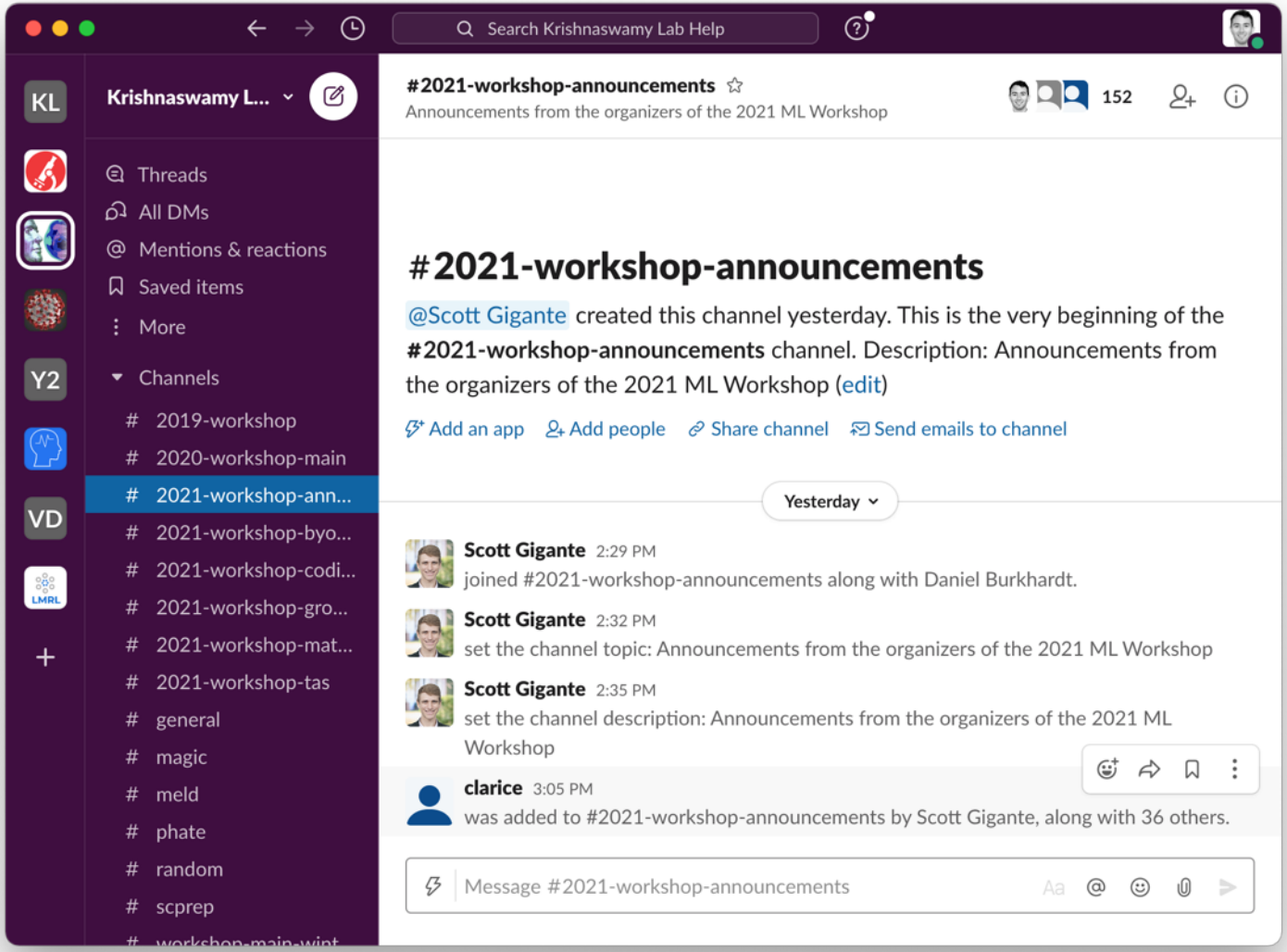
# What is your favorite model organism?

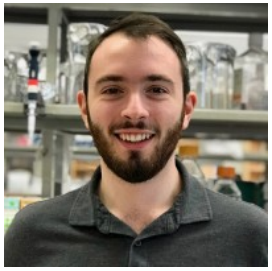
# Machine Learning for Single Cell Analysis

## Course introduction




https://krishnaswamylab.org/get-help


Smita Krishnaswamy


Daniel Burkhardt


Scott Gigante


Kofi Ansong


Andrew Benz


Egbert Castro


Pranik Chainani


Joanna Chen


Annie Gao

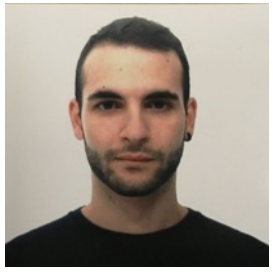

Michał Gerasimiuk

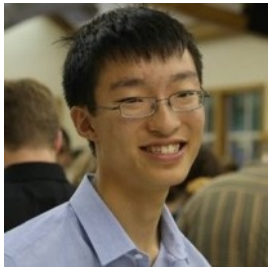

Wes Lewis


Francesc Lopez


Sameet Mehta


Sasha Safonova


Giacomo Scanavini

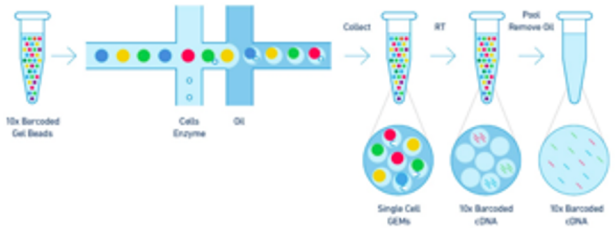

Alexander Tong


Aarthi Venkat

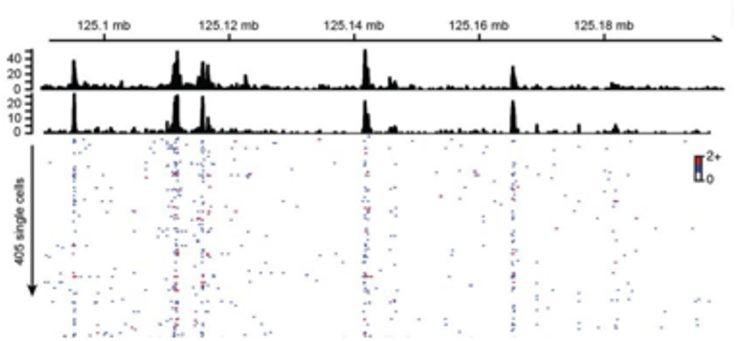

Max Yuan


Jeffrey Zhou

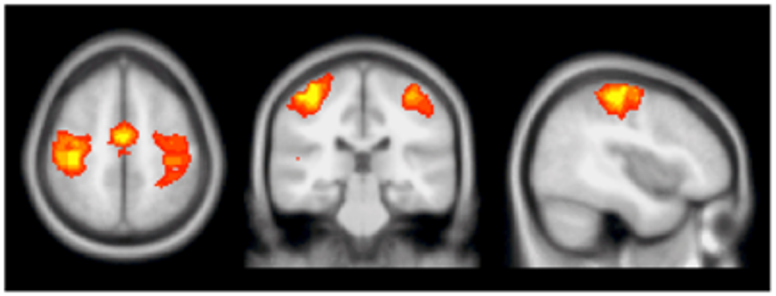
# Big biomedical data




ScRNA-seq




ScATAC-seq

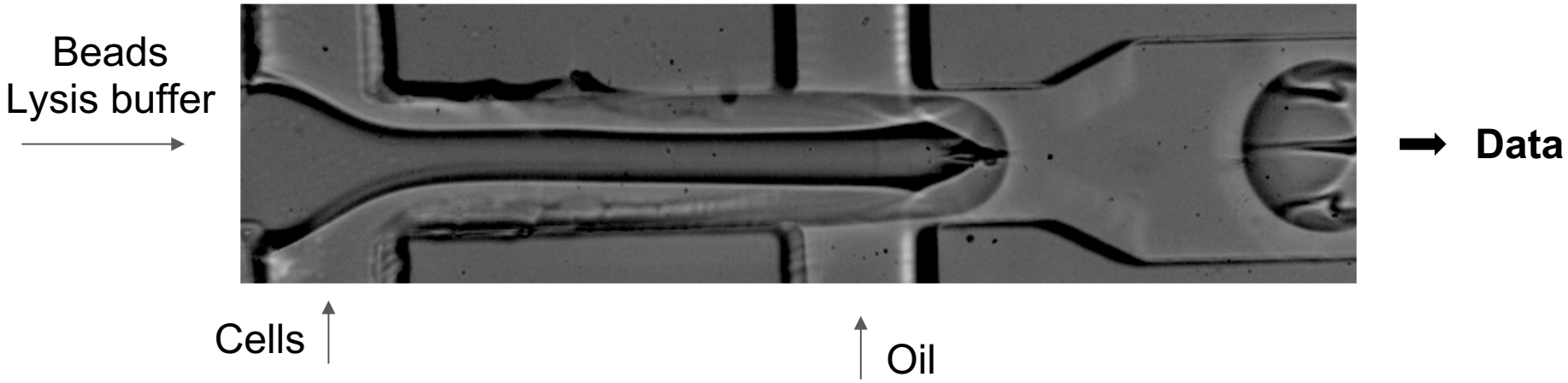


FMRI

Patient Data

Big = Any dataset with many many observations

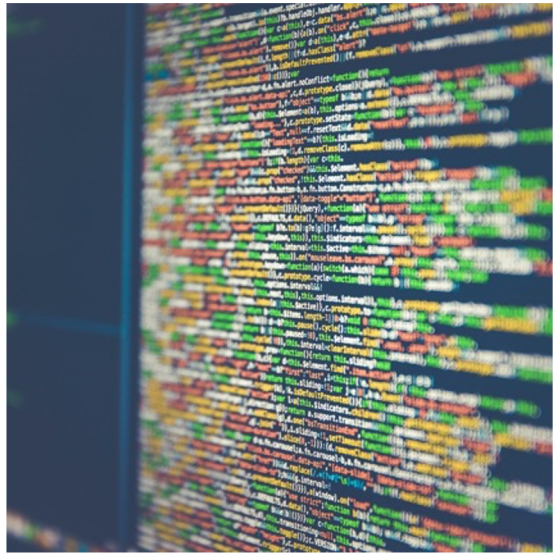
# The single cell revolution



# The single cell revolution

Interesting Biological Experiments

Computation



High impact paper

LETTER

https://doi.org/10.1038/s41586-018-0414-6

## RNA velocity of single cells

Gioele La Manno[1,2], Ruslan Soldatov[1], Amit Zeisel[1,3], Emelie Braun[1,3], Hannah Hochgerner[3,4], Viktor Petukhov[3,4], Katja Lidschreiber[5], Maria E. Kastriti[6], Peter Lönnerberg[1,2], Alessandro Furlan[7], Jean Fan[1], Lars E. Borm[3,4], Zehua Liu[1], David van Bruggen[1], Jimin Guo[1], Xiaoling He[1], Roger Barker[1], Erik Sundström[8], Gonçalo Castelo-Branco[1], Patrick Cramer[5,9], Igor Adameyko[6], Sten Linnarsson[1,2]* & Peter V. Kharchenko[3,10]*

**RNA abundance is a powerful indicator of the state of individual cells. Single-cell RNA sequencing can reveal RNA abundance with high quantitative accuracy, sensitivity and throughput[1]. However, this approach captures only a static snapshot at a point in time, posing a challenge for the analysis of time-resolved phenomena such as embryogenesis or tissue regeneration. Here we show that RNA velocity—the time derivative of the gene expression state—can be directly estimated by distinguishing between unspliced and spliced mRNAs in common single-cell RNA sequencing protocols. RNA velocity is a high-dimensional vector that predicts the future state of individual cells on a timescale of hours. We validate its accuracy in the neural crest lineage, demonstrate its use on multiple published datasets and technical platforms, reveal the branching lineage tree of the developing mouse hippocampus, and examine the kinetics of transcription in human embryonic brain. We expect RNA velocity to greatly aid the analysis of developmental lineages and cellular dynamics, particularly in humans.**

During development, differentiation occurs on a timescale of hours to days, which is comparable to the typical half-life of mRNA. The relative abundance of nascent (unspliced) and mature (spliced) mRNA can be exploited to estimate the rates of gene splicing and degradation, without the need for metabolic labelling, as previously shown in bulk[2–4]. We reasoned that similar signals may be detectable in single-cell RNA sequencing (RNA-seq) data, and could reveal the rate and direction of change of the entire transcriptome during dynamic processes.

All common single-cell RNA-seq protocols rely on oligo-dT primers to enrich for polyadenylated mRNA molecules. Nevertheless, examining single-cell RNA-seq datasets based on the SMART-seq2, STRT-C1, inDrop and 10x Genomics Chromium protocols[5–8], we found that 15–25% of reads contained unspliced intronic sequences (Fig. 1a), in agreement with previous observations in bulk[9] (14.4%) and single-cell[7] (~20%) RNA-seq. Most such reads originated from secondary priming positions within the intronic regions (Extended Data Fig. 1). In 10x Genomics Chromium libraries, we also found abundant discordant priming from the more commonly occurring intronic-polyT sequences (Extended Data Fig. 1), which may have been generated during PCR amplification by priming on the first-strand cDNA. The substantial number of intronic molecules and their correlation with the exonic counts suggest that these molecules represent unspliced precursor mRNAs. This was confirmed by metabolic labelling of newly transcribed RNA[9] followed by RNA sequencing using oligo-dT-primed single-cell-tagged reverse transcription (STRT)[10] (Extended Data Fig. 2). 83% of all genes displayed expression time courses consistent with simple first-order kinetics, as expected if unspliced reads represent nascent mRNA.

To quantify the time-dependent relationship between the abundance of precursor and mature mRNA, we assumed a simple model for transcriptional dynamics[2], in which the first time derivative of the spliced mRNA abundance (RNA velocity) is determined by the balance between production of spliced mRNA from unspliced mRNA, and the mRNA degradation (Fig. 1b and Supplementary Note 1). Under such a model, steady states are approached asymptotically when the rate of transcription $\alpha$ is constant, with the steady-state abundances of spliced ($s$) and unspliced ($u$) molecules determined by $\alpha$, and constrained to a fixed-slope relationship where $u = \gamma s$ (Supplementary Note 2 Section 1). The equilibrium slope $\gamma$ combines degradation and splicing rates, capturing gene-specific regulatory properties, the ratio of intronic and exonic lengths, and the number of internal priming sites. Using a recently published compendium of mouse tissues[11], we found that the steady-state behaviour of most genes across a wide range of cell types was consistent with a single fixed slope $\gamma$ (Extended Data Fig. 3a–c). However, 11% of genes showed distinct slopes in different subsets of tissues (Extended Data Fig. 3d, e), suggesting tissue-specific alternative splicing (Extended Data Fig. 3f) or degradation rates.

During a dynamic process, an increase in the transcription rate $\alpha$ results in a rapid increase in unspliced mRNA, followed by a subsequent increase in spliced mRNA (Fig. 1c and Supplementary Note 2 Section 1) until a new steady state is reached. Conversely, a drop in the rate of transcription first leads to a rapid drop in unspliced mRNA, followed by a reduction in spliced mRNAs. During induction of gene expression, unspliced mRNAs are present in excess of the expectation based on the equilibrium rate $\gamma$, whereas the opposite is true during repression (Fig. 1d). The balance of unspliced and spliced mRNA abundance is, therefore, an indicator of the future state of mature mRNA abundance, and thus the future state of the cell.

To demonstrate how this simple model can be used to extrapolate the mature mRNA abundance into the future, we examined a time course of bulk RNA-seq measurements of the circadian cycle in the mouse liver[12]. Unspliced mRNA levels at each time point were consistently more similar to the spliced mRNA of the subsequent time (Fig. 1e), and many circadian-associated genes showed the expected excess of unspliced mRNA relative to the slope $\gamma$ during upregulation, and a corresponding deficit during downregulation (Fig. 1f, g). Solving the proposed differential equations for each gene allowed us to extrapolate each measurement throughout the circadian cycle, accurately capturing the expected direction of progression of the circadian cycle (Fig. 1h).

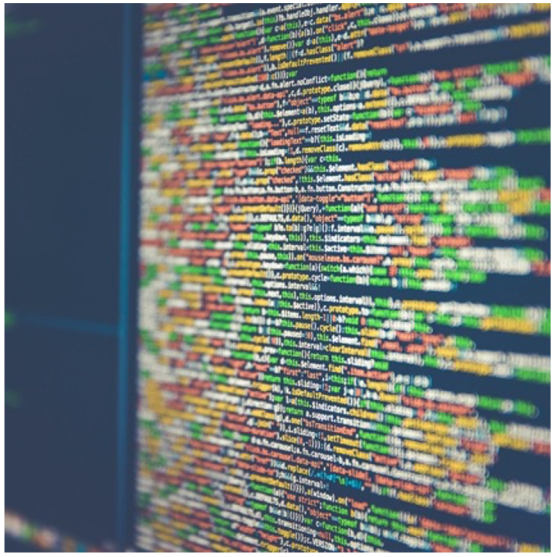
Next, to demonstrate the ability to predict transcriptional dynamics in single-cell measurements, we analysed recently published single-cell mRNA-seq data of mouse chromaffin cells[7], obtained using SMART-seq2[7] (Fig. 2). During development, a substantial proportion of chromaffin cells, which are neuroendocrine cells of the adrenal medulla, arise from Schwann cell precursors, providing a convenient test case in which the direction of differentiation can be validated by lineage tracing. Phase portraits of many genes showed the expected deviations

[1]Laboratory of Molecular Neurobiology, Department of Medical Biochemistry and Biophysics, Karolinska Institutet, Stockholm, Sweden. [2]Science for Life Laboratory, Solna, Sweden. [3]Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA. [4]Department of Applied Mathematics, Peter The Great St. Petersburg Polytechnic University, St. Petersburg, Russia. [5]Department of Biosciences and Nutrition, Karolinska Institutet, Stockholm, Sweden. [6]Department of Physiology and Pharmacology, Karolinska Institutet, Stockholm, Sweden. [7]John van Geest Centre for Brain Repair, Department of Clinical Neurosciences, University of Cambridge, Cambridge, UK. [8]Division of Neuroregeneration, Department of Neurobiology, Care Sciences and Society, Karolinska Institutet, Stockholm, Sweden. [9]Max Planck Institute for Biophysical Chemistry, Department of Molecular Biology, Göttingen, Germany. [10]Harvard Stem Cell Institute, Cambridge, MA, USA. *e-mail: sten.linnarsson@ki.se; peter_kharchenko@hms.harvard.edu

# The single cell revolution

Interesting Biological Experiments

Computation



High impact paper

LETTER

https://doi.org/10.1038/s41586-018-0414-6

RNA velocity of single cells

Gioele La Manno[1,2], Ruslan Soldatov[1], Amit Zeisel[1,3], Emelie Braun[1,3], Hannah Hochgerner[3,4], Viktor Petukhov[3,4], Katja Lidschreiber[5], Maria E. Kastriti[6], Peter Lönnerberg[1,2], Alessandro Furlan[7], Jean Fan[1], Lars E. Borm[1,2], Zehua Liu[1], David van Bruggen[1], Jimin Guo[1], Xiaoling He[1], Roger Barker[1], Erik Sundström[8], Gonçalo Castelo-Branco[7], Patrick Cramer[5,9], Igor Adameyko[6], Sten Linnarsson[1,2]* & Peter V. Kharchenko[3,10]*

**RNA abundance is a powerful indicator of the state of individual cells. Single-cell RNA sequencing can reveal RNA abundance with high quantitative accuracy, sensitivity and throughput[1]. However, this approach captures only a static snapshot at a point in time, posing a challenge for the analysis of time-resolved phenomena such as embryogenesis or tissue regeneration. Here we show that RNA velocity—the time derivative of the gene expression state—can be directly estimated by distinguishing between unspliced and spliced mRNAs in common single-cell RNA sequencing protocols. RNA velocity is a high-dimensional vector that predicts the future state of individual cells on a timescale of hours. We validate its accuracy in the neural crest lineage, demonstrate its use on multiple published datasets and technical platforms, reveal the branching lineage tree of the developing mouse hippocampus, and examine the kinetics of transcription in human embryonic brain. We expect RNA velocity to greatly aid the analysis of developmental lineages and cellular dynamics, particularly in humans.**

- Machine learning
- Linear algebra
- Probability theory
- Statistical analysis
- Algorithm design

# It's all Greek to me...

**Definition 1.** *The* $t$-step **potential distance** *is defined as* $\mathfrak{V}^t(x,y) \triangleq \|U_x^t - U_y^t\|_2$, $x, y \in \mathcal{X}$.

The following proposition shows a relation between the two metrics by expressing the potential distance in embedded diffusion map coordinates[1] for fixed-bandwidth Gaussian-based diffusion (i.e., generated by $P_\varepsilon$ from Eq. 2):

**Proposition 1.** *Given a diffusion process defined by a fixed-bandwidth Gaussian kernel, the potential distance from Def 1 can be written as* $\mathfrak{V}^t(x,y) = \left(\sum_{z\in\mathcal{X}} \log^2\left(\frac{1+\left\langle\Phi^{t/2}(x),\Phi^{t/2}(z)\right\rangle}{1+\left\langle\Phi^{t/2}(y),\Phi^{t/2}(z)\right\rangle}\right)\right)^{1/2}$

*Proof.* According to the spectral theorem, the entries of $P_\varepsilon^t$ can be written as

$$[P_\varepsilon^t]_{(x,y)} = \psi_0(y) + \sum_{i=1}^{n-1} \lambda_i^t \phi_i(x)\psi_i(y)$$

since powers of the operator $P_\varepsilon$ only affect the eigenvalues, which are taken to the same power, and since the trivial eigenvalue $\lambda_0$ is one and the corresponding right eigenvector $\phi_0$ only consists of ones. Furthermore, it can be verified that the left and right eigenvectors of $P_\varepsilon$ are related by $\psi_i(y) = \phi_i(y)\psi_0(y)$, thus, combined with Eqs. 4 and 6, we get

$$p_{\varepsilon,x}^t(y) = \psi_0(y)\left(1 + \sum_{i=1}^{n-1} \lambda_i^t \phi_i(x)\phi_i(y)\right) = \psi_0(y)\left(1 + \left\langle\Phi_\varepsilon^{t/2}(x), \Phi_\varepsilon^{t/2}(x)\right\rangle\right) .$$

By applying the logarithm to both ends of this equation we express the entries of the potential representation $U_{\varepsilon,x}^t$ as

$$U_{\varepsilon,x}^t(y) = -\log(1 + \left\langle\Phi_\varepsilon^{t/2}(x), \Phi_\varepsilon^{t/2}(y)\right\rangle) - \log(\psi_0(y)) ,$$

and thus for any $j = 1, \ldots, N$,

$$\begin{aligned}\left(U_{\varepsilon,x}^t(x_j) - U_{\varepsilon,y}^t(x_j)\right)^2 &= \left[\log(1 + \left\langle\Phi_\varepsilon^{t/2}(x), \Phi_\varepsilon^{t/2}(x_j)\right\rangle)\right. \\ &- \left.\log(1 + \left\langle\Phi_\varepsilon^{t/2}(y), \Phi_\varepsilon^{t/2}(x_j)\right\rangle)\right]^2 \\ &= \log^2\left(\frac{1 + \left\langle\Phi_\varepsilon^{t/2}(x), \Phi_\varepsilon^{t/2}(x_j)\right\rangle}{1 + \left\langle\Phi_\varepsilon^{t/2}(y), \Phi_\varepsilon^{t/2}(x_j)\right\rangle}\right) ,\end{aligned}$$

which yields the result in the proposition. □

Moon et al. (2019)

# What reading single cell methods can feel like

# What is machine learning?

# What is machine learning?

Machine learning is the process of identifying patterns in data.

