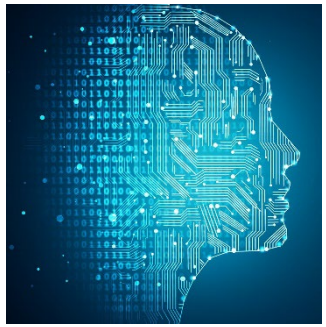
Machine Learning

Dimensionality Reduction for Visualization

Kevin Moon (kevin.moon@usu.edu)

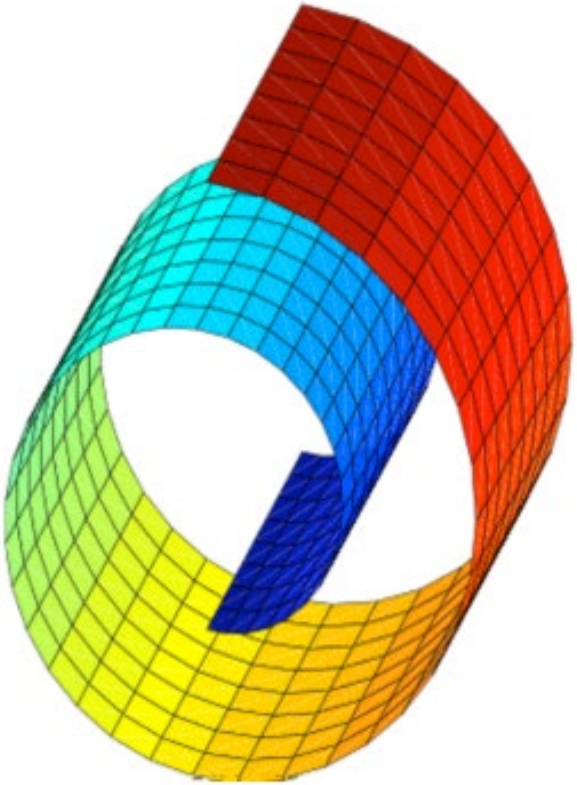
STAT/CS 5810/6655



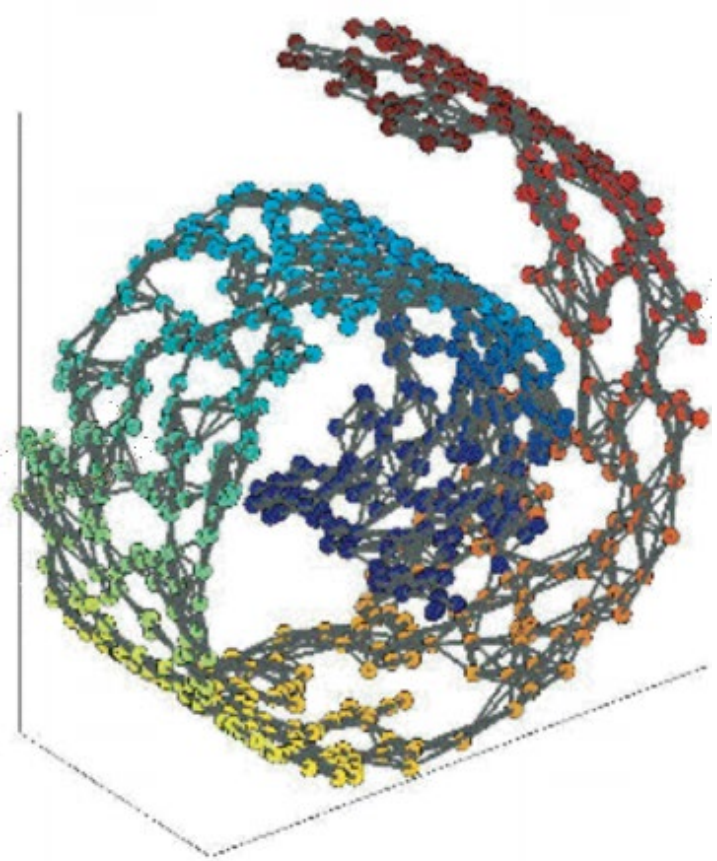
Outline

1. Manifold Learning
2. Visualizing with PCA
3. t-SNE
4. UMAP
5. PHATE
6. DIG

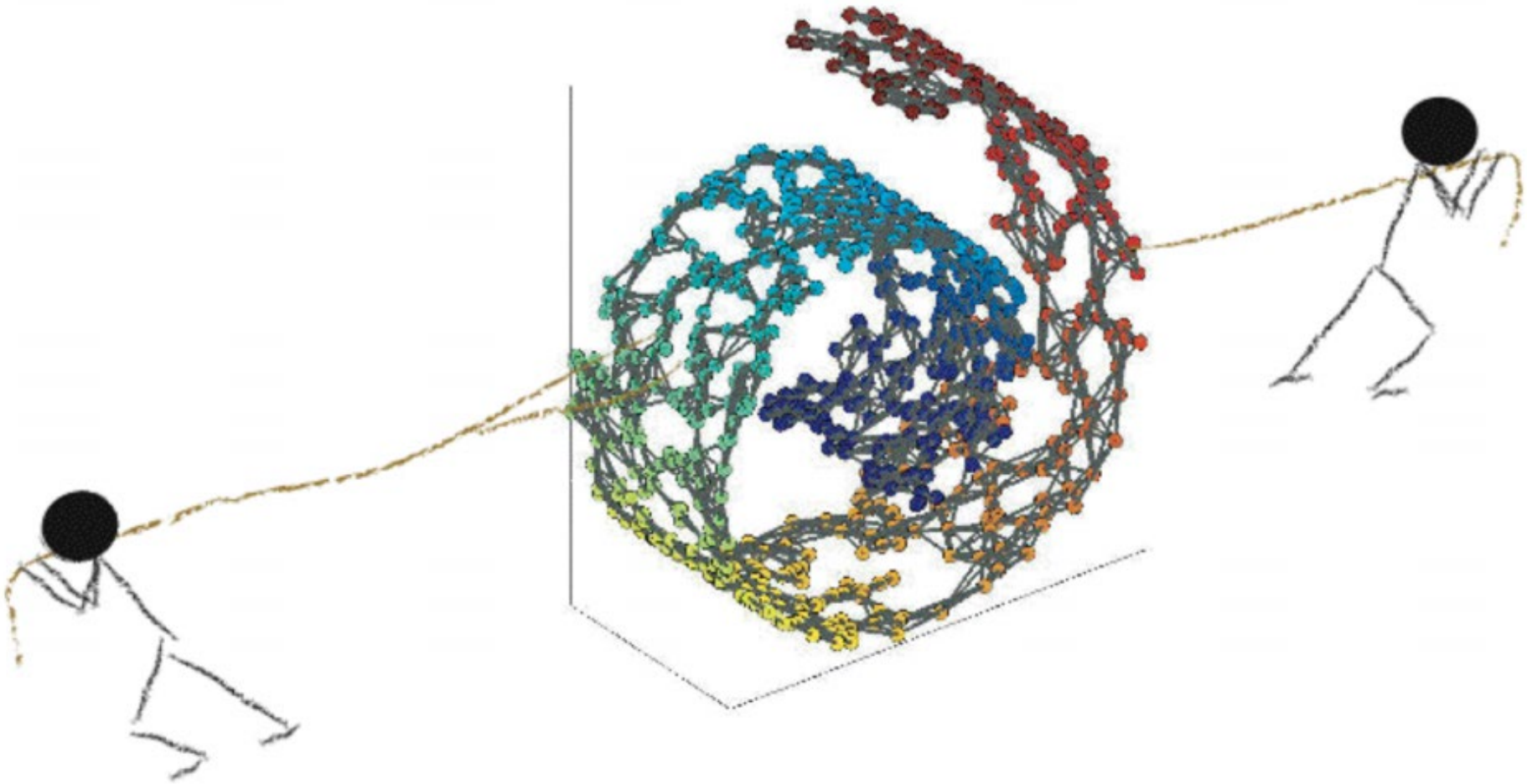
Manifold learning



Manifold learning



Manifold learning



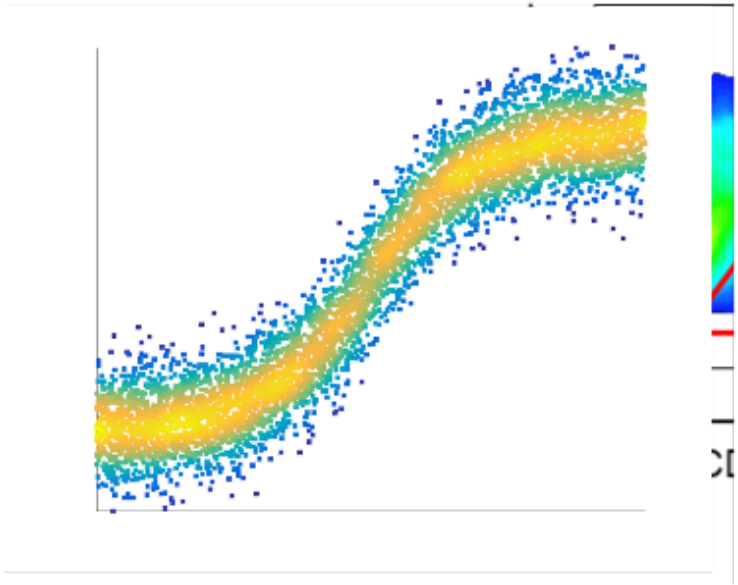
Manifold Learning for Data Visualization

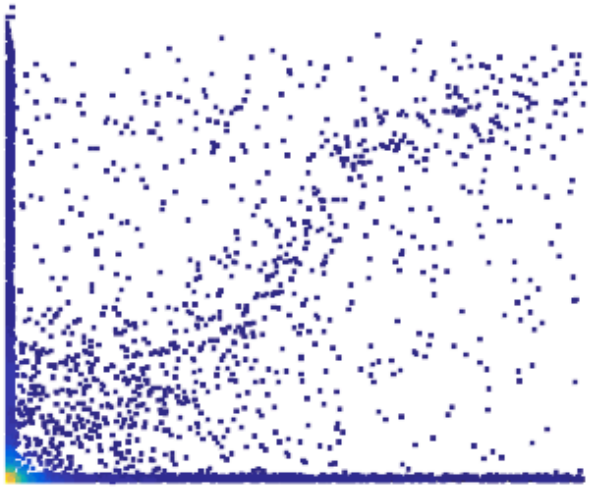
Data Visualization

- Humans are very visual
- Data visualization is a necessary tool for exploration
  - Develop intuitive understanding of the structure
  - Generate hypotheses

Challenges in Visualizing (Biomedical) Data

- High dimensions
- Noise/artifacts
- Nonlinearities
  - Most biological processes are NOT linear





Visualization with PCA

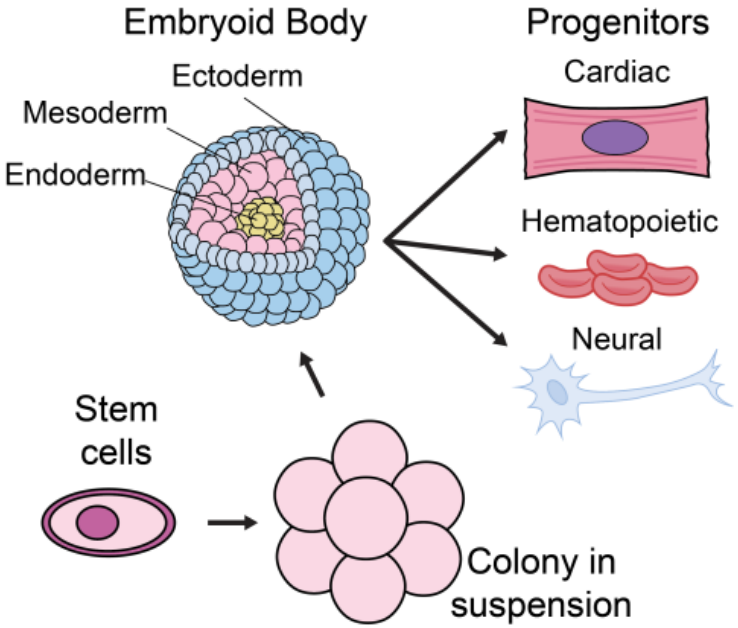
Artificial Tree Data

PCA

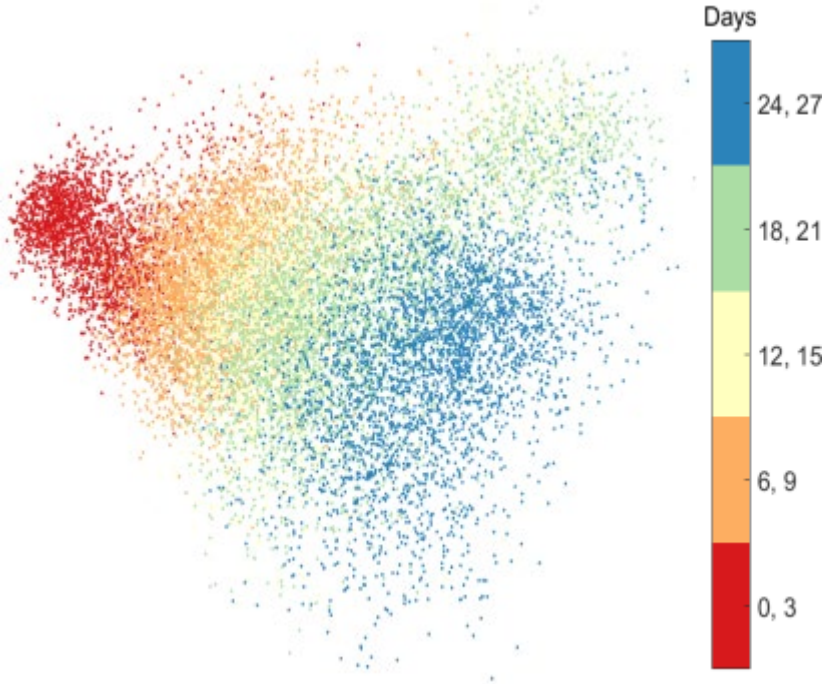
1400 points, 60 dimensions

Visualization with PCA

Newly generated scRNA-seq data (27 days)



