The Krishnaswamy Laboratory

Yale Genetics and Yale SEAS present

Machine Learning

for

Single Cell Analysis

Online - May 20-29, 2020

When poll is active, respond at **PollEv.com/yaleml**

Text **YALEML** to **22333** once to join

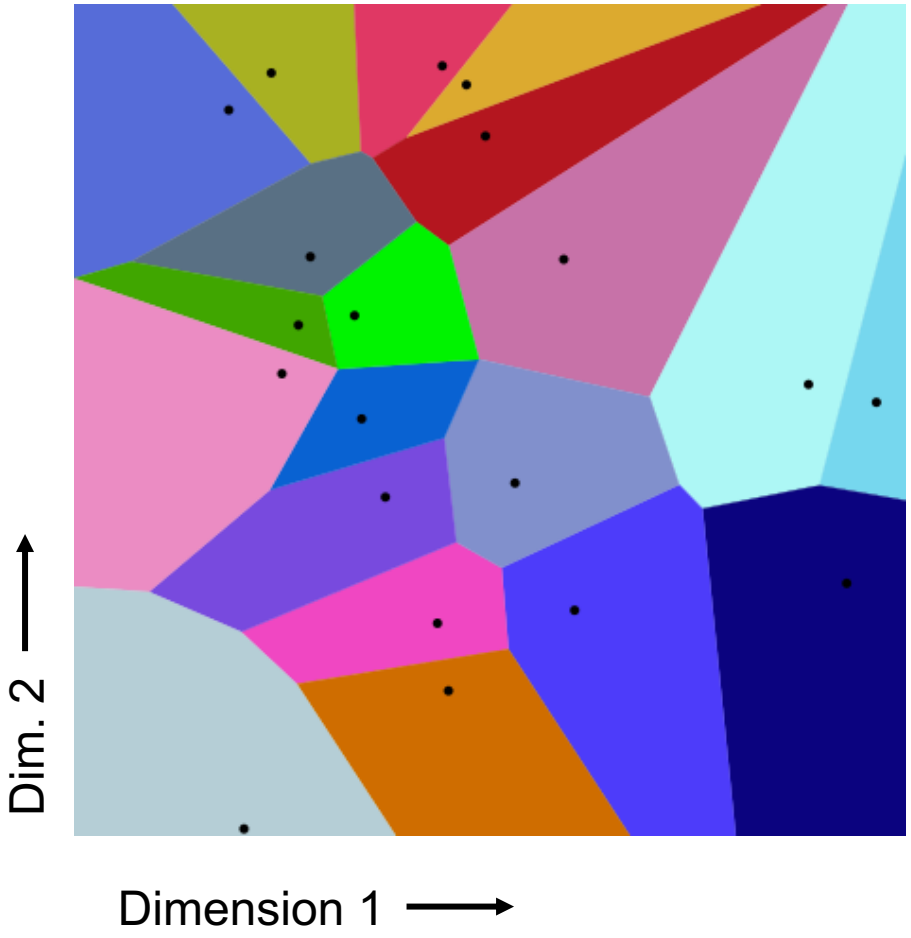
Where will you travel for your first trip post-quarantine? (use underscores for multi-word answers)

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Learning Clusters, Trajectories, and Gene Relationships from scRNA-seq data

How do clustering algorithms work?

Clustering algorithms partition the data space



K-MEANS (Macloed 1967), Fidduccia Matheyses

Voronoi Diagram

How is the partition picked?

- By minimizing different objective criteria
  - Closeness of members *within* the group
  - Distance/Separation between groups
  - Ratio of the two
  - Minimum cut of an NN-graph
- Other criteria?
- Modularity: actual edges/ expected edges

Mean Squared Error

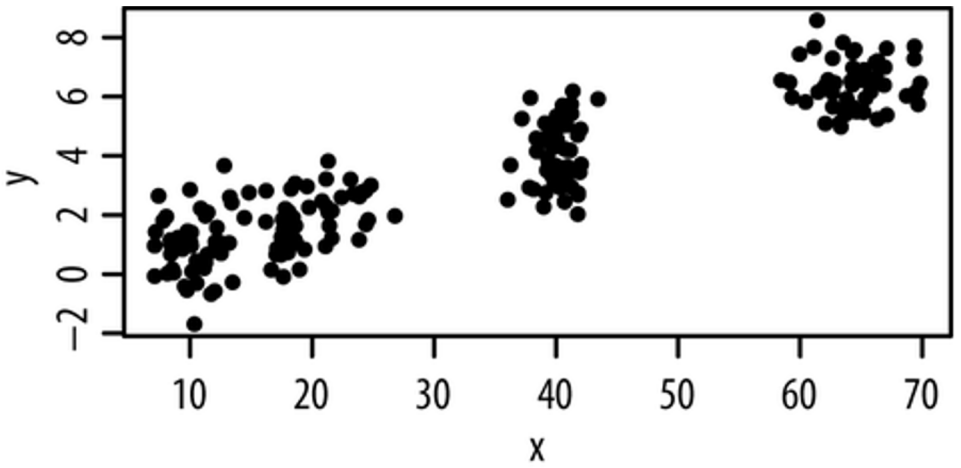
- Given data $X = \{x_1, x_2, ...x_n\}$
- Partition into k clusters
- Such that the within-cluster variance is minimized

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

"The sum of all squared euclidean distances between each point and that point's closest cluster mean"

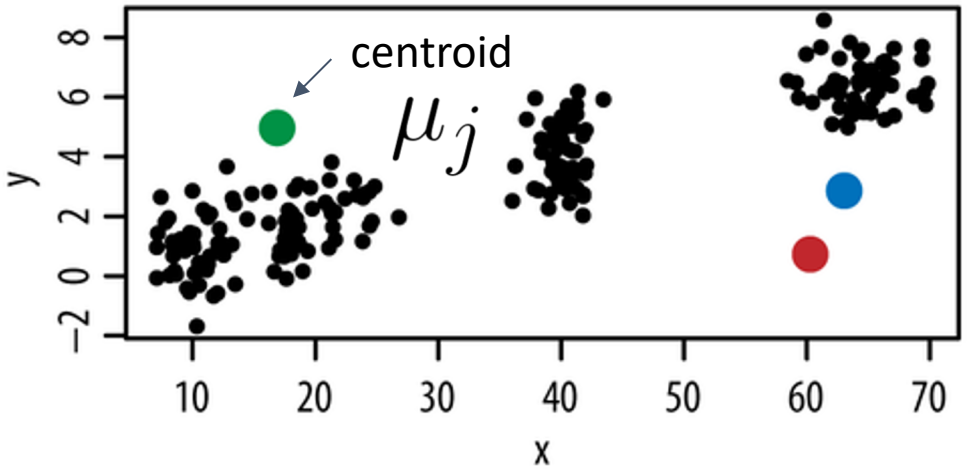
k-Means minimizes within-cluster sum-of-squares

Unlabelled data

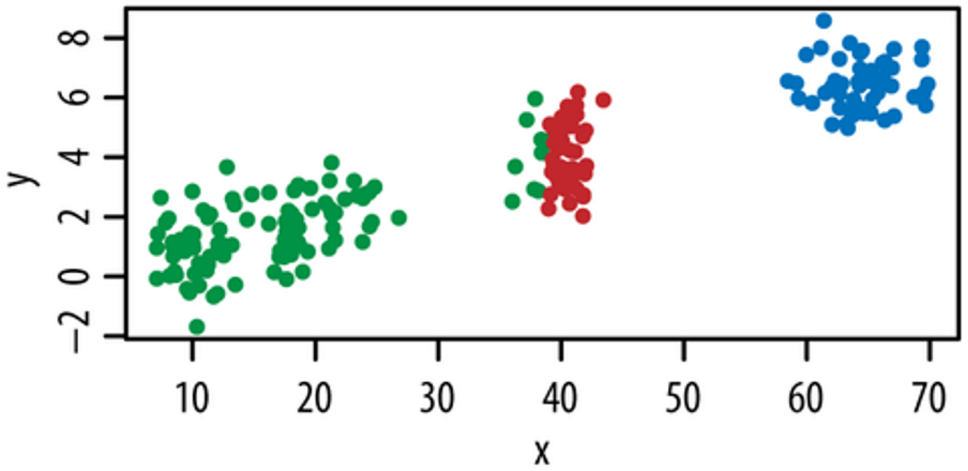


k-Means minimizes within-cluster sum-of-squares

1. Random initialization



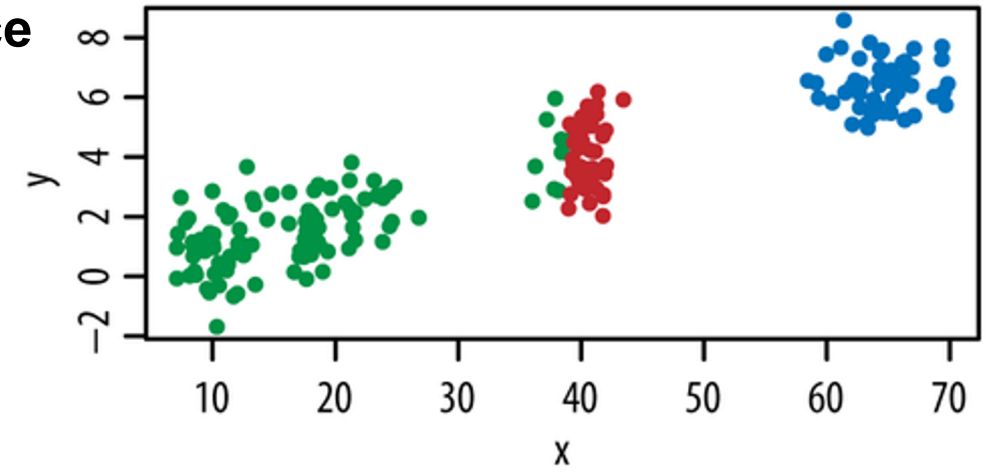
2. Assign points to nearest centroid



k-Means minimizes within-cluster sum-of-squares

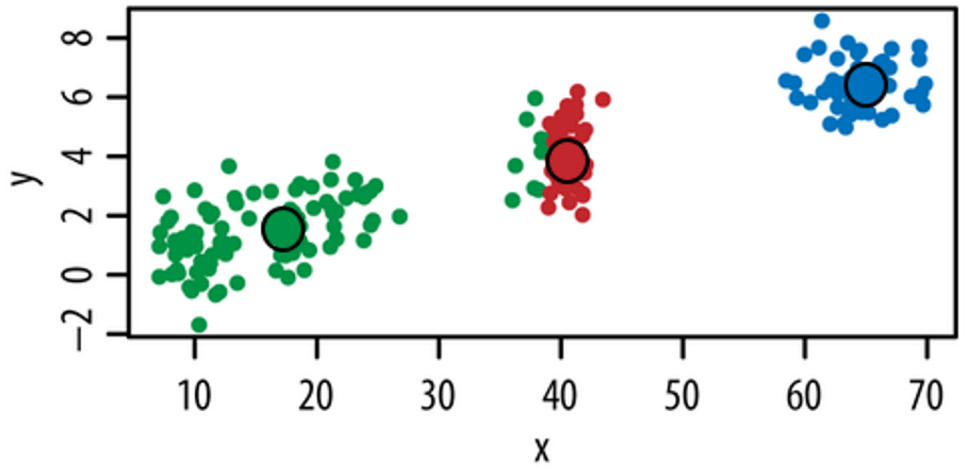
Repeat until convergence

2. Assign points to nearest centroid



3. Move centroids to middle of cluster

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

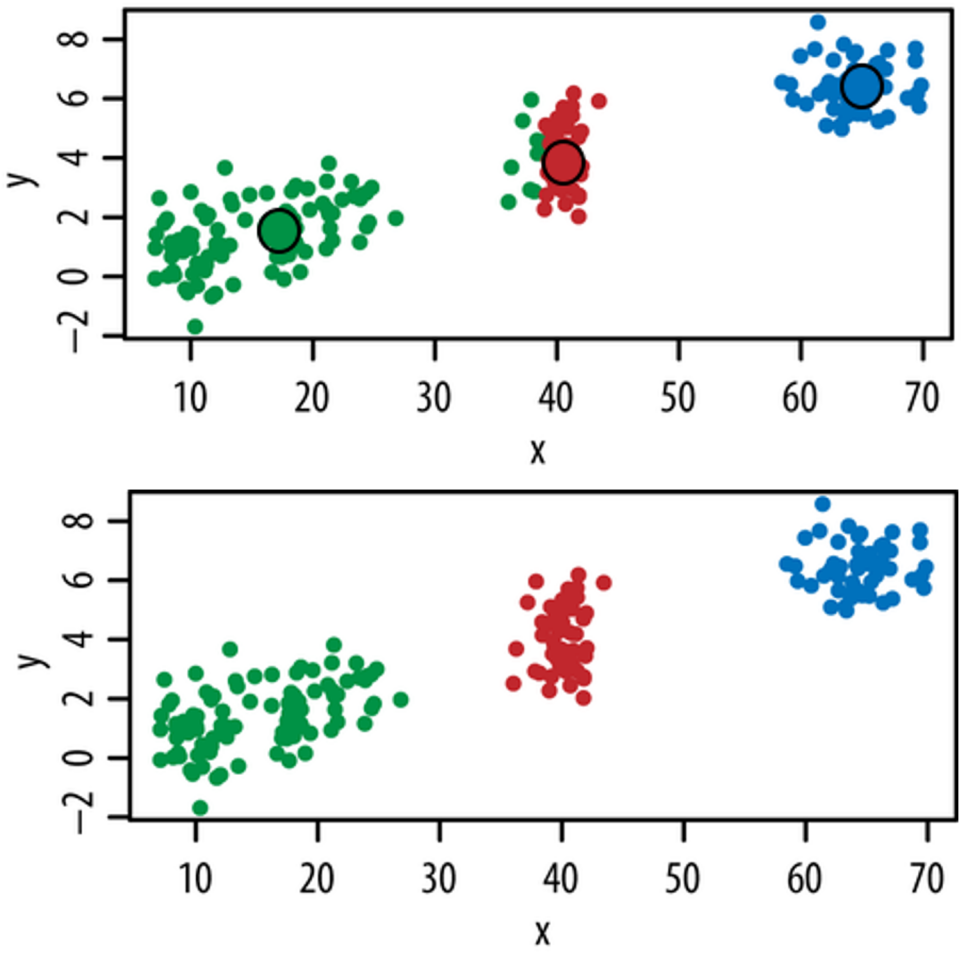


k-Means minimizes within-cluster sum-of-squares

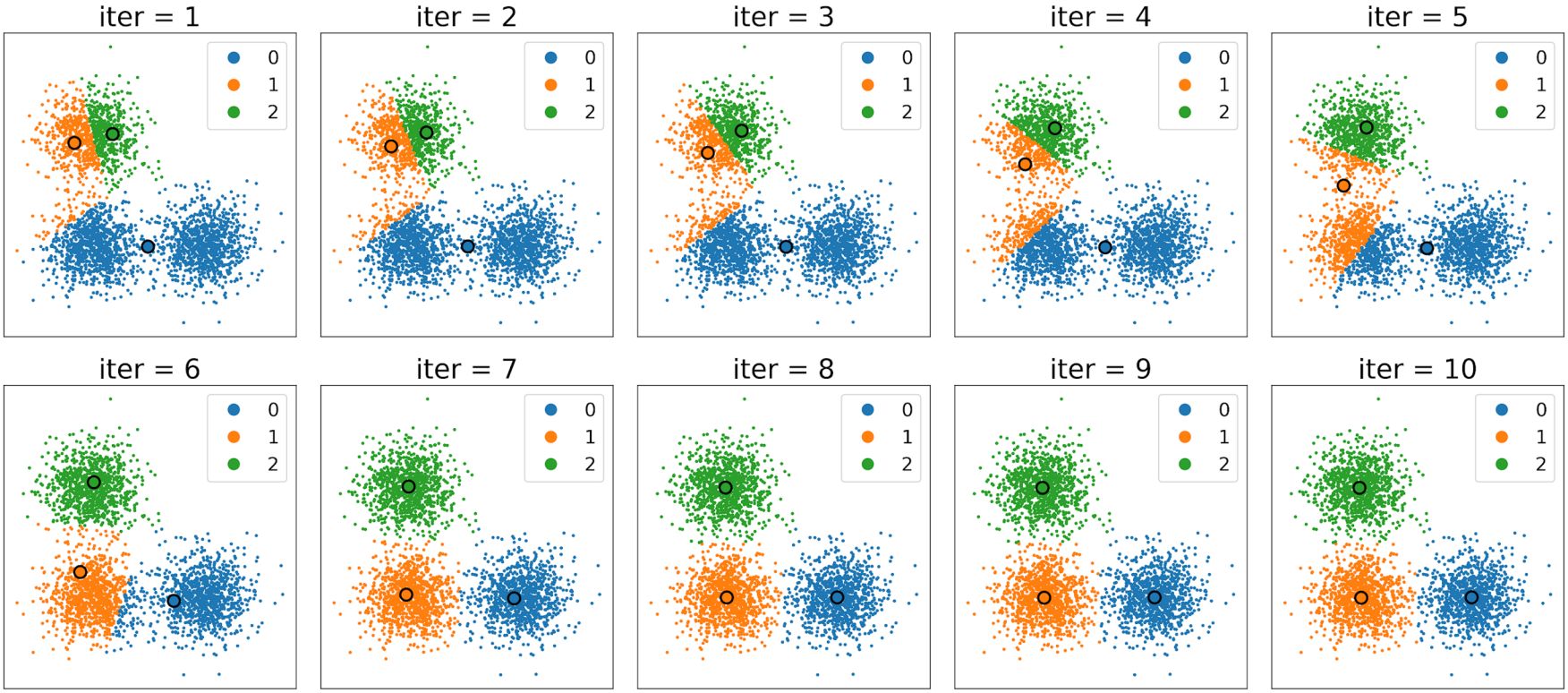
Repeat until convergence

3. Move centroids to middle of cluster

4. Assign points to nearest centroid

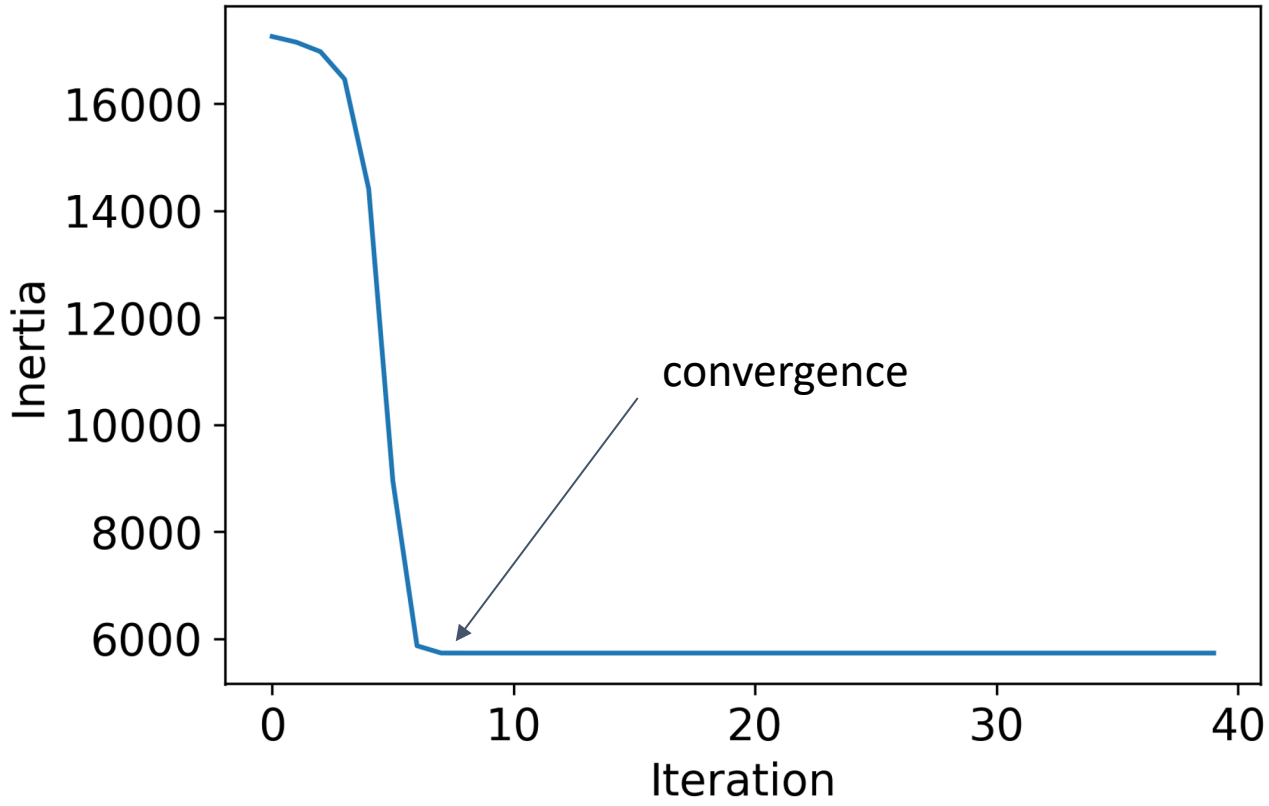


10 iterations of KMeans on Gaussians



What does convergence look like?

Inertia = within-cluster sum-of-squares = $$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$



What does it converge to?

- Generally a local minima.
- Sensitive to initial conditions
- Methods for initialization:
  - Forgy: K observations chosen as means
  - Random Partition: Randomly partitions data

Do you think KMeans interations from a given initialization will converge at a single solution?

Yes, the algorithm will always arrive at the same solution given the same initialization

No, the algorithm can produce different solutions given the same initialization

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

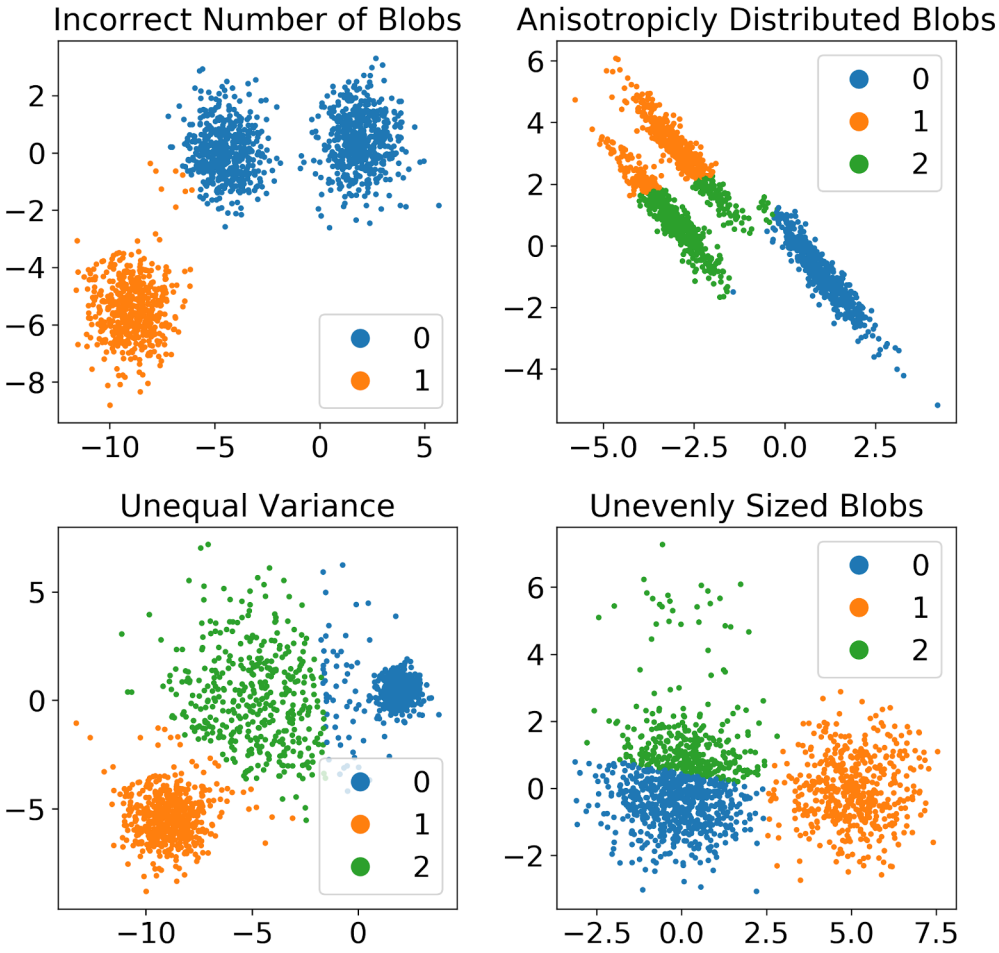
“K-means clustering is not a free lunch”

k-Means assumes:

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

K-Means is sensitive to:

1. Initialization
2. Outliers



http://varianceexplained.org/r/kmeans-free-lunch/ https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html

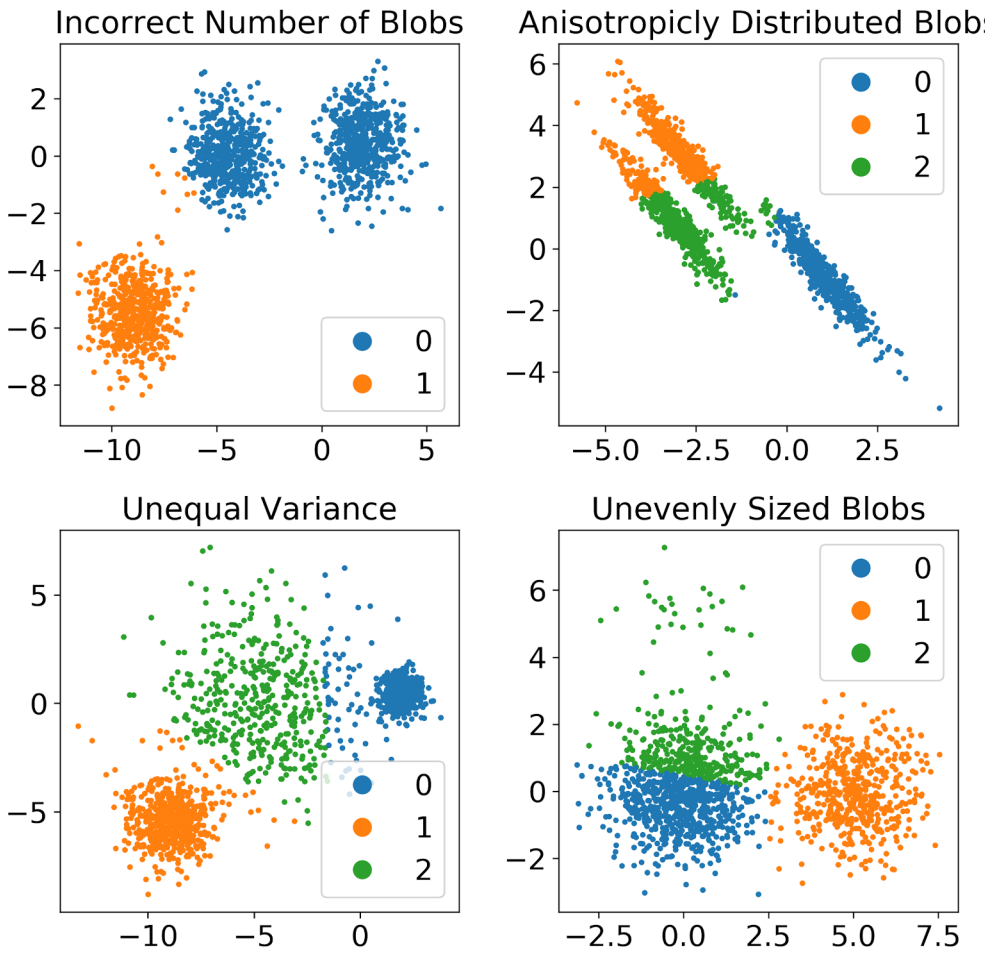
Limitations of this method

- Assumes a shape for the cluster: spectral clustering can help with this
- Must give number of clusters: there are methods to choose the number of clusters
- Sensitive to outliers: k-mediods helps with this
- Finds a local minima: repeat with different initializations

“K-means clustering is not a free lunch”

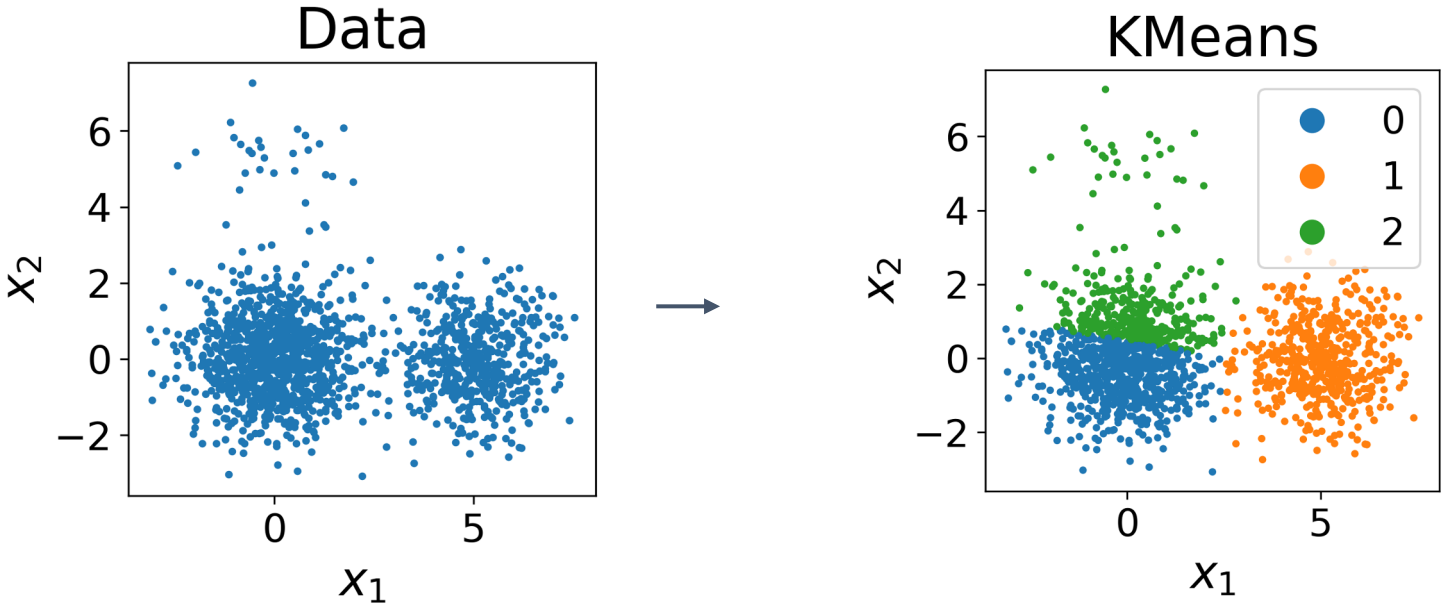
k-Means assumes:

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

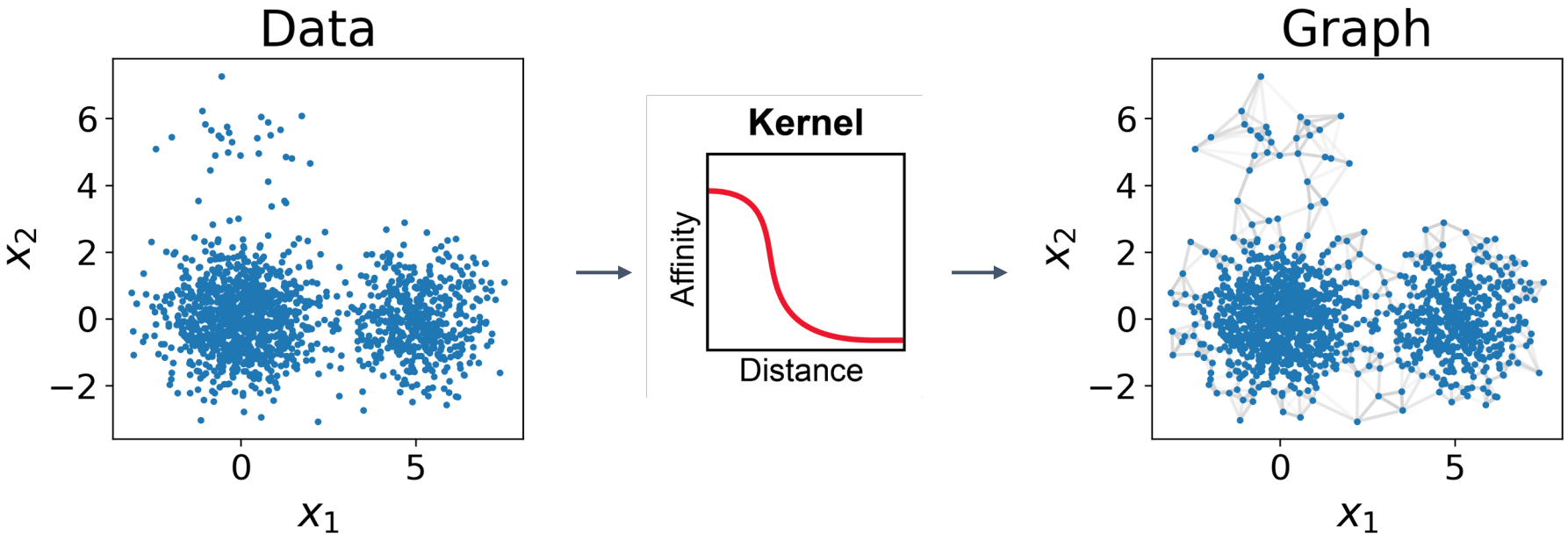


http://varianceexplained.org/r/kmeans-free-lunch/ https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html