# Two kinds of machine learning

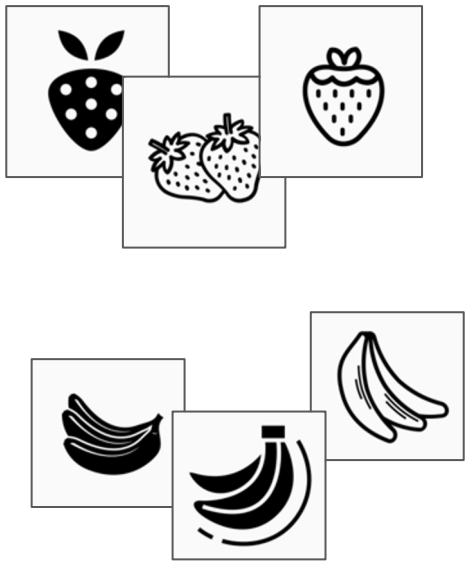
Supervised learning

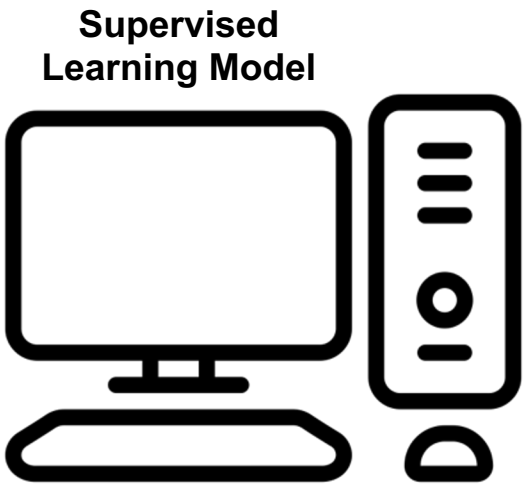
- Have a bunch of labelled data, want to label new data

