The Krishnaswamy Laboratory

Yale Genetics and Yale SEAS present

Machine Learning

for

Single Cell Analysis

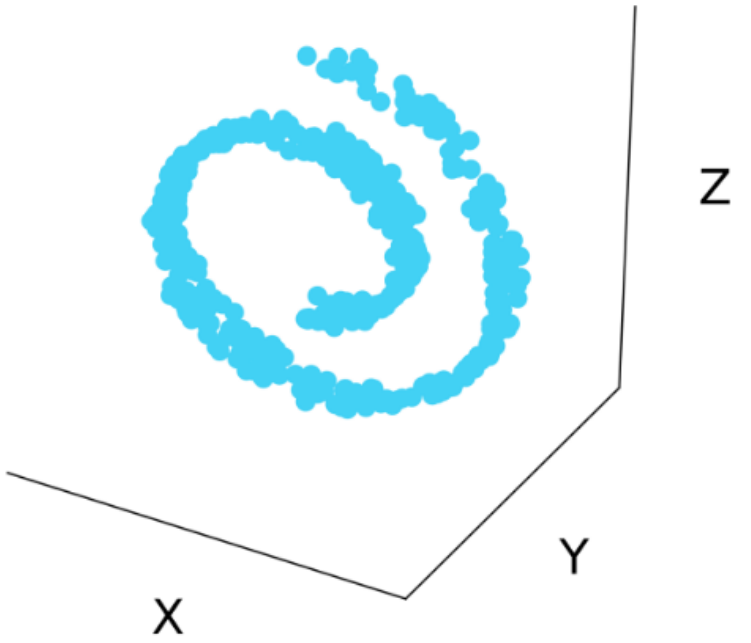
Online - May 20-29, 2020

What is your favorite sport to follow?

Day 3: Denoising, Batch Correction, and Clustering

Using the manifold model to denoise data

What is the intrinsic or latent dimensionality of this data?



0

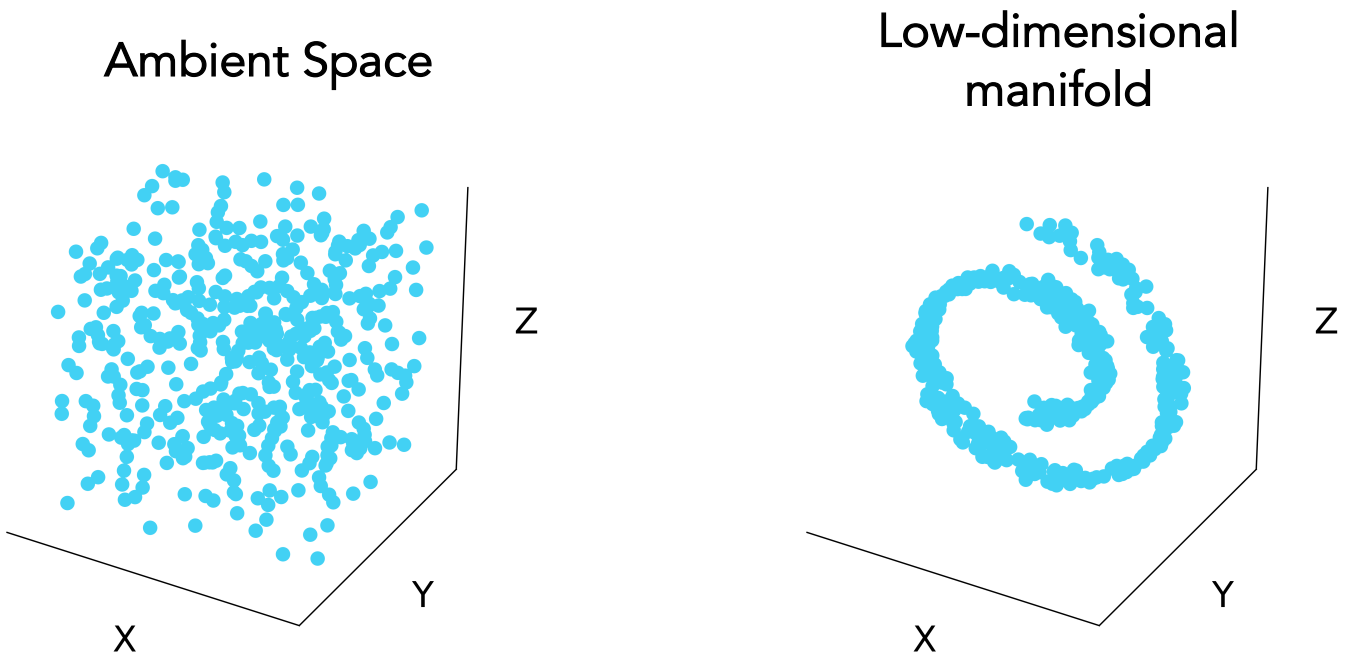
1

2

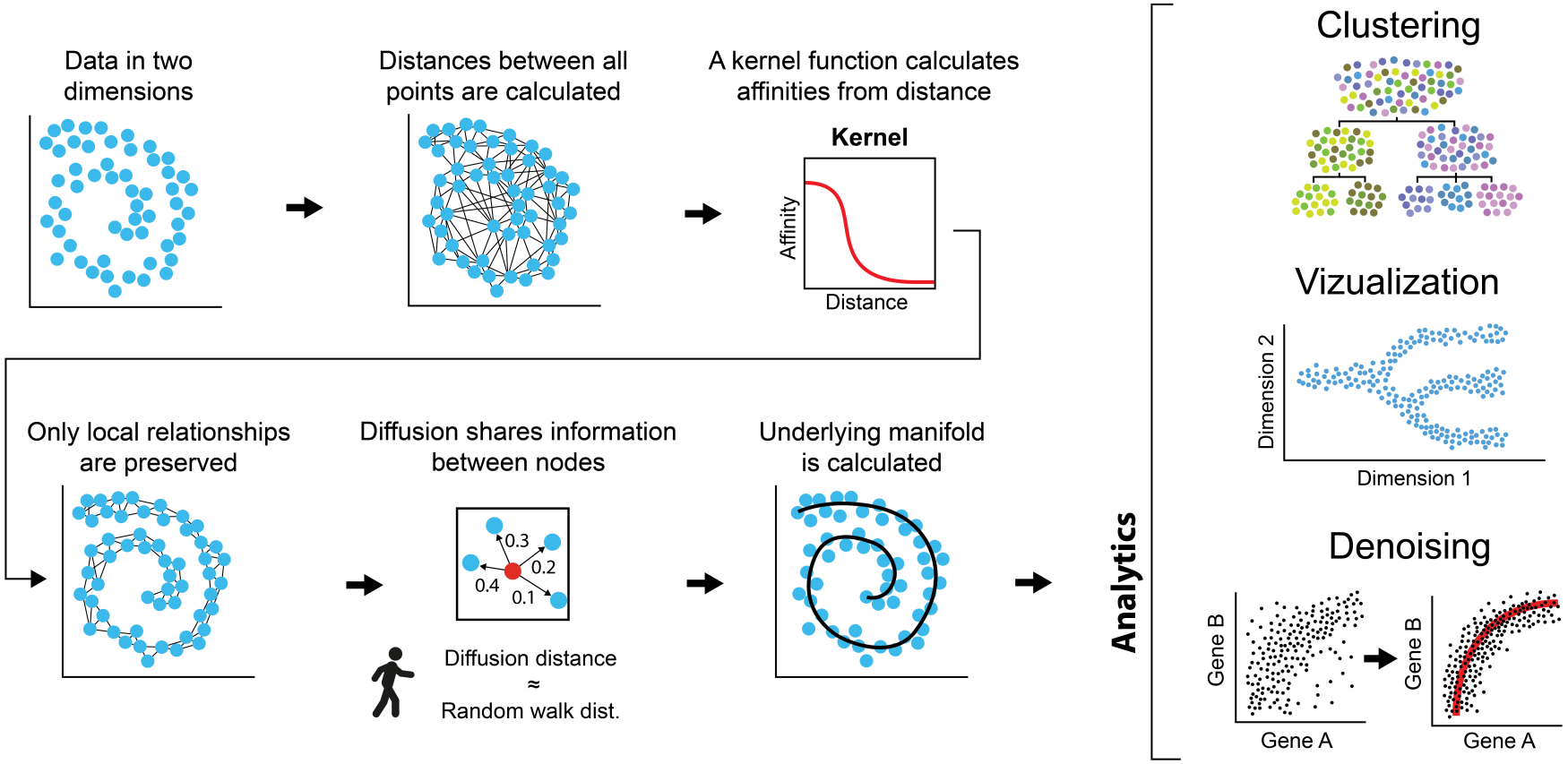
3

4

Latent structure in high dimensional data

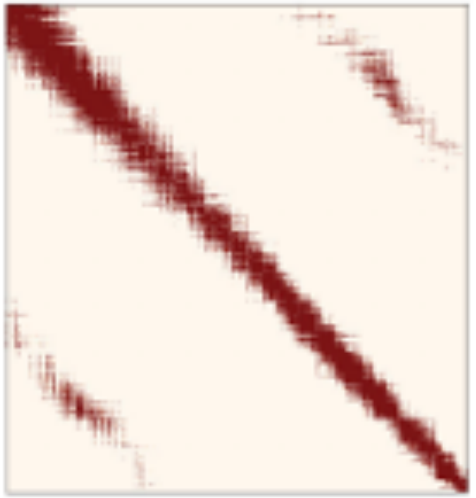


How do we learn the shape of data?



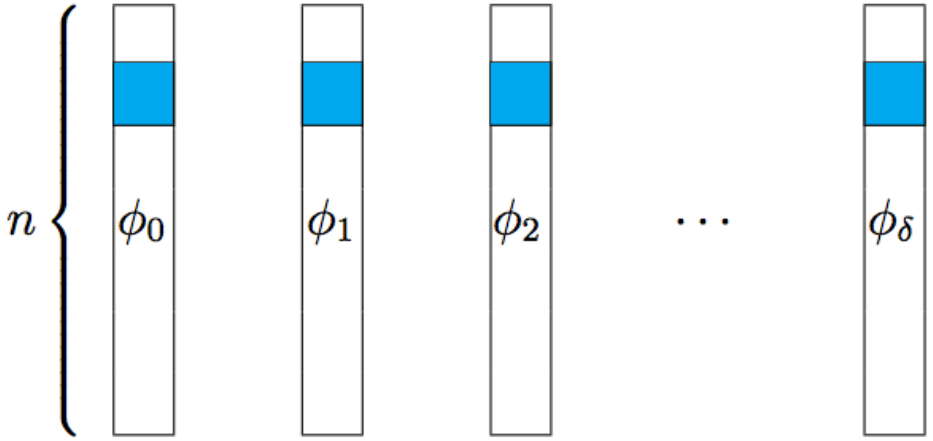
Affinity Matrix contains Manifold Information

- Regardless of Markov normalization
- Graph Adjacency Matrix contains the information about the manifold



Revealed by Eigendecomposition

$$1 = \boxed{\lambda_0} \geq \boxed{\lambda_1} \geq \boxed{\lambda_2} \geq \cdots \geq \boxed{\lambda_\delta} > 0$$

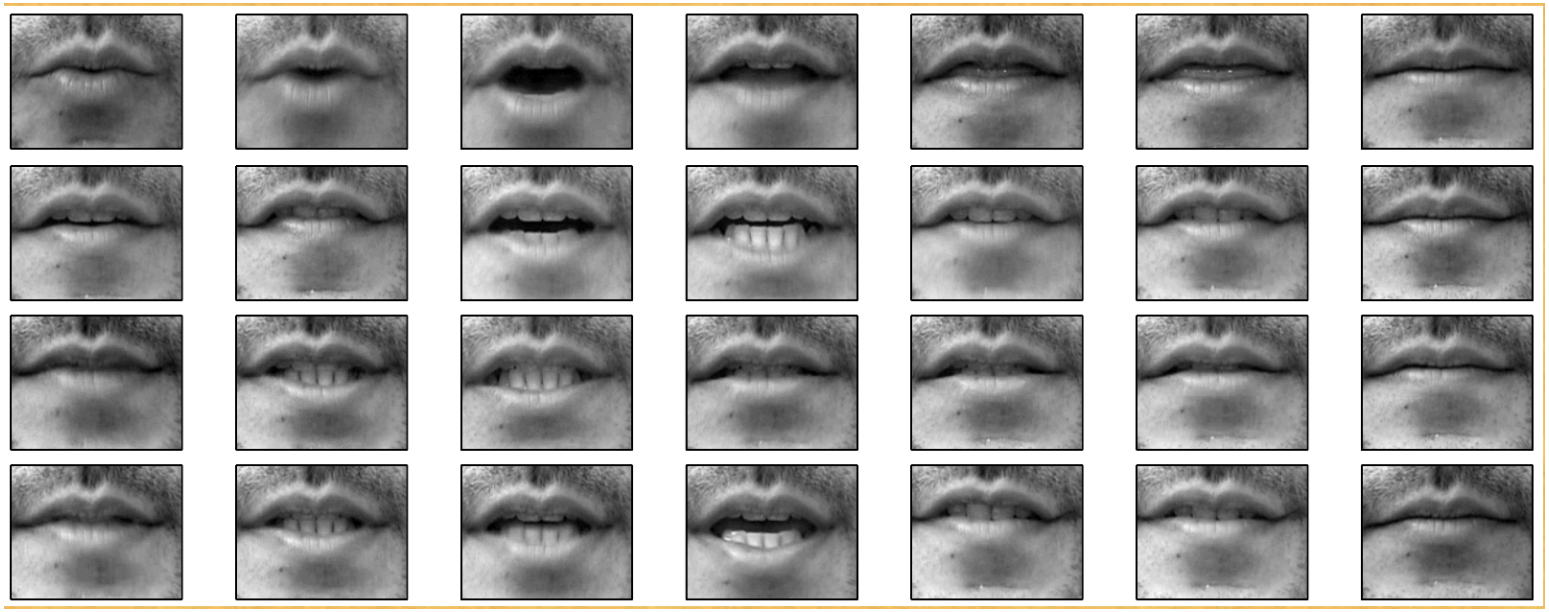


$$x \mapsto \Phi(x) \triangleq [\lambda_0\phi_0(x)\,,\ \lambda_1\phi_1(x)\,,\ \lambda_2\phi_2(x)\,,\ \ldots\,,\ \lambda_\delta\phi_\delta(x)]^T$$

[Coifman, Lafon, ACHA 2006]

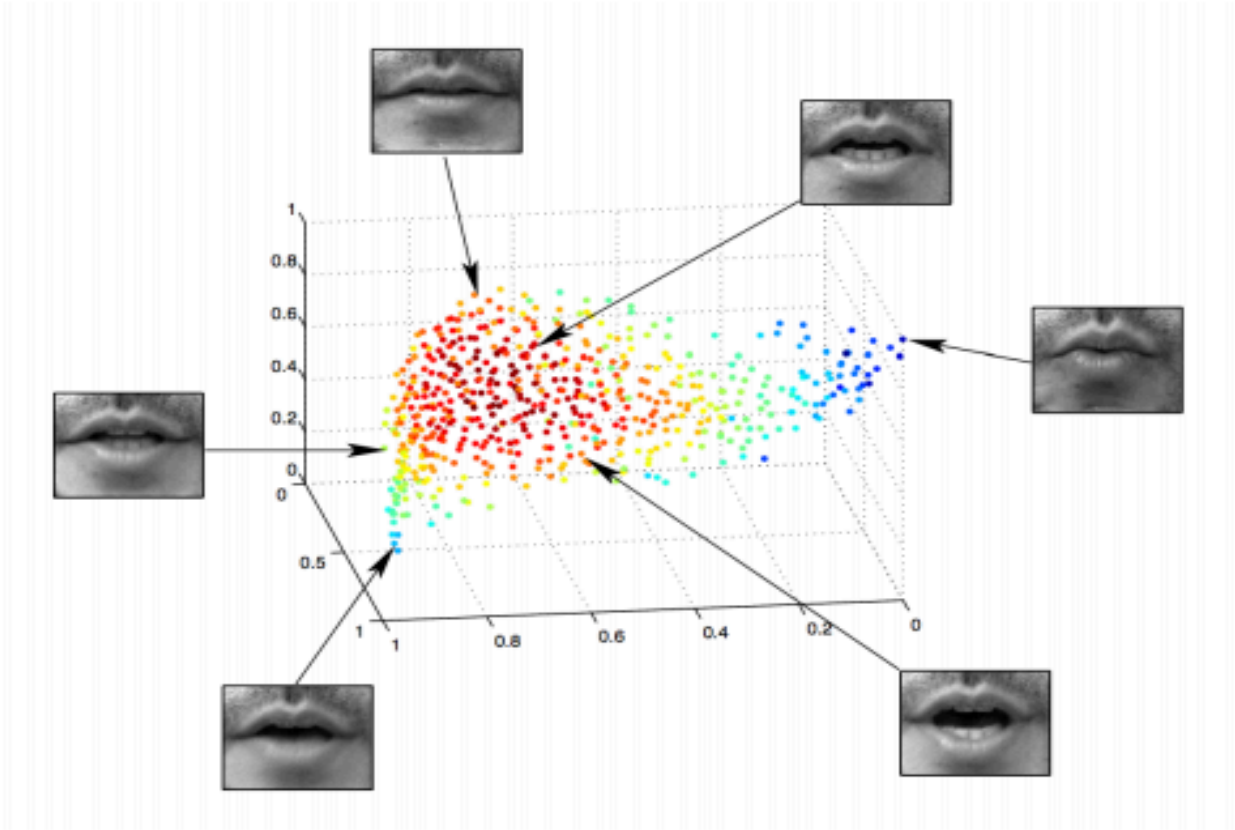
Lips Dataset: Spoken Digits

ONE



[3]S. Lafon, Y. Keller, and R. R. Coifman. Data Fusion and Multi-Cue Data Matching by Diffusion Maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 28, pages 1784-1797, 2006.

