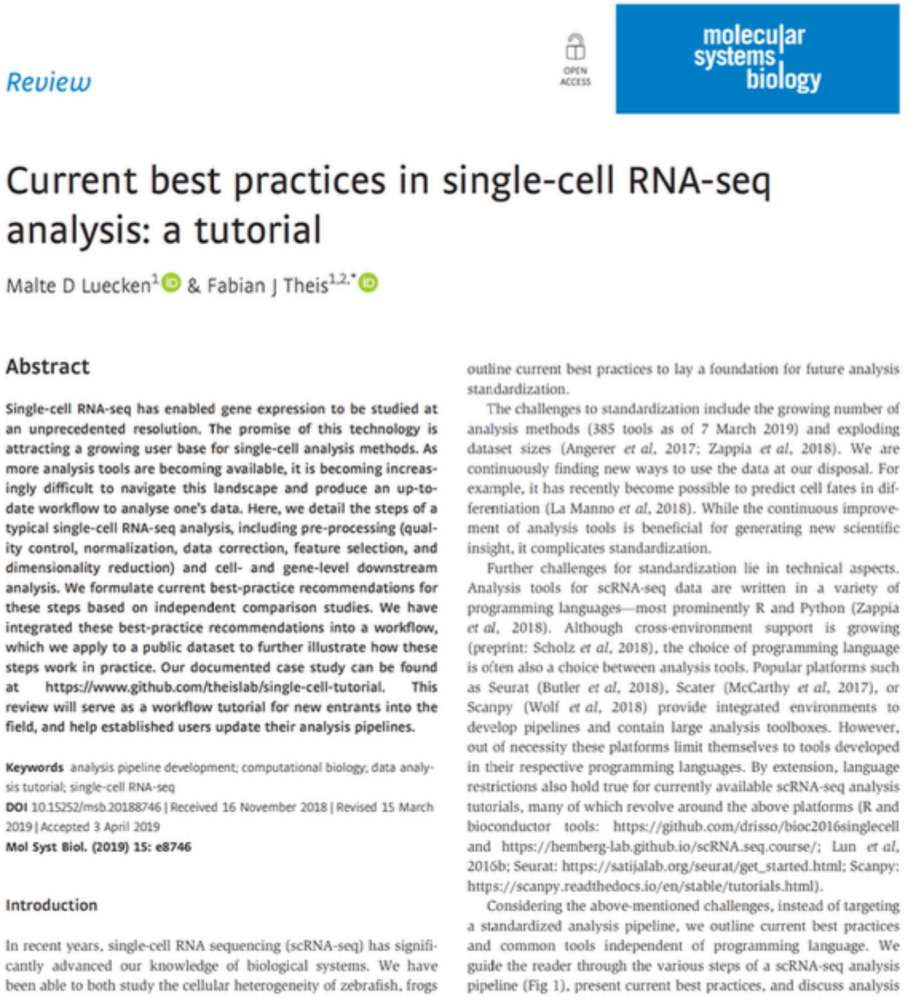
Simulated scRNA-seq data containing two batches with different cell types (with identical cell type proportions)



Simulated scRNA-seq data containing two batches with different cell types (with varying cell type proportions)



Recommended reading



Review

OPEN ACCESS

molecular systems biology

Current best practices in single-cell RNA-seq analysis: a tutorial

Malte D Luecken[1] & Fabian J Theis[1,2,*]

Abstract

Single-cell RNA-seq has enabled gene expression to be studied at an unprecedented resolution. The promise of this technology is attracting a growing user base for single-cell analysis methods. As more analysis tools are becoming available, it is becoming increasingly difficult to navigate this landscape and produce an up-to-date workflow to analyse one's data. Here, we detail the steps of a typical single-cell RNA-seq analysis, including pre-processing (quality control, normalization, data correction, feature selection, and dimensionality reduction) and cell- and gene-level downstream analysis. We formulate current best-practice recommendations for these steps based on independent comparison studies. We have integrated these best-practice recommendations into a workflow, which we apply to a public dataset to further illustrate how these steps work in practice. Our documented case study can be found at https://www.github.com/theislab/single-cell-tutorial. This review will serve as a workflow tutorial for new entrants into the field, and help established users update their analysis pipelines.

Keywords analysis pipeline development; computational biology; data analysis tutorial; single-cell RNA-seq

DOI 10.15252/msb.20188746 | Received 16 November 2018 | Revised 15 March 2019 | Accepted 3 April 2019

Mol Syst Biol. (2019) 15: e8746

Introduction

In recent years, single-cell RNA sequencing (scRNA-seq) has significantly advanced our knowledge of biological systems. We have been able to both study the cellular heterogeneity of zebrafish, frogs outline current best practices to lay a foundation for future analysis standardization.

The challenges to standardization include the growing number of analysis methods (385 tools as of 7 March 2019) and exploding dataset sizes (Angerer *et al*, 2017; Zappia *et al*, 2018). We are continuously finding new ways to use the data at our disposal. For example, it has recently become possible to predict cell fates in differentiation (La Manno *et al*, 2018). While the continuous improvement of analysis tools is beneficial for generating new scientific insight, it complicates standardization.

Further challenges for standardization lie in technical aspects. Analysis tools for scRNA-seq data are written in a variety of programming languages—most prominently R and Python (Zappia *et al*, 2018). Although cross-environment support is growing (preprint: Scholz *et al*, 2018), the choice of programming language is often also a choice between analysis tools. Popular platforms such as Seurat (Butler *et al*, 2018), Scater (McCarthy *et al*, 2017), or Scanpy (Wolf *et al*, 2018) provide integrated environments to develop pipelines and contain large analysis toolboxes. However, out of necessity these platforms limit themselves to tools developed in their respective programming languages. By extension, language restrictions also hold true for currently available scRNA-seq analysis tutorials, many of which revolve around the above platforms (R and bioconductor tools: https://github.com/drisso/bioc2016singlecell and https://hemberg-lab.github.io/scRNA.seq.course/; Lun *et al*, 2016b; Seurat: https://satijalab.org/seurat/get_started.html; Scanpy: https://scanpy.readthedocs.io/en/stable/tutorials.html).

Considering the above-mentioned challenges, instead of targeting a standardized analysis pipeline, we outline current best practices and common tools independent of programming language. We guide the reader through the various steps of a scRNA-seq analysis pipeline (Fig 1), present current best practices, and discuss analysis

https://www.embopress.org/doi/full/10.15252/msb.20188746

Acknowledgements

I thank the Krishnaswamy's group (https://www.krishnaswamylab.org/workshop), the Theis group (Maren Buttner) and the Elisabetta Mereu for providing some slides for this presentation