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data



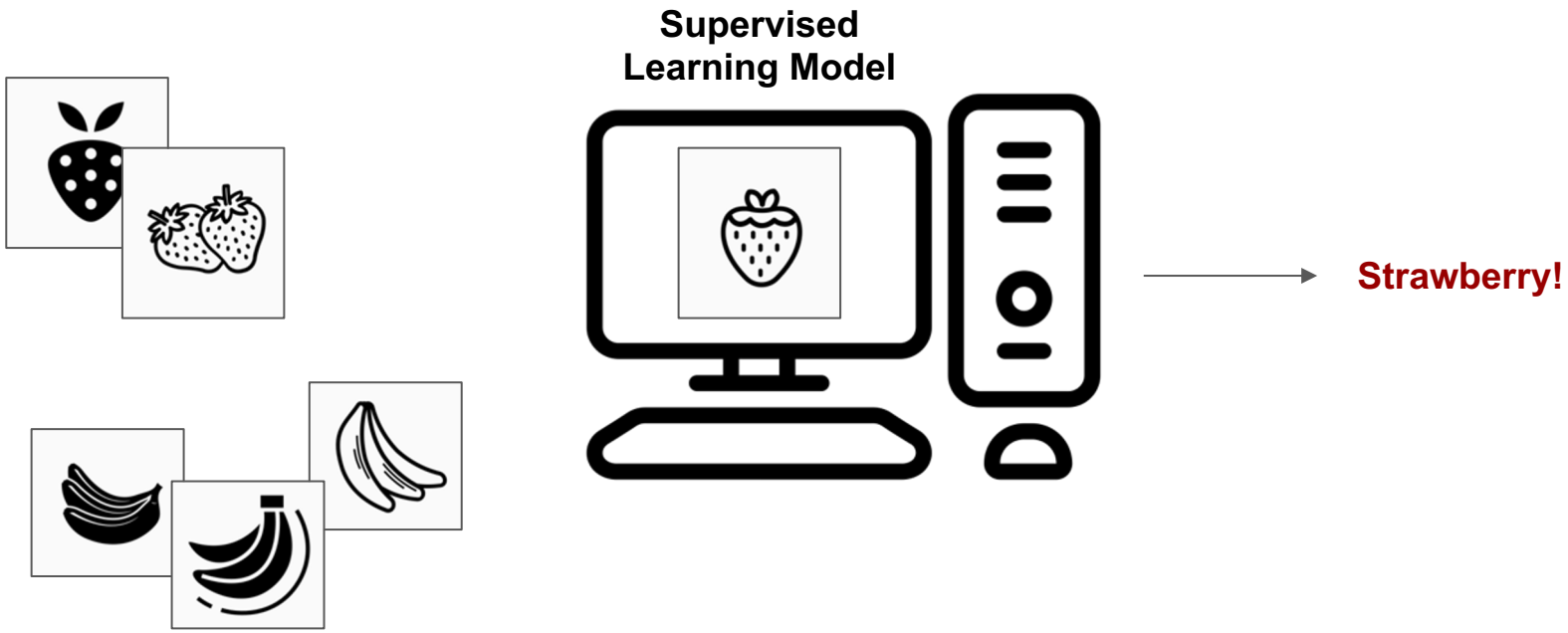



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

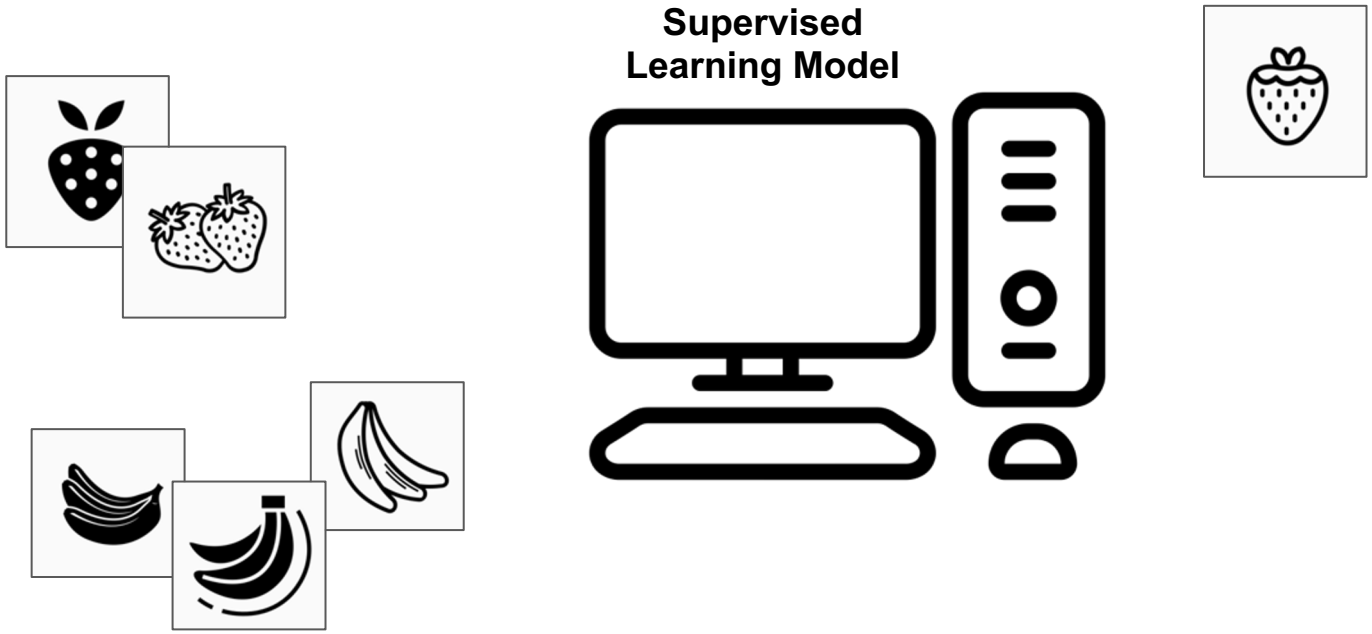



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

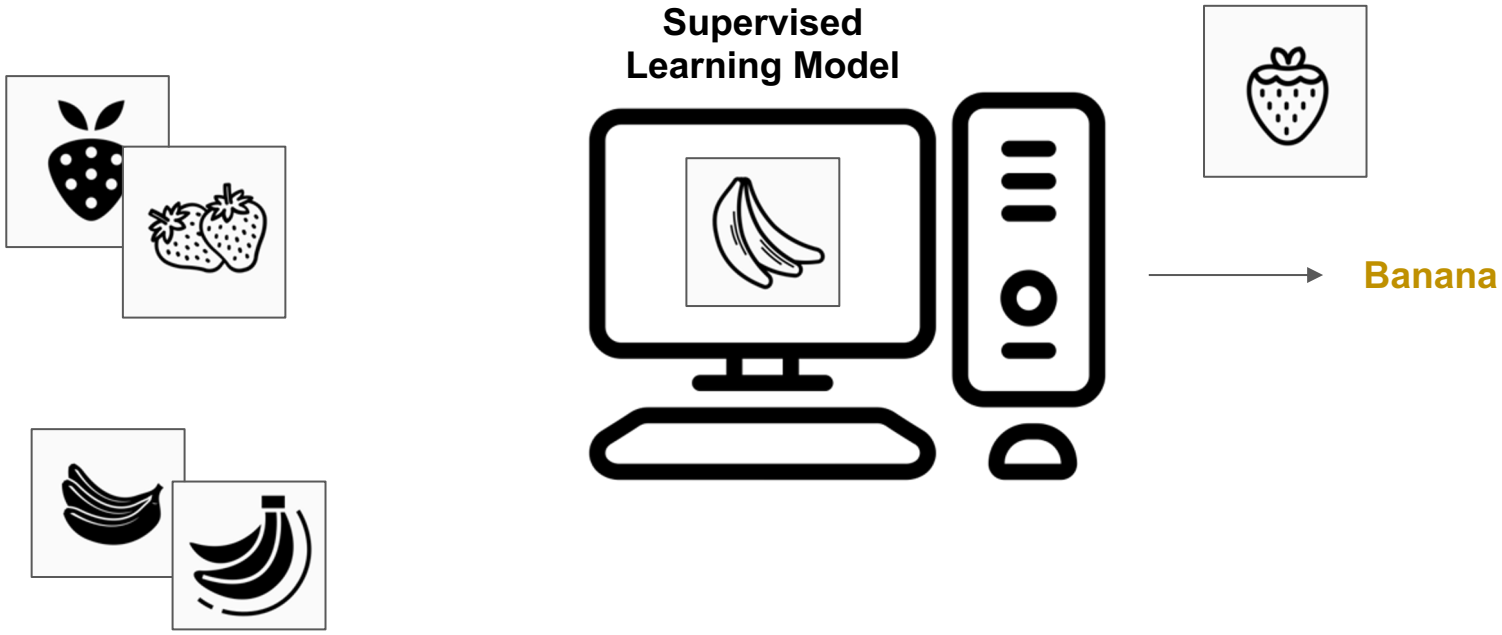



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

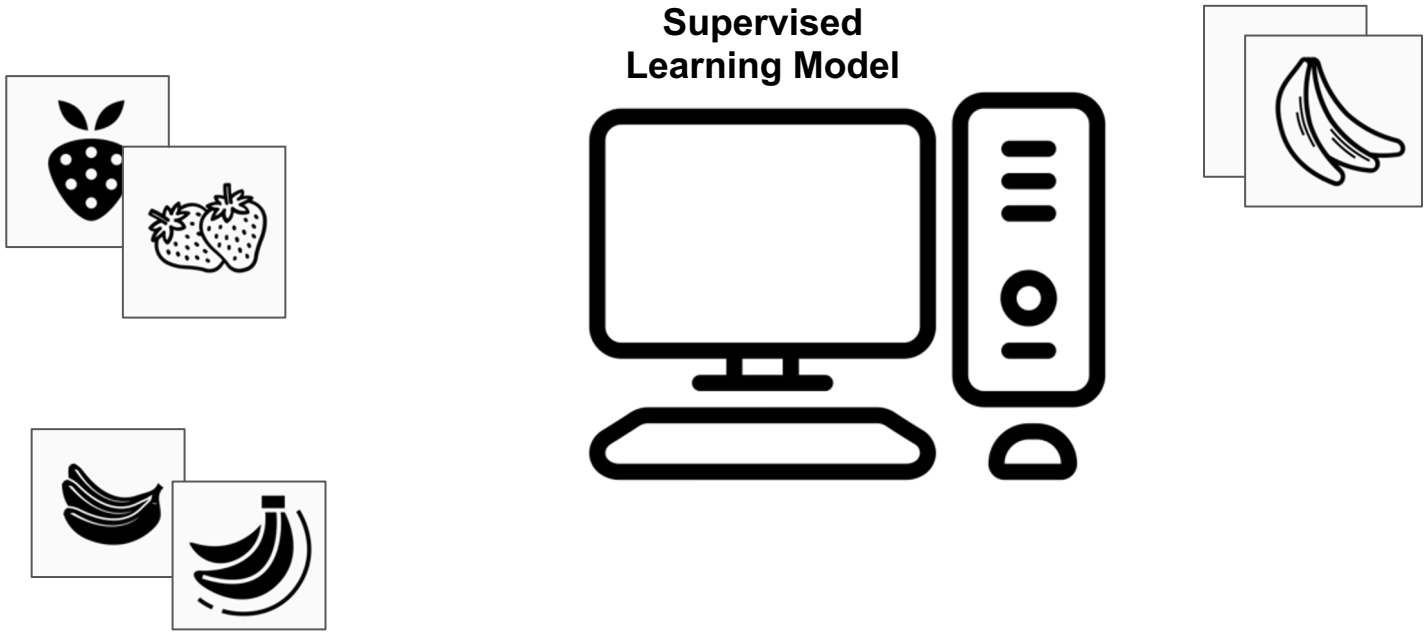



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

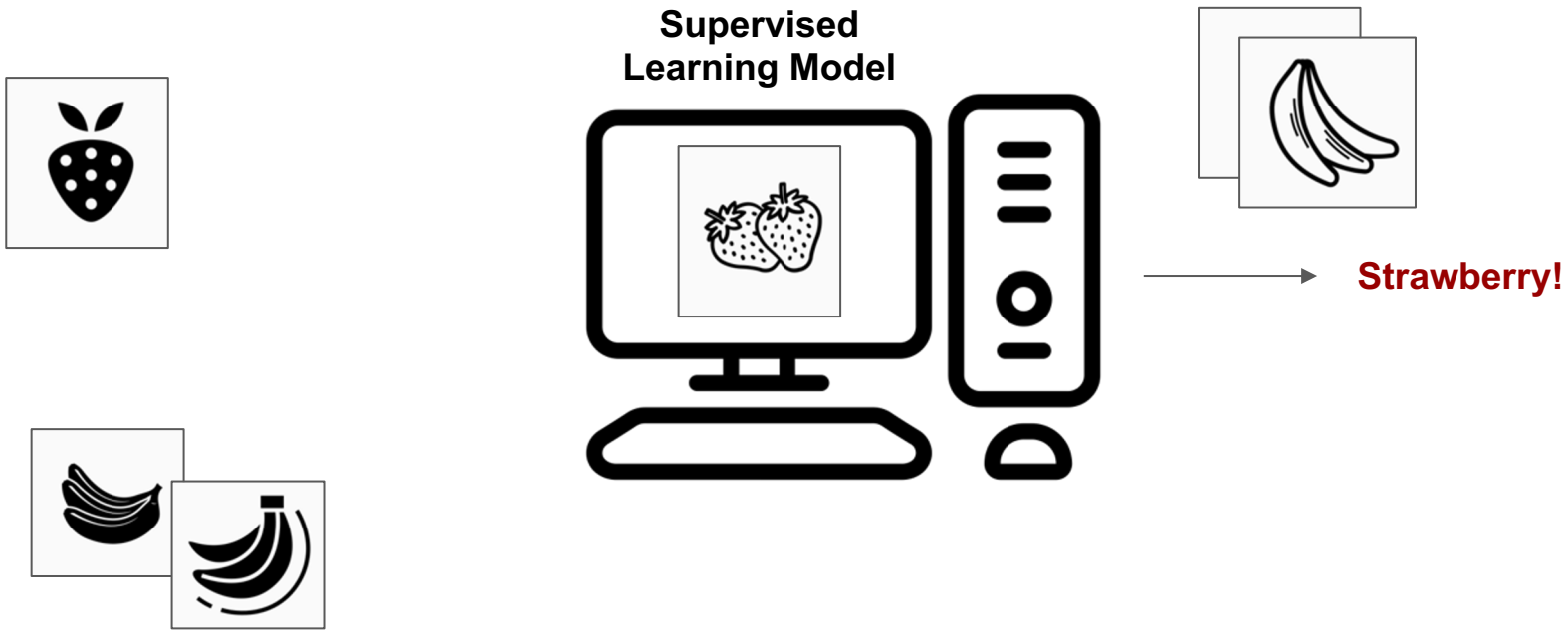



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

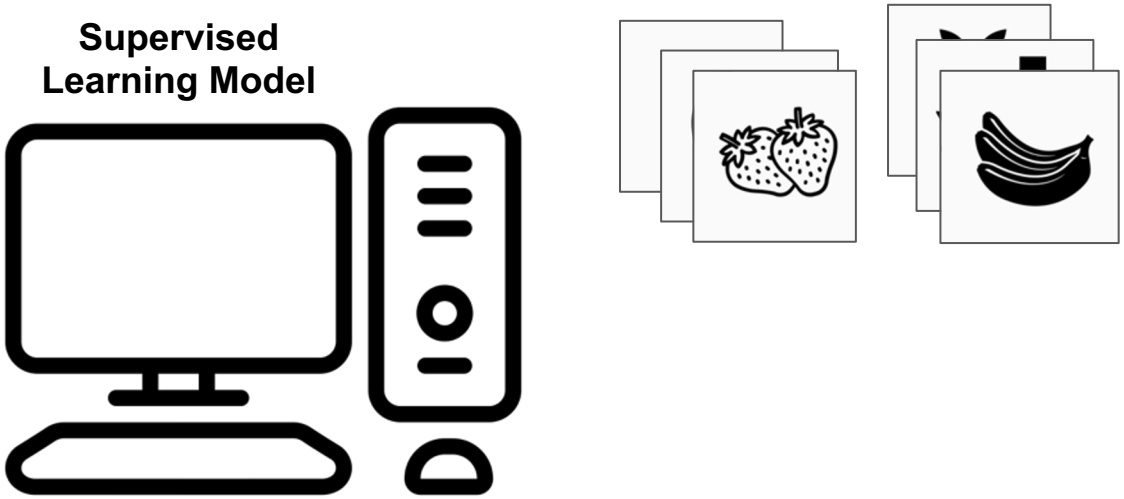



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

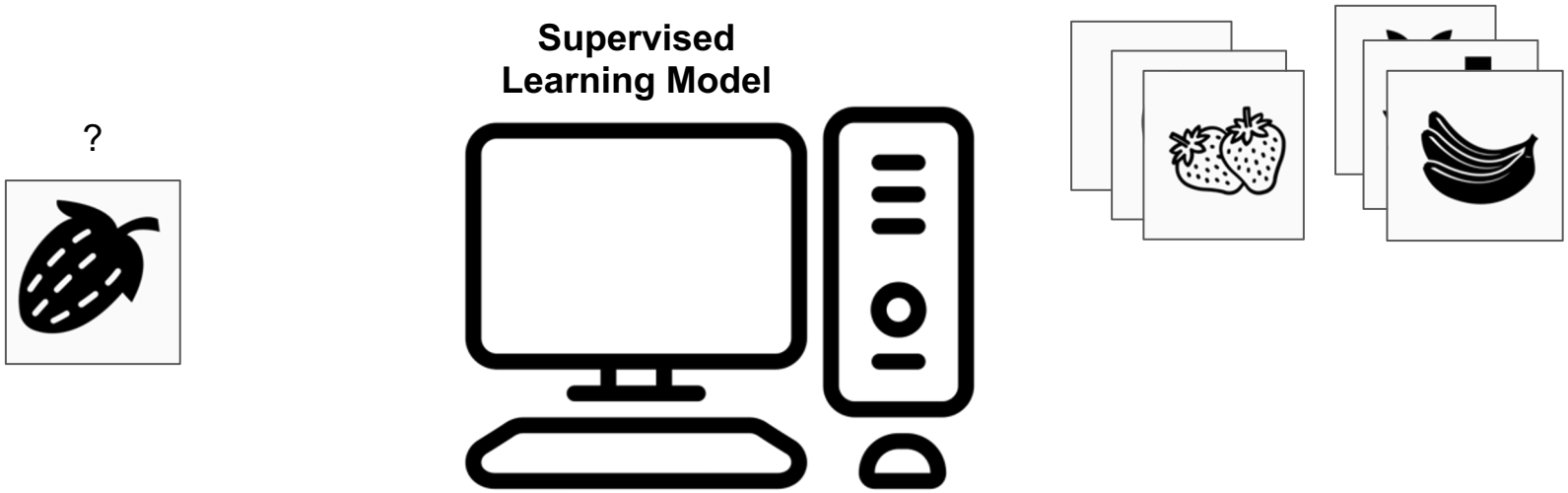



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data

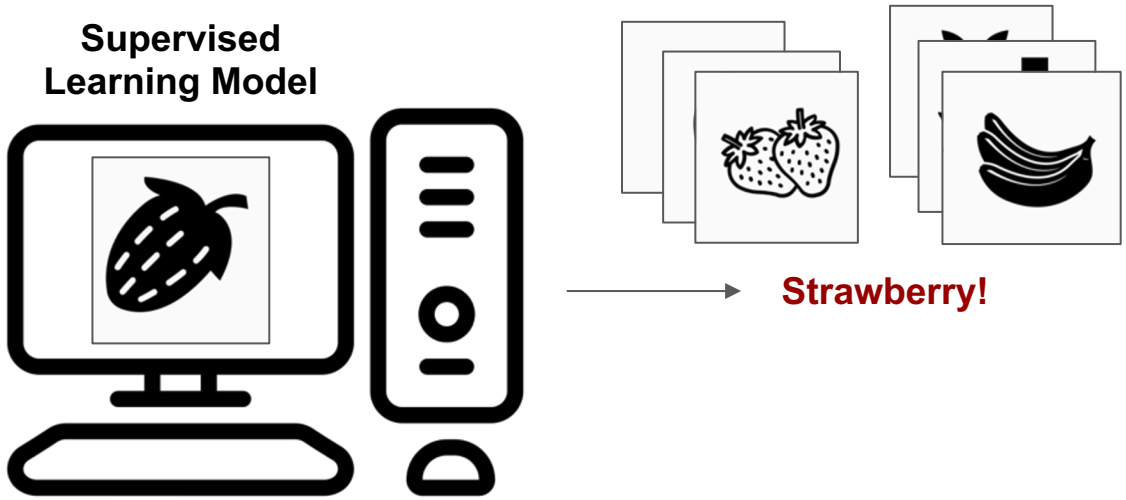



Noun project

# Two kinds of machine learning

## Supervised learning

- Have a bunch of labelled data, want to label new data




Noun project

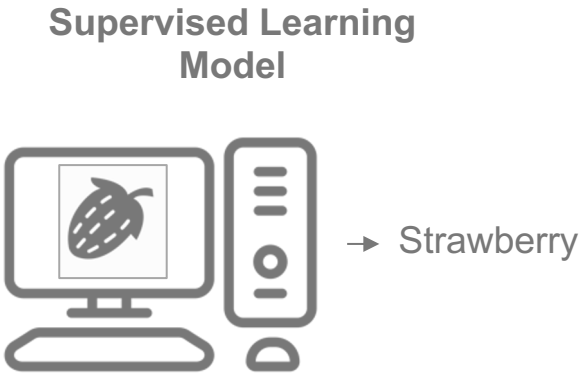
# Two kinds of machine learning

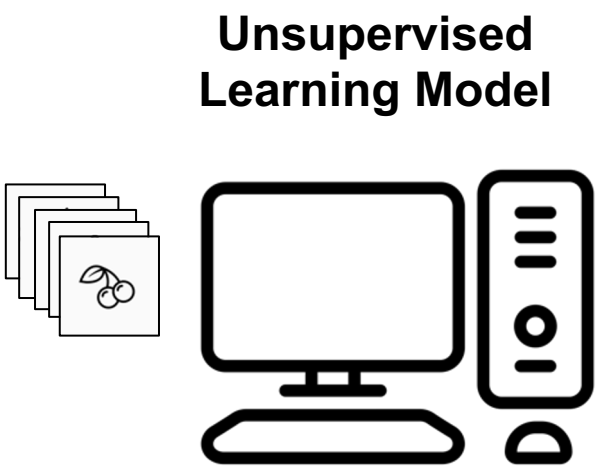
## Supervised learning

- Have a bunch of labelled data, want to label new data

## Unsupervised learning

- Have a bunch of unlabeled data, want to organize it




Noun project

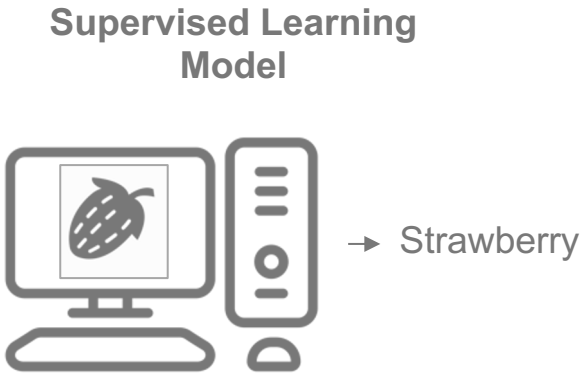
# Two kinds of machine learning

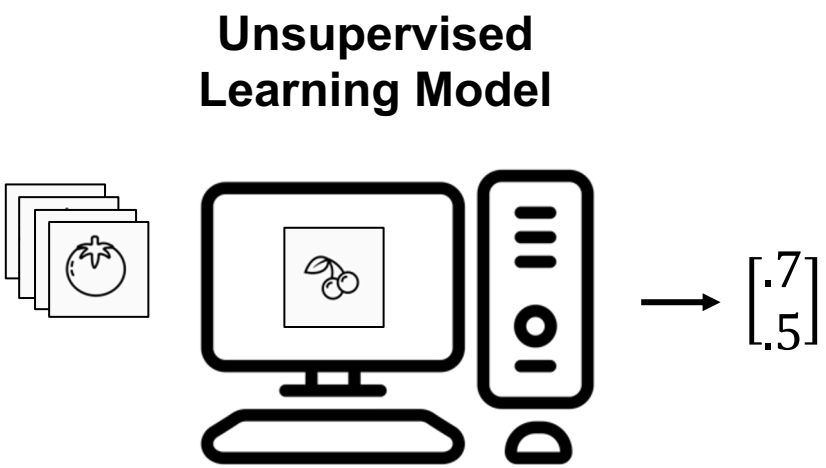
## Supervised learning

- Have a bunch of labelled data, want to label new data

## Unsupervised learning

- Have a bunch of unlabeled data, want to organize it




Noun project

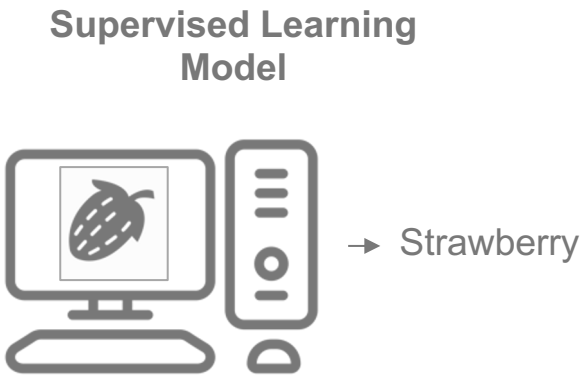
# Two kinds of machine learning

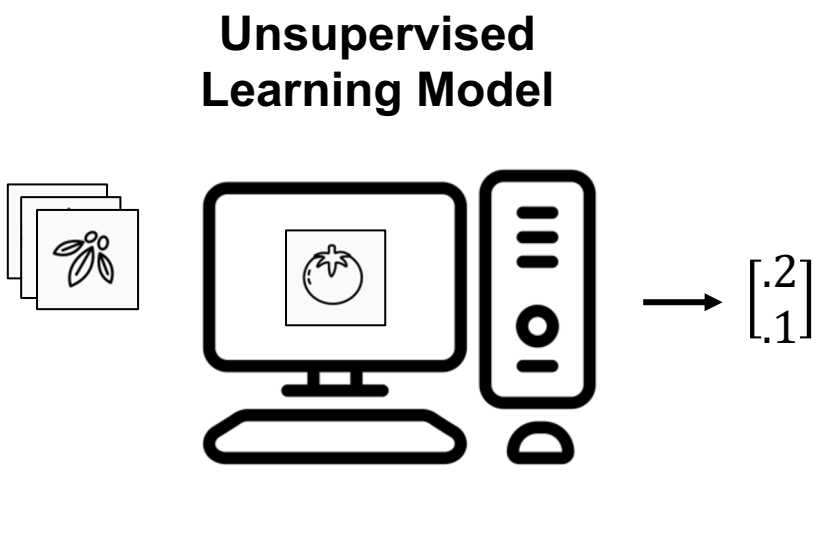
## Supervised learning

- Have a bunch of labelled data, want to label new data

## Unsupervised learning

- Have a bunch of unlabeled data, want to organize it




Noun project

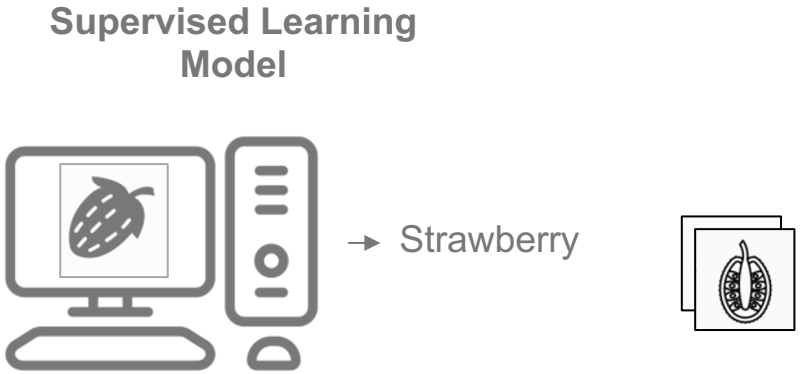
# Two kinds of machine learning

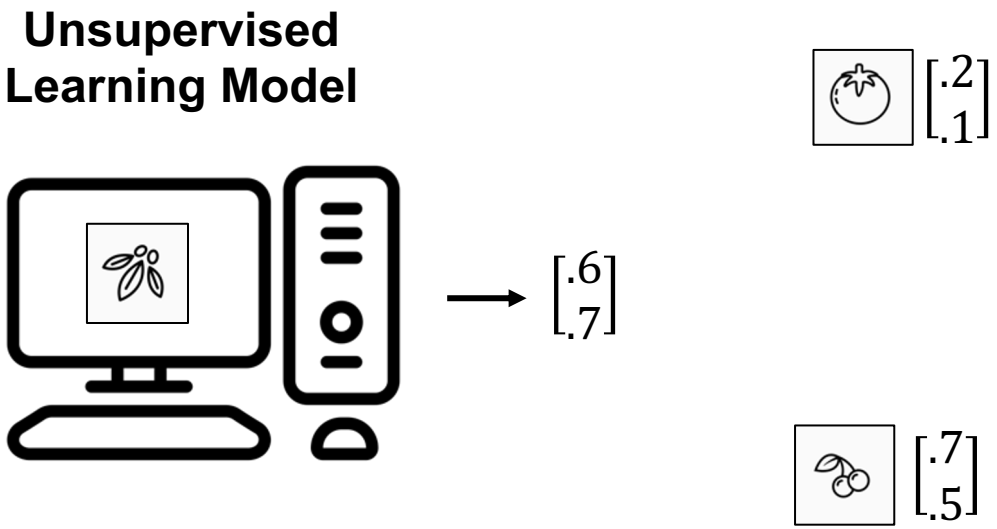
## Supervised learning

- Have a bunch of labelled data, want to label new data

## Unsupervised learning

- Have a bunch of unlabeled data, want to organize it



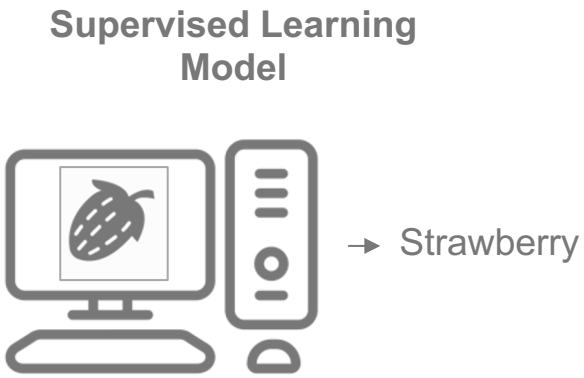




Noun project

# Two kinds of machine learning

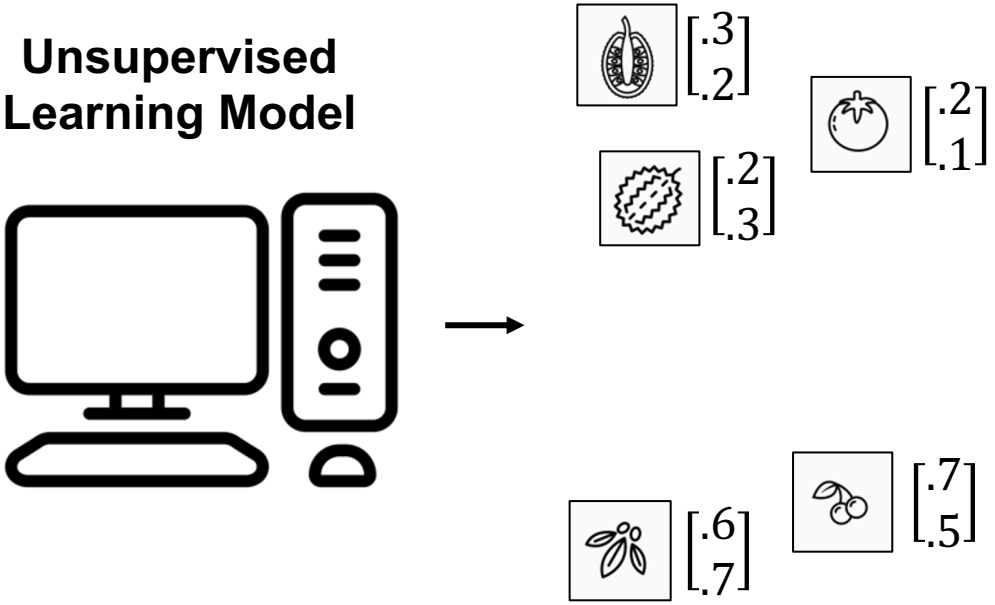
## Supervised learning

- Have a bunch of labelled data, want to label new data




## Unsupervised learning

- Have a bunch of unlabeled data, want to organize it

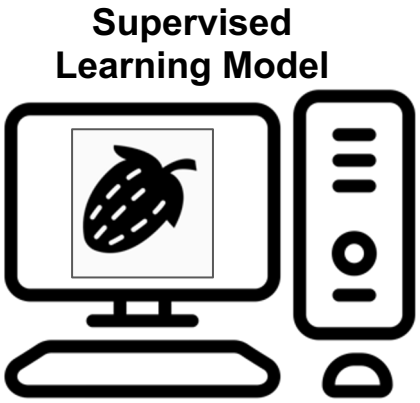



Noun project

# Two kinds of machine learning

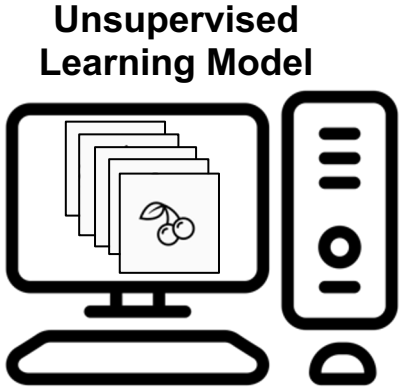
## Supervised learning

- Have a bunch of labelled data, want to label new data
- Learn a function $f(X) \rightarrow Y$ where all values of $Y$ are known for some samples of $X$




## Unsupervised learning

- Have a bunch of unlabeled data, want to organize it
- Learn an embedding $f(X) \rightarrow Y, X \in \mathbb{R}^n, Y \in \mathbb{R}^m, n \gg m$
- Lower dimensional, easier to interpret (e.g. as clusters)




Noun project

# Is linear regression an example of supervised or unsupervised machine learning?

Supervised machine learning

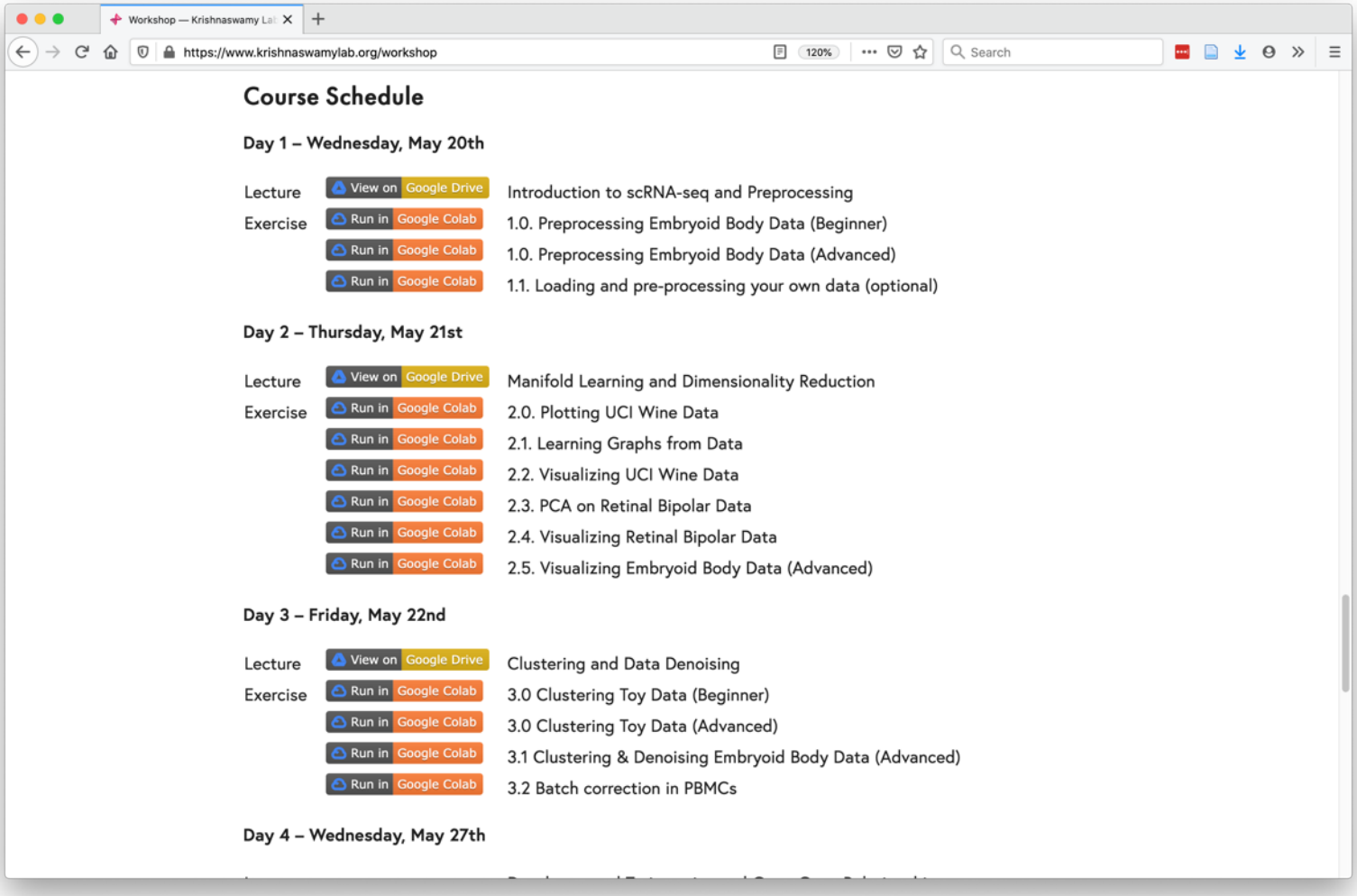
Unsupervised machine learning

# Is clustering an example of supervised or unsupervised machine learning?

Supervised machine learning **A**

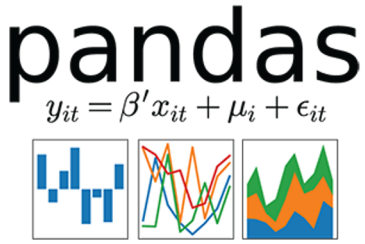
Unsupervised machine learning **B**

# Course Schedule




https://www.krishnaswamylab.org/workshop

# Why Python?

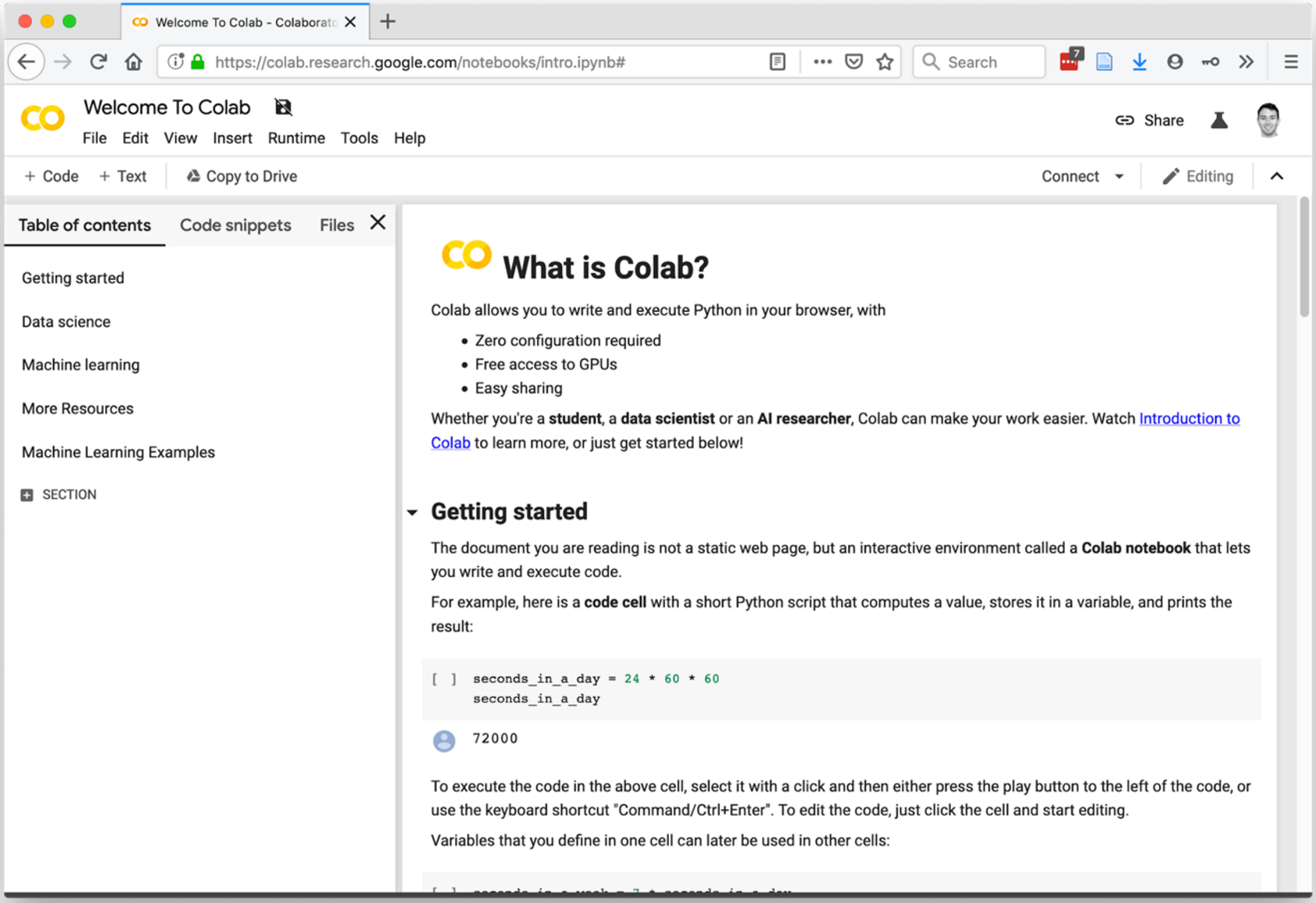





## Why Python?

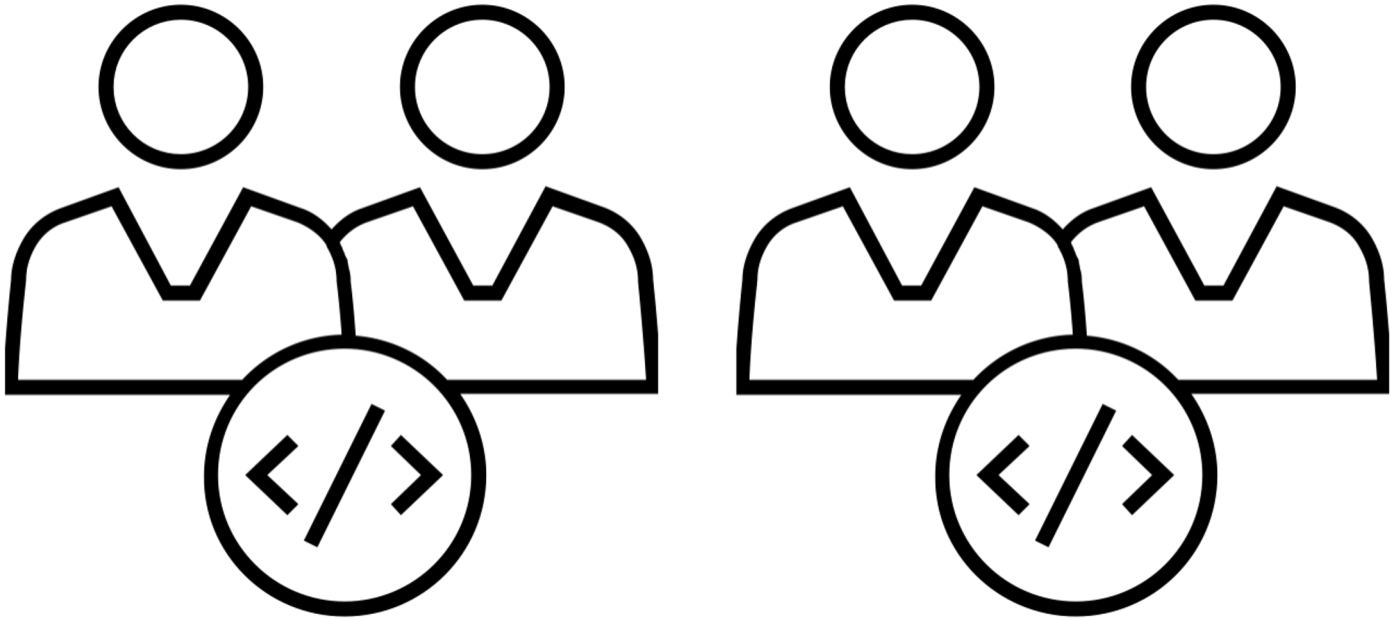
Tensorflow

Pytorch

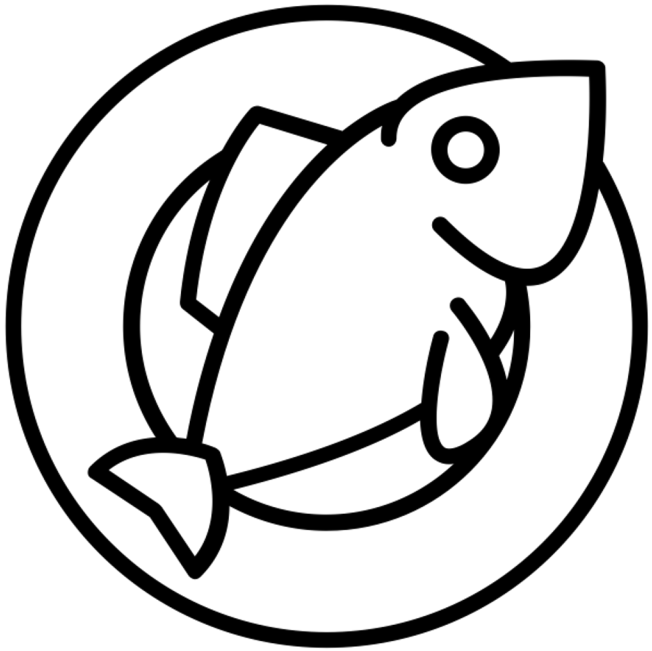




https://research.google.com/colaboratory/faq.html

# Team programming

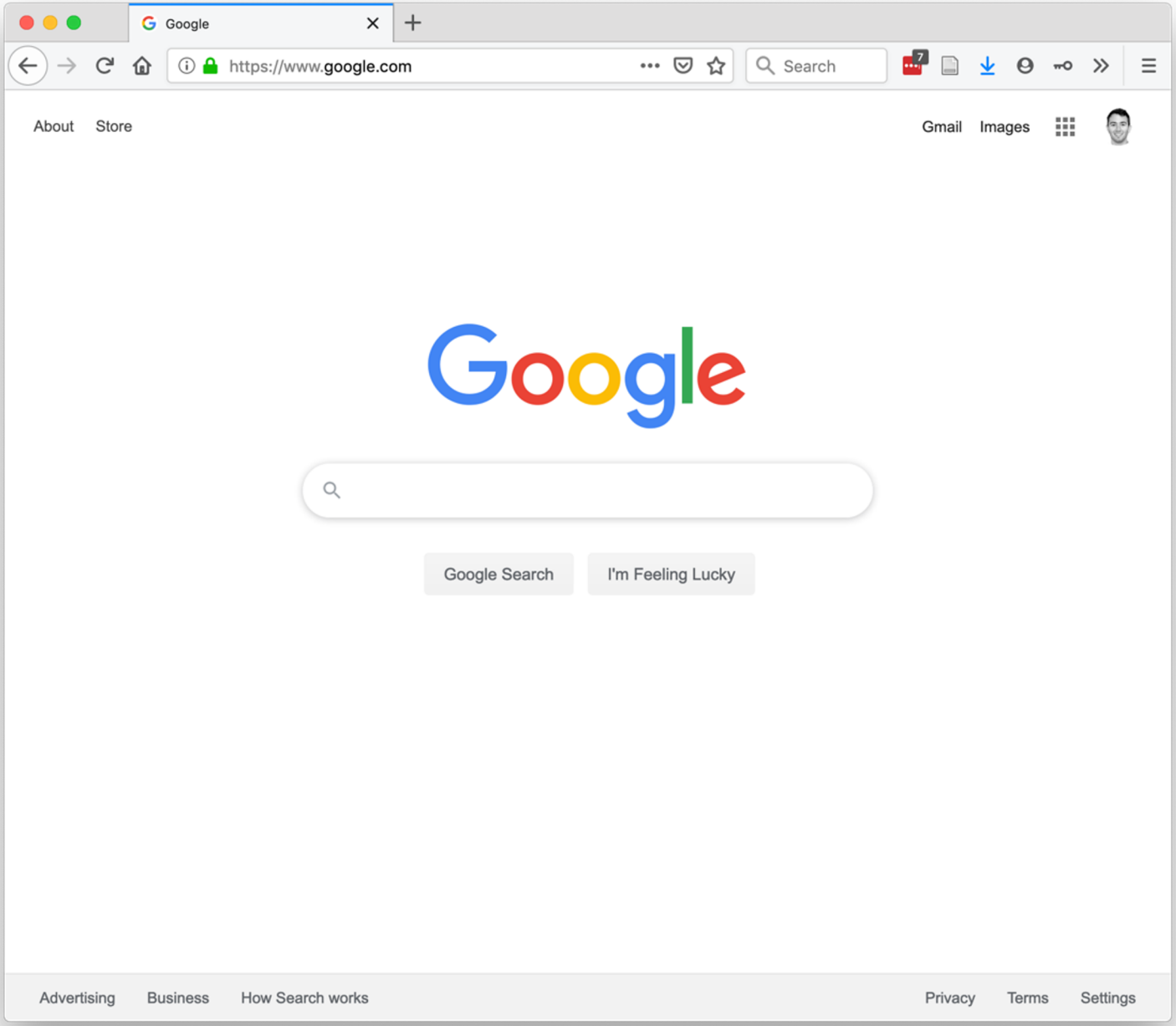


Noun project



vs.