Lips Manifold Diffusion Component

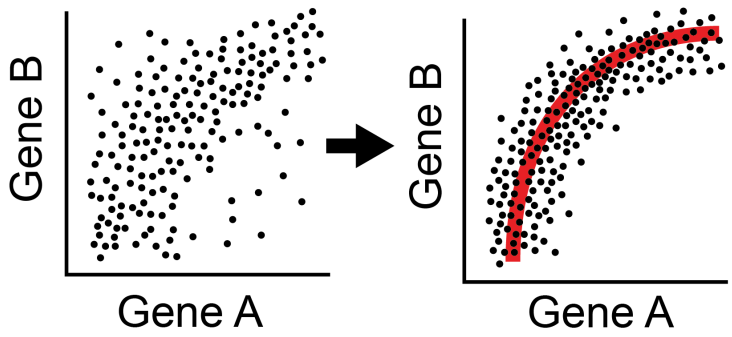


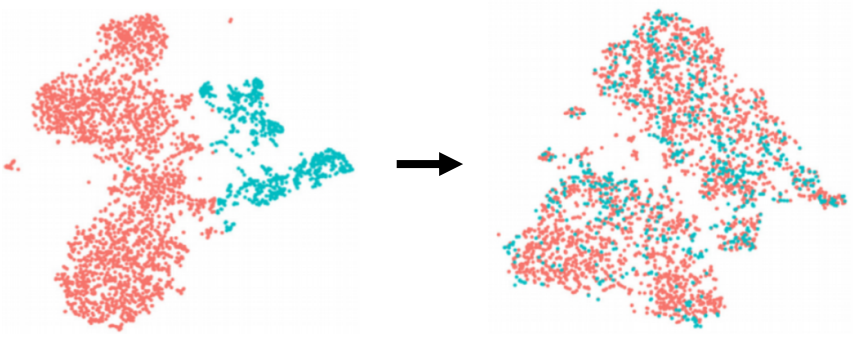
Non-Linear Dimensionality Reduction

- Almost always uses the graph affinity matrix
- Sometimes involves eigendecomposition Laplacian eigenmaps
  - Diffusion maps
- Sometimes involves preserving information from graph in another way
- tSNE: Preserves nearest neighbor probabilites from Markov Affinity matrix in 2D using a randomized algorithm
- PHATE: Preserves probability-distribution distances from diffused Markov affinity matrix in 2D using MDS

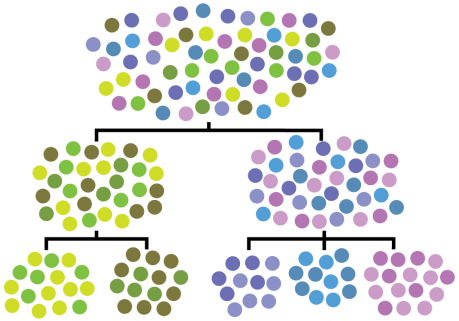
Other uses of affinity matrices, eigenvectors

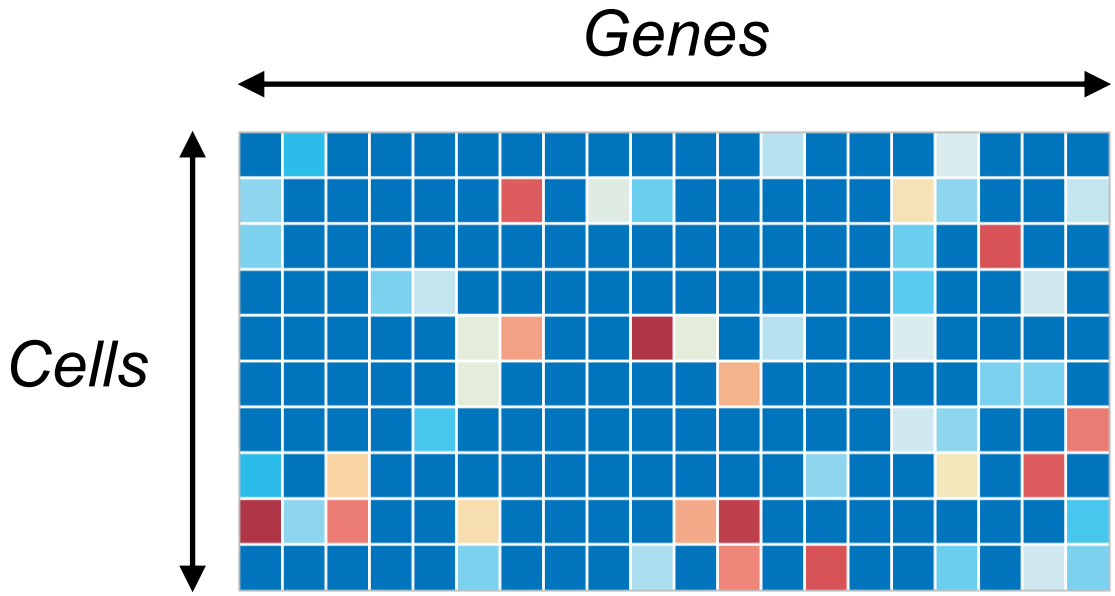
Data denoising and batch normalization



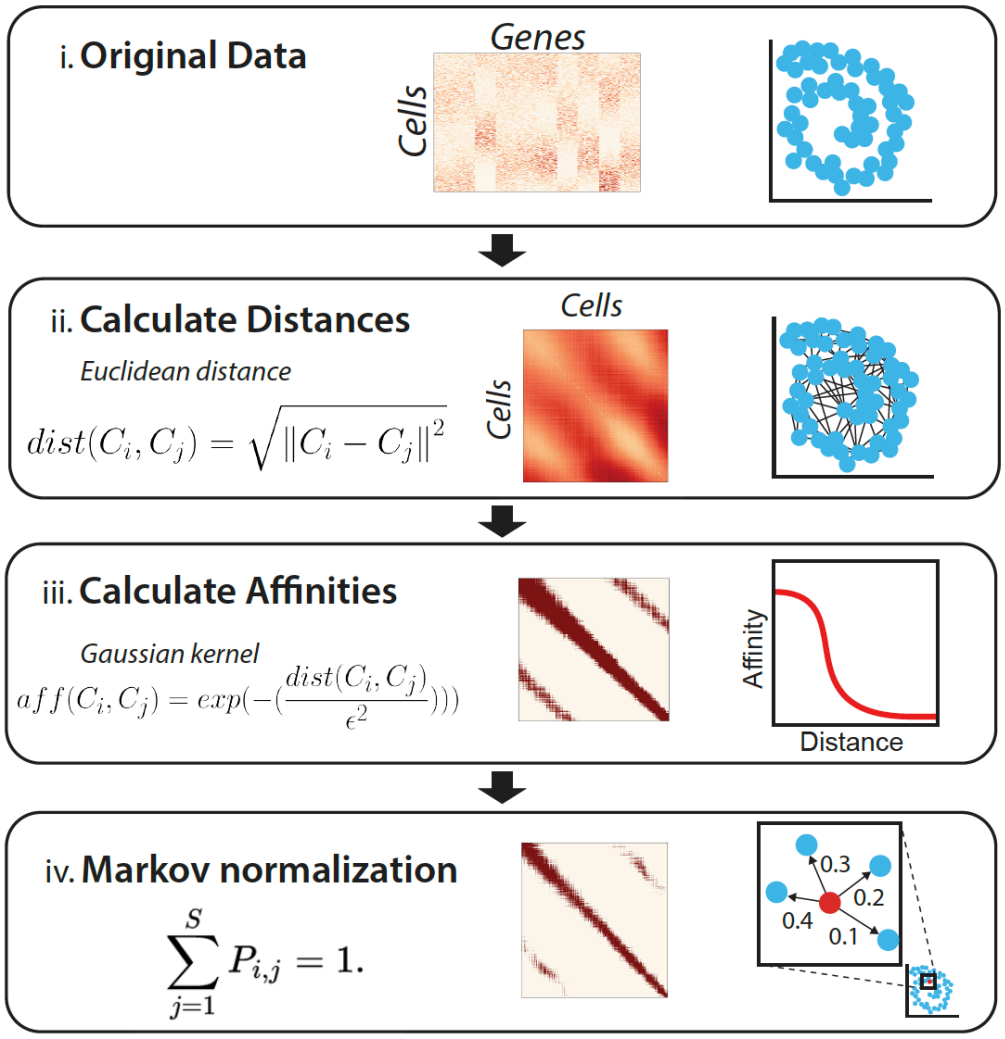


Clustering



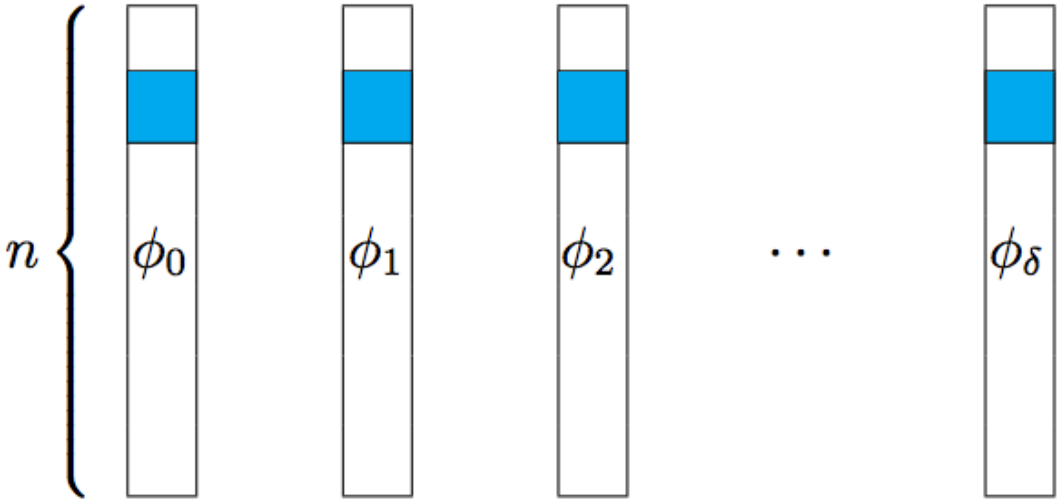


Data is noisy and sparse (scRNA-seq)


i. Original Data
Genes
Cells
ii. Calculate Distances
Euclidean distance
$dist(C_i, C_j) = \sqrt{\|C_i - C_j\|^2}$
Cells
Cells
iii. Calculate Affinities
Gaussian kernel
$aff(C_i, C_j) = exp(-(\frac{dist(C_i, C_j)}{\epsilon^2}))$
Affinity
Distance
iv. Markov normalization
$\sum_{j=1}^{S} P_{i,j} = 1.$
0.3
0.2
0.4
0.1

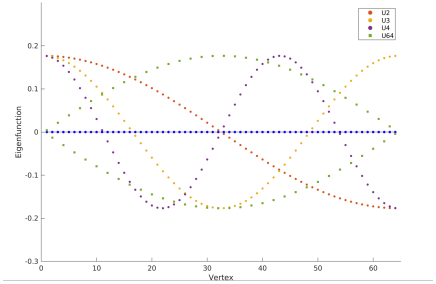
Eigenvectors of Affinity Matrix

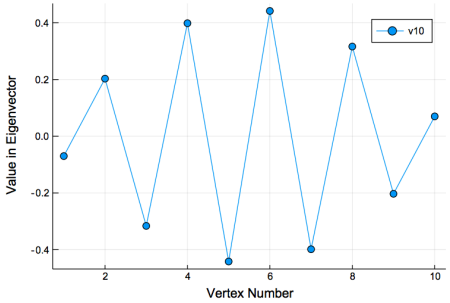
$$1 = \boxed{\lambda_0} \geq \boxed{\lambda_1} \geq \boxed{\lambda_2} \geq \cdots \geq \boxed{\lambda_\delta} > 0$$

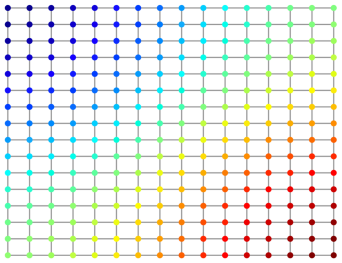


$$x \mapsto \Phi(x) \triangleq [\lambda_0\phi_0(x)\,,\ \lambda_1\phi_1(x)\,,\ \lambda_2\phi_2(x)\,,\ \ldots\,,\ \lambda_\delta\phi_\delta(x)]^T$$

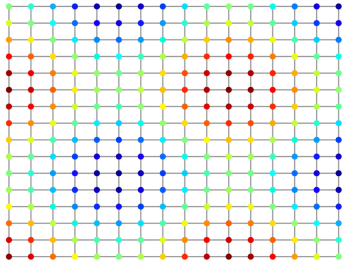
Eigenvectors are frequency harmonics



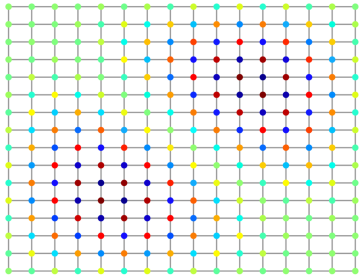




2nd Eigenvector

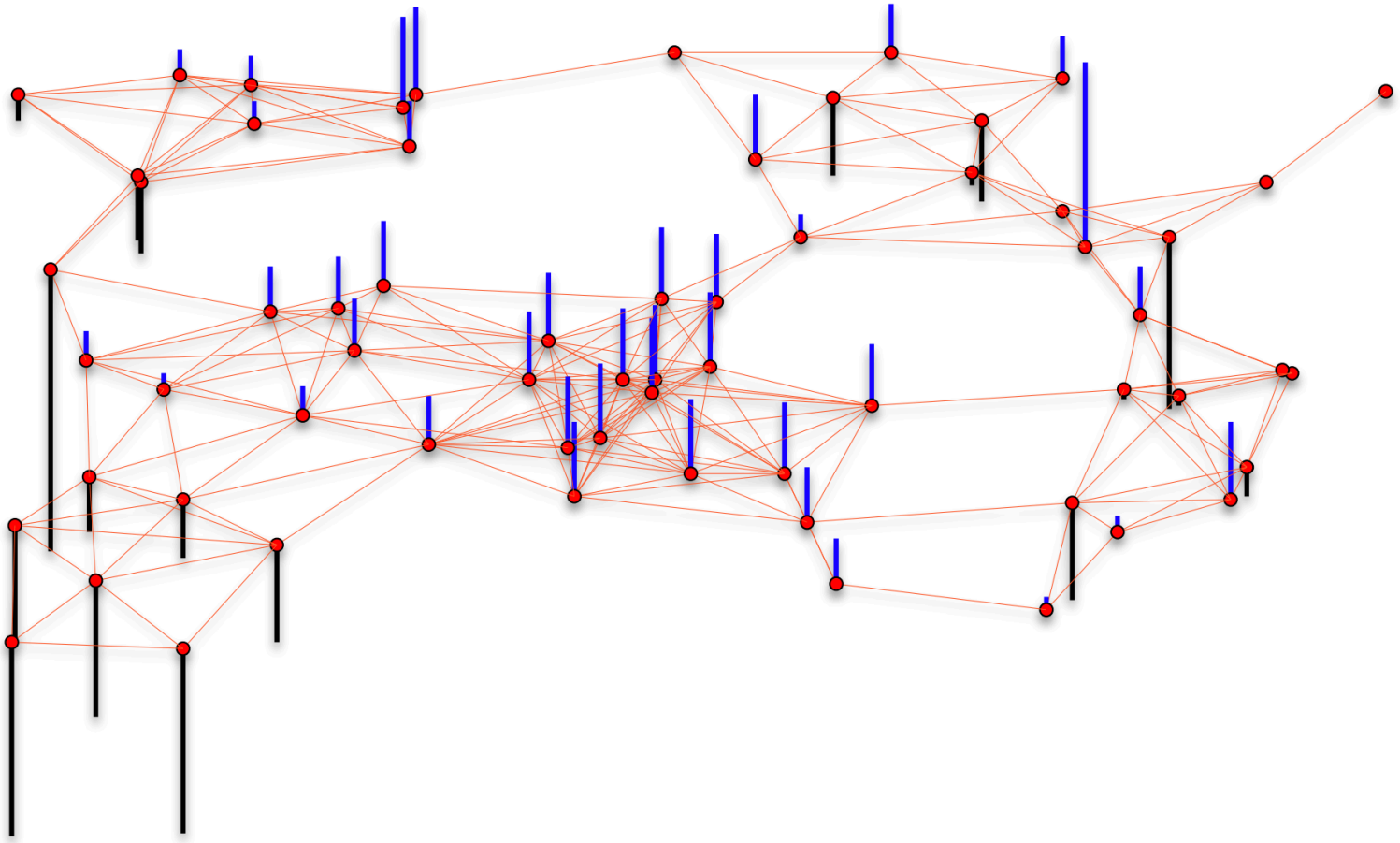


10th Eigenvector

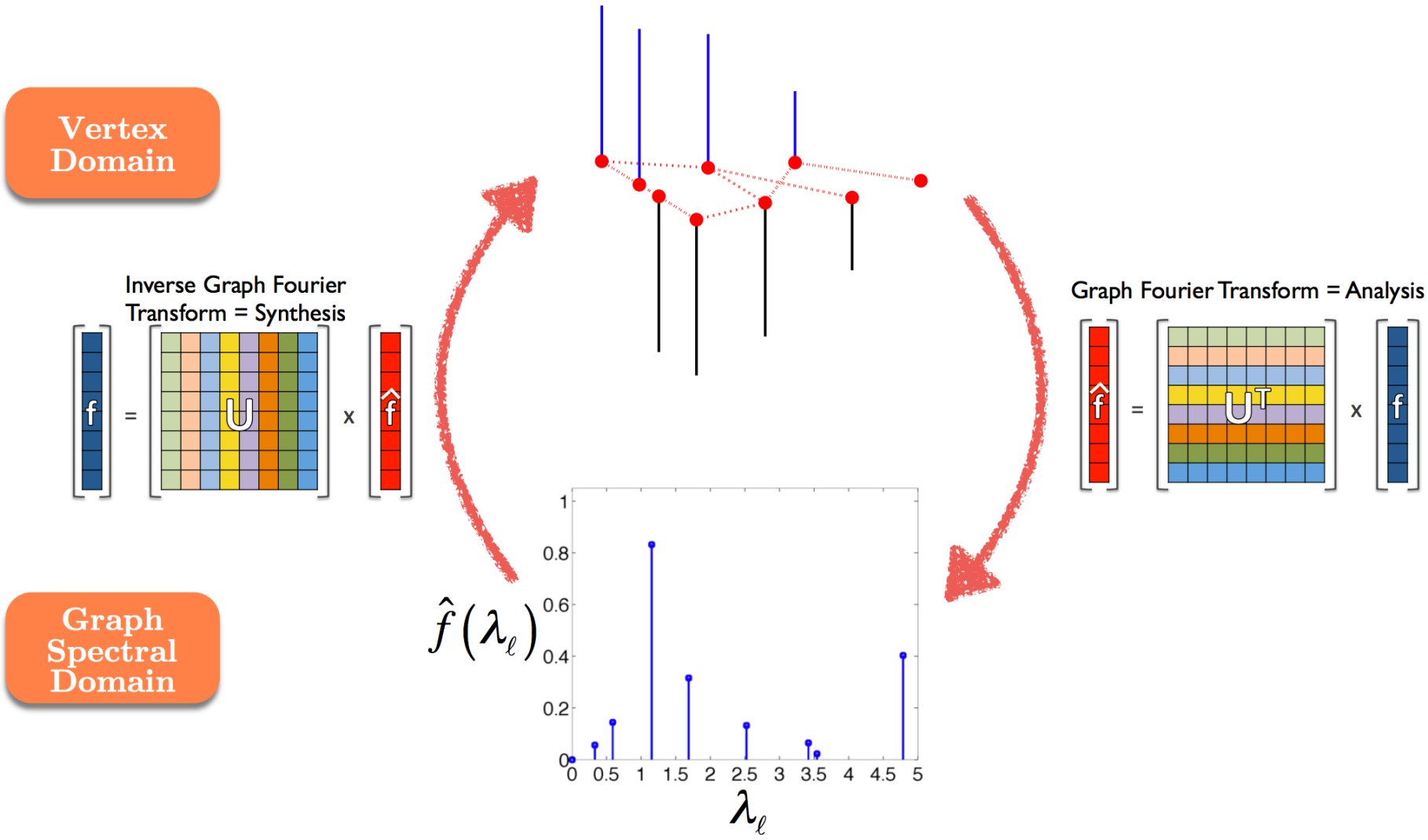


2nd to last eigenvector

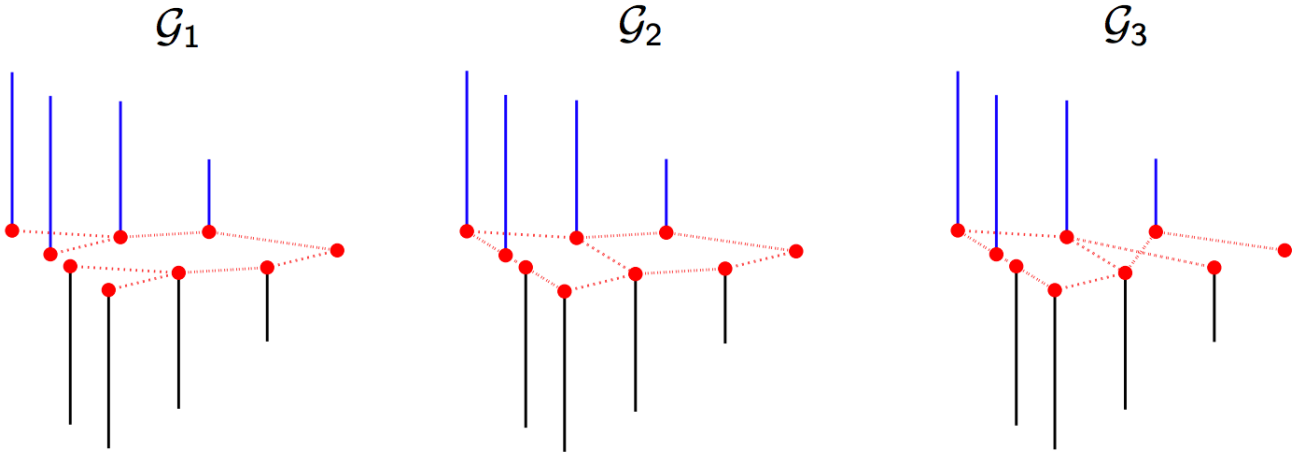
Cells are nodes, mRNA gene-counts are signals on a graph