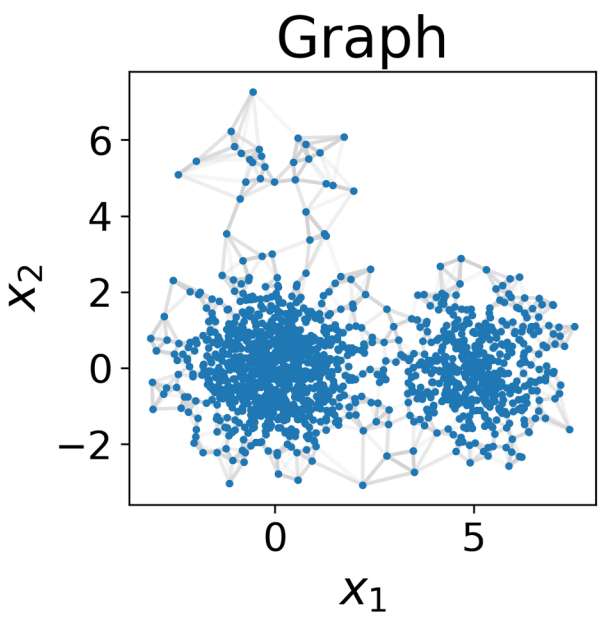
Spectral clustering uses eigenvectors of the graph Laplacian for clustering



Spectral clustering uses eigenvectors of the graph Laplacian for clustering



Spectral clustering uses eigenvectors of the graph Laplacian for clustering



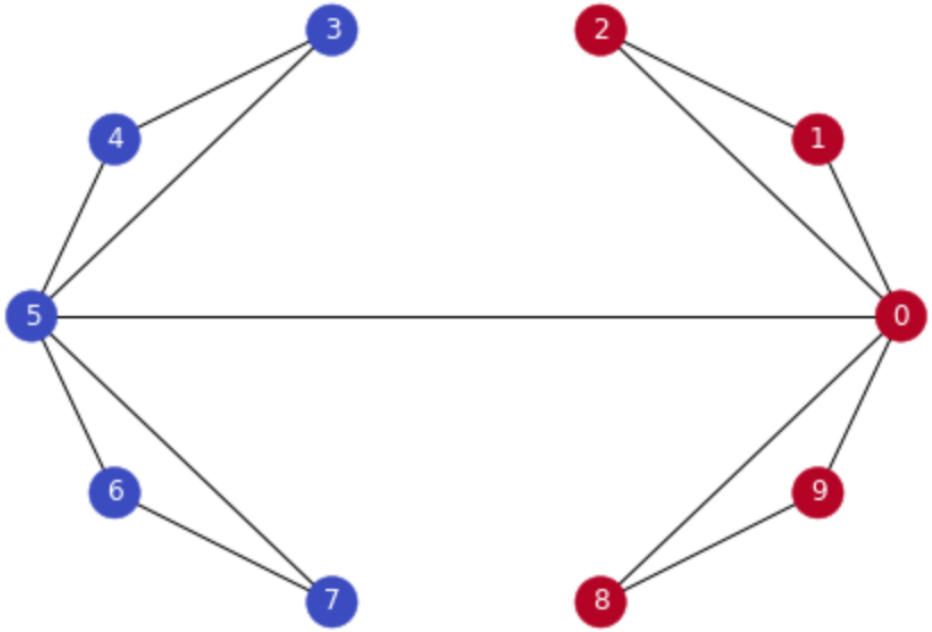
$\mathbf{A}$ = adjacency matrix

$\mathbf{A}_{i,j}$ = edge weight between nodes i and j

$\mathbf{D}$ = degree matrix

$\mathcal{L}$ = Laplacian matrix = $\mathbf{D} - \mathbf{A}$

Fiedler Vector

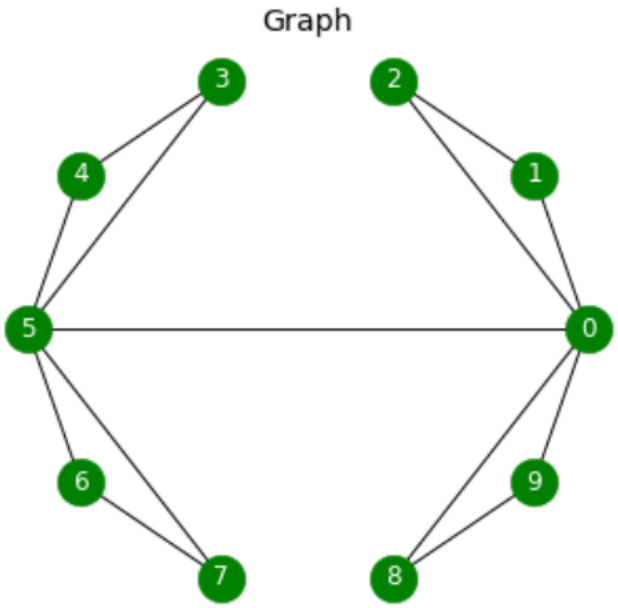


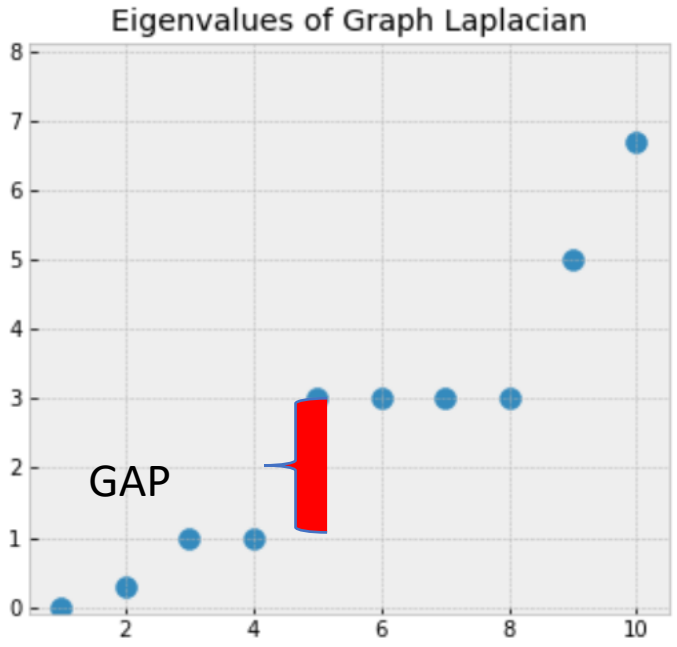
Second-smallest Eigenvalue is called Fiedler value, corresponding vector is the Fiedler vector

Fiedler value := approximates minimum cut needed to partition graph

Value if graph was already in two components?

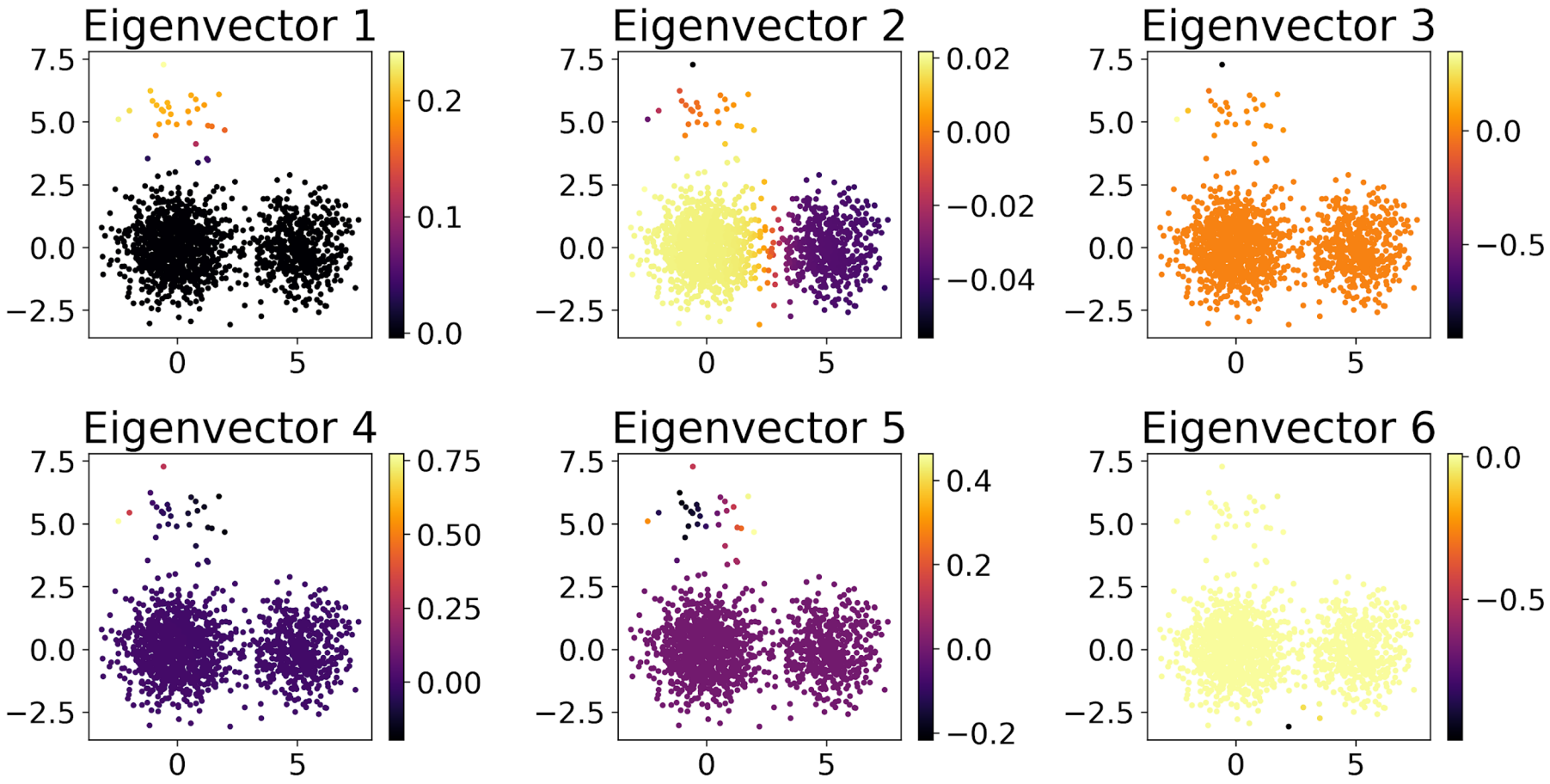
Vector can be used for partitioning, positive in one partition, negative in another



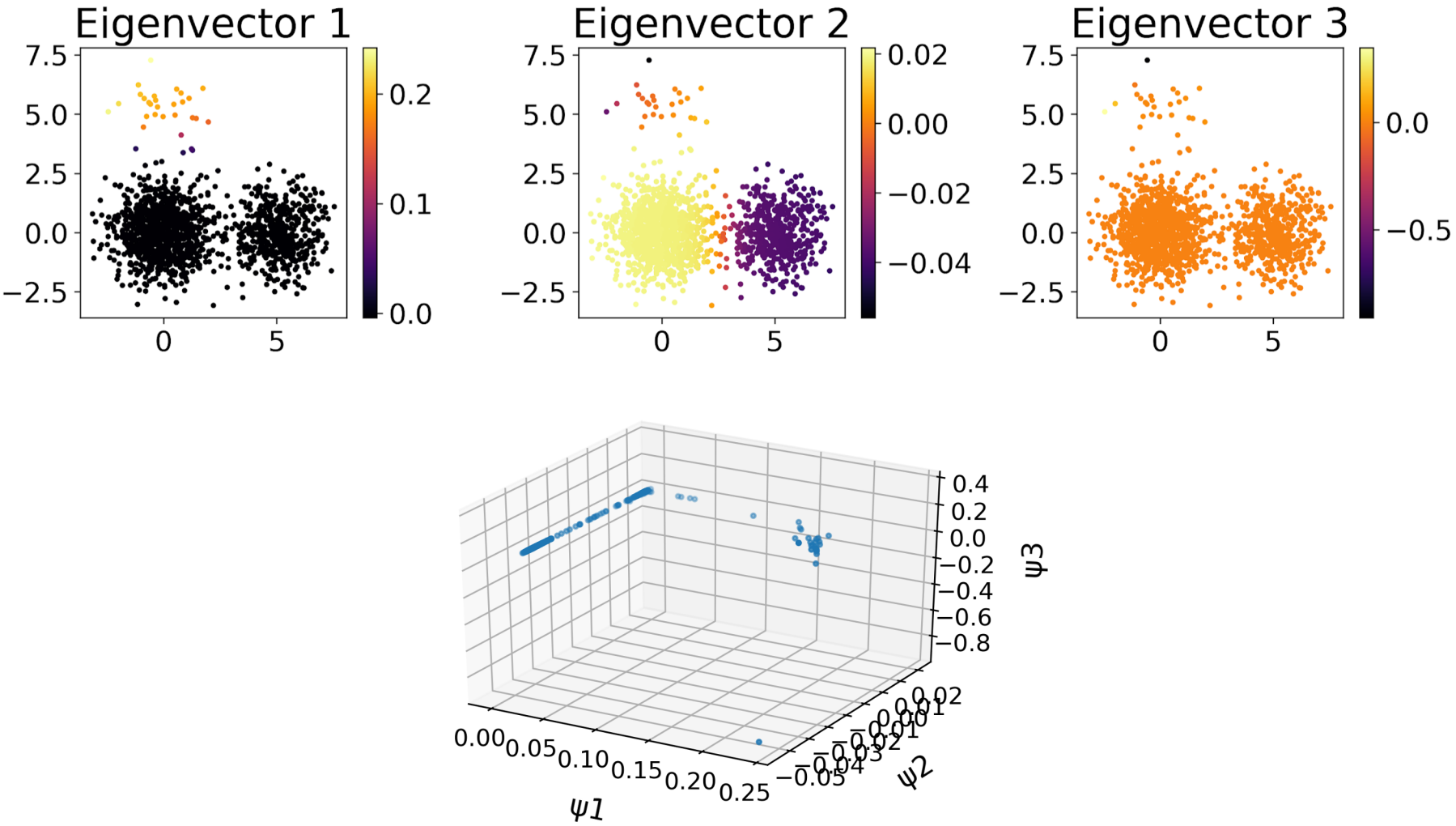


There is a gap between 4th and 5th values, increases suddenly, indicates that there are 4 clusters in the graph roughly.

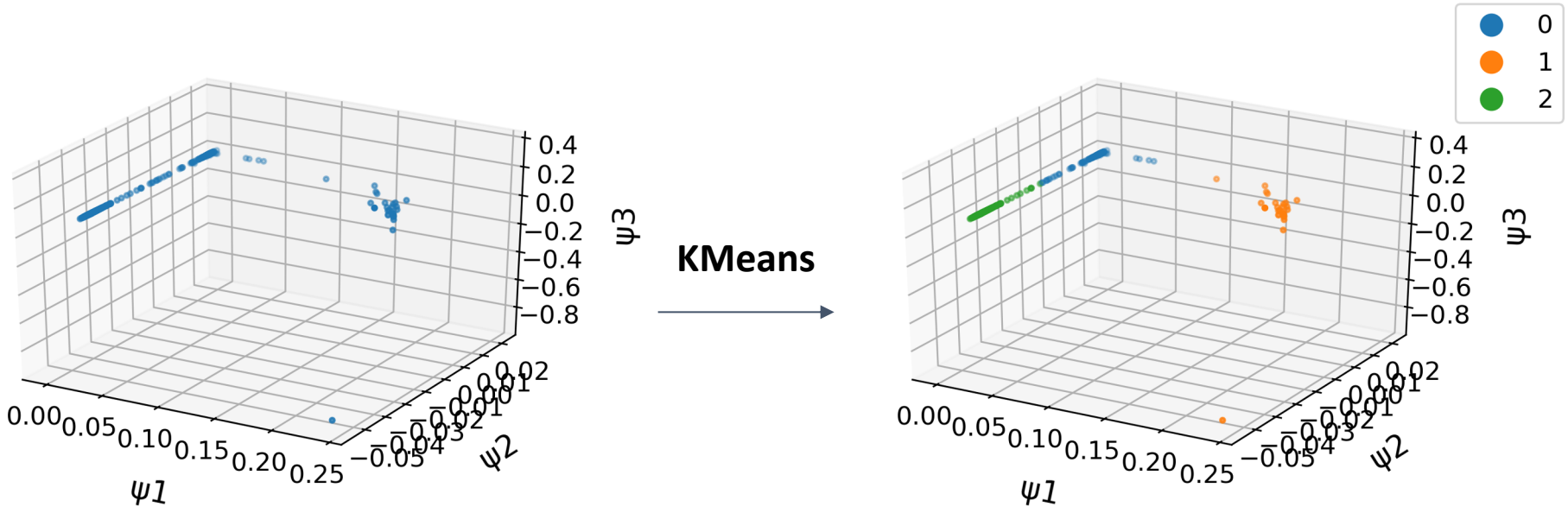
Spectral clustering uses eigenvectors of the graph Laplacian for clustering



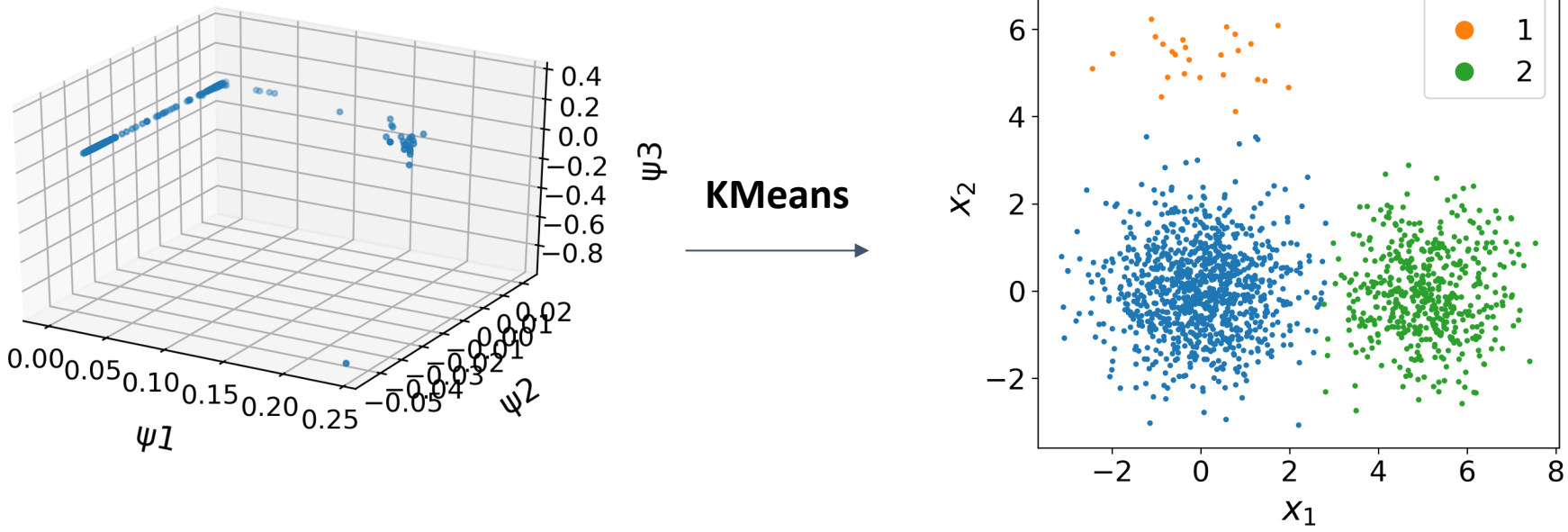
Spectral clustering uses eigenvectors of the graph Laplacian for clustering



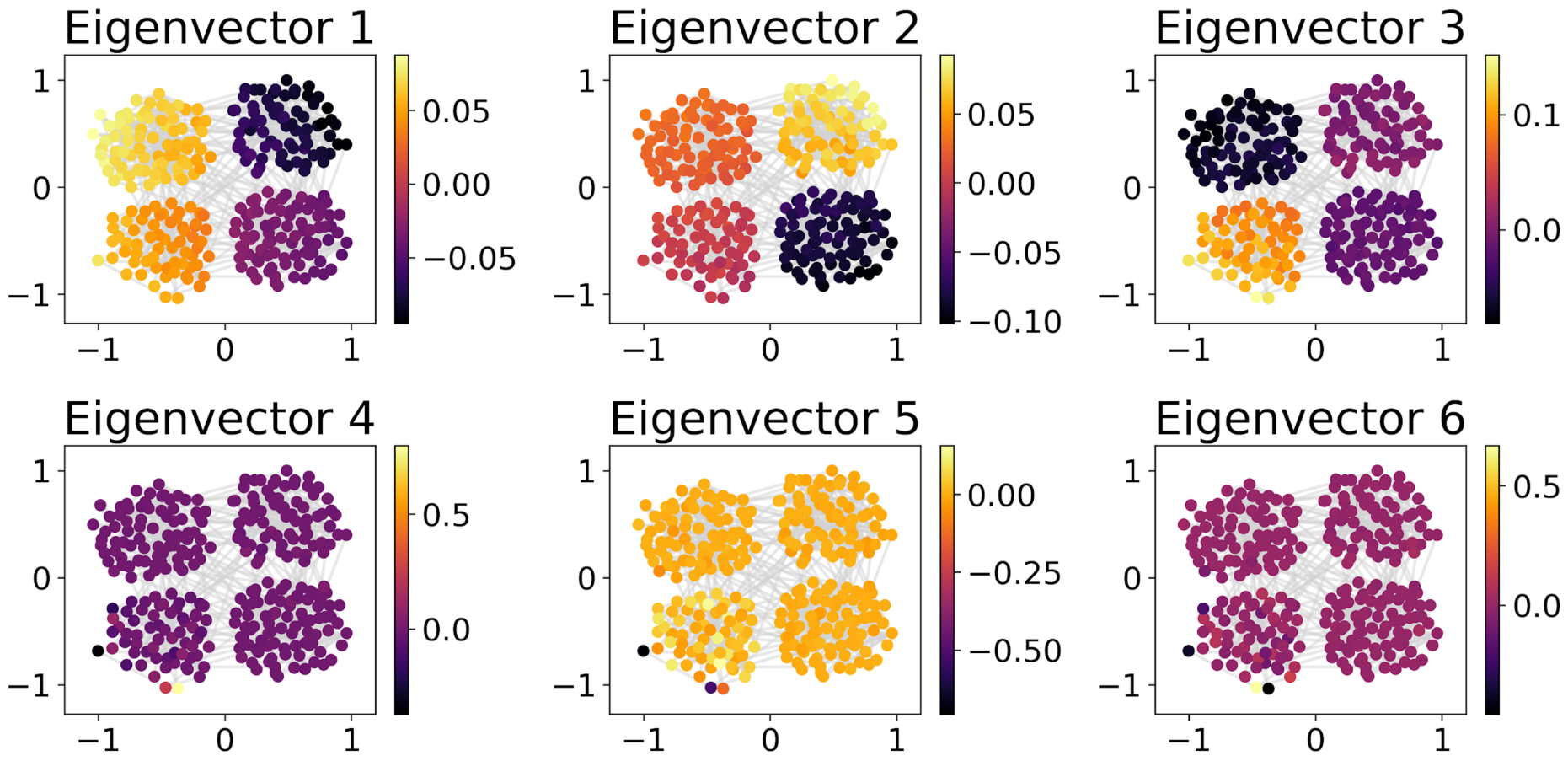
Spectral clustering uses eigenvectors of the graph Laplacian for clustering



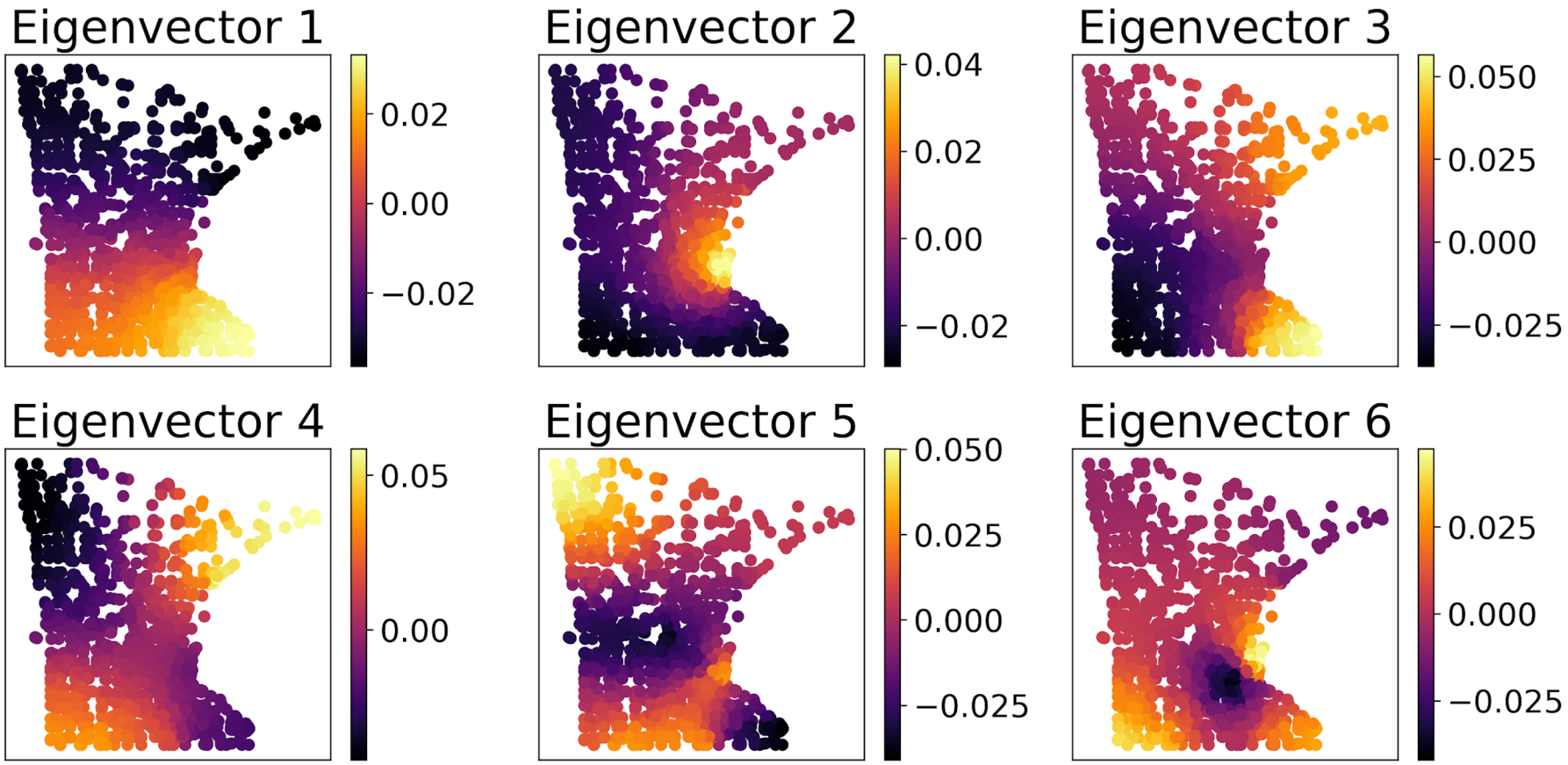
Spectral clustering uses eigenvectors of the graph Laplacian for clustering



Spectra of various graphs



Spectra of various graphs

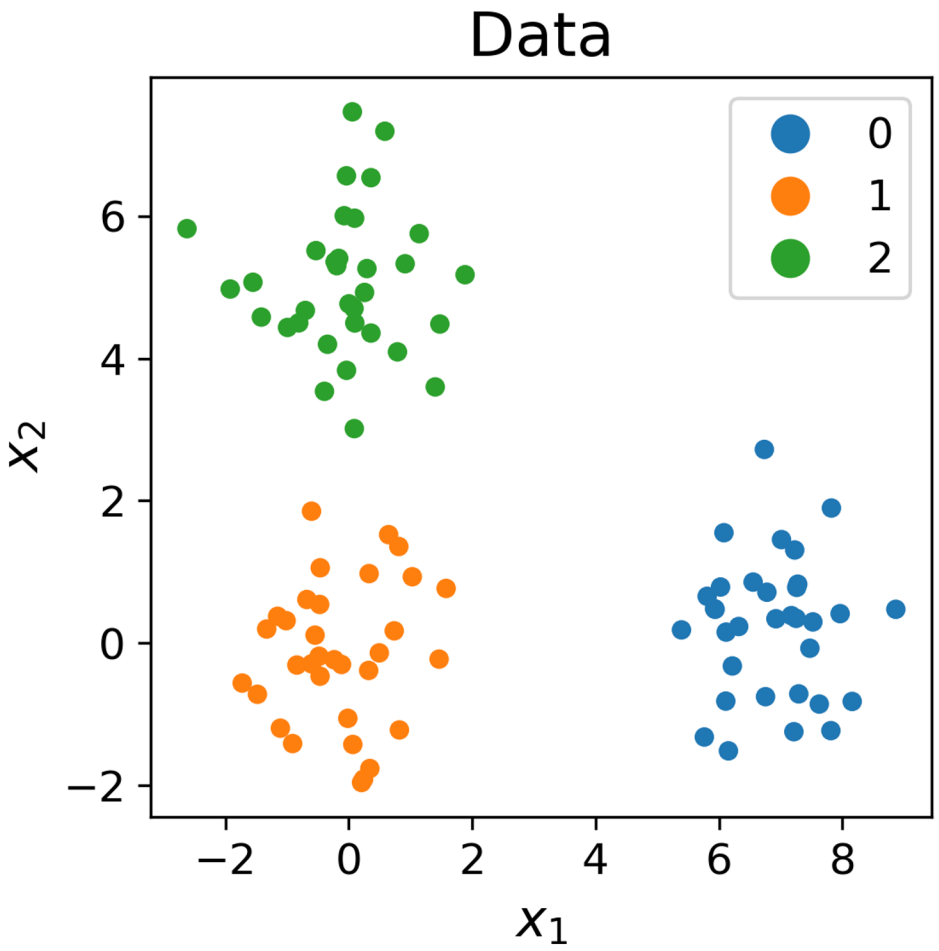


What are advantages of using the graph spectrum for k-Means?

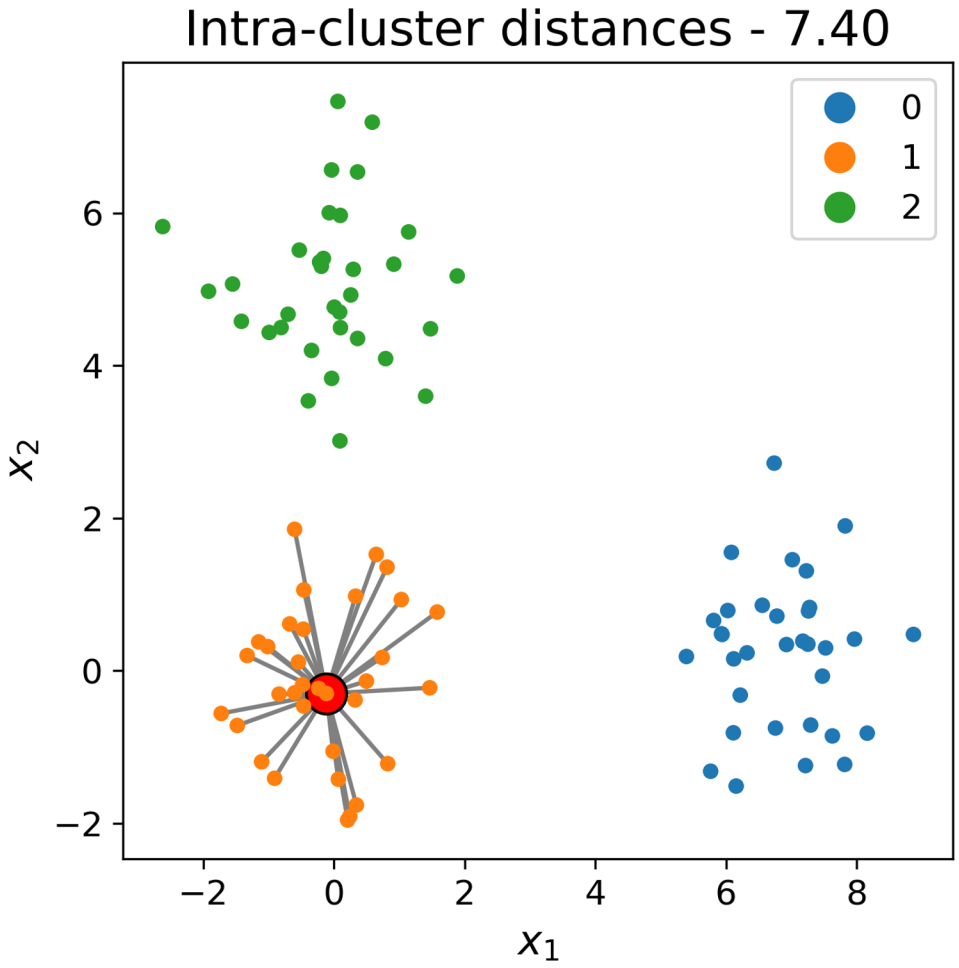
What are advantages of using the graph spectrum for k-Means?