Noun project


Google
https://www.google.com
Search
About
Store
Gmail
Images
Google
Google Search
I'm Feeling Lucky
Advertising
Business
How Search works
Privacy
Terms
Settings

**scprep.io.load_10X**(*data_dir, sparse=True, gene_labels='symbol', allow_duplicates=None*) [source]

Basic IO for 10X data produced from the 10X Cellranger pipeline.

A default run of the *cellranger count* command will generate gene-barcode matrices for secondary analysis. For both "raw" and "filtered" output, directories are created containing three files: 'matrix.mtx', 'barcodes.tsv', 'genes.tsv'. Running *scprep.io.load_10X(data_dir)* will return a Pandas DataFrame with genes as columns and cells as rows.

**Parameters:**
- **data_dir** (*string*) – path to input data directory expects 'matrix.mtx', 'genes.tsv', 'barcodes.tsv' to be present and will raise an error otherwise
- **sparse** (*boolean*) – If True, a sparse Pandas DataFrame is returned.
- **gene_labels** (*string, {'id', 'symbol', 'both'} optional, default: 'symbol'*) – Whether the columns of the dataframe should contain gene ids or gene symbols. If 'both', returns symbols followed by ids in parentheses.
- **allow_duplicates** (*bool, optional (default: None)*) – Whether or not to allow duplicate gene names. If None, duplicates are allowed for dense input but not for sparse input.

**Returns:** **data** – If sparse, data will be a pd.DataFrame[pd.SparseArray]. Otherwise, data will be a pd.DataFrame.

**Return type:** array-like, shape=[n_samples, n_features]

**scprep.io.load_10X_HDF5**(*filename, genome=None, sparse=True, gene_labels='symbol', allow_duplicates=None, backend=None*) [source]

Basic IO for HDF5 10X data produced from the 10X Cellranger pipeline.

# Bring-your-own-data workshop

#2020-workshop-byod-help

https://krishnaswamylab.org/get-help

Noun project

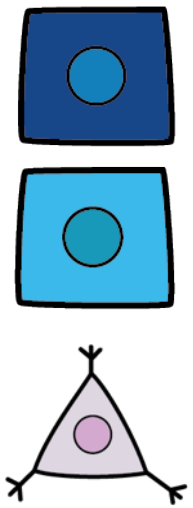
# Data Matrices and Representations

# Single Cell Data

- Each cell is a vector of measurements
  - e.g. Cell A = [40 0 20 18 5 0 …]
- The whole data is a matrix with many observations (cells) and features (proteins, genes)

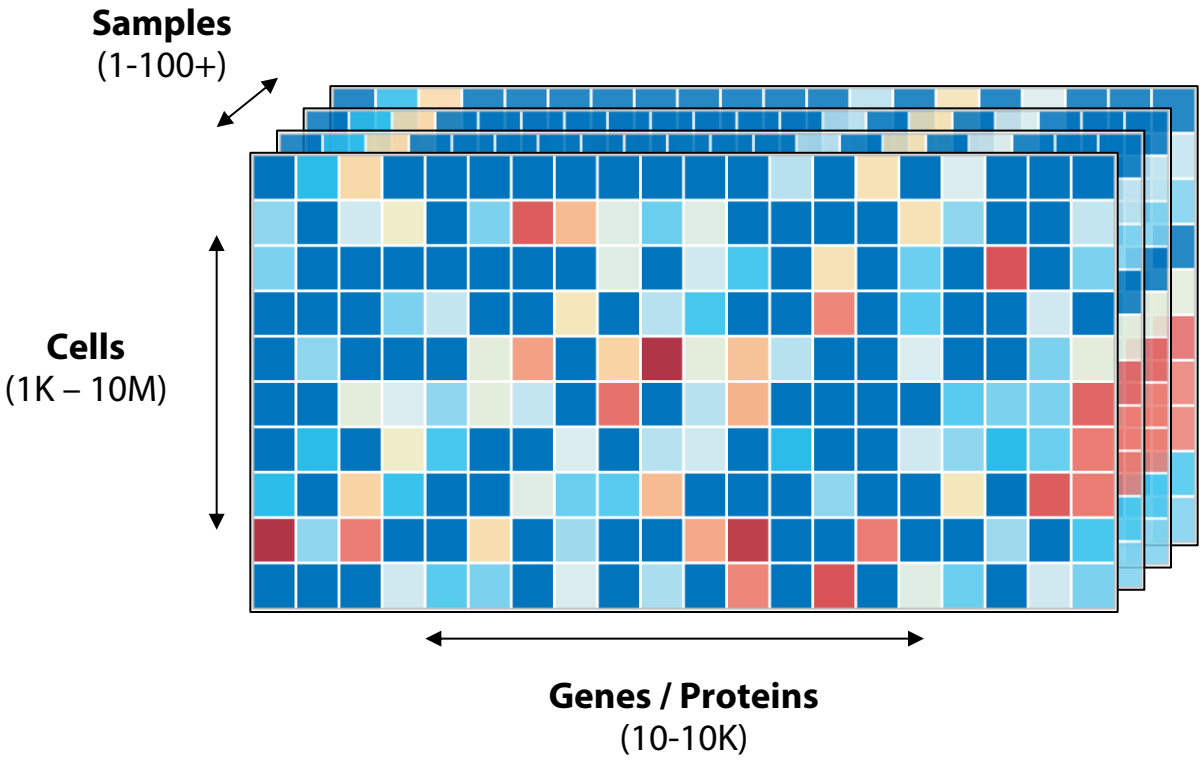
Features
(e.g. genes)

Observations
(e.g. cells)