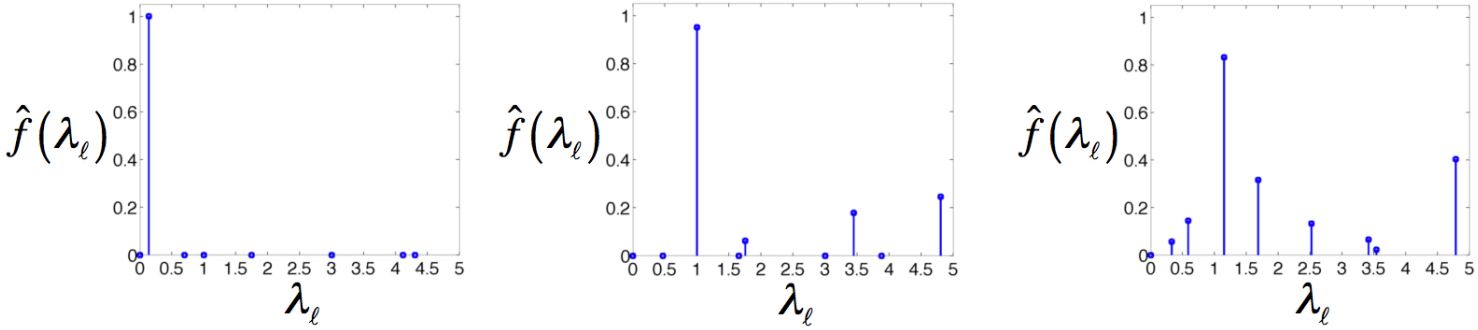
Graph Fourier Transform



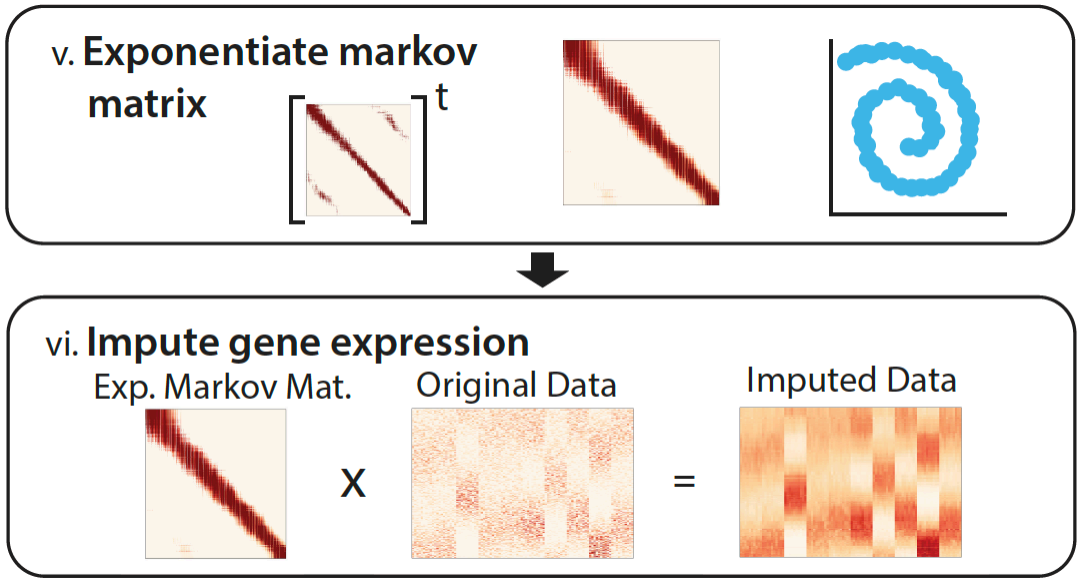
Vertex Domain

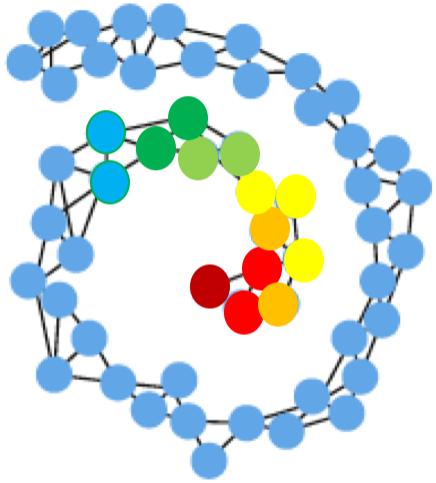


Graph Spectral Domain



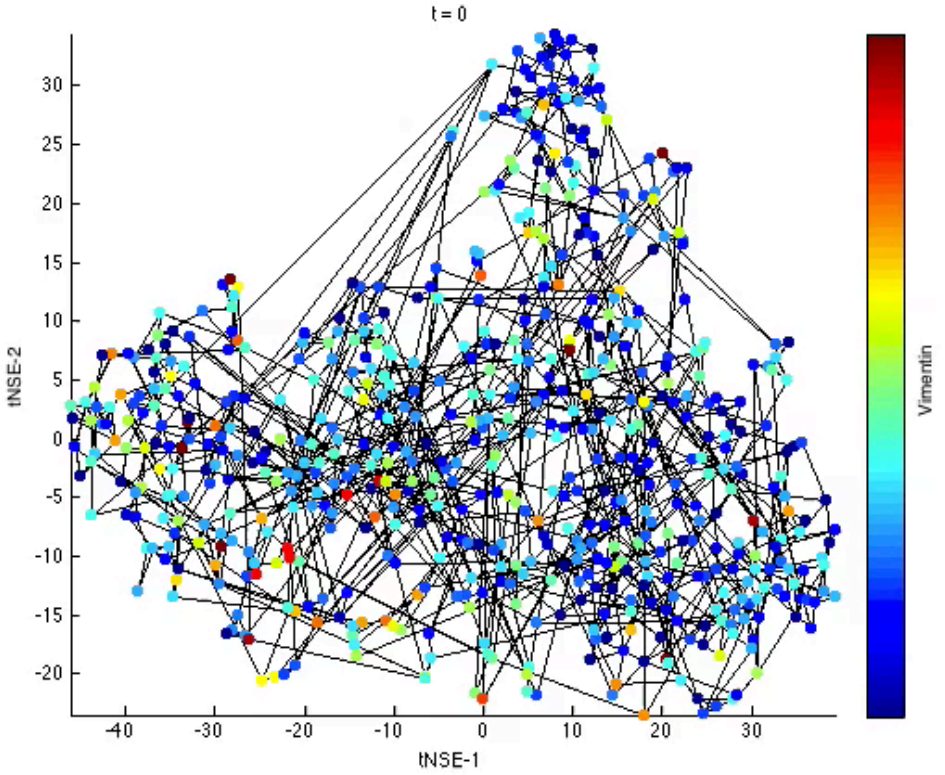
Imputation Step = Smoothing on Graph





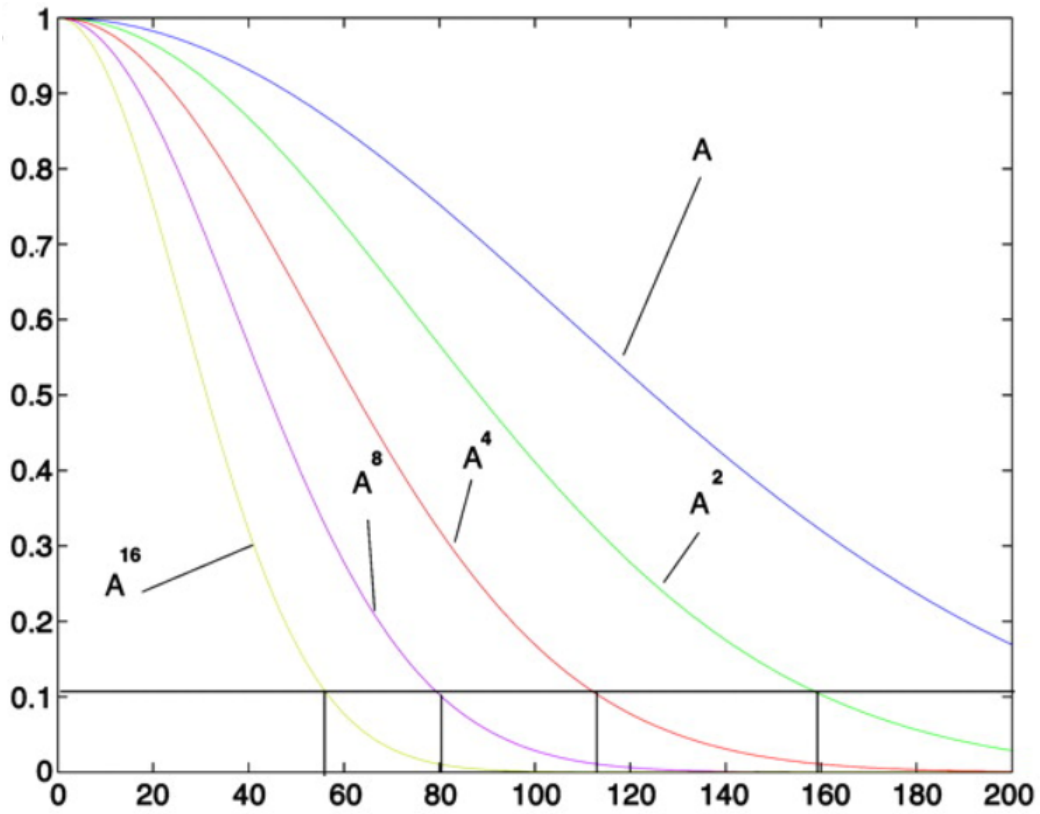
Vertex Domain

- Smooths signal on graph
- Takes weighted average of neighbor

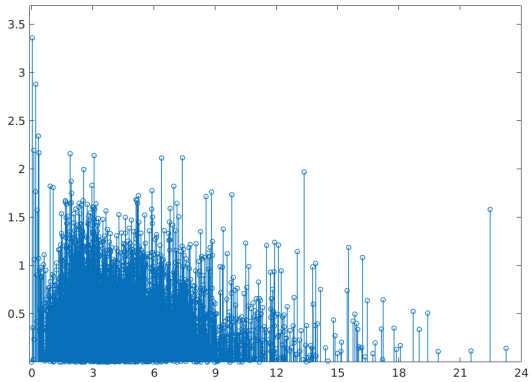


Low pass filter of Eigenvectors

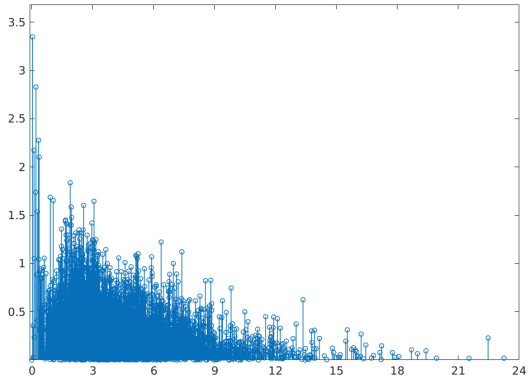
$$\boldsymbol{\Phi}_t(\boldsymbol{x}_i) : \boldsymbol{x}_i \longmapsto \left[\lambda_1^t \phi_1(i), \lambda_2^t \phi_2(i), \lambda_3^t \phi_3(i), ..., \lambda_{M-1}^t \phi_{M-1}(i)\right] \in \mathbb{R}^{M-1}$$



