Single-cell data analysis I: QC, normalization, inference

Davide Risso

CSAMA 2019

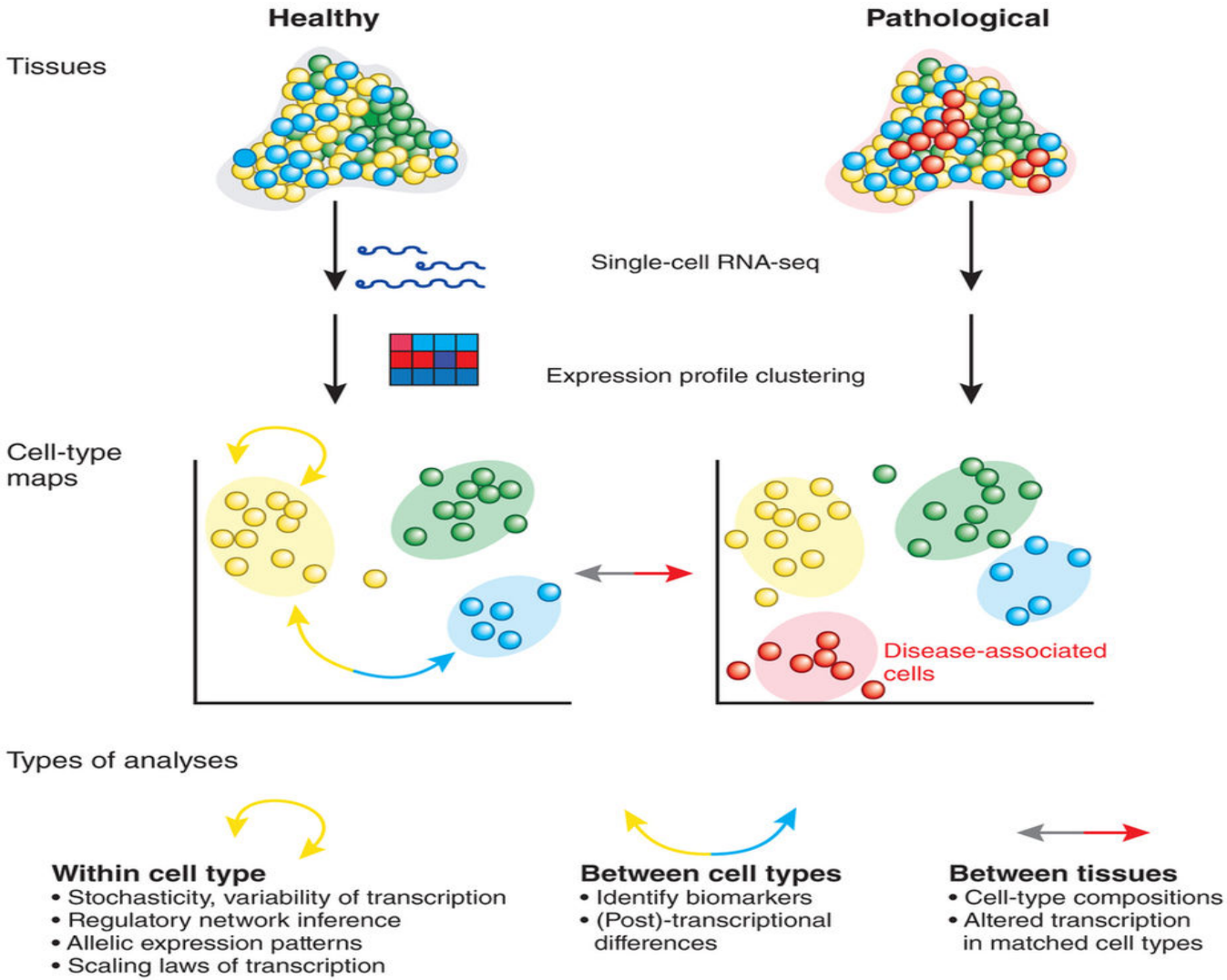
UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI SCIENZE
STATISTICHE

1. Introduction
2. Model specification
3. Quality Control and Filtering
4. Normalization
5. Dimensionality reduction
6. Lineage Inference

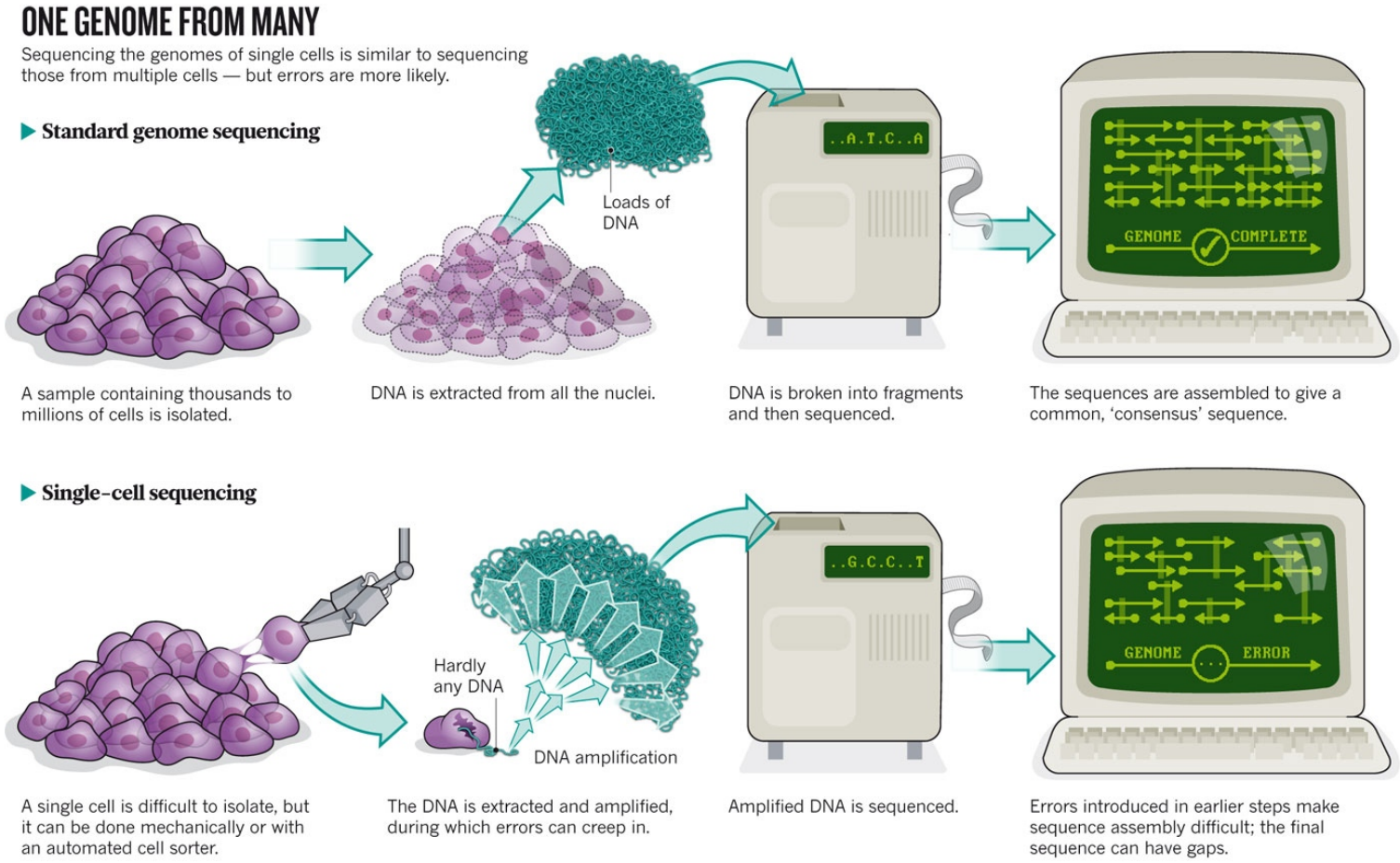
Introduction

Single-cell RNA-seq



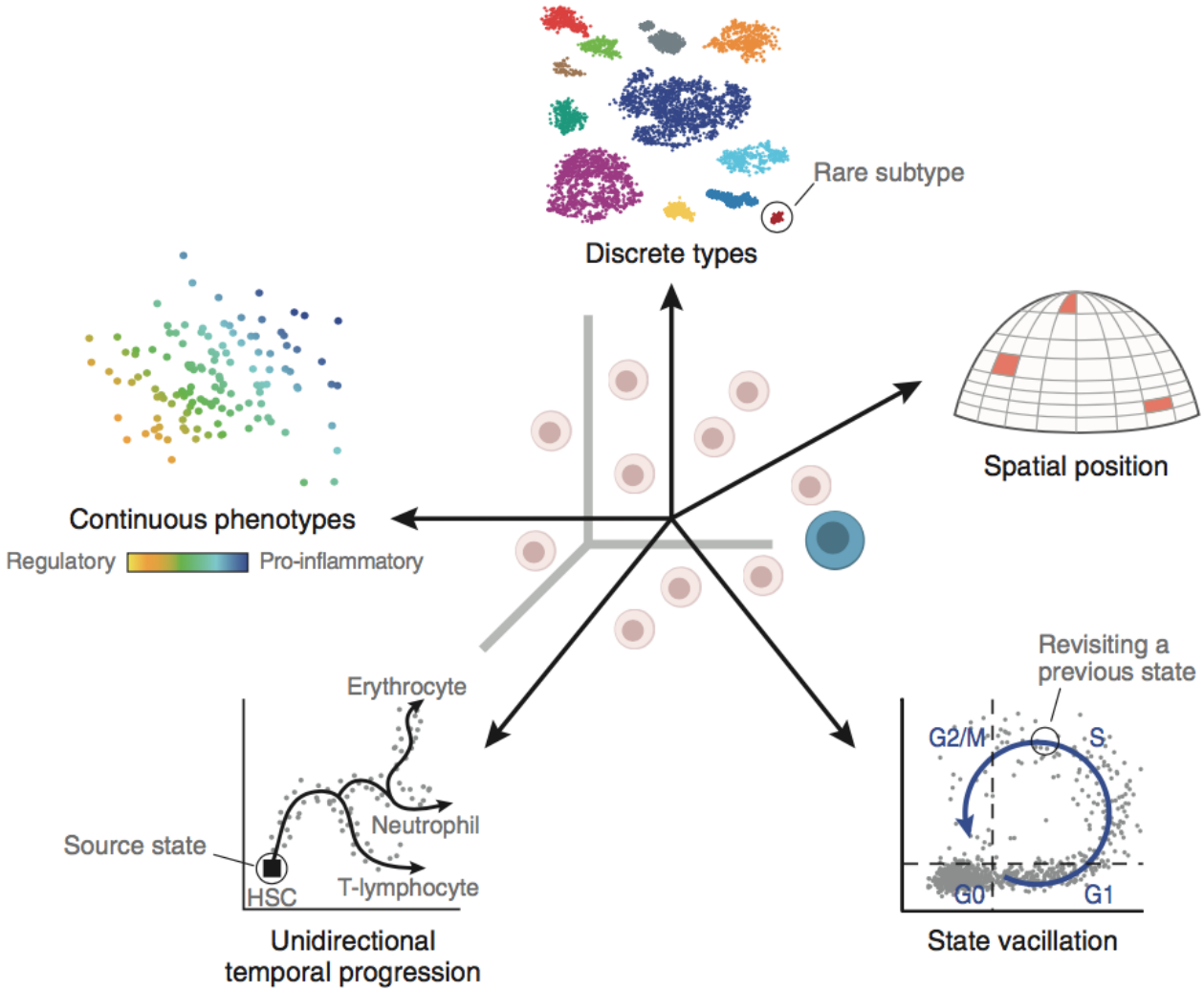
Sandberg (2014). *Nature Methods*

Single-cell signal is noisy



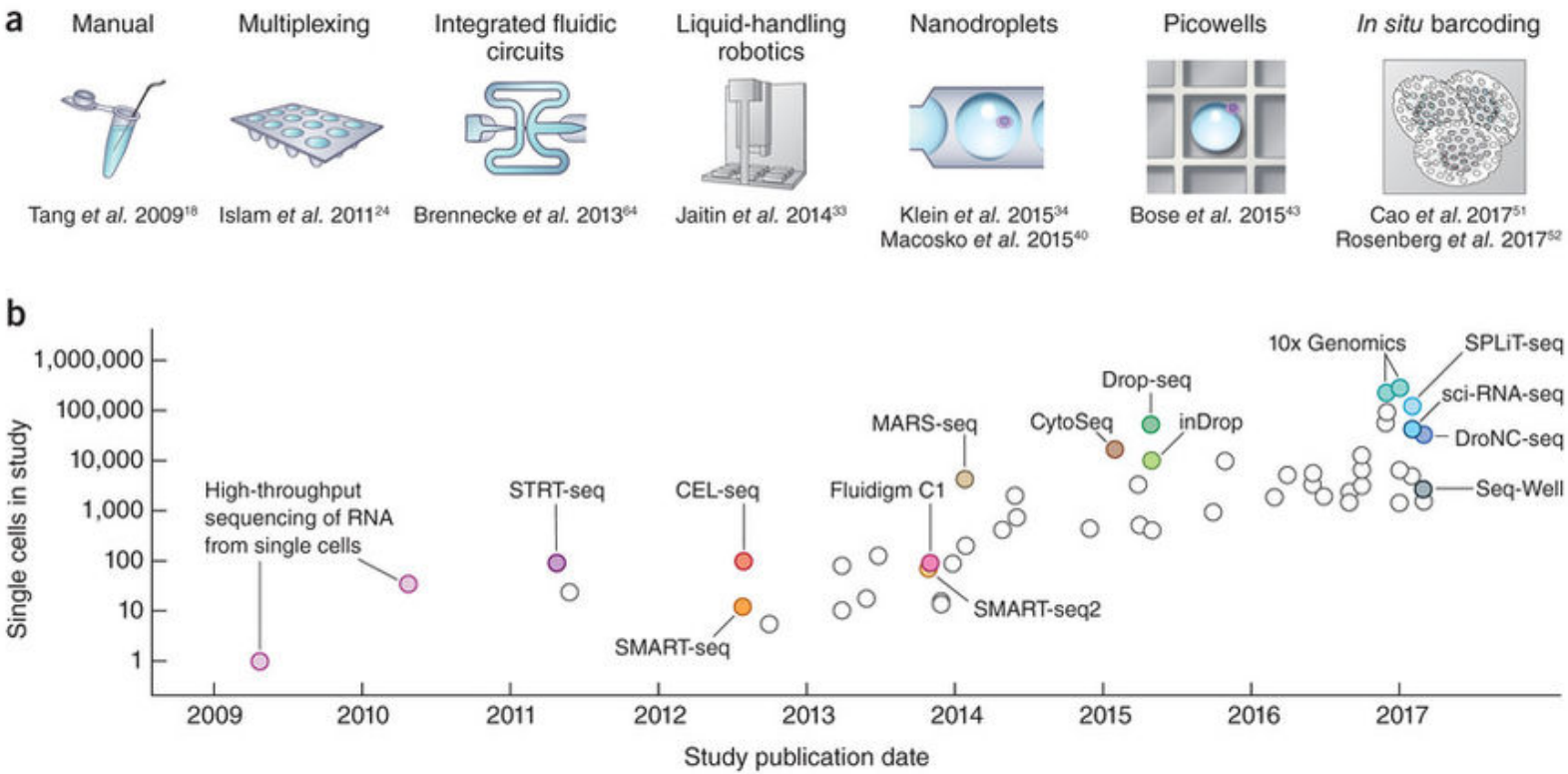
Owens (2012). *Nature*

Single-cell data let us ask new questions



Wagner, Regev, Yosef (2016). *Nature Biotechnology*

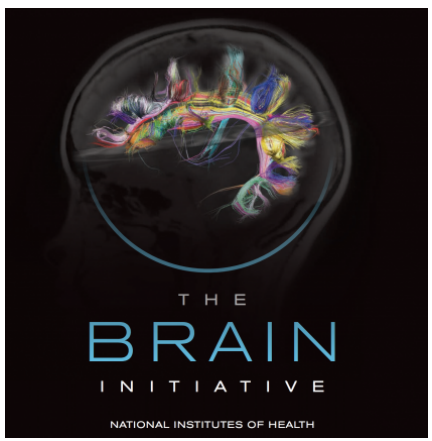
Single-cell data meets big data



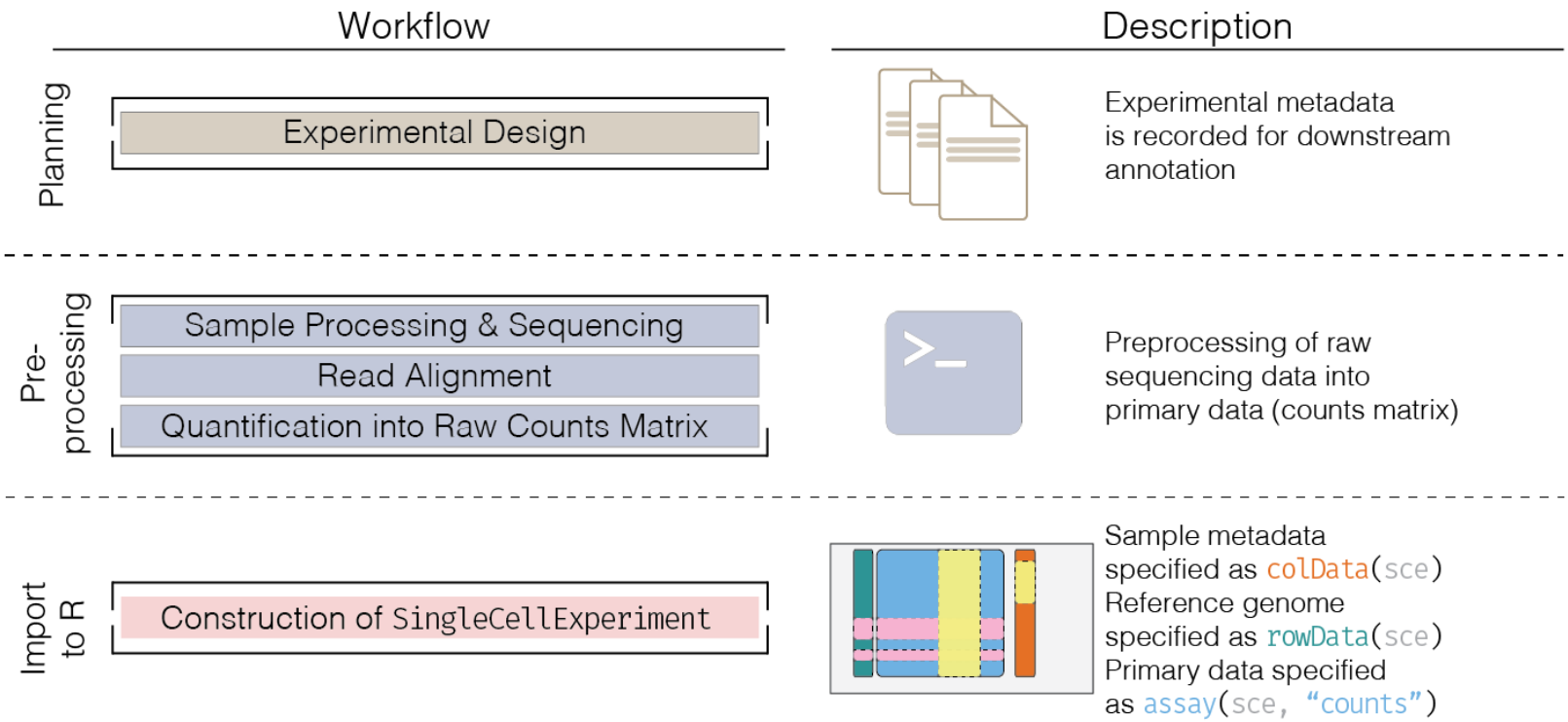
Svensson, Vento-Tormo, Teichmann (2018). *Nature Protocols*