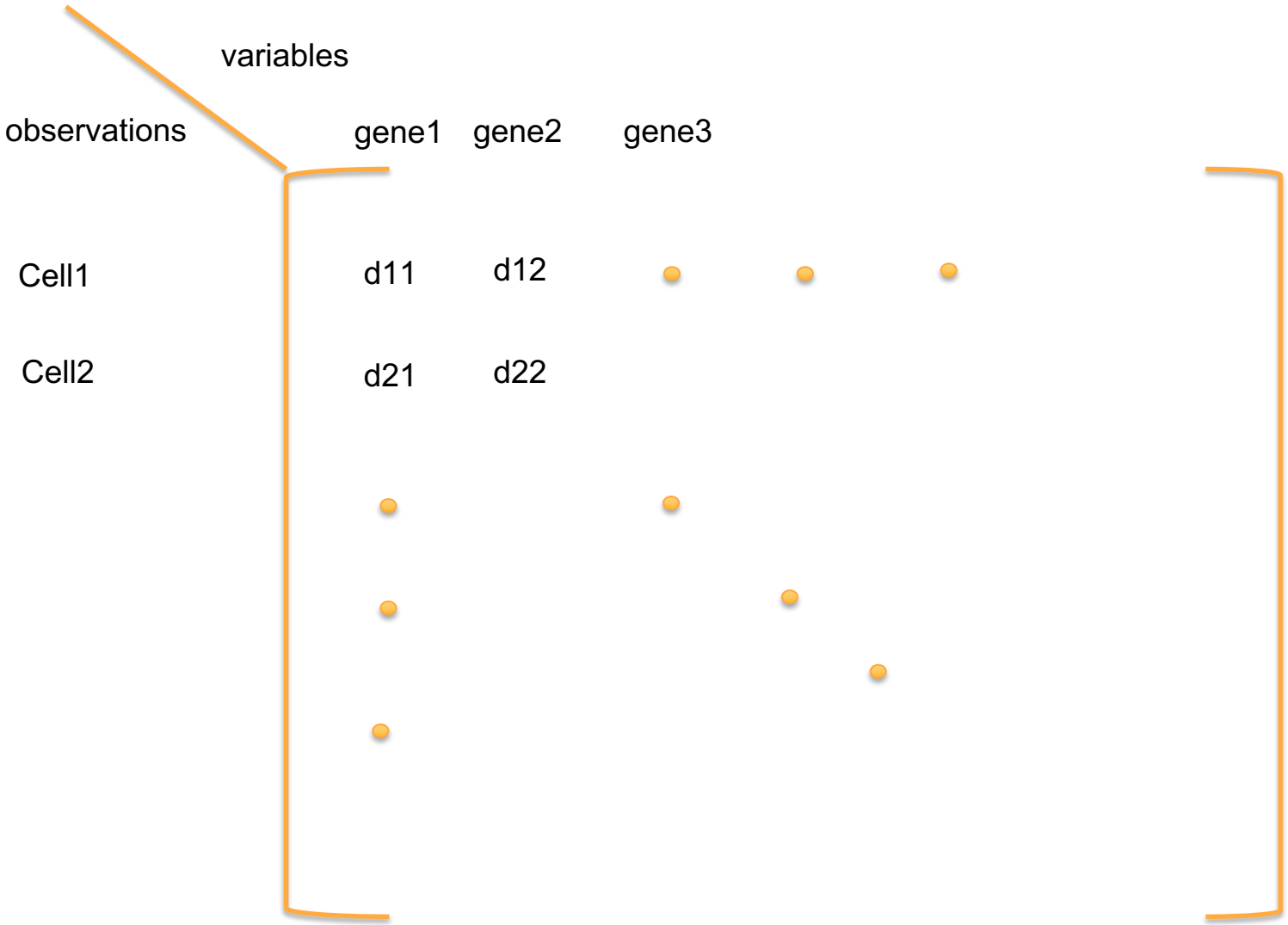


| | X | Y | Z |
|---|---|---|---|
| **A** | 10 | 20 | 70 |
| **B** | 20 | 40 | 140 |
| **C** | 20 | 0 | 80 |

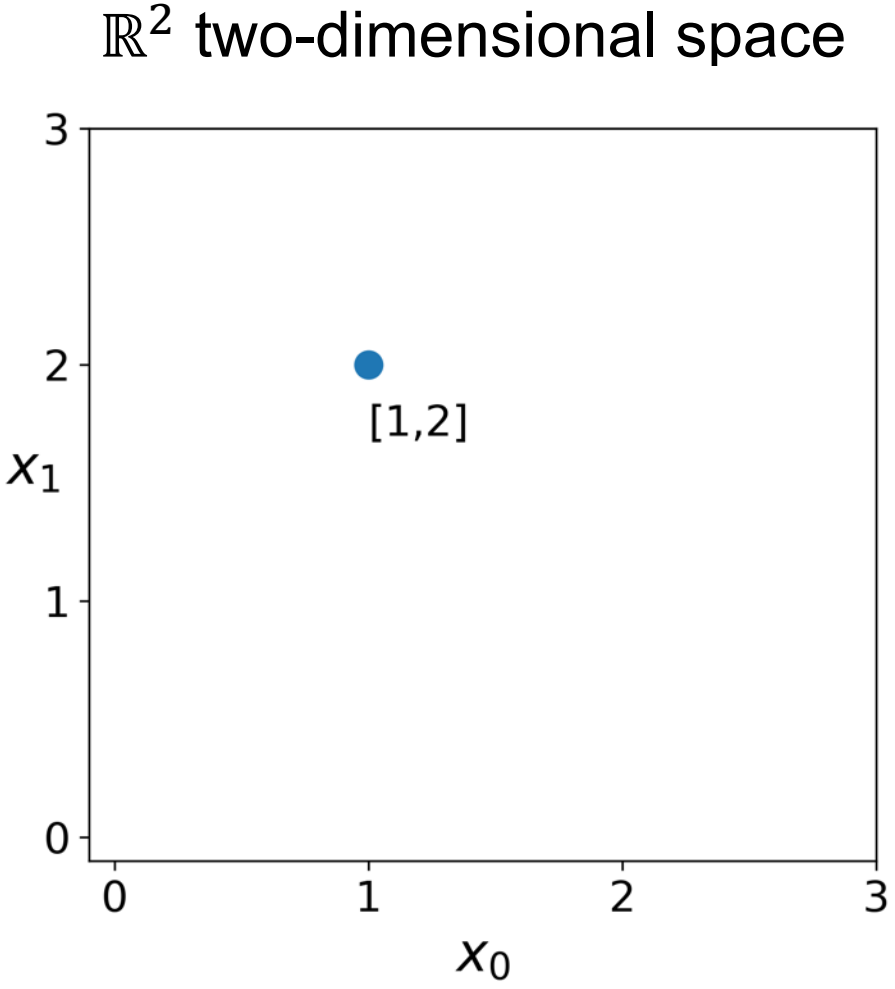
# Single Cell Data



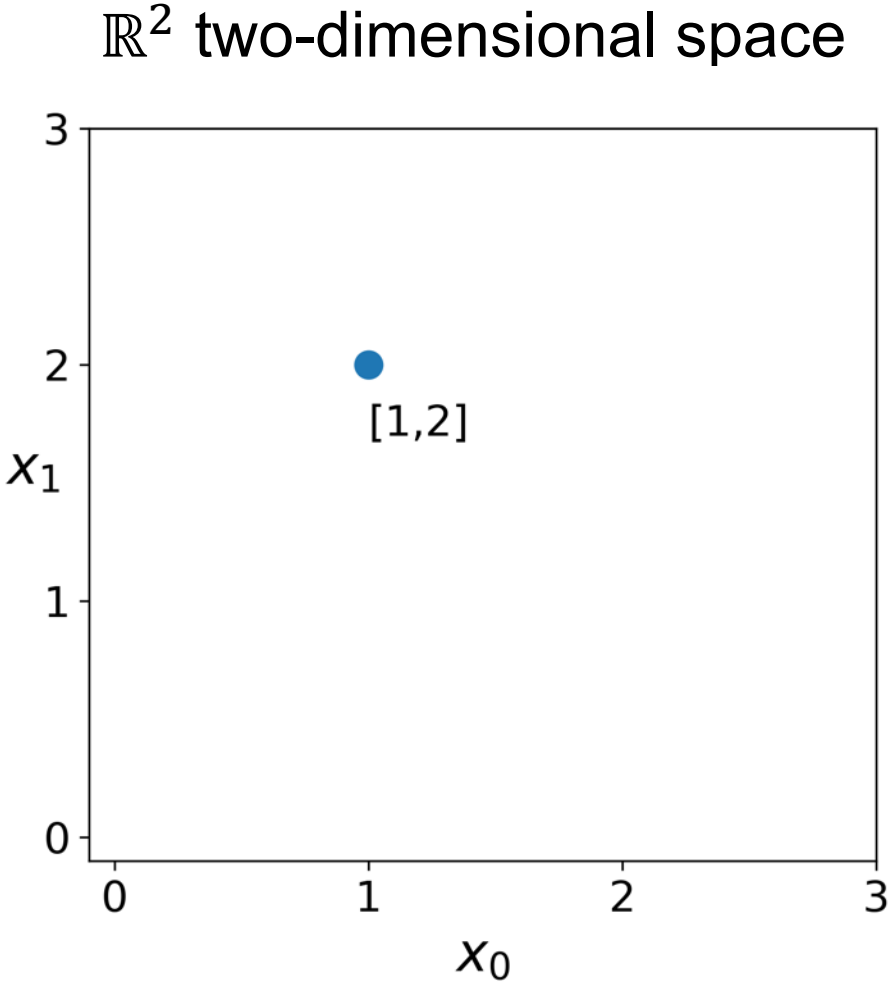
# Our Data is a High-dimensional Matrix



# Spatial Representation of Data



# Spatial Representation of Data

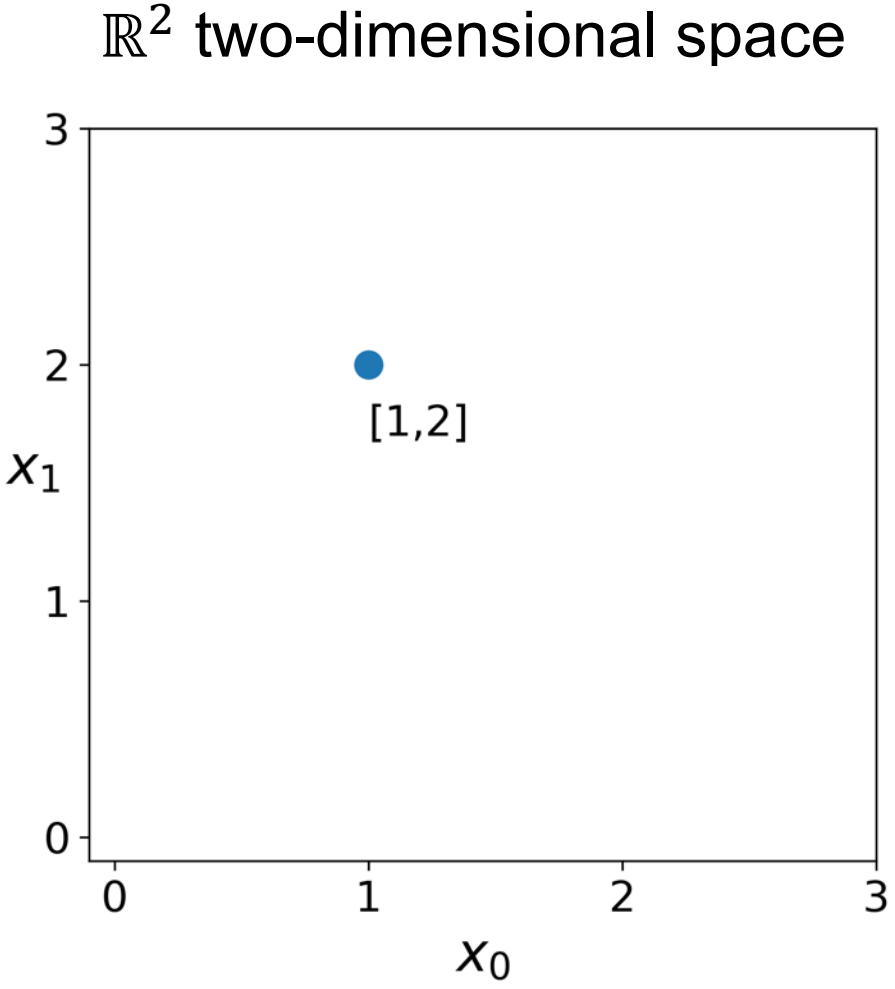


# Spatial Representation of Data

$\boldsymbol{a} \in \mathbb{R}^2$

$\boldsymbol{a} = [1,2]$

$x_0$ $x_1$



# Features are dimensions

$\boldsymbol{a} \in \mathbb{R}^2$

$\boldsymbol{a} = [1,2]$

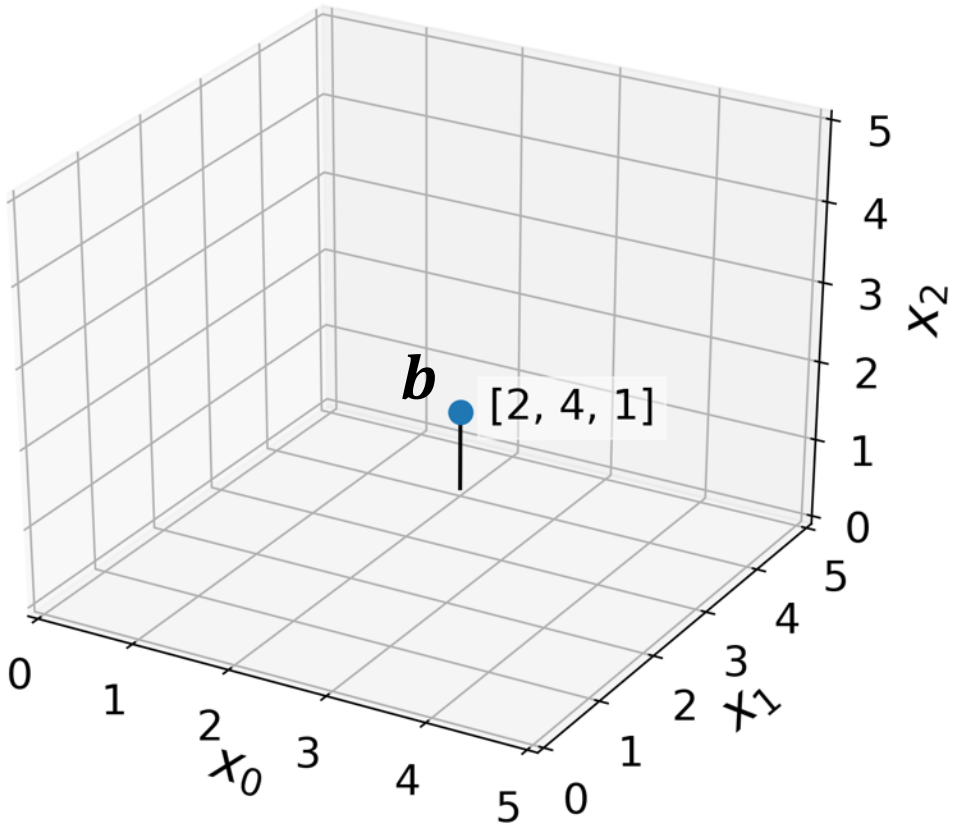
$x_0$ $x_1$

$\boldsymbol{b} \in \mathbb{R}^3$

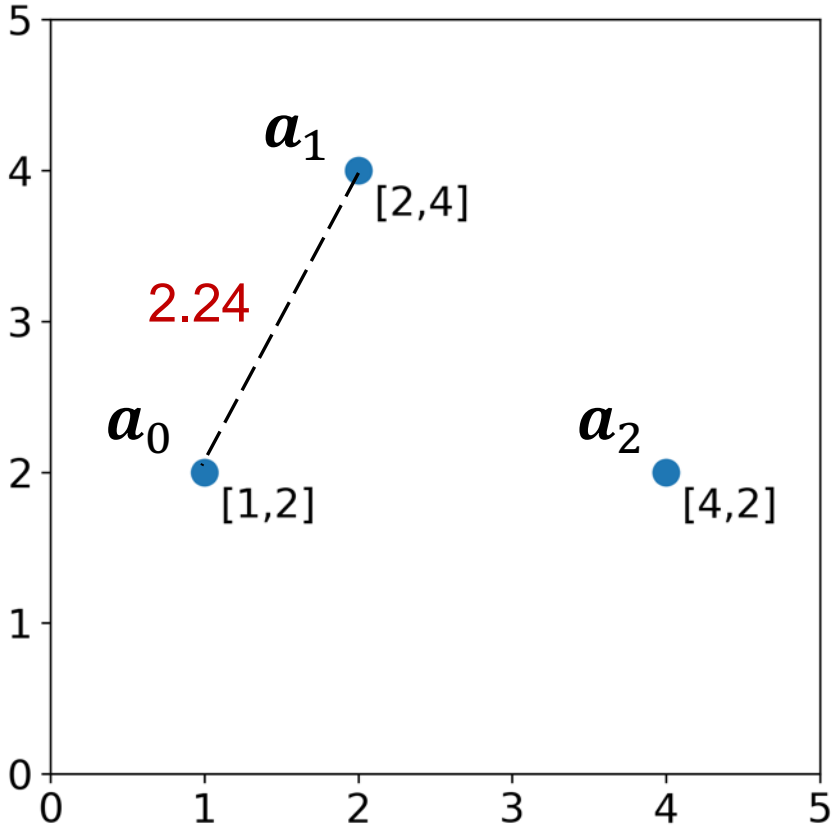
$\boldsymbol{b} = [2,4,1]$

$x_0$ $x_1$ $x_2$

$\mathbb{R}^3$ three-dimensional space



# Euclidean Distance between observations



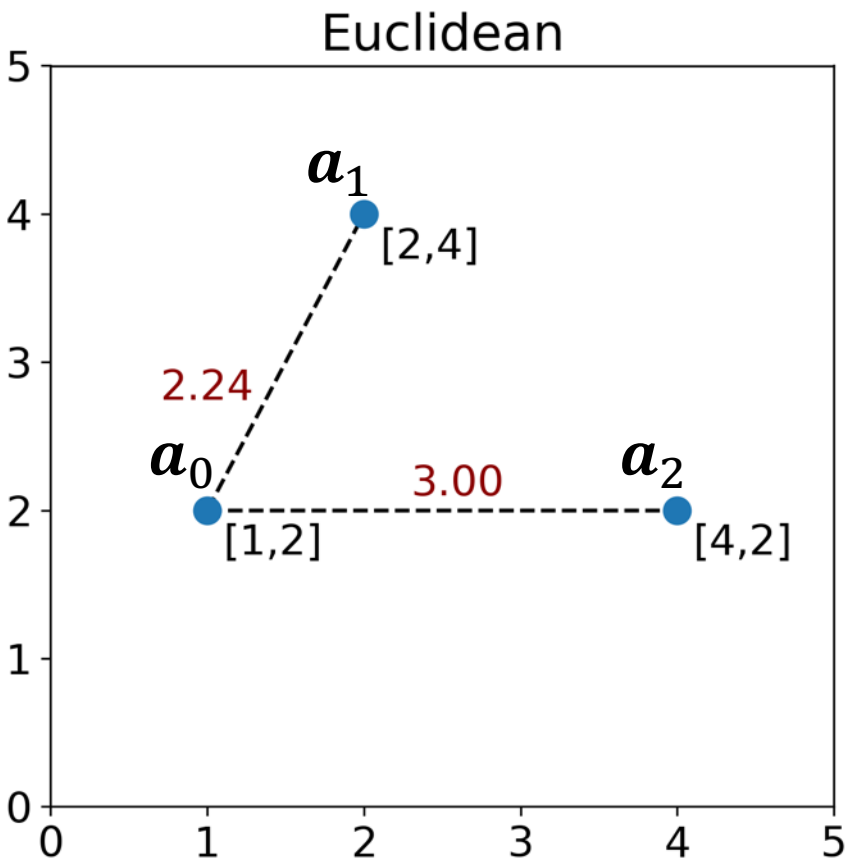

$$d_{euclidean}(a_0, a_1) = \|a_0 - a_1\|_2^2$$

$$= \sqrt{(a_{0,0} - a_{1,0})^2 + (a_{0,1} - a_{1,1})^2}$$

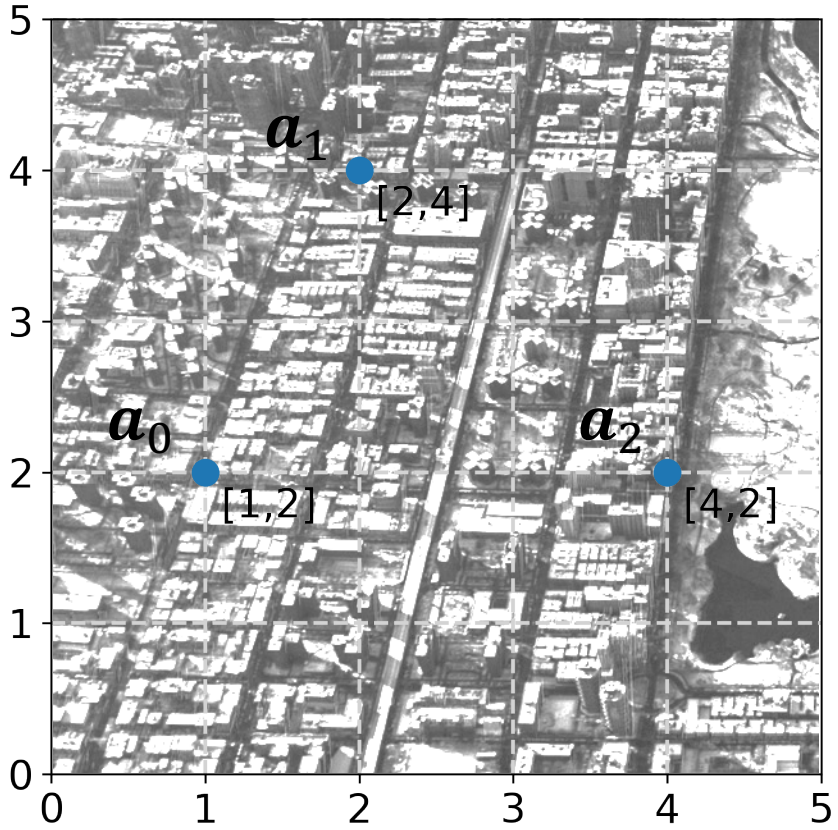
$$= \sqrt{(1 - 2)^2 + (2 - 4)^2}$$

$$\approx 2.24$$

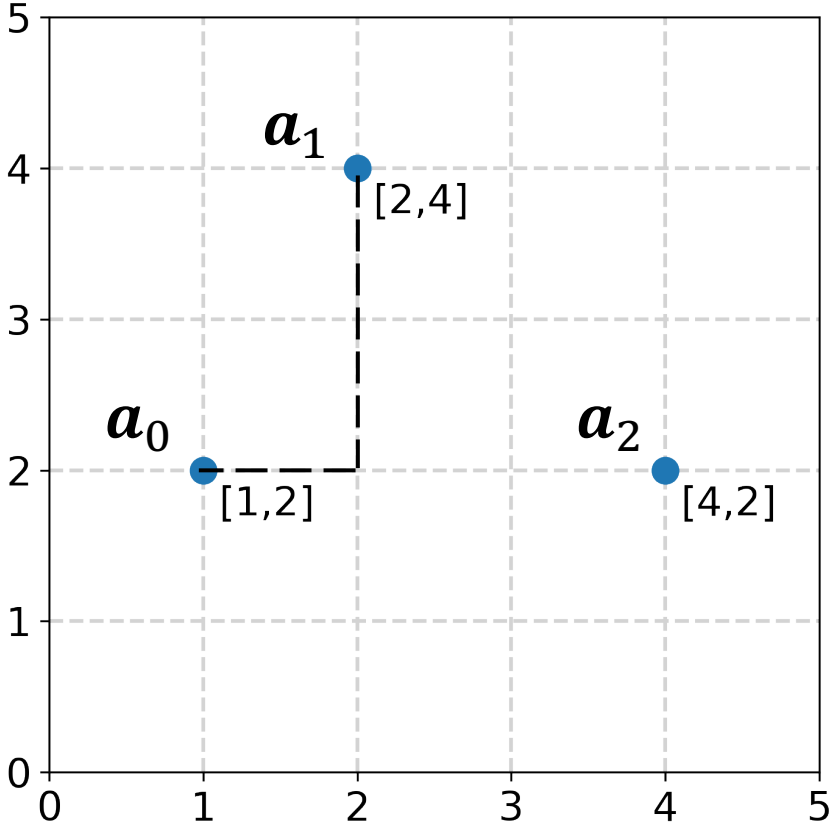
# How far away is each point from $a_0$?



# How far away is each point from $a_0$?



# Manhattan Distance




**Manhattan distance**

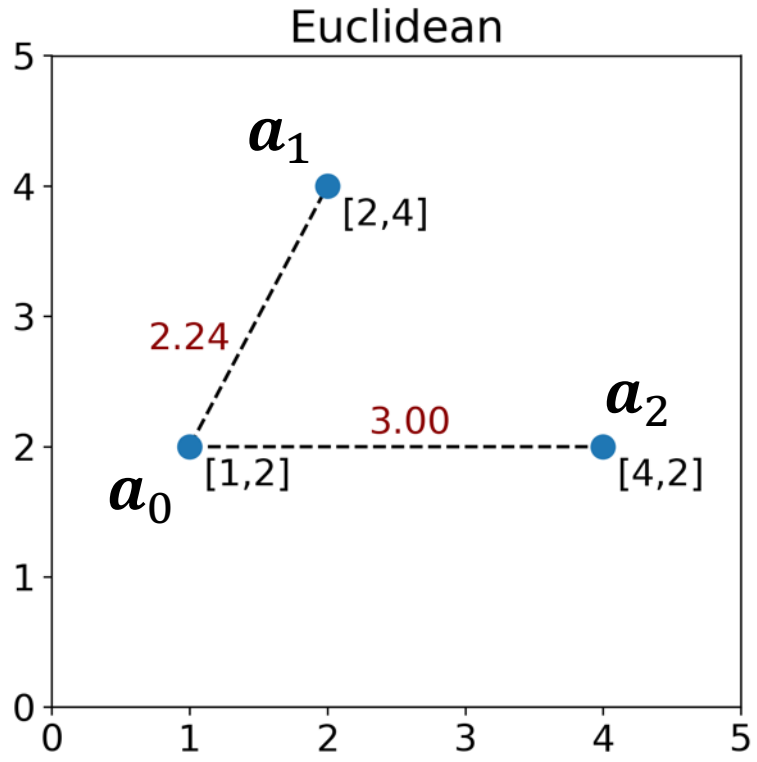
$$d_{manhattan}(a_0, a_1) = |a_{0,1} - a_{1,1}|$$

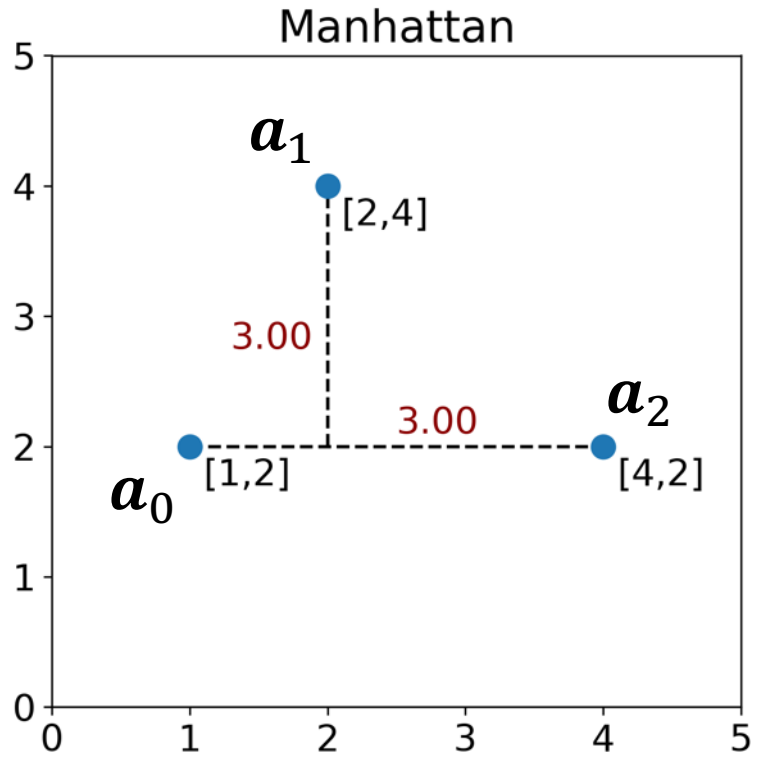
$$= |a_{0,0} - a_{1,0}| + |a_{0,1} - a_{1,1}|$$

$$= |1 - 2| + |2 - 4|$$

$$= 3$$

# How far away is each point from $a_0$?

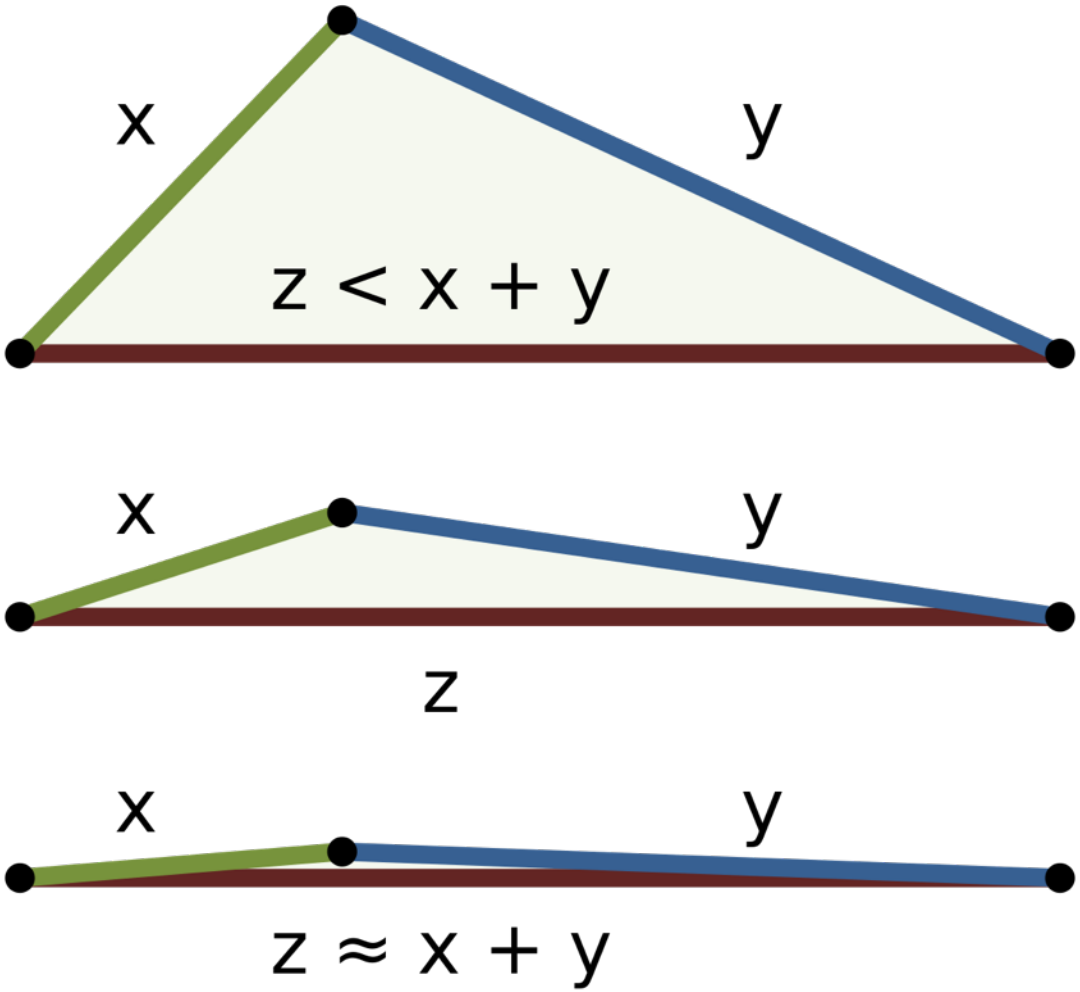


# Distances

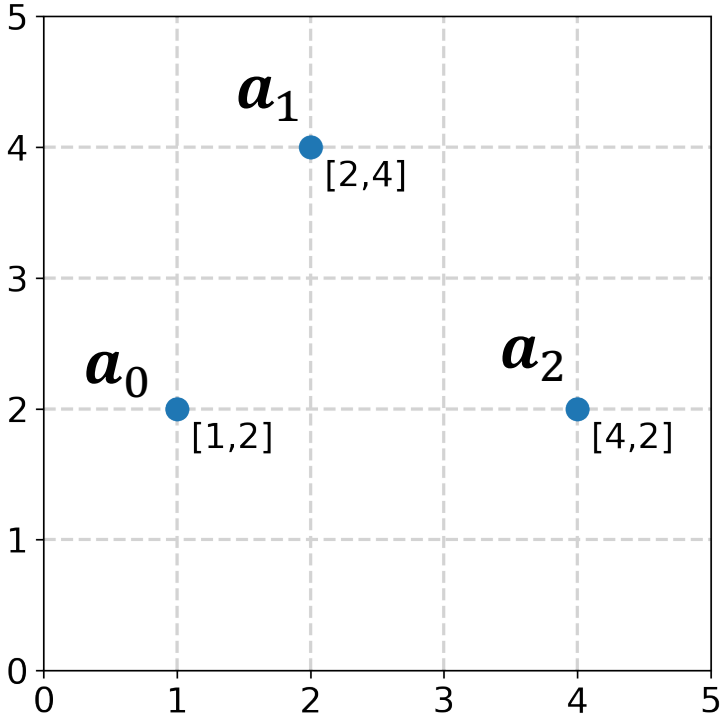
- There are many ways to measure distance
  - Hamming, Euclidean, Cosine
- Distances are functions that take two points and return a real number that is **positive or 0**
- Distances can be any function that is:
  - **Symmetric:** dist(a → b) = dist(b → a)
  - **Non-negative:** dist(a→b)>= 0
  - **Follow triangle inequality:** dist(a→c) <= dist(a→b) + dist(b→c)

# Triangle Inequality

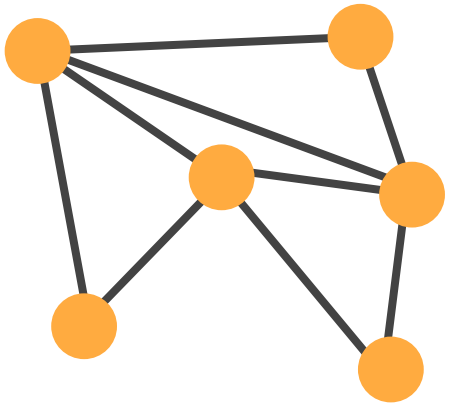


# Representing Data as a Graph

Plot




Graph

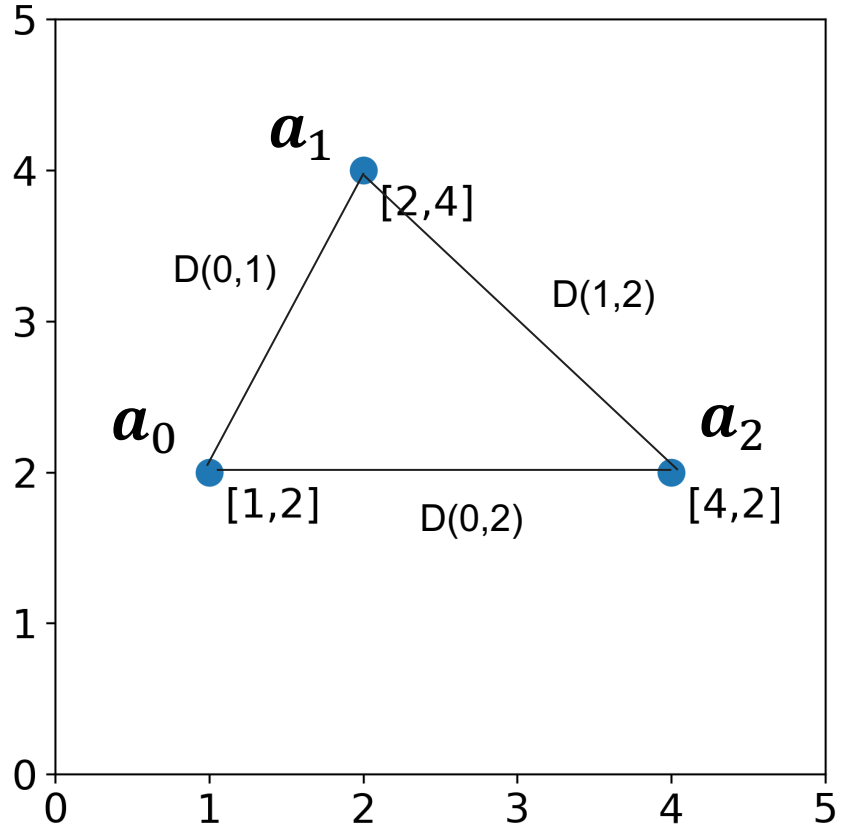


$G = (V, E)$

$= V =$ vertices / nodes

$= E =$ edges

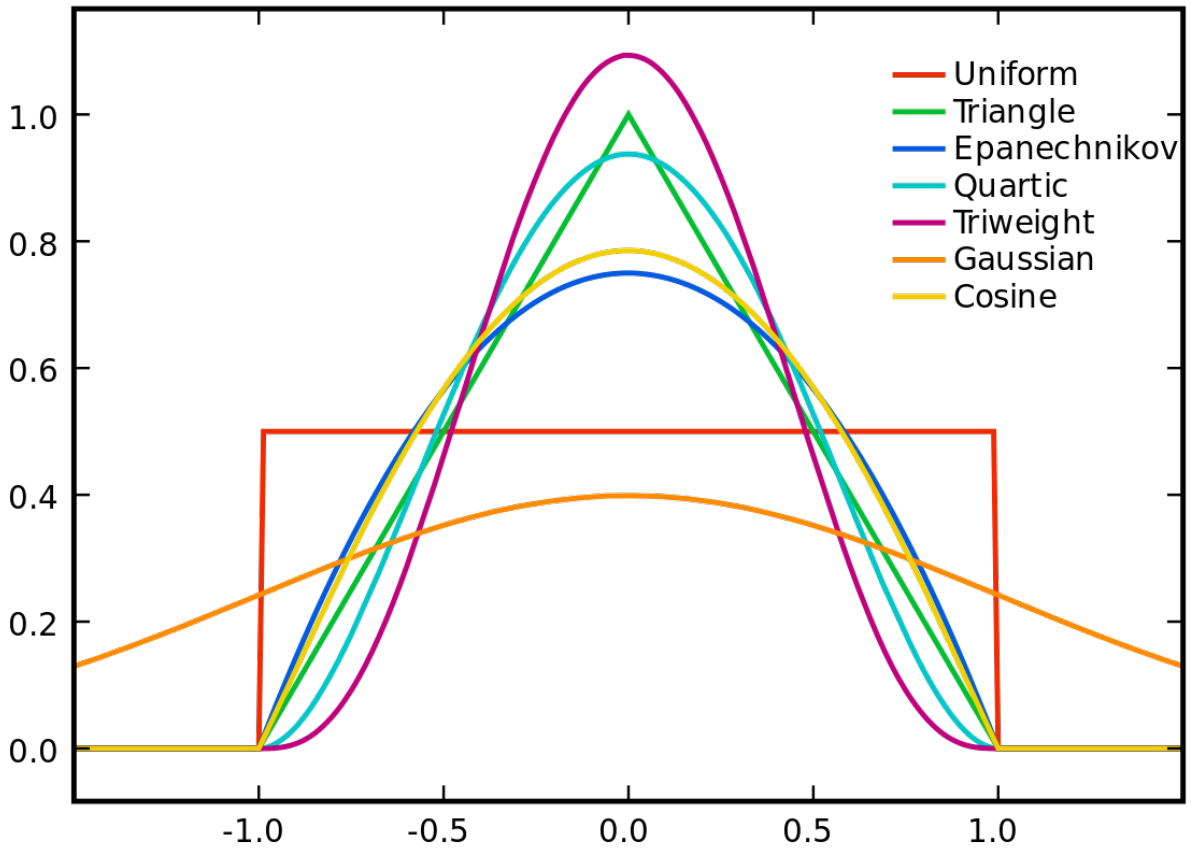
# Fully connected Graph




| | $a_0$ | $a_1$ | $a_2$ |
|---|---|---|---|
| $a_0$ | 0 | $\sqrt{5}$ | 3 |
| $a_1$ | $\sqrt{5}$ | 0 | $\sqrt{8}$ |
| $a_2$ | 3 | $\sqrt{8}$ | 0 |