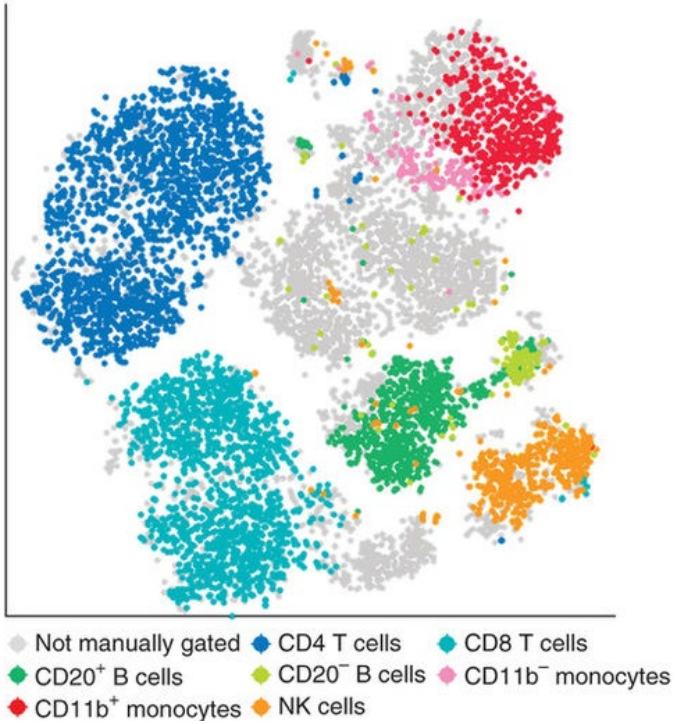
PCA



≈31k cells, more than 17k genes

t-SNE (van der Maaten & Hinton, *JMLR*, 2008)

- t-distributed stochastic neighbor embedding (t-SNE)
- Widely used on single cell data
  - Biology version (Amir et al, *Nature Biotech*, 2013) has 1400+ citations
- Attempts to preserve local relationships in both the high and low-dimensional spaces
  - Designed for separating clusters



(Amir et al, 2013)

t-SNE

Basic idea: neighbors in the high-dimensional space should be neighbors in low-dimensions

- Compute affinities/probabilities from distances:

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2 / 2\sigma_i^2)}$$

- Symmetrize:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

- Probabilities in low dimensions:

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

t-SNE

- Compute affinities/probabilities from distances:

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$$

- Symmetrize:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

- Probabilities in low dimensions:

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

- Minimize KL divergence between them:

$$\mathrm{KL}\left(P \parallel Q\right) = \sum_{i\neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Visualization with t-SNE

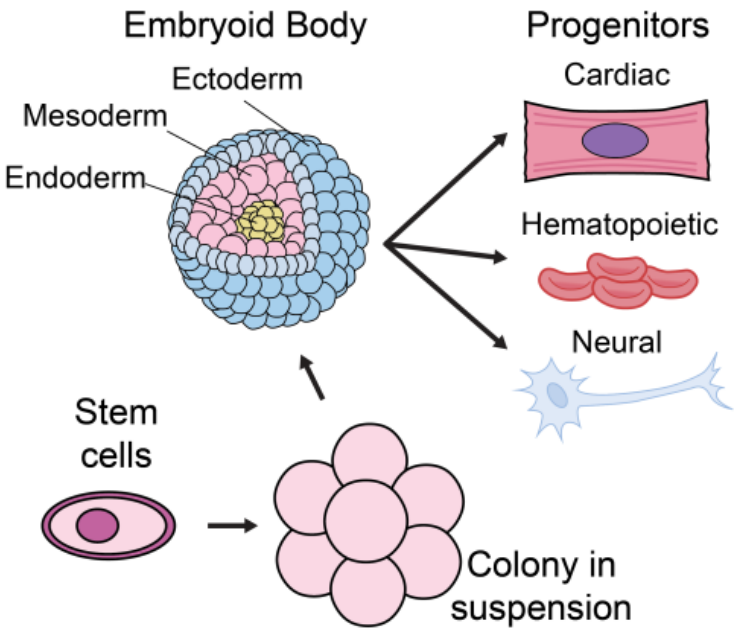
Artificial Tree Data

t-SNE

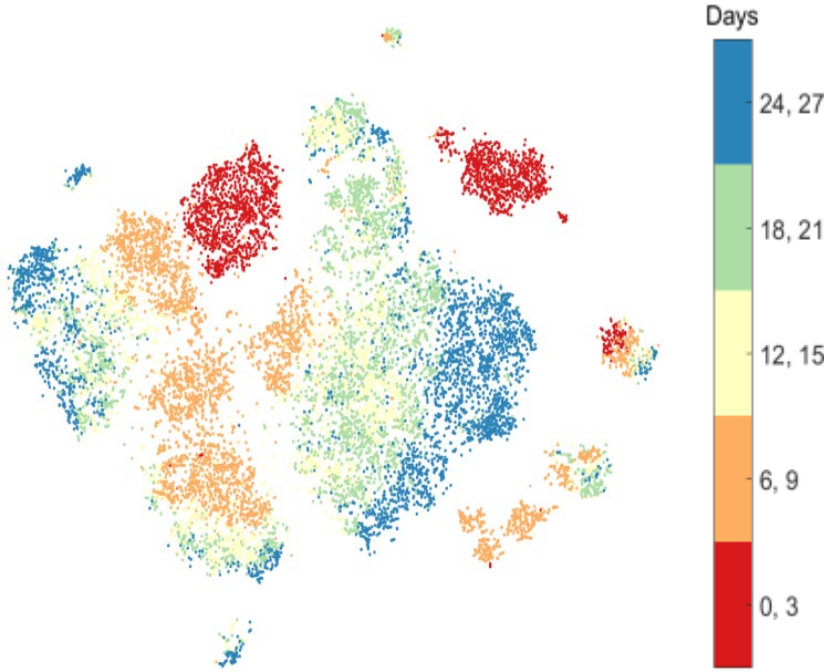
1400 points, 60 dimensions

Visualization with t-SNE

Newly generated scRNA-seq data (27 days)



t-SNE



≈31k cells, more than 17k genes

Issues with t-SNE

- Sensitive to hyperparameters
- Cluster sizes in a t-SNE plot are meaningless
  - Distance adapts to density
- Distances between clusters are generally meaningless
  - Global relationships are not preserved
- Random noise may not look random
- See https://distill.pub/2016/misread-tsne/ for more details

UMAP

- UMAP was proposed in 2018 to counter t-SNE's issues with preserving global structure
- It's also faster than t-SNE
- UMAP does do better at global structure, but not for the reasons the original authors claimed

UMAP vs. t-SNE

T-SNE

- Uses “perplexity” to determine similarity scale
- Normalizes similarities
- Averages similarities to symmetrize
- Uses KL-divergence as a loss function
- Initializes randomly
- Considers most pairwise similarities

UMAP

- Uses k-nn distances to determine similarity scale
- Doesn’t normalize
- Symmetrizes similarities differently
- Uses cross entropy as a loss function
- Initializes with Laplacian Eigenmaps
- Uses noise contrastive estimation (NCE)

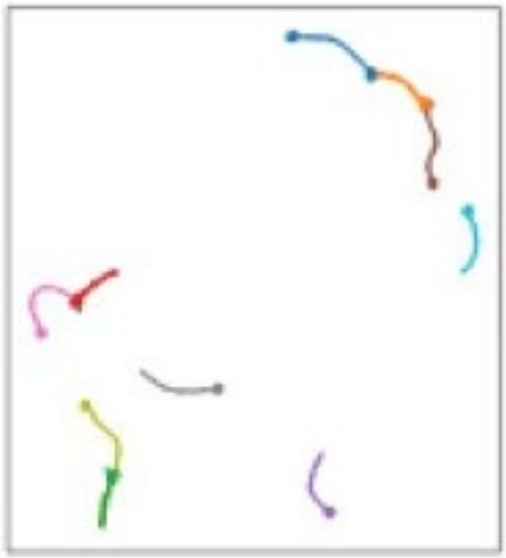
Importance of Initialization

- The main reason UMAP does better at global relationships than t-SNE is the Laplacian eigenmaps initialization
- Initializing t-SNE with PCA gives near identical results (Kobak and Linderman, 2021)

UMAP still fails at global structure

Artificial Tree Data

UMAP



1400 points, 60 dimensions

Why is UMAP faster?

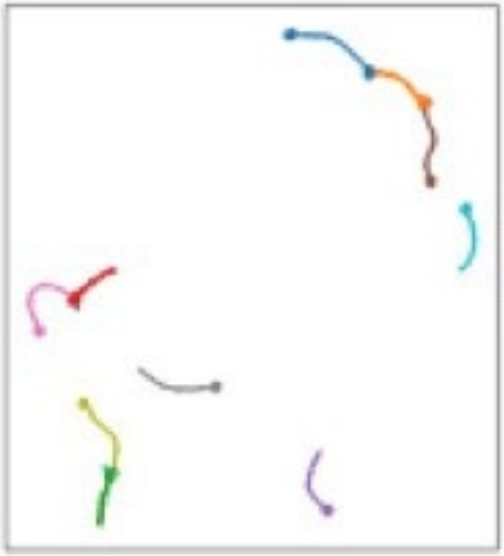
- Loss function is faster to compute and differentiate (no similarity normalizations)
- Biggest gain is in NCE
- NCE compares points to only a few real points as well as a bunch of “fake” points
  - Allows for faster computations
  - Used in a lot of neural network settings to speed up training as part of “self-supervised learning”
  - However, it changes the effective loss function (Damrich and Hamprecht, 2021)

Balancing local and global information

- T-SNE and UMAP are good for local structure
- PCA is good for global structure
- Diffusion maps (DM) does well at capturing both
- **Idea**: use DM to visualize data

t-SNE

UMAP



PCA

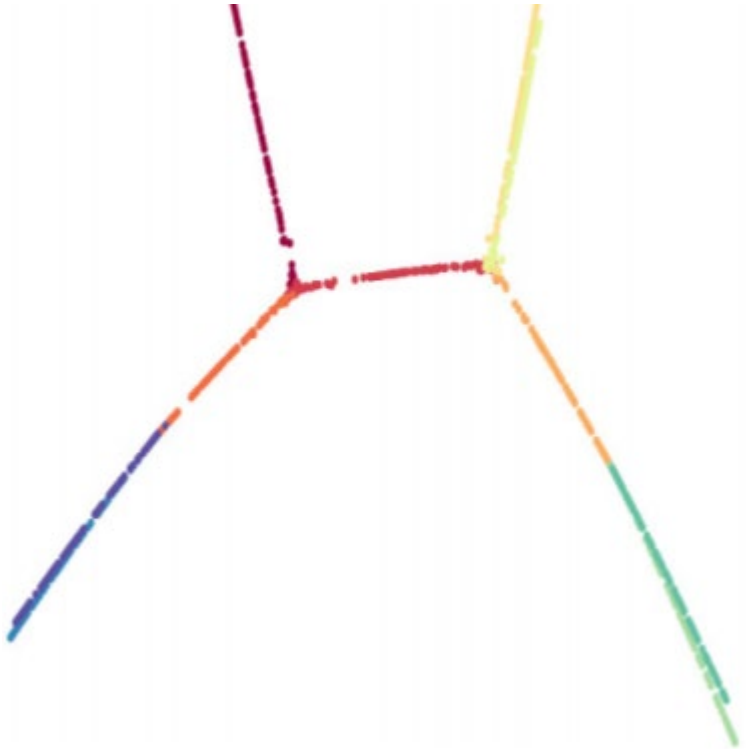
Visualization with Diffusion Maps

- DM captures the information accurately and robustly, but not for visualization
- DM tends to place the structure in higher dimensions
- DM can also be unstable with boundaries and intersections