Single-cell meets big data

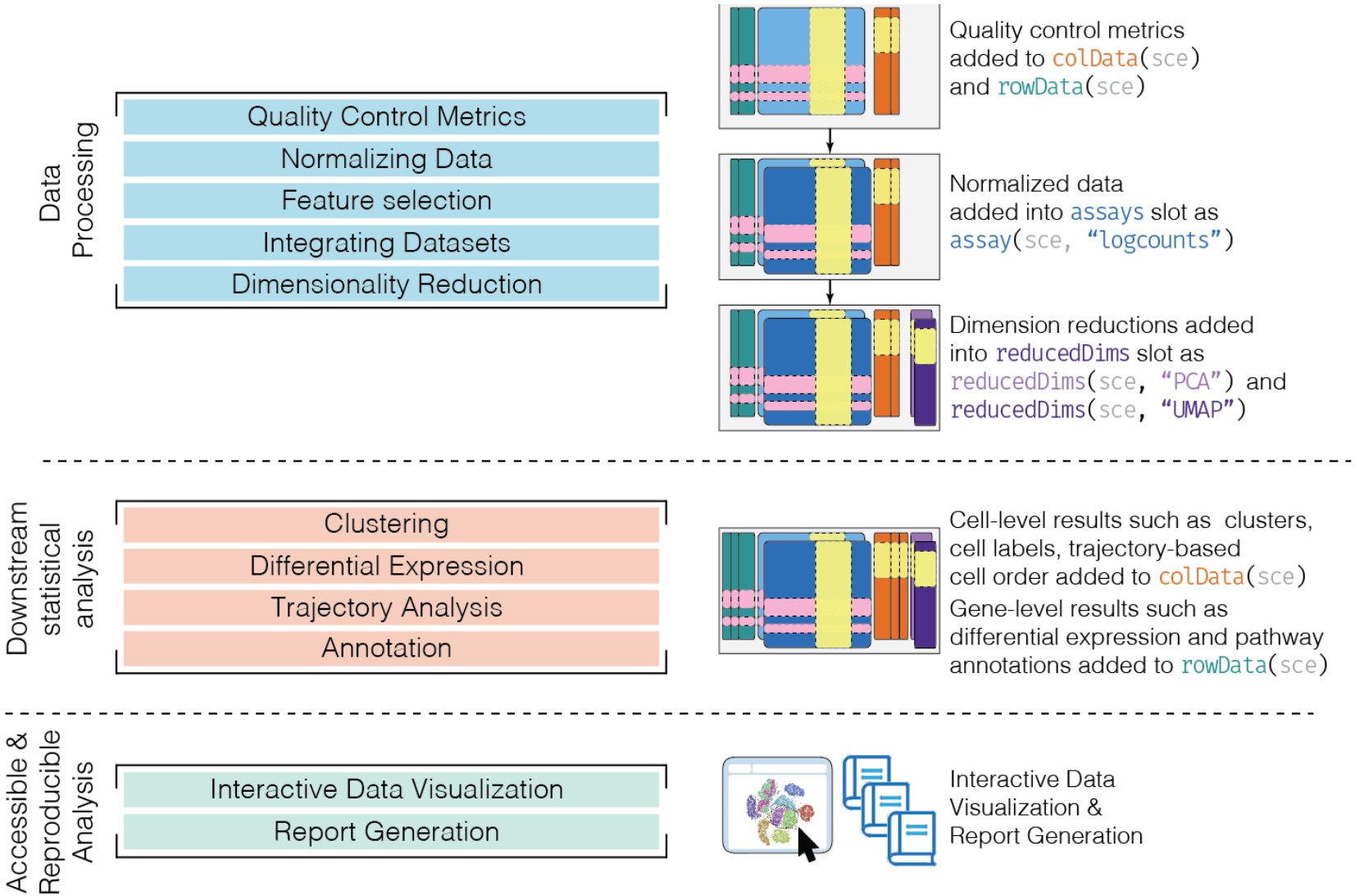
- BRAIN Initiative "mini" brain atlas
  - $340{,}000$ cells and nuclei from the Mouse Primary Motor Cortex
  - Plans to sequence 3M cells for the whole brain (less than 1%).
- The Human Cell Atlas "preview" dataset
  - $530{,}000$ cells from umbilical cord blood and bone marrow
  - Millions expected "soon".



A typical workflow

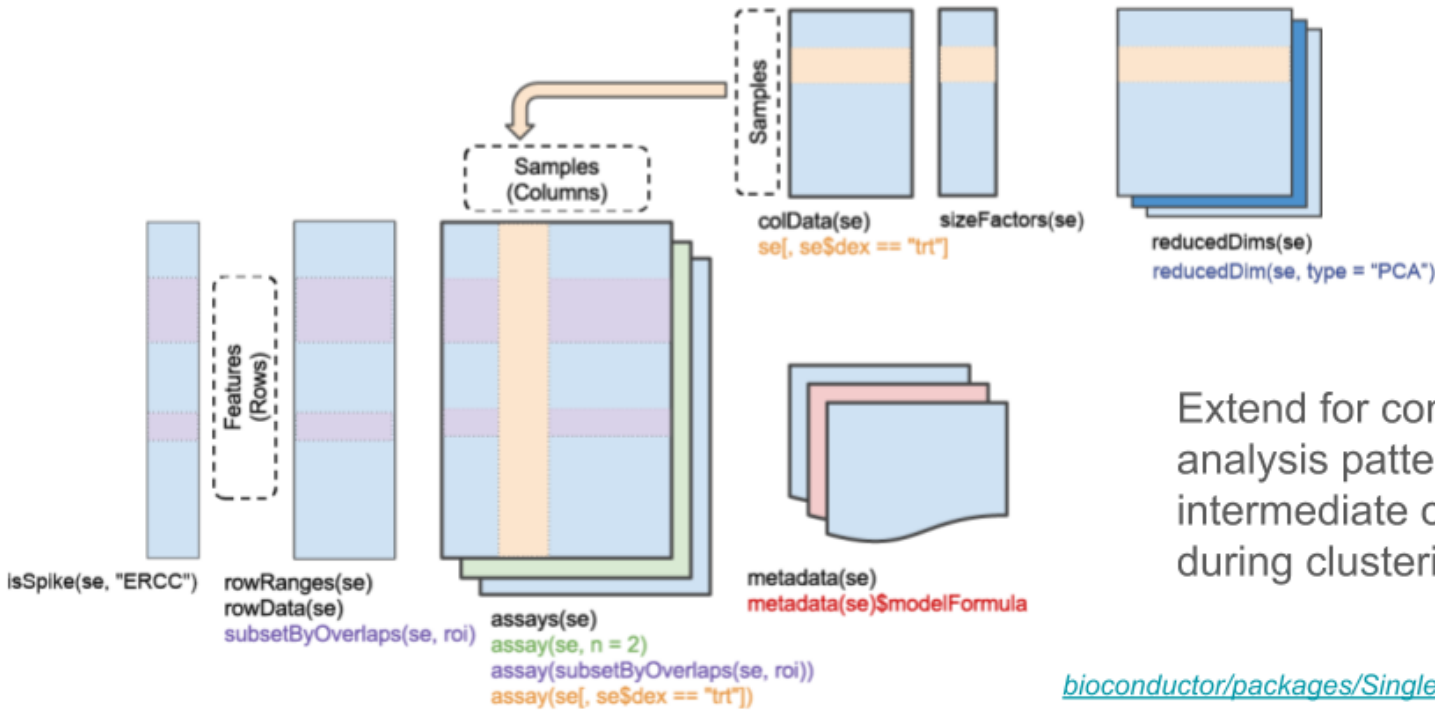


A typical workflow



The SingleCellExperiment class

Common data structures for single-cell data



Extend for common analysis patterns, e.g., intermediate objects during clustering.

bioconductor/packages/SingleCellExperiment

The SingleCellExperiment class

```
sce
```

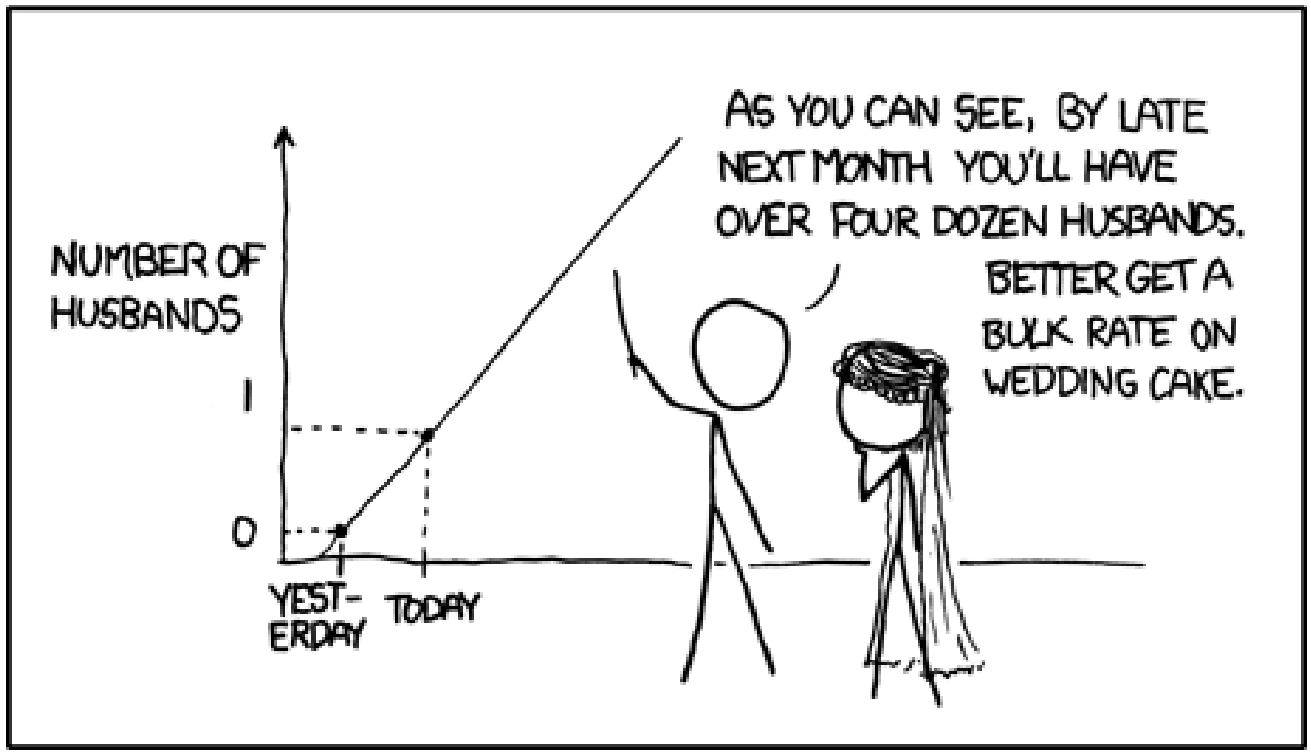
```
## class: SingleCellExperiment
## dim: 3079 1000
## metadata(1): log.exprs.offset
## assays(2): counts logcounts
## rownames(3079): ENSG00000188976 ENSG00000187608 ...
##   ENSG00000198727 ENSG00000220023
## rowData names(12): ENSEMBL_ID Symbol_TENx ... total_counts
##   log10_total_counts
## colnames(1000): Cell1 Cell2 ... Cell999 Cell1000
## colData names(56): Sample Barcode ...
##   pct_counts_in_top_200_features_mito
##   pct_counts_in_top_500_features_mito
## reducedDimNames(2): PCA zinbwave
## spikeNames(0):
```

Resources

- A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor
  - https://f1000research.com/articles/5-2122/v2
- Bioconductor workflow for single-cell RNA sequencing
  - https://f1000research.com/articles/6-1158/v1
- github.com/seandavi/awesome-single-cell
- scrna-tools.org
- Seurat
  - https://satijalab.org/seurat/
- Bioconductor workshop materials
  - https://bioconductor.org/help/course-materials/
- Orchestrating Single Cell Analysis review
  - https://www.biorxiv.org/content/10.1101/590562v1.abstract
  - https://osca.bioconductor.org

Model specification

MY HOBBY: EXTRAPOLATING



Statistical Inference

Statistical inference is the process of *learning some properties of the population* starting *from a sample* drawn from this population.

For instance, we may be interested in learning about the difference in the gene expression of immune cells among cancer patients, but we cannot measure the whole population.

We can however measure the expression of a *random sample* of the population and then *infer* or generalize the results to the entire population.

Statistical Inference

There are some terms that we need to define.

- The *data generating distribution* is the *unknown* probability distribution that generates the data.
- The *empirical distribution* is the *observable* distribution of the data in the sample.

We are usually interested in a *function* of the data generating distribution. This is often referred to as *parameter* (or the parameter of interest).

We use the sample to estimate the parameter of interest, using a function of the empirical distribution, referred to as *estimator*.

Data generating distribution

The data generating distribution is unknown.

However, we can make *some assumptions* about it. These assumptions are sometimes based on domain knowledge or on mathematical convenience.

One commonly used strategy is to assume a *family of distributions* for the data generating distribution, for instance the *Gaussian distribution*.

Data generating distribution for RNA-seq

The sequencing process can be seen as a *simple random sampling* of the reads along the genome.

Hence, if we sequence a total number w of reads, we can model the number of reads mapped to a gene j as

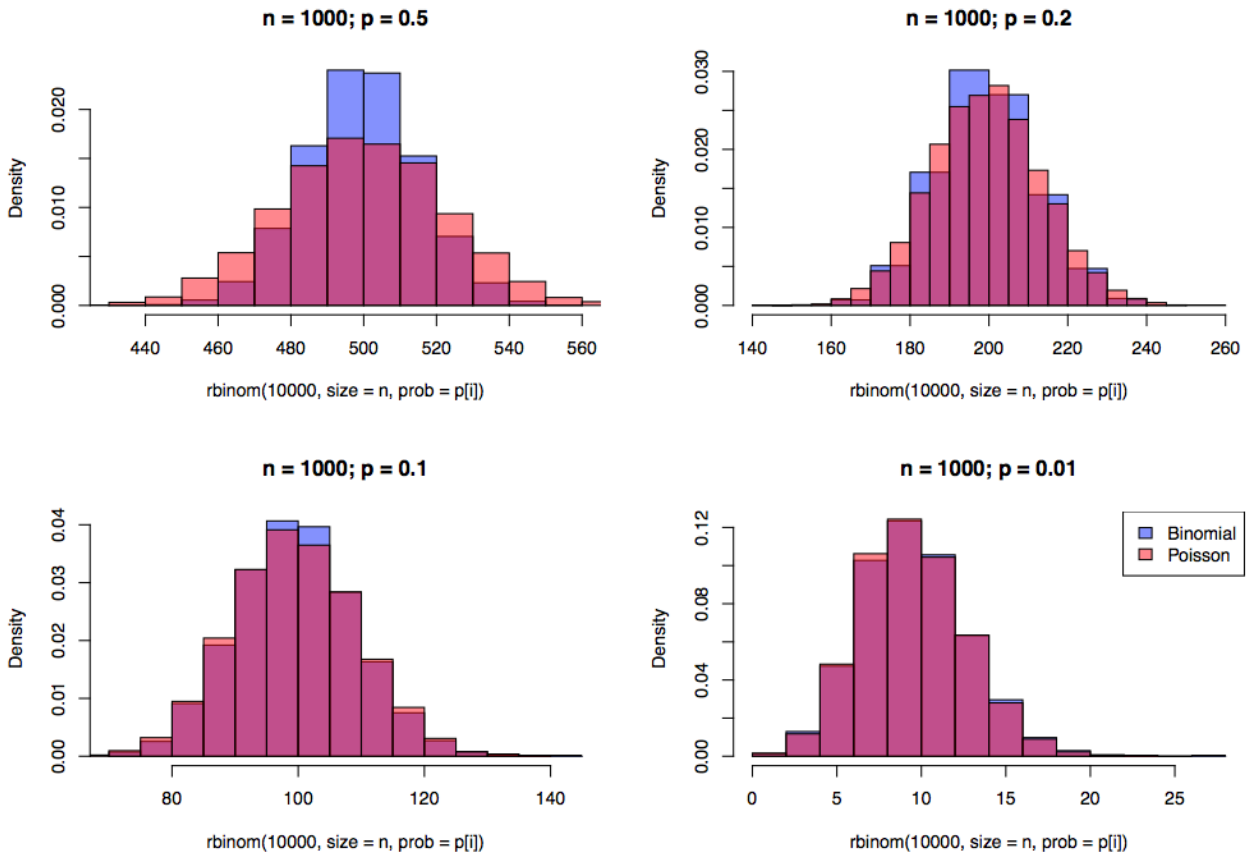
$$X_j \sim Bi(w, p_j = \theta_j \, l_j),$$

where p_j is proportional to the *number of RNA copies* for gene j (θ_j) and to its length (l_j).

Since w is big and p_j is small, the binomial is well approximated by the Poisson distribution

$$X_j \sim Poi(\lambda = w \, \theta_j \, l_j)$$

Poisson approximation



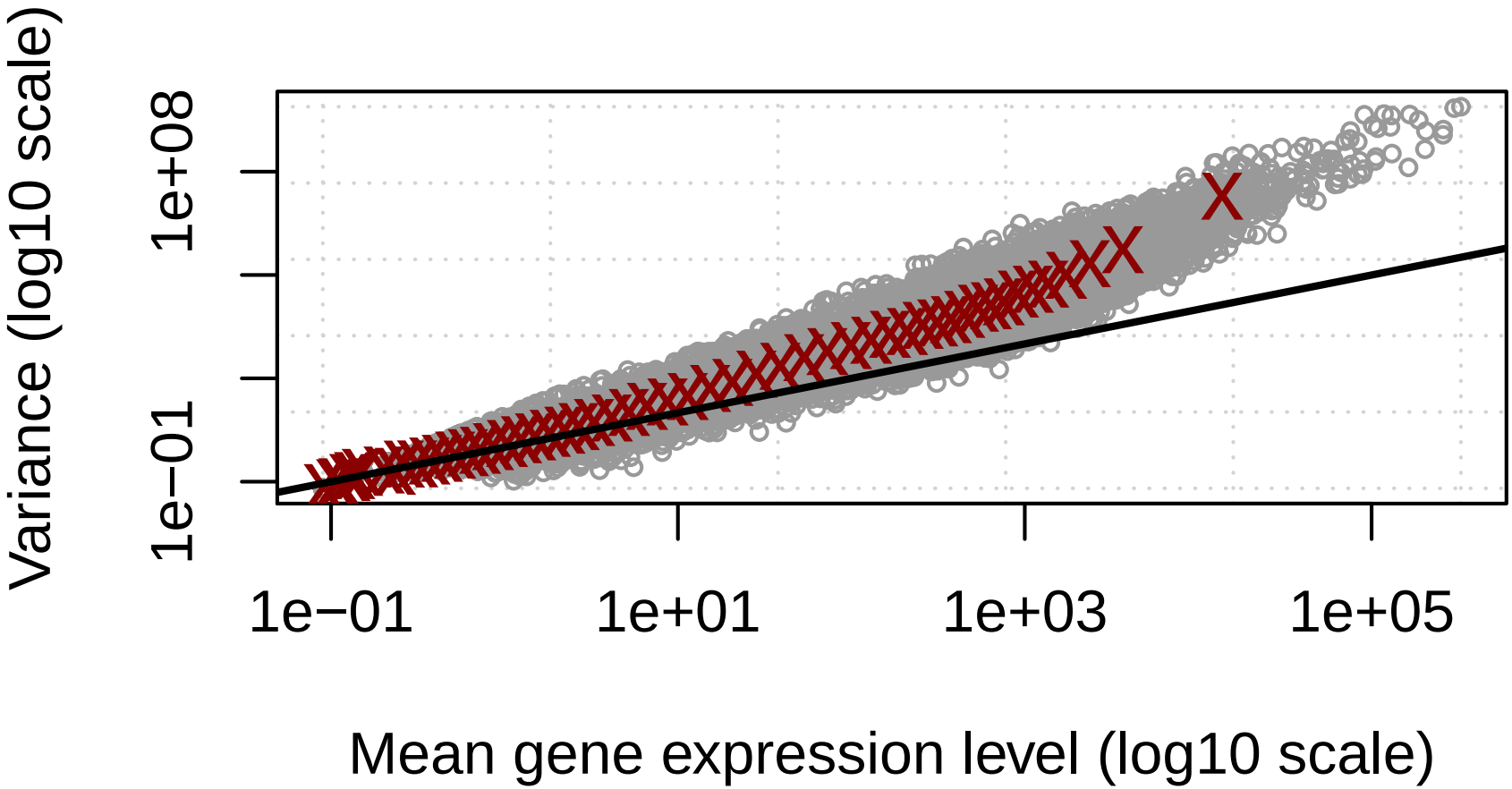
Overdispersion

The Poisson distribution has one important property:

$$E[Y] = Var(Y) = \lambda.$$

Because of biological variability, RNA-seq counts exhibit higher variance, leading to *overdispersion*.