Distance matrix

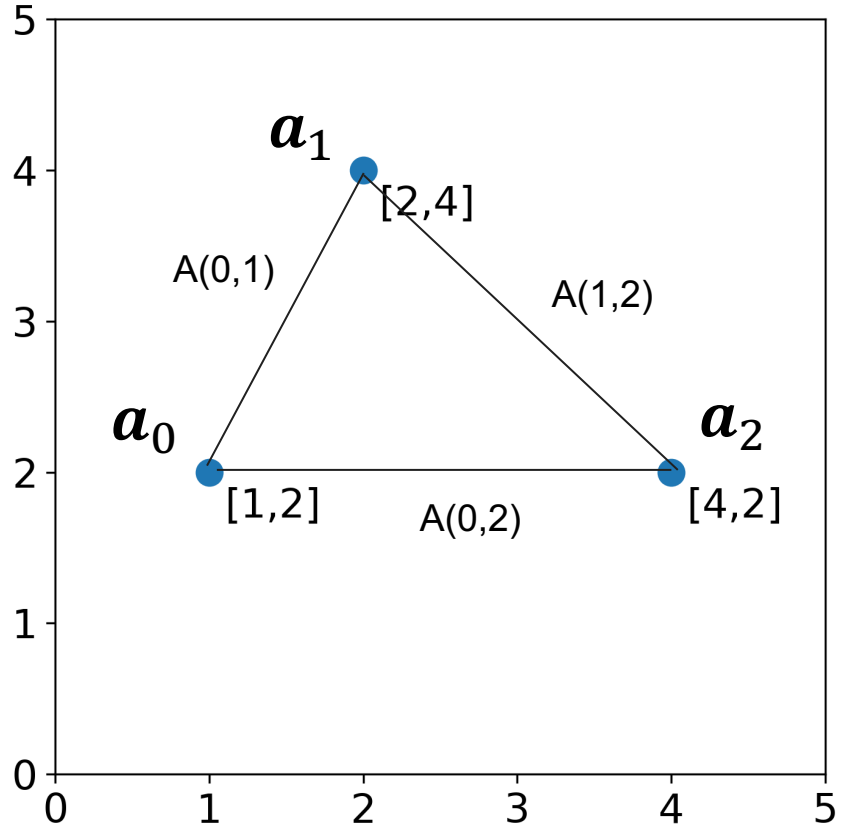
# Distance to Affinity via Kernels




Real-valued
Non-negative
Symmetric

Affinities correlations in this hidden hypothetical space

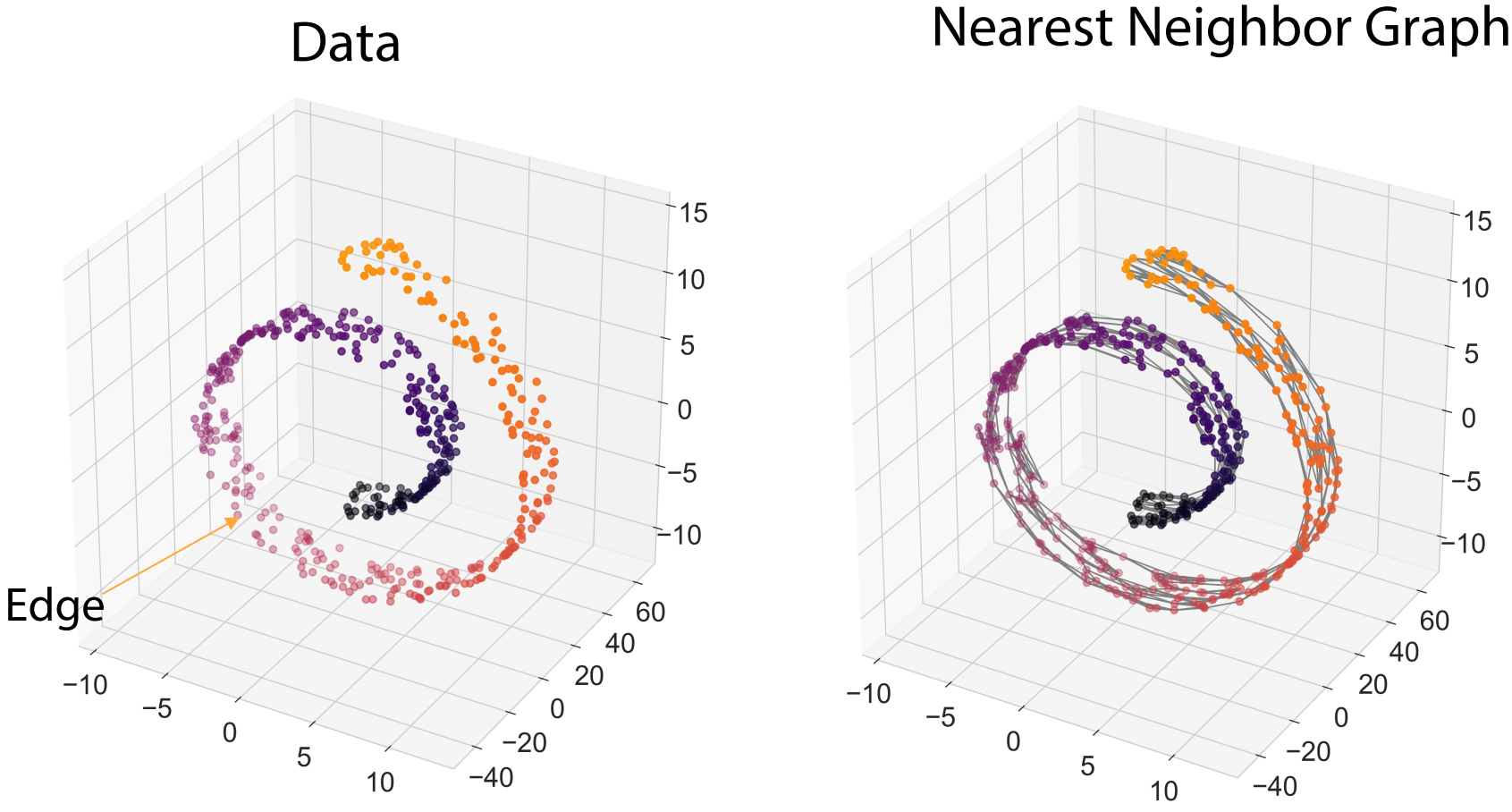
# Affinities



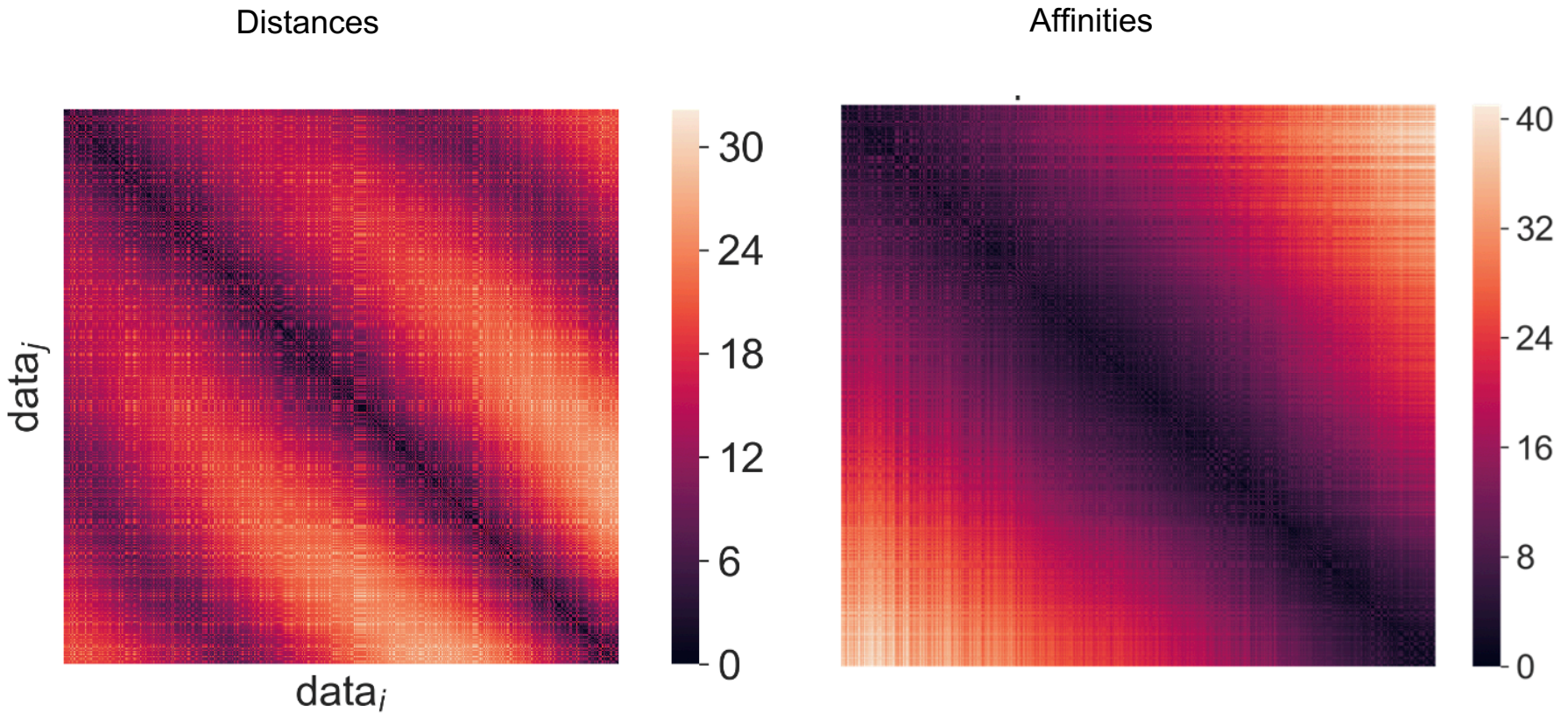

| | $a_0$ | $a_1$ | $a_2$ |
|---|---|---|---|
| $a_0$ | 1 | 0.032 | 0.0044 |
| $a_1$ | 0.032 | 1 | 0.0075 |
| $a_2$ | 0.0044 | 0.0075 | 1 |

Affinity or Adjacency Matrix

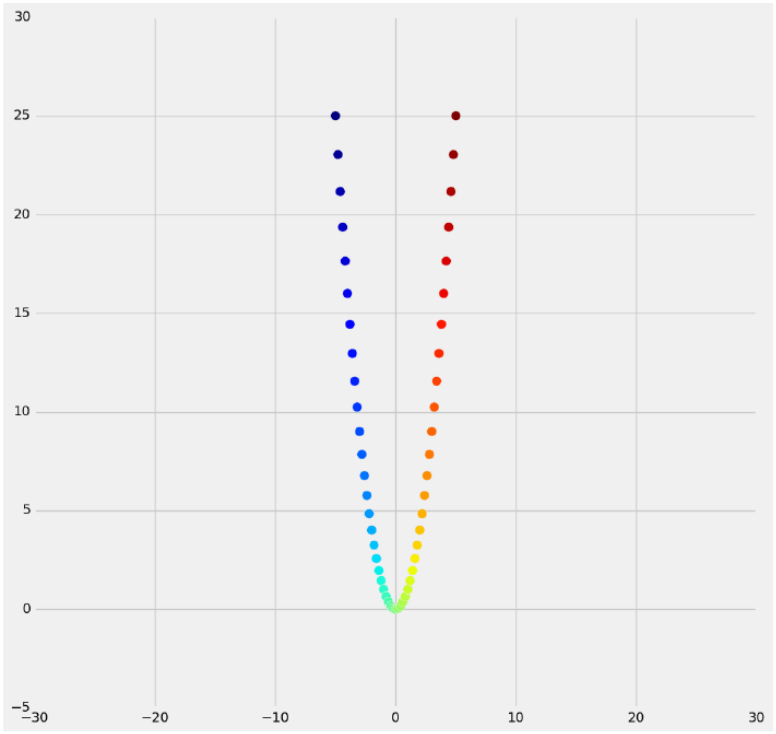
# Nearest Neighbors Graph

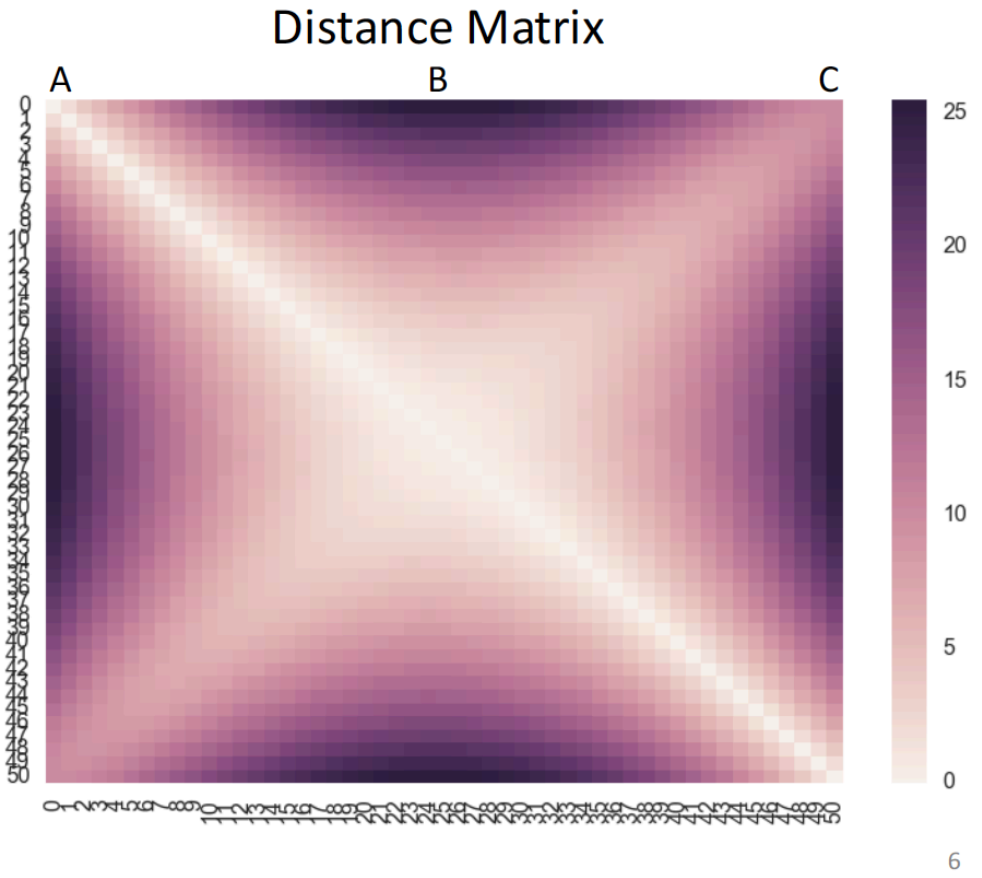


# Swiss Roll

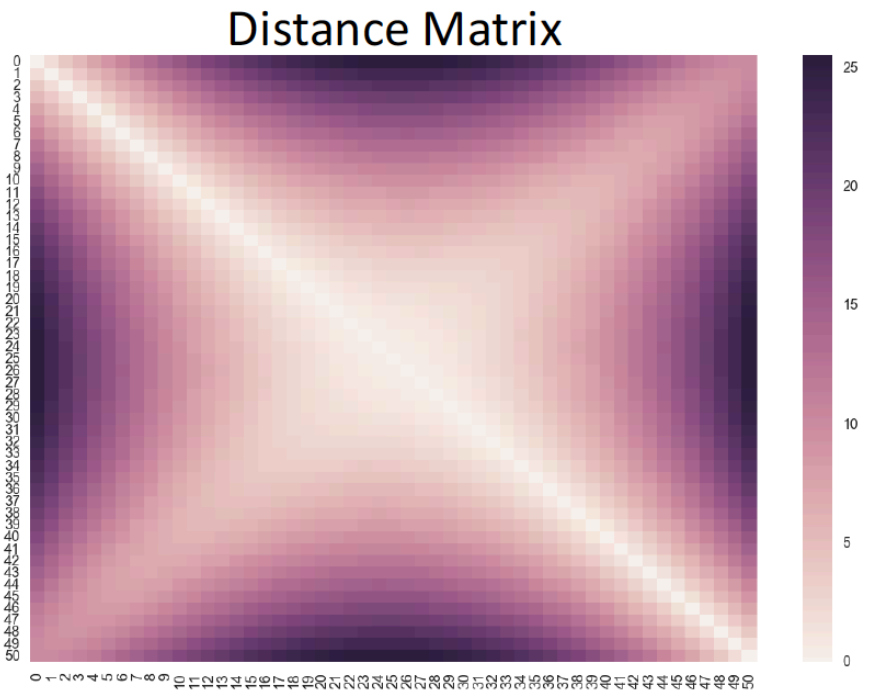


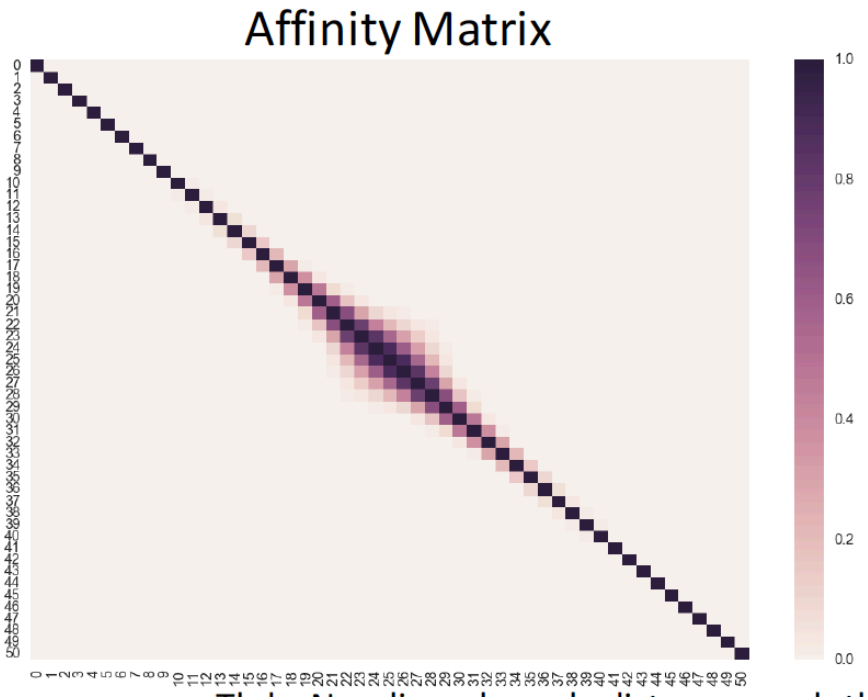
# Example


30
25
20
15
10
5
0
−5
−30
−20
−10
0
10
20
30

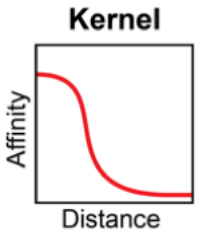

Distance Matrix
A
B
C
25
20
15
10
5
0

# Affinity is the inverse proportional to distance

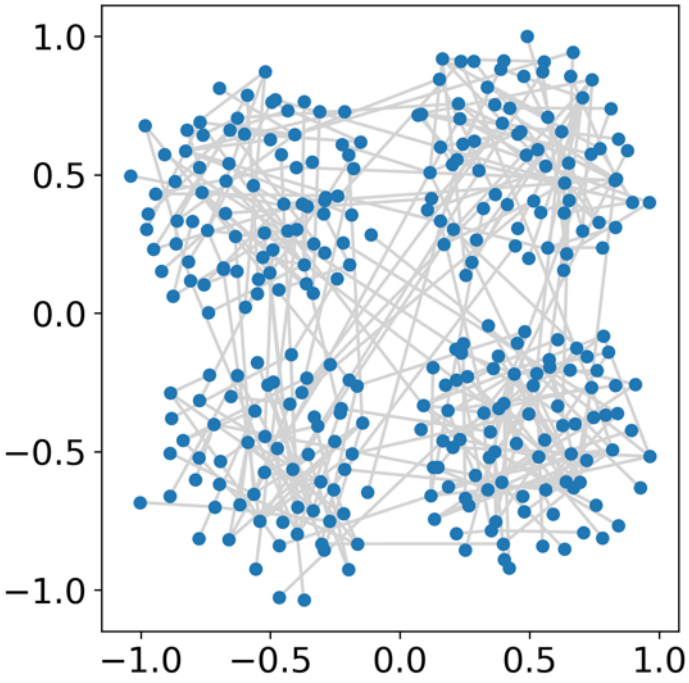



$$Affinity_{i,j} = s_{i,j} = exp\left(-\frac{dist(x_i, x_j)^2}{2\sigma^2}\right)$$



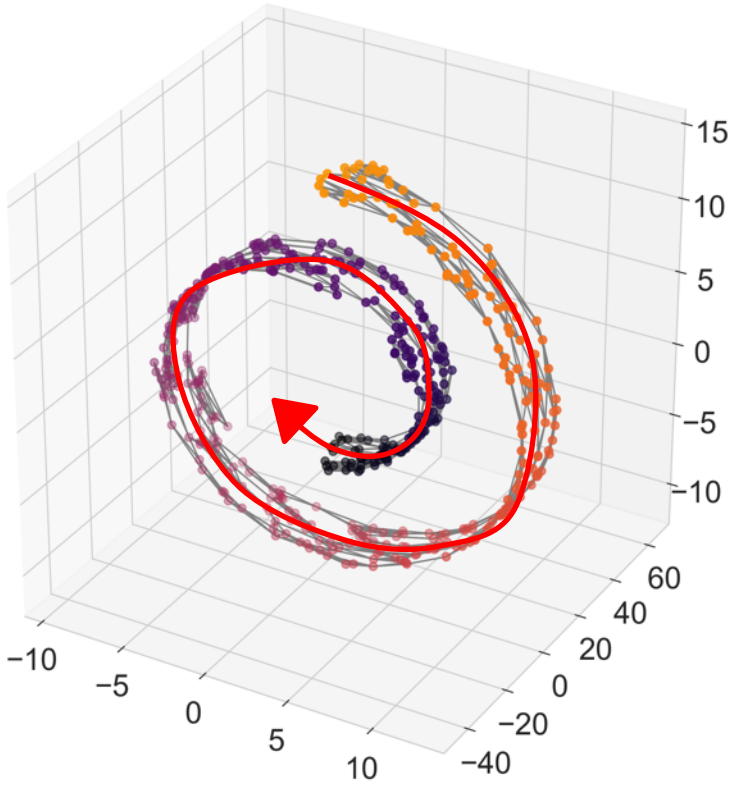
# Why Represent Data as a Graph?




Graphs can be easy to cluster---look for a minimal way of “cutting edges” to form groups

Graphs avoid needing to operate in high dimensions they can just be represented as list of edges

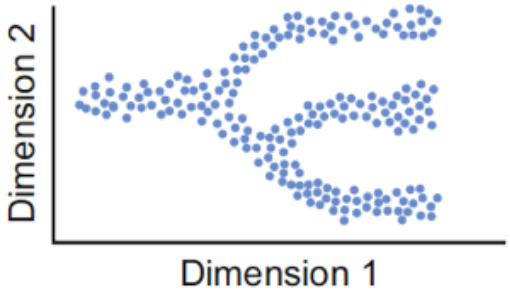
# Why Represent Data as a Graph?



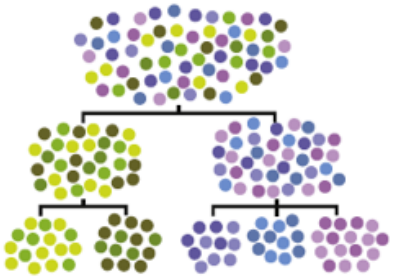

Paths through data graphs can represent progression trajectories

# And much more to come!!

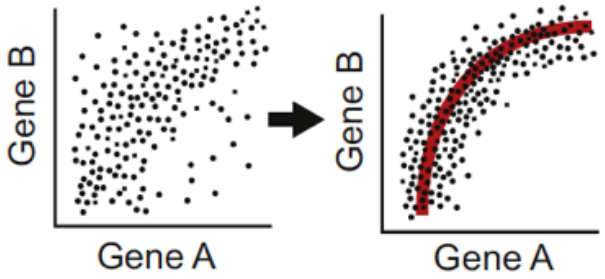
## Vizualization



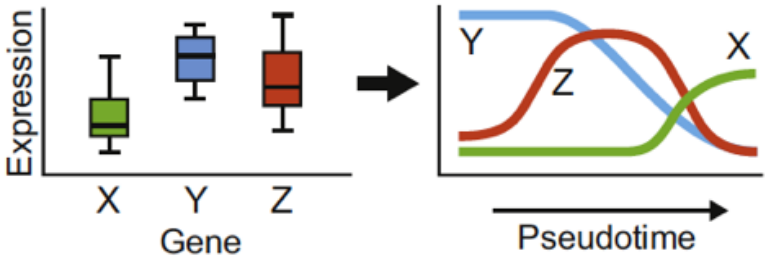

## Clustering



## Denoising




## Pseudotime analysis



# Preprocessing single-cell data

*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.