Artificial Tree Data

1400 points, 60 dimensions

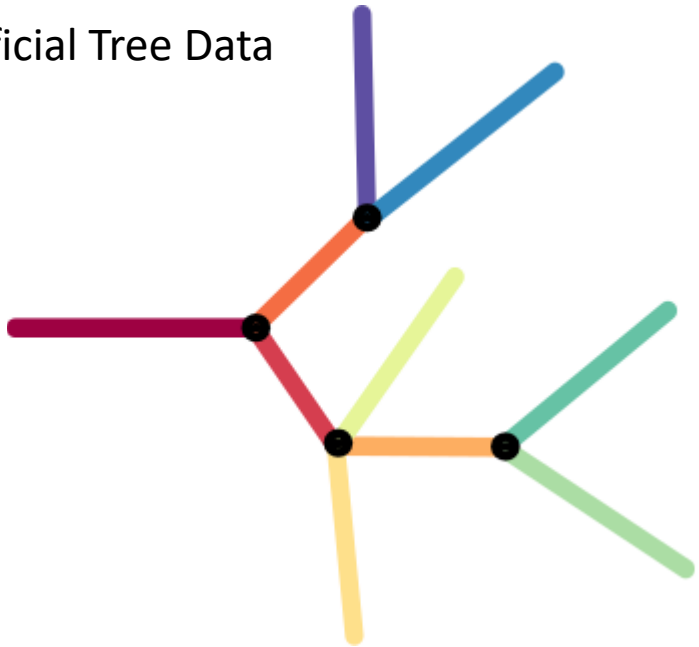
Diffusion Maps



Visualizing Structure and Transitions in High-Dimensional Biological Data (Nature Biotechnology, 2019)

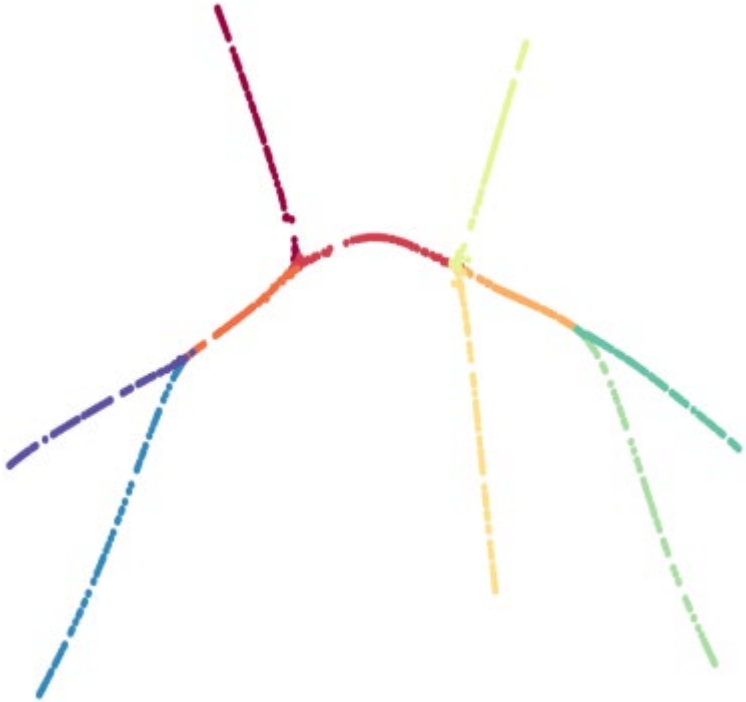
- A method for visualizing data built off of the diffusion framework