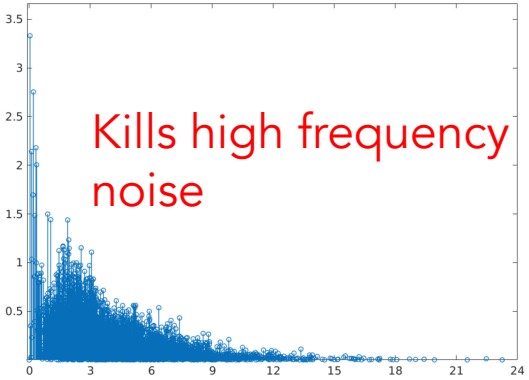
Frequency domain



No Smoothing

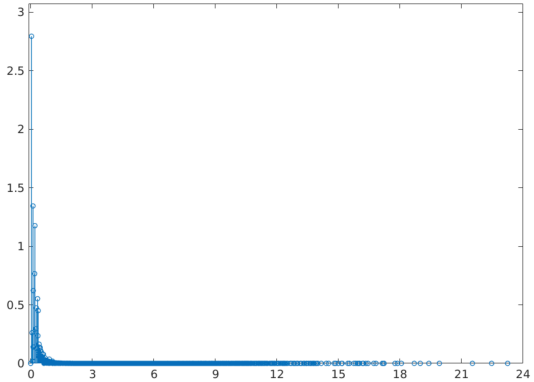


T=2

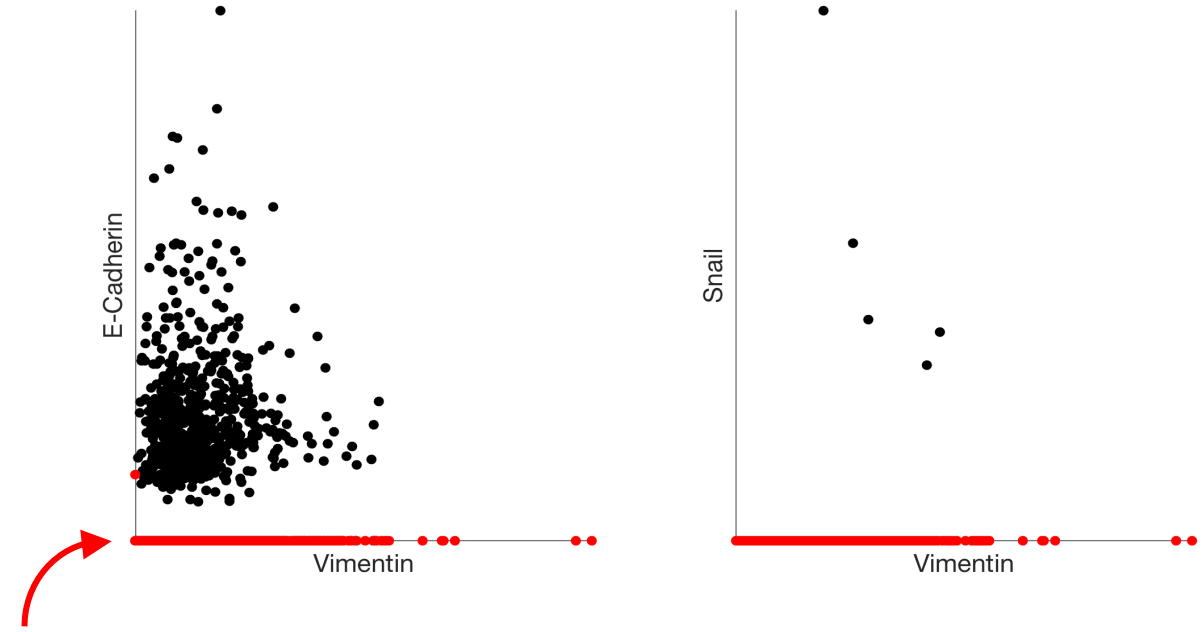


Kills high frequency noise

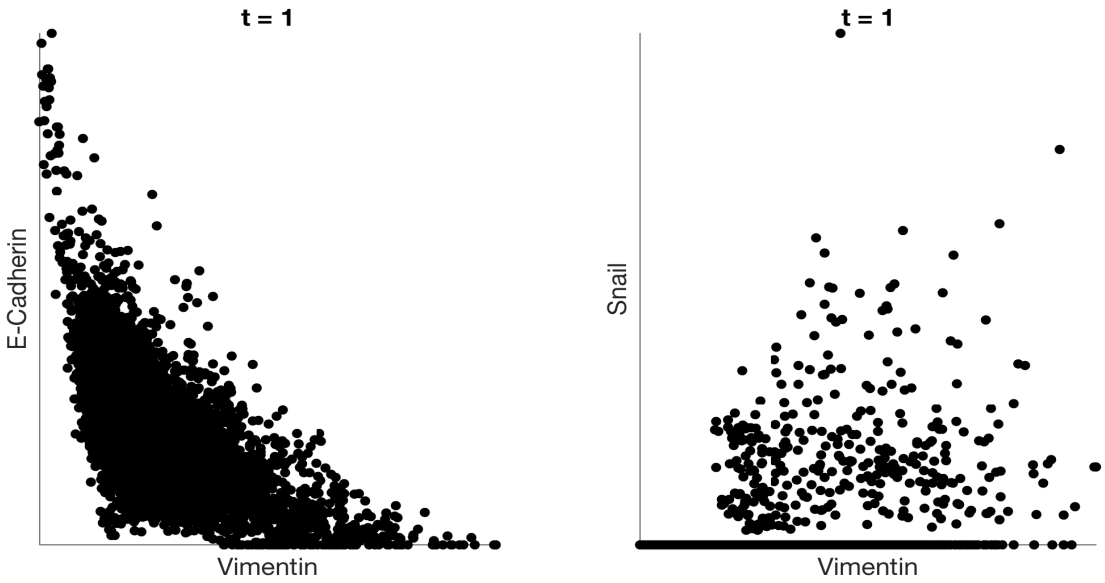
T=5



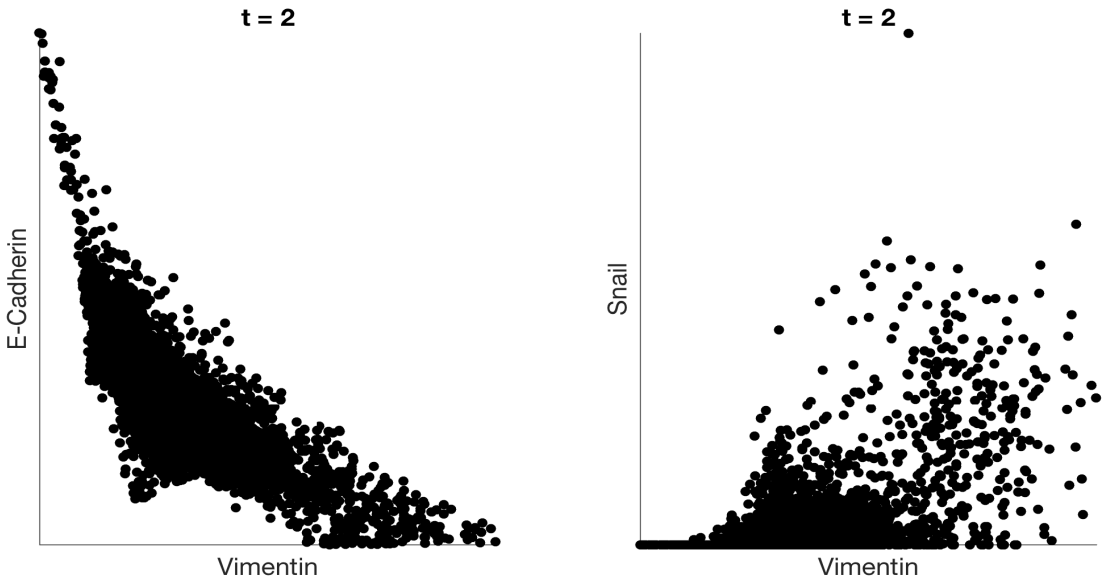
T=100



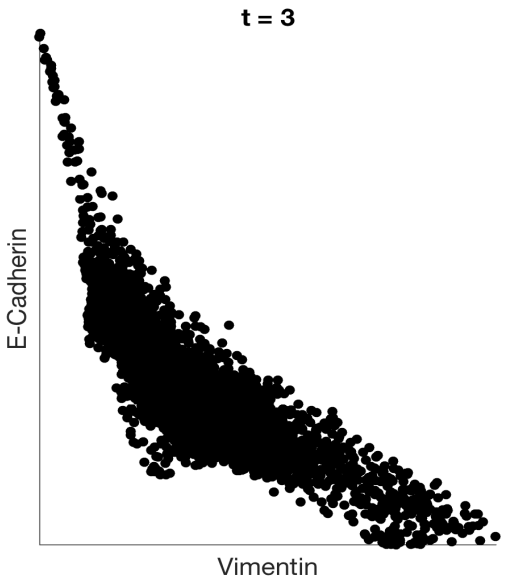
original data with dropout

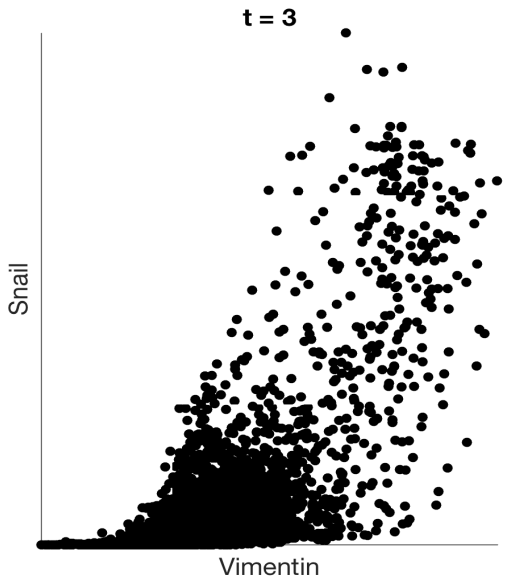


imputation with MAGIC

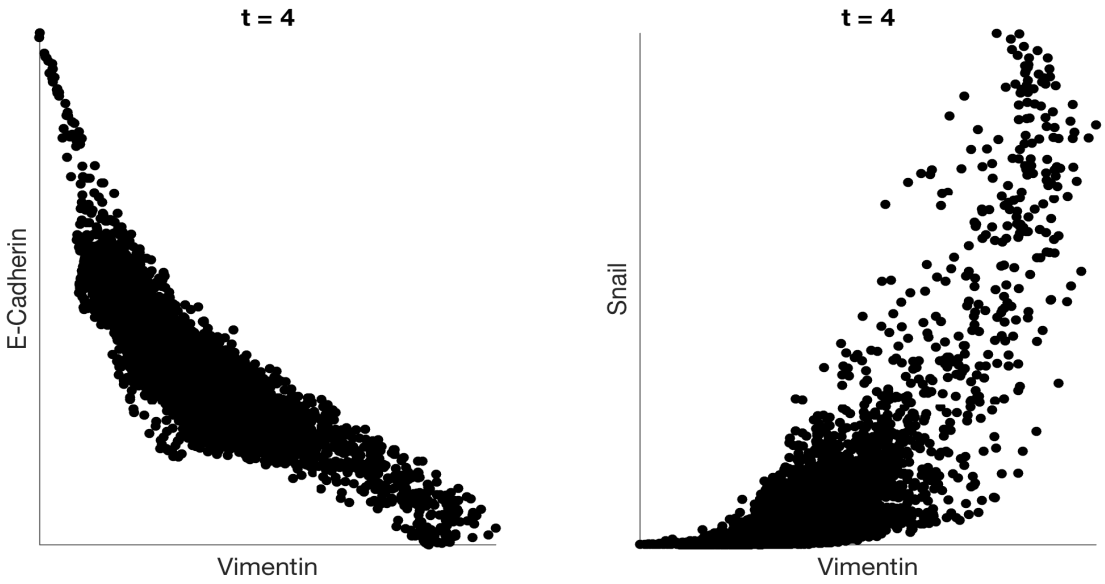


imputation with MAGIC

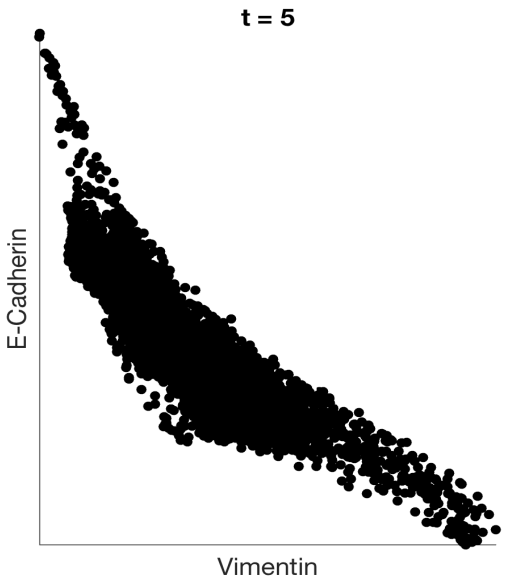


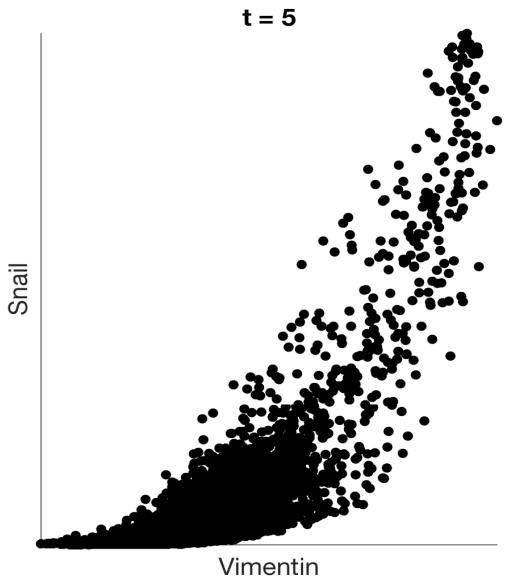


imputation with MAGIC

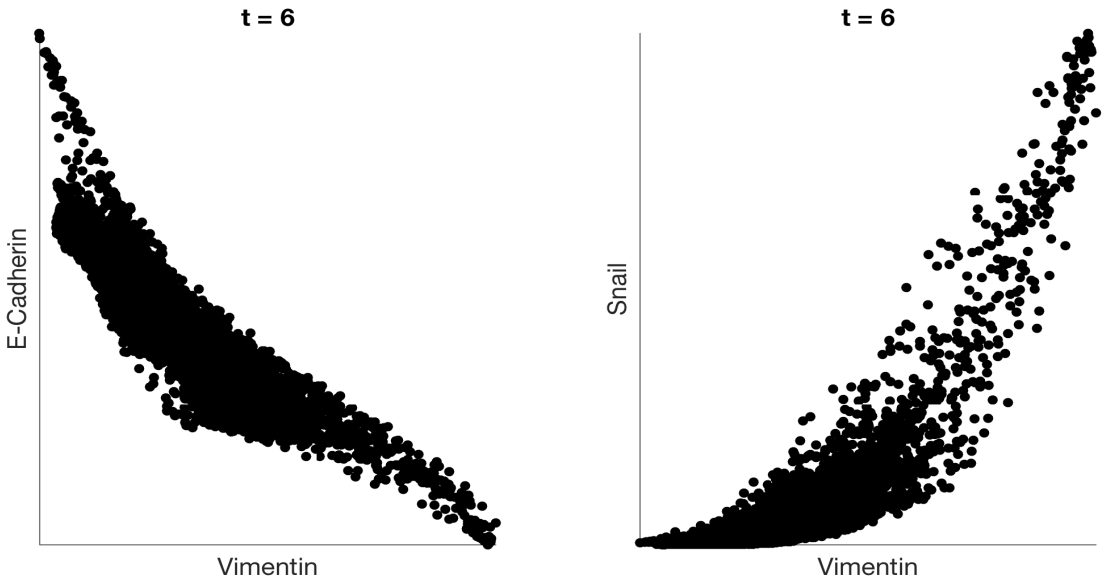


imputation with MAGIC



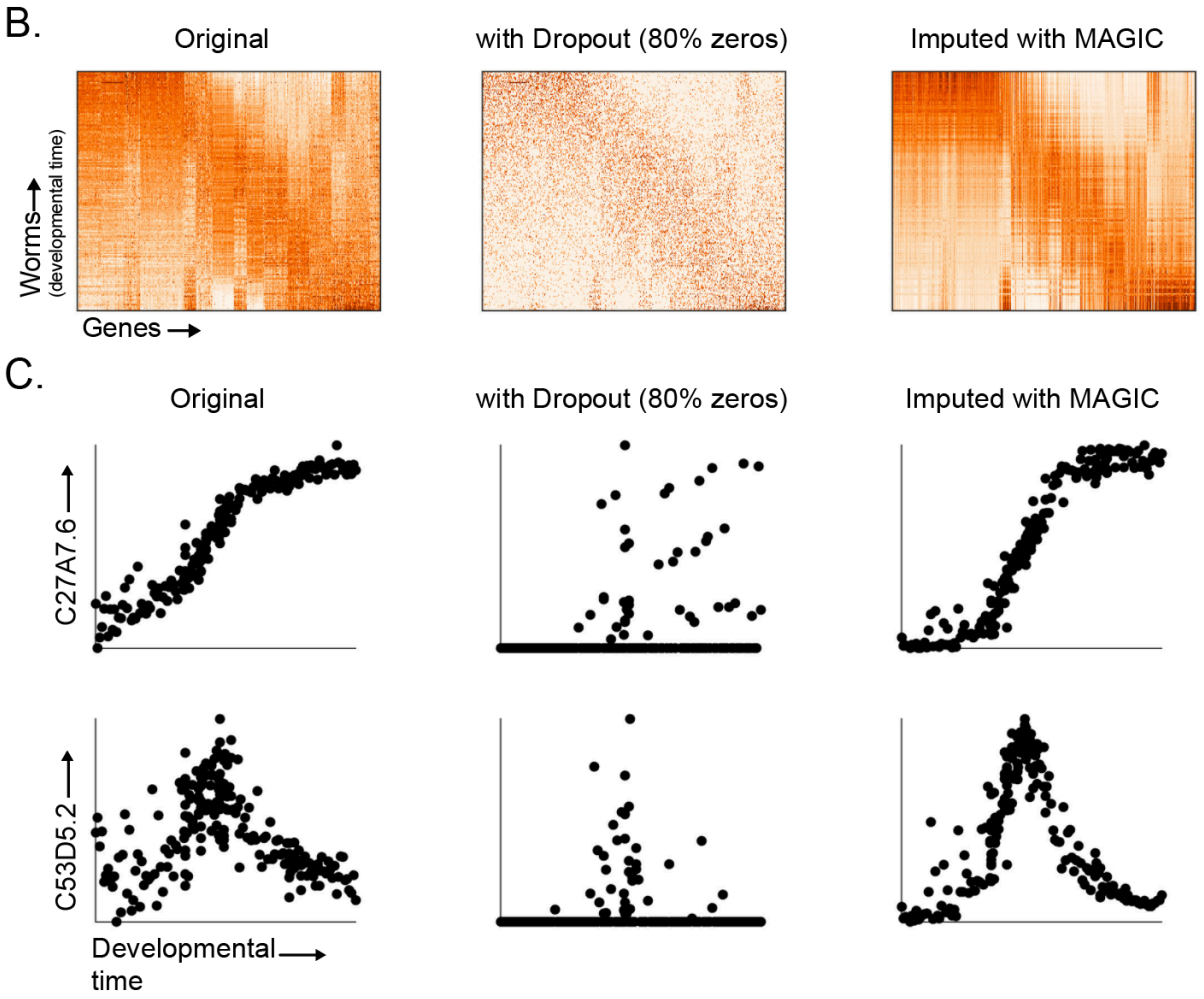


imputation with MAGIC



imputation with MAGIC

MAGIC recovers gene-gene relationships in an artificially dropped-out dataset



Summary of data denoising

- Diffusing or smoothing values over a graph can denoise data
- This kind of denoising is similar to averaging expression values across neighbors
- More diffusion = more denoising

What questions do you have?

Please submit on Slack

Batch correction

Batch Effect in Single Cell Data

Systematic, non-biological differences between samples due to measurement conditions

Differences could be due to ambient conditions (temperature, humidity), machine calibration, differences in titration, bead/antibody batch, etc

Sometimes refers to actually actual biological differences but those that are “uninteresting” (background demographics rather than immediate drug effect).

Problem with Batch Effect

Samples become hard to compare

All genes, cell types seem different!

Batch Correction

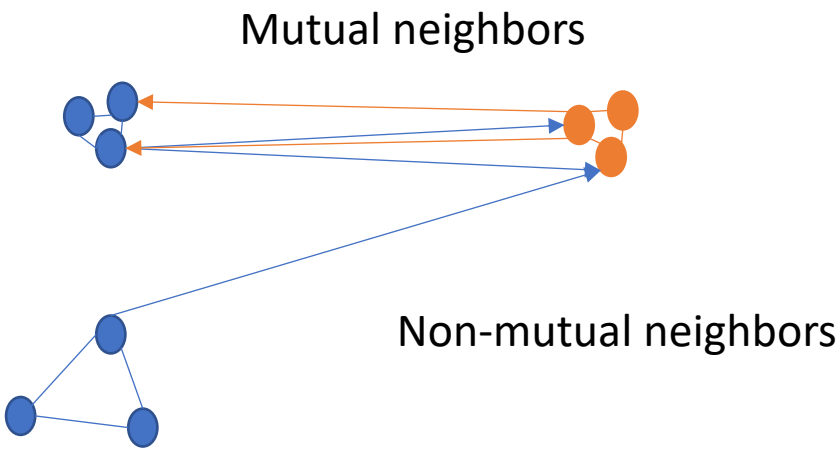
Take off artifactual variation and keep biological variation!

Batch correction methods

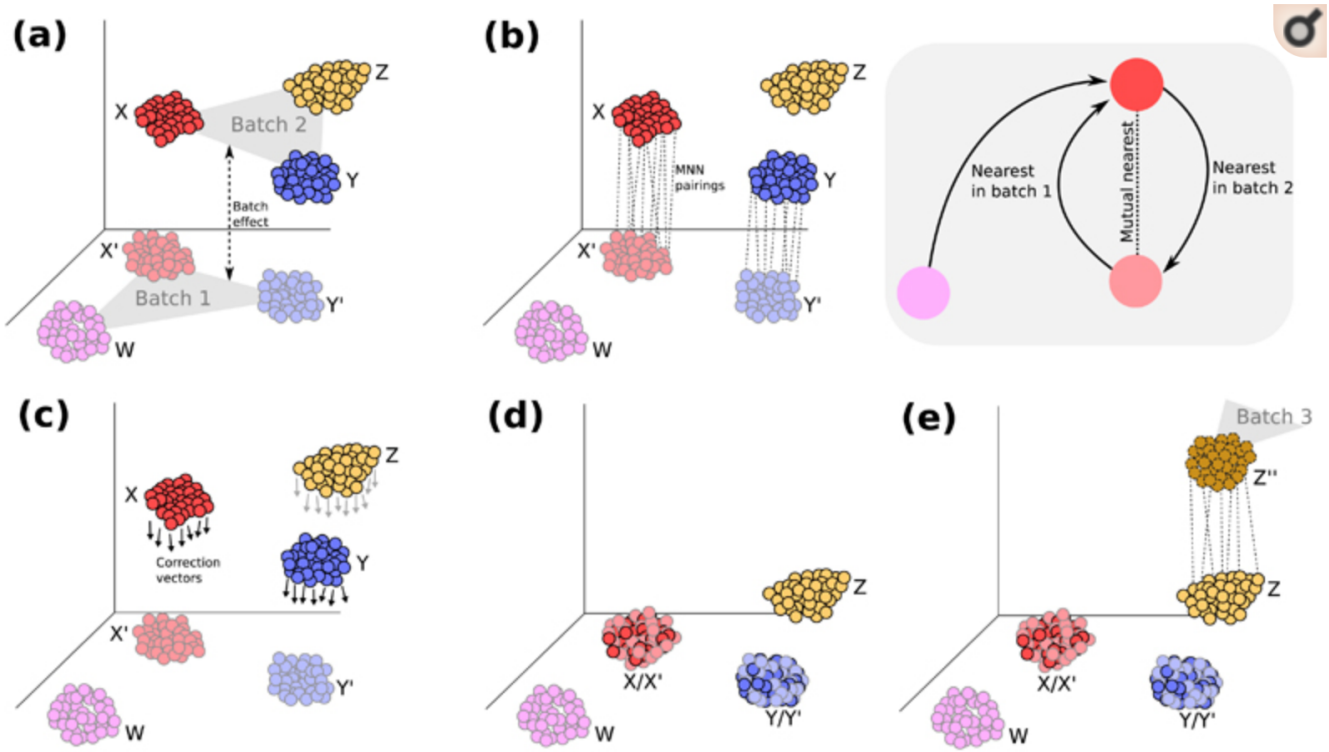
- Several batch correction methods exist but all have drawbacks
- CCA : uses a variant of PCA that brings two sets of samples to a common space
  - Can cluster or perform other analysis in common space but cannot go back to original space
  - Finds linear axes of common variation
  - Thrown off by non-matching populations
- Harmonic Alignment: Aligns diffusion dimensions instead of PCA dimensions, uses MAGIC to go back to original space
  - Aligning diffusion components is difficult, currently limited to rigid rotations of diffusion space
  - We’re working on it!