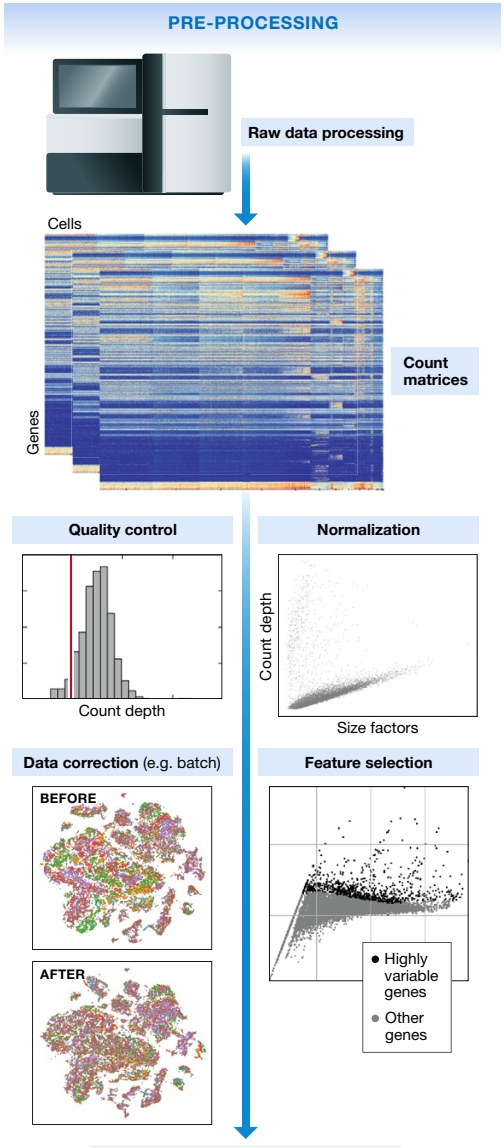
Further challenges for standardization lie in technical aspects. Analysis tools for scRNA-seq data are written in a variety of programming languages—most prominently R and Python (Zappia *et al*, 2018). Although cross-environment support is growing (preprint: Scholz *et al*, 2018), the choice of programming 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://doi.org/10.15252/msb.20188746

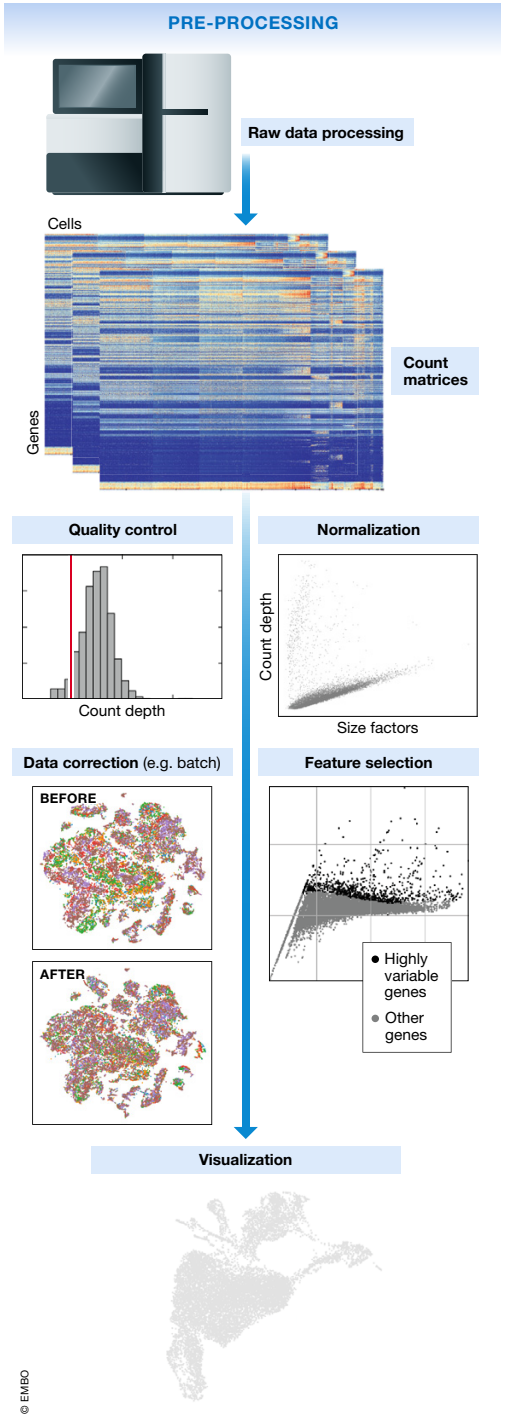
# Standard Single-Cell RNA-seq Workflow

1. Sequencing and read mapping
2. Quality control and filtering
3. Normalization
4. Data Correction



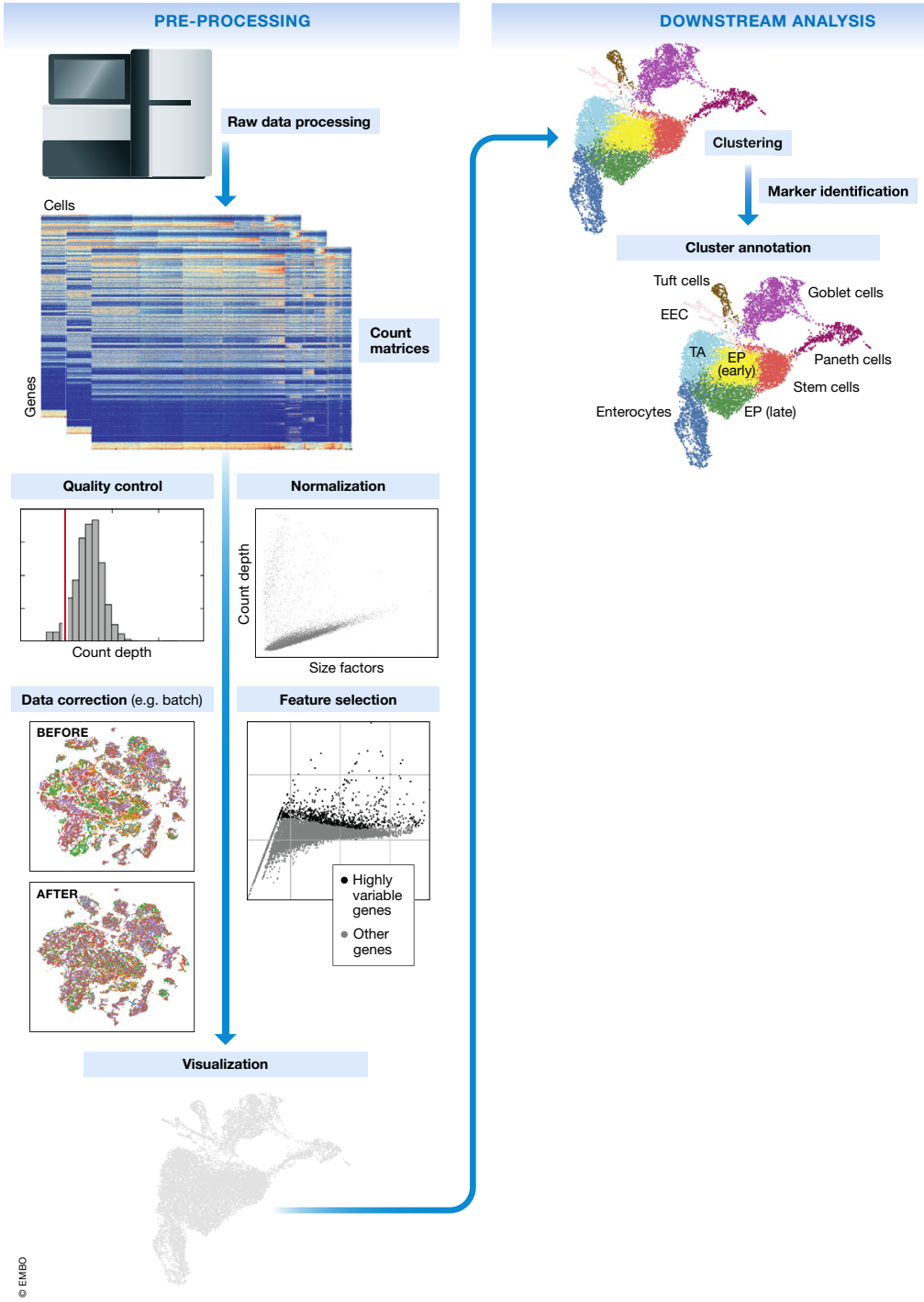
# Standard Single-Cell RNA-seq Workflow

1. Sequencing and read mapping
2. Quality control and filtering
3. Normalization
4. Data Correction
5. Dimensionality reduction and visualization



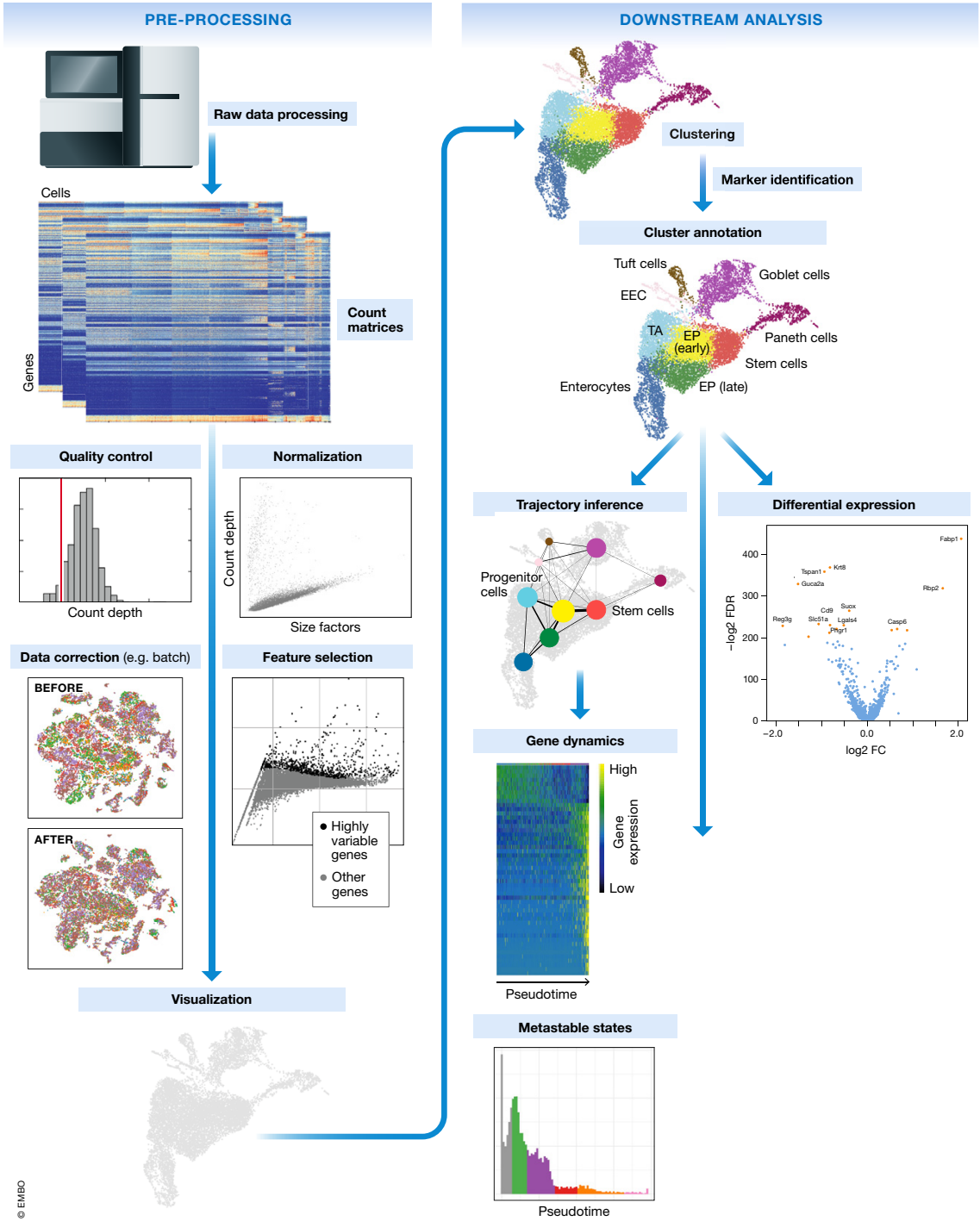
# Standard Single-Cell RNA-seq Workflow

1. Sequencing and read mapping
2. Quality control and filtering
3. Normalization
4. Data Correction
5. Dimensionality reduction and visualization
6. Downstream analysis
    1. Clustering
    2. Trajectory inference



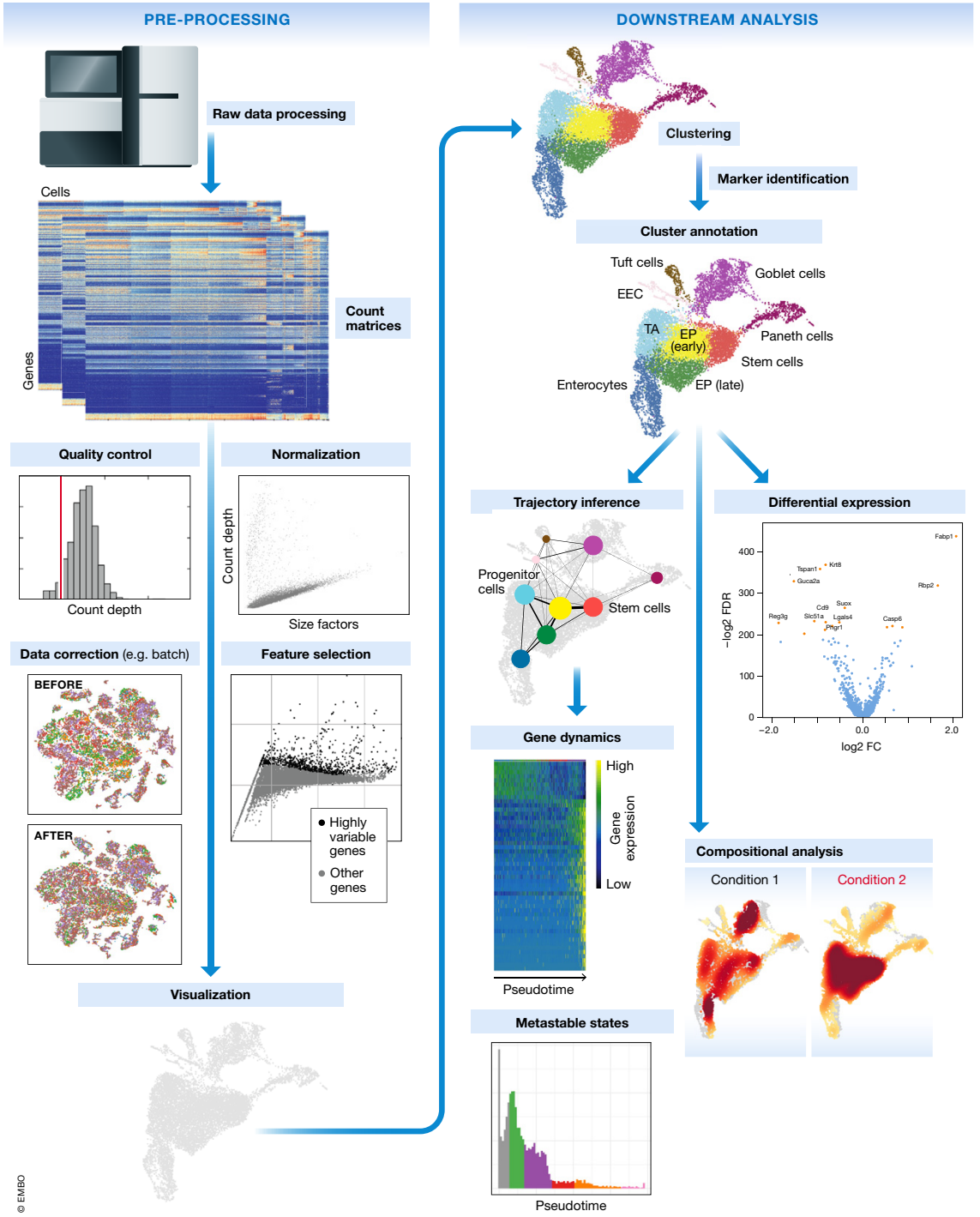
# Standard Single-Cell RNA-seq Workflow

1. Sequencing and read mapping
2. Quality control and filtering
3. Normalization
4. Data Correction
5. Dimensionality reduction and visualization
6. Downstream analysis
    1. Clustering
    2. Trajectory inference
    3. Differential expression

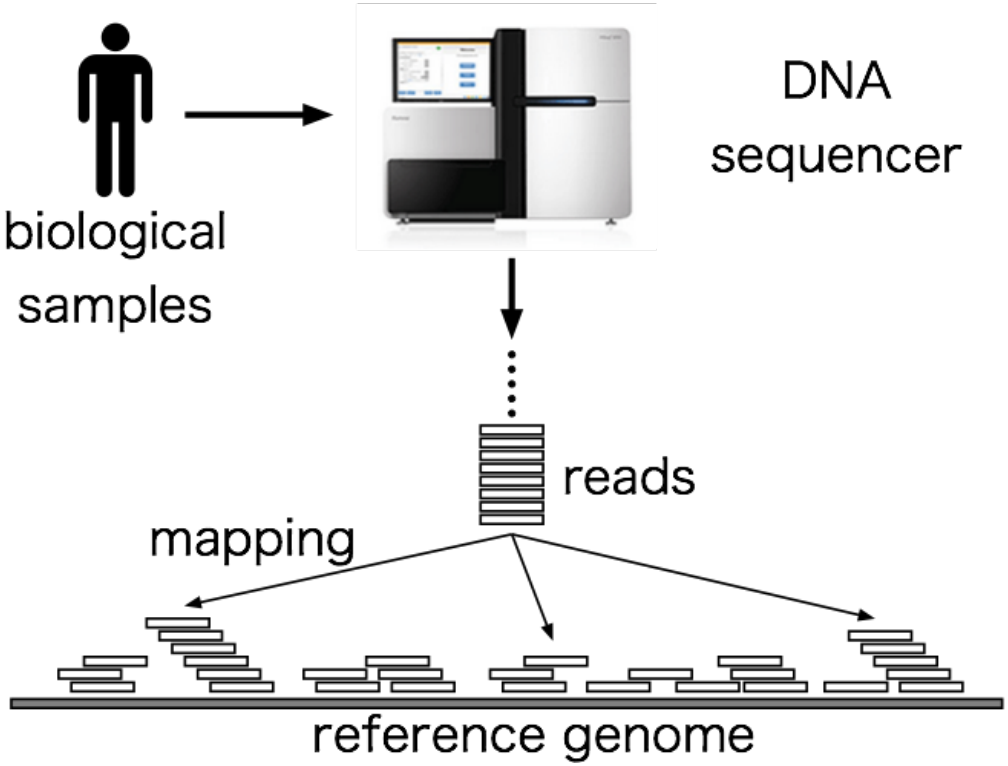


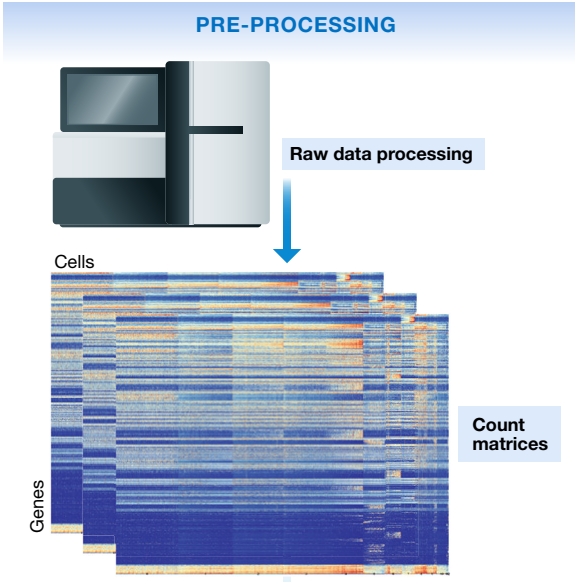
# Standard Single-Cell RNA-seq Workflow

1. Sequencing and read mapping
2. Quality control and filtering
3. Normalization
4. Data Correction
5. Dimensionality reduction and visualization
6. Downstream analysis
    1. Clustering
    2. Trajectory inference
    3. Differential expression
7. Comparison of multiple conditions



# Step 1 - Sequencing & read mapping

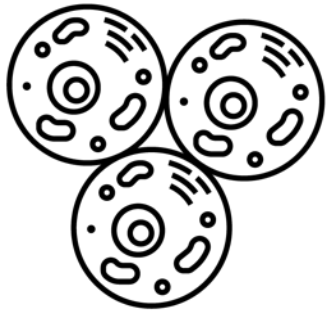


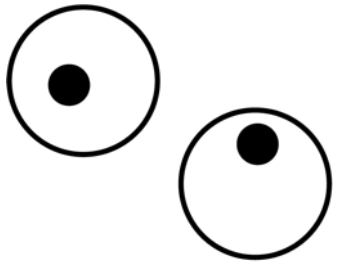
# Step 2 – Quality control and filtering

## Dying cells

## Multiplets



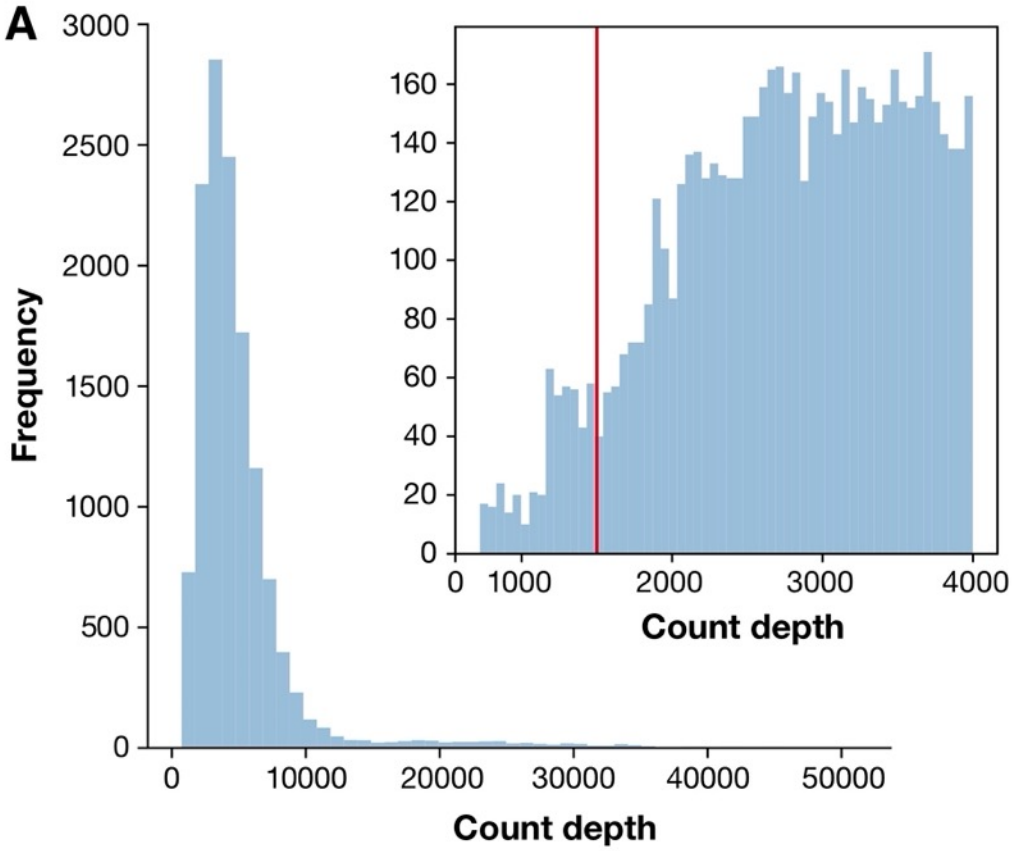
## Empty Droplets



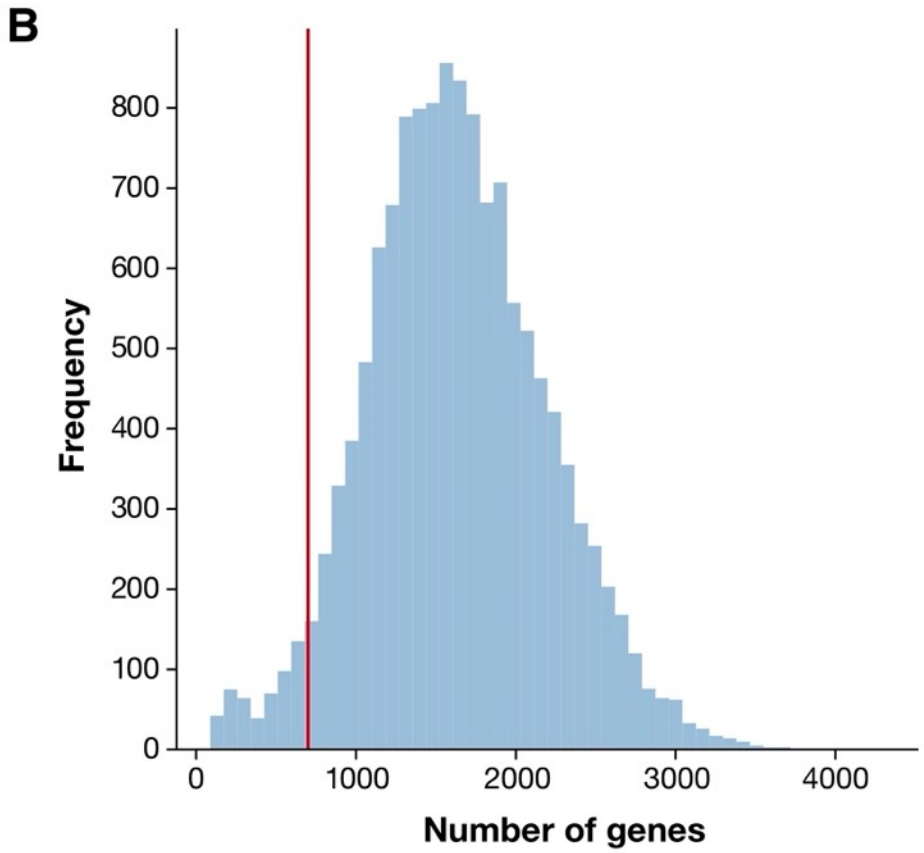
# What could we look at to discriminate between dying cells, multiplets, or empty droplets and healthy single cells?