Artificial Tree Data



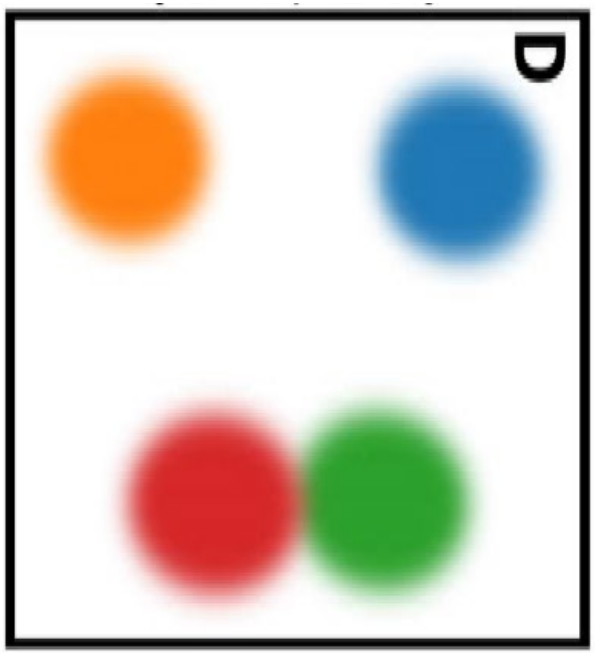
1400 points, 60 dimensions

PHATE

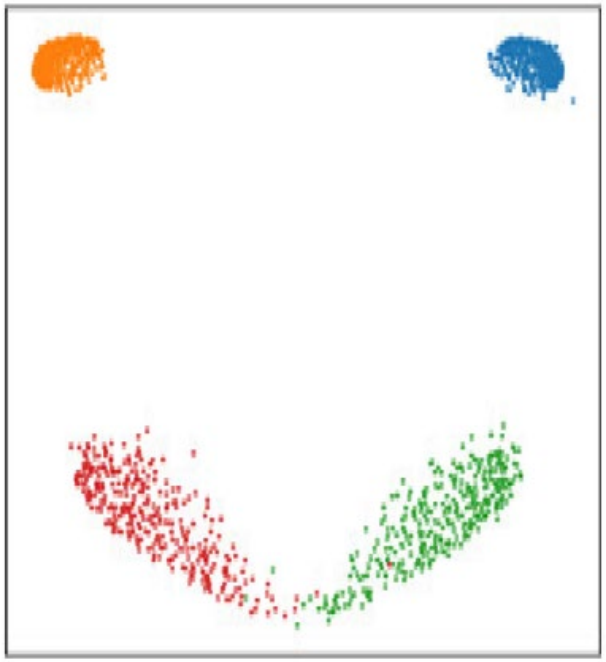


PHATE on GMM

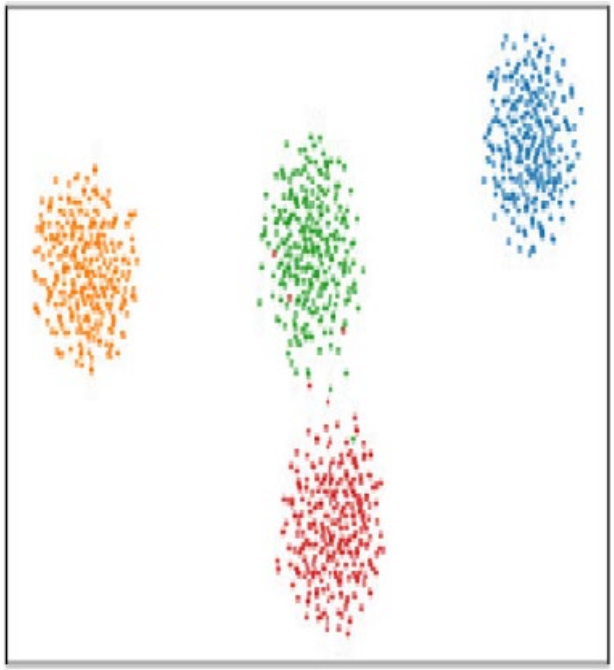
Truth



PHATE

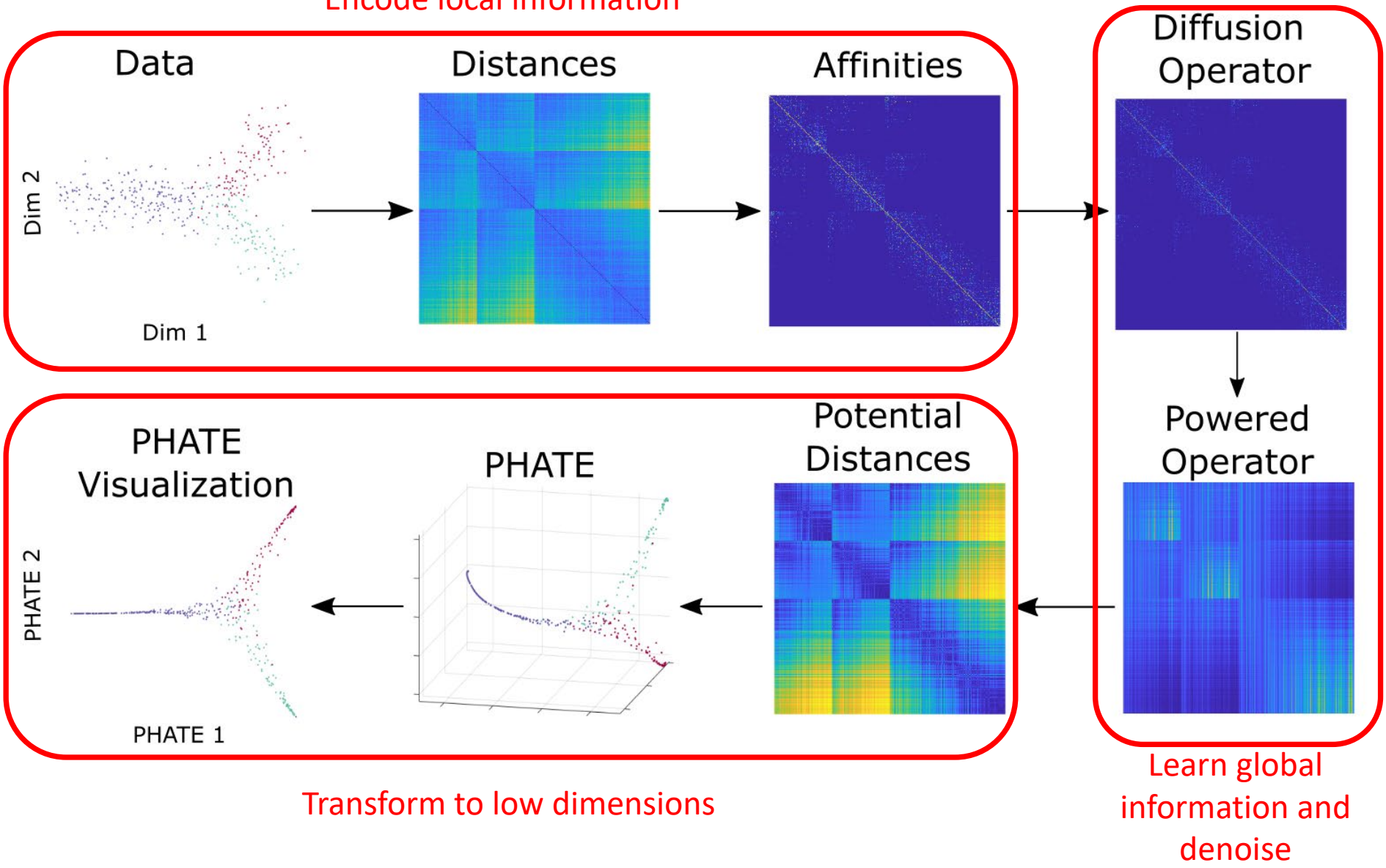


t-SNE



The PHATE Algorithm

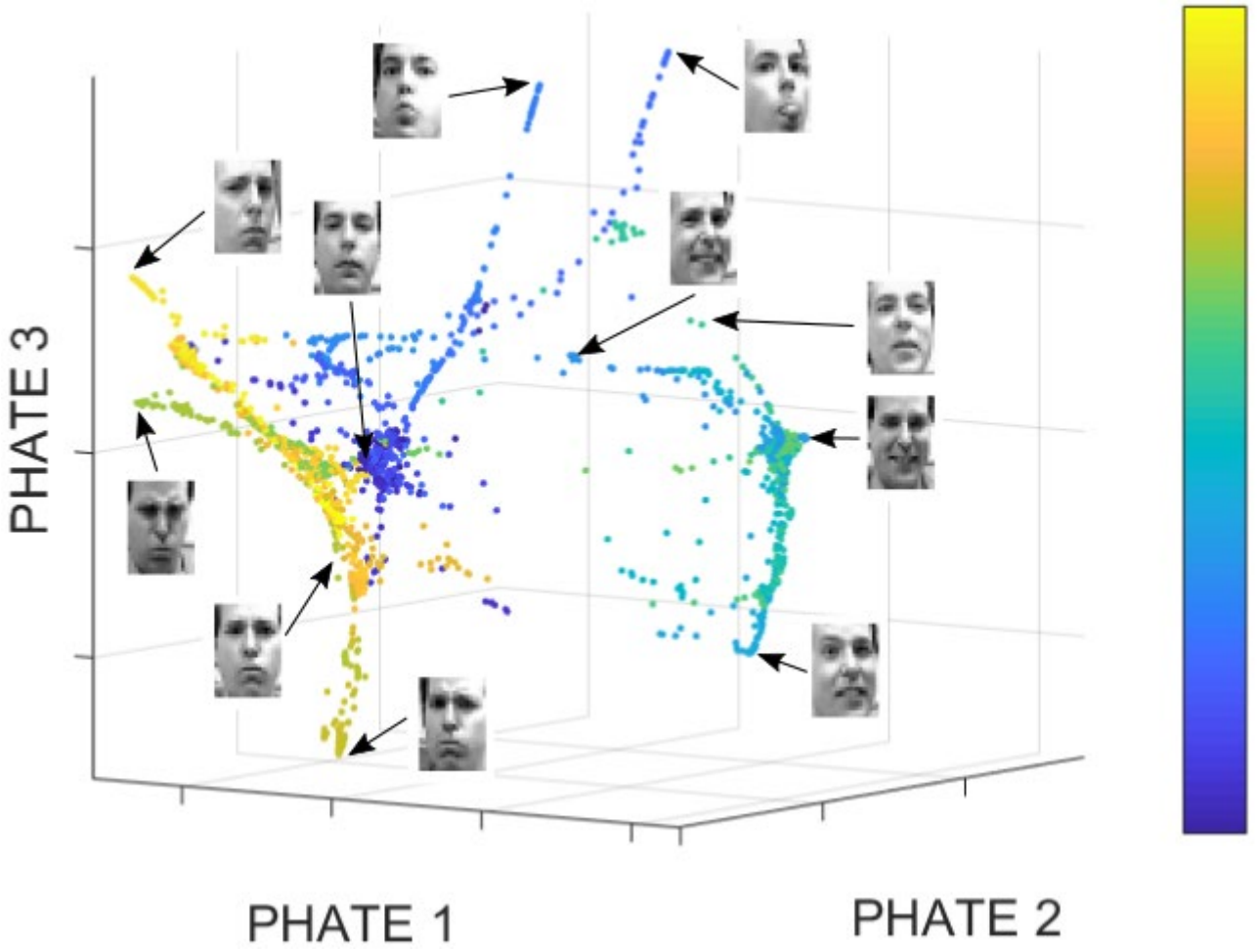
Encode local information



PHATE on Frey Faces

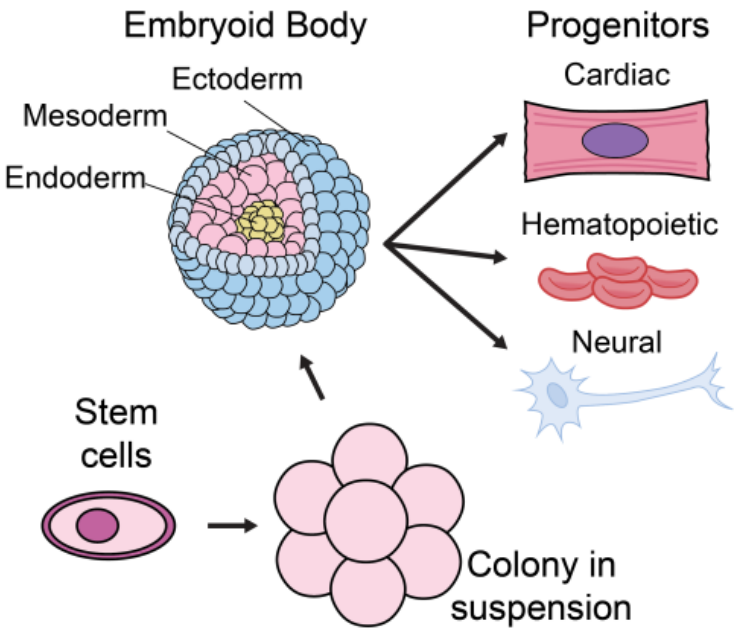
- Frey Faces dataset
  - 1965 frames taken from a video of Brendan Frey making various faces in front of a camera
  - Resolution: 20x28
- Frames are given out of order to PHATE w/o information about sequential ordering

PHATE on Frey Faces



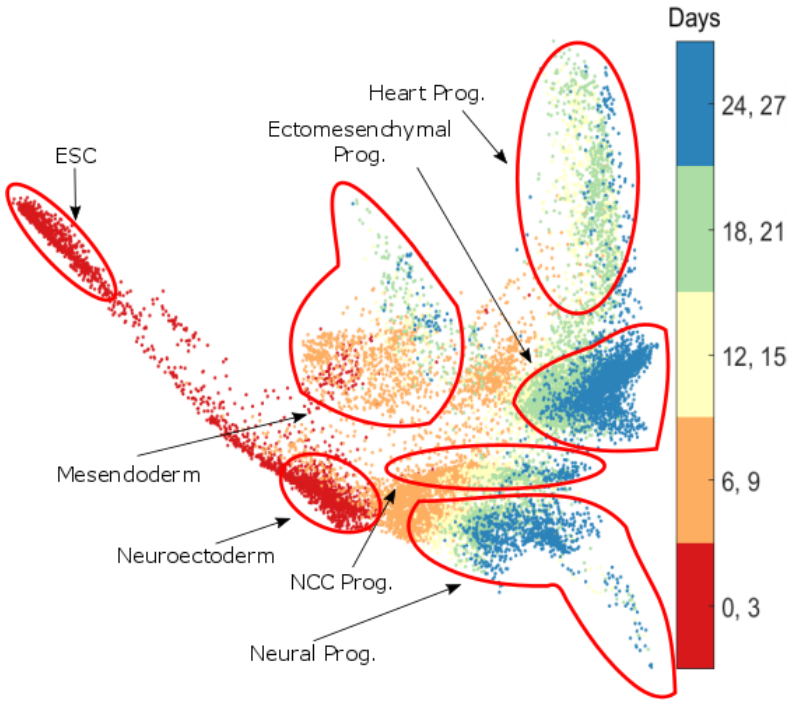
PHATE on EB scRNA-seq Data

Newly generated scRNA-seq data (27 days)



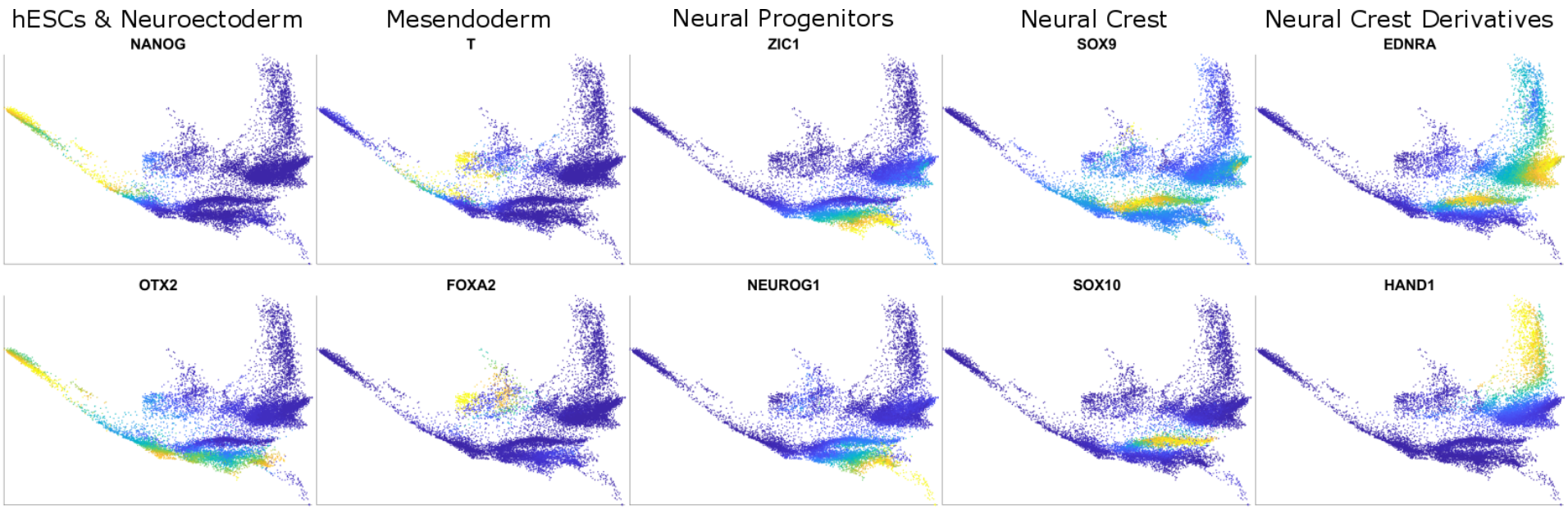
≈31k cells, more than 17k genes

PHATE



Exploratory Data Analysis with PHATE

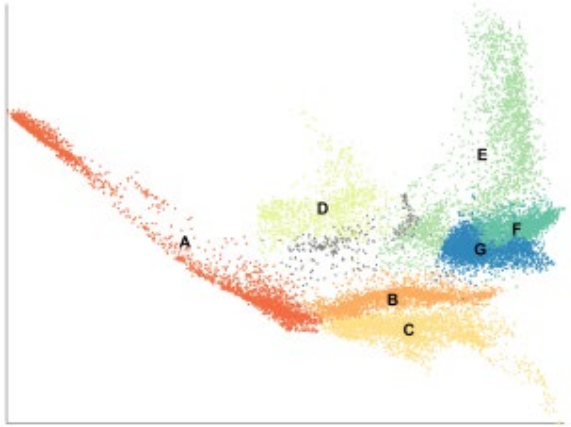
- Coloring the embedding by gene expression after MAGIC* (van Dijk,…Moon et al., 2018) reveals lineages



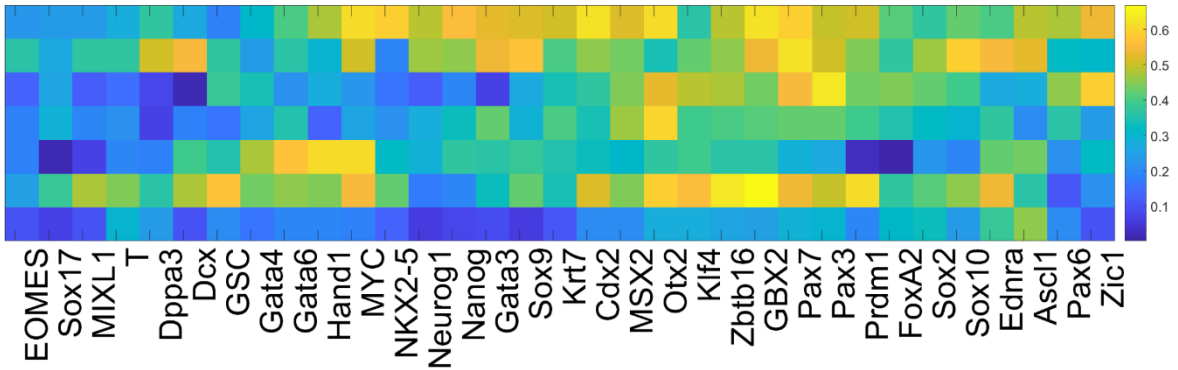
- Interactive web tool: krishnaswamylab.org/phatewebtool