- Good for clusters of arbitrary shape
- Good for data that is just a graph
- Only need connectivity information!

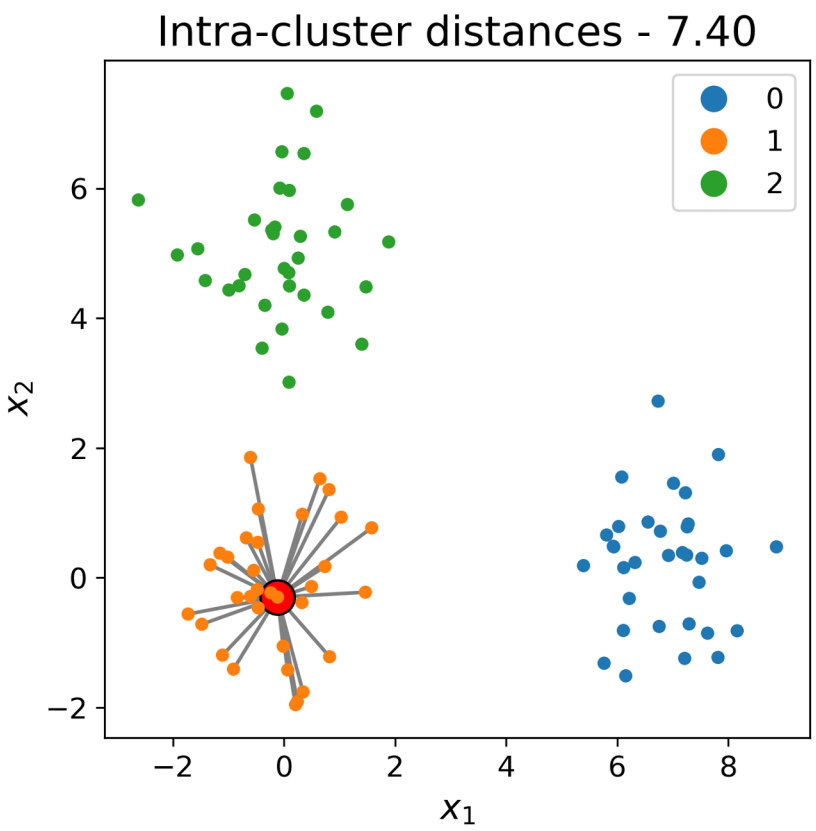
How do we decide the correct number of clusters?

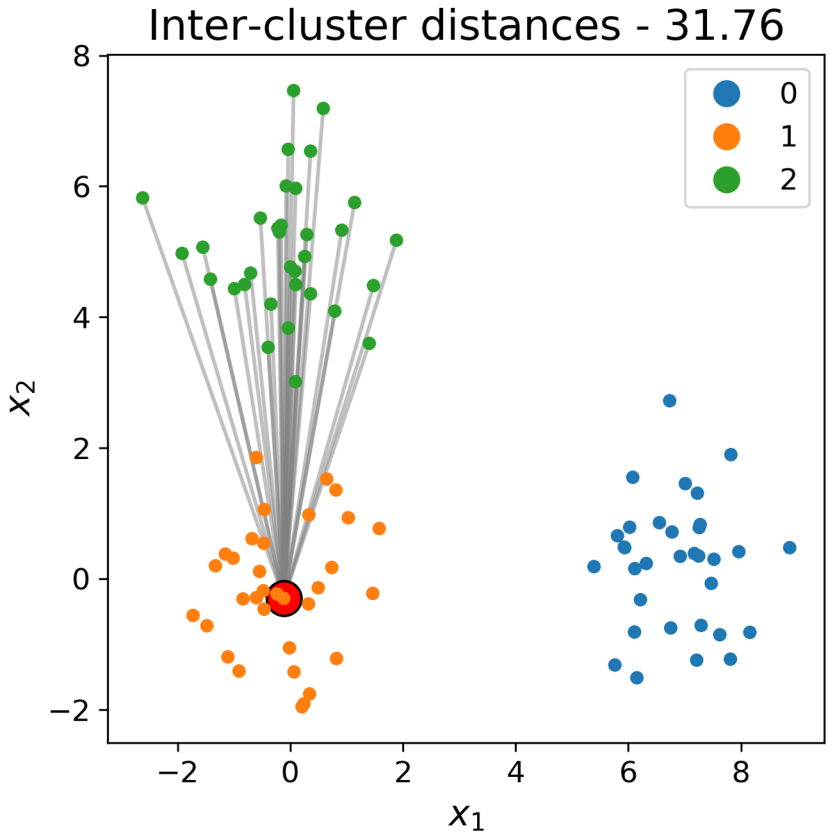


How do we decide the correct number of clusters?

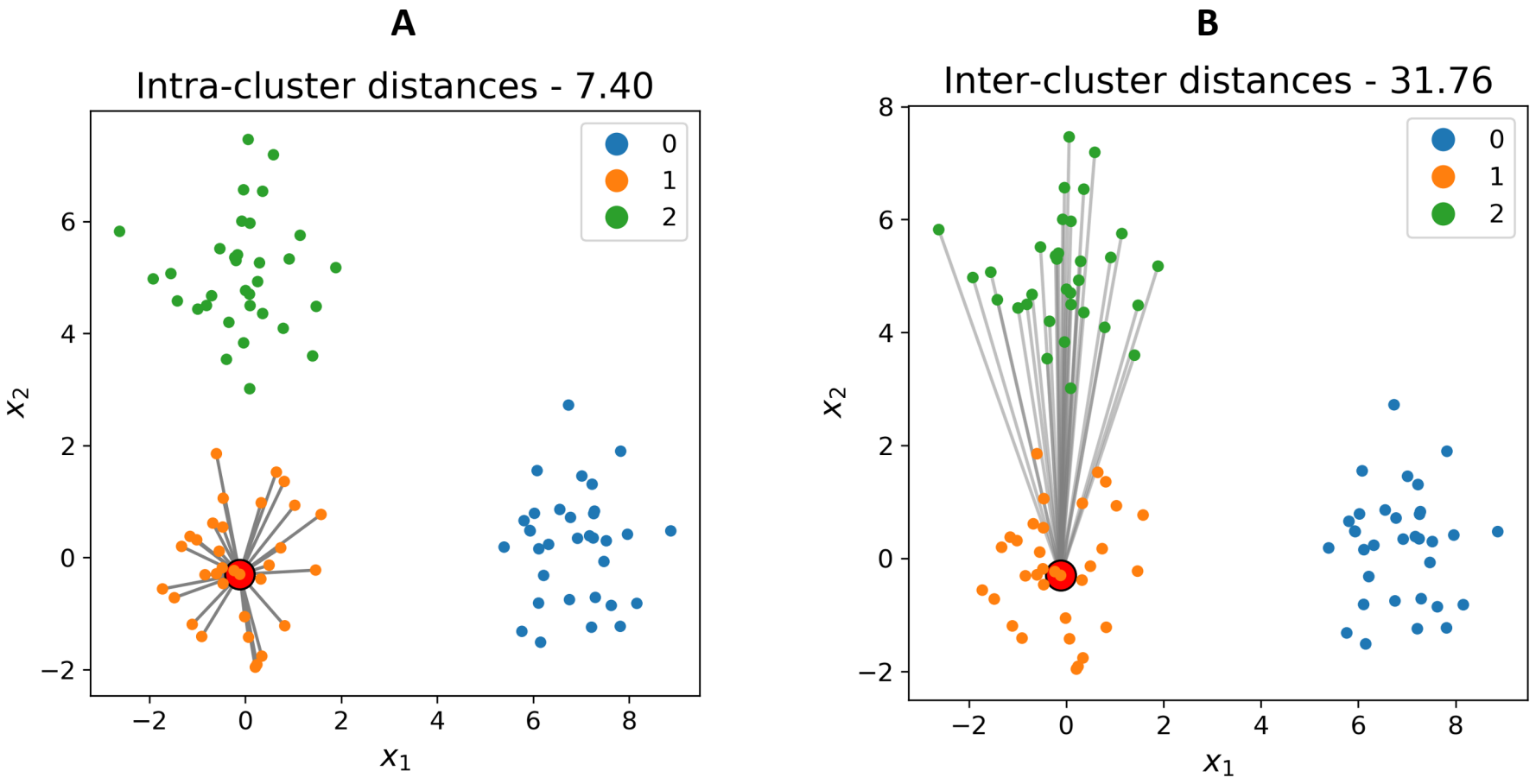


How do we decide the correct number of clusters?

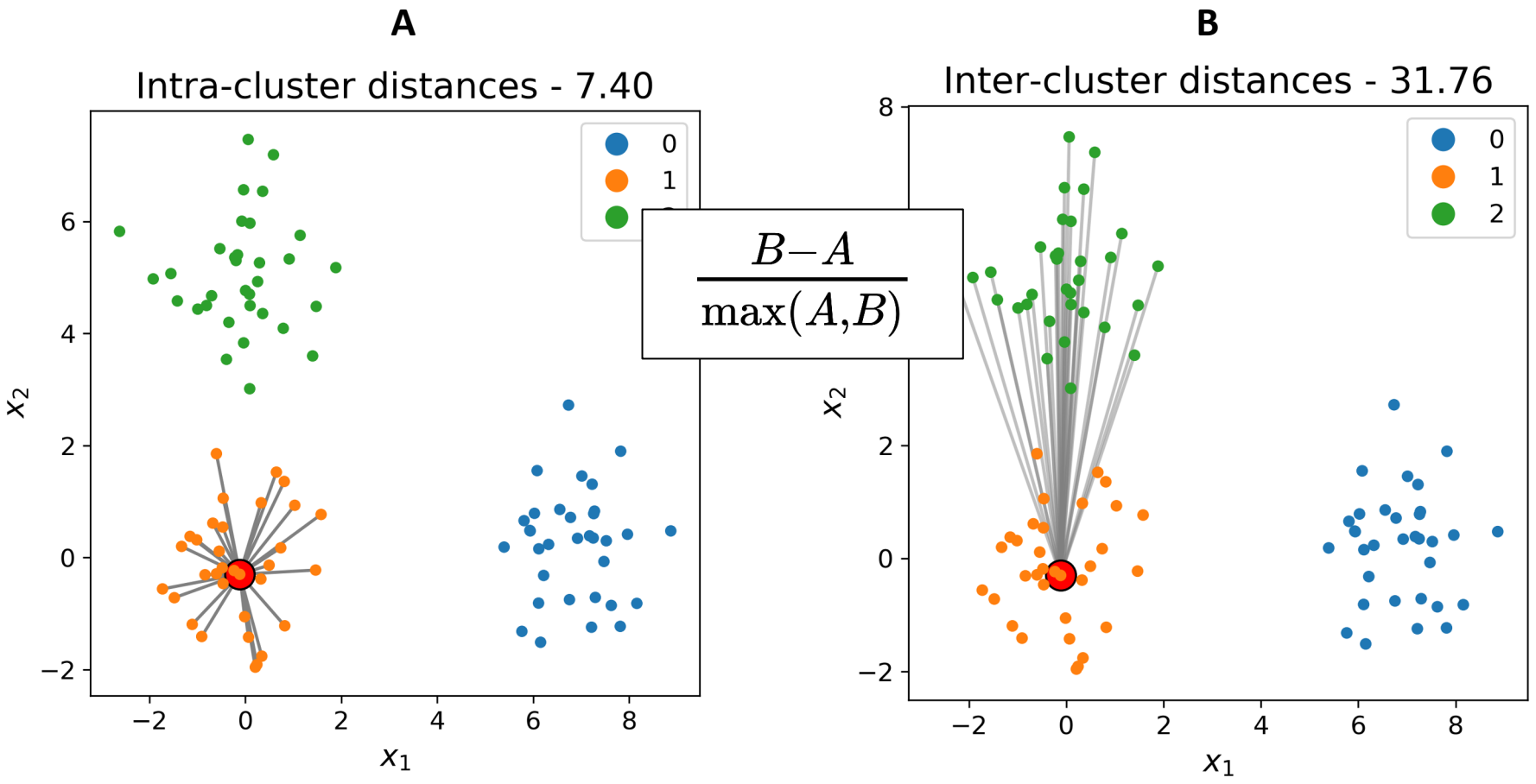




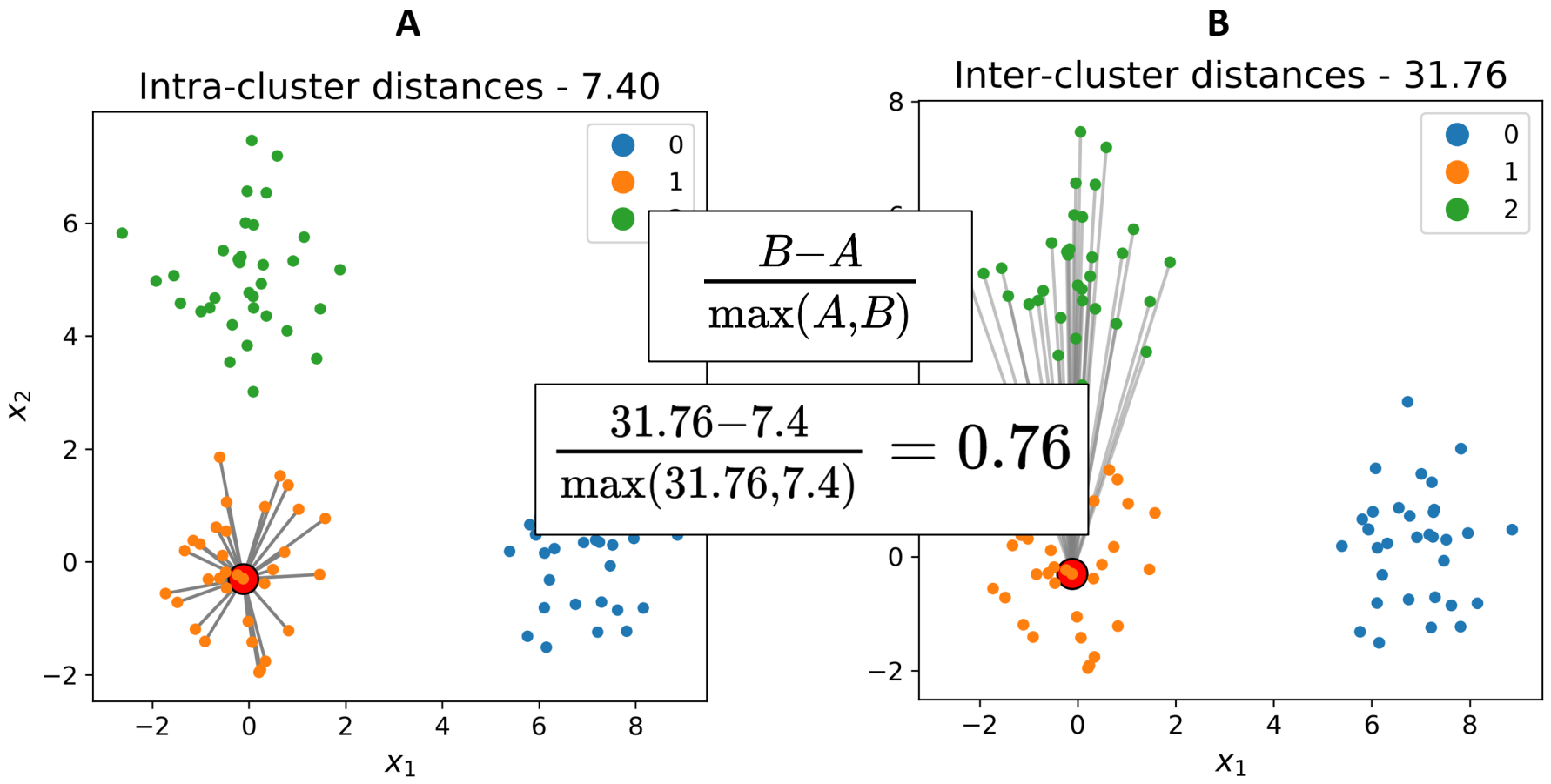
How do we decide the correct number of clusters?



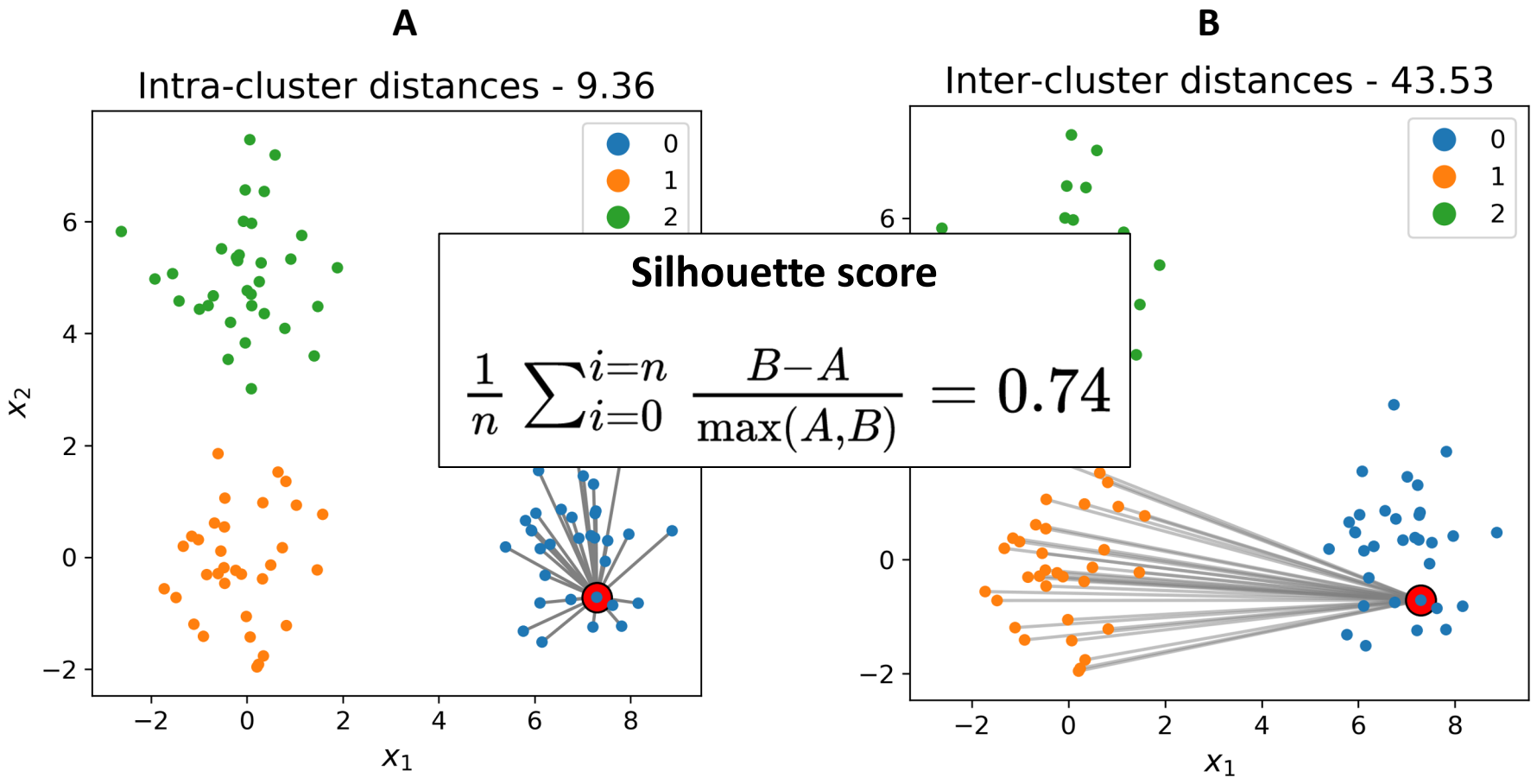
How do we decide the correct number of clusters?



How do we decide the correct number of clusters?



How do we decide the correct number of clusters?



Silhouette score is a measure of fit for cluster assignments

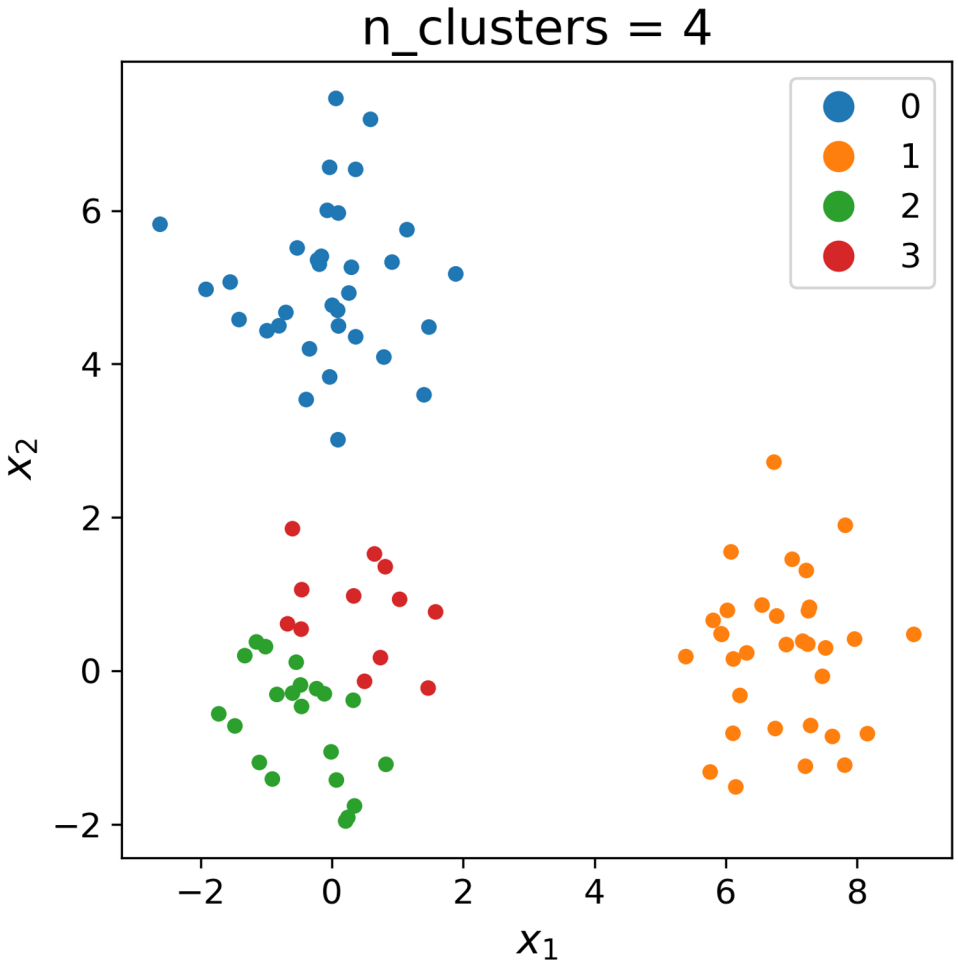
The Silhouette Score is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample.

The Silhouette Score for a sample is (b - a) / max(a, b).

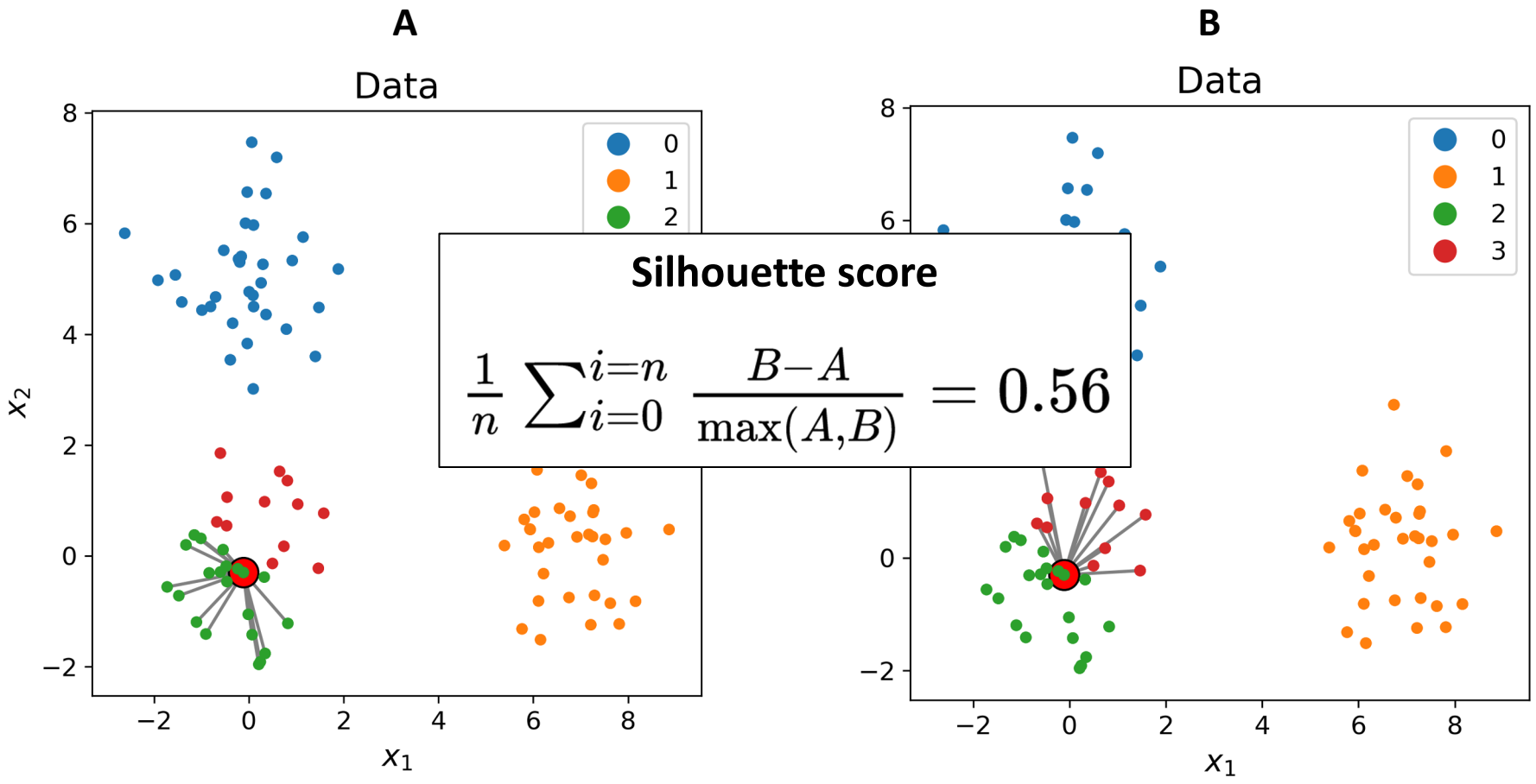
The best value is 1 and the worst value is -1. Values near 0 indicate overlapping clusters. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.s

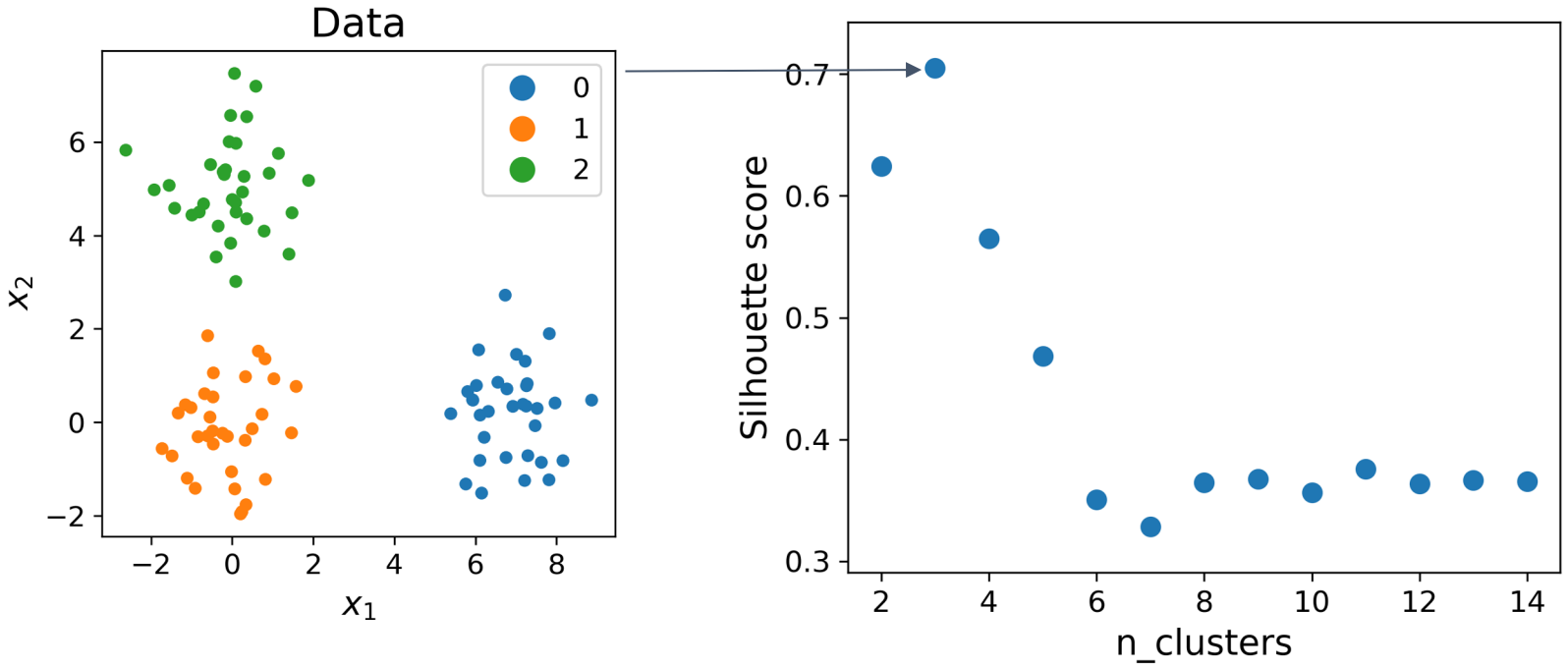
What happens when we change k?