Overdispersion



The negative binomial distribution

For any $\mu \geq 0$ and $\phi > 0$, the probability mass function (PMF) of the negative binomial (NB) distribution is

$$f_{NB}(y; \mu, \phi) = \frac{\Gamma(y + \phi^{-1})}{\Gamma(y+1)\Gamma(\phi^{-1})} \left(\frac{1}{1+\mu\phi}\right)^{\phi^{-1}} \left(\frac{\mu}{\mu + \phi^{-1}}\right)^{y}, \quad \forall y \in \mathbb{N}.$$

The mean of the NB distribution is μ and its variance is:

$$Var(Y) = \mu + \phi\,\mu^2.$$

In particular, the NB distribution boils down to a Poisson distribution when $\phi \to 0$.

Interpretation in the context of RNA-seq

The negative binomial can be derived as a *Gamma-Poisson mixture*.

$$Y|\lambda \sim Poi(\lambda)$$

e

$$\lambda \sim Ga\left(\frac{1}{\phi}, \frac{1}{\phi\mu}\right)$$

hence

$$Y \sim NB(\mu, \phi).$$

Interpretation in the context of RNA-seq

In other words,

- λ_{ij} can be interpreted as the true (unobserved) value of gene j in sample i;
- For each gene j, λ_{ij} varies between samples according to a Gamma distribution (biological variation).
- The observed counts, Y_{ij} are the results of biological variation $+$ technical variation due to the sequencing process, which can be modeled as a Poisson.

Interpretation in the context of RNA-seq

Recall the the variance function can be written as

$$V(\mu) = \mu + \phi\mu^2.$$

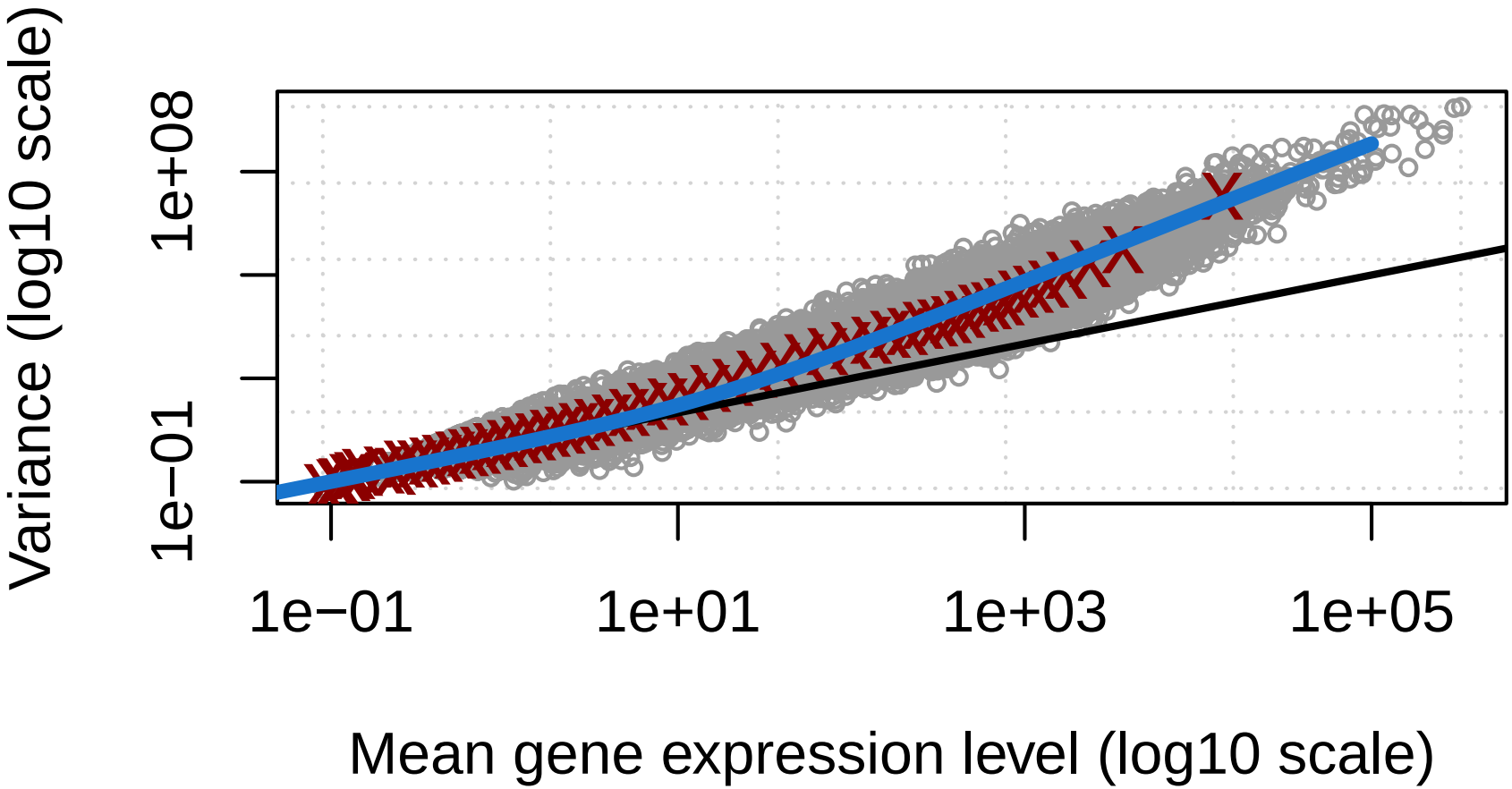
Dividing for the square of the expected value, we obtain the square of the *coefficient of variation*.

$$CV^2 = \frac{1}{\mu} + \phi.$$

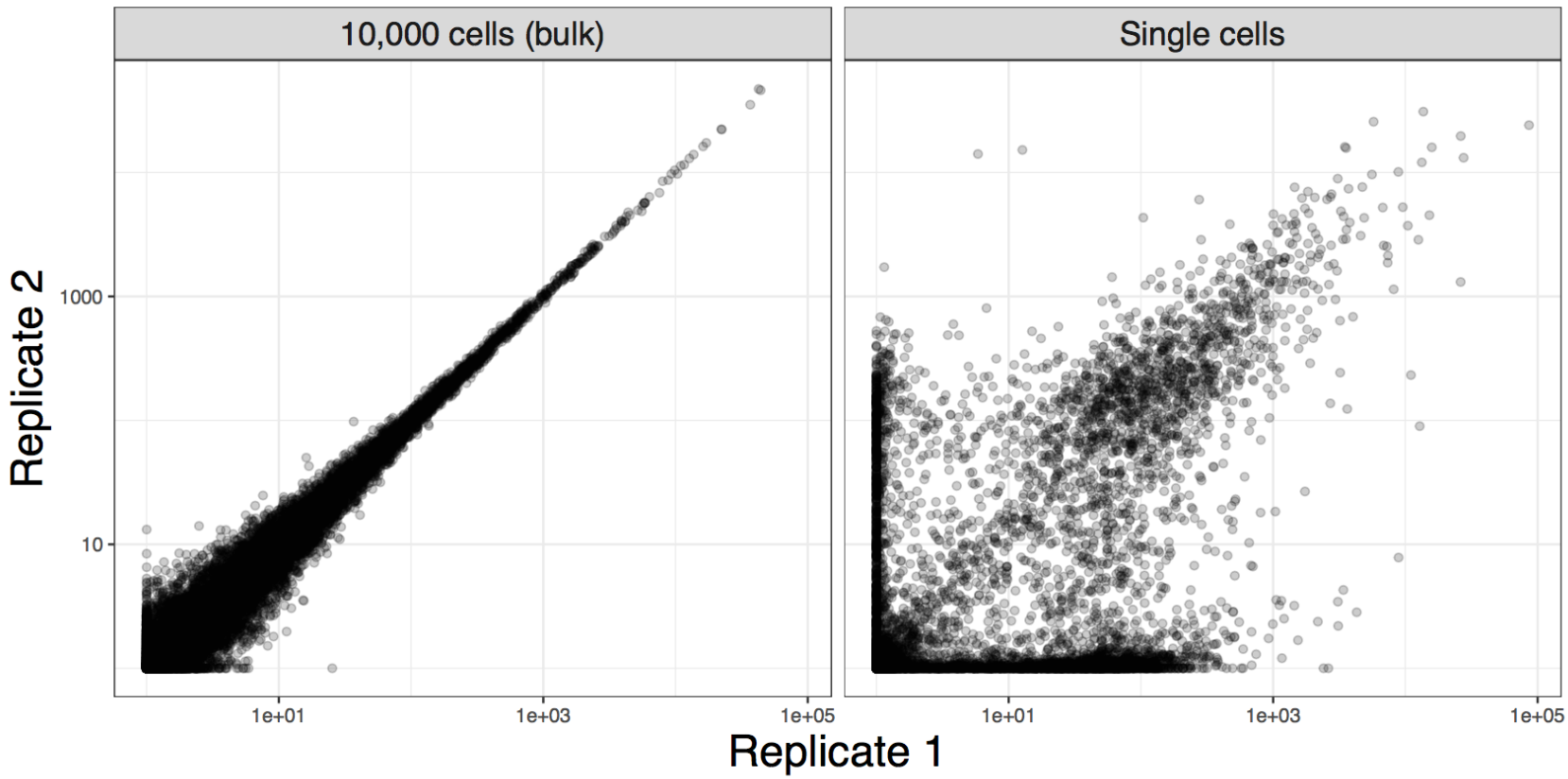
CV^2 can be interpreted as the sum of two terms:

- The first, given by the Poisson distribution, describes technical variability and tends to 0 as we sequence more reads;
- The second, which does not depend on the mean, represents biological variability.

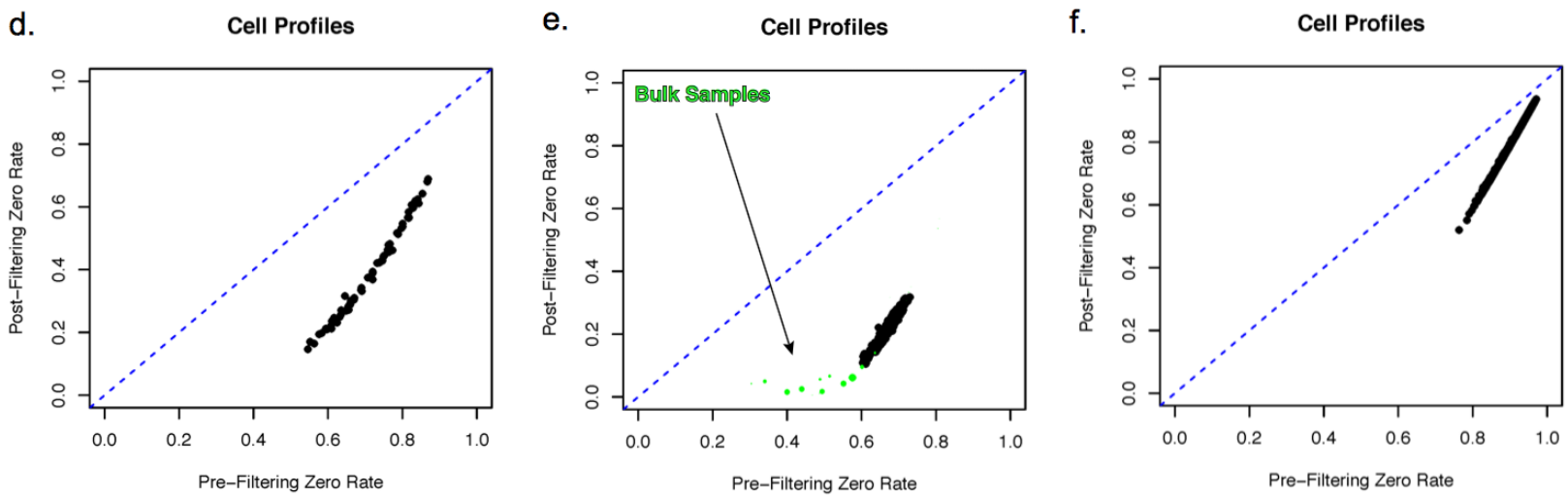
Interpretation in the context of RNA-seq



Enters single-cell RNA-seq



Single-cell data have more zeros than bulk RNA-seq



Cole et al. (2019). *Cell Systems*

The zero-inflated negative binomial

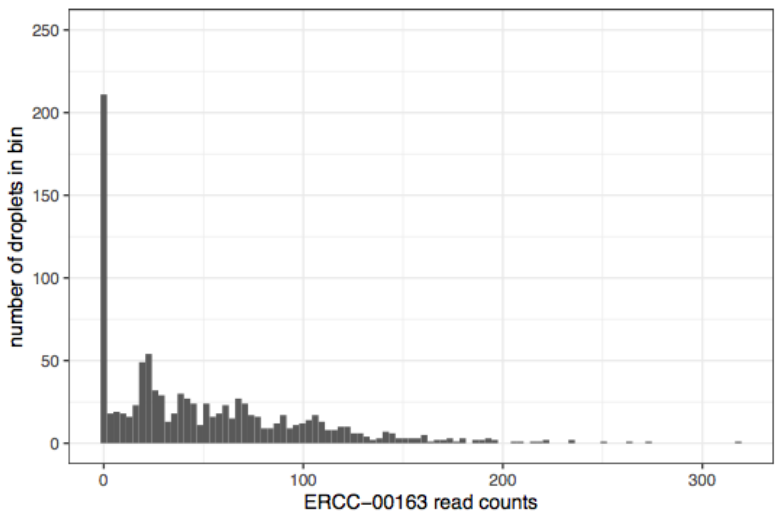
For any $\pi \in [0, 1]$, the PMF of the zero-inflated negative binomial (ZINB) distribution is given by

$$f_{ZINB}(y; \mu, \theta, \pi) = \pi \delta_0(y) + (1 - \pi) f_{NB}(y; \mu, \theta), \quad \forall y \in \mathbb{N},$$

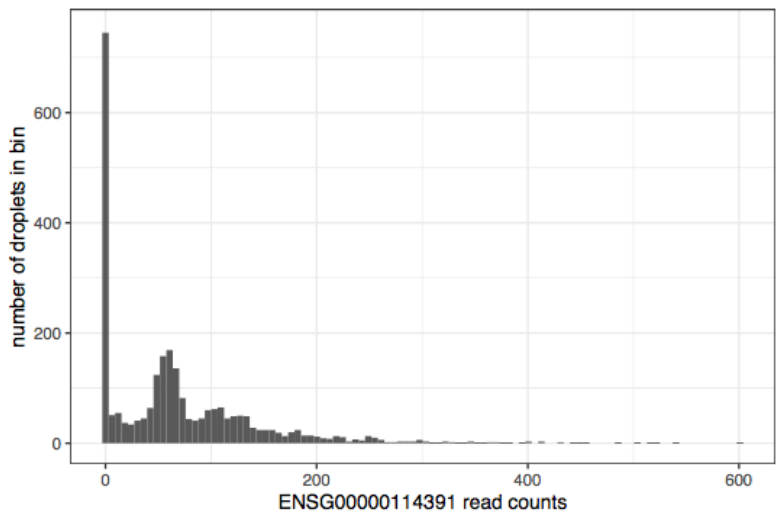
where $\delta_0(\cdot)$ is the Dirac function.

Here, π can be interpreted as the probability that a 0 is observed instead of the actual count, resulting in an inflation of zeros compared to the NB distribution, hence the name ZINB.

Single-cell RNA-seq read counts



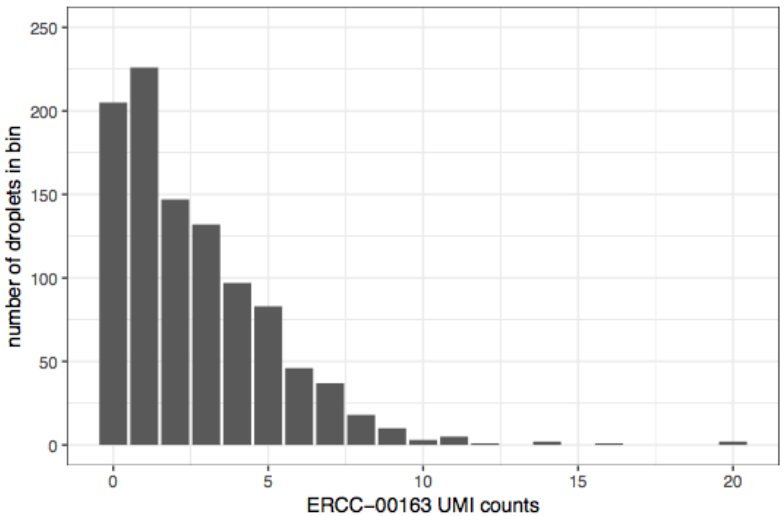
(a) Read counts- technical replicates



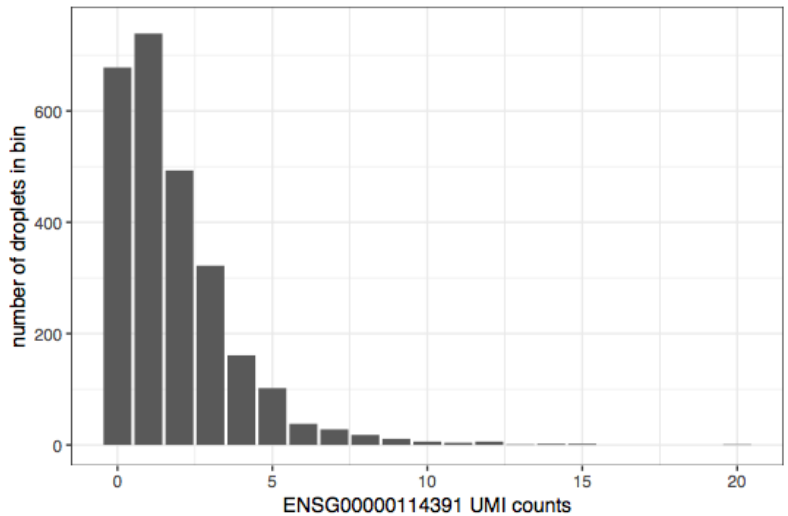
(b) Read counts- biological replicates

Townes et al. (2019). *bioRxiv*

UMIs help!