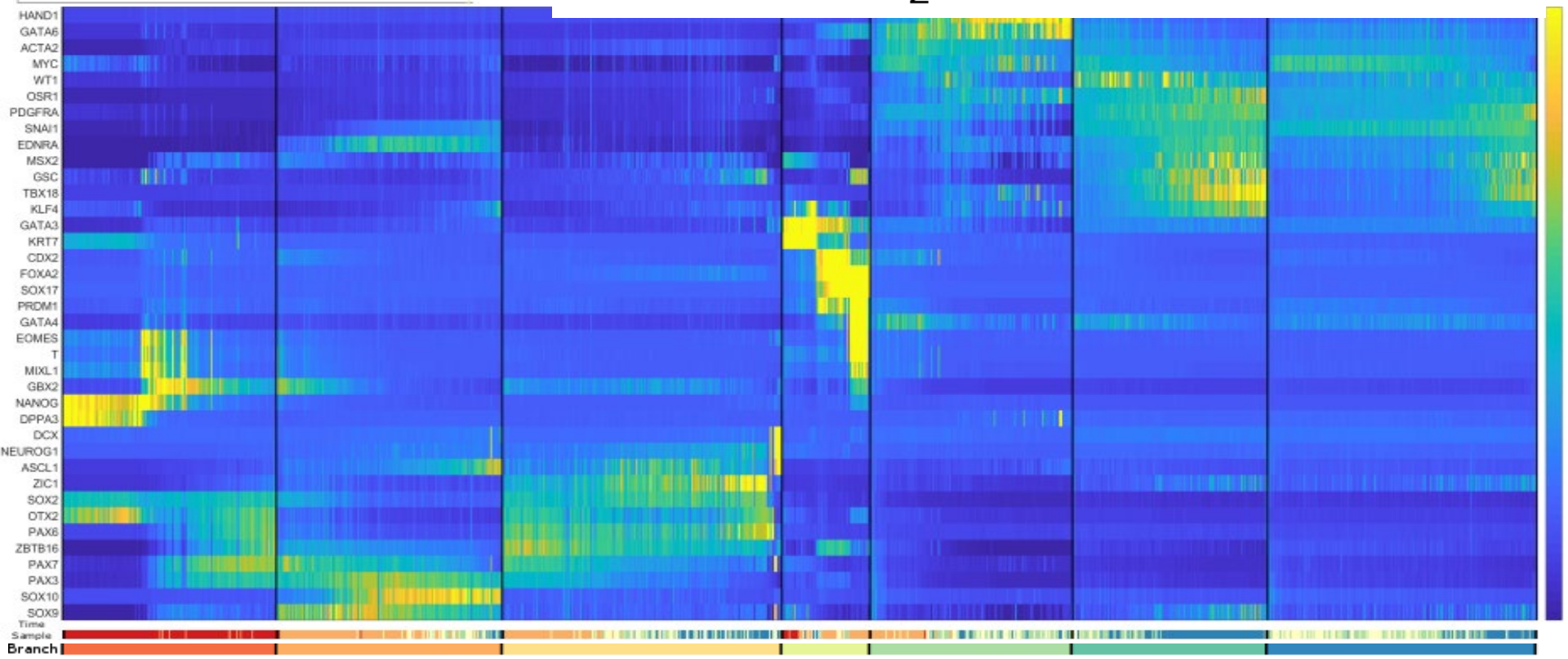
Published in *Cell

Exploratory Data Analysis with PHATE

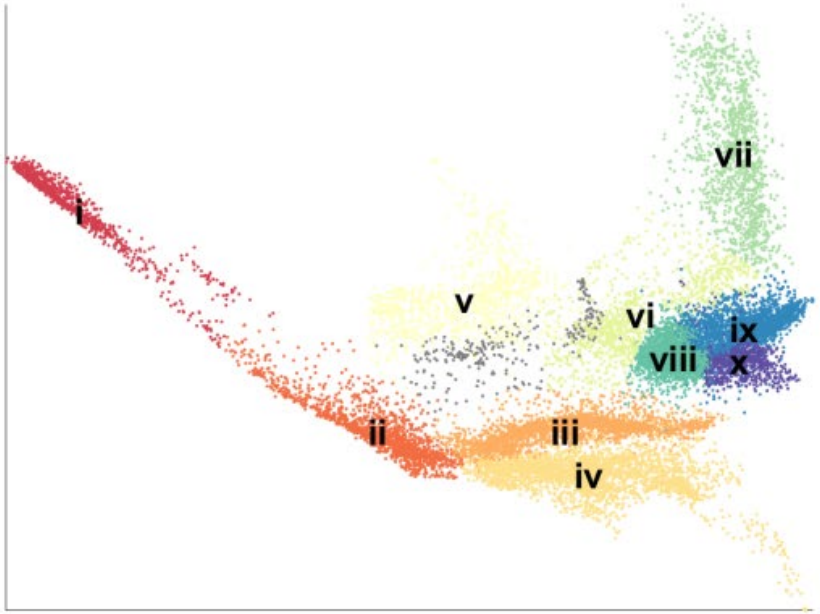


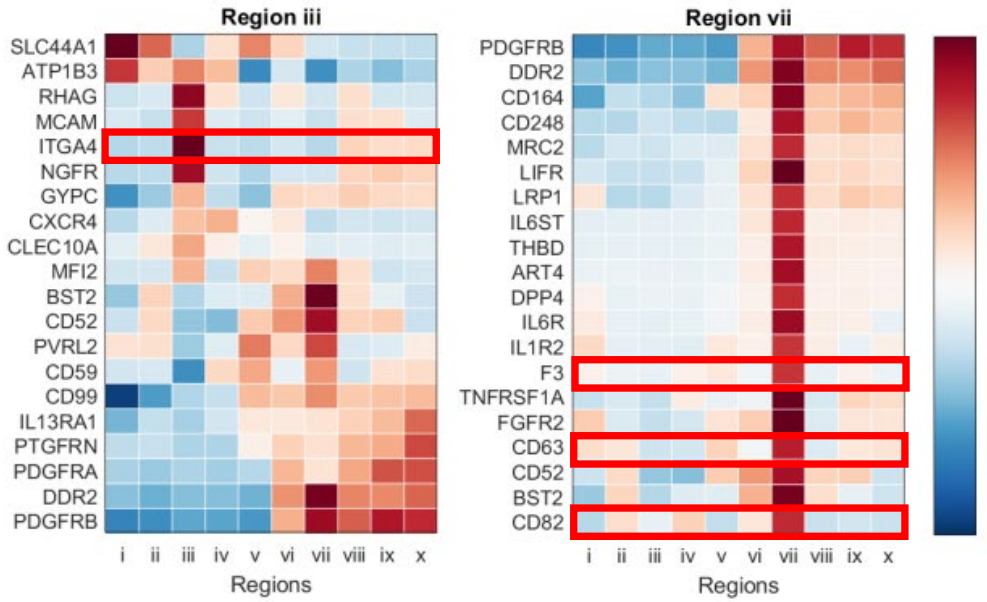
DREMI Score (Krishnaswamy et al., 2014)





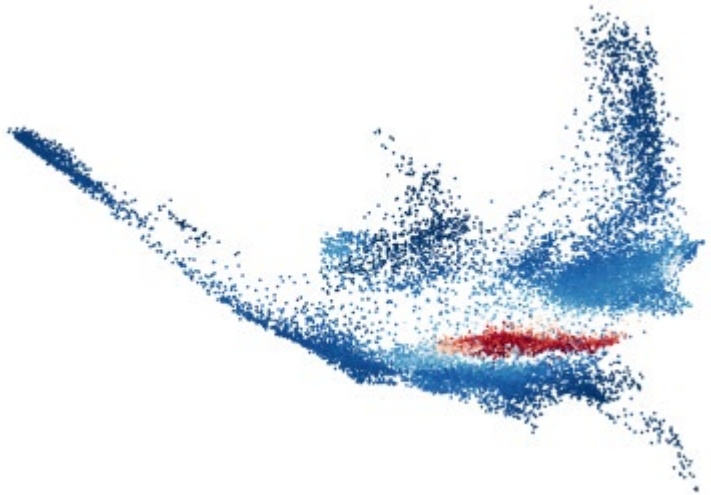
Discovering New Surface Markers for Sorting Populations