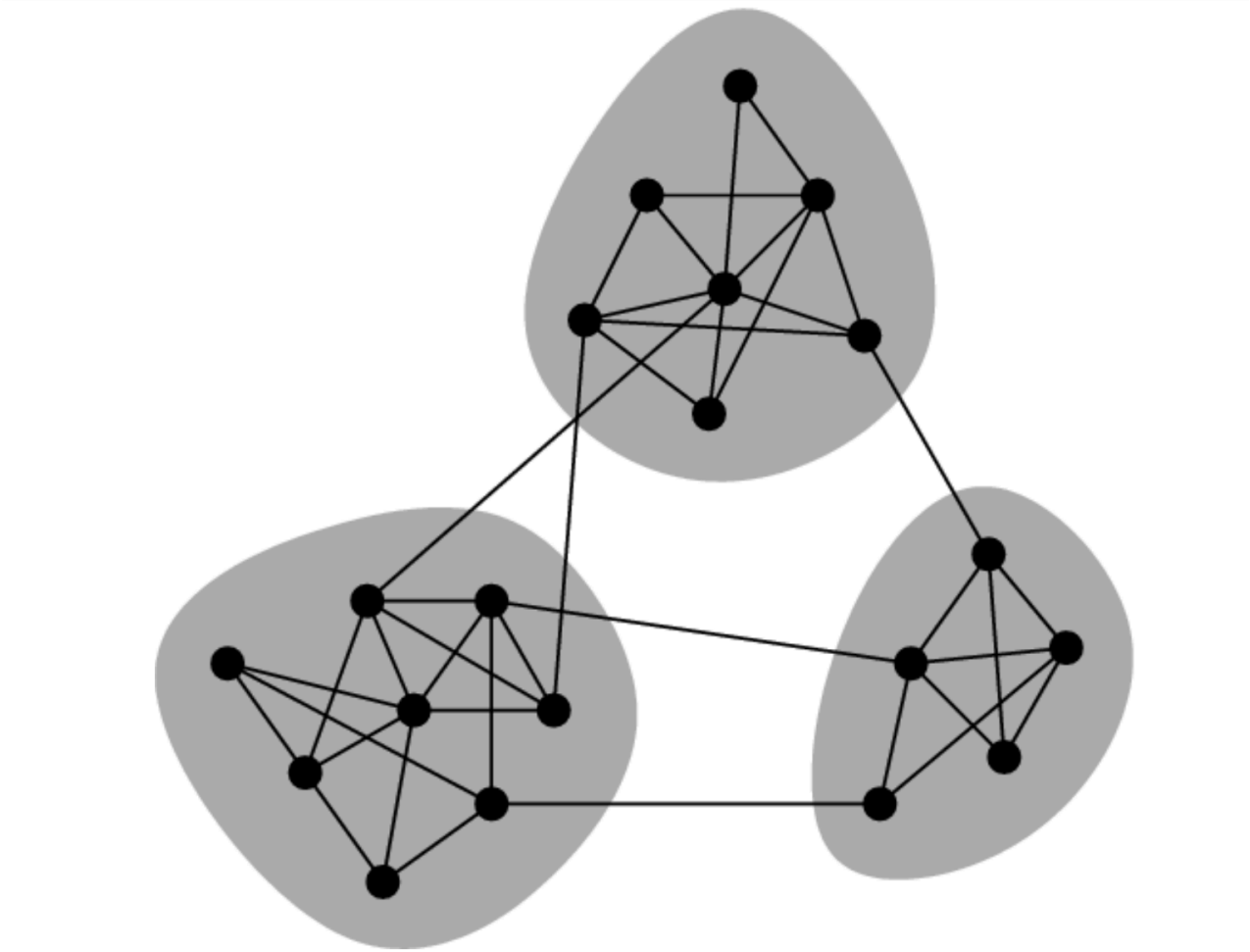
How do we decide the correct number of clusters?



Examining Silhouette score across n_clusters indicates ideal `k`

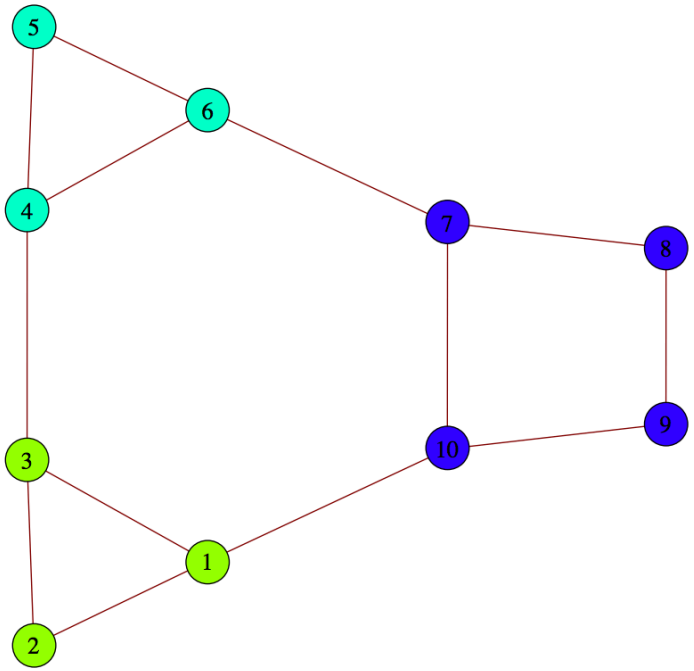


Graph-based Clusters

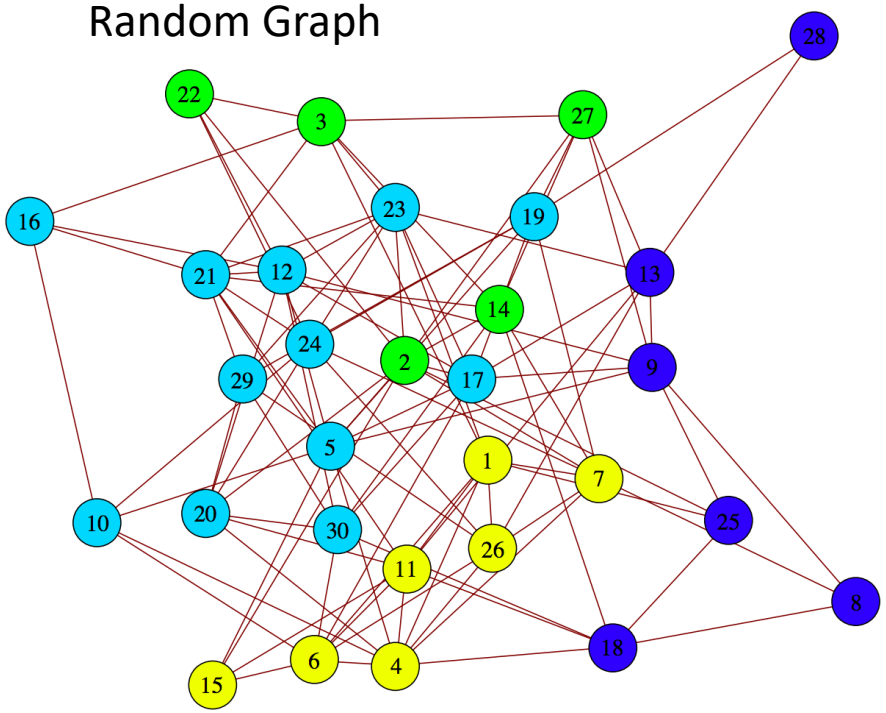


Clustering by “modularity”

Modular Graph



Random Graph



Newman and Girvan (2004)

Louvain clustering maximizes modularity of clusters on a graph

Modularity = $$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

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

Louvain clustering maximizes modularity of clusters on a graph

constant

Adjusts for # edges per node

Modularity = $$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

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

Louvain clustering maximizes modularity of clusters on a graph

constant

Connectivity between nodes

Adjusts for # edges per node

Modularity = $$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

Only consider connections **within** communities

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

Louvain clustering maximizes modularity of clusters on a graph

constant

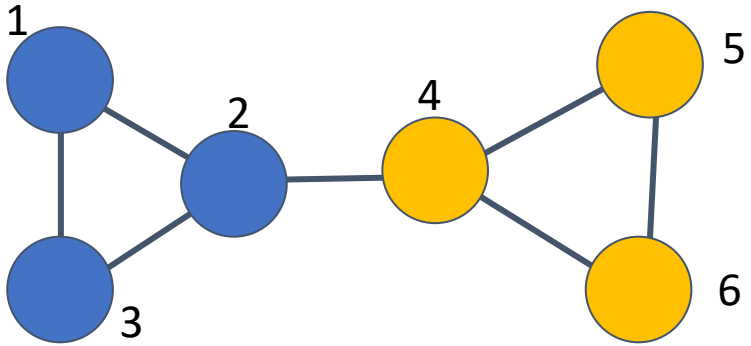
Connectivity between nodes

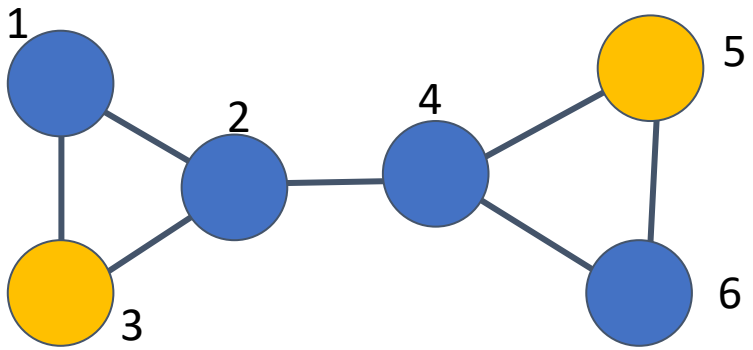
Adjusts for # edges per node

Modularity = $$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

Only consider connections **within** communities

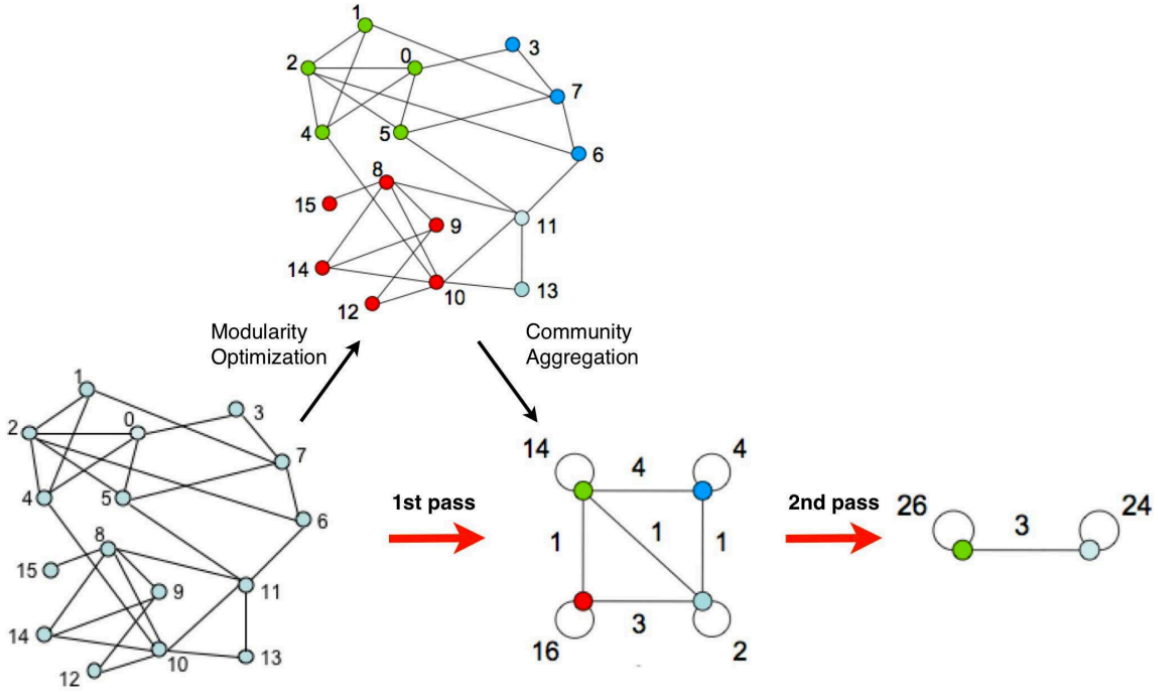
m is the sum of all edge weights in the graph
Aij is the edge weight between nodes *i* and *j*
ki is the sum of all edges around node i
δ is 1 if i and j are in the same community and 0 otherwise

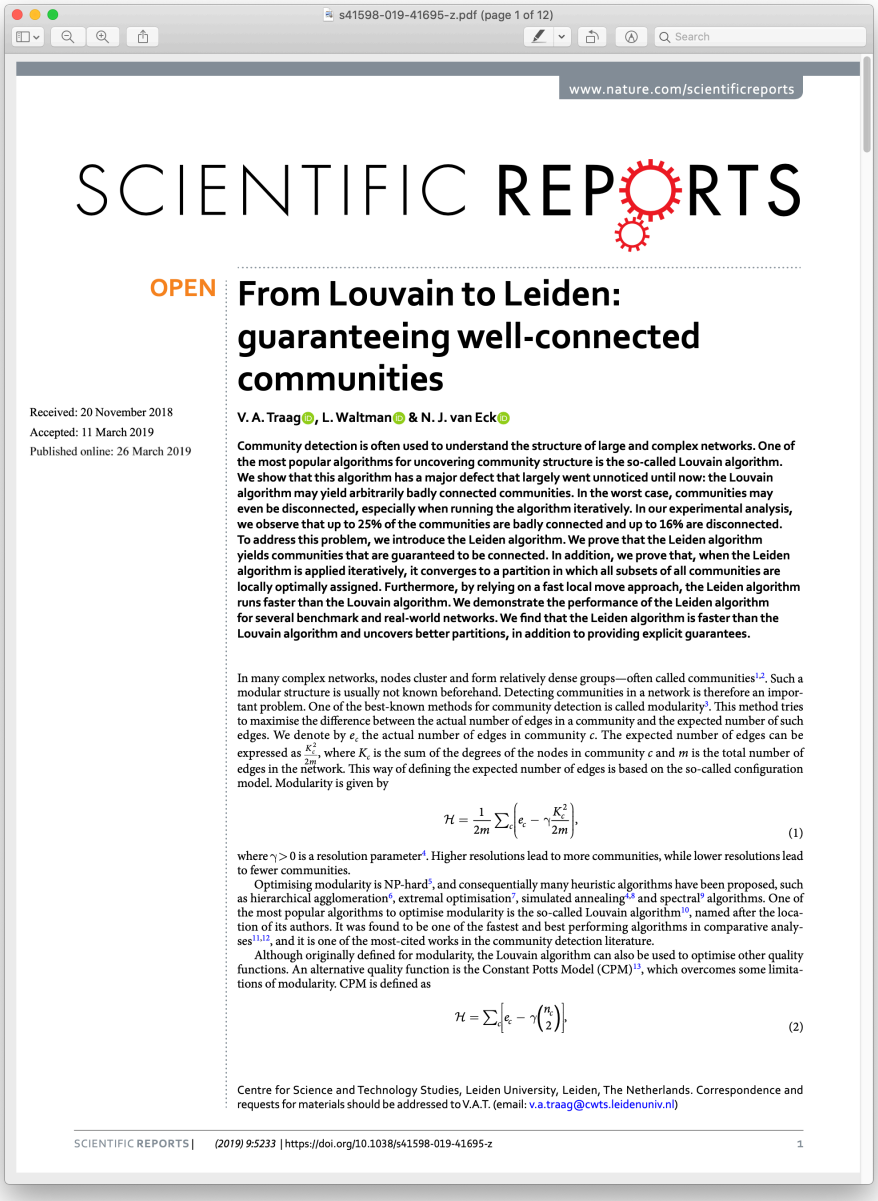




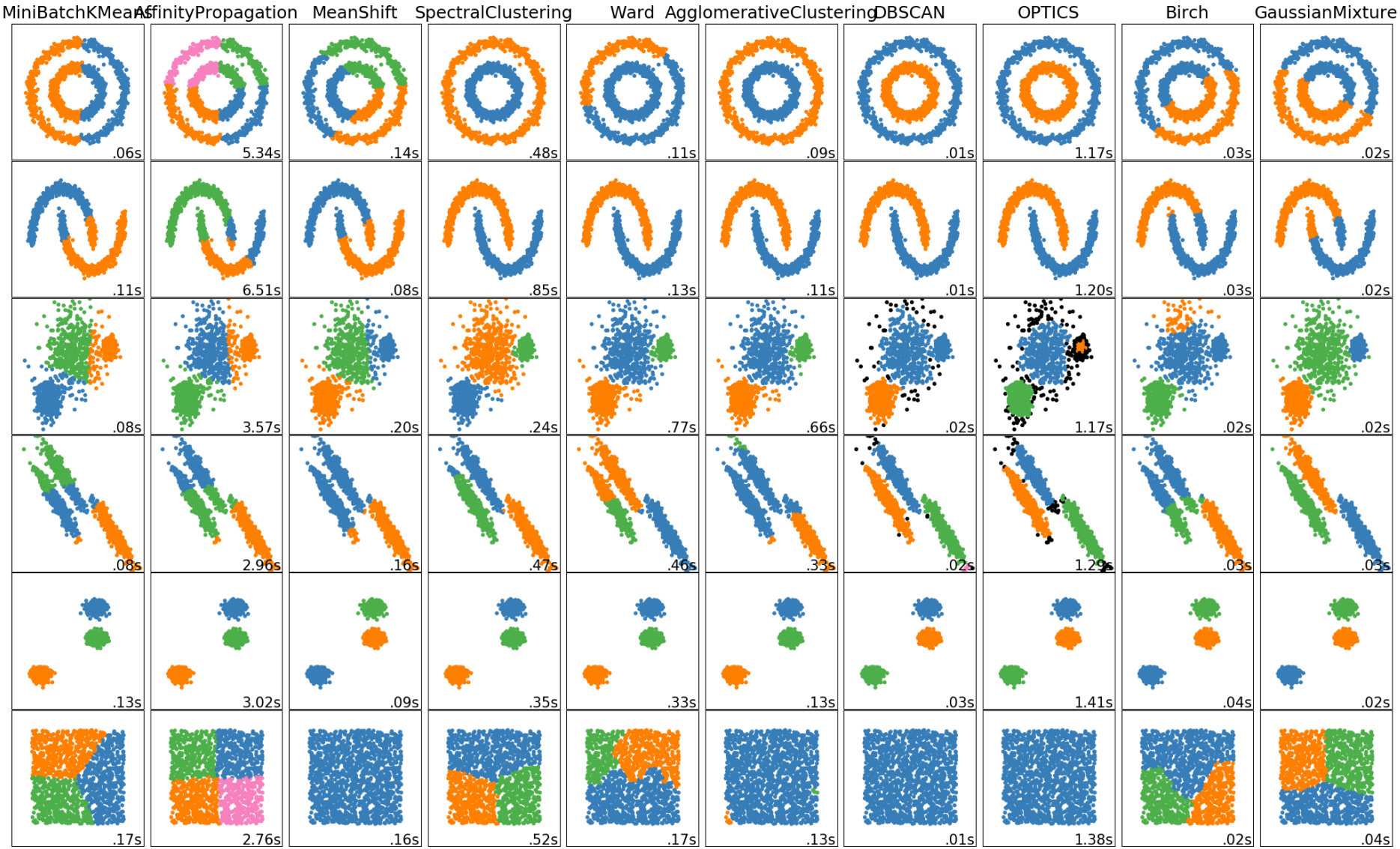
Greedy Modularity Optimization (Newman 2003)

- Start with each point in its own cluster
- Merge two clusters that would increase modularity by the most
- Stop when all merges reduce modularity





https://www.nature.com/articles/s41598-019-41695-z



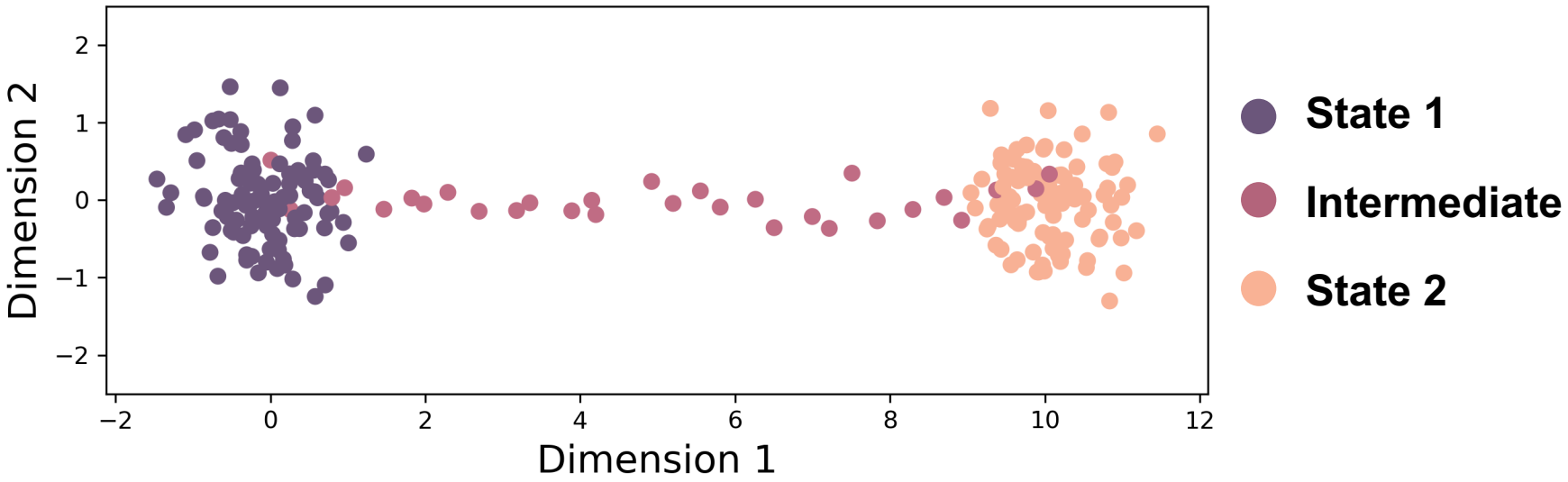
https://scikit-learn.org/stable/modules/clustering.html

Conclusions

- Clustering algorithms partition data to identify groups of similar observations
- KMeans is an iterative algorithm that minimizes within-cluster distances in the data space
- Spectral clustering minimizes within-cluster distances using eigenvectors of the laplacian
- Louvain / Leiden iteratively aggregate clusters on a graph to increase a modularity heuristic

How do we infer continuous changes in cell state from single cells?

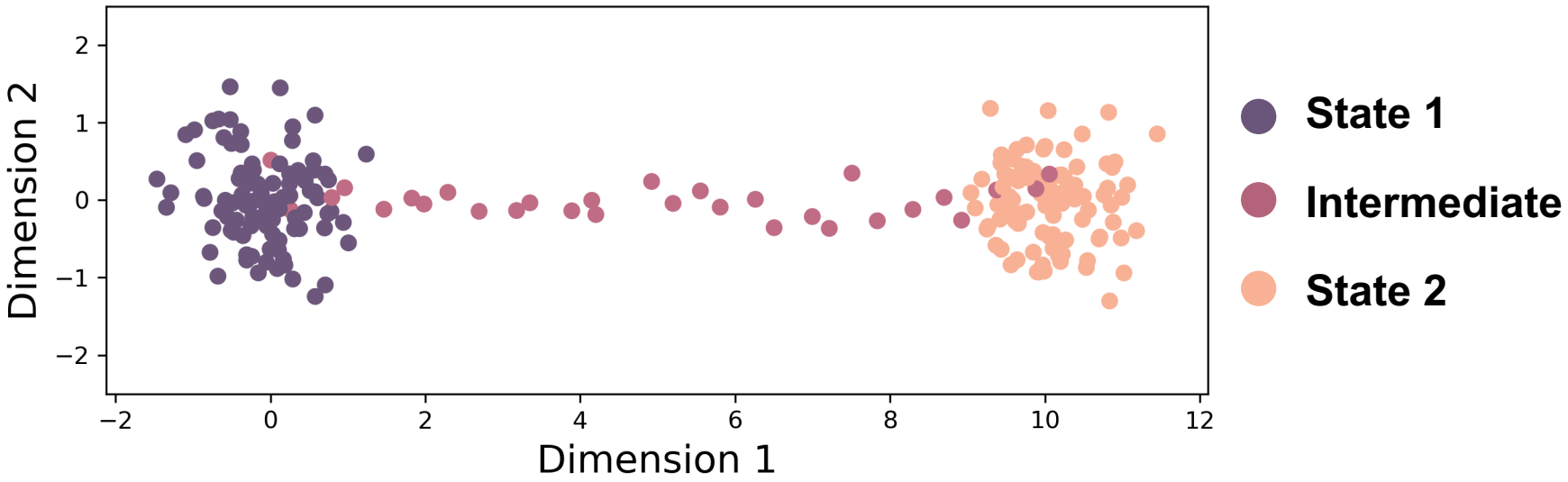
A simple developmental system



Dimension 1: “Genes that change from State 1 to State 2”

Dimension 2: “Noise genes”

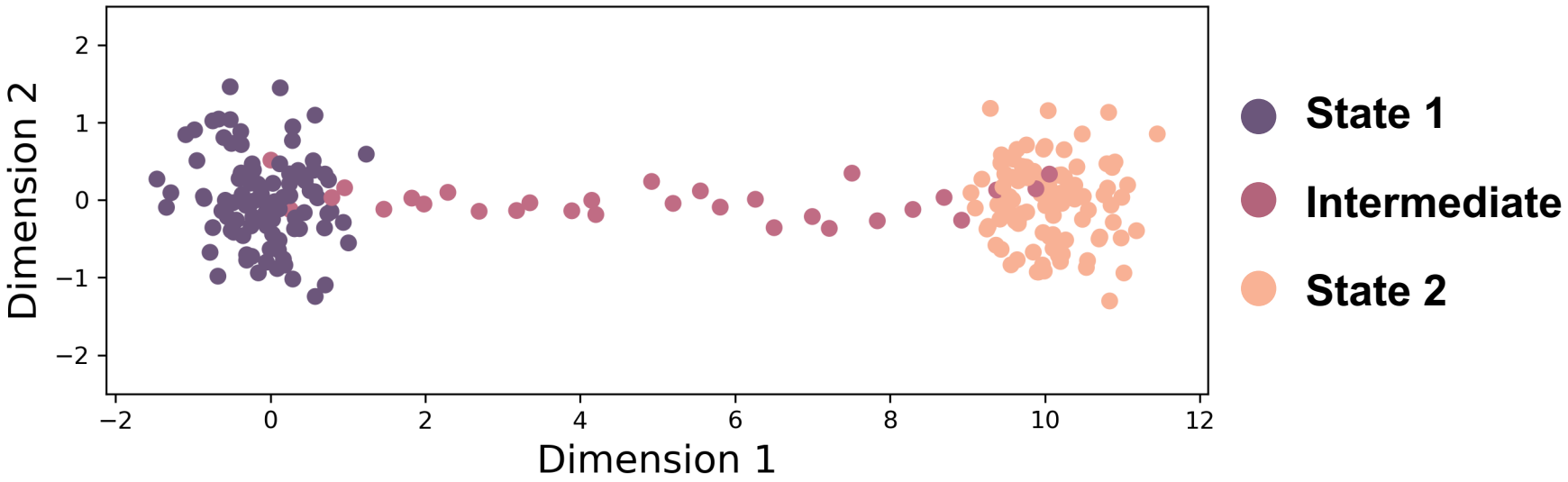
A simple developmental system

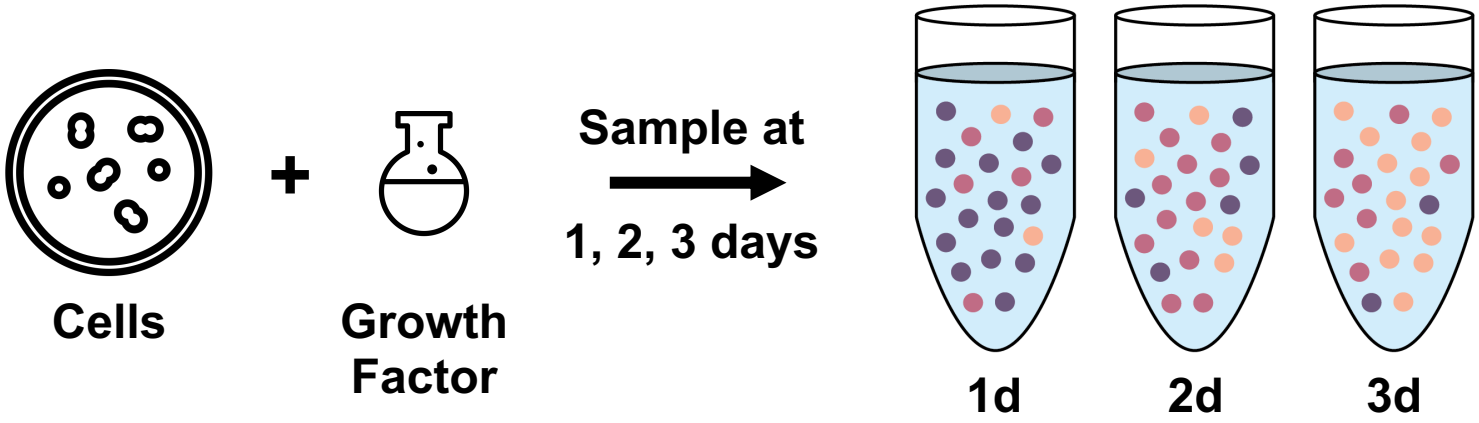


Things we might like to know:

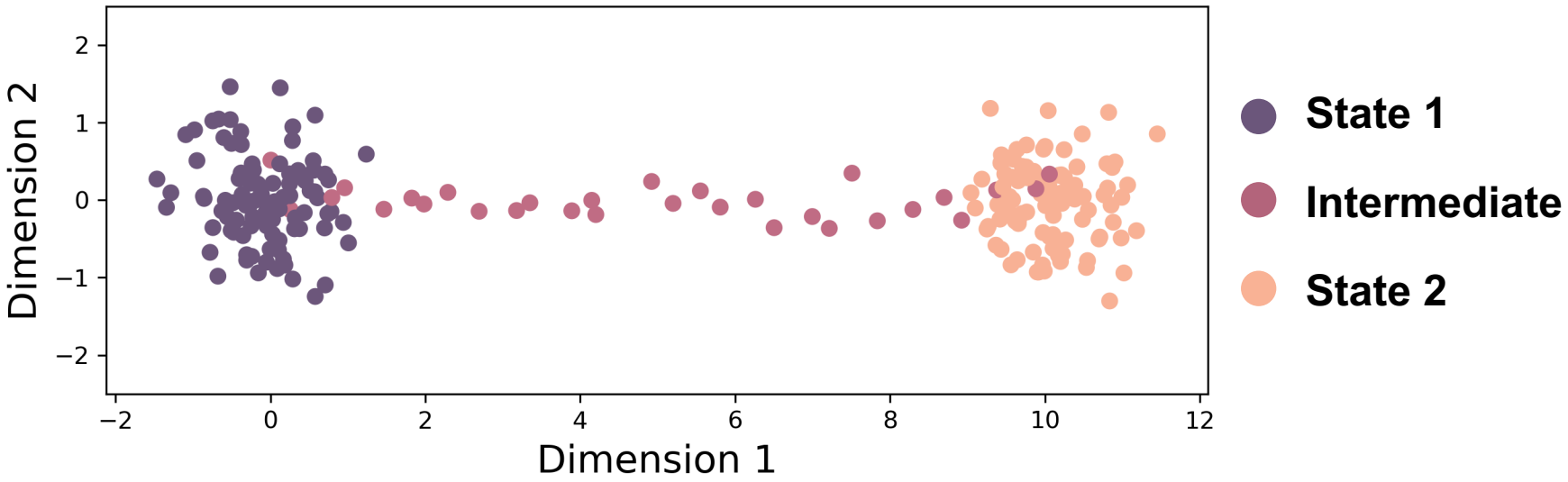
- What genes are expressed in each population?
- What genes change the most from State 1 -> State 2?
- What is the ordering of gene expression?
- How do gene-gene relationships change across states?