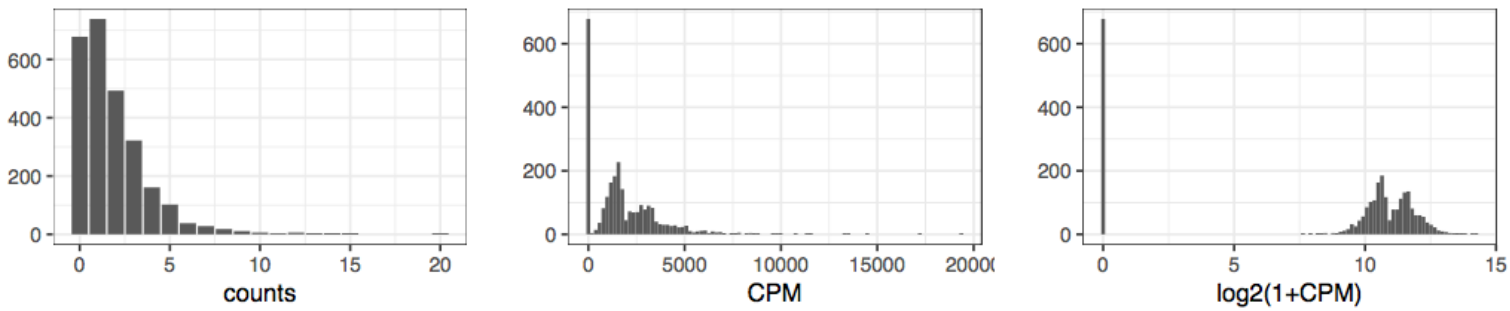
(c) UMI counts- technical replicates



(d) UMI counts- biological replicates

Townes et al. (2019). *bioRxiv*

Log transformation does not!



(a) UMI counts (b) counts per million (CPM) (c) log of CPM

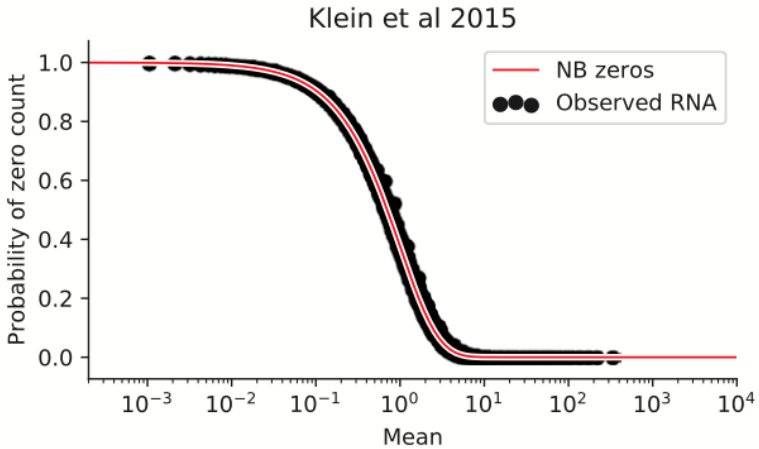
Townes et al. (2019). *bioRxiv*

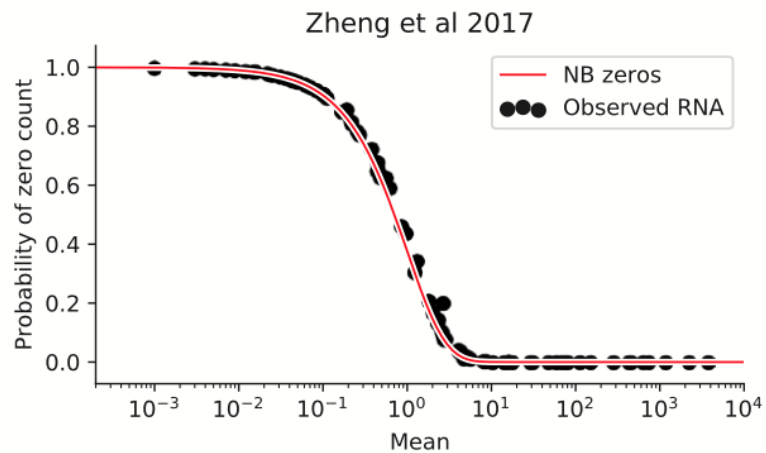
Do we need to account for the extra zeros in the model?

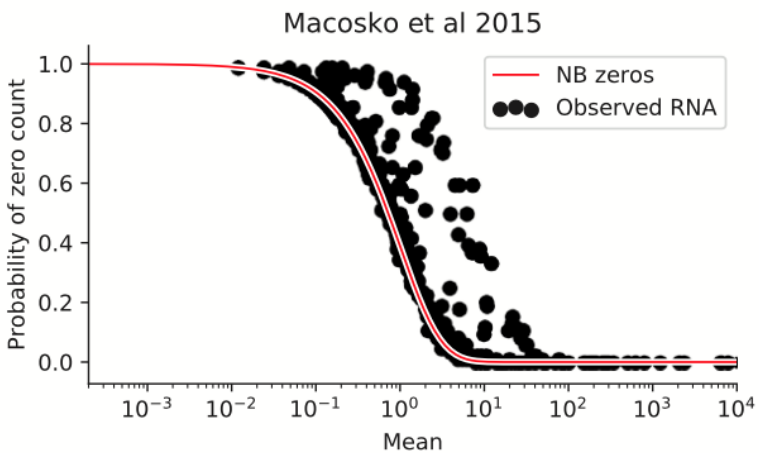
- Non-UMI data: yes!
- UMI data: probably not...

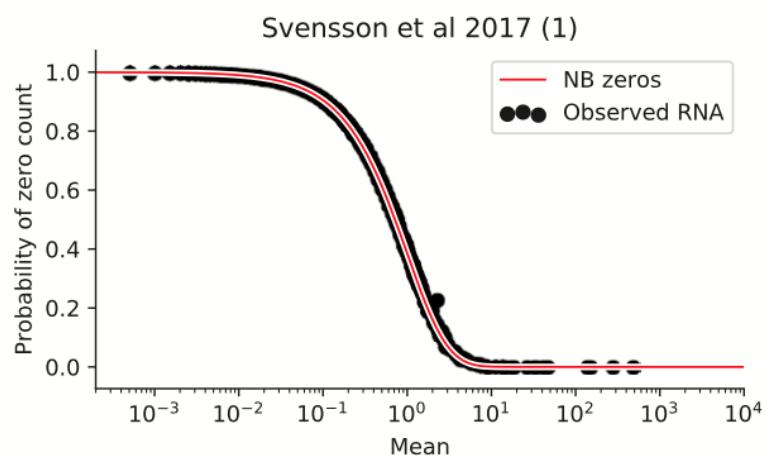
Recent results suggest that in UMI data, in particular in droplet-based data, zero-inflation may not be an important issue.

Do we need to account for the extra zeros in the model?









Svensson (2019). *bioRxiv*

Quality Control and Filtering


DO YOUR DATA SPARK JOY?
imgflip.com

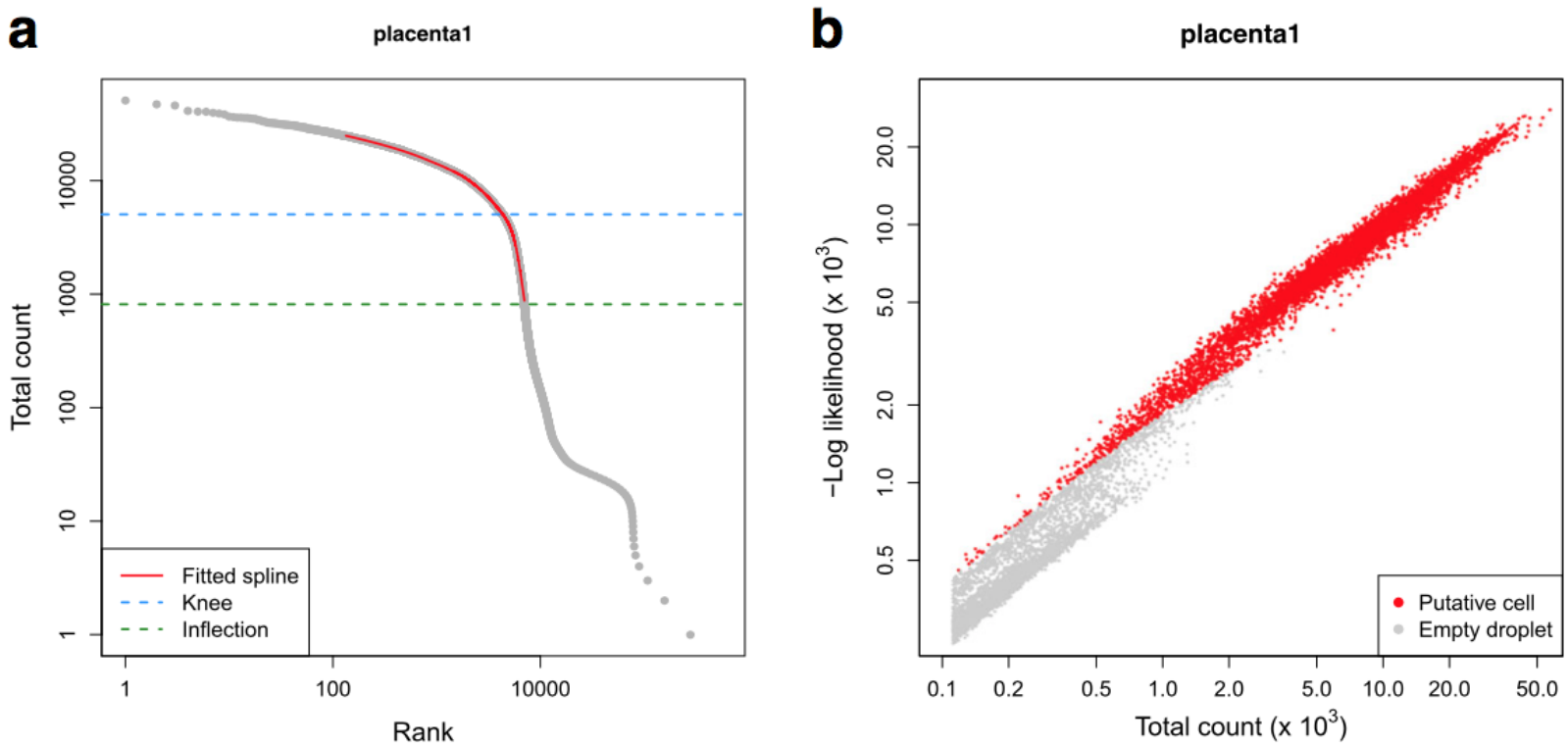
Quality Control and Filtering

Exploratory data analysis (EDA) and quality control (QC) are of utmost importance in genomics.

With single cell data we have the luxury of having a large number of samples, hence we can filter out low quality cells as well as lowly expressed genes.

There are some simple metrics that we can compute as a proxy of the quality of the samples.

Identifying empty droplets

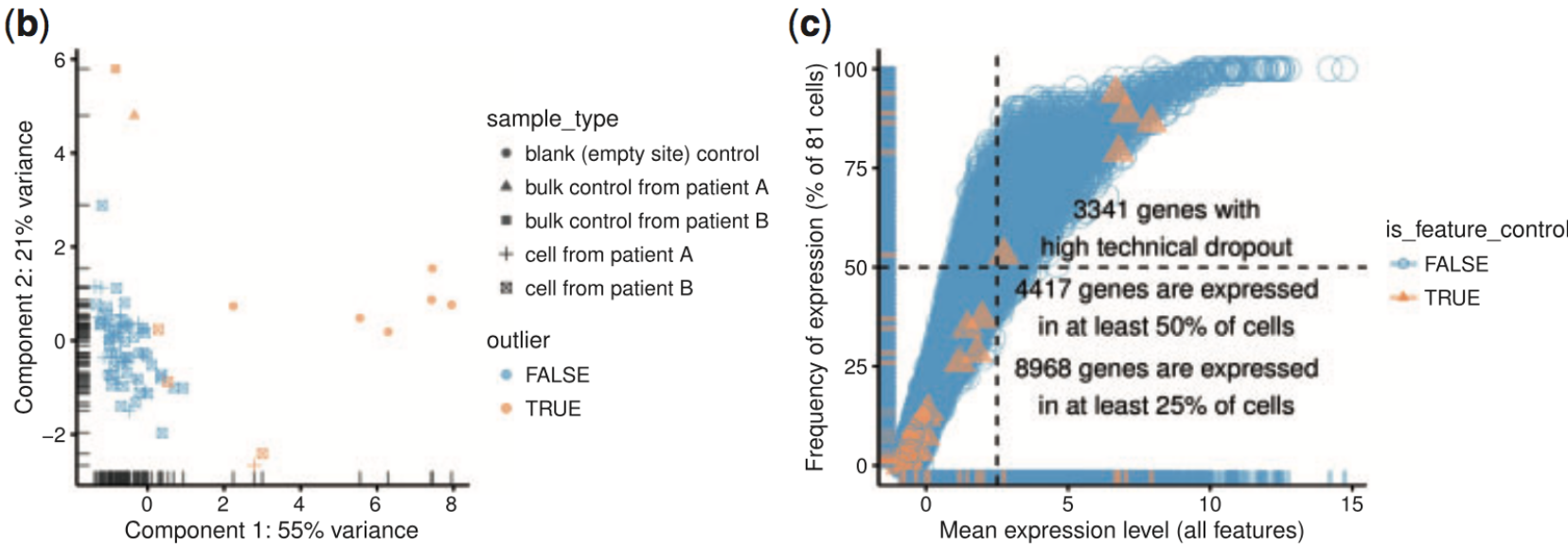


DropletUtils Bioconductor package

Computing QC metrics

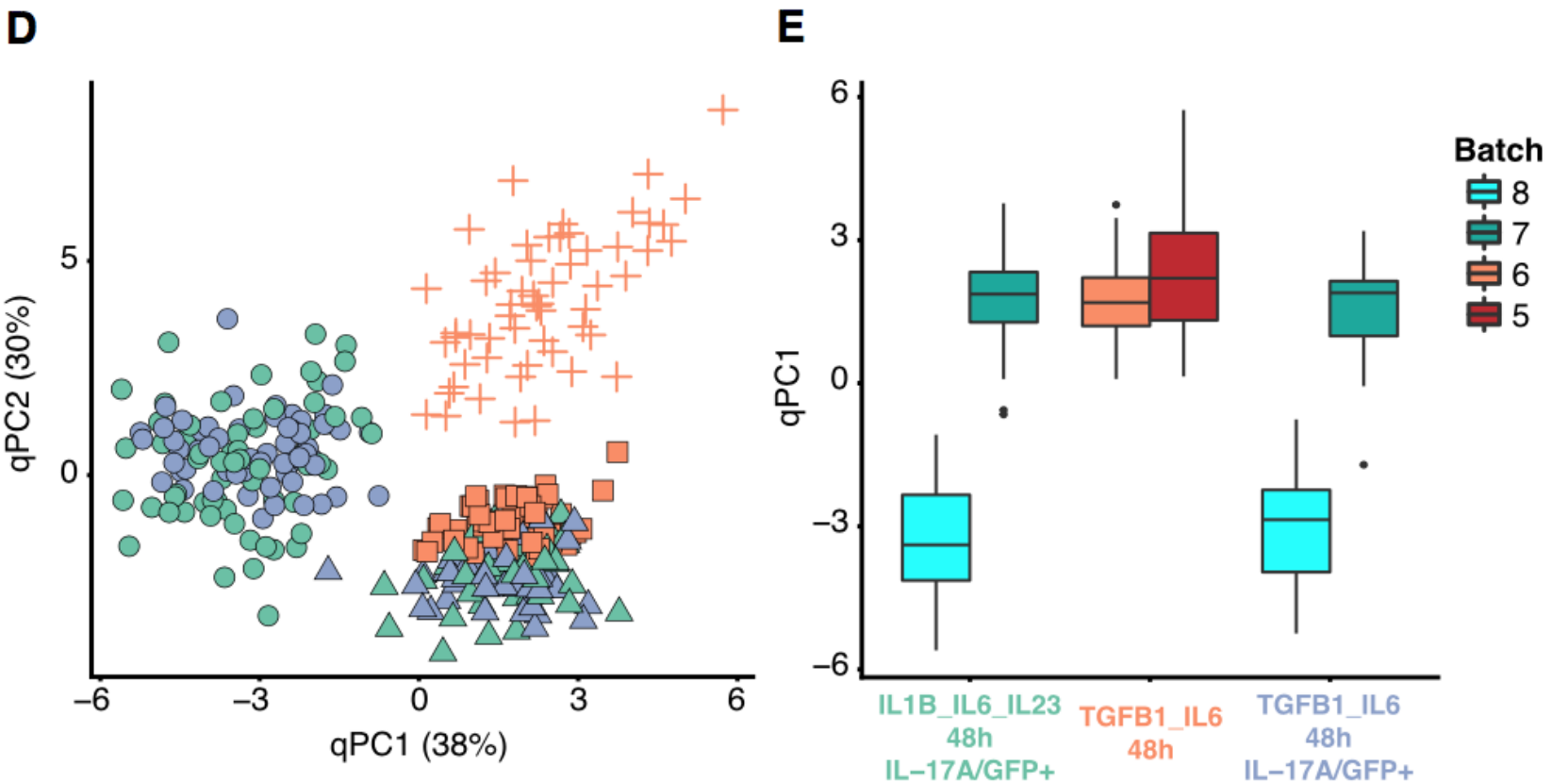
```
sce <- TENxPBMCData::TENxPBMCData("pbmc4k")
sce <- scater::calculateQCMetrics(sce)
```