**Top**

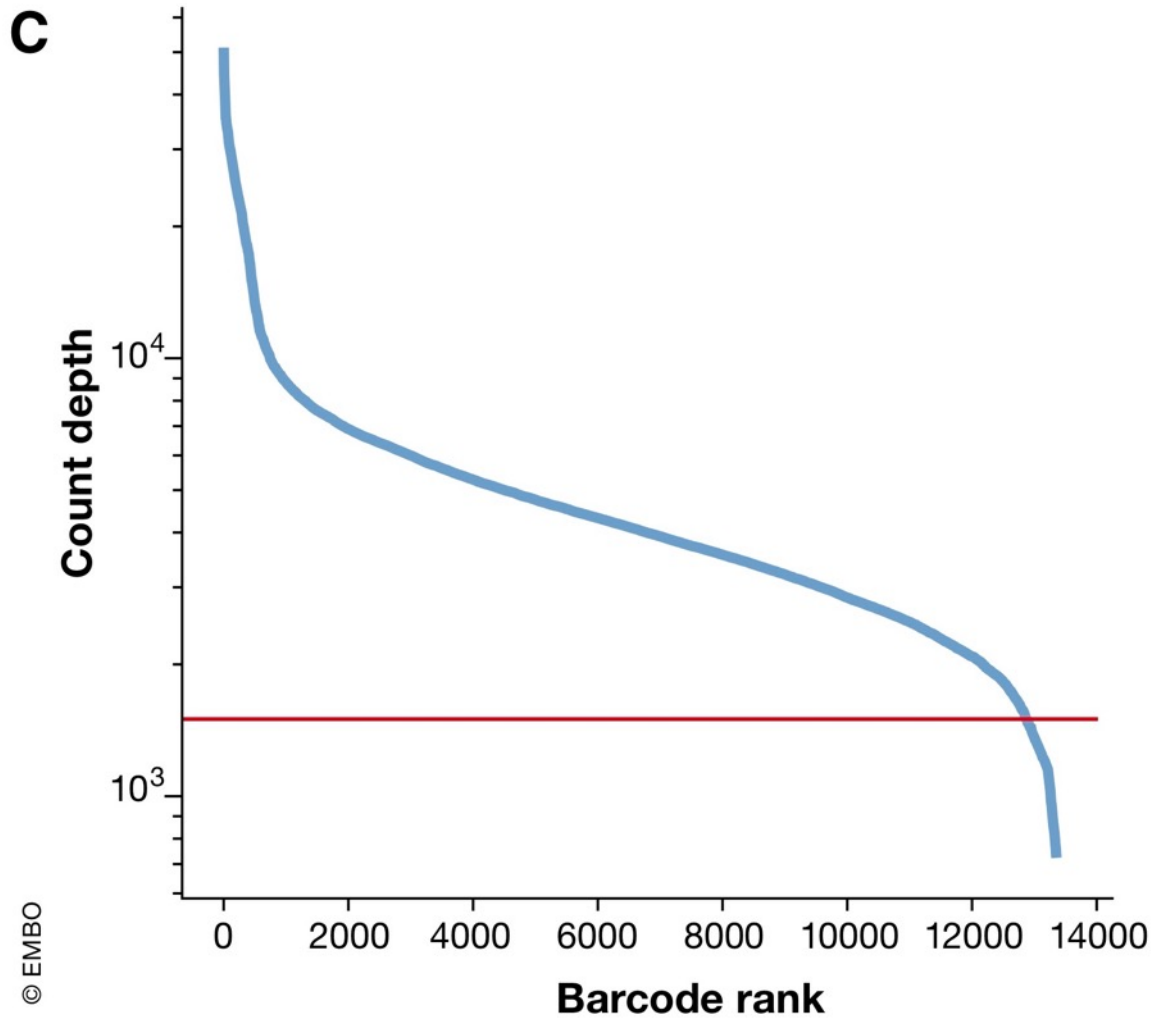
# Step 2 – Quality control and filtering



# Step 2 – Quality control and filtering

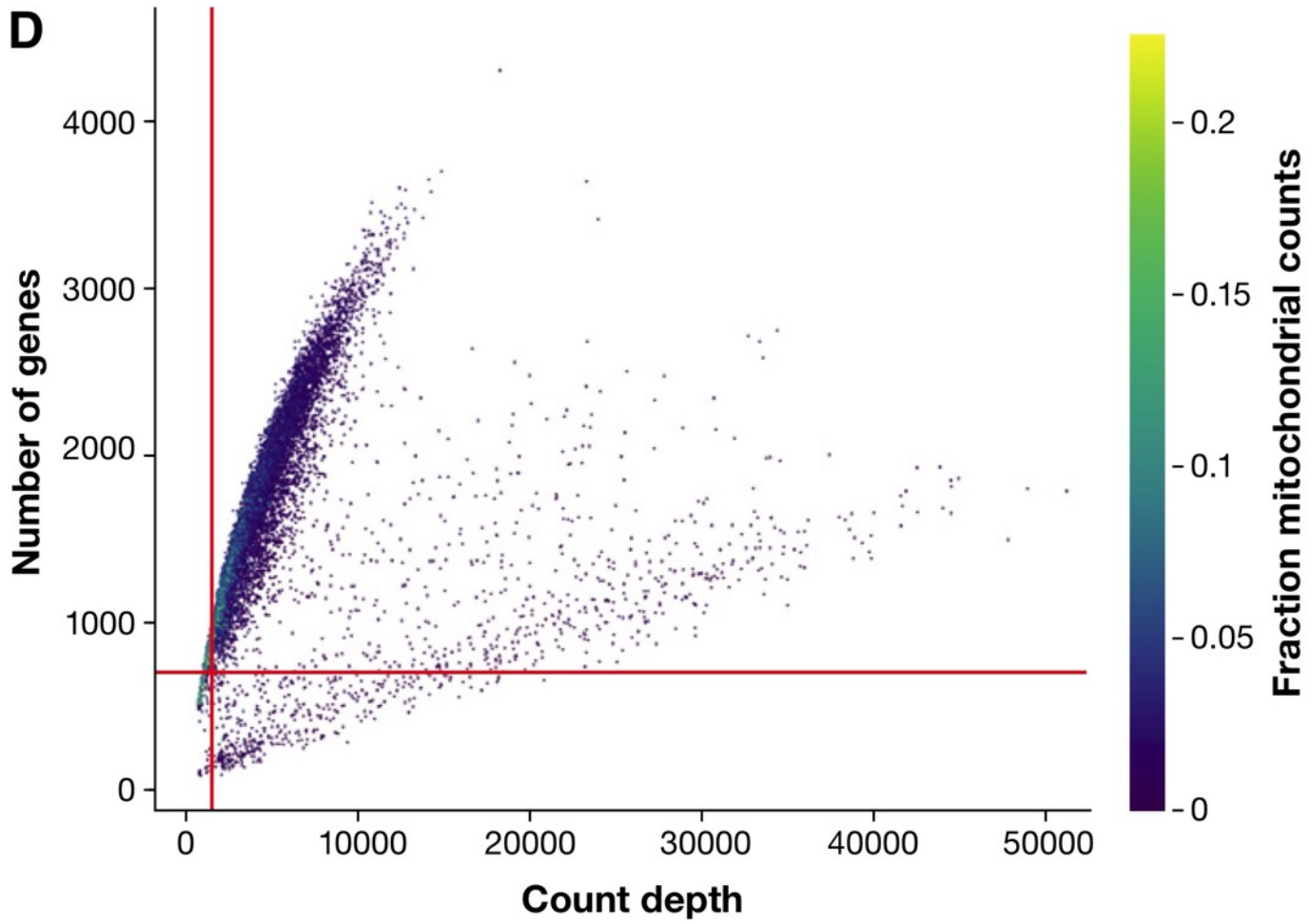


# Step 2 – Quality control and filtering

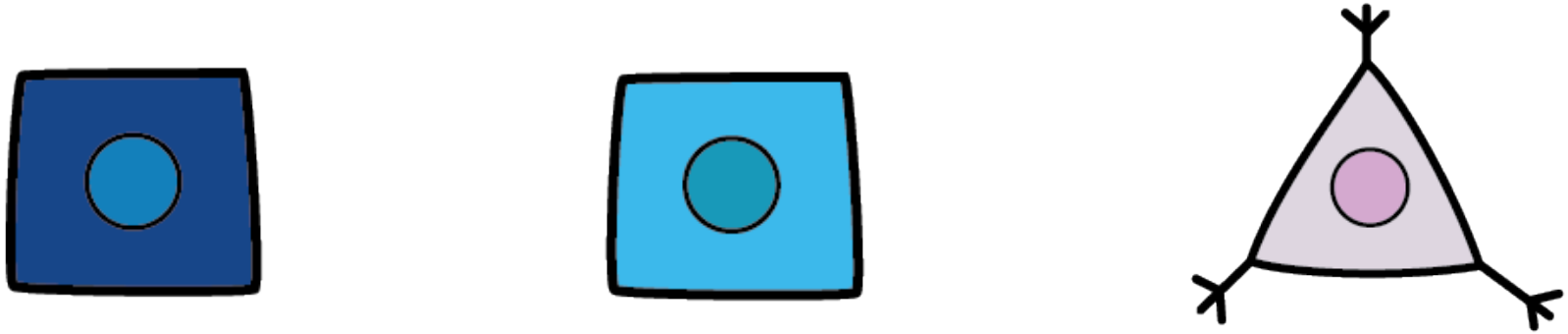



Building Counts Matrix → QC & Filtering → Normalization → Visualization → Analysis

# Step 2 – Quality control and filtering

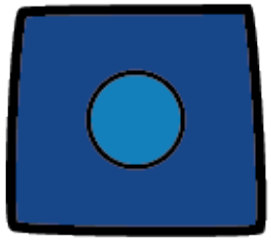


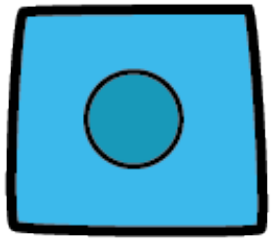
# Step 3 - Normalization

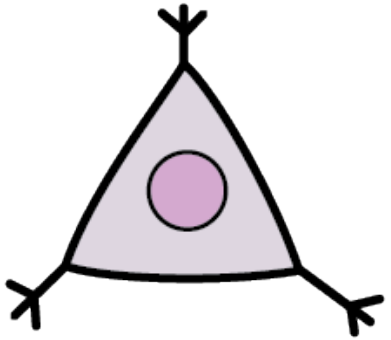


**If we only have gene expression, how can we determine which cells are similar?**

# Step 3 - Normalization



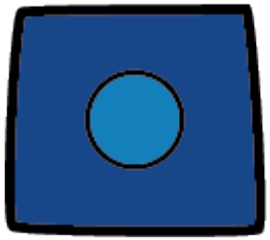


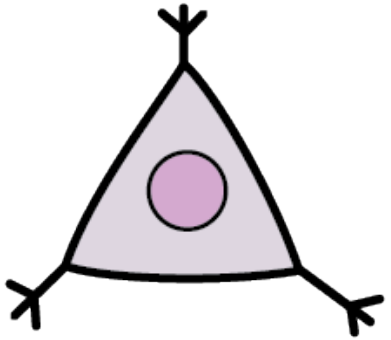


10% Capture Efficiency

| Gene | Cell A |
|---|---|
| X | 10 |
| Y | 20 |
| Z | 70 |

# Step 3 - Normalization





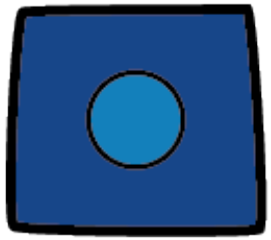
10% Capture Efficiency

| Gene | Cell A |
|---|---|
| **X** | 10 |
| **Y** | 20 |
| **Z** | 70 |

20% CE

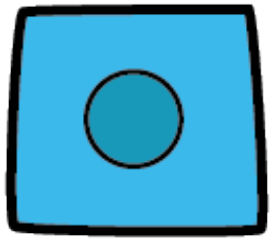
| Gene | Cell B |
|---|---|
| **X** | 20 |
| **Y** | 40 |
| **Z** | 140 |

# Step 3 - Normalization



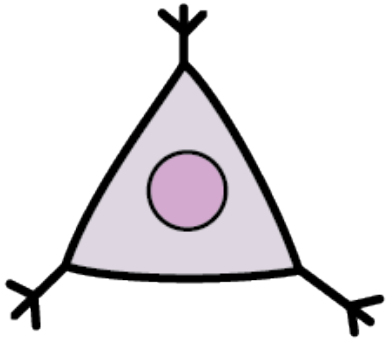
10% Capture Efficiency

| Gene | Cell A |
|---|---|
| X | 10 |
| Y | 20 |
| Z | 70 |



20% CE

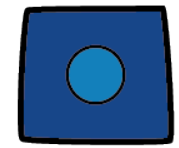
| Gene | Cell B |
|---|---|
| X | 20 |
| Y | 40 |
| Z | 140 |

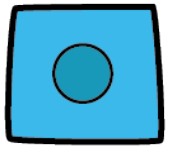


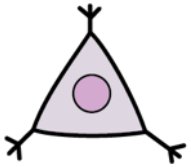
20% CE

| Gene | Cell C |
|---|---|
| X | 20 |
| Y | 0 |
| Z | 80 |

# Step 3 - Normalization





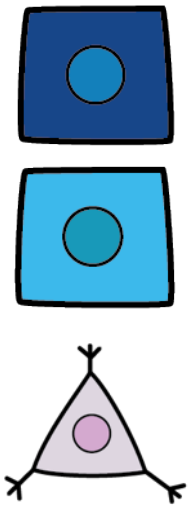


**Raw counts**

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

# Step 3 - Normalization

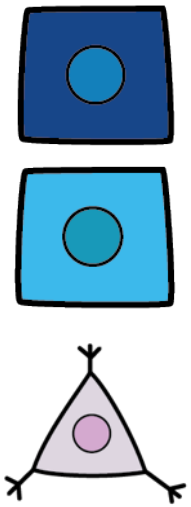
**Raw counts**



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

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

# Step 3 - Normalization



**Raw counts**

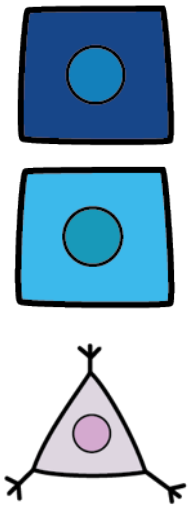
| | X | Y | Z |
|---|---|---|---|
| **A** | 10 | 20 | 70 |
| **B** | 20 | 40 | 140 |
| **C** | 20 | 0 | 80 |

**Pairwise distances**

dist(A,B) = 71.4

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

# Step 3 - Normalization



**Raw counts**

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

**Pairwise distances**

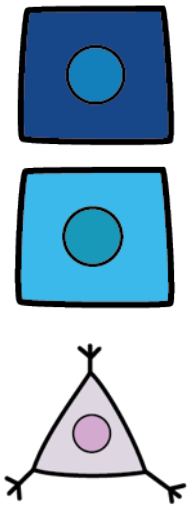
dist(A,B) = 71.4

dist(A,C) = 24.5

dist(B,C) = 67.1

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

# Step 3 - Normalization



**Raw counts**

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

**Pairwise distances**

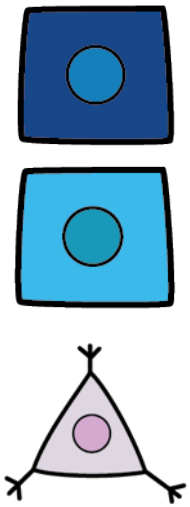
dist(A,B) = 71.4

dist(A,C) = 24.5

dist(B,C) = 67.1

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

# Step 3 - Normalization



**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**

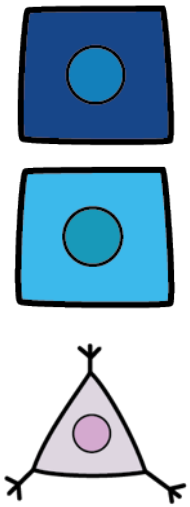
dist(A,B) = 71.4

dist(A,C) = 24.5

dist(B,C) = 67.1

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

# Step 3 - Normalization



**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

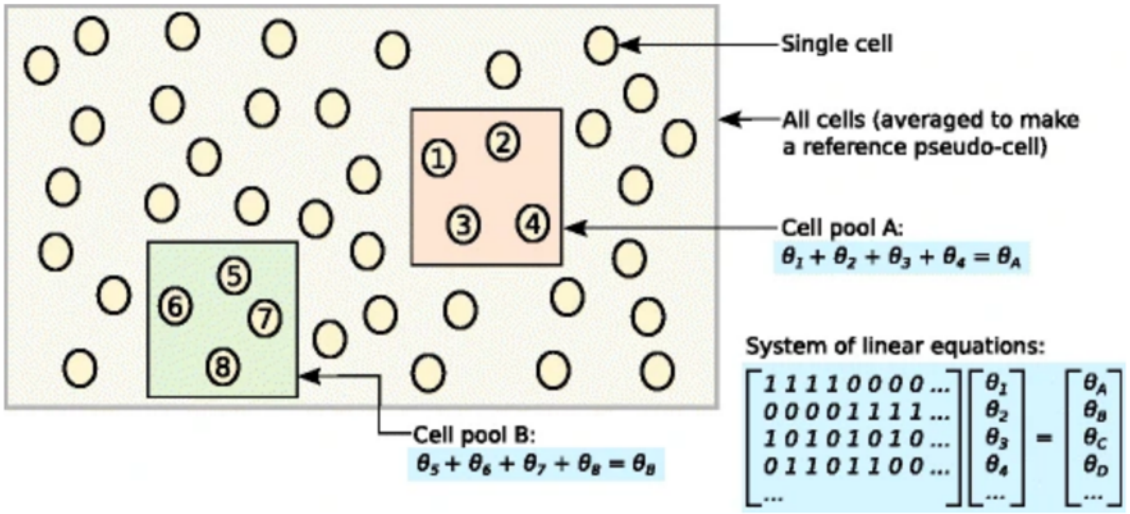
dist(A,C) = 0.25

dist(B,C) = 0.25

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

# More complex normalization approaches exist

**Fig. 3**

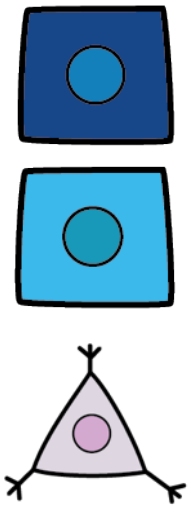



Schematic of the deconvolution method. All cells in the data set are averaged to make a reference pseudo-cell. Expression values for cells in pool A are summed together and normalized against the reference to yield a pool-based size factor $\theta_A$. This is equal to the sum of the cell-based factors $\theta_j$ for cells $j$=1–4 and can be used to formulate a linear equation. (For simplicity, the $t_j$ term is assumed to be unity here.) Repeating this for multiple pools (e.g., pool B) leads to the construction of a linear system that can be solved to estimate $\theta_j$ for each cell $j$

https://doi.org/10.1186/s13059-016-0947-7

# Step 3.5 – Transformation / Scaling

**Normalized counts**



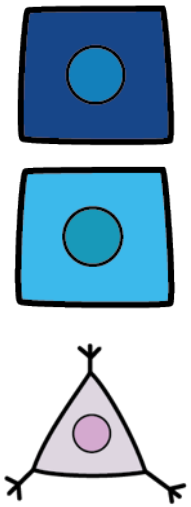
| | X | Y | Z |
|---|---|---|---|
| **A** | 0.1 | 0.2 | 0.7 |
| **B** | 0.1 | 0.2 | 0.7 |
| **C** | 0.2 | 0 | 0.8 |

What if this is a housekeeping gene?

What if this is a transcription factor?

# Step 3.5 – Transformation / Scaling

**sqrt(normalized counts)**



| | X | Y | Z |
|---|---|---|---|
| **A** | 0.32 | 0.45 | 0.84 |
| **B** | 0.32 | 0.45 | 0.89 |
| **C** | 0.2 | 0 | 0.89 |

What if this is a housekeeping gene?

What if this is a transcription factor?

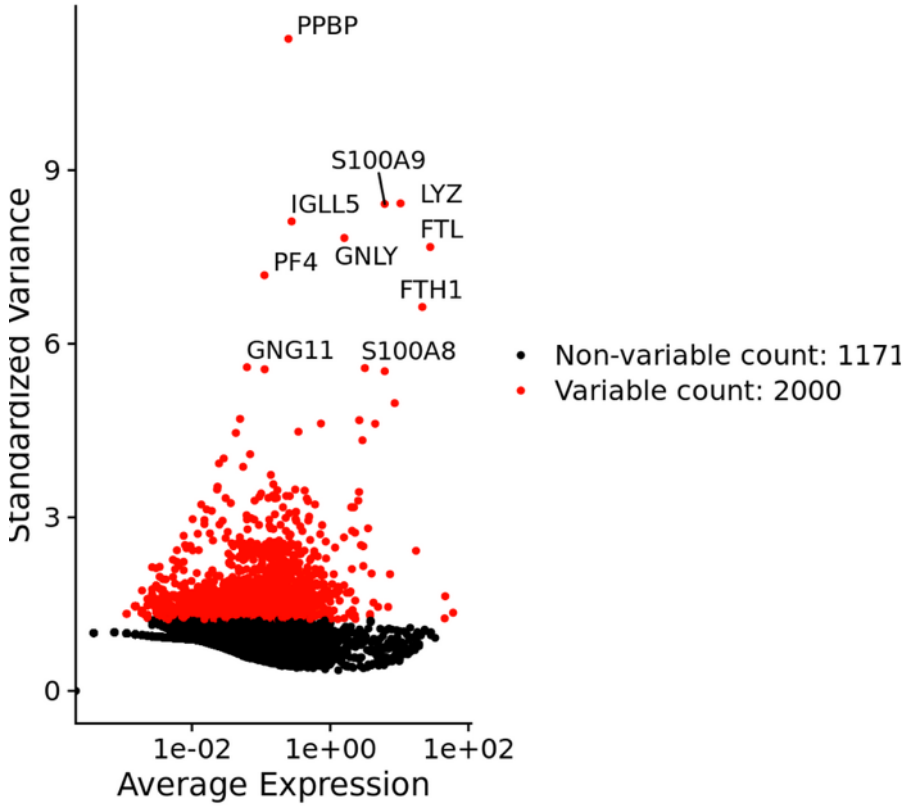
# What kind of transformations, other than square-root, could we apply to single cell data?

**Top**

# Step 5 – Dimensionality reduction and visualization

Selecting highly variable genes (HVGs):

- Calculate log10 mean expression and variance
- Fit a loess curve
- Standardize variance to mean 0 std 1
- Take the top 2000 HVGs

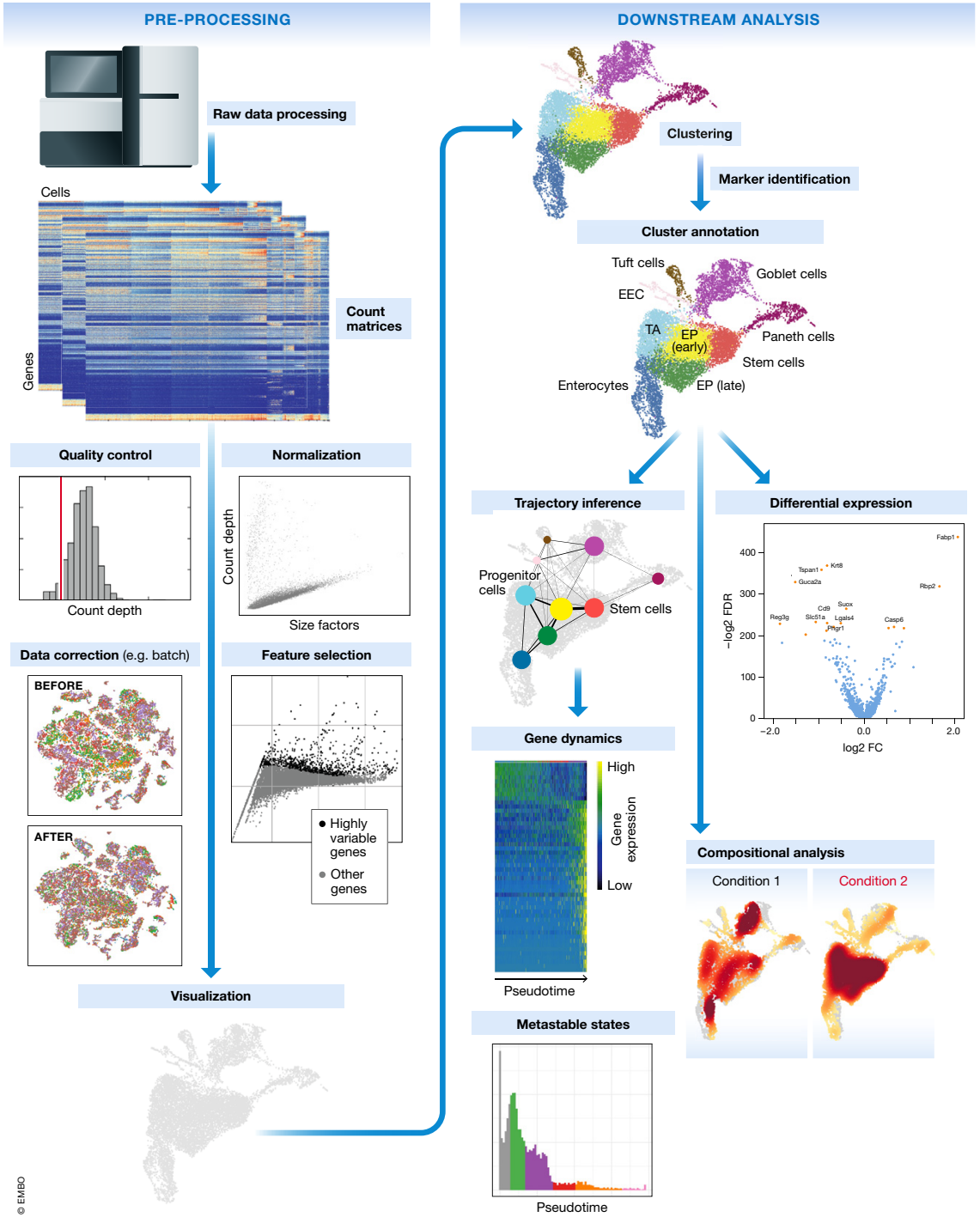



Building Counts Matrix → QC & Filtering → Normalization → Visualization → Analysis

https://satijalab.org/seurat/v3.1/pbmc3k_tutorial.htm

# Standard Single-Cell RNA-seq Workflow

1. Sequencing and read mapping
2. Quality control and filtering
3. Normalization
4. Data Correction
5. Dimensionality reduction and visualization
6. Downstream analysis
    1. Clustering
    2. Trajectory inference
    3. Differential expression
7. Comparison of multiple conditions



# What questions do you have about today's material?

**Top**

# Exercise!

Load, preprocess, and visualize a scRNAseq dataset generated from a time course of embryoid bodies