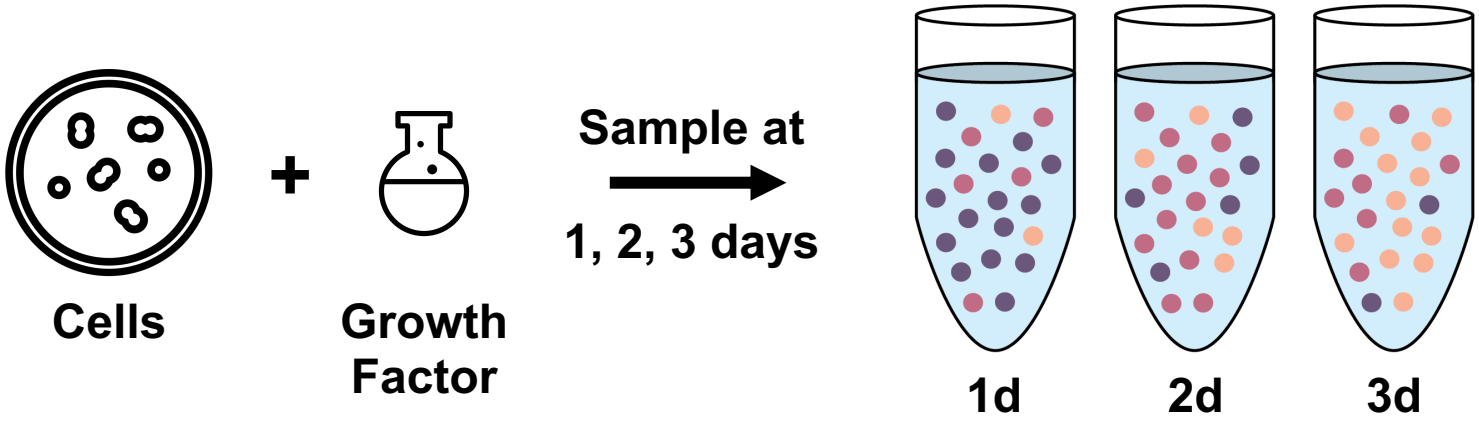
A simple developmental system



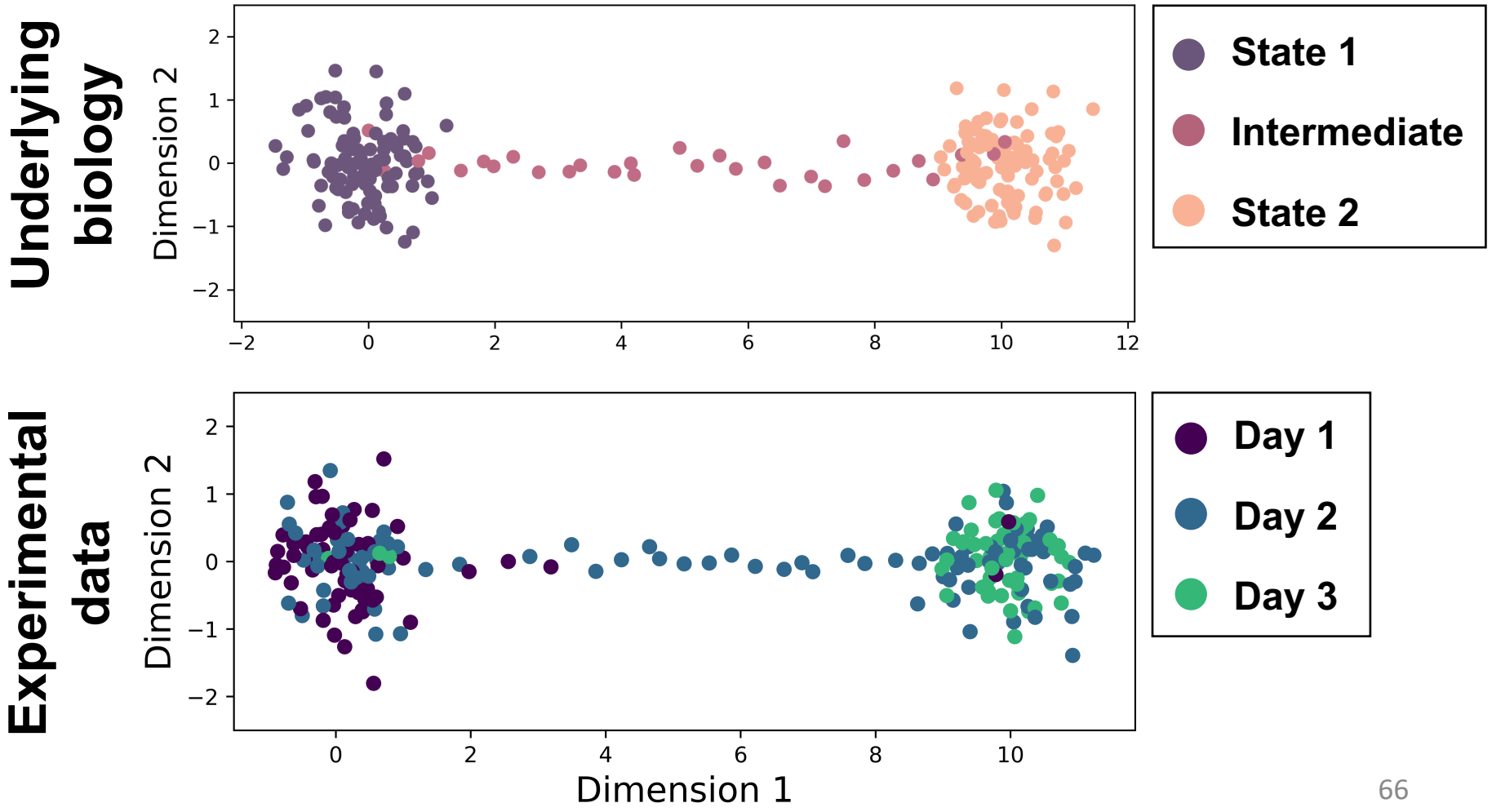


A simple developmental system





We must infer underlying biology from experimental observations

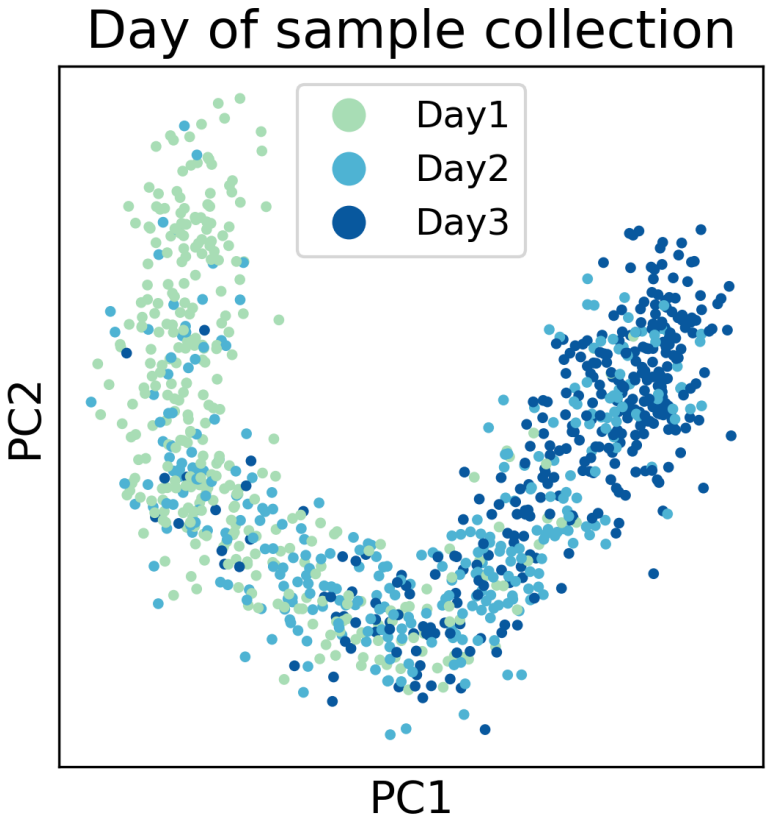


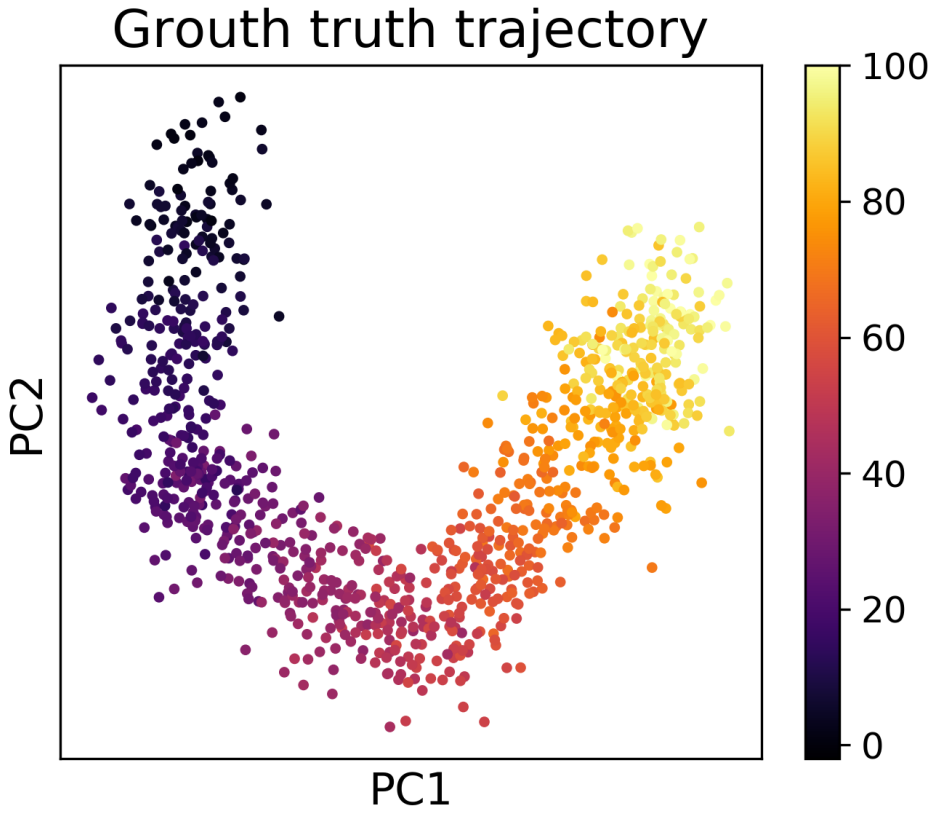
The goal of trajectory inference (TI) is to infer a temporal ordering of cells

Summary of 45 TI methods

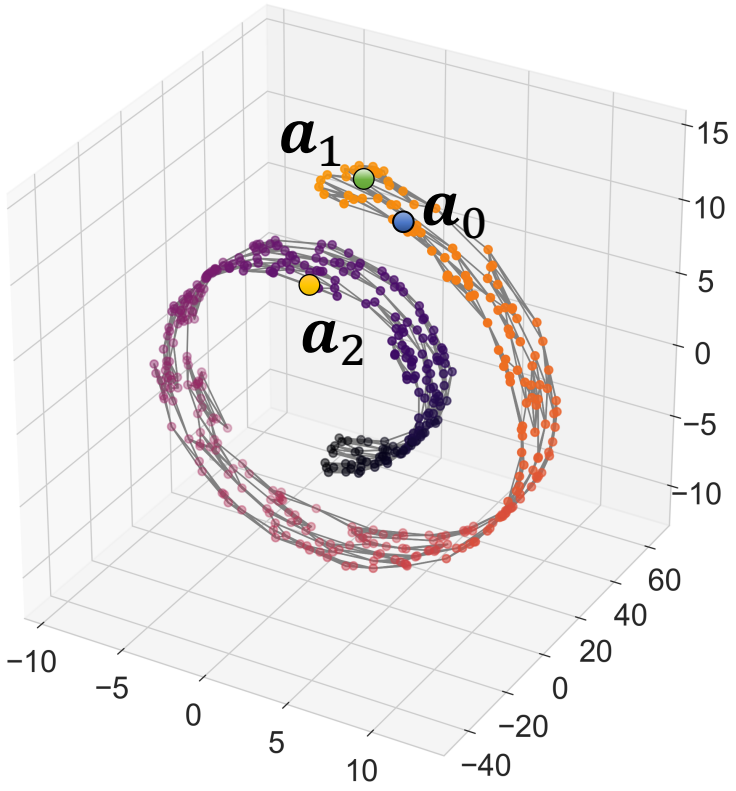
dynverse.org

Learning pseudotime via graph walks

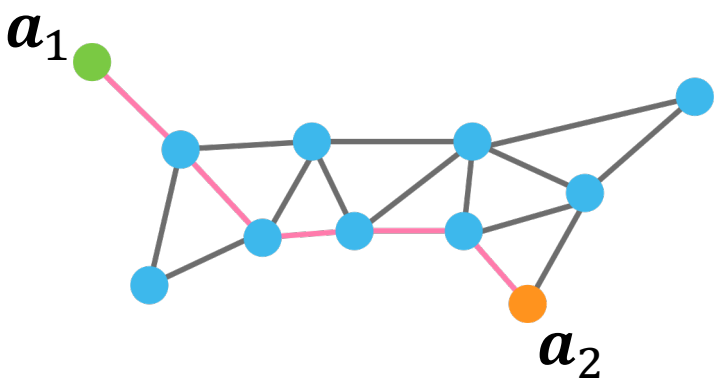




Graph walks approximate manifold distances

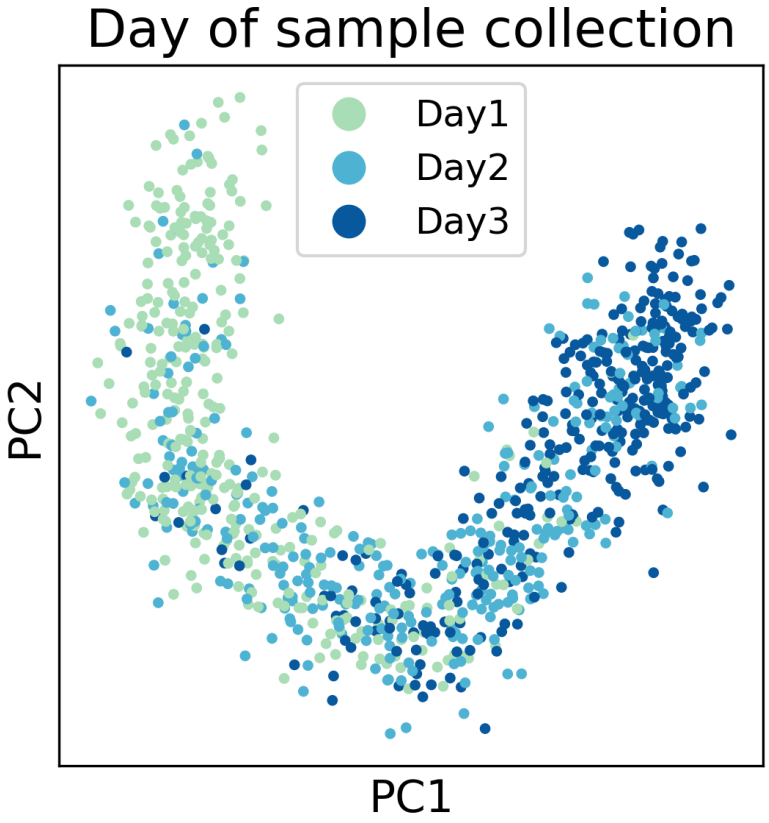


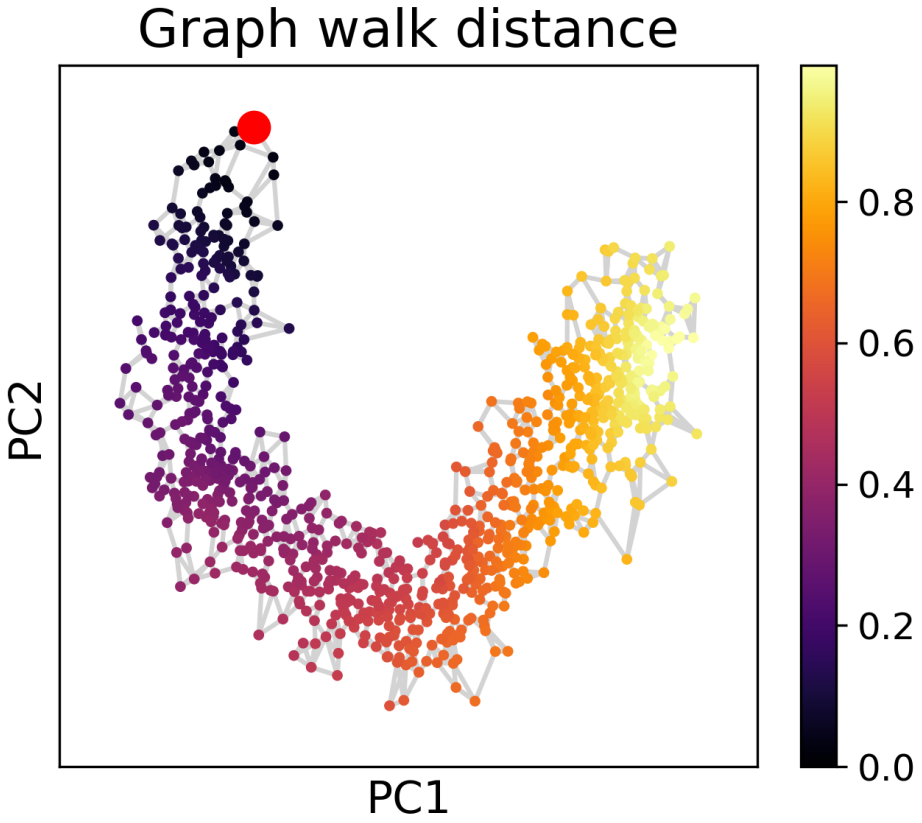
Shortest path between points



$$d_{geodesic}(a_1, a_2) = \text{shortestpath}(a_1, a_2)$$
$$= 5$$

Learning pseudotime via graph walks

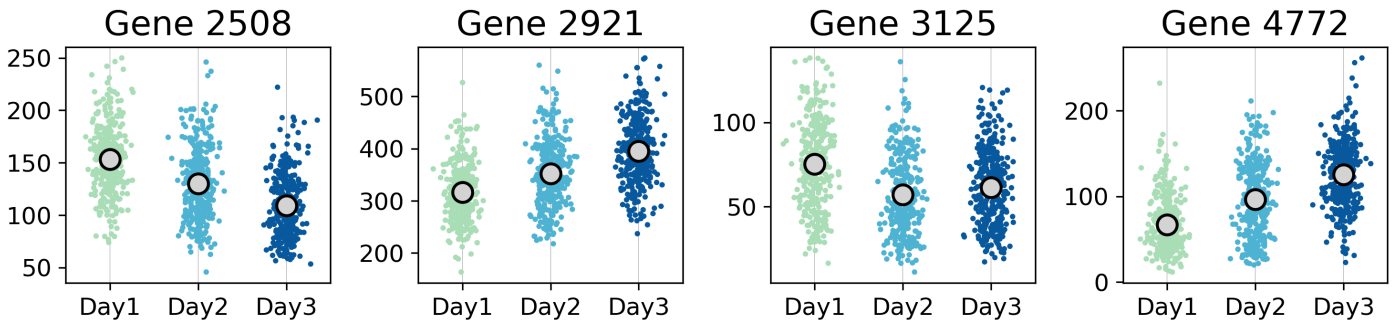




Pseudotime facilitates analysis of genes driving cellular progression

Ordering

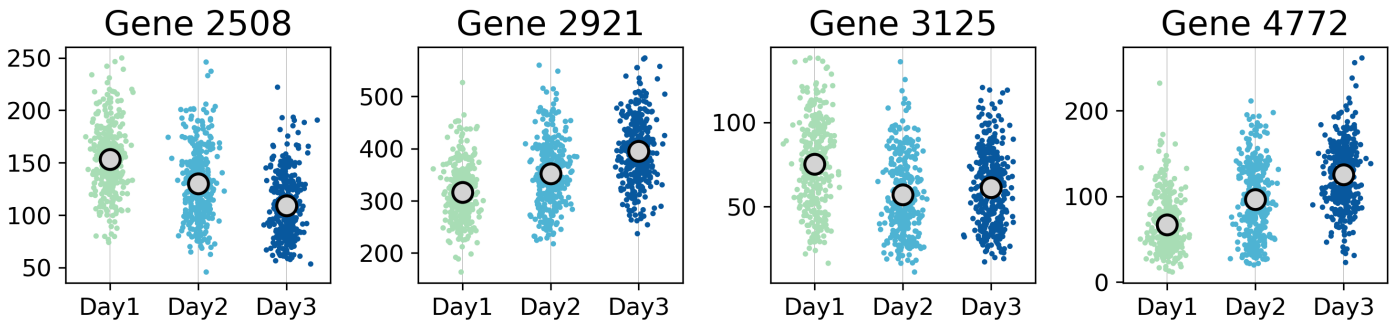
Sample



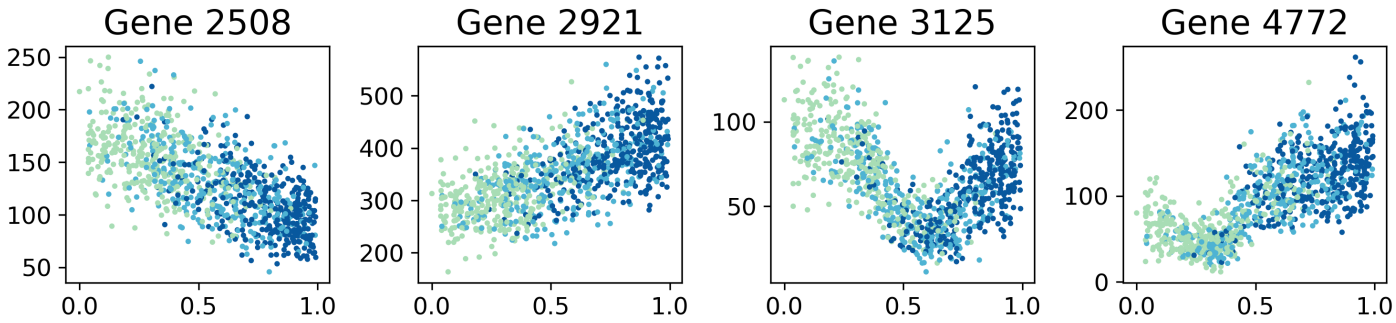
Pseudotime facilitates analysis of genes driving cellular progression

Ordering

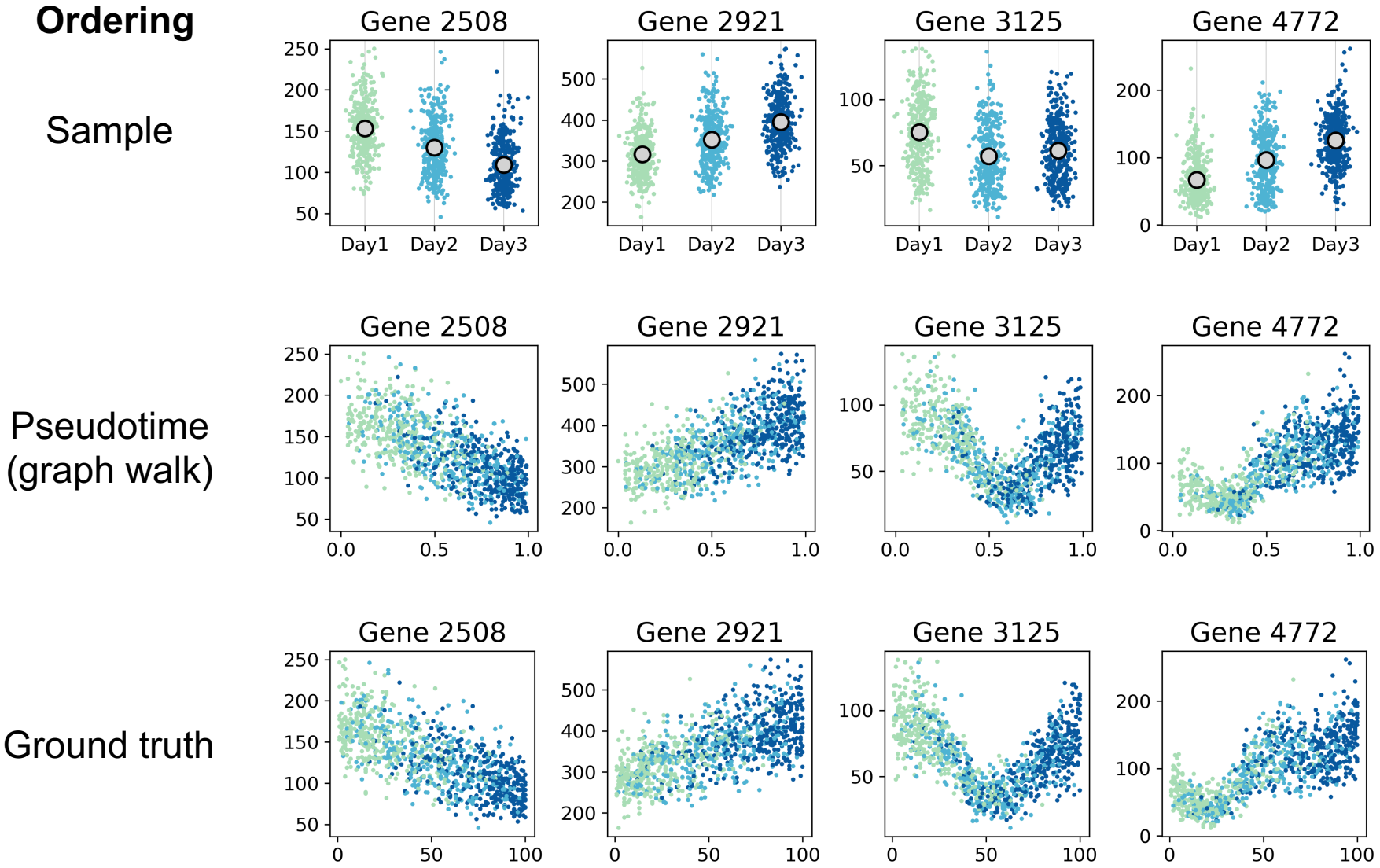
Sample



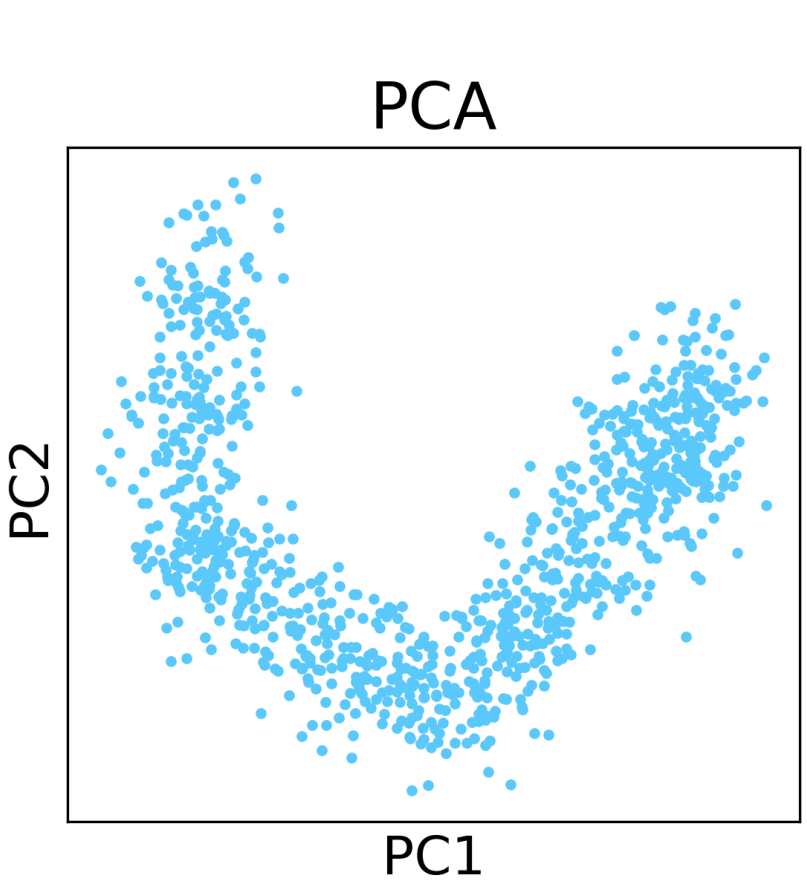
Pseudotime (graph walk)

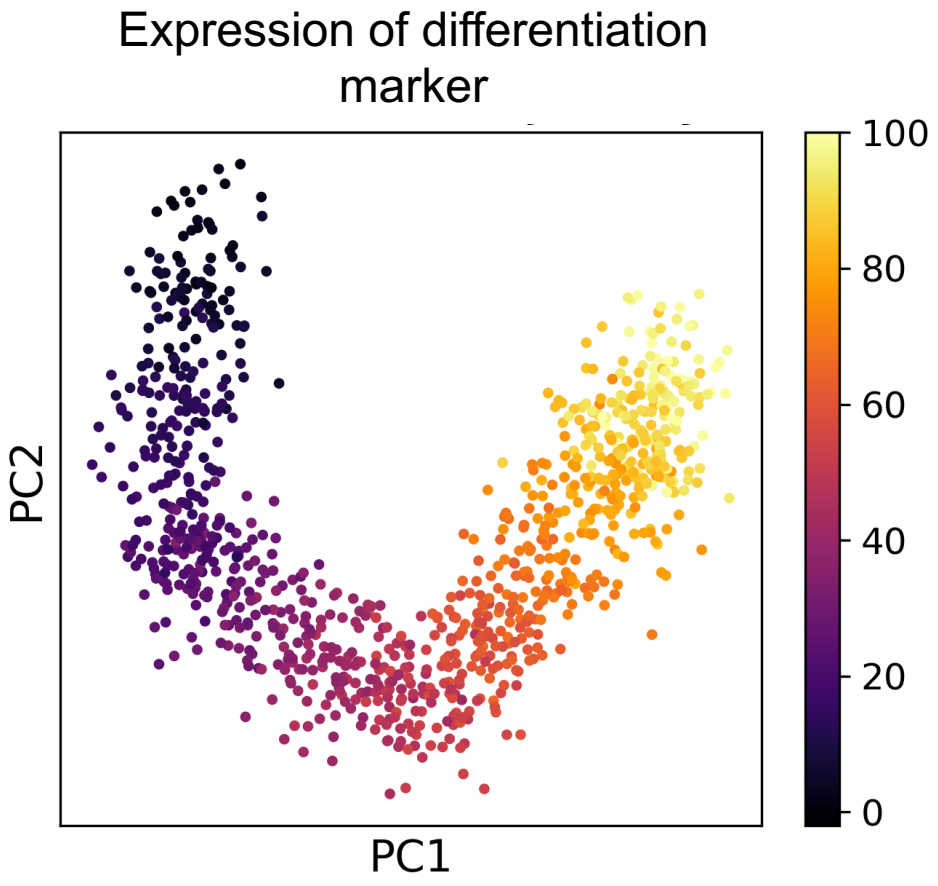


Pseudotime facilitates analysis of genes driving cellular progression



Note: you do not need multiple time points for pseudotime analysis





Diffusion pseudotime – from graph distance to random walks

Euclidean distance between two vectors

Sum over values of *t*

$$\mathrm{dpt}(x,y) = ||\, \boldsymbol{M}(x,.) - \boldsymbol{M}(y,.)\,||,\ \boldsymbol{M} = \sum_{t=1}^{\infty} \tilde{\boldsymbol{T}}^{t}.$$

$$\boldsymbol{M} = \sum_{t=1}^{\infty} \tilde{\boldsymbol{T}}^{t} = (I - \tilde{\boldsymbol{T}})^{-1} - I \text{ where } \tilde{\boldsymbol{T}} = \boldsymbol{T} - \psi_0 \psi_0^{T},$$

ψ is used for eigenvectors
ψ_0 represents starting probabilities for T

Provides a calculatable solution to the infinite series

T is the random walk matrix

I is the identity matrix with 1's on the diagonal

$$\begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}$$

Diffusion pseudotime – from graph distance to random walks

Euclidean distance between two vectors

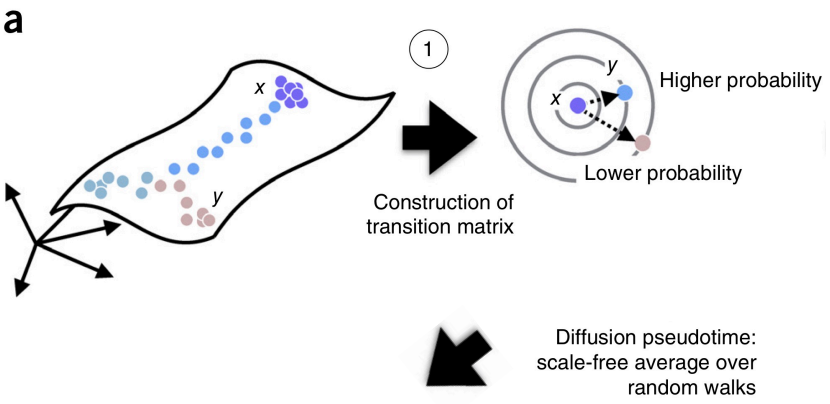
Sum over values of t

$$\mathrm{dpt}(x,y) = ||\, \boldsymbol{M}(x,.) - \boldsymbol{M}(y,.)\,||,\ \boldsymbol{M} = \sum_{t=1}^{\infty} \tilde{\boldsymbol{T}}^t.$$

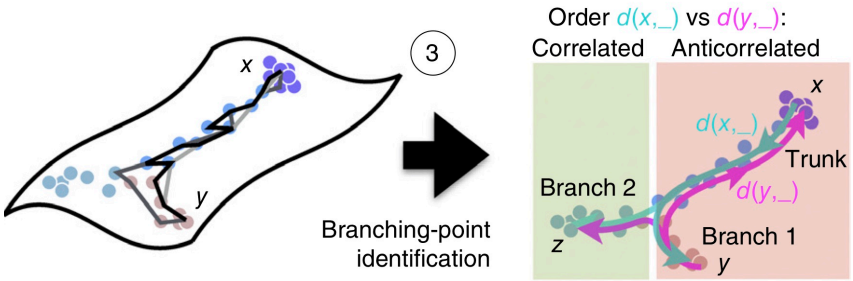
$$\boldsymbol{M} = \sum_{t=1}^{\infty} \tilde{\boldsymbol{T}}^t = (I - \tilde{\boldsymbol{T}})^{-1} - I \text{ where } \tilde{\boldsymbol{T}} = \boldsymbol{T} - \psi_0 \psi_0^T,$$