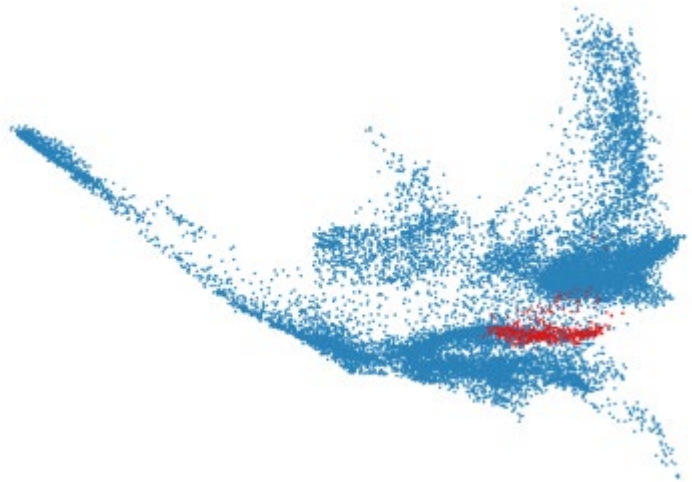




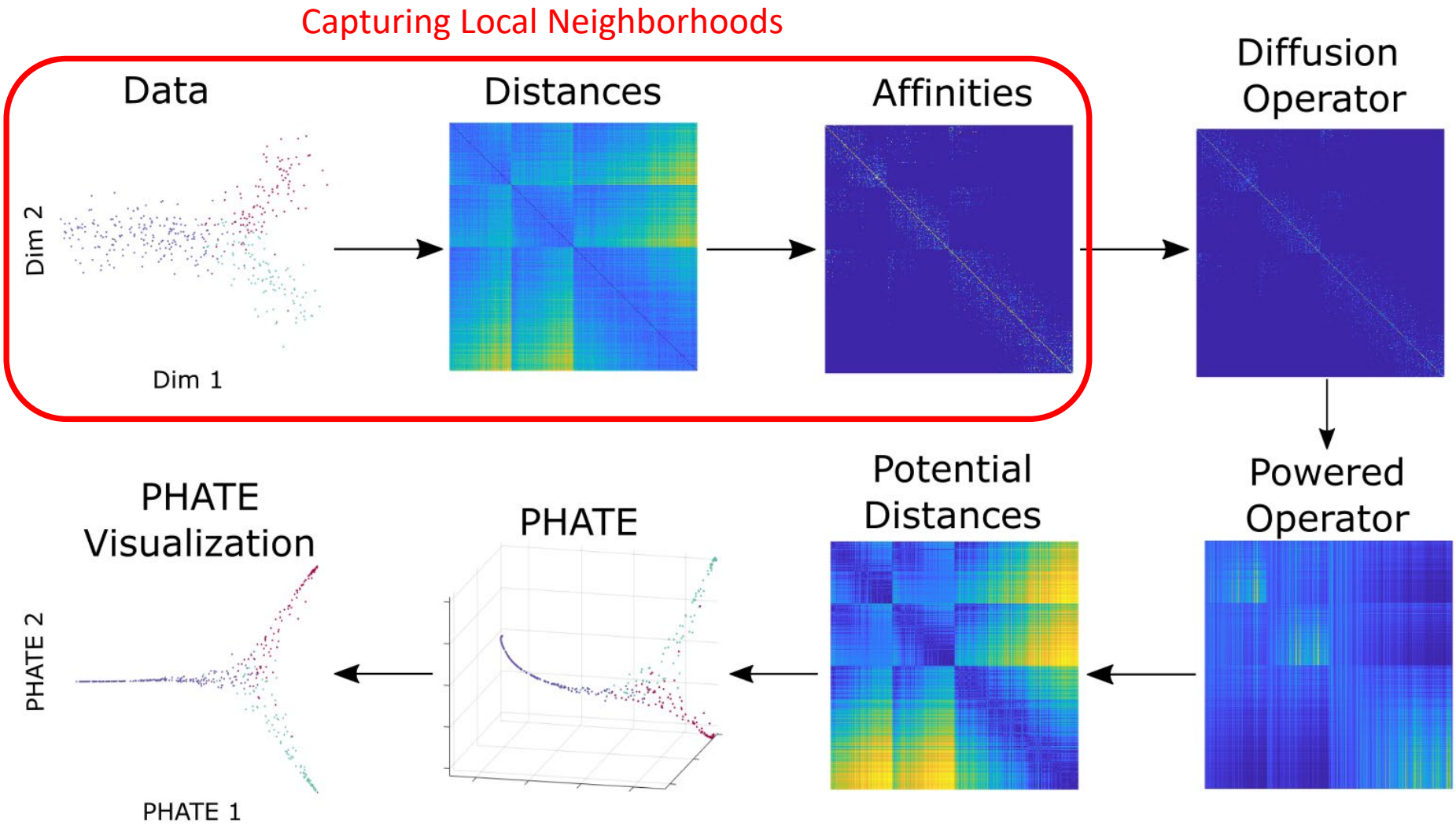
ITGA4

ITGA4 sorted bulk RNA-seq correlation

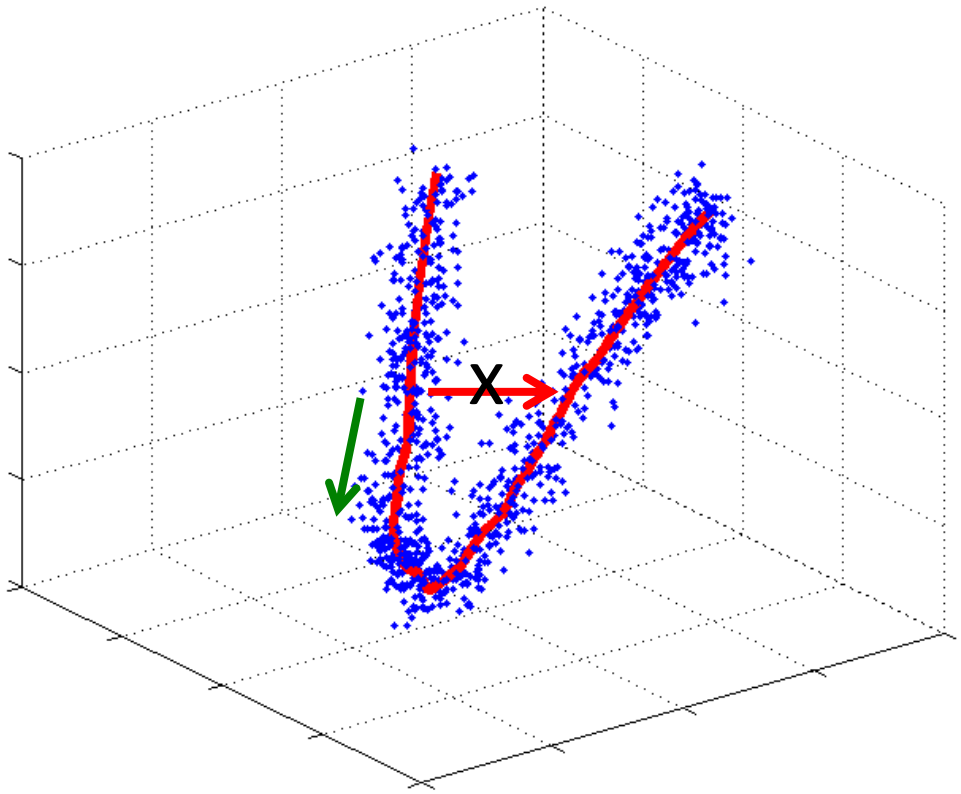




The PHATE Algorithm



Capturing local neighborhoods



- Large Euclidean distance gives wrong global structure for visualizing
- Small Euclidean distances ok
  - Encodes local structure

From distances to affinities

- Step 1: Calculate all pairwise Euclidean distances
- Step 2: Convert the distances to pairwise affinities using a kernel function
  - E.g. the Gaussian kernel
  - Kernel function must be near zero for large distances and nonzero for small distances
  - Affinity: a measure of similarity between points
- **Problem**: many data have both dense and sparse regions
- A kernel bandwidth fixed for dense regions doesn't work well for sparse regions and vice versa
- Idea: use locally adaptive bandwidth

Capturing Local Neighborhoods

- **Adaptive α-decaying kernel**

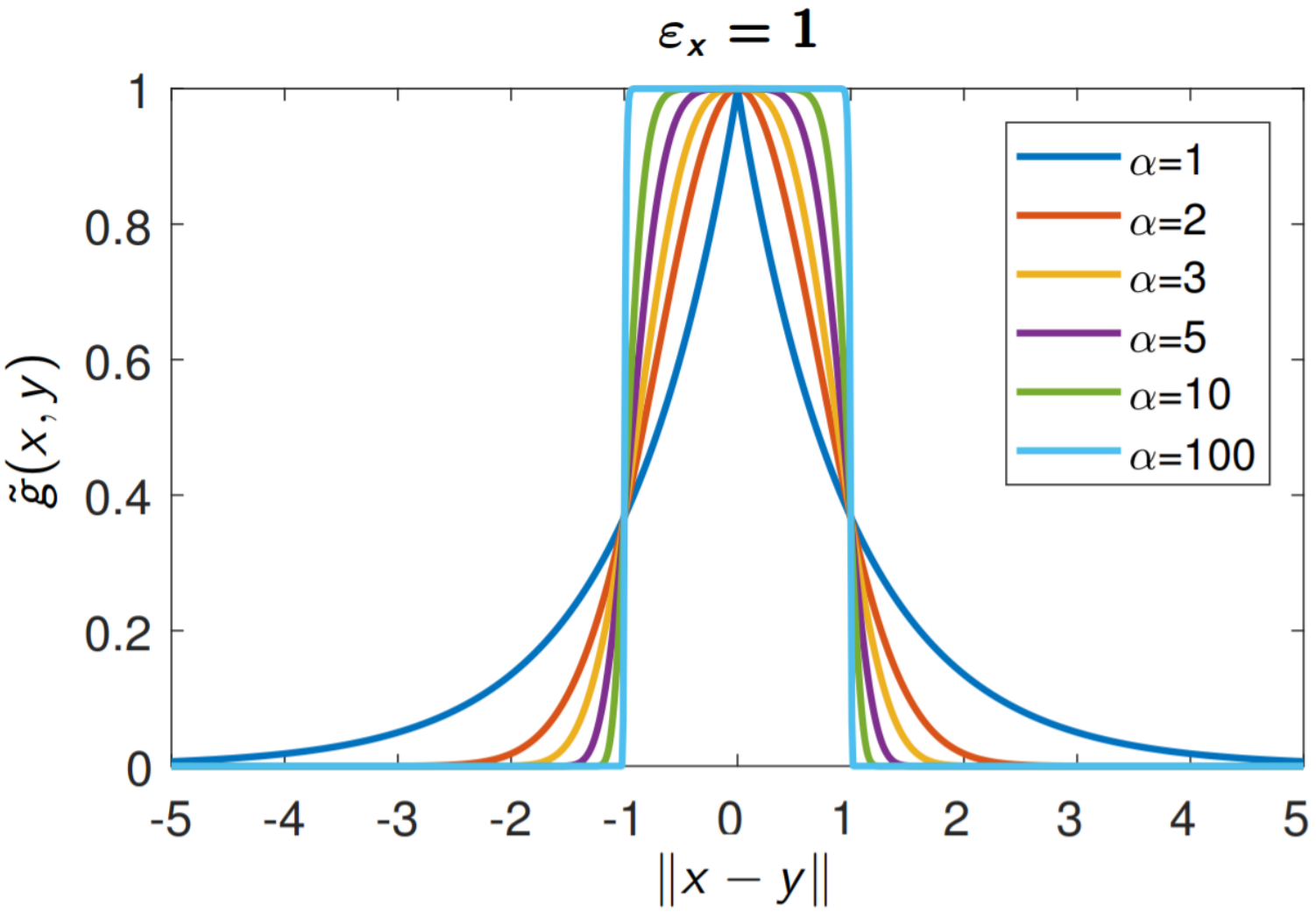
$$\tilde{g}(x,y) = \exp\left(-\left(\frac{||x-y||}{\epsilon_x}\right)^{\alpha}\right), \rightarrow g(x,y) = \frac{\tilde{g}(x,y)+\tilde{g}(y,x)}{2}$$

Where

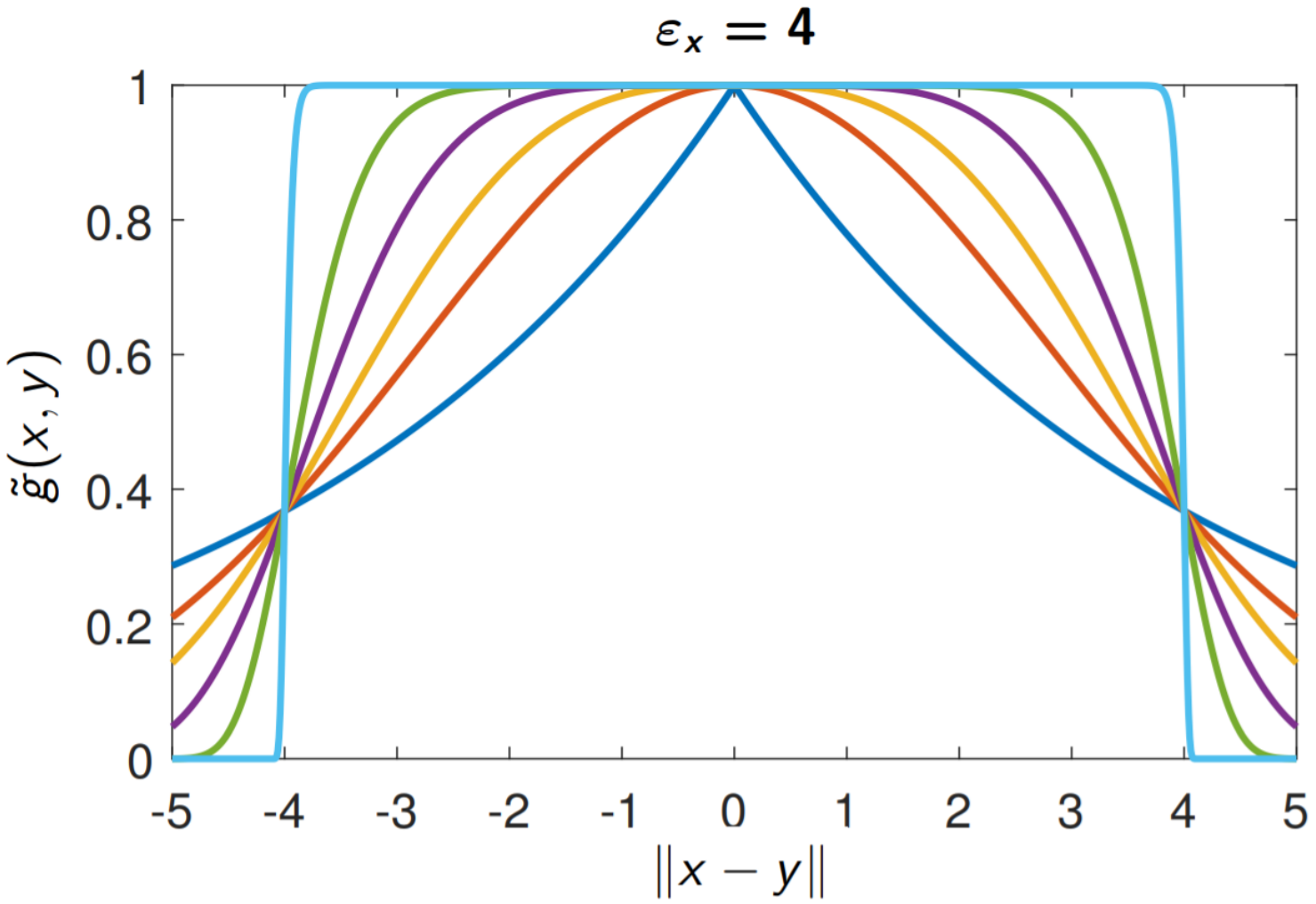
- ϵ_x = distance from x to its kth nearest neighbor
- α controls the decay rate of $\tilde{g}$