Filtering genes and samples



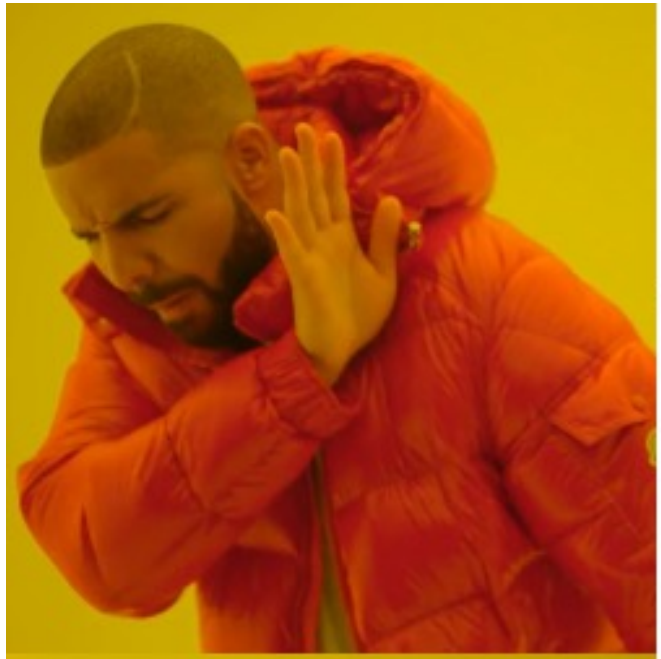
scater Bioconductor package

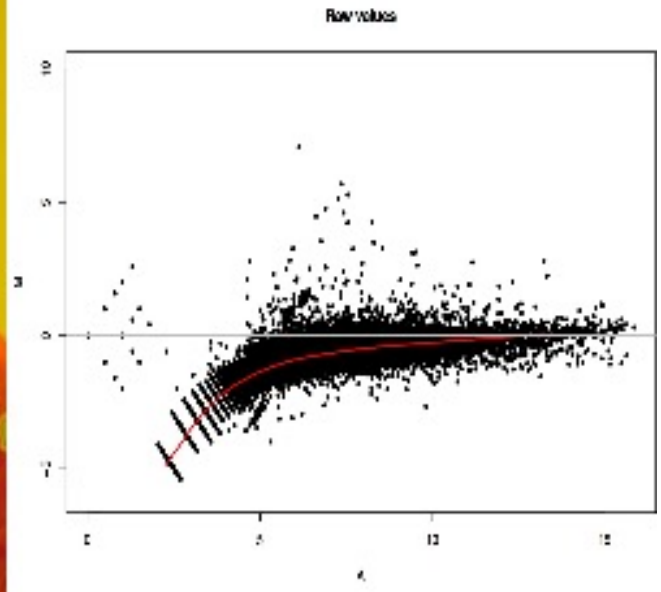
Exploring batch effects



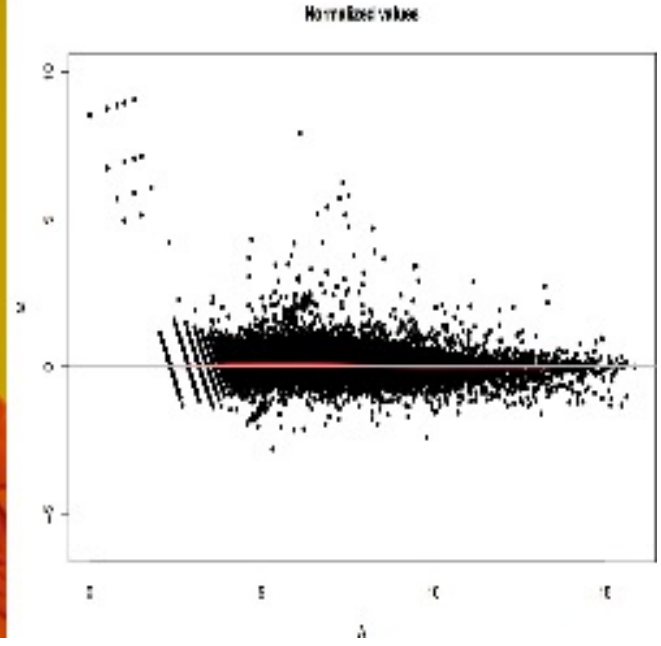
scone Bioconductor package

Normalization



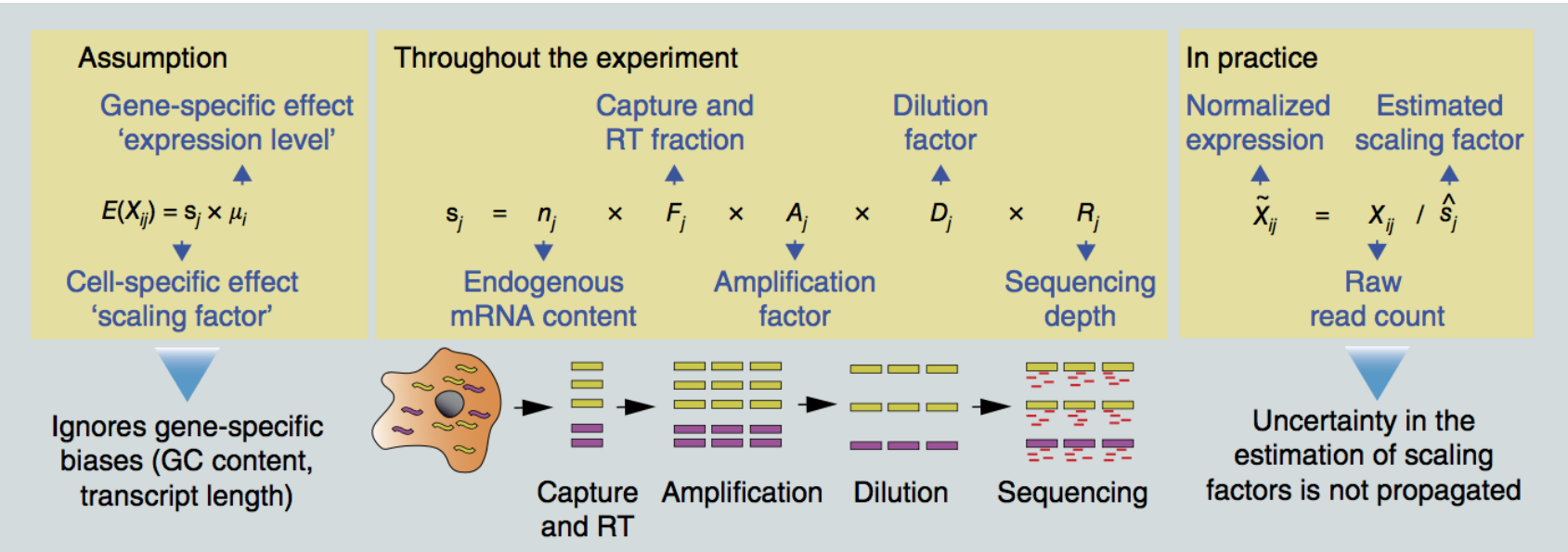

Raw values


imgflip.com


Normalized values

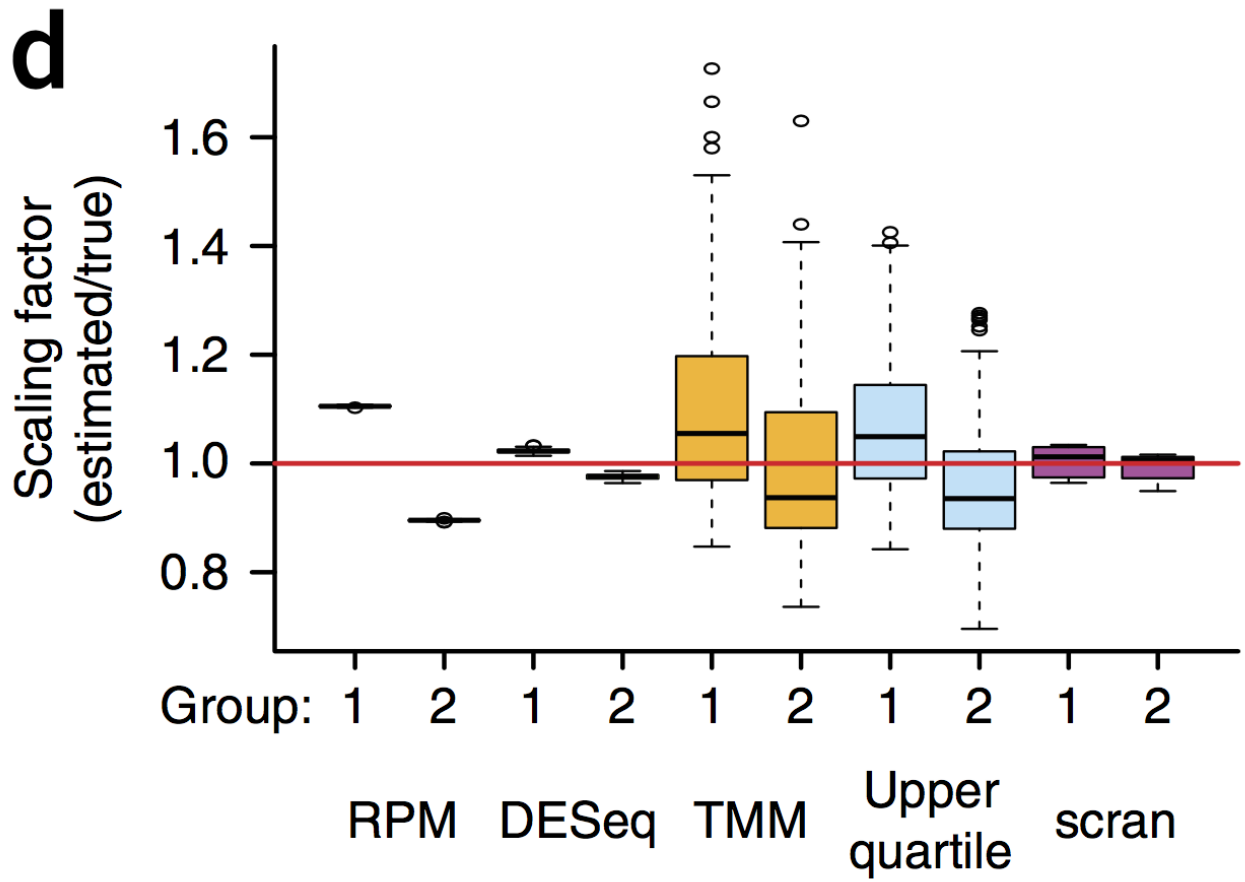
Normalization

As with bulk RNA-seq, it is important to account for the differences in sequencing depth and the other biases that may affect the expression levels.