Provides a calculatable solution to the infinite series

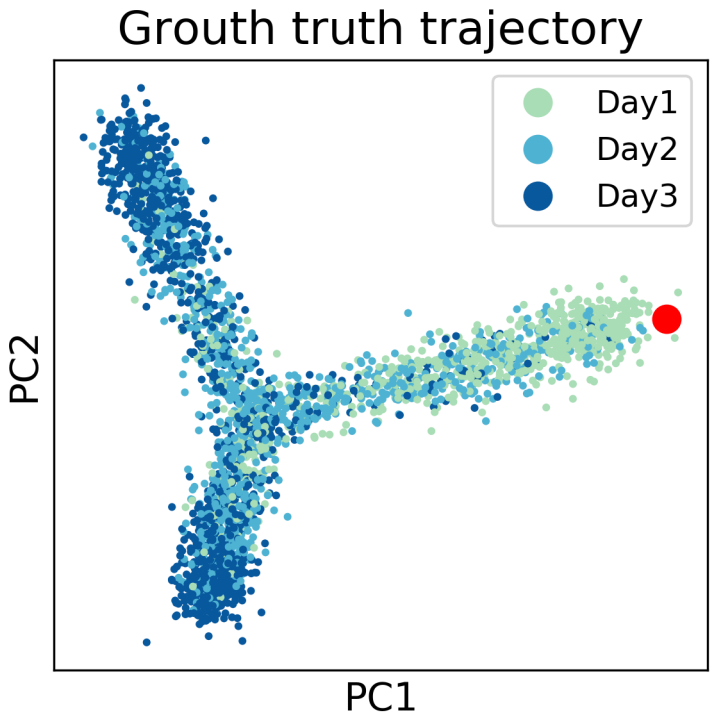
Transition probabilities give data geometry

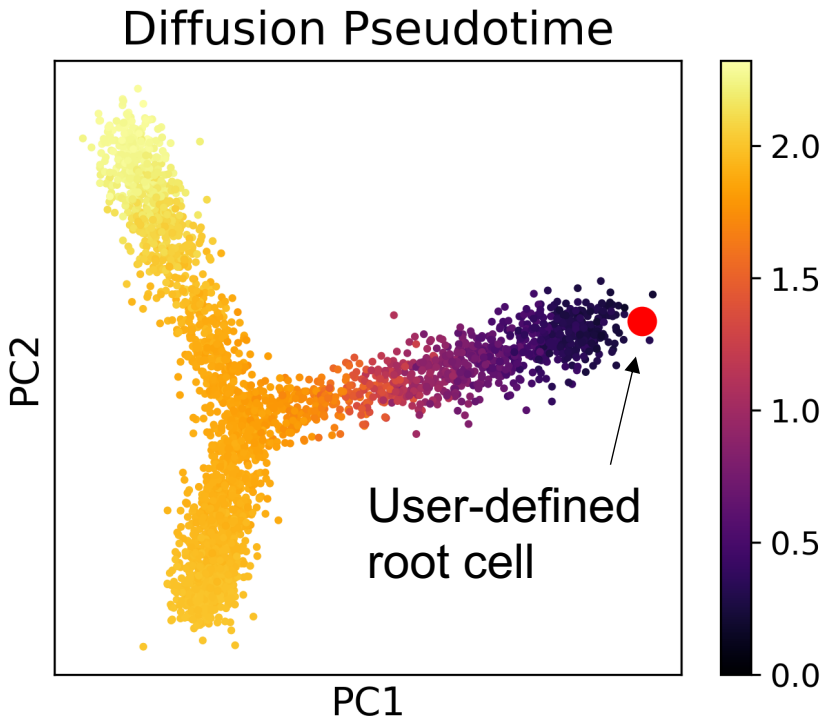


Correlation reveals branches

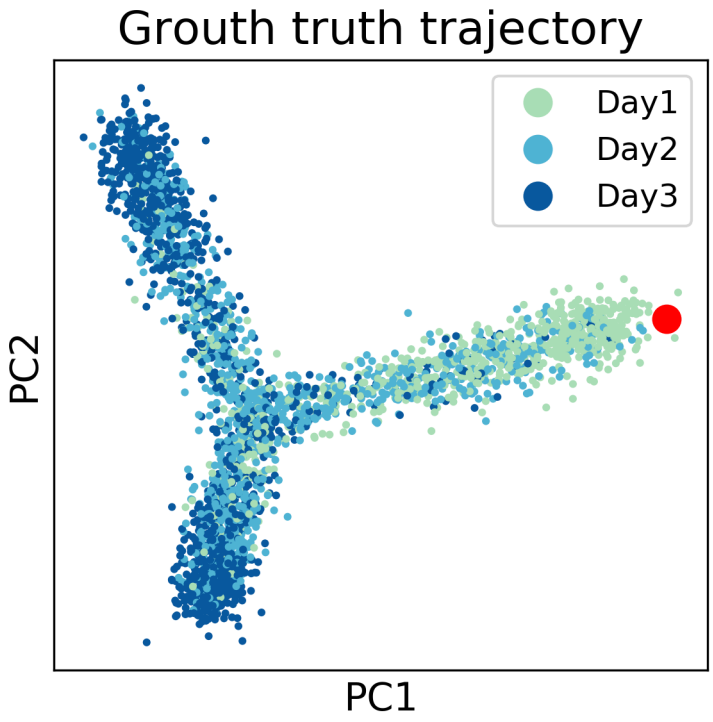


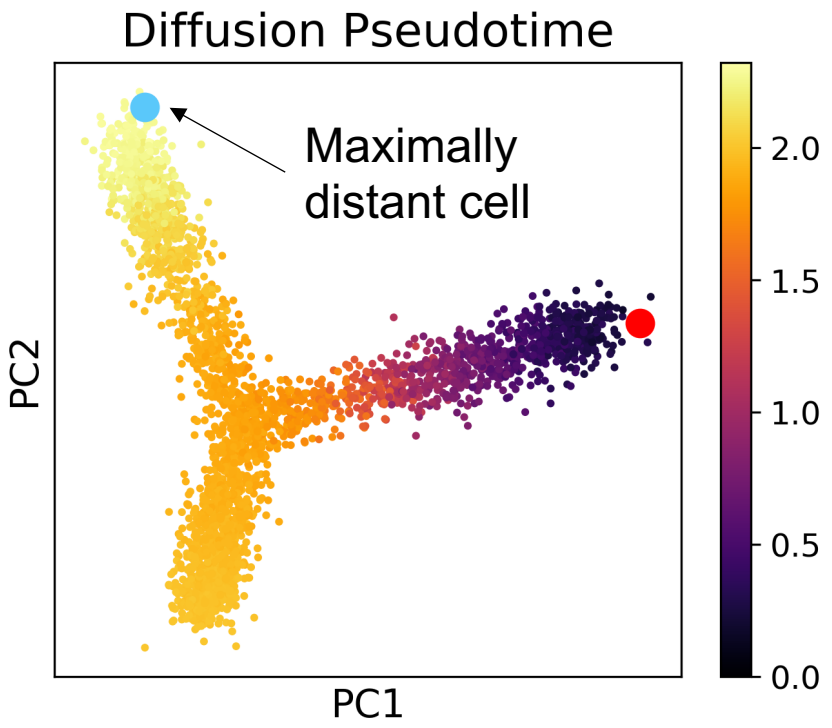
Using DPT to detect branches



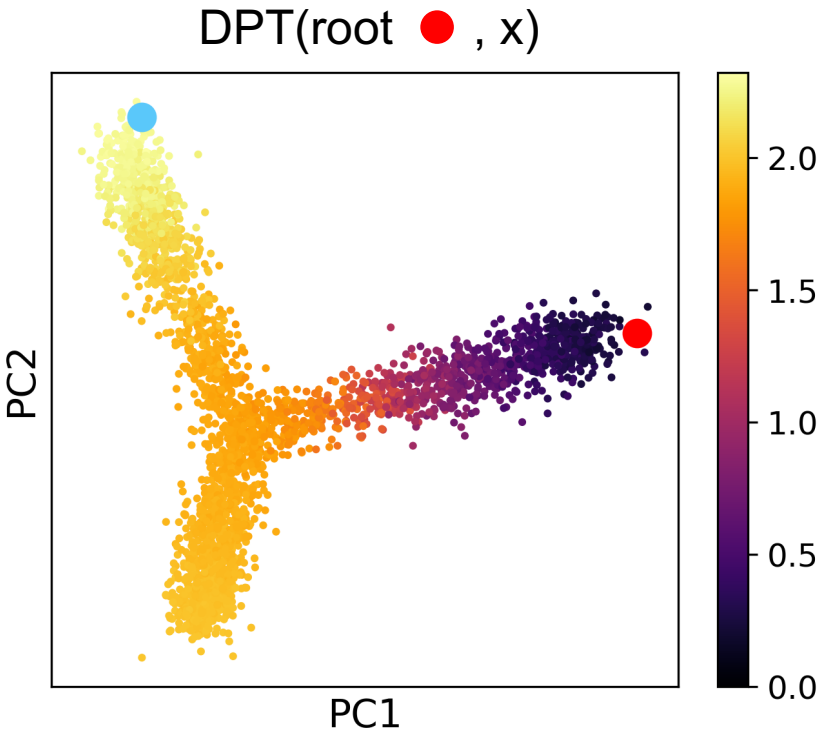


Using DPT to detect branches

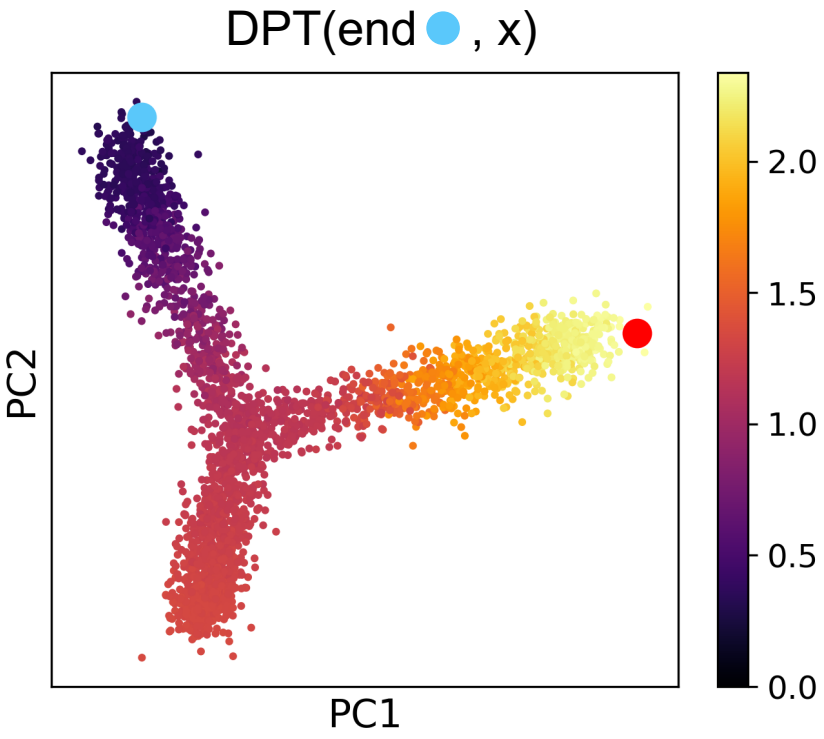




Using DPT to detect branches



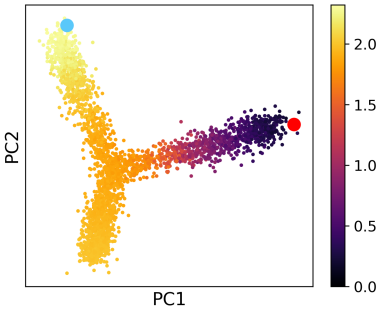
“Forward pseudotime”



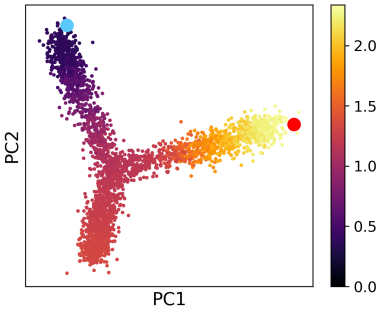
“Backward pseudotime”

Using DPT to detect branches

DPT(root ● , x)

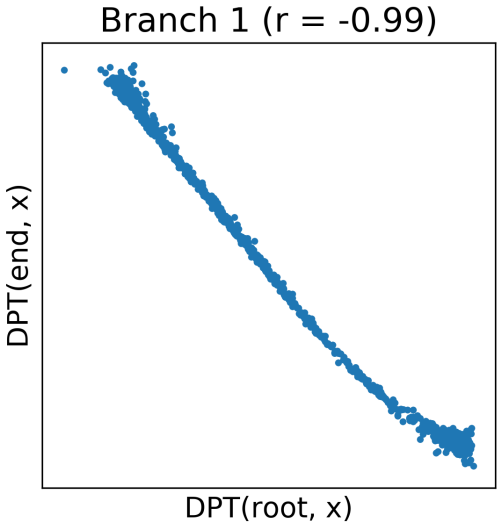


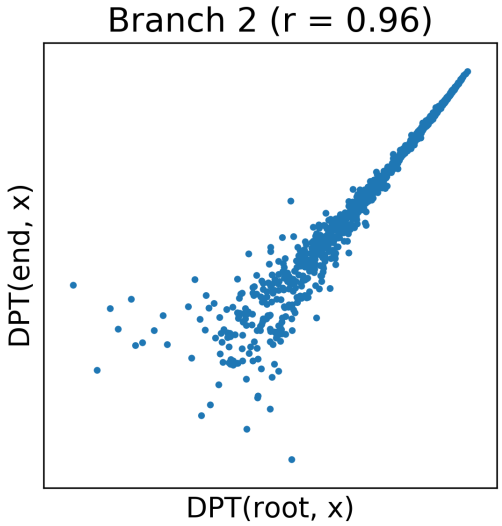
DPT(end ● , x)

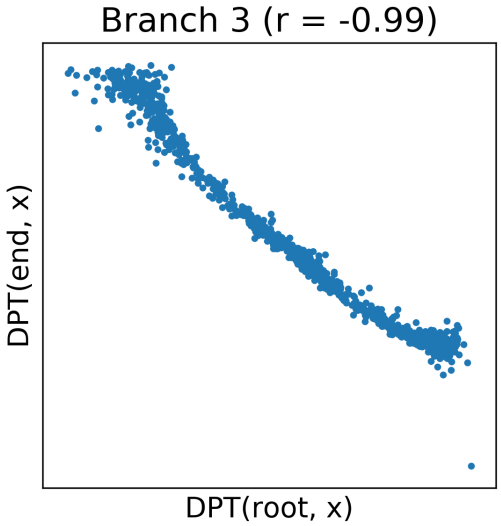


Cells on the same “branch” as the root and end cell have negative DPT correlation.

Cells on different branches have positive DPT correlation

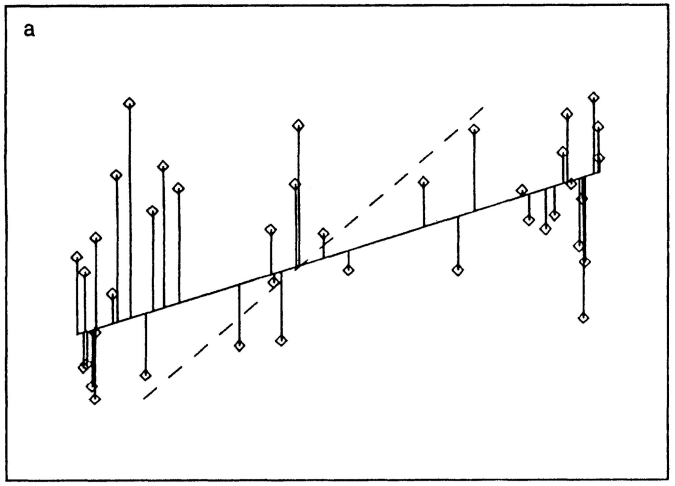




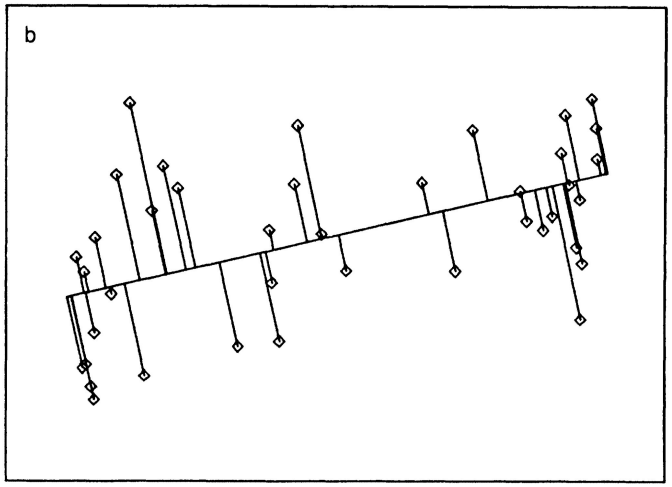


Principal curves – non-linear PCA

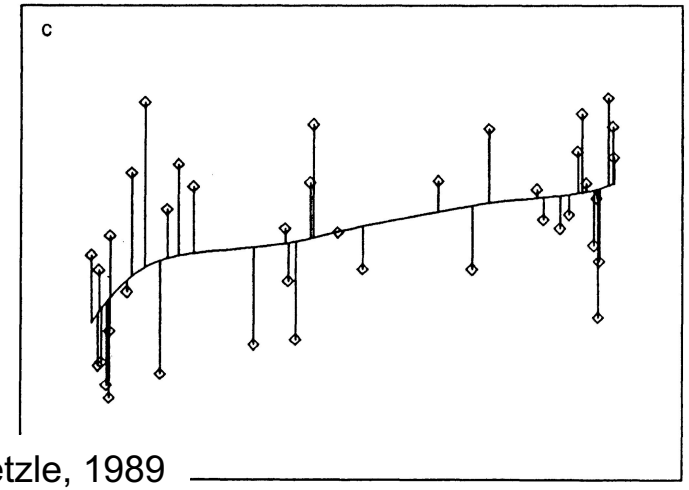
Linear Regression



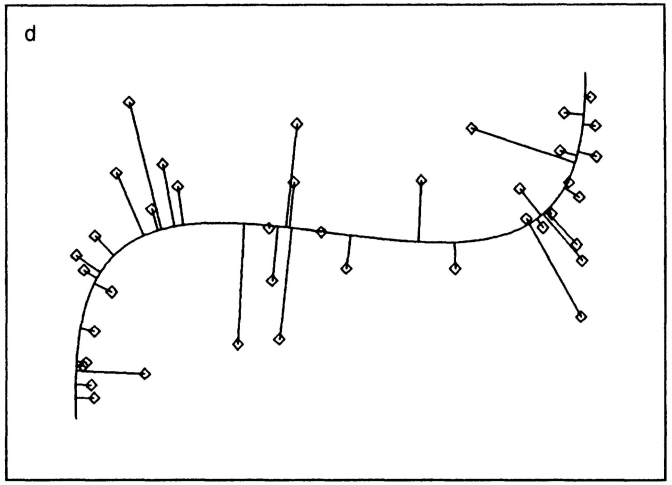
PCA



Non-linear Regression

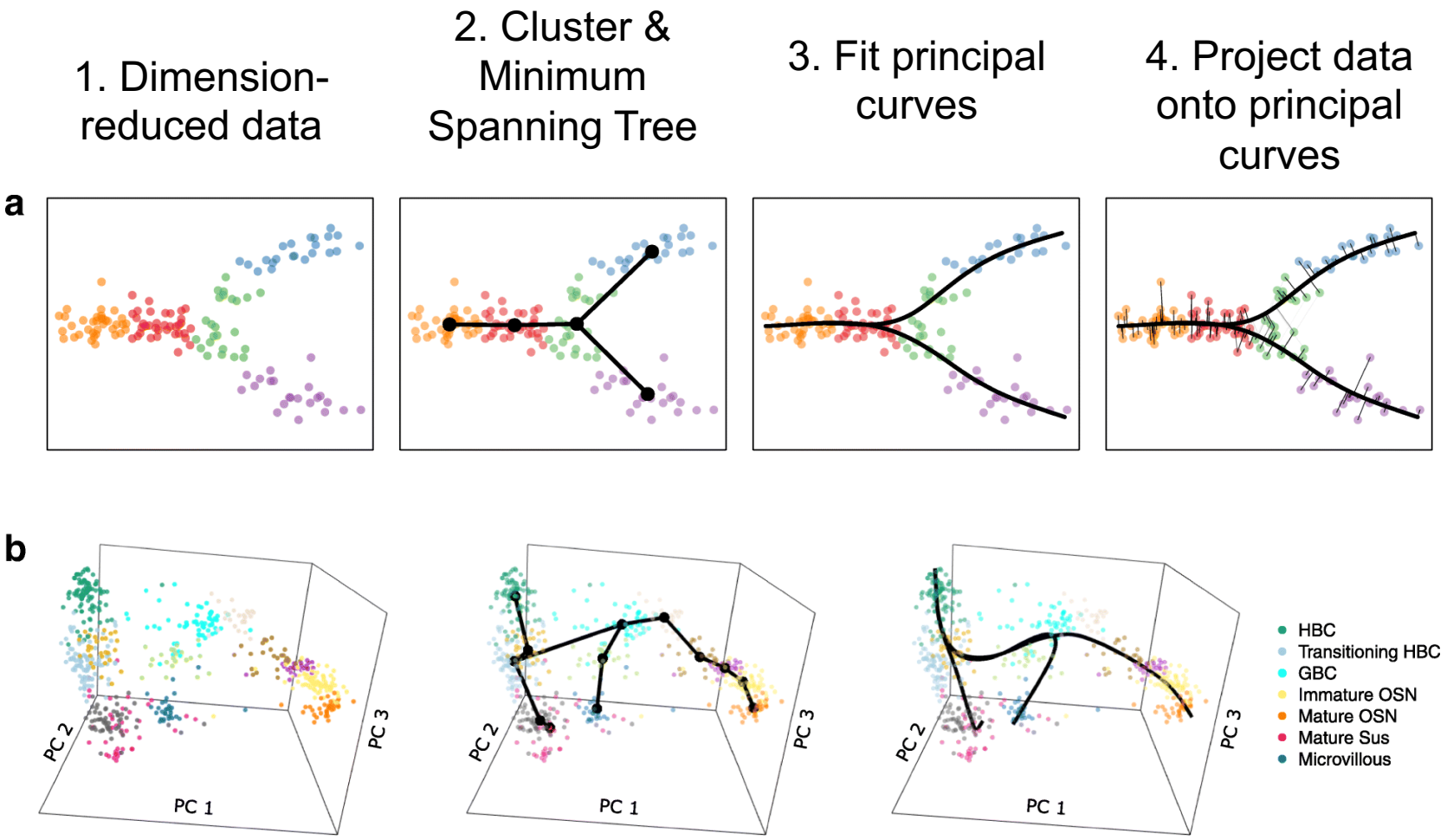


Principal Curves

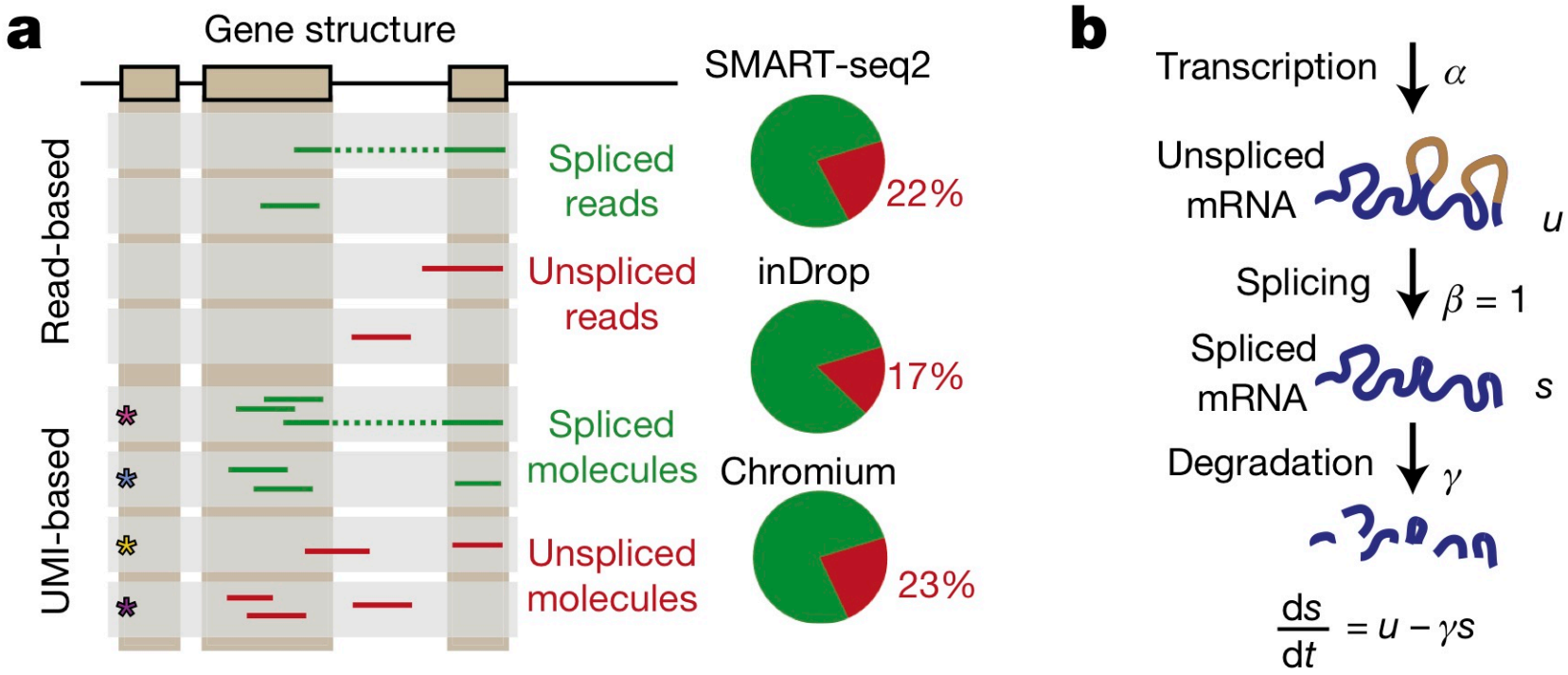


Hastie and Stuetzle, 1989

Slingshot - Street et al. 2018

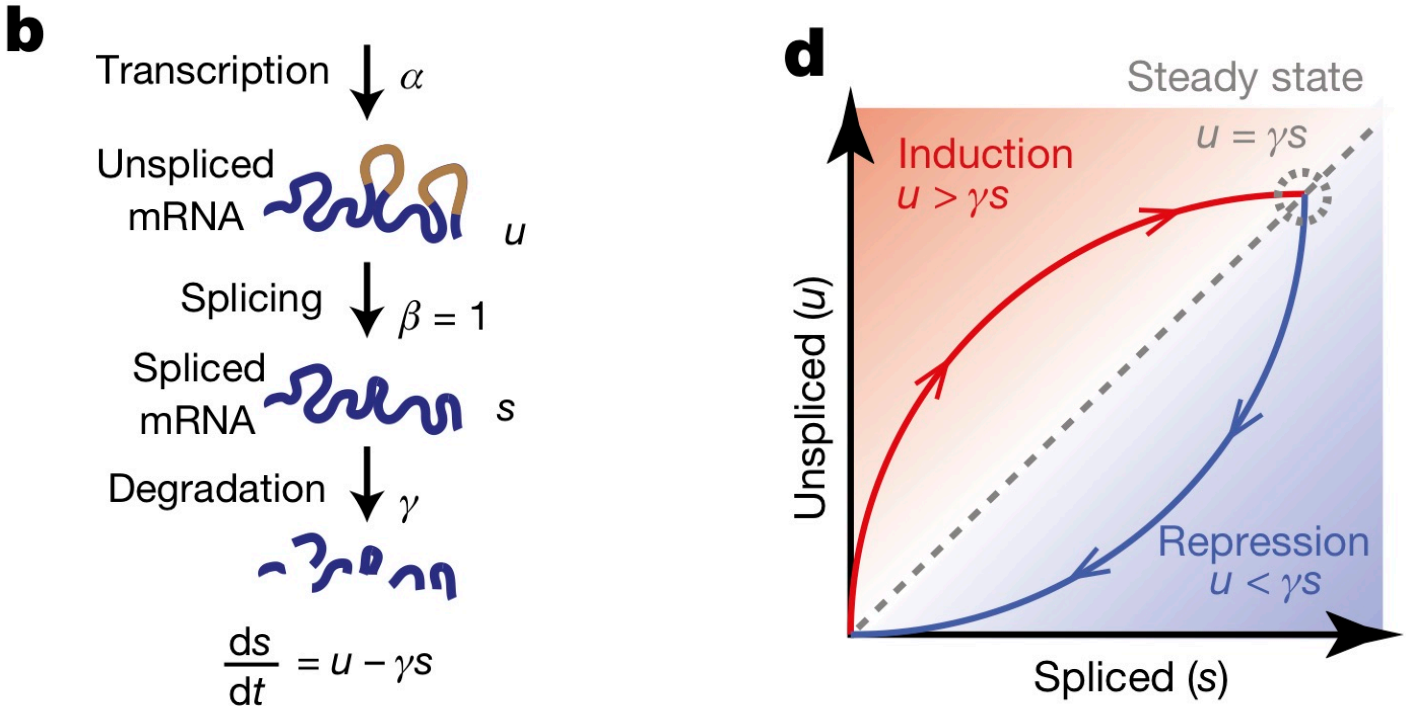


RNA Velocity: using gene splicing genes to infer future cell state



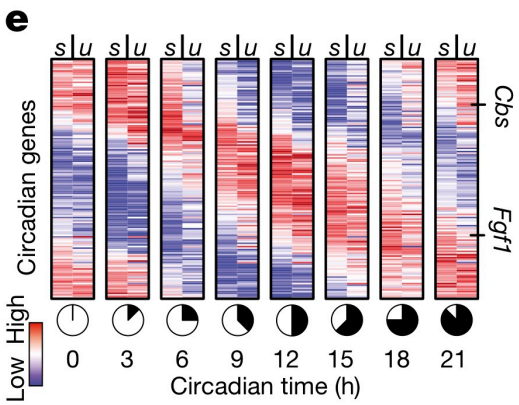
Le Manno, et al. Nature. 2018.

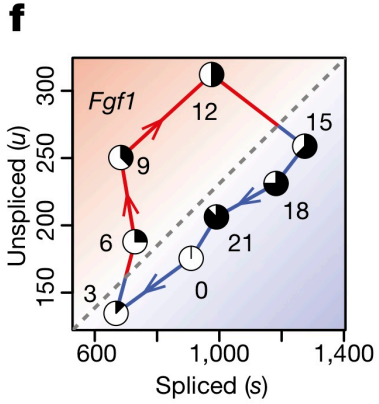
RNA Velocity: using gene splicing genes to infer future cell state

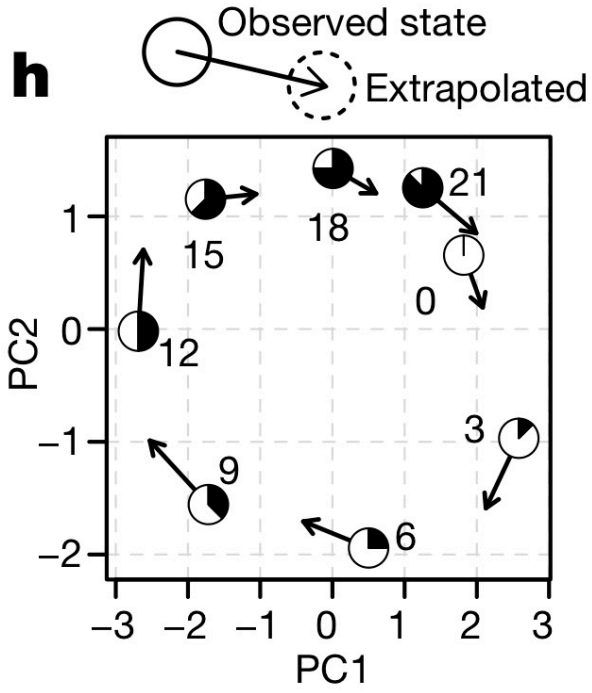


Le Manno, et al. Nature. 2018.