- Provides a robust notion of adaptive locality

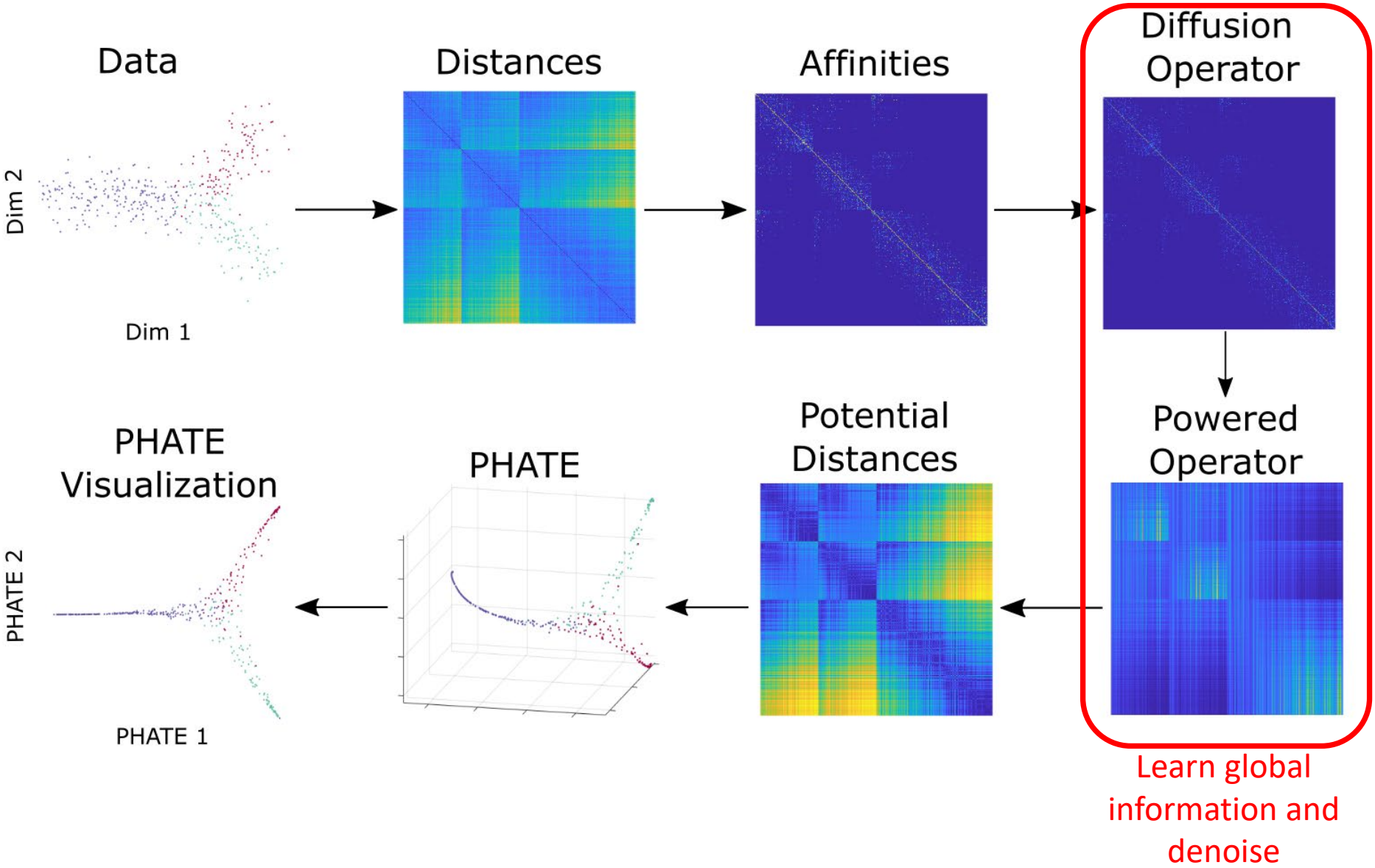
Capturing Local Neighborhoods



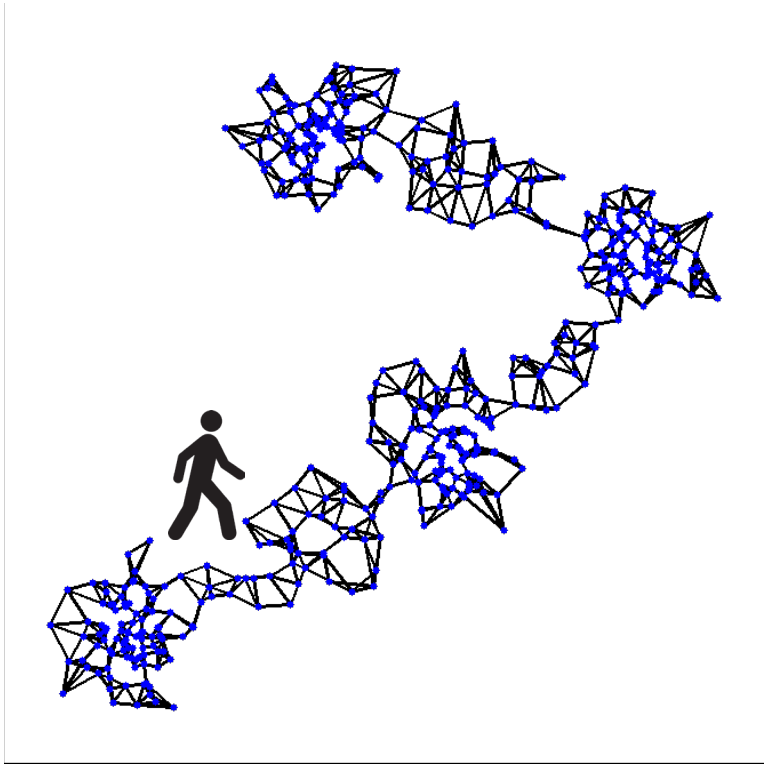
Capturing Local Neighborhoods



The PHATE Algorithm



Diffusion Denoises and Recovers Global Structure



- Big Euclidean steps bad, likely to exit structure
- Small steps good, likely to stay within structure
- To learn global structure via small local steps we use random walk (diffusion)
  - Normalize affinity matrix to create Markov transition matrix (the diffusion operator) P (Coifman & Lafon, *ACHA*, 2006)
  - Power P by time step t

How much diffusion?

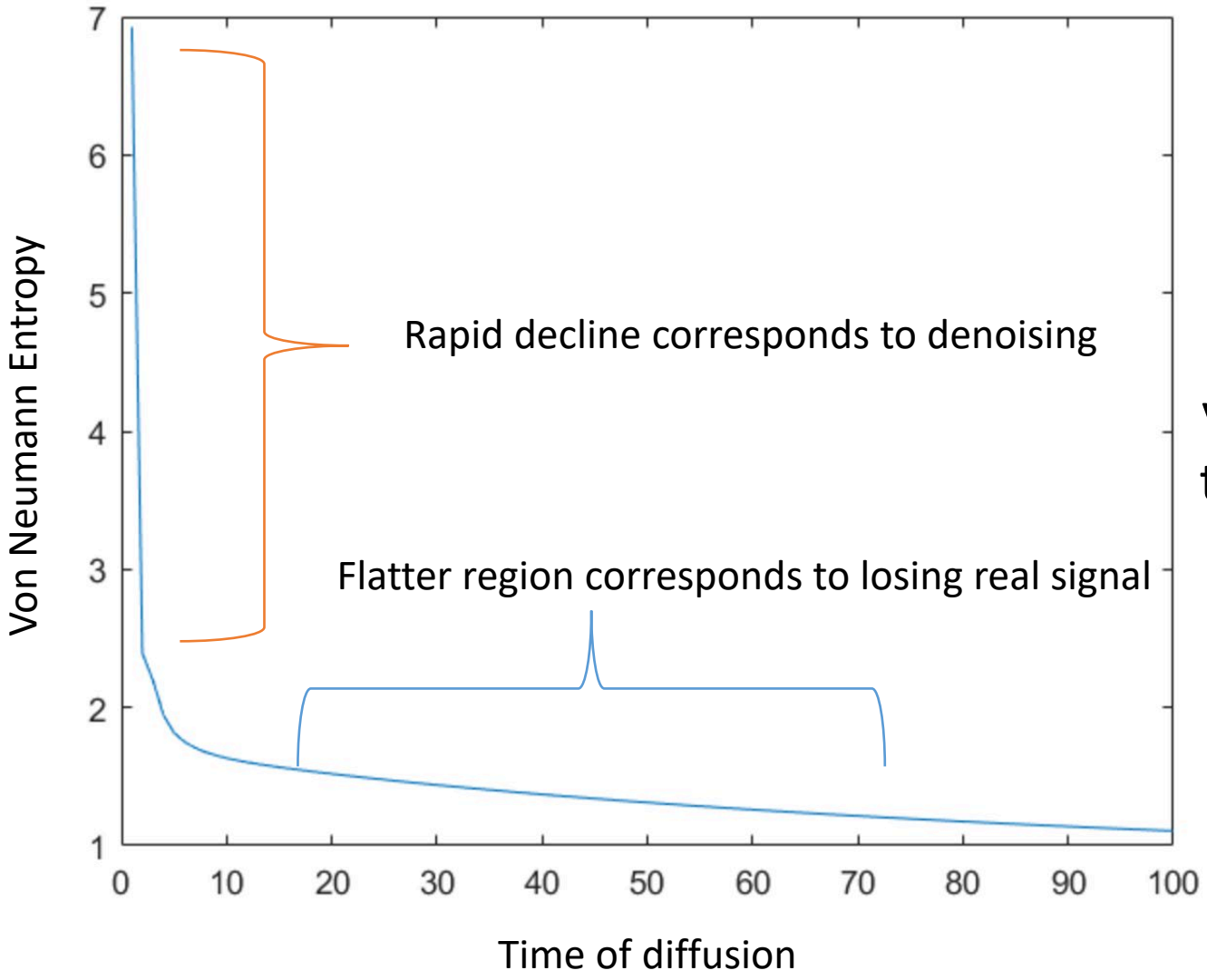
- Von Neumann entropy (von Neumann, 1932) of diffused operator P^t

$$VNE(P^t) = -\sum_j \eta_j \log \eta_j \,, \eta_j = \lambda_j^t / \left|\left|\boldsymbol{\lambda}^t\right|\right|_1$$

Where $\boldsymbol{\lambda}^t = \{\lambda_0^t, \lambda_1^t, \dots\}$ are the eigenvalues of P^t

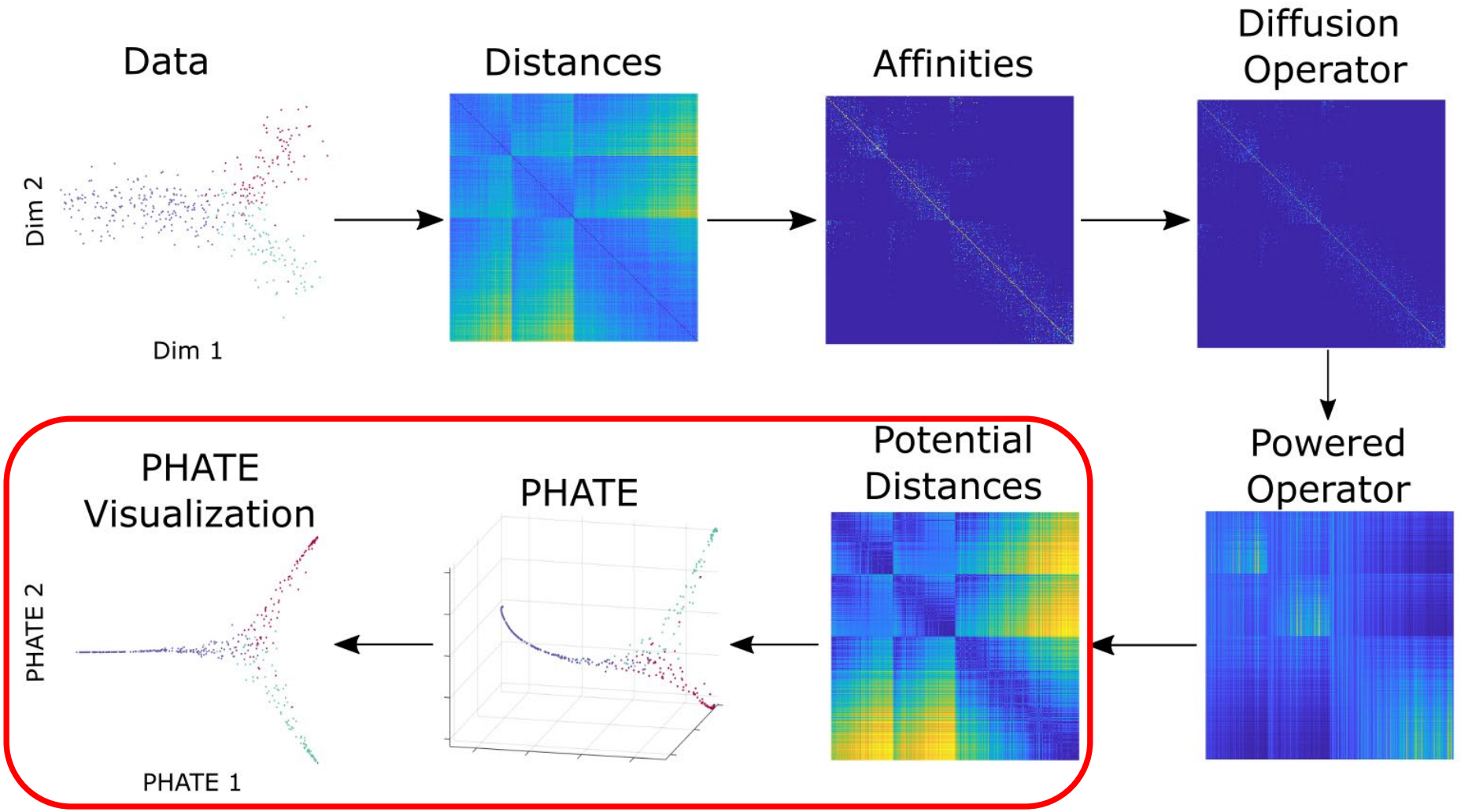
- VNE is a soft proxy of numerical rank
- Decays as diffusion time increases, $\lim_{t\to\infty} VNE(P^t) = 0$
- Use rate of decay to choose time scale

How much diffusion?