Mutual Nearest Neighbors

- Creates a graph between two datasets
- Nearest neighbors in the other dataset could be “matching cells”
- But they have to be mutual!

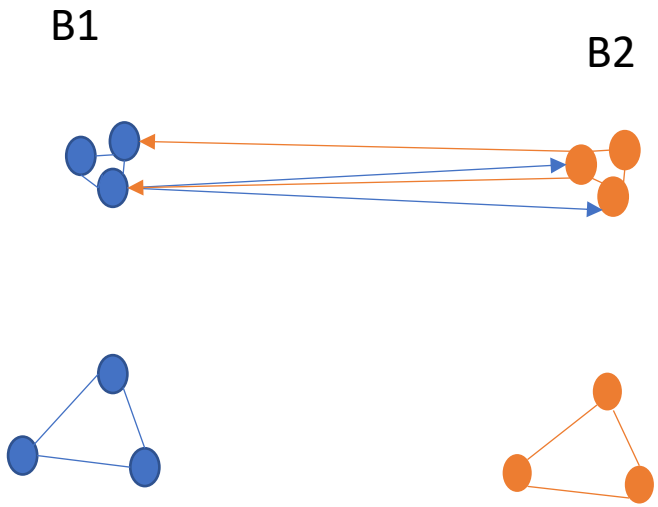


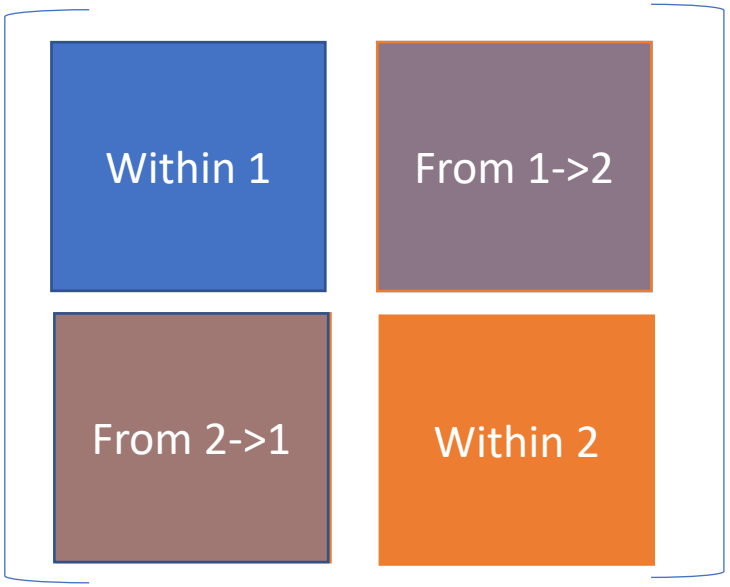
Mutual Nearest Neighbors



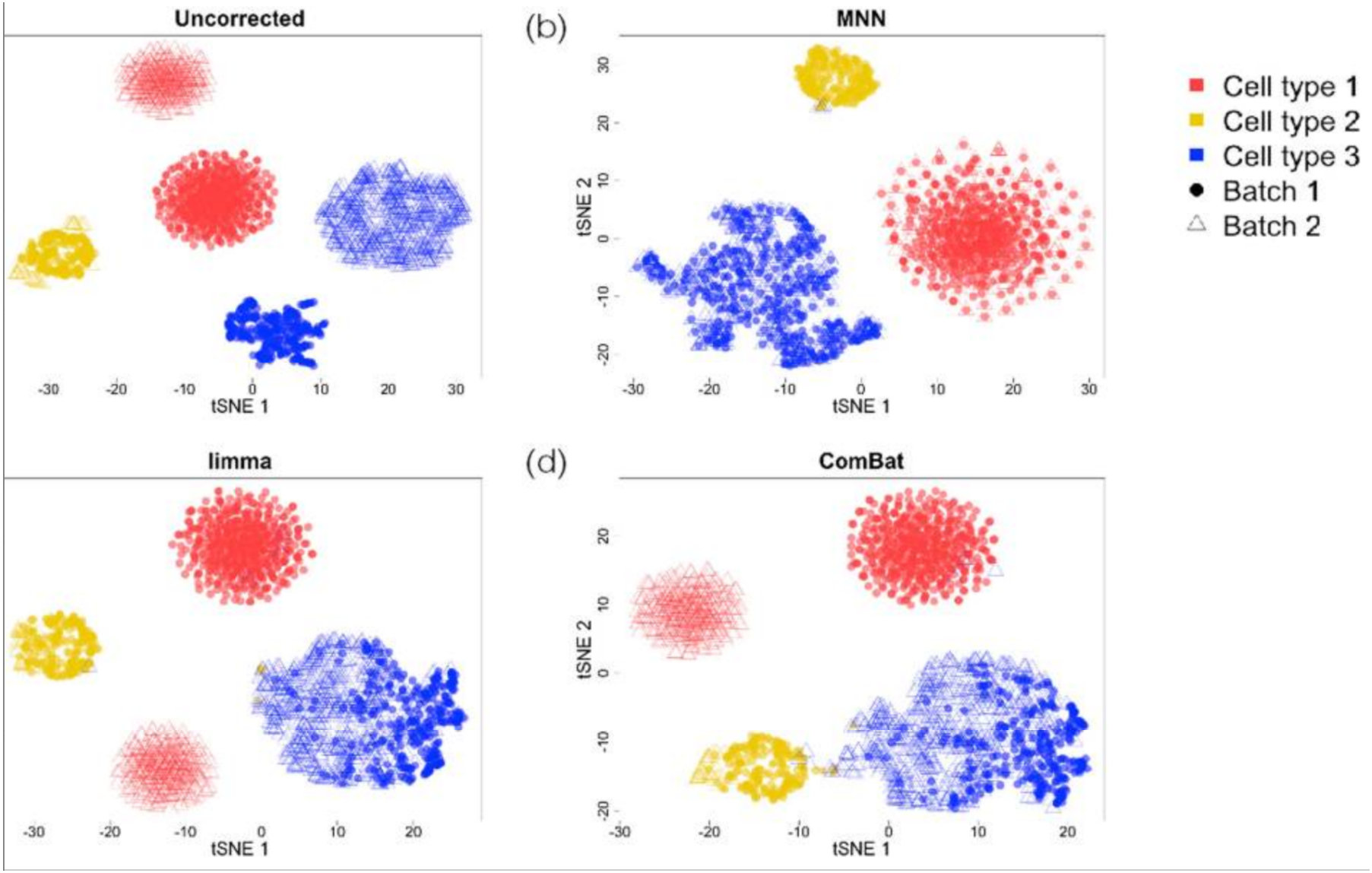
Use MNNs to find out the direction of the batch effect and correct it.

MNN with MAGIC





Uses softer affinity matrix and pulls the data in to correct the result


Uncorrected
(b)
MNN
limma
(d)
ComBat
tSNE 1
tSNE 2
Cell type 1
Cell type 2
Cell type 3
Batch 1
Batch 2

Is mutual nearest neighbor (MNN) normalization linear or non-linear?

Linear

Non-linear

Summary of data denoising

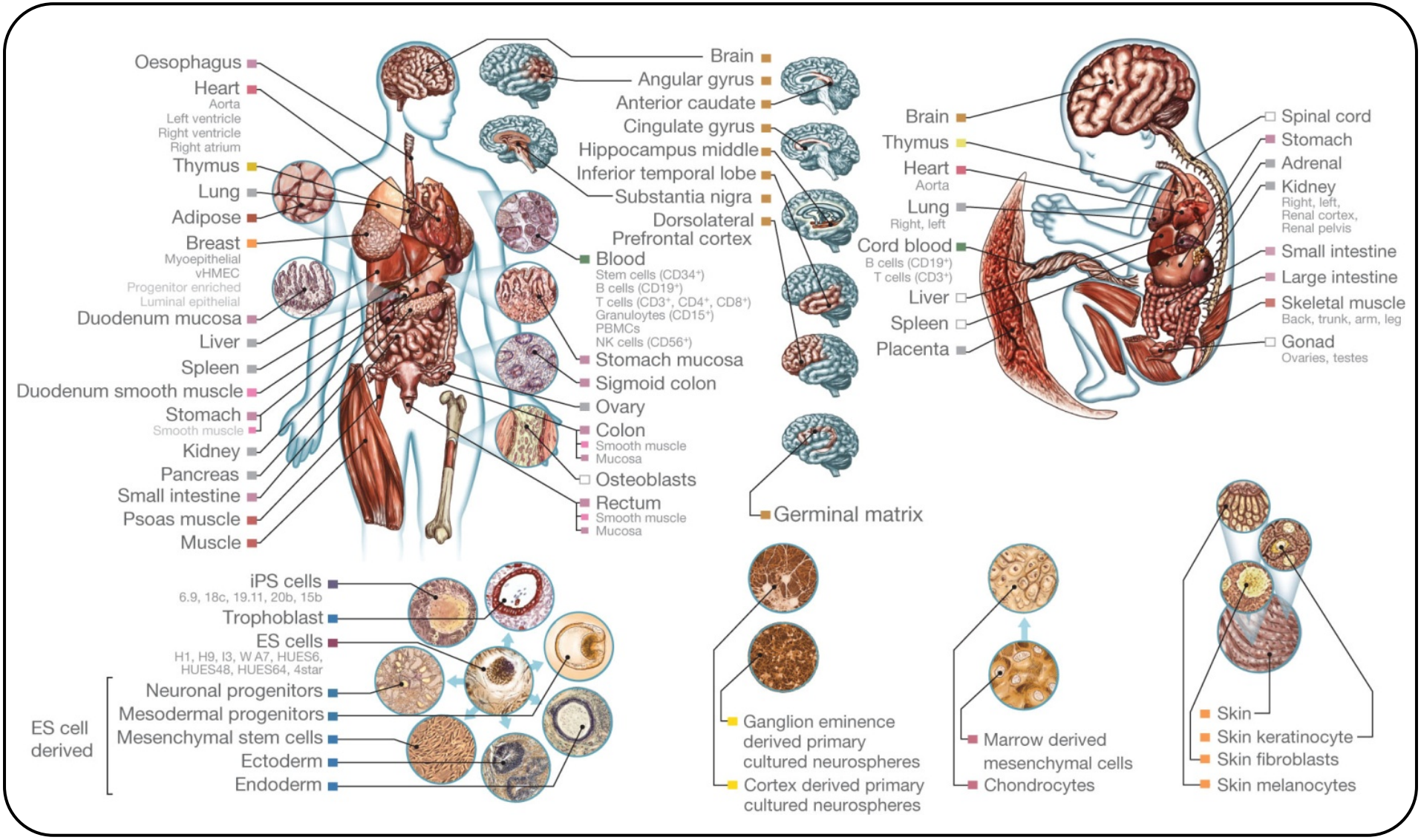
- Batch effects are sample-specific changes in measurements
- The goal of batch-normalization is to align cells of the same “type” across samples
- Mutual nearest neighbors (MNN) normalized batches by matching cells that are both close to each other

What questions do you have?

Please submit on Slack

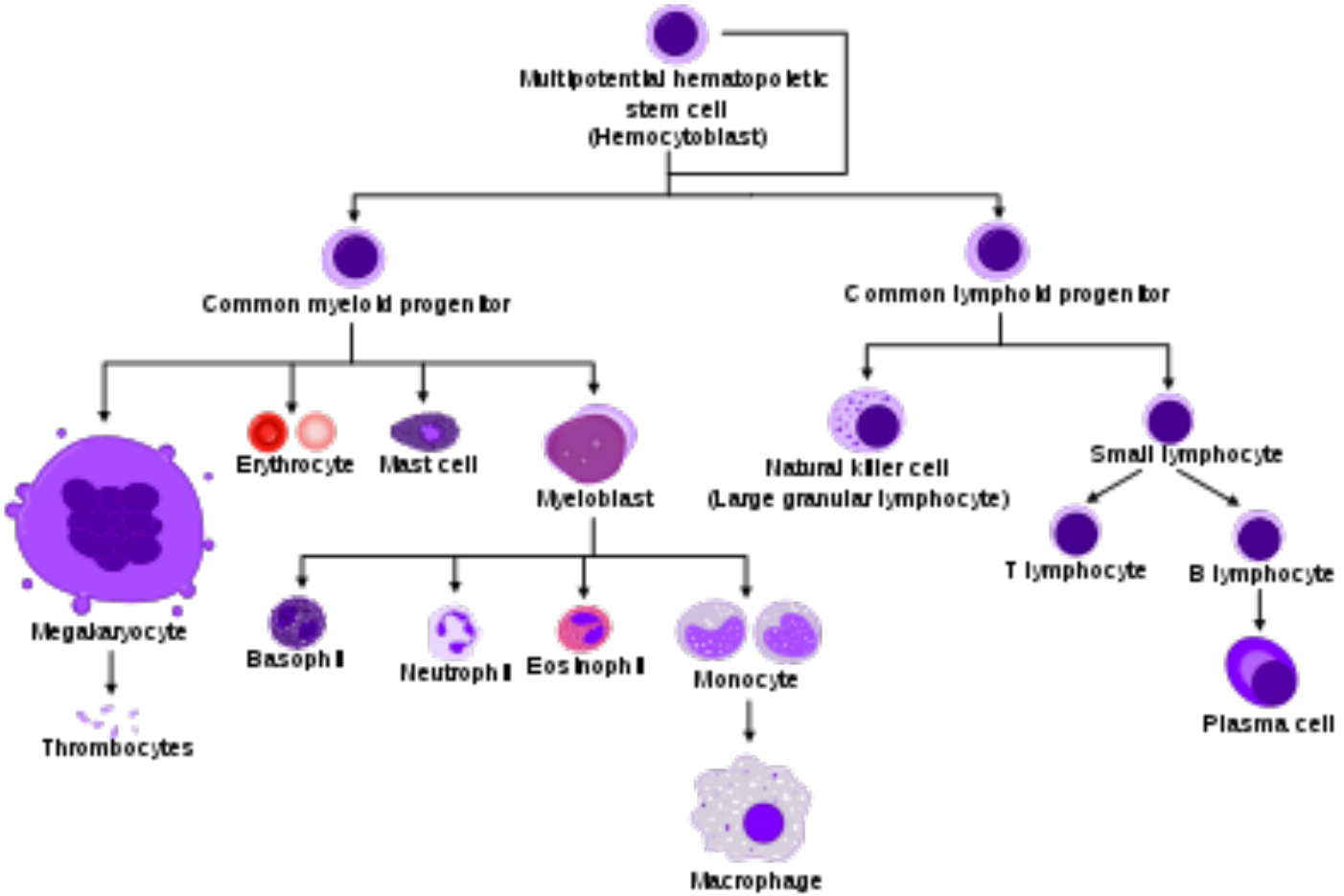
Clustering

High Degree of Complexity

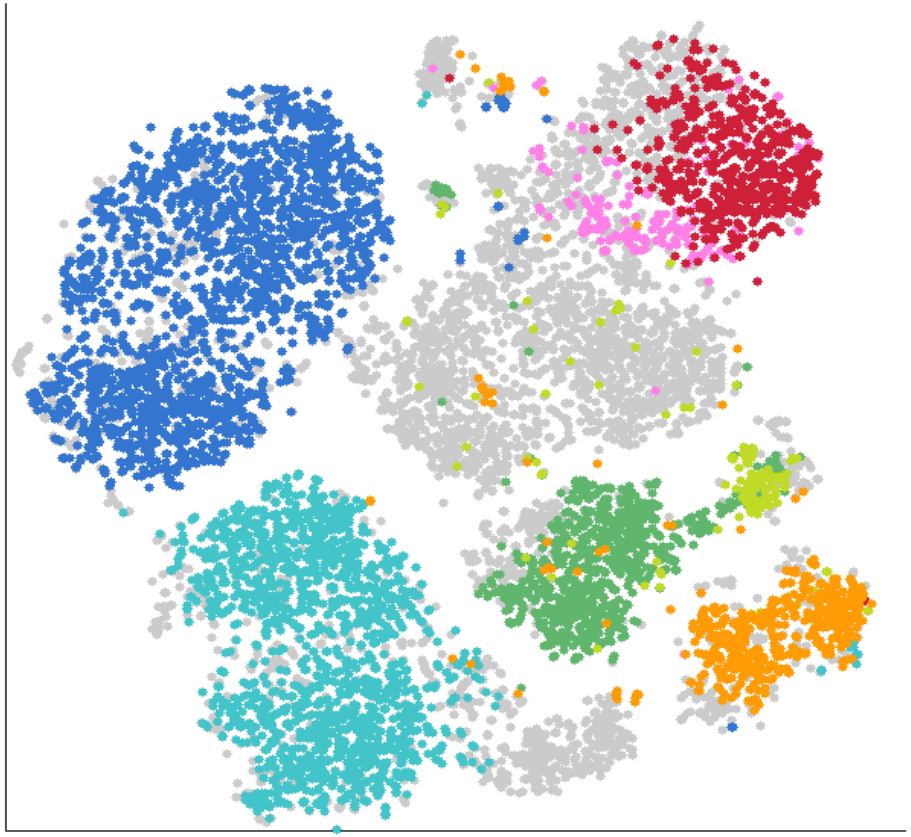


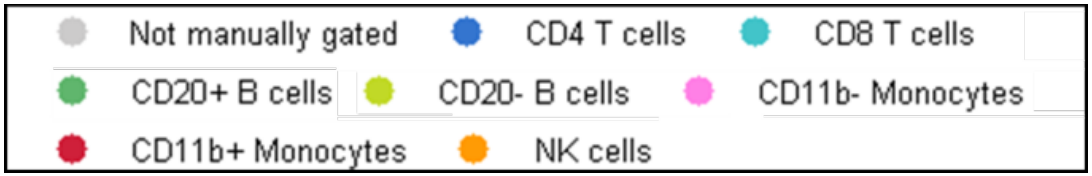
Roadmap Epigenomics Consortium Nature 2015