RNA velocity correctly infers circadian rhythm in mouse liver

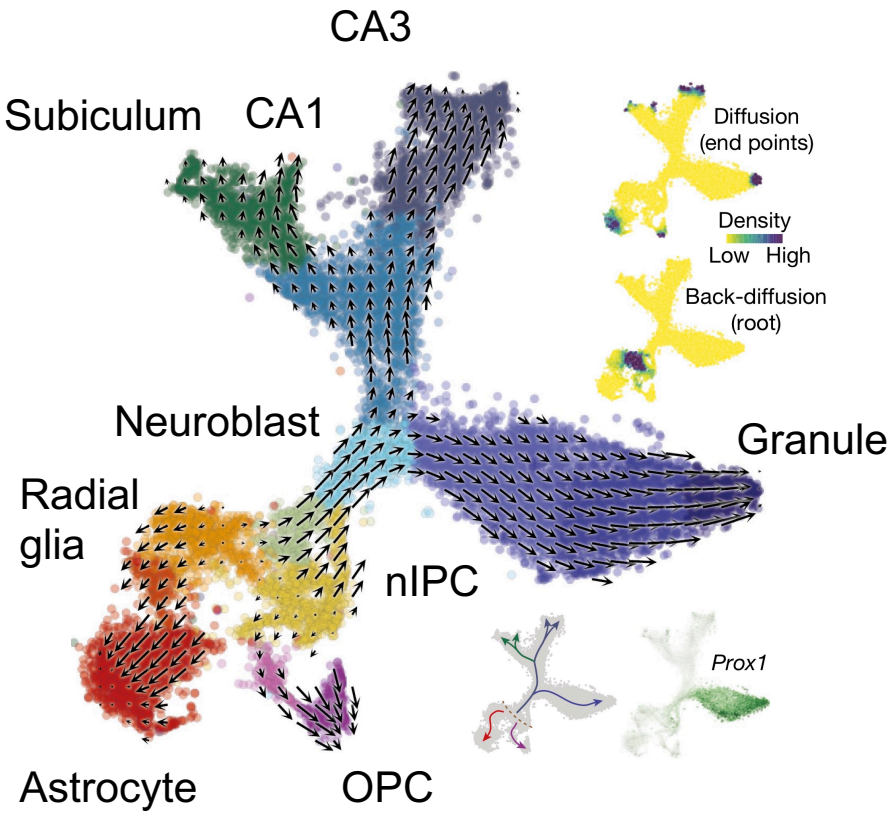




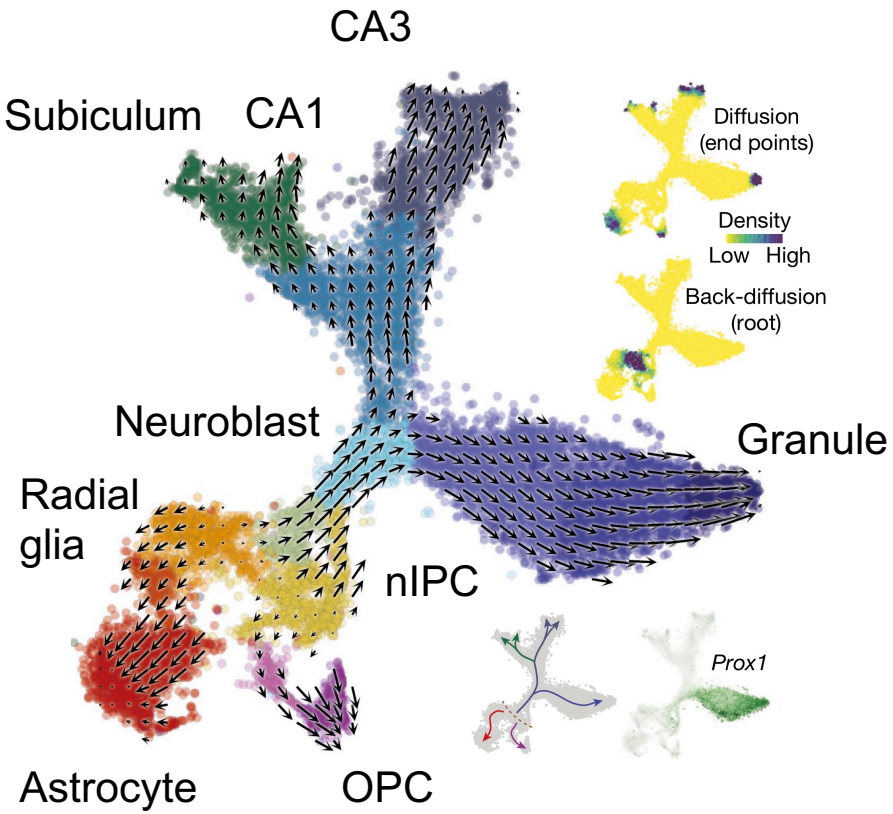


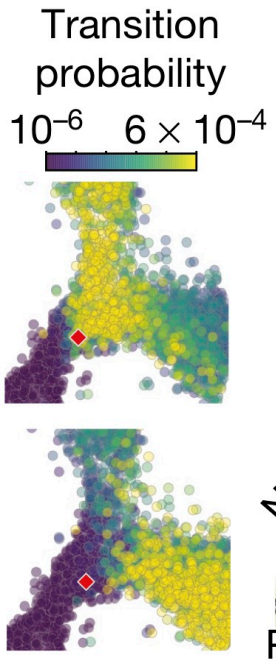
mRNA was measured in bulk over 24 hours in the mouse liver

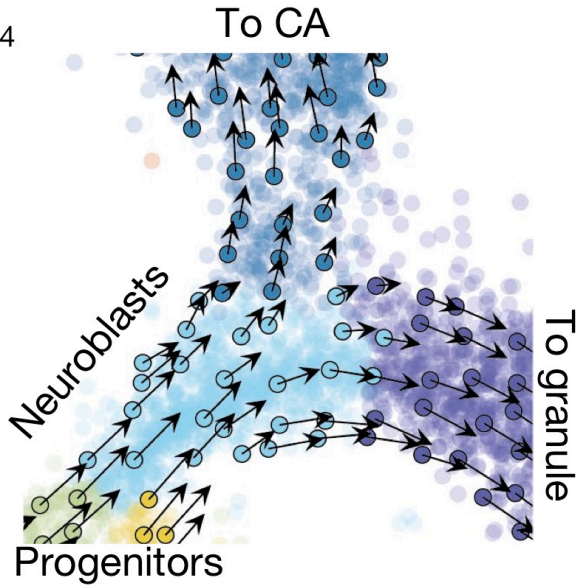
Neural fate decisions in the mouse hippocampus



Neural fate decisions in the mouse hippocampus







Conclusions

1. Pseudotime identifies an ordering of cells that match the developmental progression of gene expression

2. Pseudotime can be identified from a single timepoint

3. Methods for inferring pseudotime use graph walks or identify axes through the data

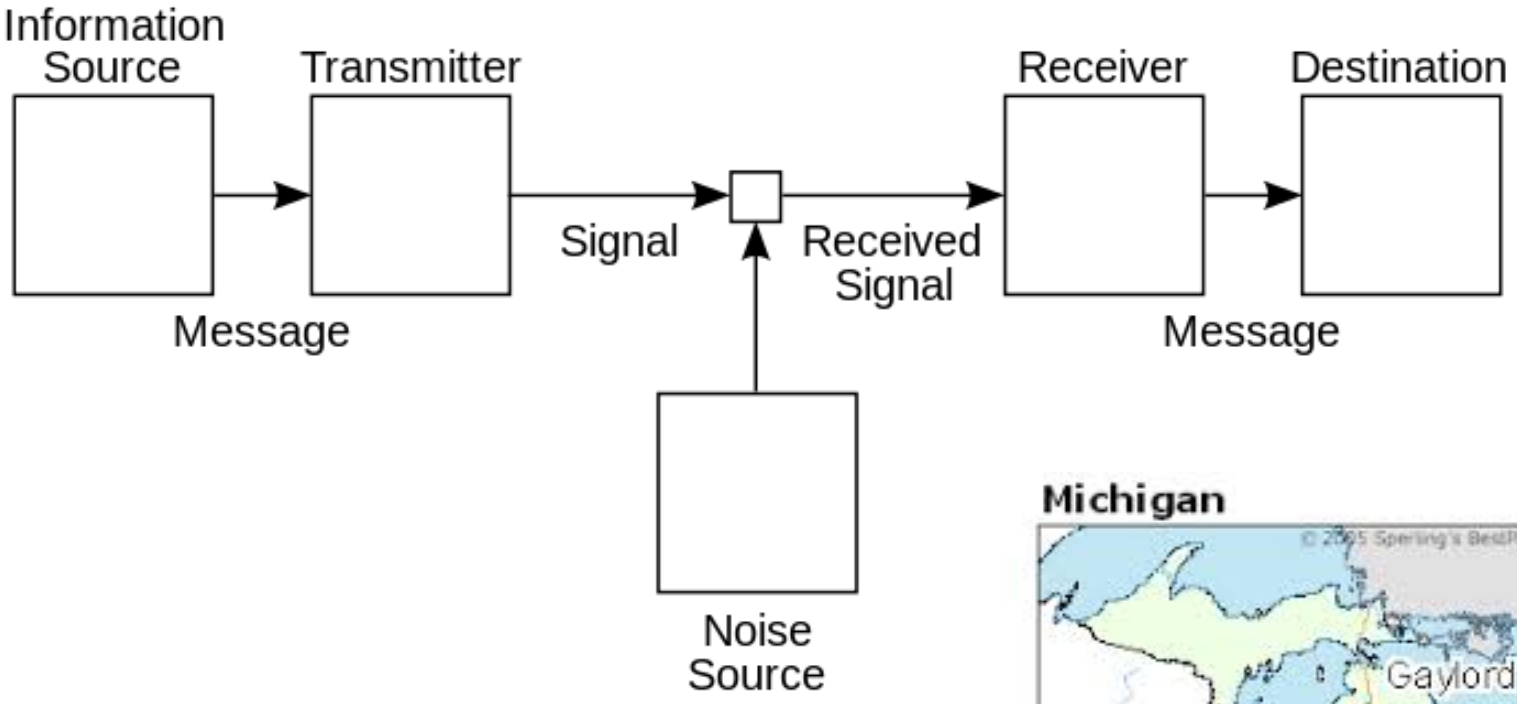
4. A comprehensive comparison of trajectory inference methods is available as dynverse.org

Learning gene-gene relationships

Entropy

- Average amount of information produced by a source of stochasticity
- How much information is produced by a fair coin?

Claude Shannon 1948



" A Mathematical Theory of Communication"

Introduced the term bit

Entropy formulation

- P(heads) = ½ , P(tails)=1/2
- H(coin) = -(1/2) log(1/2) –(1/2)log(1/2) = 1
- Generally

$$H(X) = -\sum_i p(x_i)\log(p(x_i))$$

- When data has lower probability it carries more information (more of a surprise)

Properties

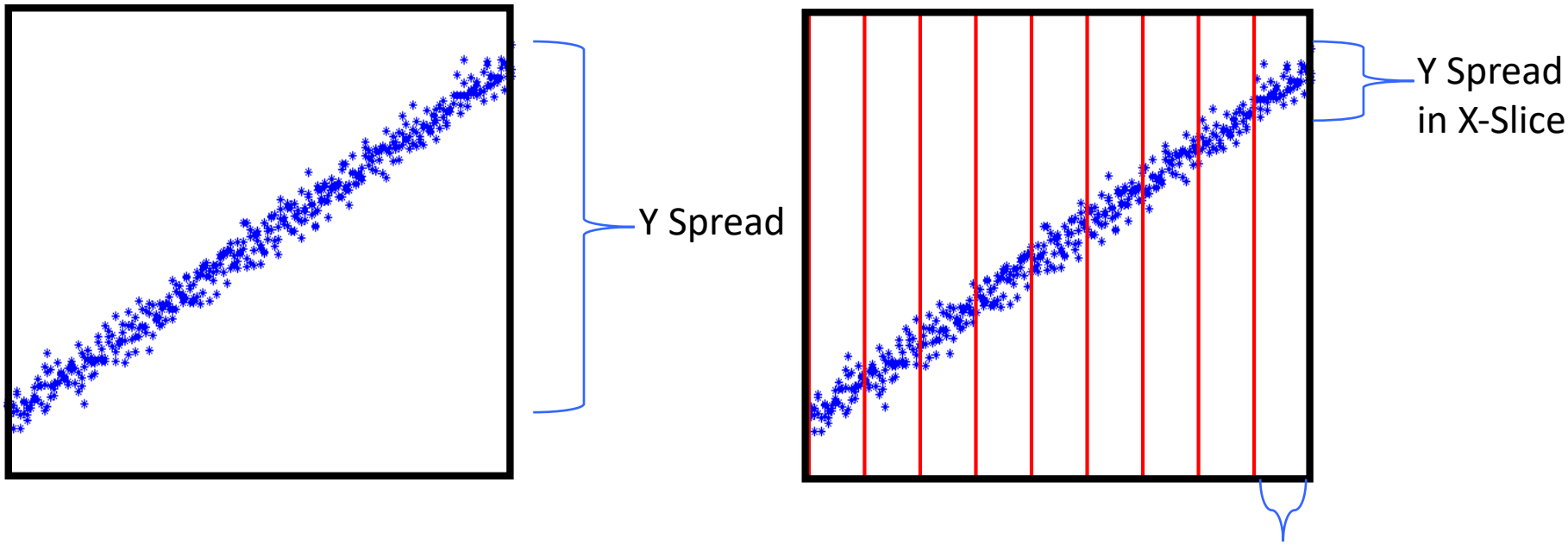
- Maximum at uniform distribution – because here each outcome is equally likely
- Symmetric with respect to probabilities – flipping probabilities of events should not change information
- Continuous – small change in probabilities should result in small change in entropy
- Additive – Independent sources of information should add

Why log?

- Try to define an information function F
  - $F(p)>=0$
  - $F(1)= 0$
  - $F(p,q) = F(p)+F(q)$

- Log satisfies these properties

Entropy

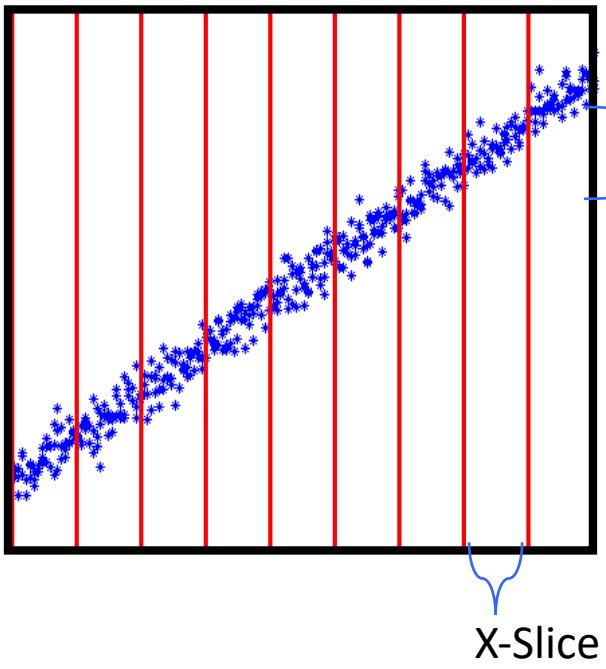


Measure of Uncertainty in a random variable.

$$-\sum_{i=1}^{n} \mathrm{P}(x_i) \log_b \mathrm{P}(x_i),$$

Units of bits tells us how many bits are needed to represent the outcome

Conditional Entropy



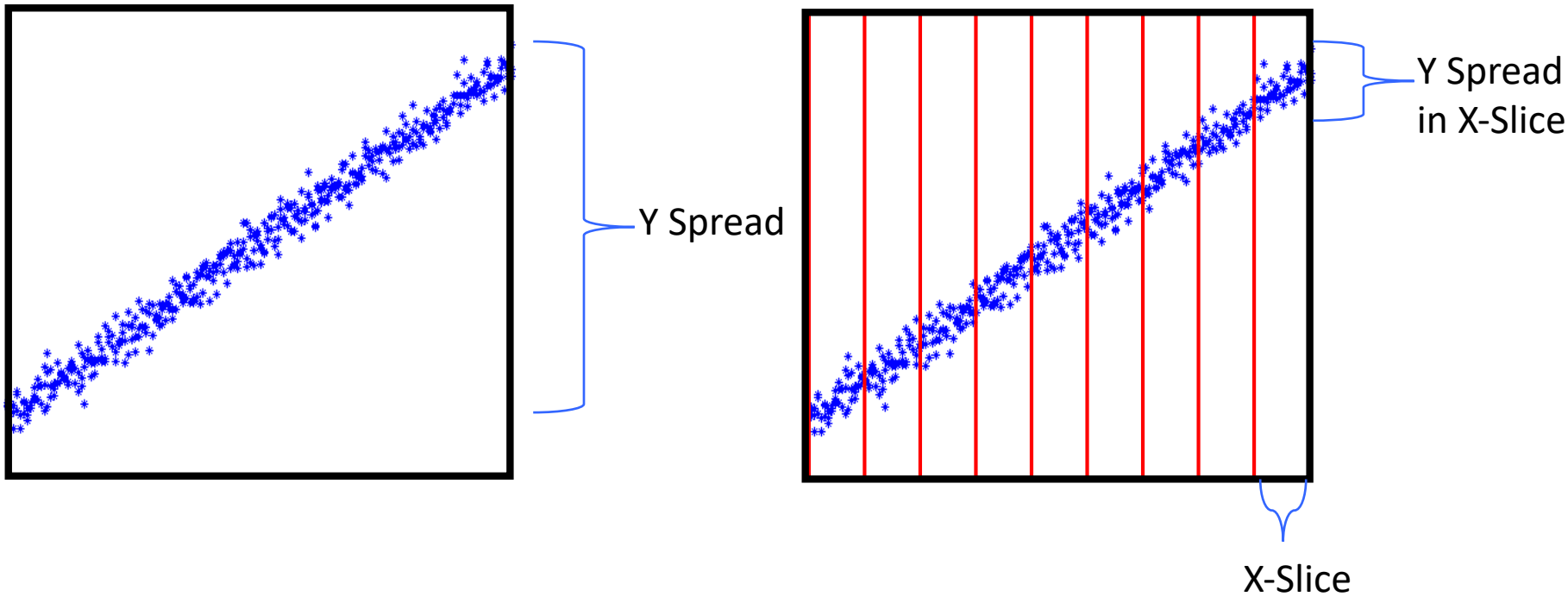
Measure of Uncertainty in a random variable given knowledge about another variable

Can conditioning increase entropy??

No.
H(Y|X)<= H(Y)

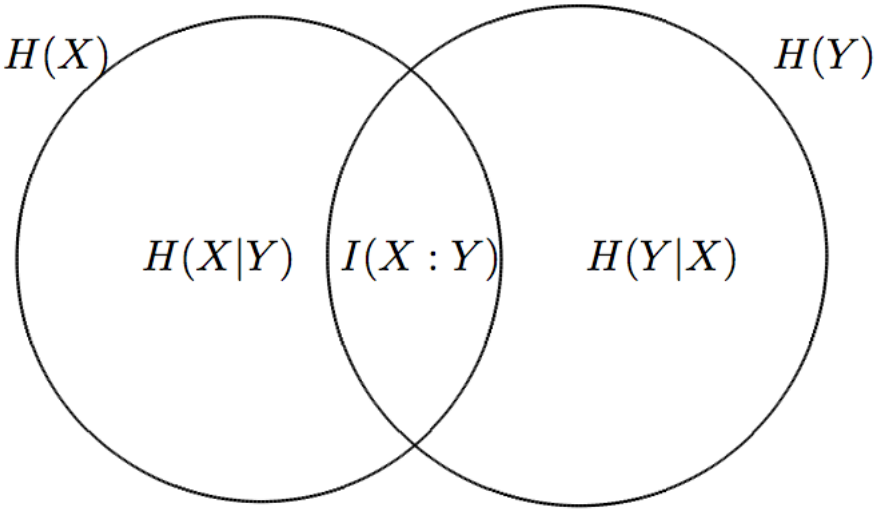
$$
\begin{aligned}
H(Y|X) &\equiv \sum_{x\in\mathcal{X}} p(x)\, H(Y|X=x) \\
&= -\sum_{x\in\mathcal{X}} p(x) \sum_{y\in\mathcal{Y}} p(y|x)\, \log\, p(y|x) \\
&= -\sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p(x,y)\, \log\, p(y|x) \\
&= -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y) \log\, p(y|x) \\
&= -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)}. \\
&= \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y) \log \frac{p(x)}{p(x,y)}.
\end{aligned}
$$

Mutual Information

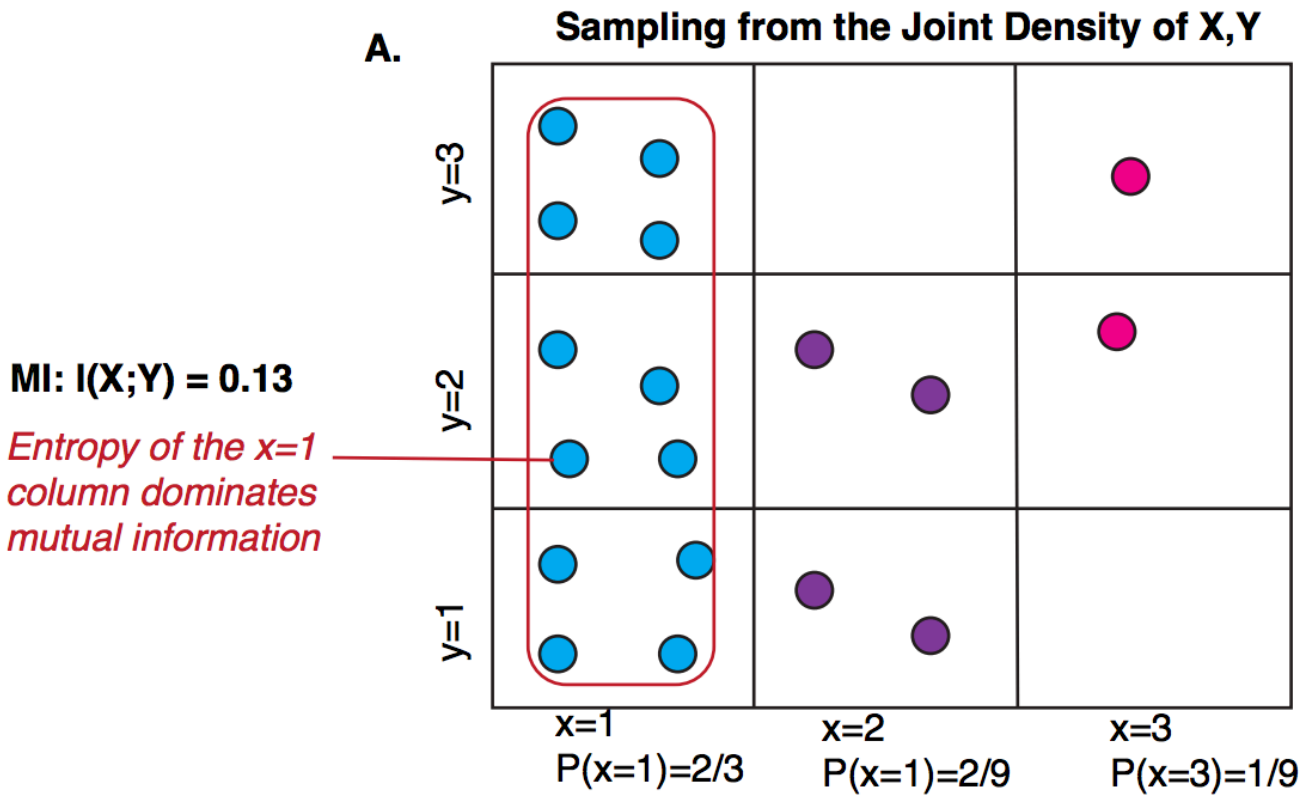


Mutual Information: $I(X,Y) = H(Y) - H(Y|X)$

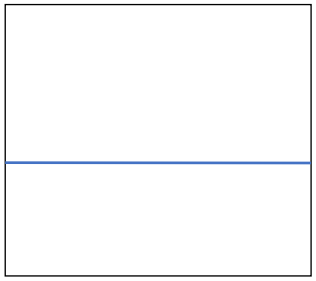
Entropy and Mutual Information

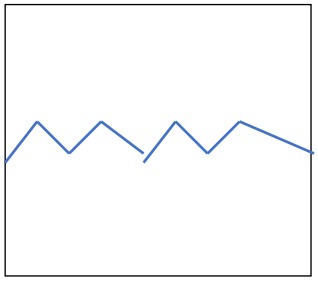


MI Computation

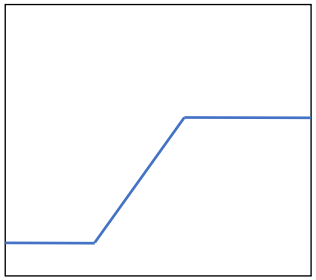


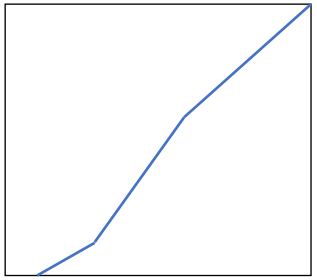
Trends with High and Low MI





Low MI





High MI

Another Definition

- Compares joint to marginal distributions

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)\,p(y)} \right)$$

- What is the difference in entropy between joint and product of marginals ?

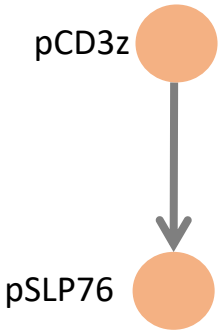
Properties of Mutual Information

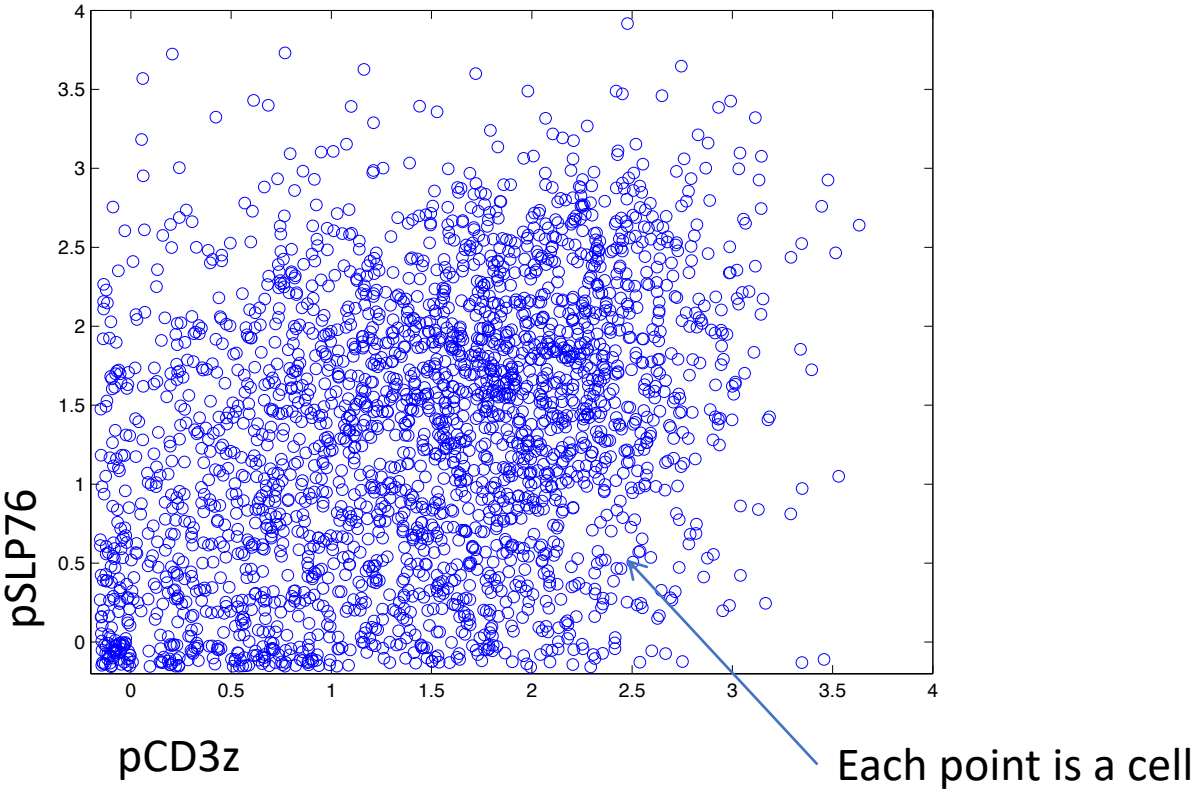
- I(X;Y)>=0
- Data processing inequality:

$$X \rightarrow \boxed{\text{Channel}} \rightarrow Y \rightarrow \boxed{\text{Processing}} \rightarrow Z$$

- I(X;Z)<=I
- Conditioning on a third variable can increase or decrease MI between two variables
  - Why?