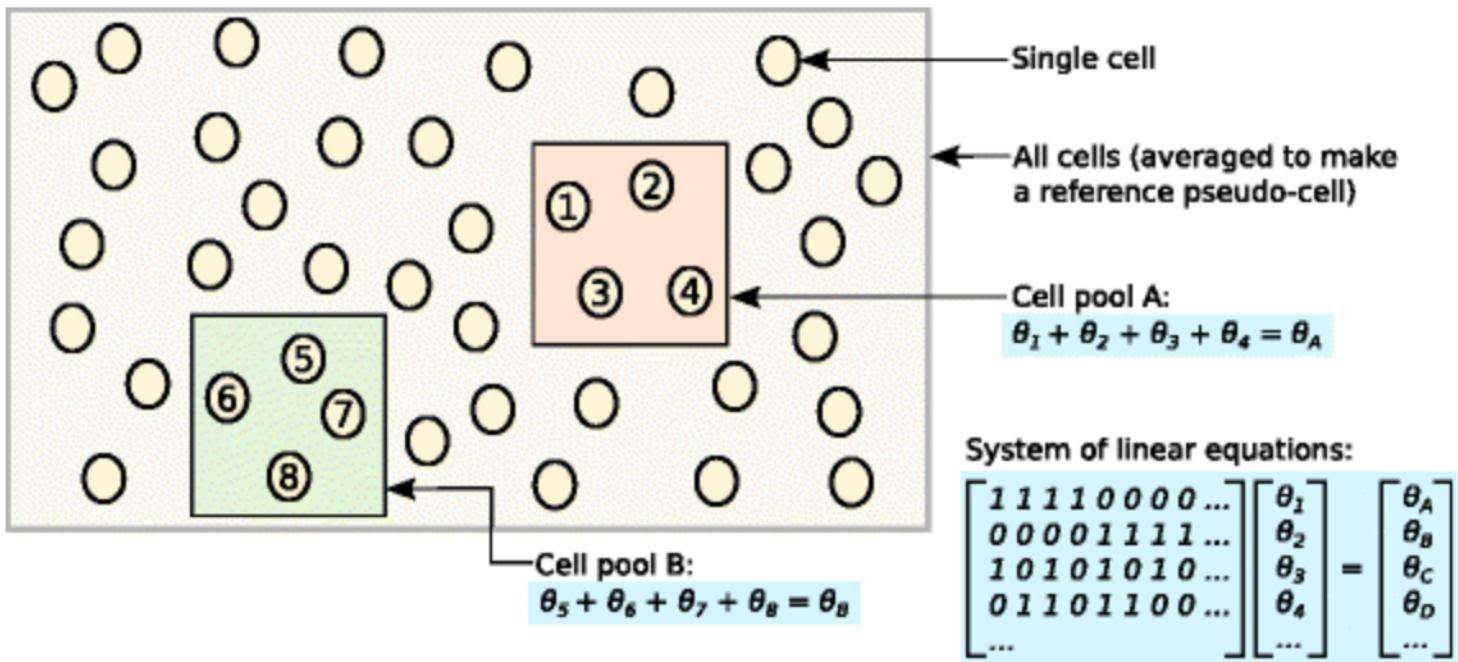
Vallejos et al (2017). *Nature Methods*

Bulk RNA-seq normalization do not always work



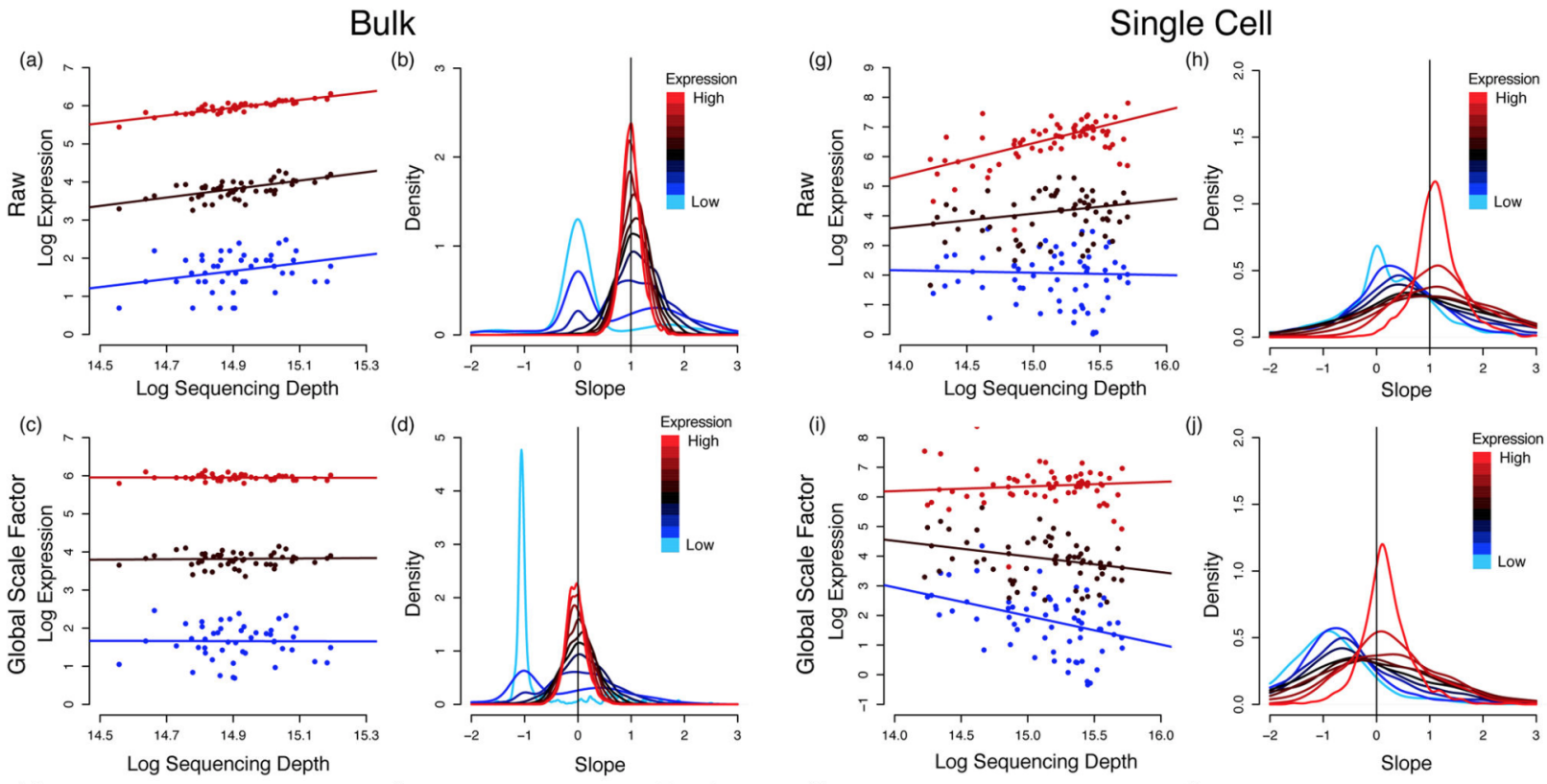
Vallejos et al (2017). *Nature Methods*

Pooling across cells helps



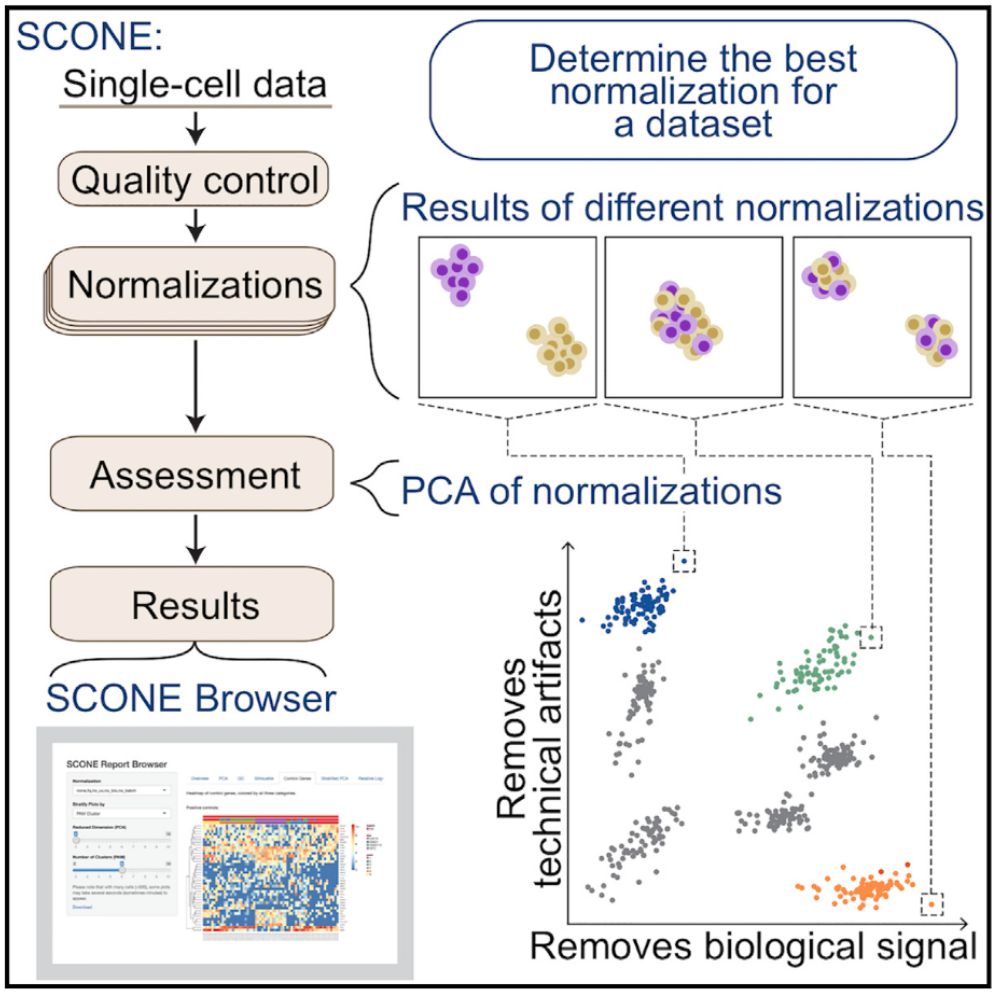
scran Bioconductor package

Non-linear normalization



SCnorm Bioconductor package

Ranking normalization by performance



scone Bioconductor package

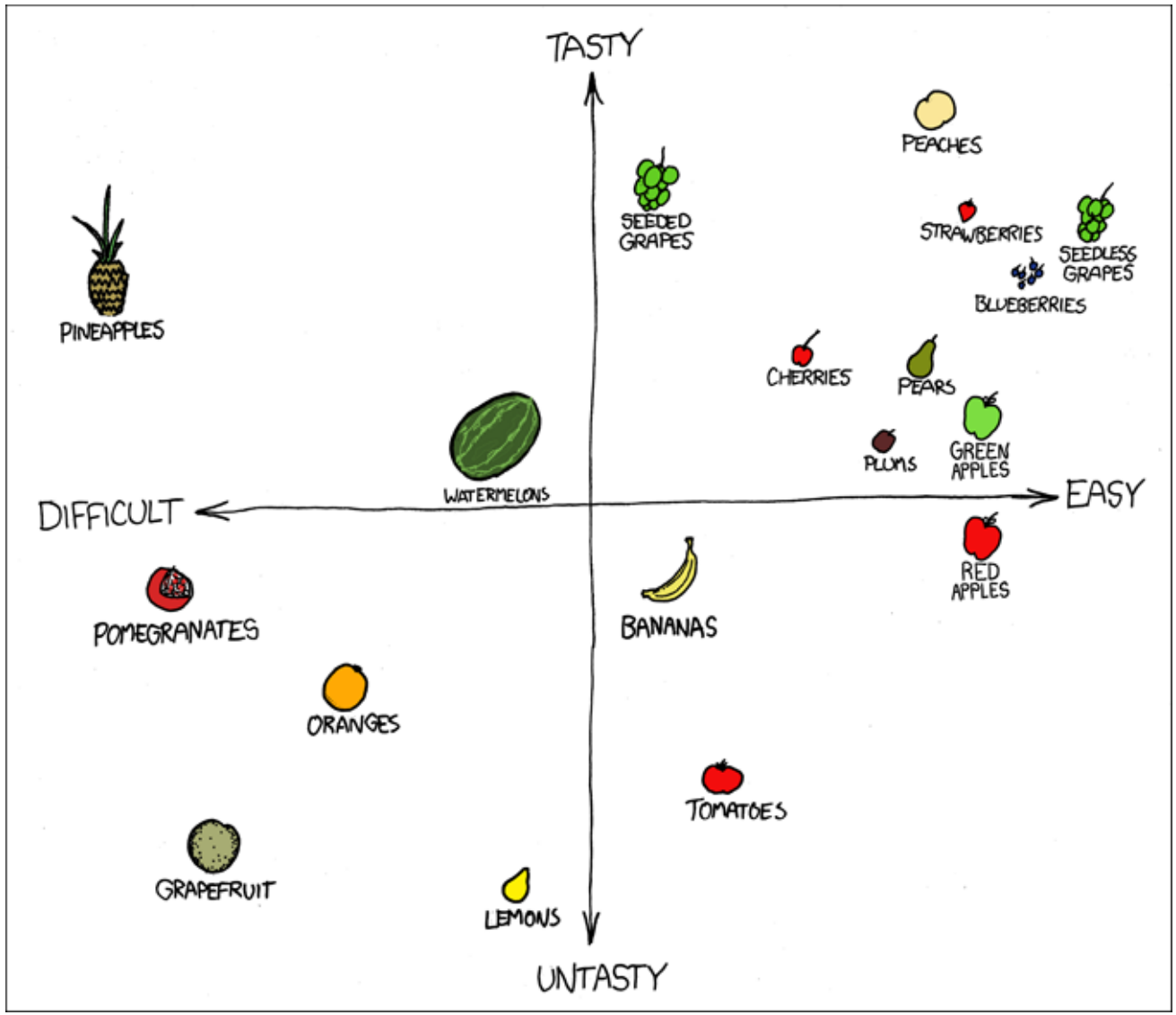
Model-based approach

An alternative, is to include normalization as a parameter of the statistical model.

This has the advantage of propagating the uncertainty in the estimation of the scaling factors.

This is the approach of the *BASiCS* and *zinbwave* Bioconductor packages.

Dimensionality reduction


TASTY
UNTASTY
DIFFICULT
EASY
PINEAPPLES
SEEDED GRAPES
PEACHES
STRAWBERRIES
SEEDLESS GRAPES
BLUEBERRIES
CHERRIES
PEARS
PLUMS
GREEN APPLES
WATERMELONS
RED APPLES
POMEGRANATES
BANANAS
ORANGES
TOMATOES
GRAPEFRUIT
LEMONS

Dimensionality reduction

Dimensionality reduction is useful for two related goals

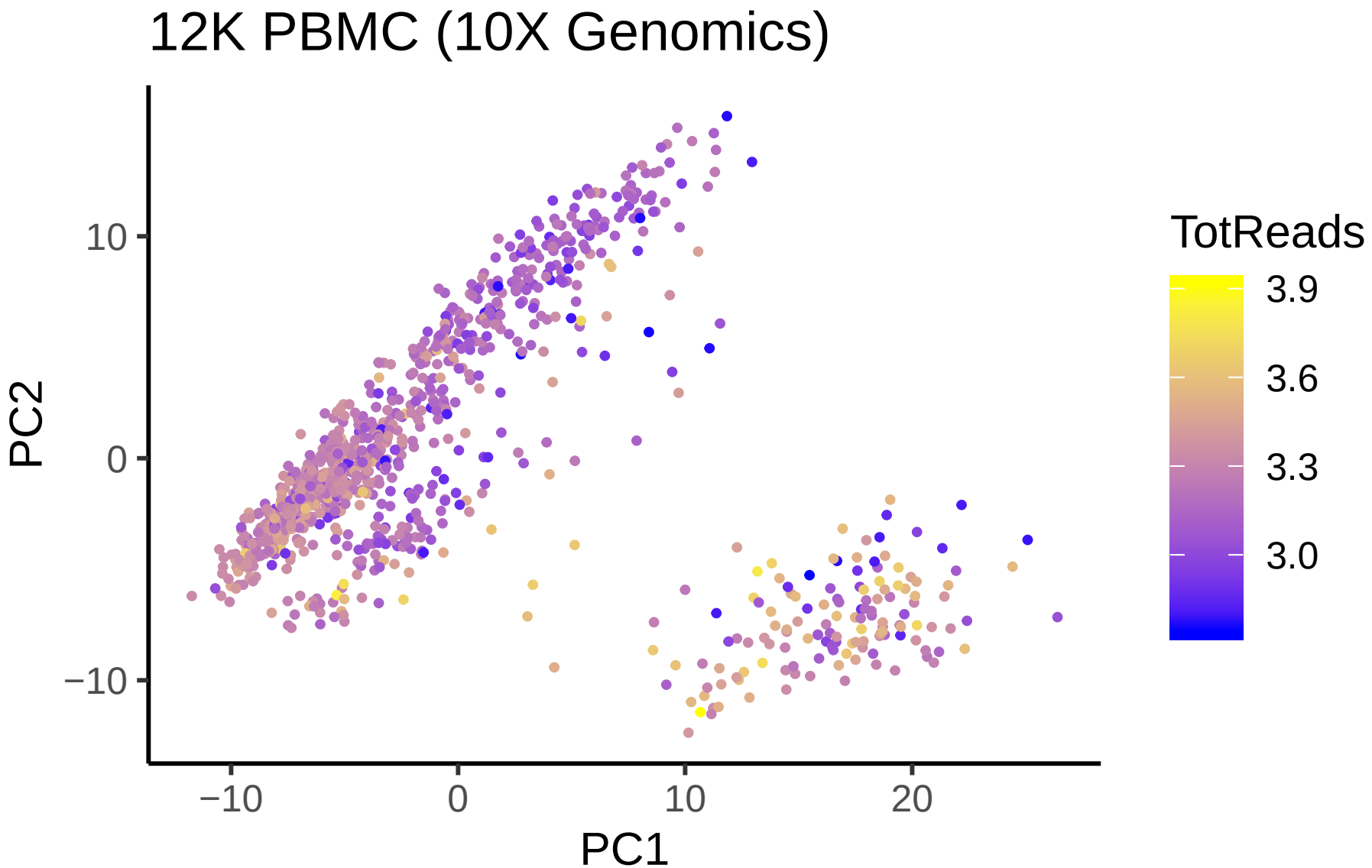
1. *Visualize* high dimensional data (usually in two dimensions)
   - PCA
   - t-SNE
   - UMAP
2. *Infer* low-rank signal from high dimensional data (2 – 50 dimensions)
   - PCA as a factor analysis model
   - ZIFA
   - ZINB-WaVE

Principal Component Analysis (PCA)

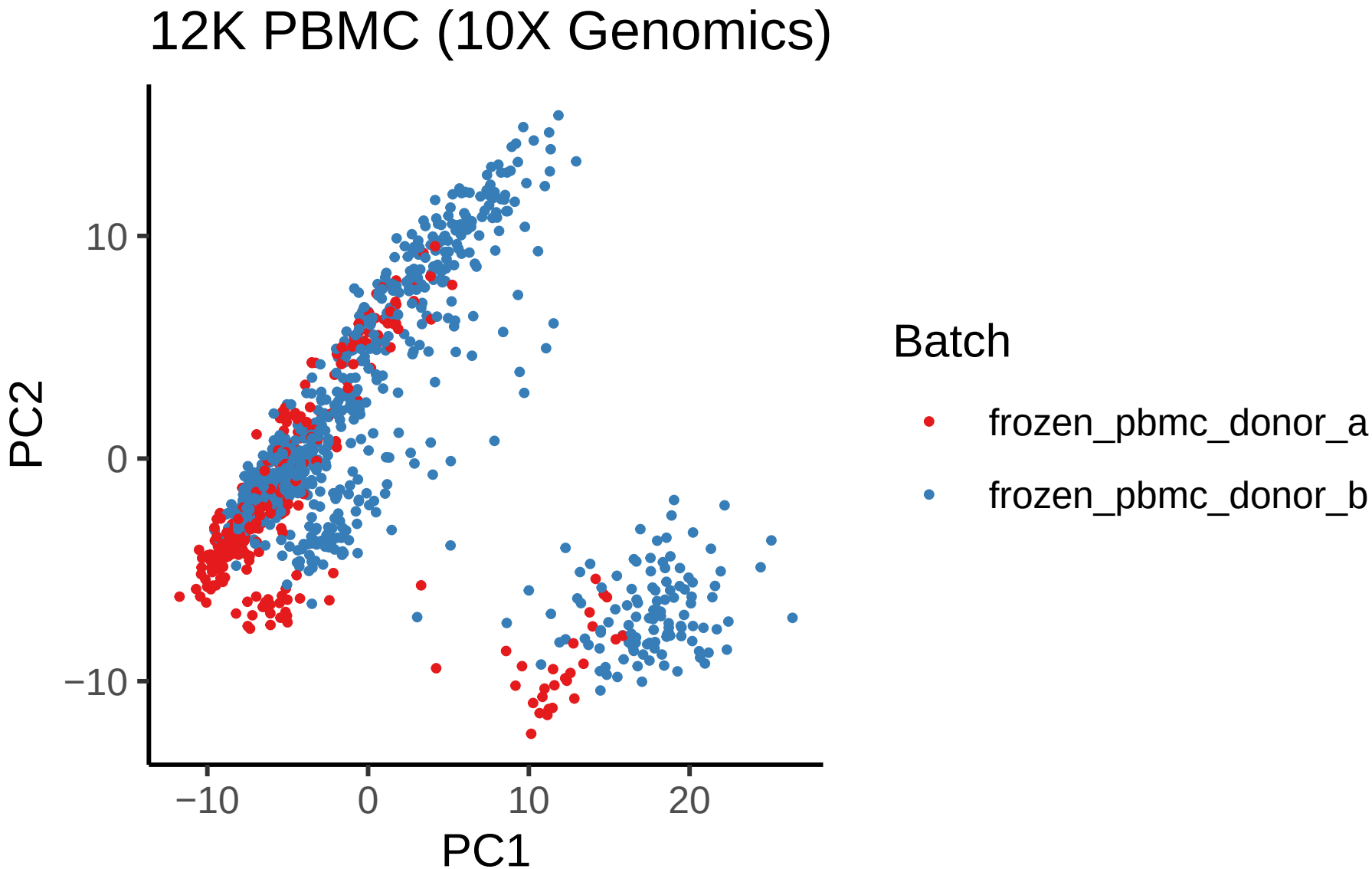
Principal component analysis (PCA) is a dimensionality reduction technique that provides a parsimonious summarization of the data by replacing the original variables by fewer *linear combinations* of these variables, that are *orthogonal* and have successively *maximal variance*.

Such linear combinations seek to "separate out" the observations, while loosing as little information as possible.

Sample quality affects PCA



Batch effects!



Desired properties

- Accounting for zero inflation (dropouts), over-dispersion, and the count nature of the data.
- General and flexible.
- Extract low-dimensional signal from the data.
- Adjust for complex, non-linear effects (batch effects)
- Scalable

Desired properties

- Accounting for zero inflation (dropouts), over-dispersion, and the count nature of the data.
- General and flexible.
- Extract low-dimensional signal from the data.
- Adjust for complex, non-linear effects (batch effects)
- Scalable

Desired properties

- Accounting for zero inflation (dropouts), over-dispersion, and the count nature of the data.
- General and flexible.
- Extract low-dimensional signal from the data.
- Adjust for complex, non-linear effects (batch effects)
- Scalable

Desired properties

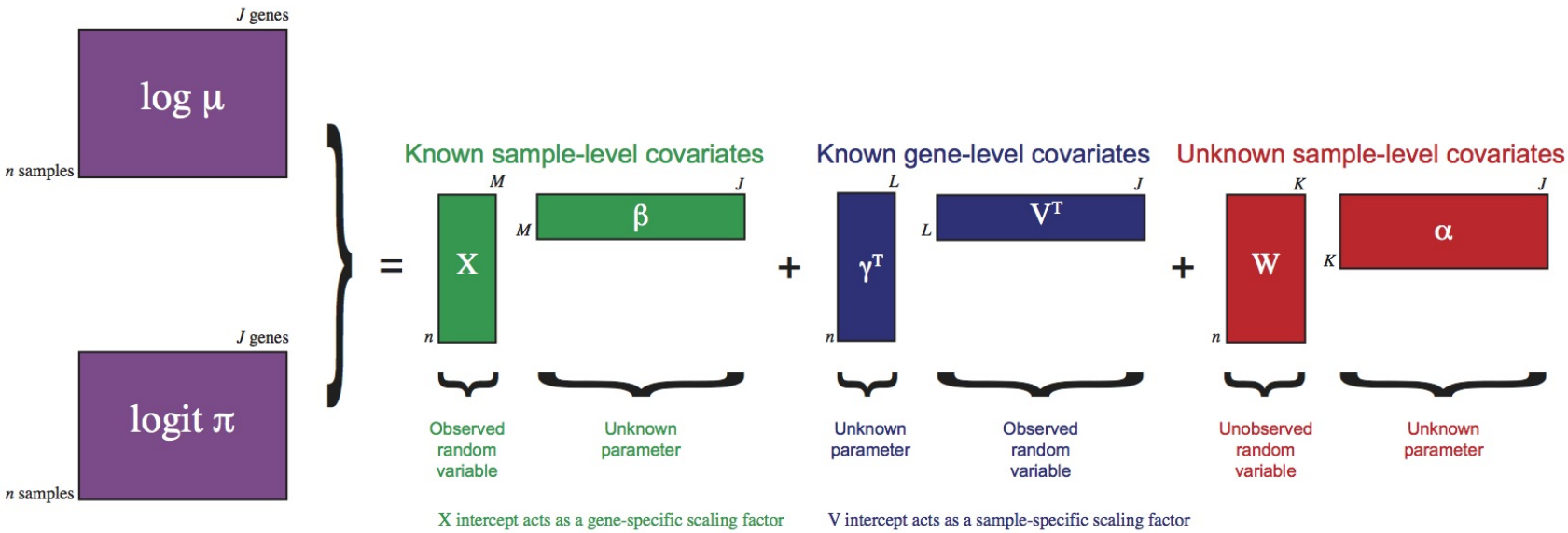
- Accounting for zero inflation (dropouts), over-dispersion, and the count nature of the data.
- General and flexible.
- Extract low-dimensional signal from the data.
- Adjust for complex, non-linear effects (batch effects)
- Scalable

Desired properties

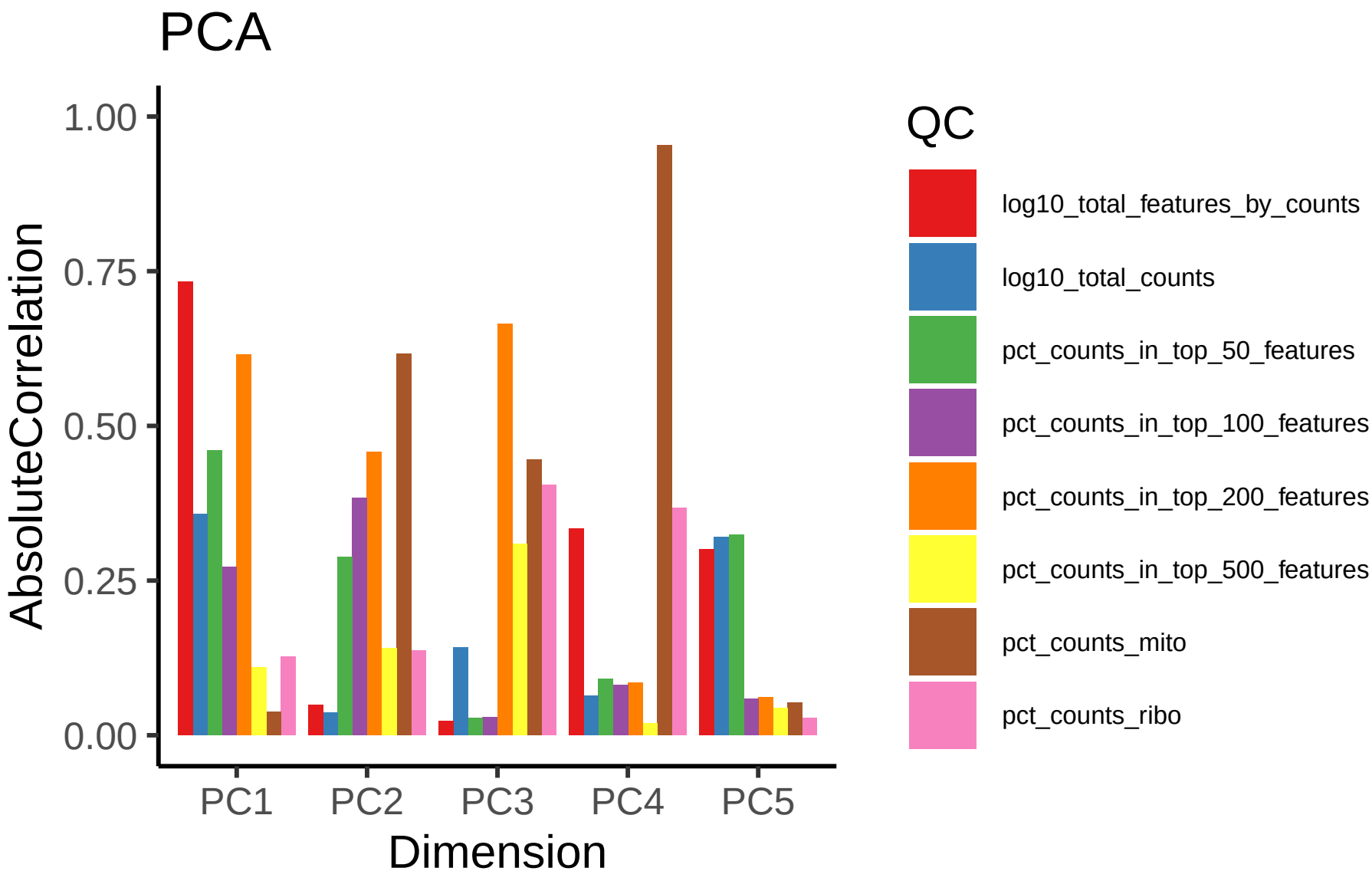
- Accounting for zero inflation (dropouts), over-dispersion, and the count nature of the data.
- General and flexible.
- Extract low-dimensional signal from the data.
- Adjust for complex, non-linear effects (batch effects)
- Scalable