Phenotyping in Biology



tSNE Map of Immune Cells

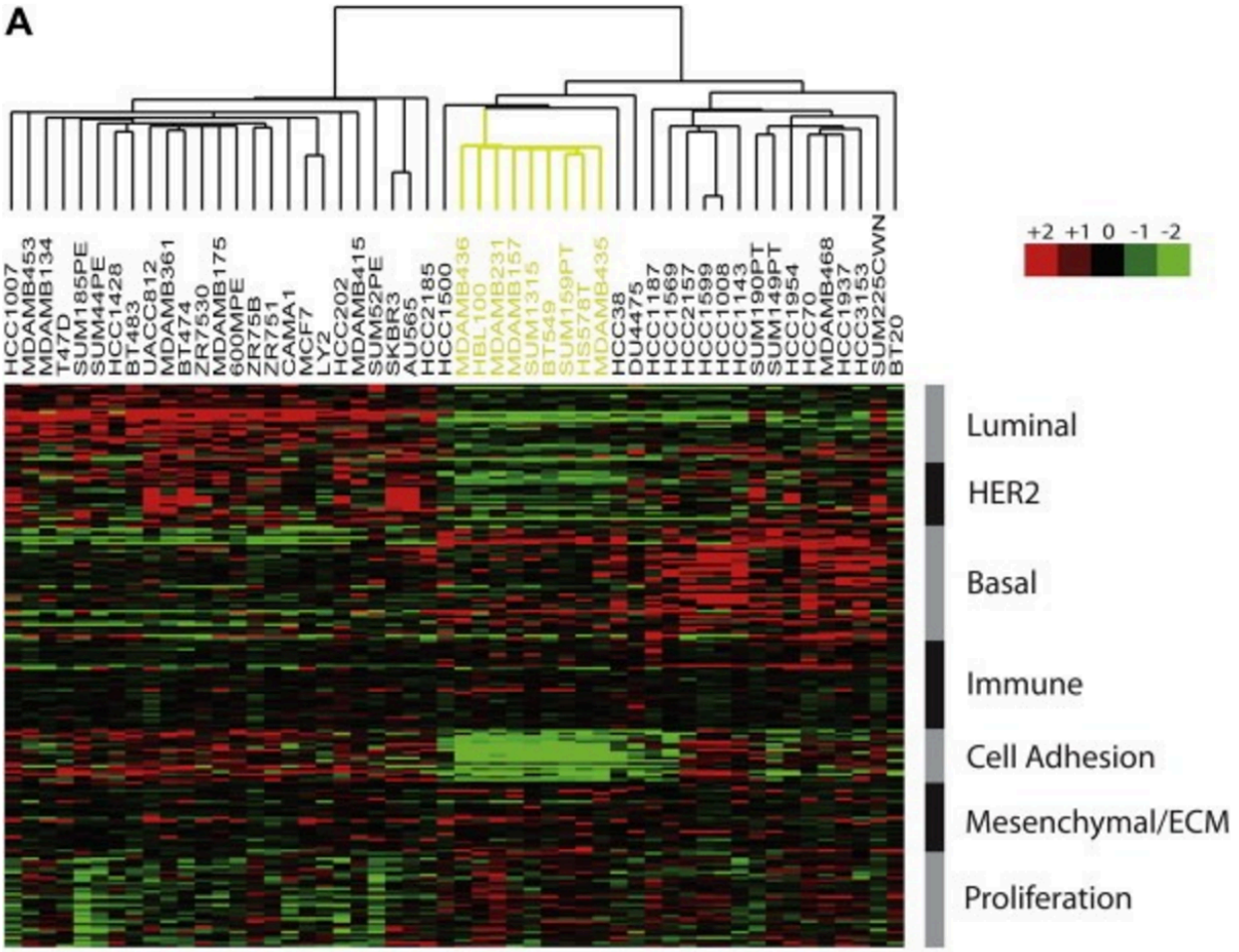




van der Maaten JMLR 2008, 2014

Amir *et al.* Nat. Biotech 2013

Breast Cancer Subtypes



Deconstructing the molecular portraits of breast cancer

Aleix Prat [1, 2, 3] and Charles M. Perou [1, 2, 3]

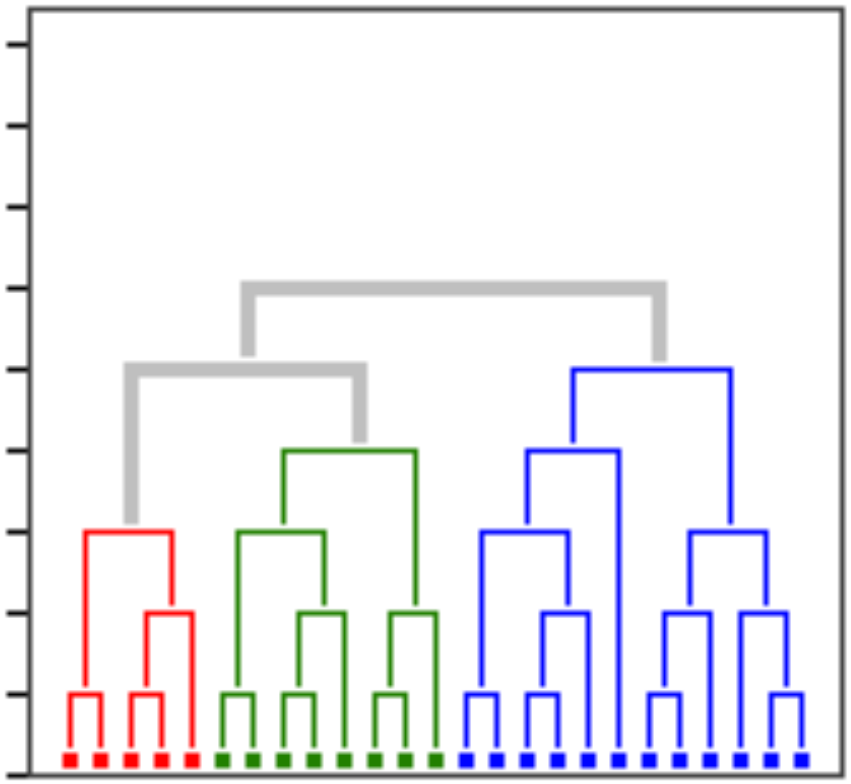
Author information ► Article notes ► Copyright and License information ►

Clustering Used in Many Ways in Biology

- Clustering gene expression profiles to find “modules” or groups of genes that work together
- Clustering patient data to see if patients have similar disease
- Clustering cells to find different cell types

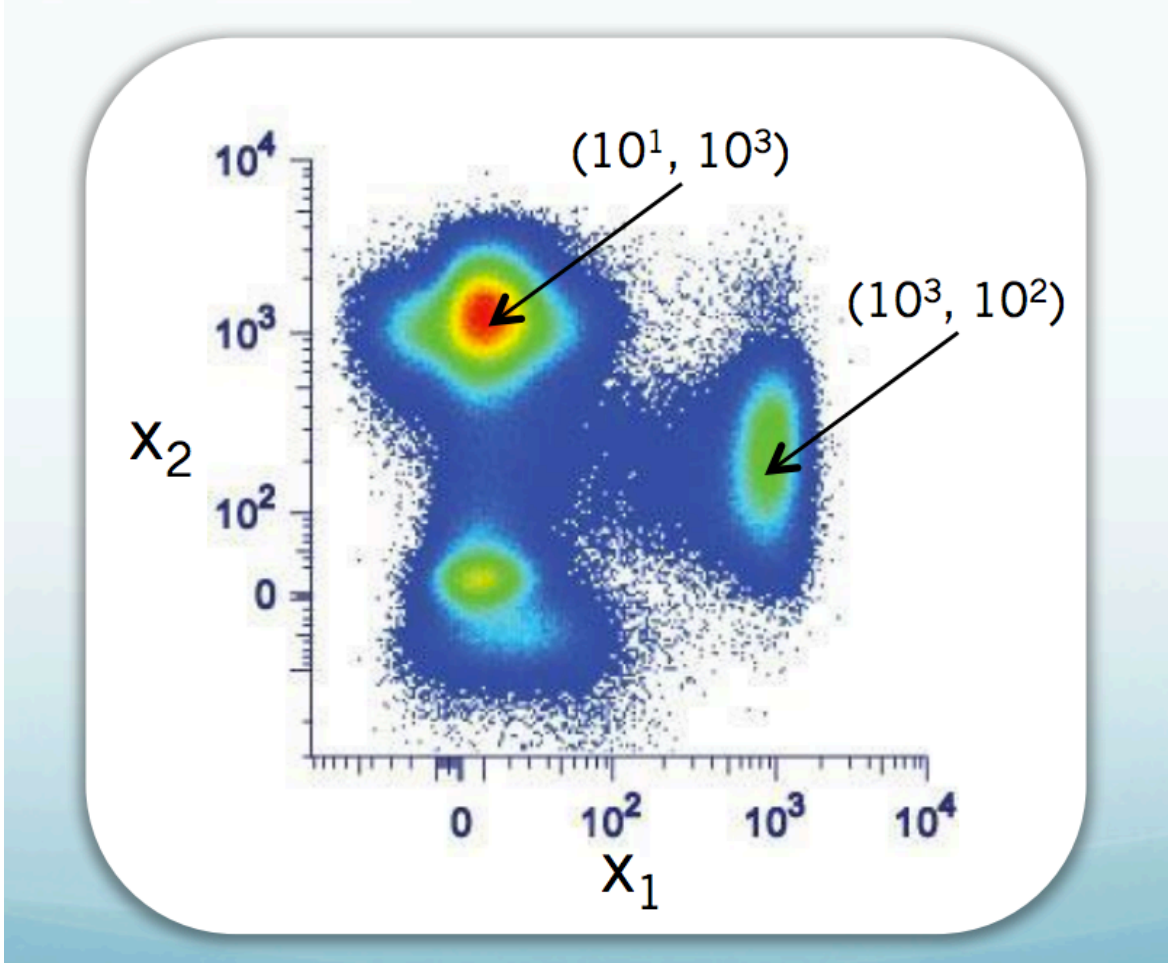
What is clustering?

Grouping of Similar Objects



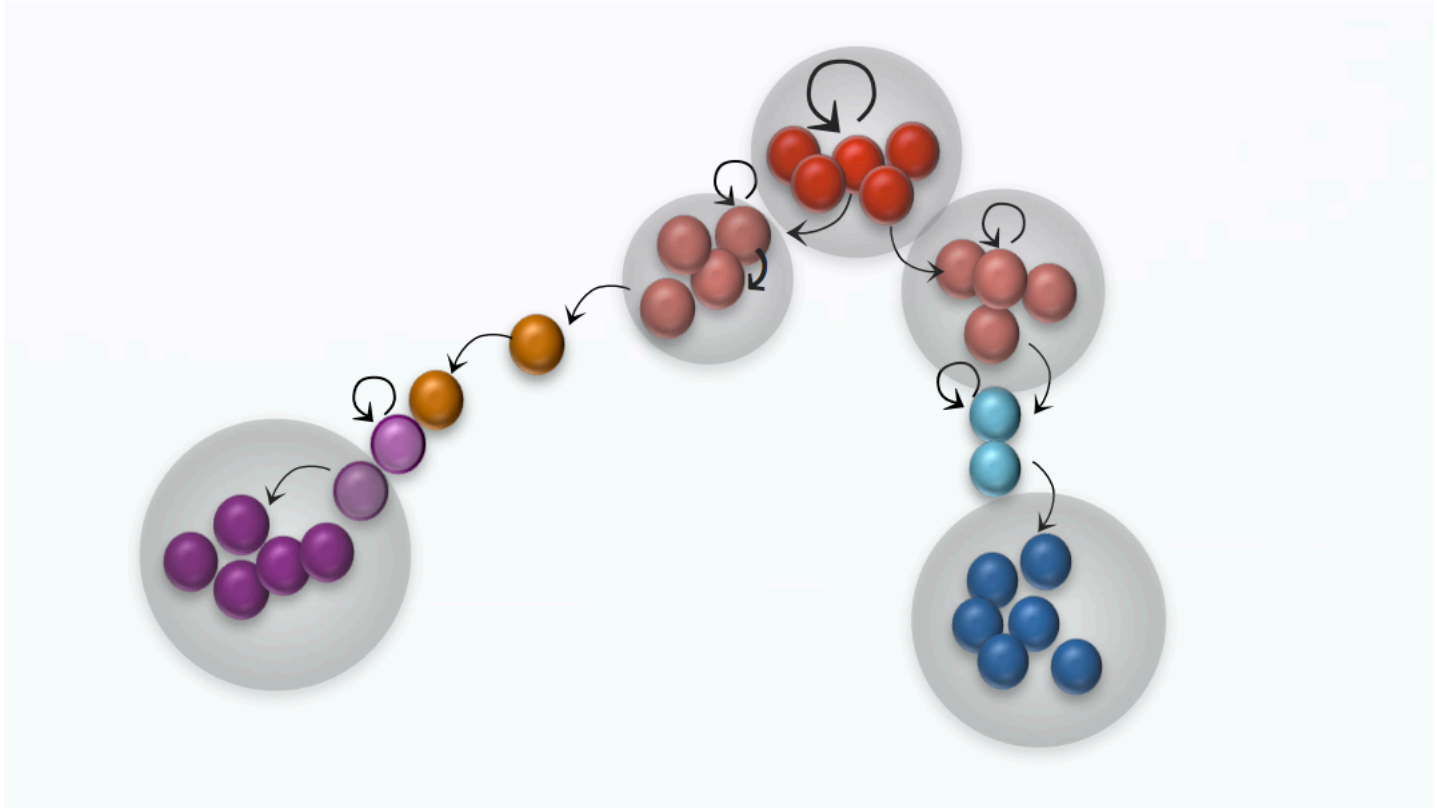
Linkage Clustering, Community detection

Density Centers



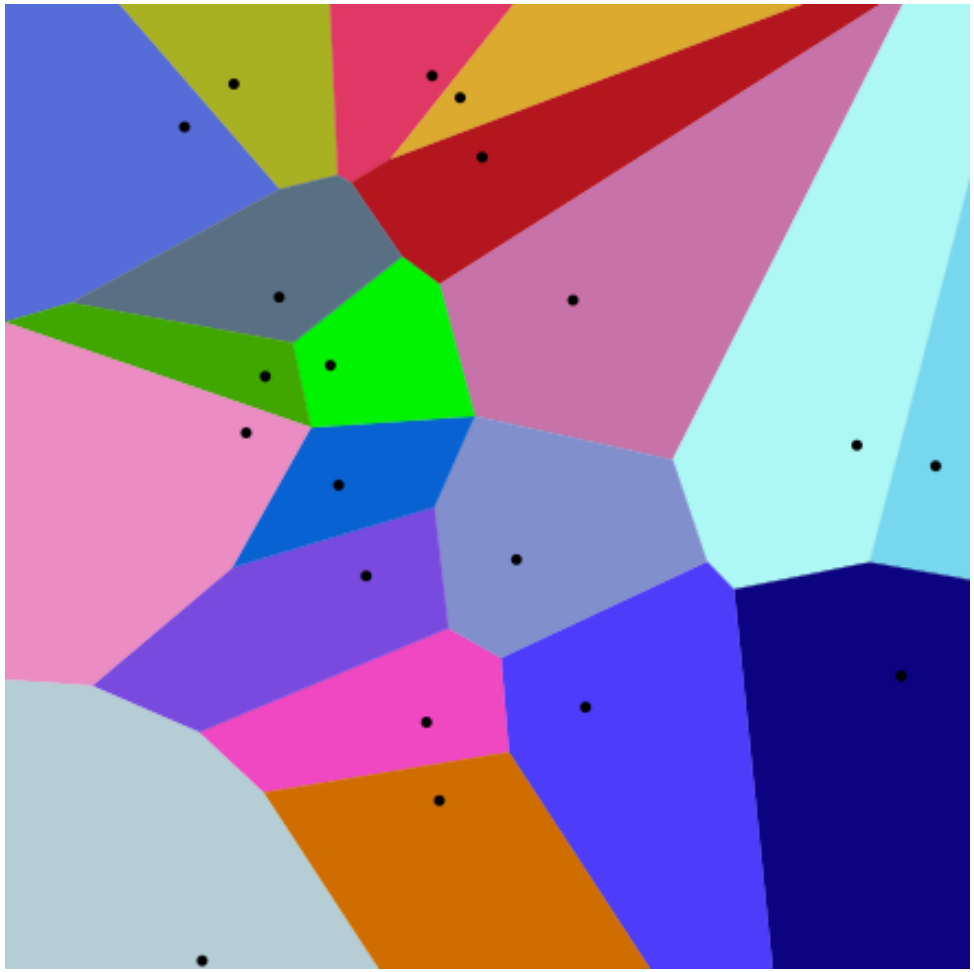
DBSCAN, Gaussian Mixture Model

Metastable States in A Manifold



PHENOGRAPH

A partitioning of the data space



Voronoi Diagram

K-MEANS (Macloed 1967), Fidduccia Matheyses

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 $X = \{x_1, x_2, \ldots x_n\}$
- Partition into k clusters
- Such that the within-cluster variance is minimized $S = \{S_1, S_2, \ldots S_k\}$

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

K-Means Clustering

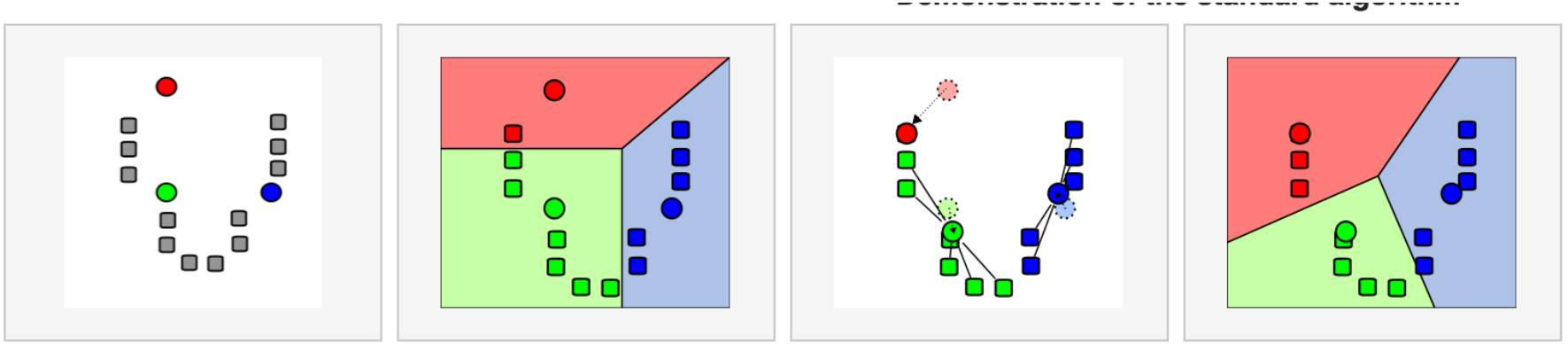
- Randomly partition data into k clusters
-
- **Update step**: Compute the means of clusters

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

- **Assignment step:** Reassign each x_i to cluster S_j that has the nearest mean u_j

Nearest mean minimizes squared Euclidean distance

Algorithm Iterations



Will this process stop?

Yes: K-means Converges

- Why?
- Each step LOWERS mean squared error
- Update of means lowers variance
- Reassignment of points to nearby means lowers variance

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

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

How do you pick K?