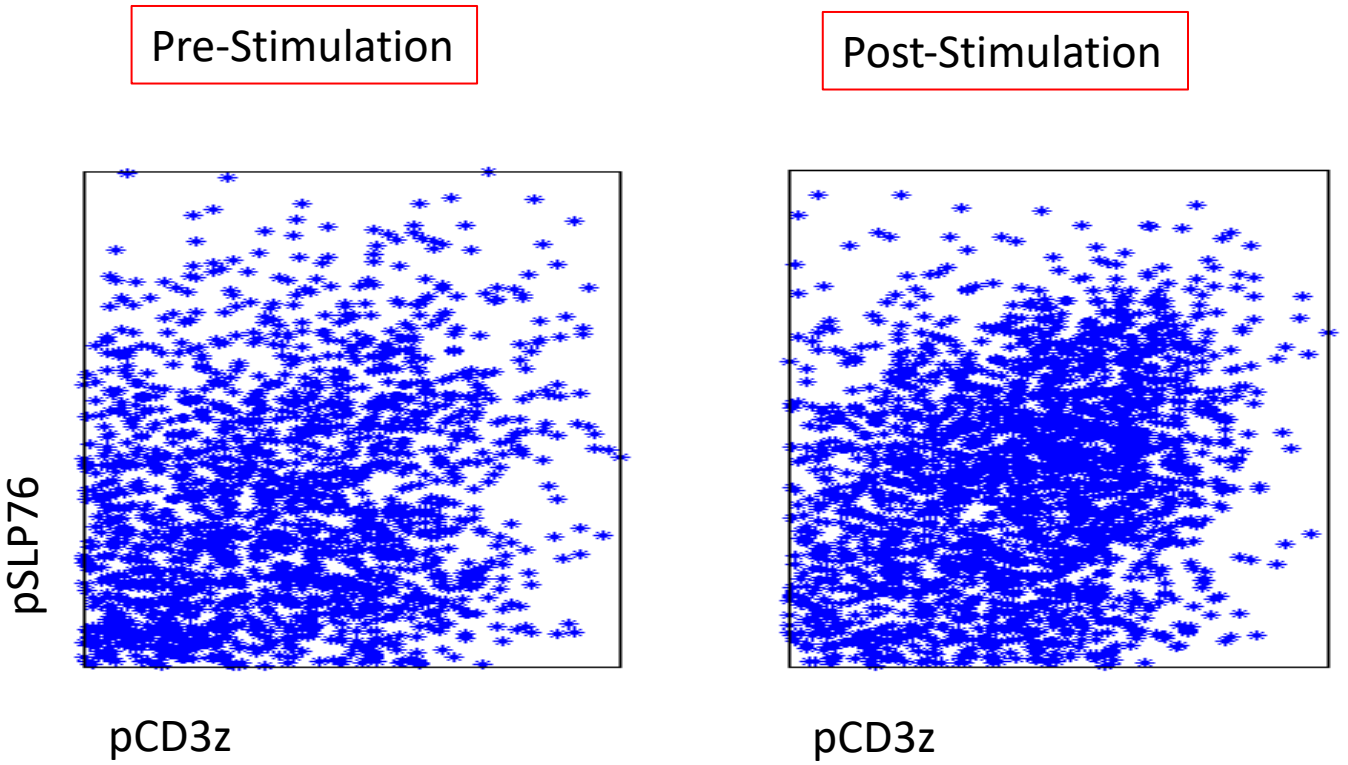
Data distribution





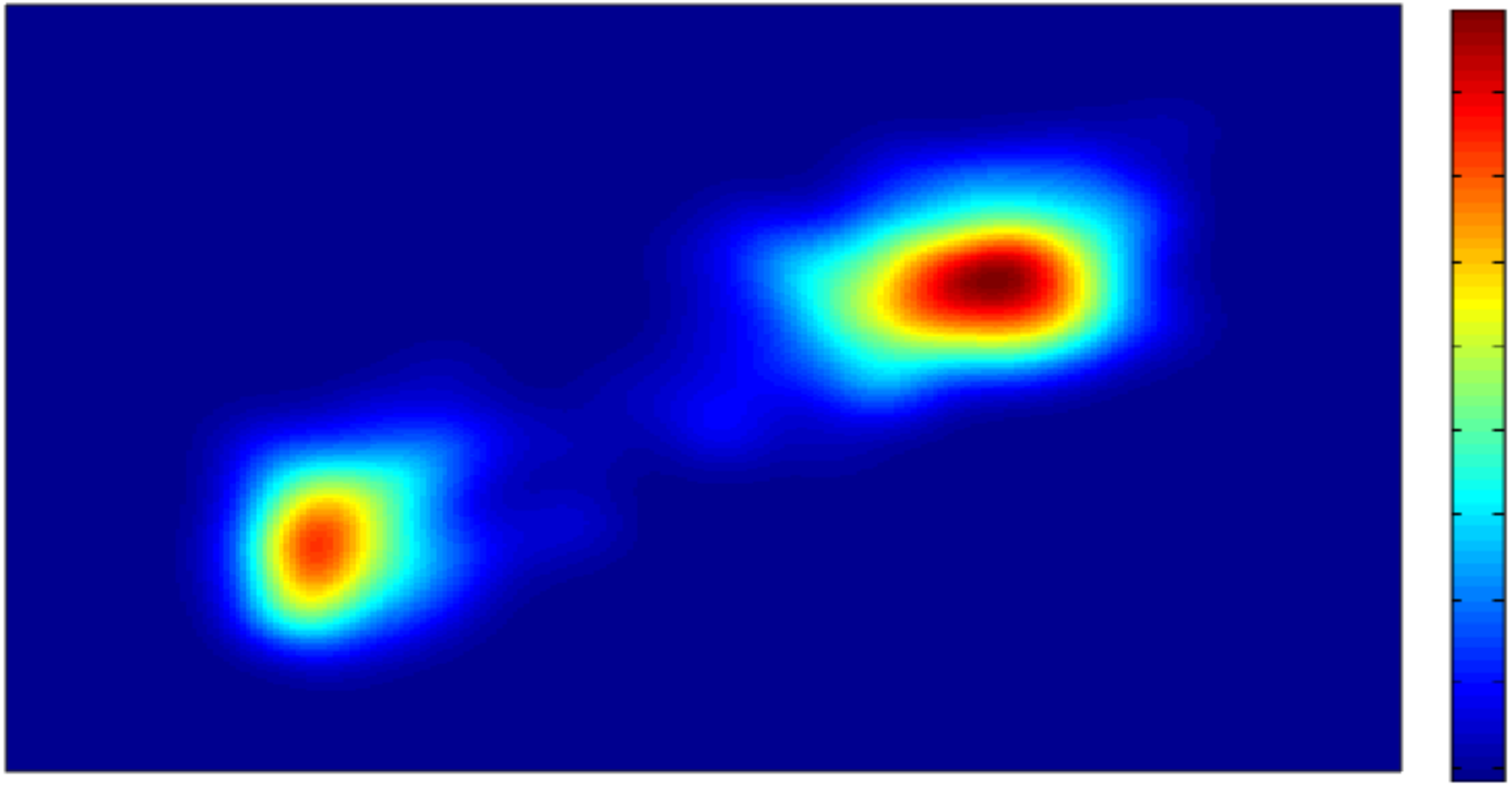
Units of measurement: log-scale transformed molecule counts

Data Density



Highly varied response to stimulus

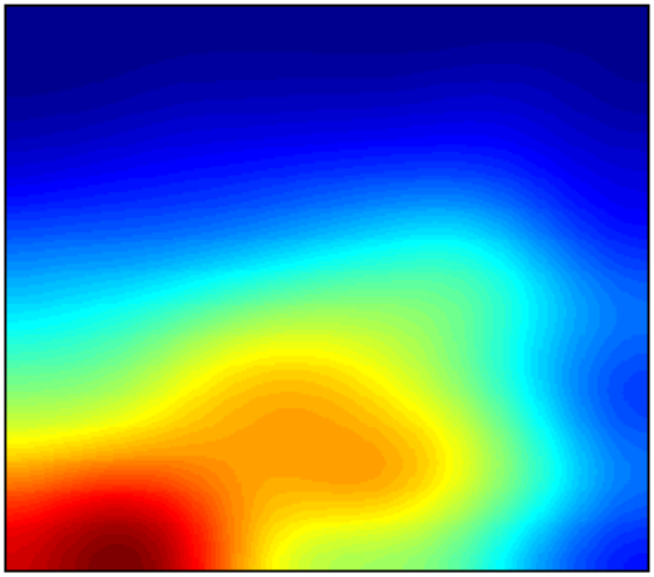
Kernel Density Estimation



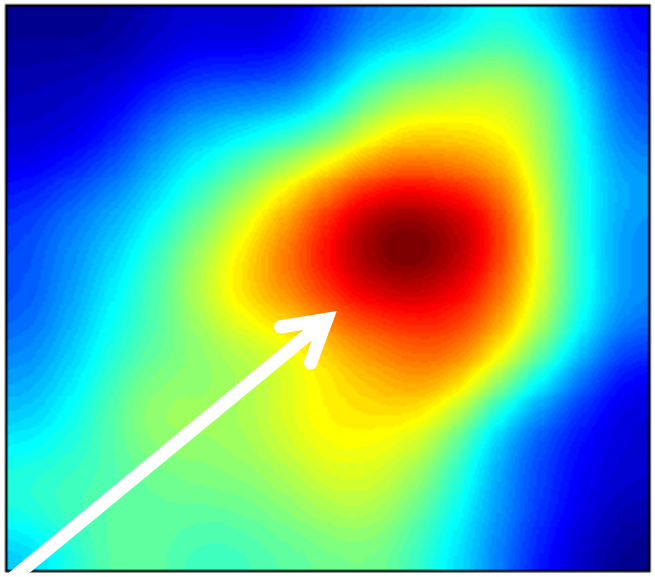
KDE learns underlying probability distribution, smooths data

KDE: Reveals Activation Details

Pre-Stimulation



Post-Stimulation



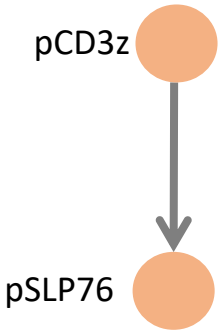
- Molecules increase together
- Shape of influence or relationship unclear

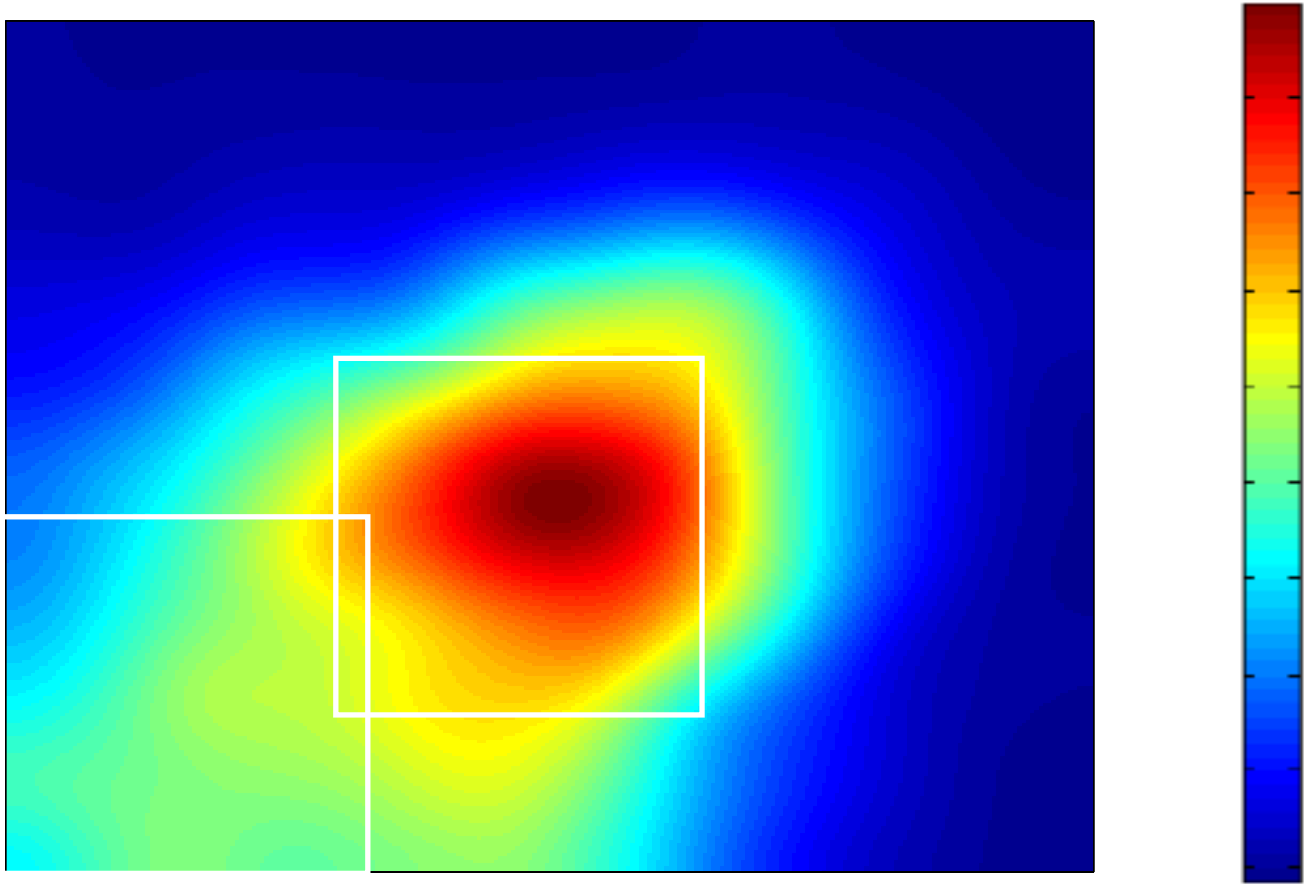
Conditional Density

$$\hat{f}_h(x \mid y) = \hat{f}_h(x, y) / \hat{f}_h(y)$$

- Main idea:
  - Obtain the 2D joint density estimate
  - Normalize the joint density by the *marginal* density

Learning in Sparser Regions

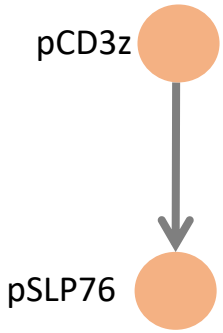


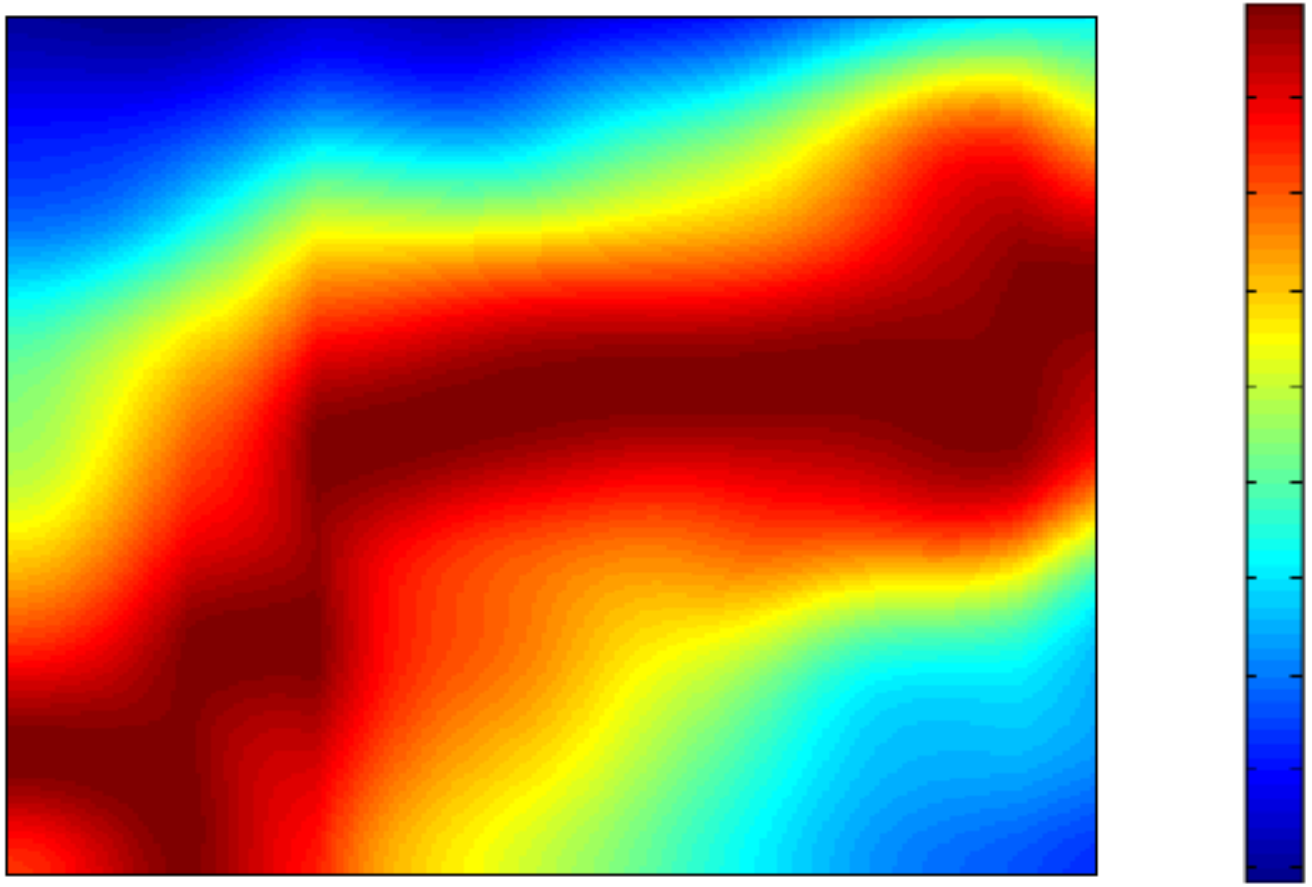


Joint distribution highlights dense areas

Account for sparse regions to learn relationship

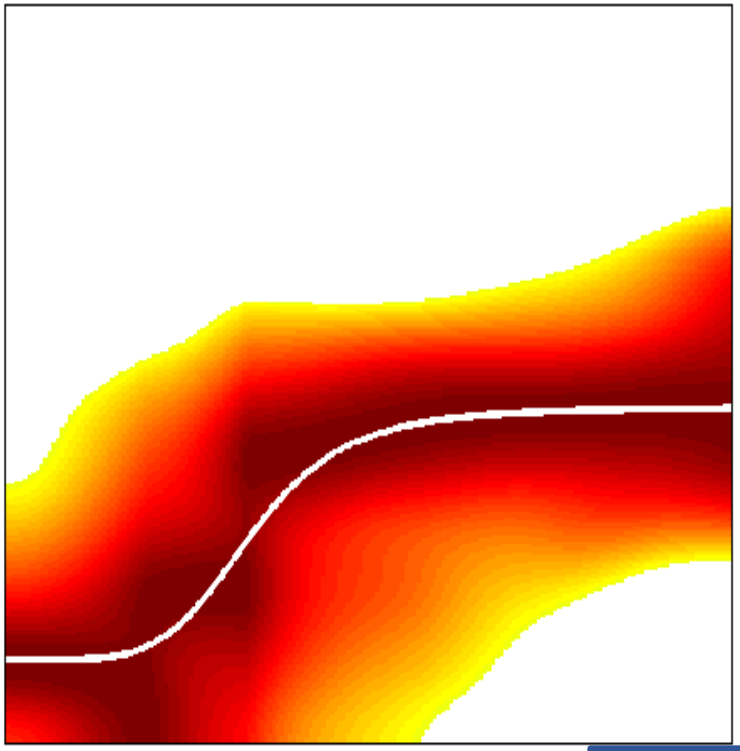
conditional-**D**ensity **Re**scaled **VI**sual





- Captures behavior across full dynamic range
- Captures behavior of small populations of responding cells

Curve-Fitting Dense Region



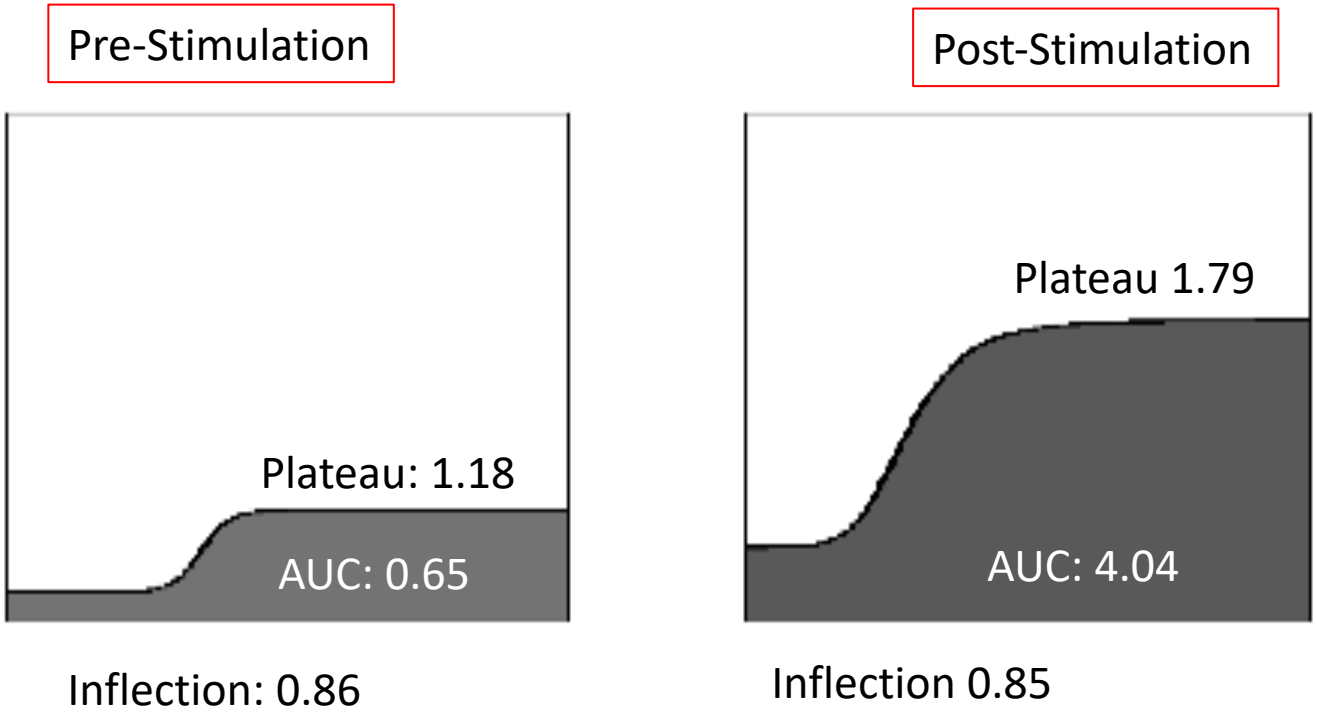
Sigmoid: $$f(x) = \frac{(A - D)}{1 + (\frac{x}{C})^B}$$

A= upper asymptote
B= related to slope of change
C= inflection point

Inflection (C) : 0.86
Plateau (A): 1.79
Area-Under-Curve: 4.04

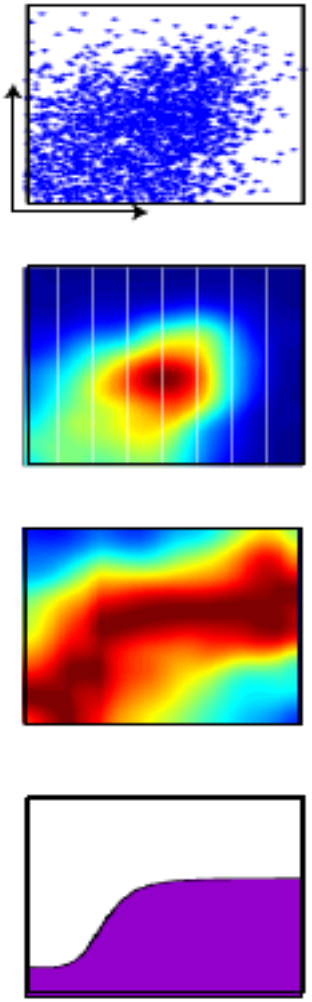
- Can potentially obtain kinetic rate parameters circuit-wide
- Linear, double-sigmoidal models also used

Regression: Parameterizes Response



Quantifies comparison

DREVI (visual)



Raw data

↓

Compute Kernel Density Estimate on a fine grid

↓

Column-normalize grid to obtain Conditional Density

↓

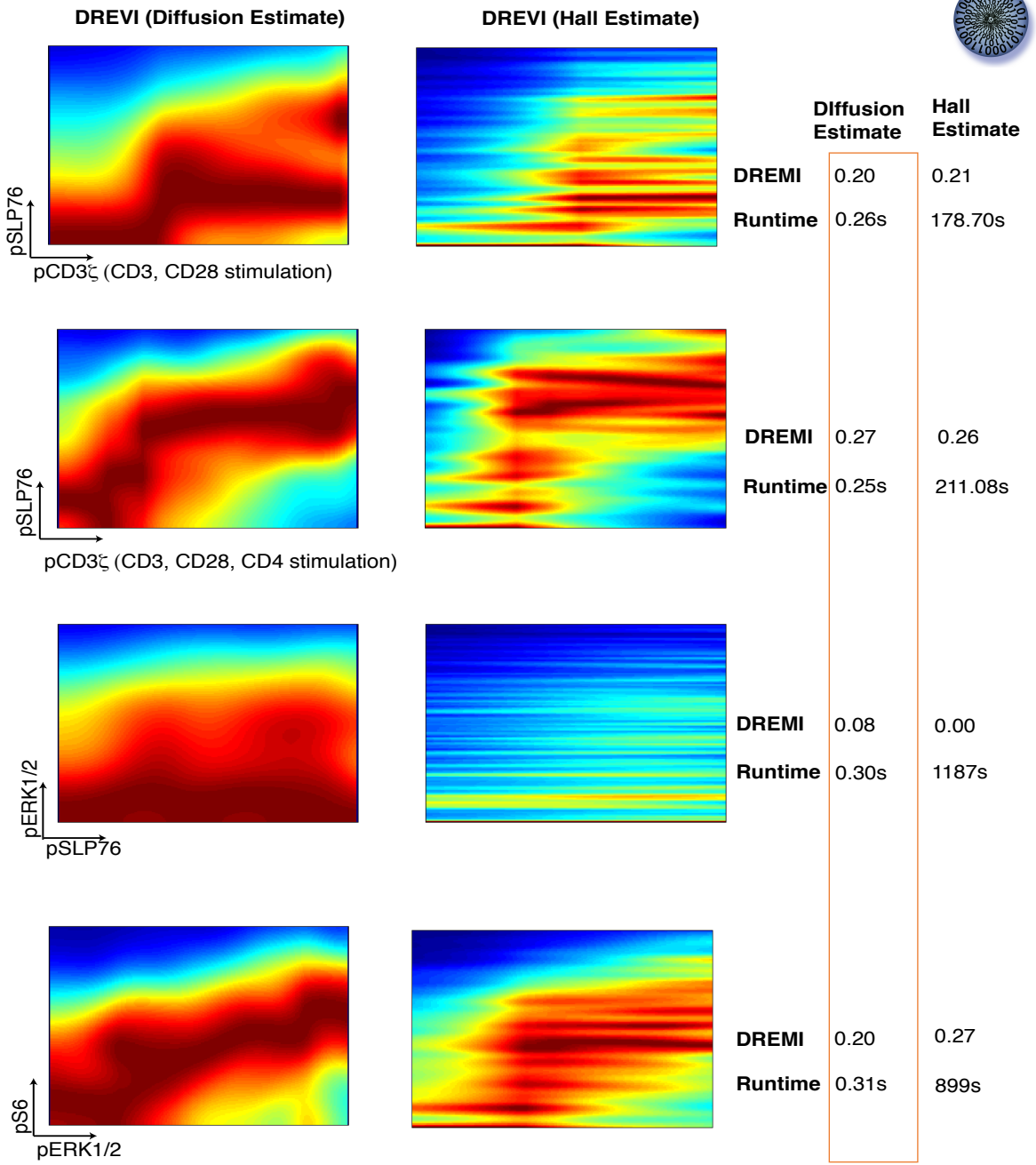
Rescale each column and render heatmap

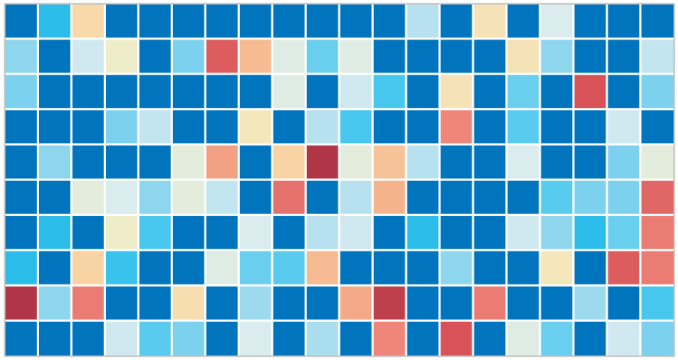
⋮

Compute best-fit curve to high density region

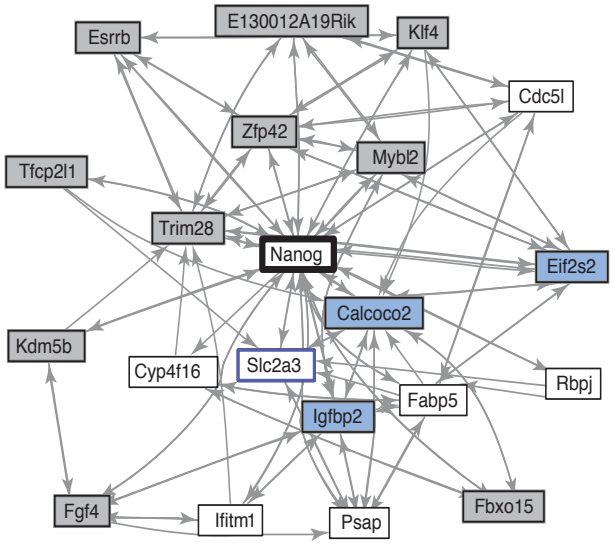
Diffusion KDE

Diffusion-based KDE estimate is faster and smoother



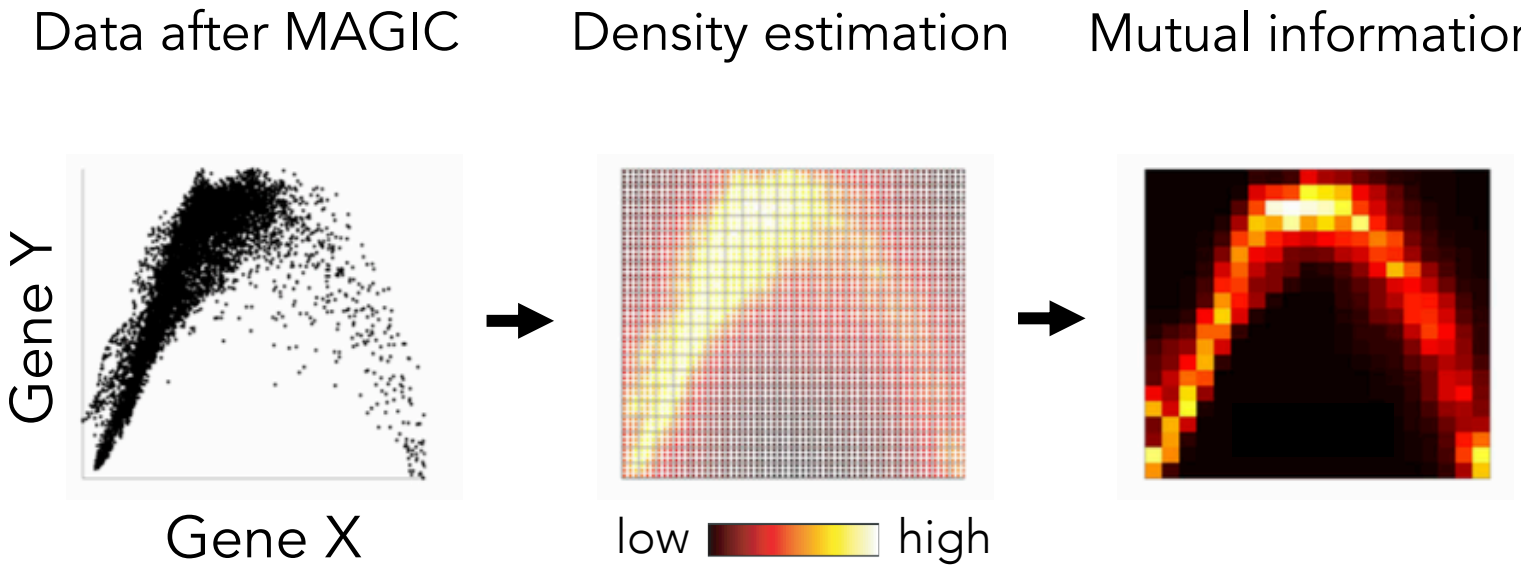


scRNA-seq data



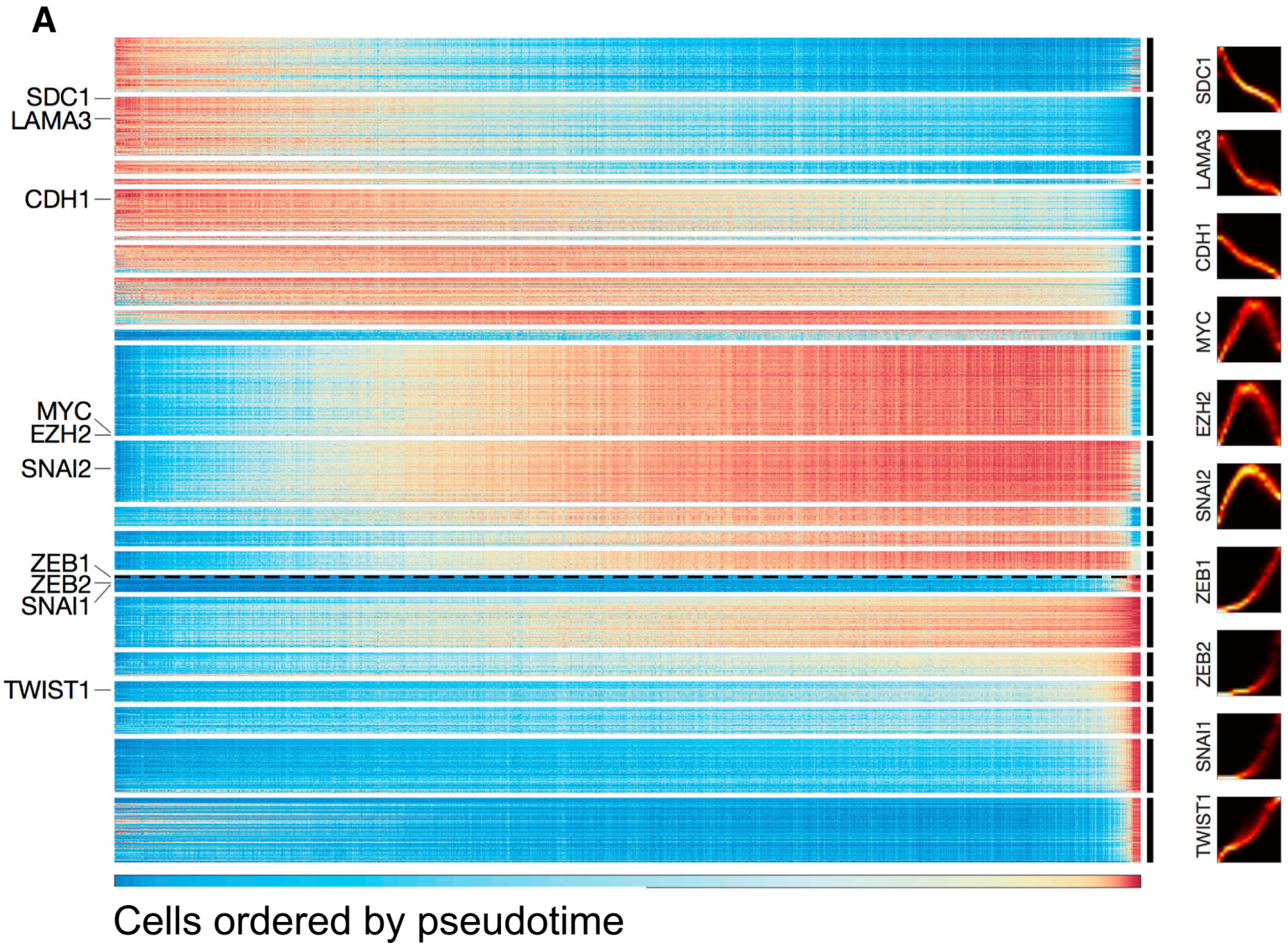
regulatory network

Using “snapshot” scRNA-seq data to predict transcription factor-target interactions involved in the epithelial—mesenchymal transition.



quantifying gene-gene relationships

Ranking genes by association with pseudotime identifies drivers of differentiation



Conclusions

1. Mutual information (MI) can be used to identify non-linear relationships between variables

2. Density normalization allows for quantifying MI in regions of data sparsity

3. MI can quantify gene-gene relationships to infer regulatory relationships

4. MI of genes with pseudotime identifies drivers of differentiation