The ZINB-WaVE model

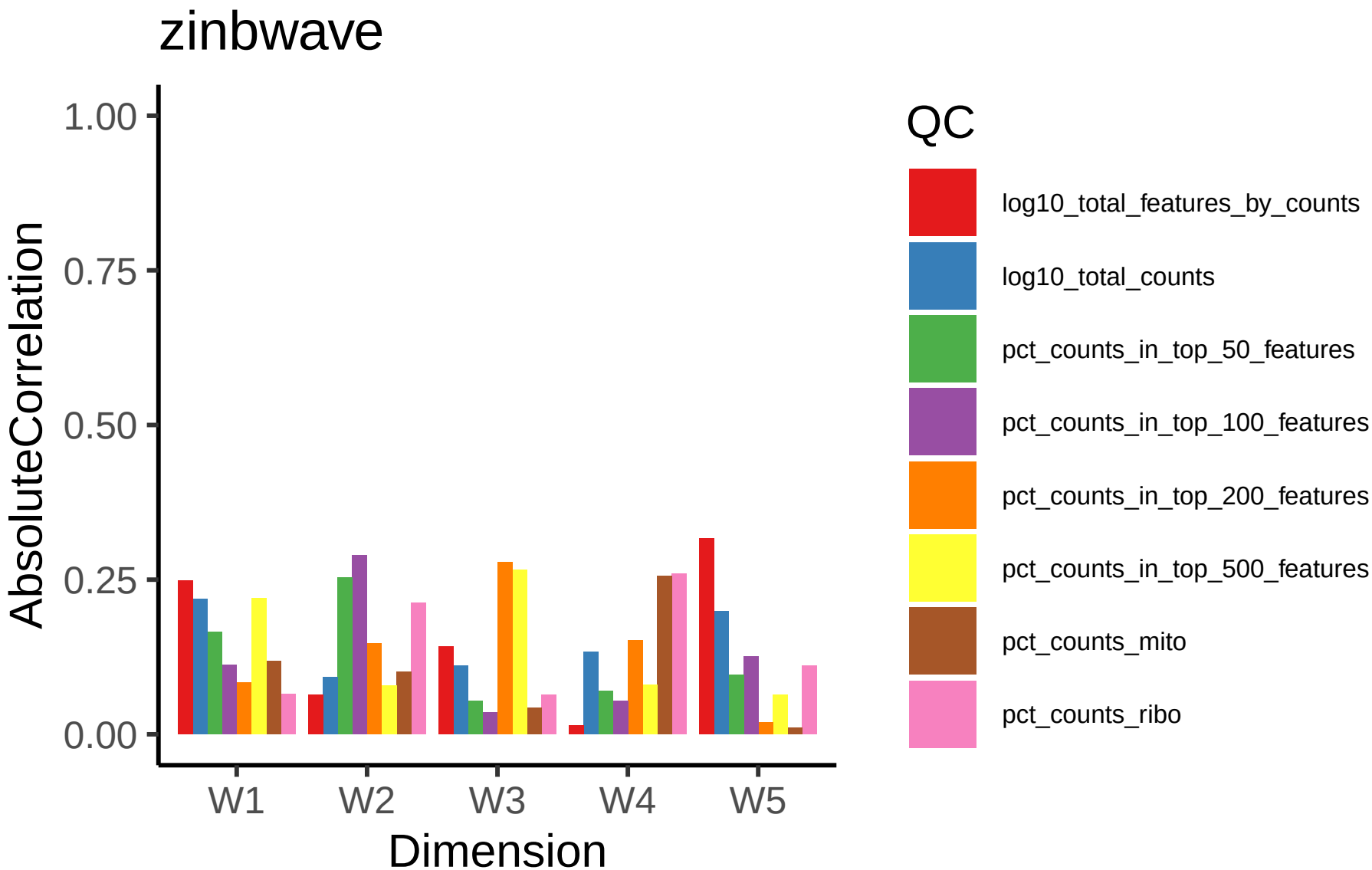
Given n samples and J genes, let Y_{ij} denote the count of gene j (for $j = 1, \ldots, J$) for sample i (for $i = 1, \ldots, n$).



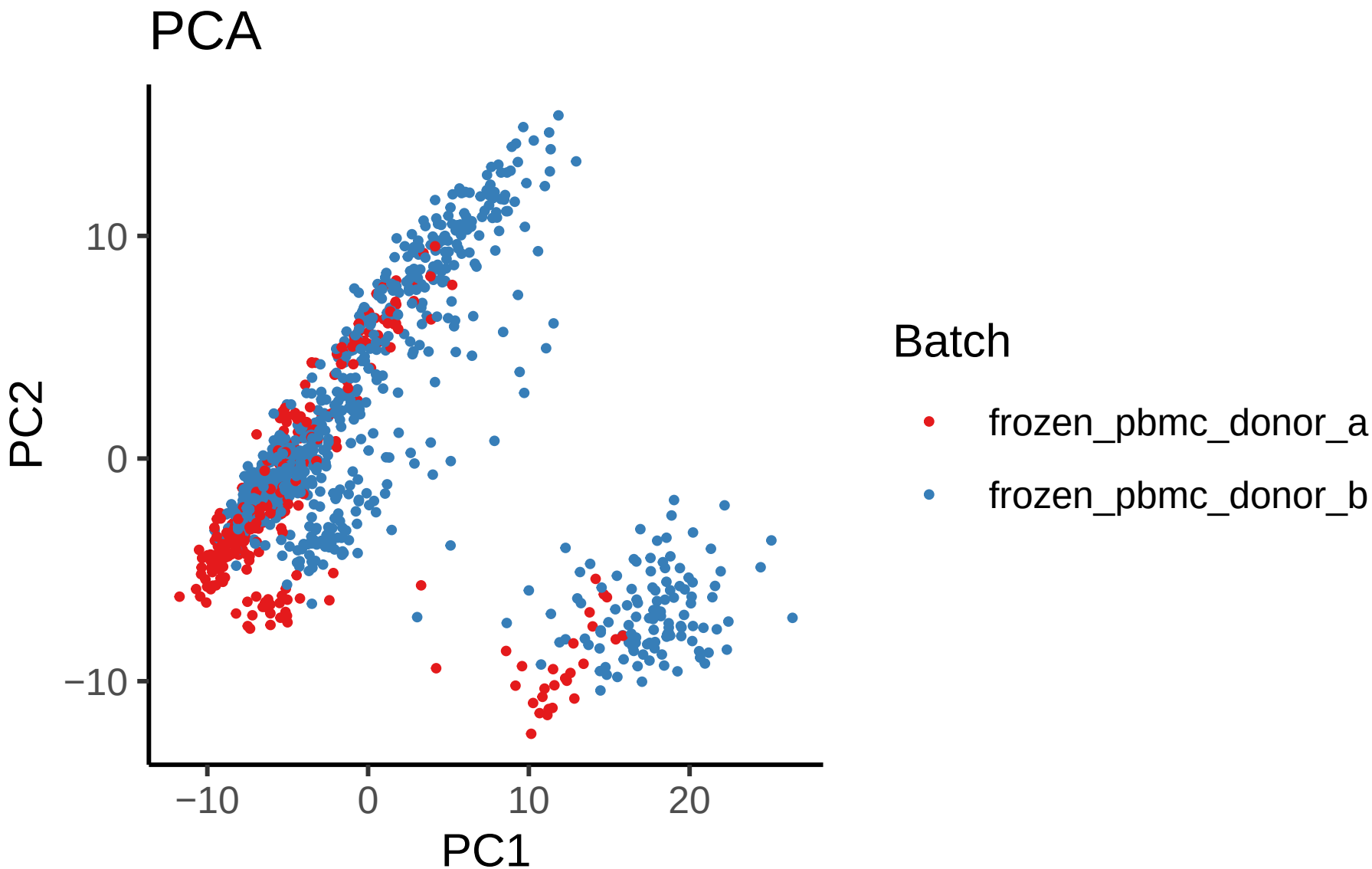
Sample quality affects PCA



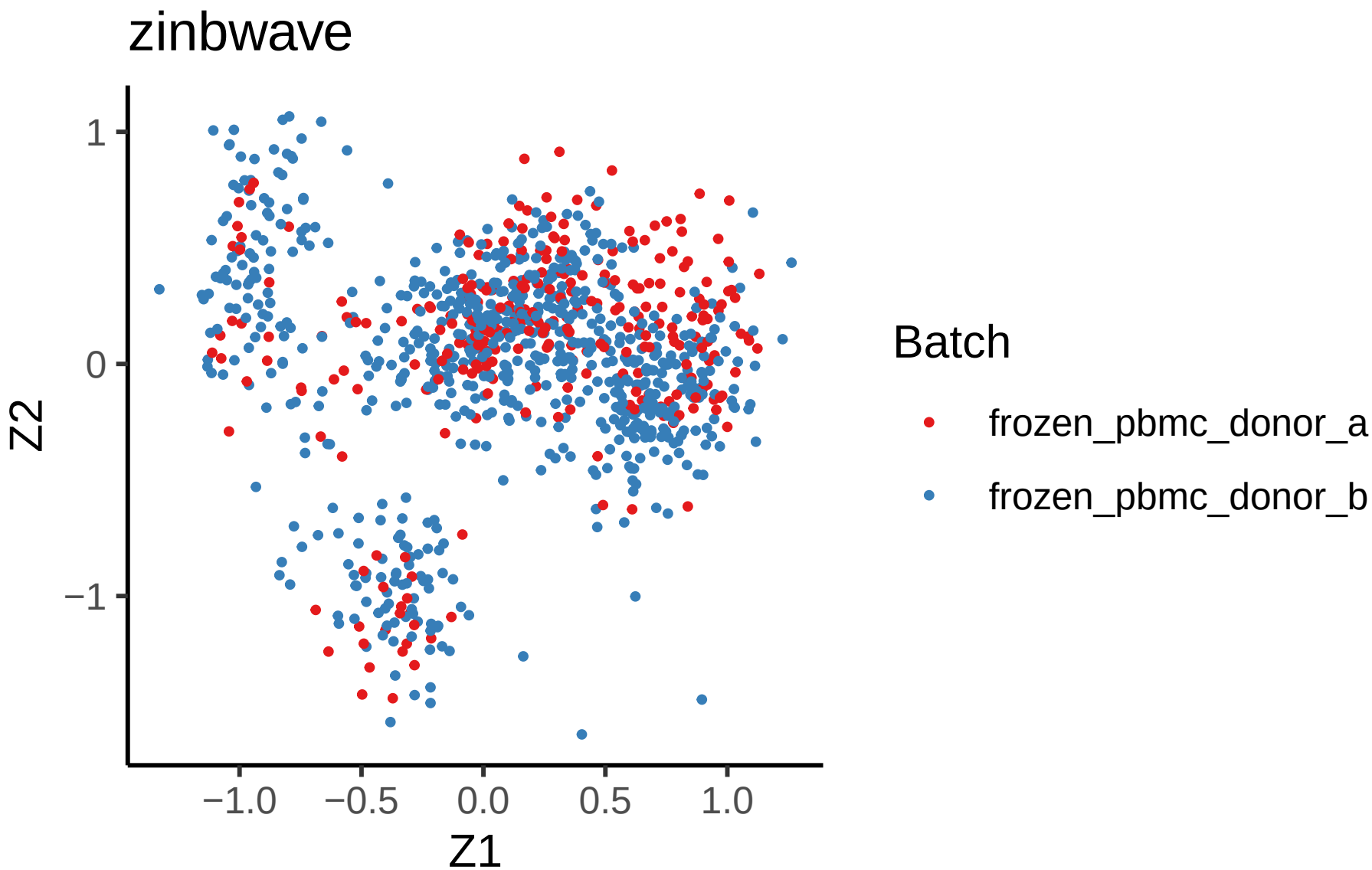
ZINB-WaVE adjusts for quality



Evident batch effects



ZINB-WaVE adjusts for batch effects



ZINB-WaVE adjusts for batch effects

```
library(zinbwave)

sce <- zinbwave(sce, X = "~batch", K = 10)
```

GLM-PCA

In droplet-based data, it might be safe to ignore zero inflation.

We can of course use a simpler "NB-WAVE" model.

Alternatively, we can exploit the fact that the negative binomial distribution (with known dispersion) belongs to the exponential family.

The GLM-PCA method is a generalization of PCA for the exponential family.

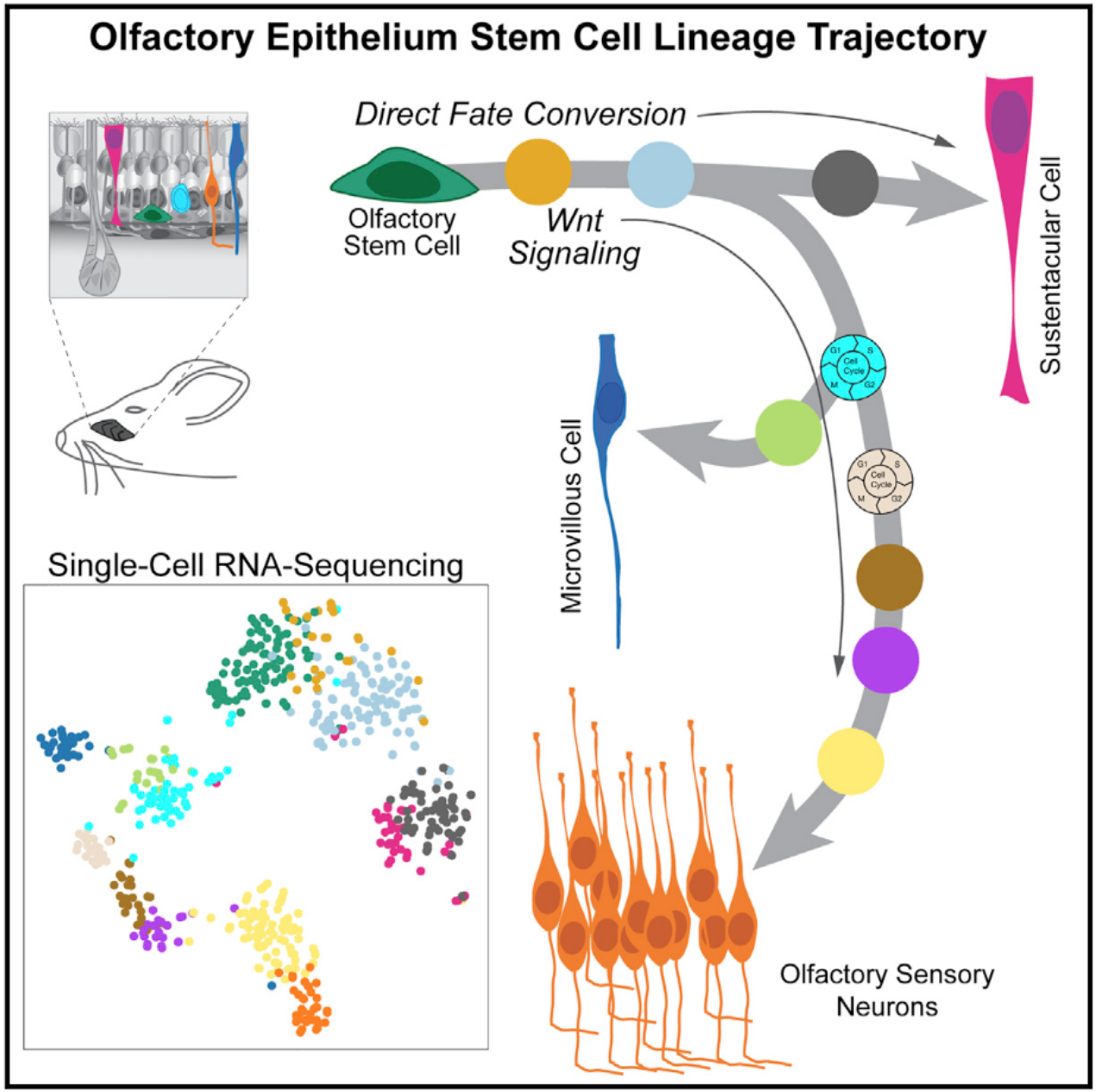
Townes et al. (2019) propose a fast approximation to GLM-PCA based on deviance residuals that is much faster than ZINB-WAVE and gives comparable results.

Approximate PCA

Even regular PCA is not scalable enough to very large datasets (millions of cells) and approximations are needed.

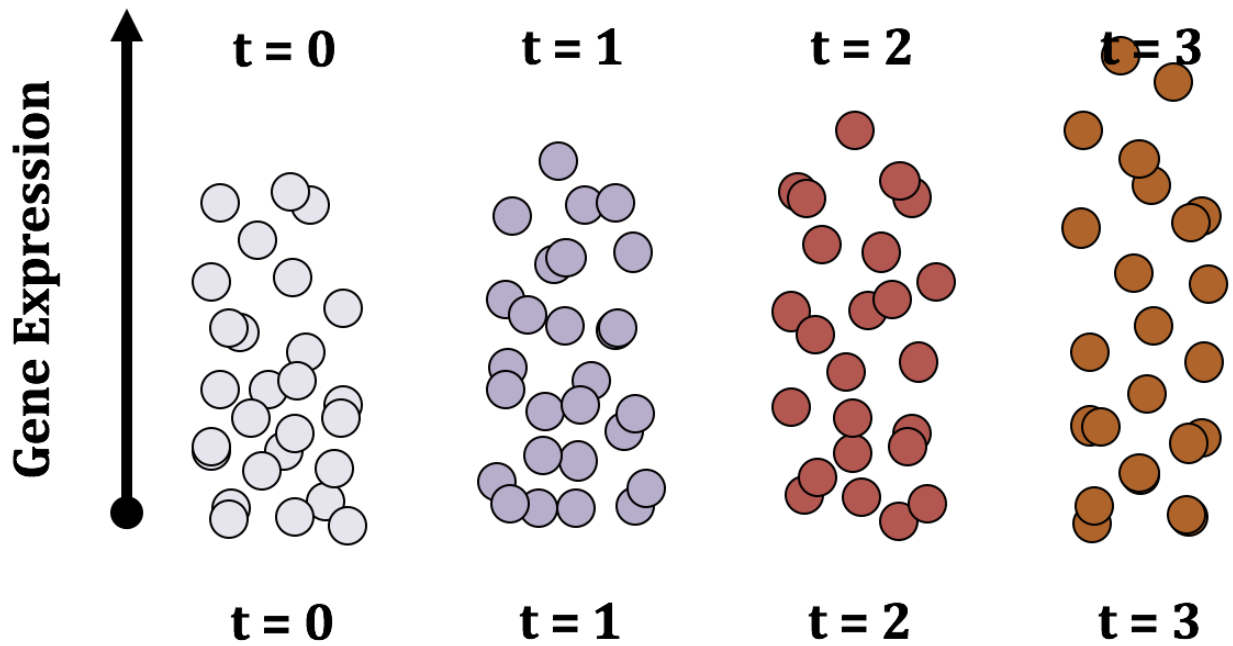
The *BiocSingular* package provides implementations of the random PCA algorithm and the implicitly restarted Lanczos bidiagonalization algorithm (IRLBA).