- If you keep increasing K the fit keeps getting better!
- K has to balance between overfitting, having too many parameters and modeling the data
- Akaike Information Criterion:

- Model selection

$$AIC = 2m - 2\ln(L)$$

$$L = P(x \mid \theta) = likelihood$$

Cluster Silhouette

- Measures the appropriateness of cluster assignments
- Within cluster dissimilarity
- Between cluster dissimilarity:

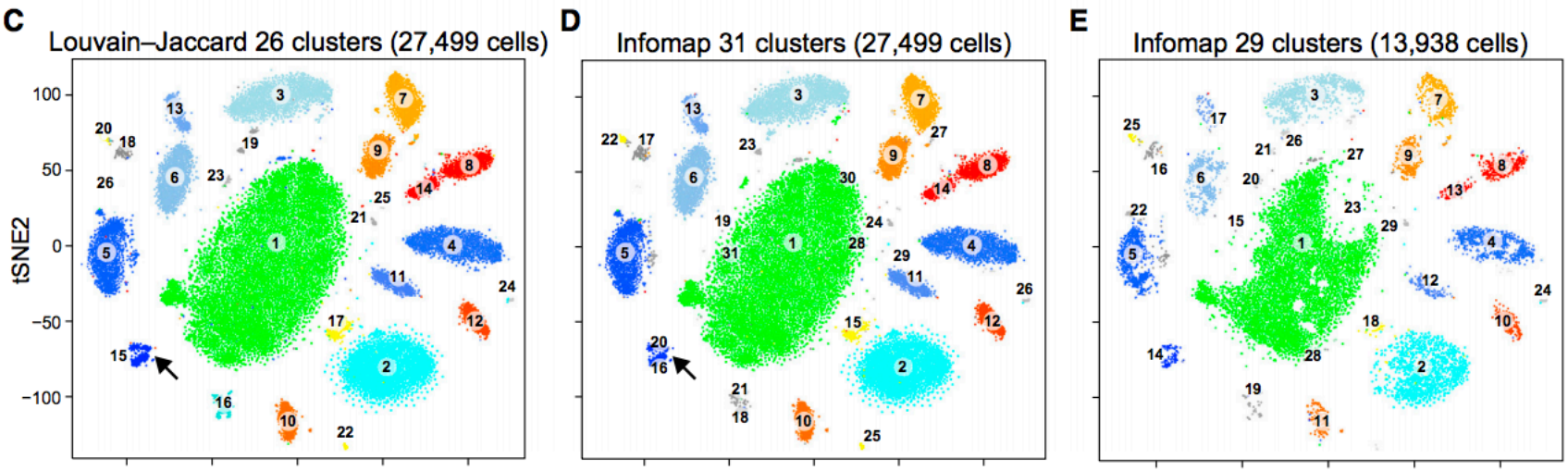
$$a(i) = \sum_{y \in S_i} \| x_i - y \|$$

- Silhouette score
- Can be used for picking k

$$b(i) = \min_{j \neq i} \sum_{y \in S_j} \| x_i - y \|$$

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

Clustering in Single-Cell RNA-seq



Resource

Cell

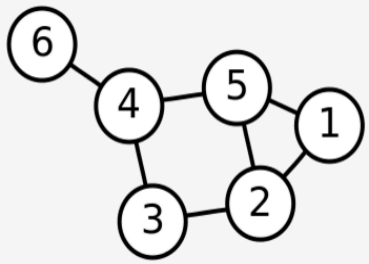
Comprehensive Classification of Retinal Bipolar Neurons by Single-Cell Transcriptomics

Graph Laplacian

- W = similarity or adjacency matrix
  - Adjacency matrix can be a binarized 0-1 matrix
- D = Degree Matrix

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

- Graph Laplacian: L = I

Labeled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

Normalizations of the Laplacian

- Two common normalizations:

$$L_{\text{sym}} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$
$$L_{\text{rw}} := D^{-1} L = I - D^{-1} W.$$

- First is called a symmetric normalization
  - Spectral clustering on EV of this [Ng et al 2002]
- Second is called a random walk normalization
  - Spectral clustering on EV of this [Shi and Malik 2000]
  - Related to Markov transition matrix

Laplacian vs Markov Matrix

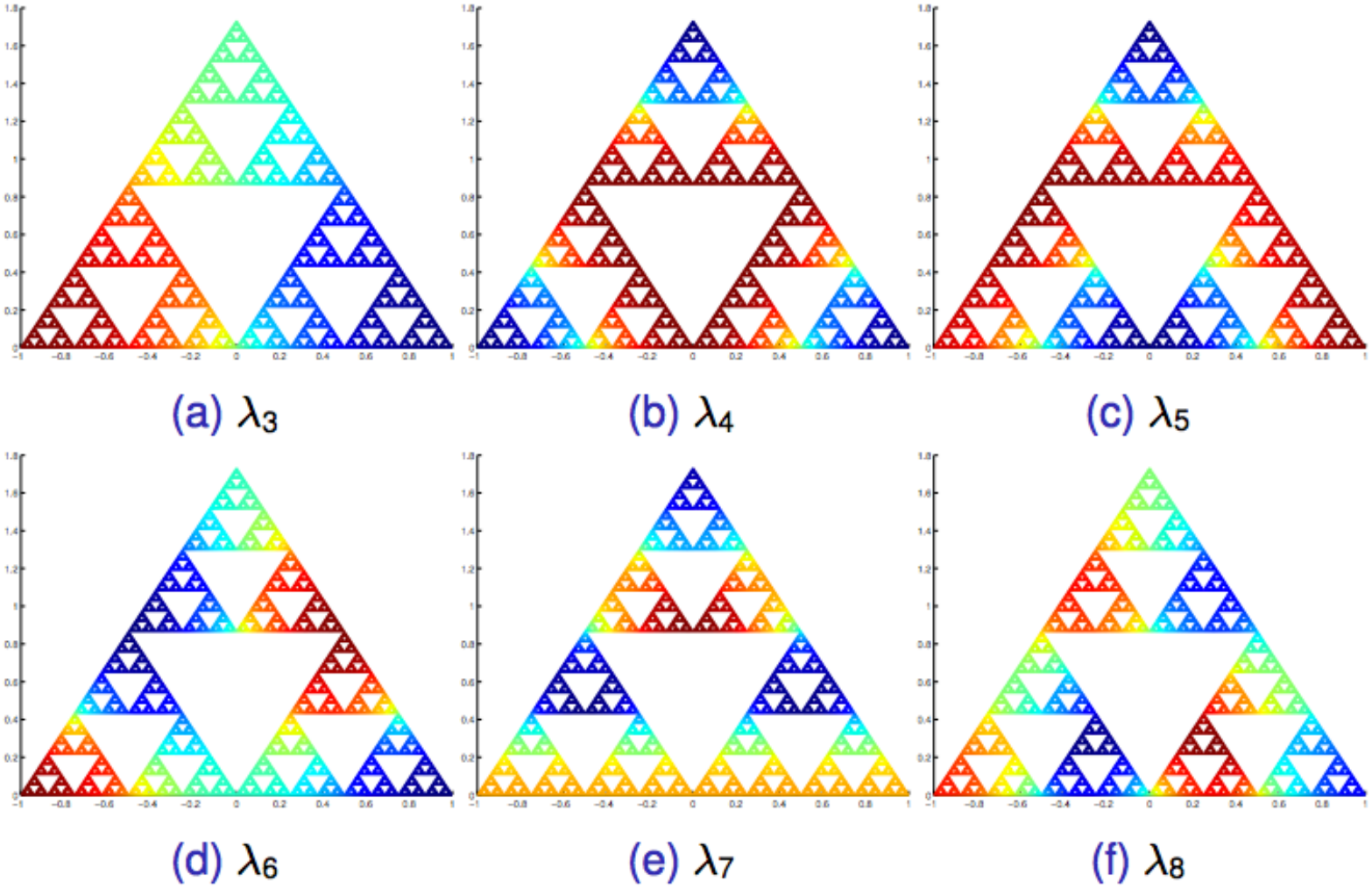
- $L=I-D^{-1}W$ is Laplacian
- $M=D^{-1}W$ is the Markov affinity matrix
- Same eigenvectors
- The ordering of eigenvectors is flipped
- Eigenvalues are now $1-\lambda_i$

Laplacian vs Graph Laplacian

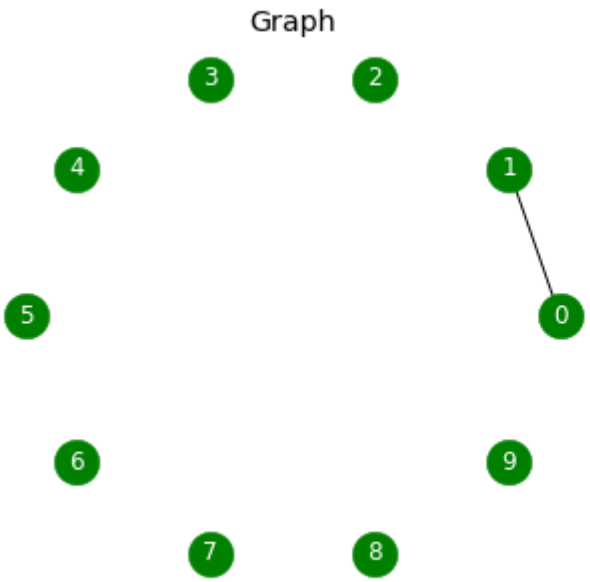
- The ***laplacian operator*** is the divergence of the gradient also denoted

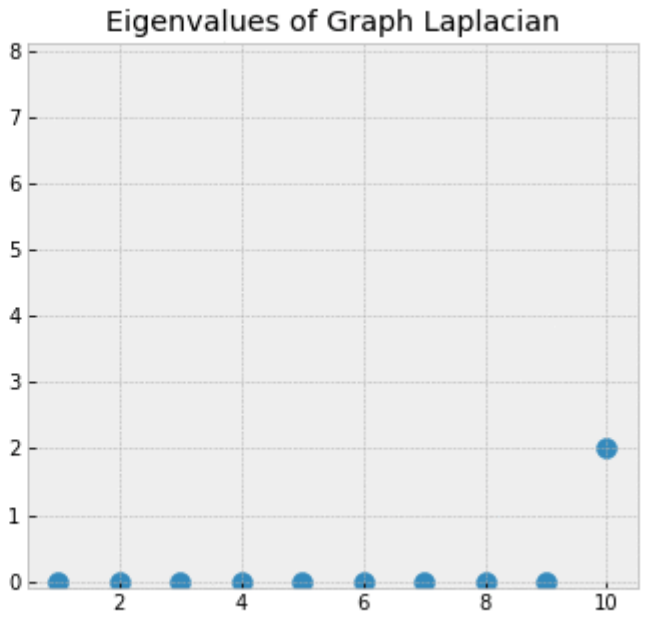
$$\nabla^2 = \left[\frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_N}\right] \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \vdots \\ \frac{\partial}{\partial x_N} \end{bmatrix} = \sum_{n=1}^{N} \frac{\partial^2}{\partial x_n^2}$$

- The graph discretized Laplacian operator
- The transition matrix gives derivatives
- I-M gives something akin to second derivatives



Same eigenvectors as affinity matrix

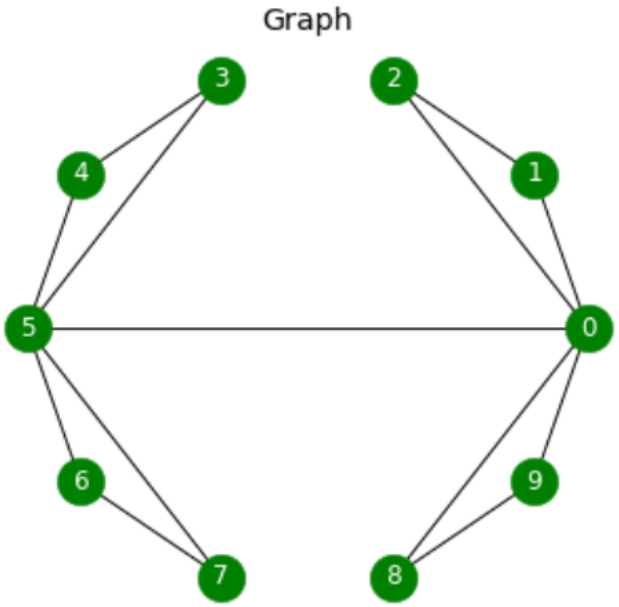


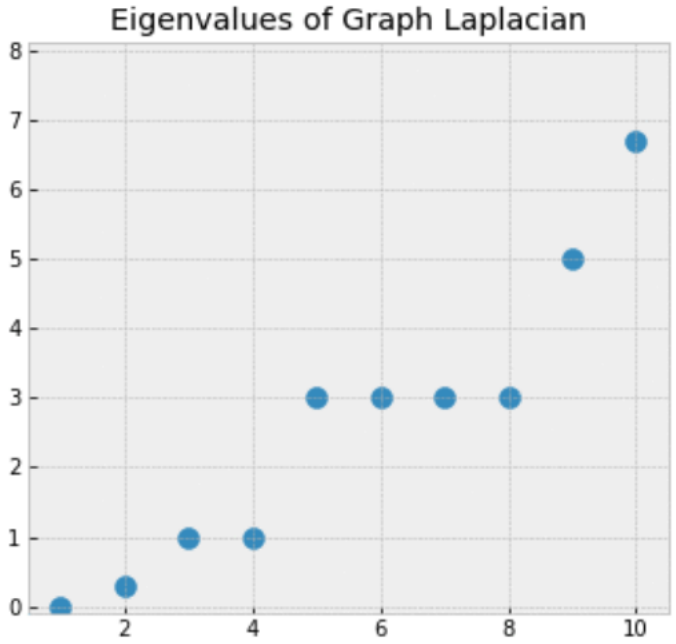


When there are 10 disconnected nodes all 10 eigenvalues are 0

As we add connections we lift eigenvalues

First eigenvalue will always stay 0

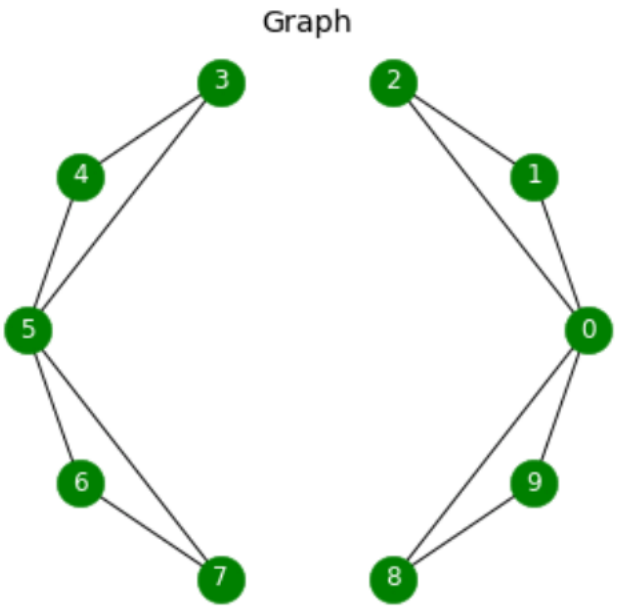


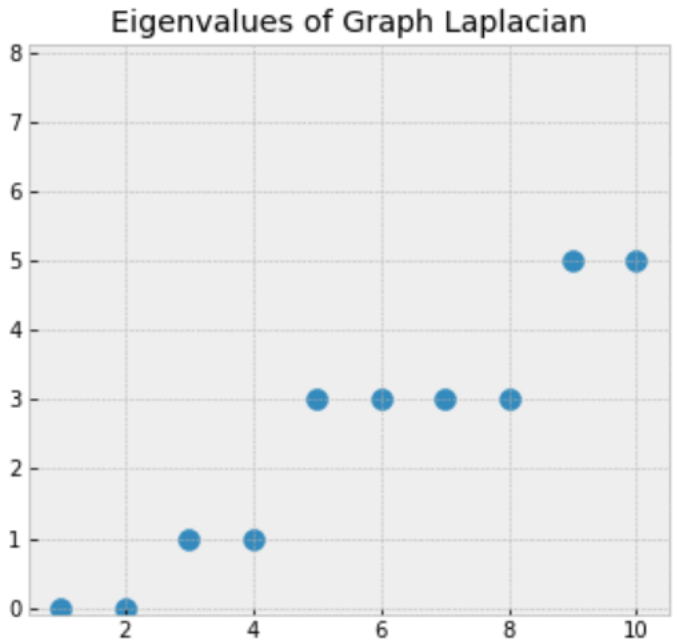


First nonzero eigenvalue is called the “algebraic connectivity” or “Fiedler value”

Estimates density of the graph.

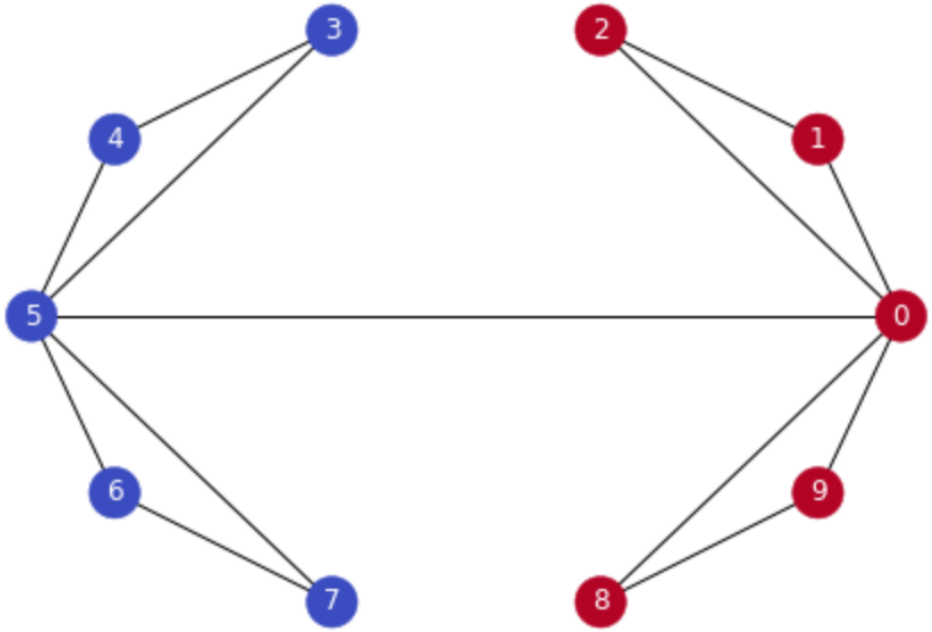
If this graph was densely connected, then it would be 10





Number of 0 eigenvalues is the number of connected components

Fiedler Vector

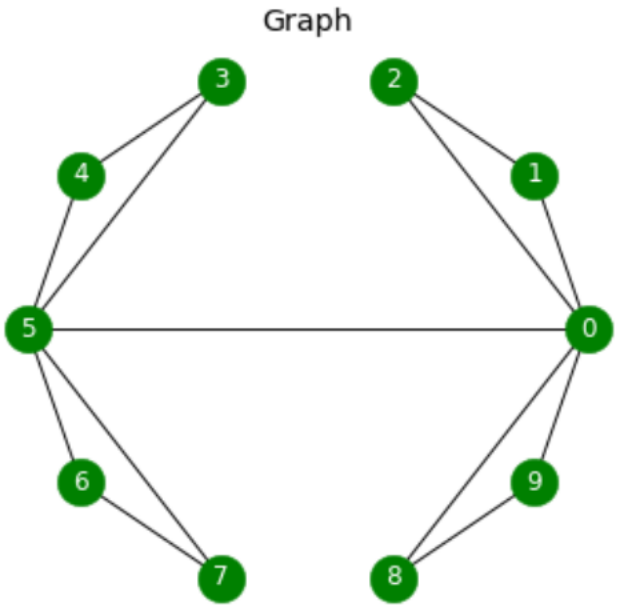


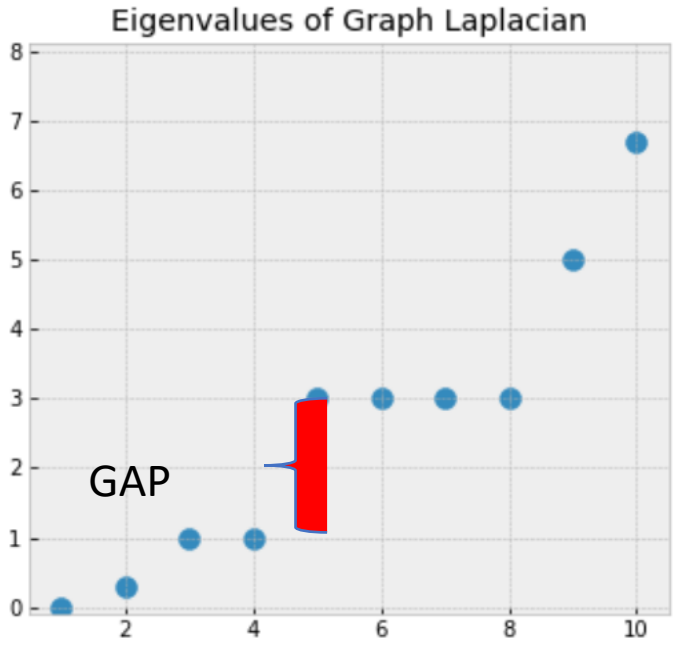
Second-smallest Eigenvalue is called Fiedler value, corresponding vector is the Fiedler vector

Fiedler value:= approximates minimum cut needed to partition graph

Value if graph was already. In two components?