Lineage Inference


Olfactory Epithelium Stem Cell Lineage Trajectory
Direct Fate Conversion
Olfactory
Stem Cell
Wnt
Signaling
Sustentacular Cell
Microvillous Cell
Single-Cell RNA-Sequencing
Olfactory Sensory
Neurons

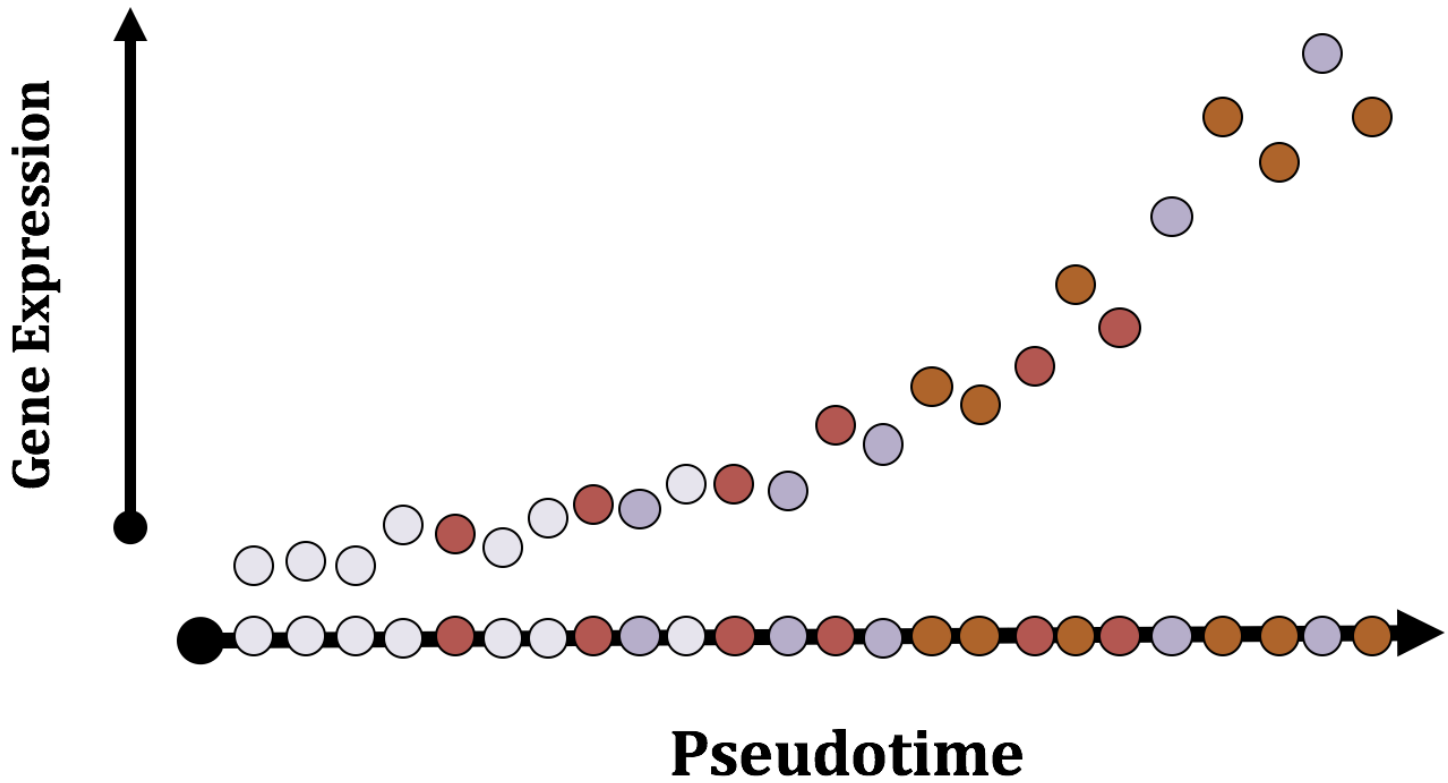
Aim:

High-resolution view of transcriptional changes during continuous developmental processes.



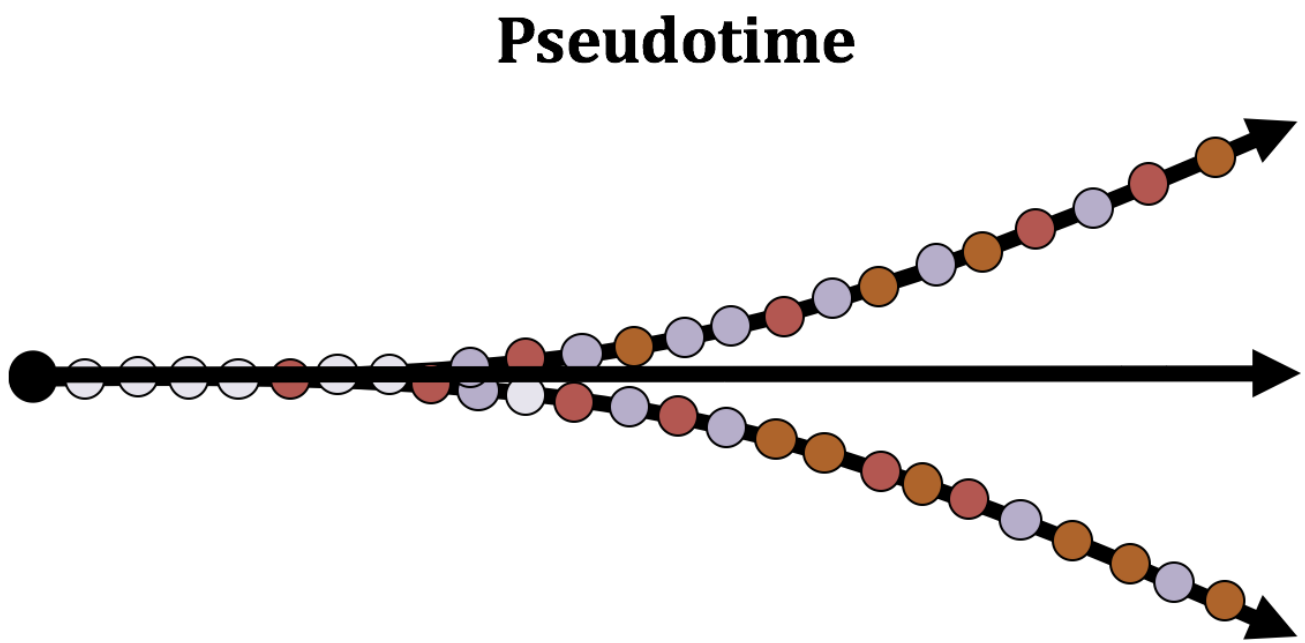
Aim:

High-resolution view of transcriptional changes during continuous developmental processes.



Aim:

High-resolution view of transcriptional changes during continuous developmental processes.



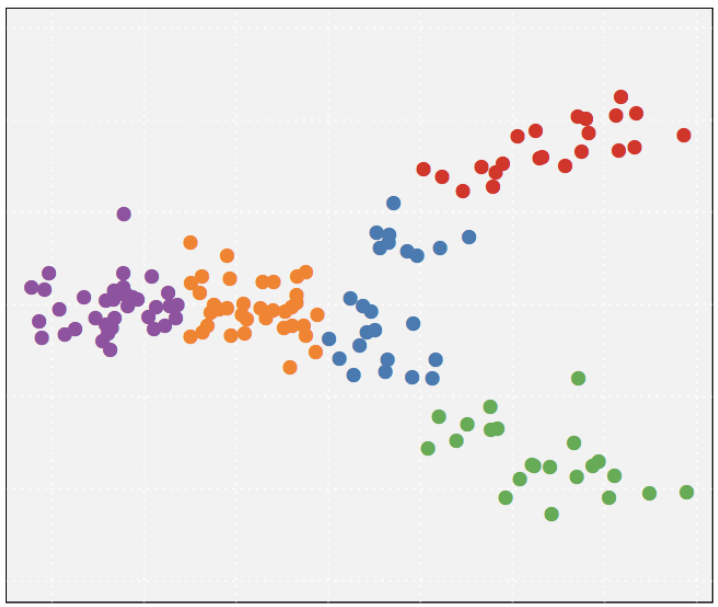
The slingshot algorithm

We start from a proper representation of the cells in some space defined by their gene expression (usually after dimensionality reduction).

We have identified a set of K clusters.

1. Identification of lineages
   - treat clusters as nodes in a graph
   - draw a minimum spanning tree (MST) between the nodes
   - lineages are ordered sets of clusters
   - semi-supervised: set the starting cluster (root of the tree) and optionally a set of known end points (leaves)
2. Draw a "smooth" path through the lineages
   - use of *principal curves* (Hastie and Stuetzle, 1989)
   - shrink curves together in shared paths (simultaneous principal curves)
   - project each cell onto the principal curve(s) to infer pseudotime

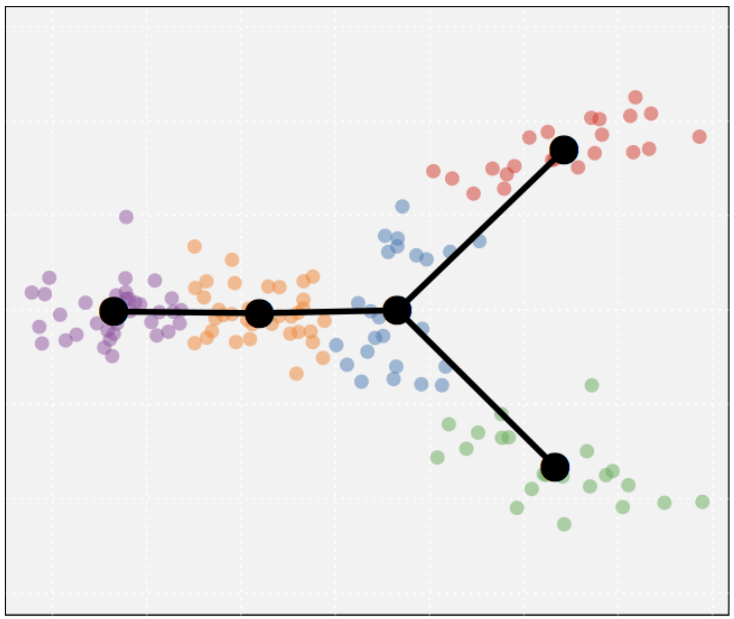
Step 0: Input



Clustered data in low-dimensional space.

We consider dimensionality reduction and clustering to be separate problems, but generally prefer PCA and RSEC.

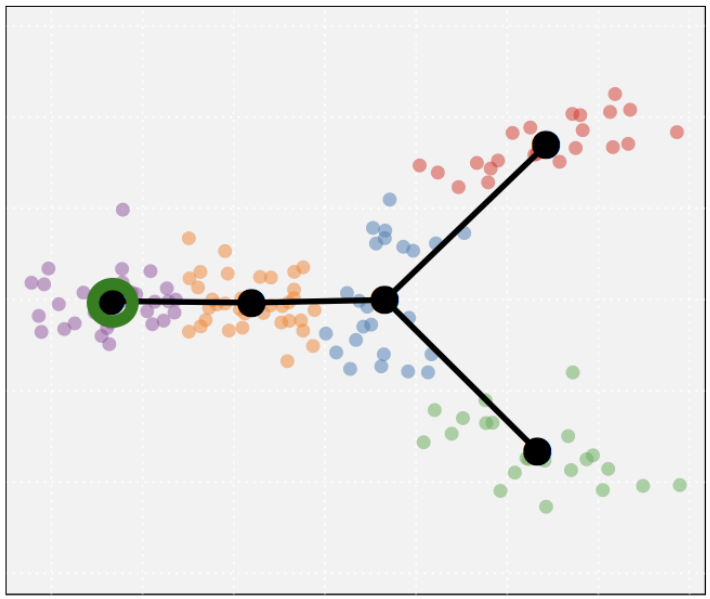
Step 1: Cluster MST



Construct minimum spanning tree (MST) on clusters.

The nodes are clusters, not cluster centers. Requires a distance metric.

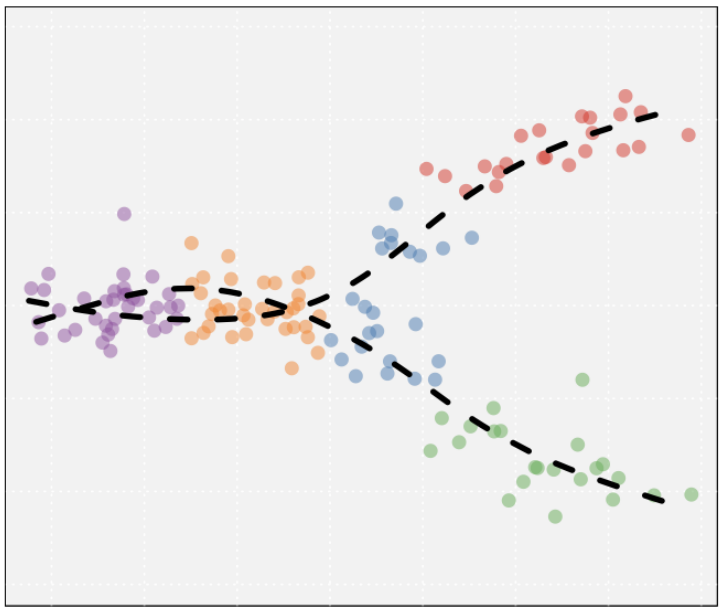
Step 1: Cluster MST



Construct minimum spanning tree (MST) on clusters.

Select starting cluster based on marker genes or parsimony.

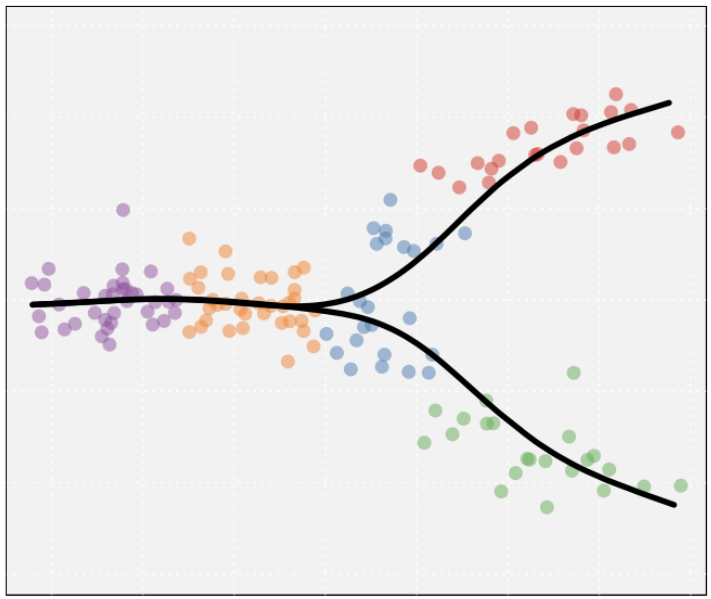
Step 1.5: Principal Curves



Highly stable, nonlinear generalization of principal components. Fits a curve to the "middle" of the data.

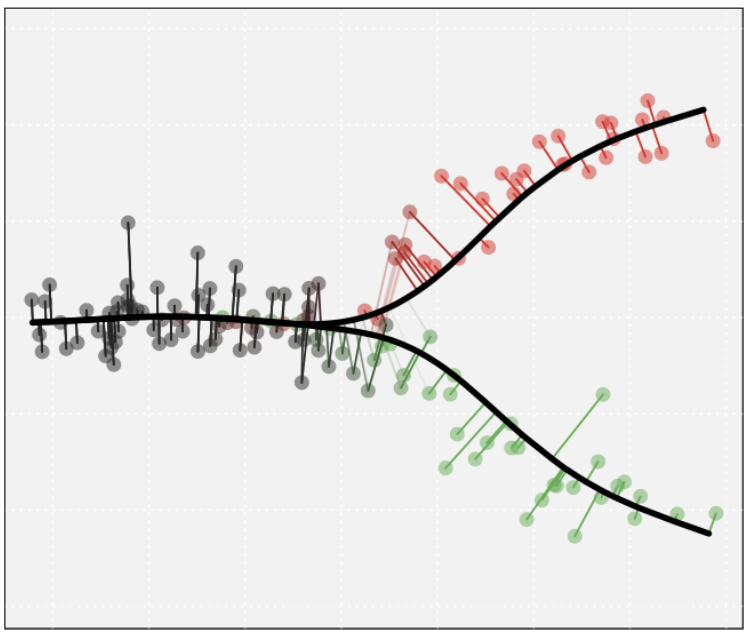
Inconsistent, fails to reflect underlying biology (i.e. branching).

Step 2: Simultaneous Principal Curves



Still highly stable and nonlinear. Extends the concept of principal curves to multiple, branching curves with common origin (smooth tree structures).

Step 2: Simultaneous Principal Curves



Project cells onto curves to obtain pseudotime values.

Like linear principal components, the curves seek to minimize squared projection distance (subject to some constraints).

Shape-sensitive distance

Constructing an MST involves specifying a distance measure between nodes.

A Mahalanobis-like distance, i.e., a covariance-scaled Euclidean distance, that accounts for cluster shape, works well in practice.

$$d^2(\mathcal{C}_i, \mathcal{C}_j) \equiv (\bar{X}_i - \bar{X}_j)^T (S_i + S_j)^{-1} (\bar{X}_i - \bar{X}_j),$$

Biological meaningful supervision

Slingshot allows two forms of supervision during lineage identification:

- initial state (root)
- terminal states (leaves)

Like other methods, Slingshot requires the user to identify the *initial cluster* or *root node*.

Slingshot optionally allows the specification of *terminal cell states*, imposing a *local constraint* on the MST algorithm.

Principal Curves algorithm (Hastie and Stuetzle, 1989)

Iteratively follow these steps:

1. Project all data points onto the curve and calculate the arc length from the beginning of the curve to each point's projection. Setting the lowest value to zero, this produces pseudotimes.
2. For each dimension j, use the cells' pseudotimes to predict their coordinates, typically with a smoothing spline.
    - This produces a set of J' functions which collectively map pseudotime values defining a smooth curve in J' dimensions.
3. Repeat this process until convergence, using the sum of squared distances between cells' actual coordinates and their projections on the curves to determine convergence.

Simultaneous Principal Curves

To allow for multiple lineages, we modify the principal curves algorithms in two ways:

- We incorporate cell weights to allow cells to contribute differently to different lineages.
- We add a *shrinkage* procedure to ensure smooth branching events.

The shrinkage is performed by first recursively constructing an average curve for each branching event, then recursively shrinking the branching lineage curves toward this average.

Shrinkage

We construct *non-increasing* curve-specific weights, with $w_m(0) = 1$ (maximum shrinkage) - diverging curves always share the same initial point

Shrink the diverging curves toward the average curve:

$$\mathbf{c}_m^{\text{new}}(t) \equiv w_m(t)\mathbf{c}_{\text{avg}}(t) + (1 - w_m(t))\mathbf{c}_m(t).$$

The slingshot package

```
library(slingshot)
ce <- slingshot(ce, reducedDim = "MDS",
                start.clus = "c1")
```

Thank you!

Email: `risso.davide@gmail.com`

Twitter: `@risso1893`

Github: `github.com/drisso`