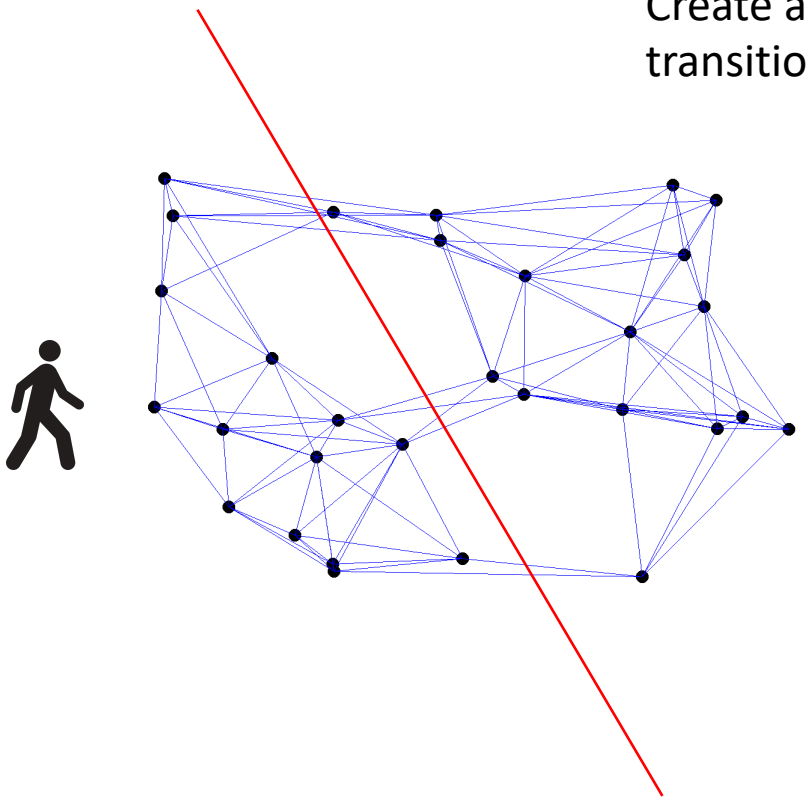
Vector can be used for partitioning, positive in one partition, negative in another





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

Connection to Random Walk



Create a cut such that the random walk seldom transitions from one cut to another

$$cut(S_i, S_j) = \sum_{x \in S_i, y \in S_j} A(x, y)$$

$$\sum_{i=1}^{k} \frac{cut(S_i, S_i')}{|S_i|}$$

Since an eigenvector encodes a path in the data splitting the eigenvector gives lowest chance of random walk crossing boundaries

Cut Algorithm

1. Given an image or image sequence, set up a weighted graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ and set the weight on the edge connecting two nodes to be a measure of the similarity between the two nodes.
2. Solve $(\mathbf{D} - \mathbf{W})\boldsymbol{x} = \lambda \mathbf{D}\boldsymbol{x}$ for eigenvectors with the smallest eigenvalues.
3. Use the eigenvector with the second smallest eigenvalue to bipartition the graph.
4. Decide if the current partition should be subdivided and recursively repartition the segmented parts if necessary.

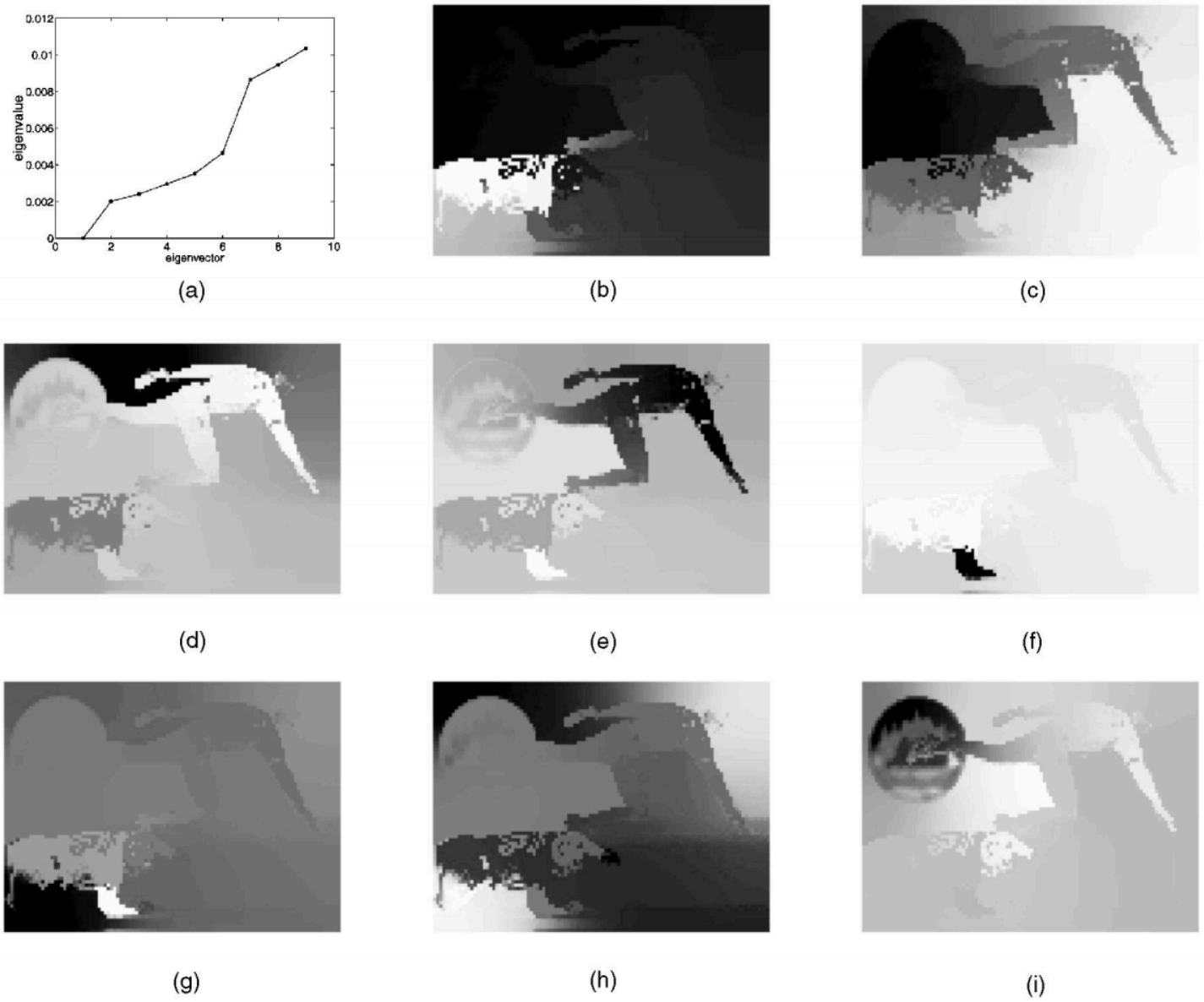
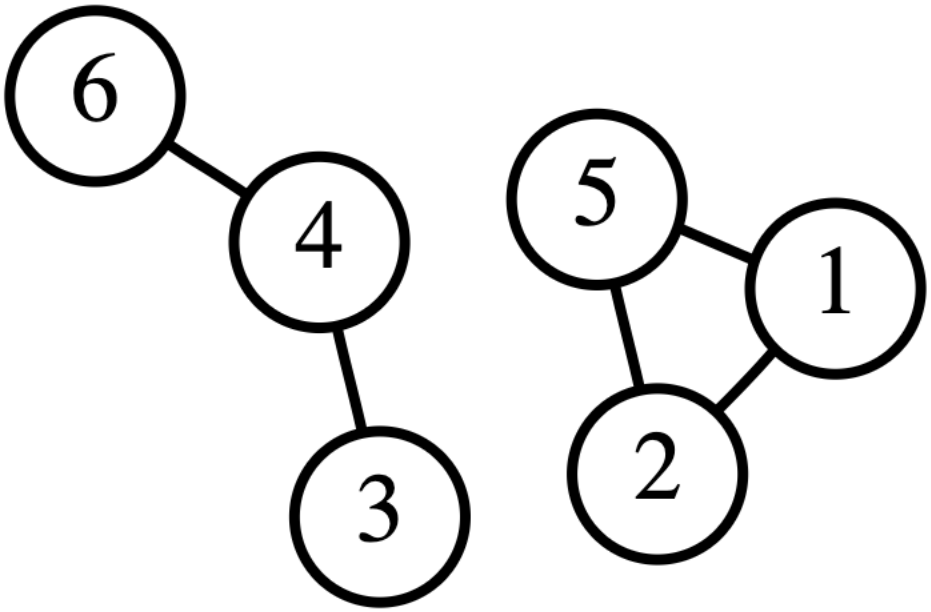


Fig. 3. Subplot (a) plots the smallest eigenvectors of the generalized eigenvalue system (11). Subplots (b)-(i) show the eigenvectors corresponding the second smallest to the ninth smallest eigenvalues of the system. The eigenvectors are reshaped to be the size of the image.

Example



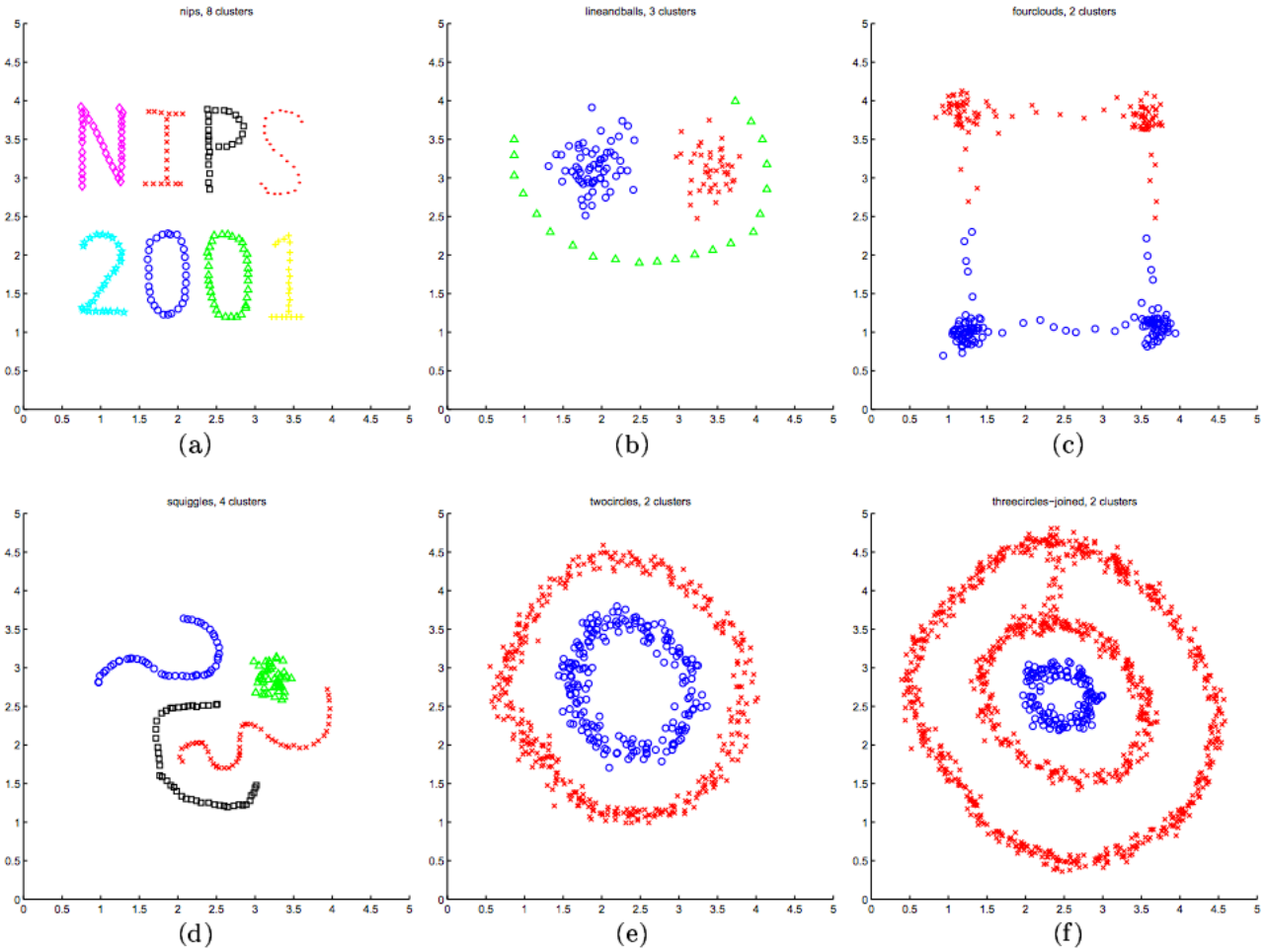
K-means Spectral Clustering

- Compute first K eigenvectors of L (of the smallest eigenvalue)
  - k-dimensional representation of datapoints
- Use K-means clustering on these representations
- Why?

Spectral Clustering Algorithm

1. Form the affinity matrix $A \in \mathbb{R}^{n \times n}$ defined by $A_{ij} = \exp(-||s_i - s_j||^2/2\sigma^2)$ if $i \neq j$, and $A_{ii} = 0$.
2. Define D to be the diagonal matrix whose (i,i)-element is the sum of A's i-th row, and construct the matrix $L = D^{-1/2}AD^{-1/2}$.[1]
3. Find $x_1, x_2, \ldots, x_k$, the k largest eigenvectors of L (chosen to be orthogonal to each other in the case of repeated eigenvalues), and form the matrix $X = [x_1 x_2 \ldots x_k] \in \mathbb{R}^{n \times k}$ by stacking the eigenvectors in columns.
4. Form the matrix Y from X by renormalizing each of X's rows to have unit length (i.e. $Y_{ij} = X_{ij}/(\sum_j X_{ij}^2)^{1/2}$).
5. Treating each row of Y as a point in $\mathbb{R}^k$, cluster them into k clusters via K-means or any other algorithm (that attempts to minimize distortion).
6. Finally, assign the original point s_i to cluster j if and only if row i of the matrix Y was assigned to cluster j.

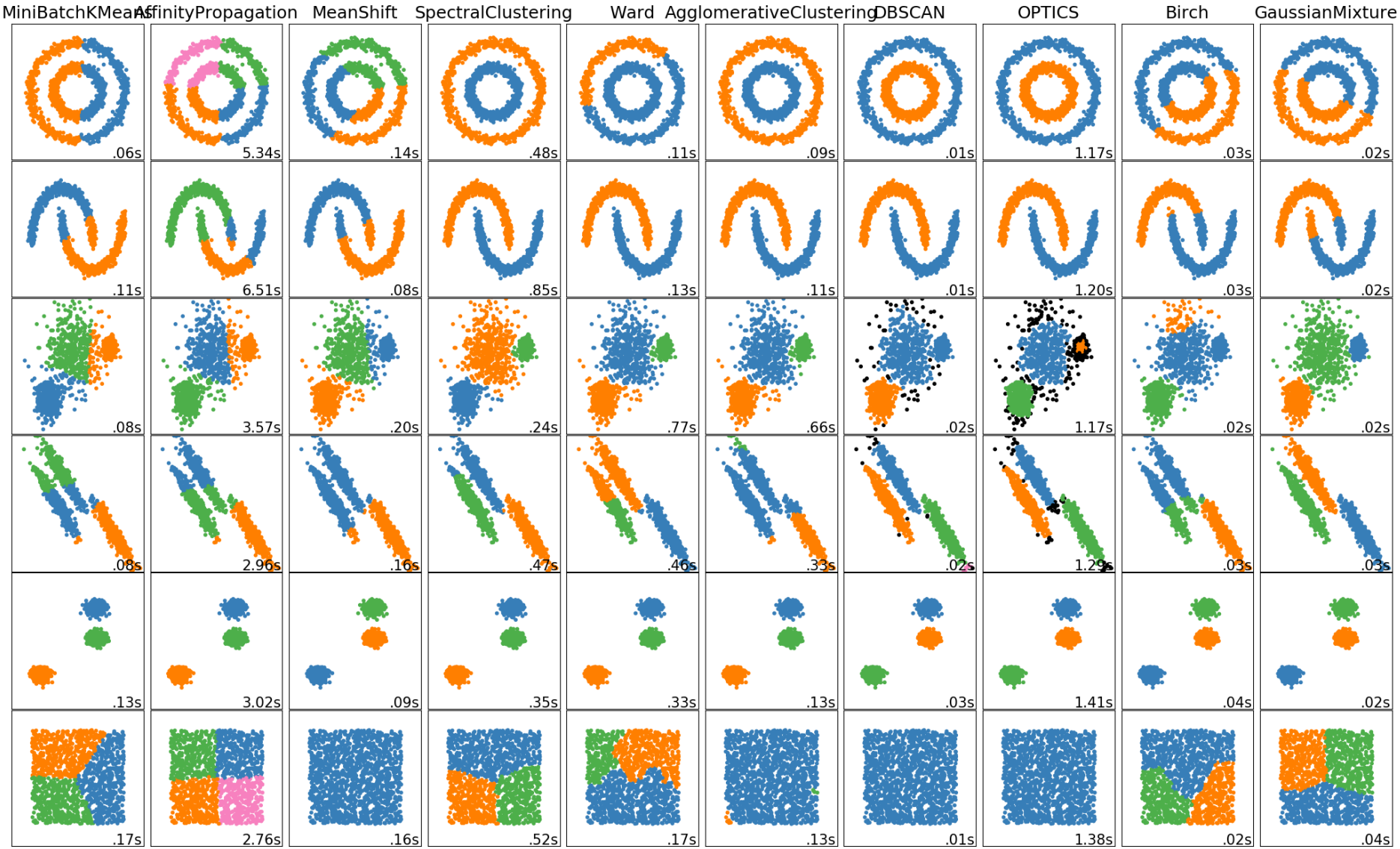
Illustration of Algorithm



What are advantages of this?

Advantages

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



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 minimized within-cluster distances using eigenvectors of the laplacian

What questions do you have about the lecture material from today?

Top