Embedding is visually robust in the flatter region

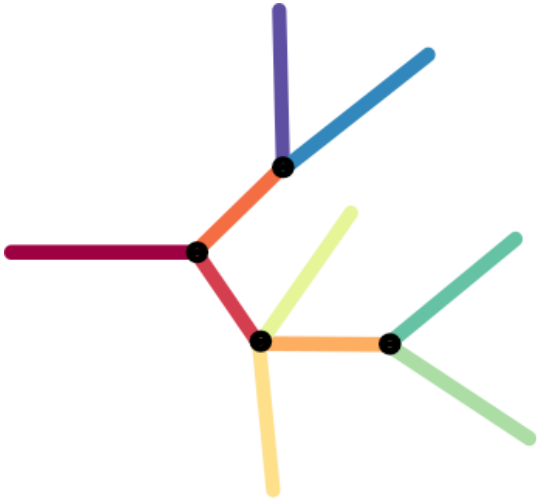
The PHATE Algorithm

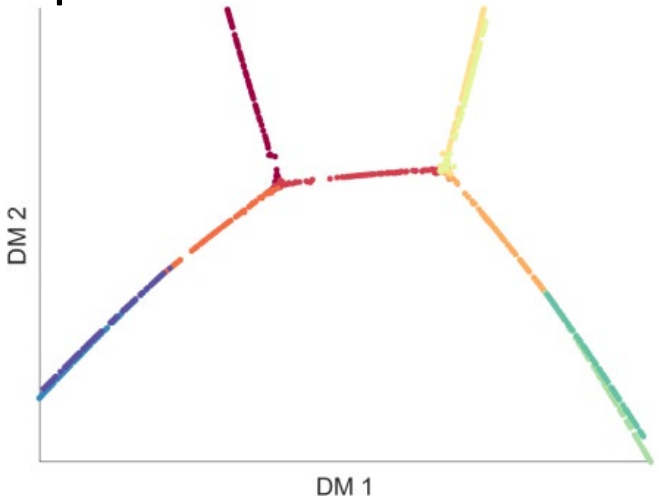


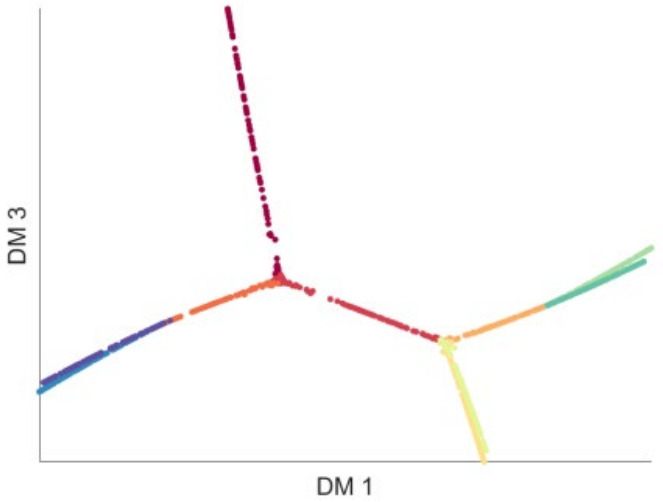
Transform to low dimensions

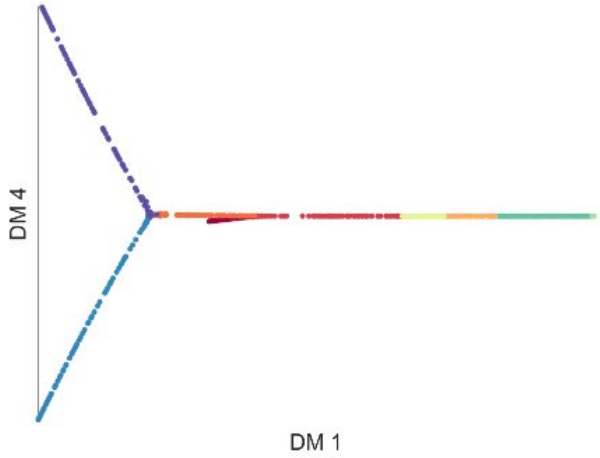
Potential Distances

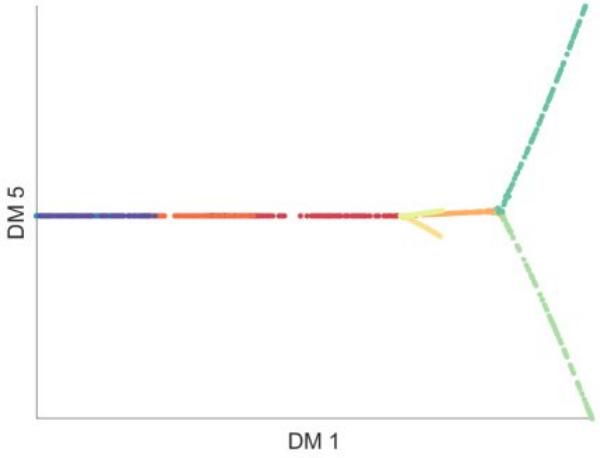
- Diffusion operator contains global and local structure
  - Encoded in multiple dimensions

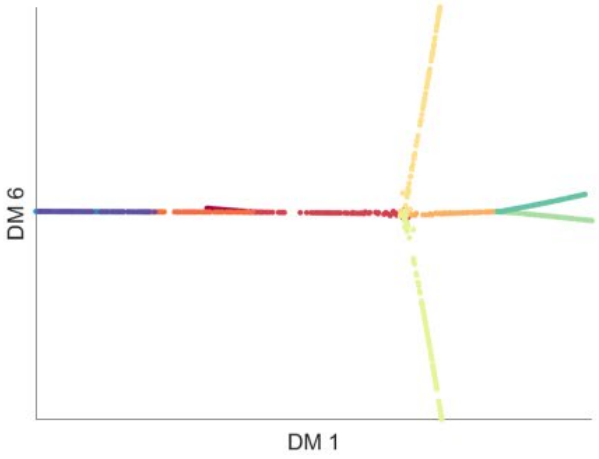






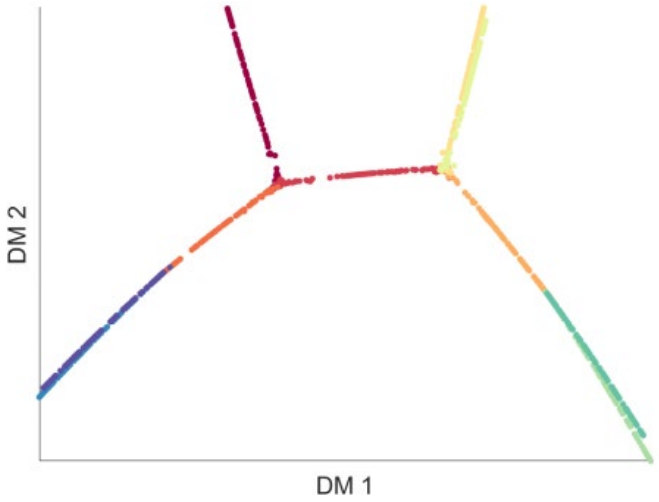


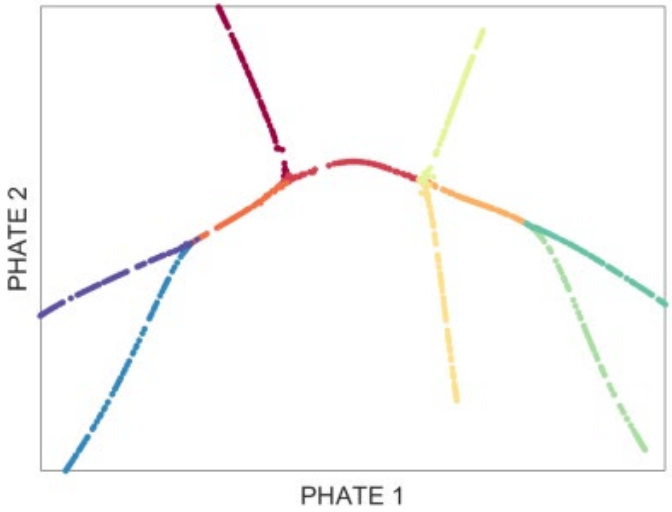




Potential Distances

- Diffusion operator contains global and local structure
  - Encoded in multiple dimensions
- To extract this information, we transform diffused probabilities using a ***potential transformation***
  - Forms an information distance between diffused probabilities
  - Connected to heat potential
- Embed for visualization using multidimensional scaling (MDS)





Full PHATE Algorithm

Input: Data matrix X, neighborhood size k, locality scale α, desired embedding dimension m (usually 2 or 3 for visualization)

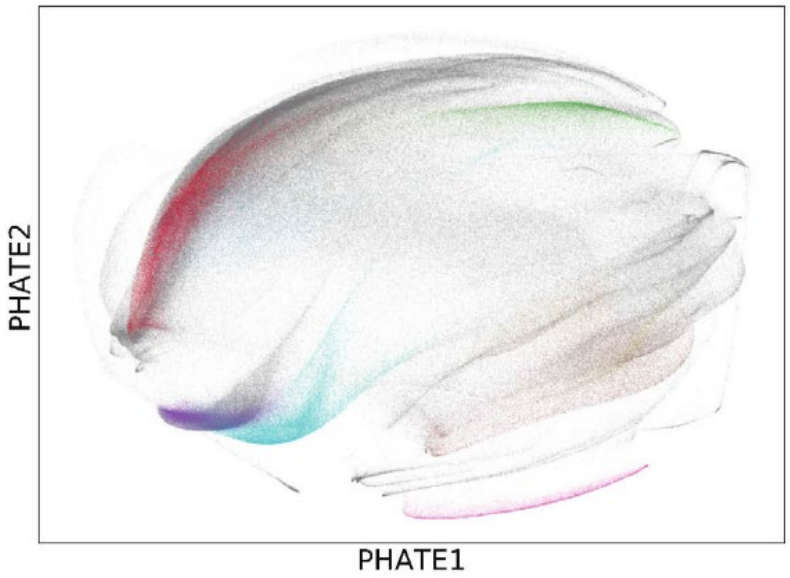
Output: The PHATE embedding Y_m

1: $D \leftarrow$ compute pairwise distance matrix from X
2: Compute the k-nearest neighbor distance $\varepsilon_k(x)$ for each column x of X
3: $K_{k,\alpha} \leftarrow$ compute local affinity matrix from D and ε_k (see Eq. 3)
4: $P \leftarrow$ normalize $K_{k,\alpha}$ to form a Markov transition matrix (diffusion operator; see Eq. 2)
5: $t \leftarrow$ compute time scale via Von Neumann Entropy (see Eq. 5)
6: Diffuse P for t time steps to obtain P^t
7: Compute potential representations: $U_t \leftarrow -\log(P^t)$
8: $\mathfrak{V}^t \leftarrow$ compute potential distance matrix from U_t (see Eq. 6)
9: $Y_{class} \leftarrow$ apply classical MDS to $\mathfrak{V}^t$
10: $Y_m \leftarrow$ apply metric MDS to $\mathfrak{V}^t$ with Y_{class} as an initialization

Supplemental Table S1: Detailed steps in the PHATE algorithm.

Scalability of PHATE

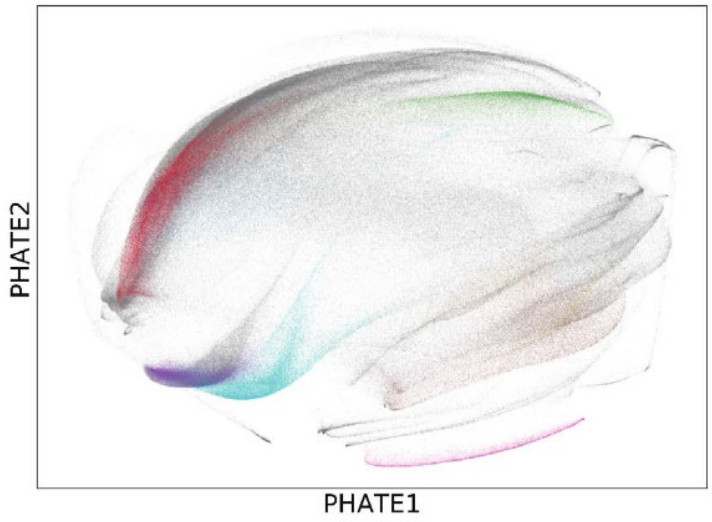
- Storing and performing operations on the diffusion matrix can be difficult for large samples
- Can reduce computation by diffusing through "landmarks" (compressed diffusion, Gigante et al., 2018)
  - Landmarks are chosen by clustering
  - Obtain an embedding of all points by projection

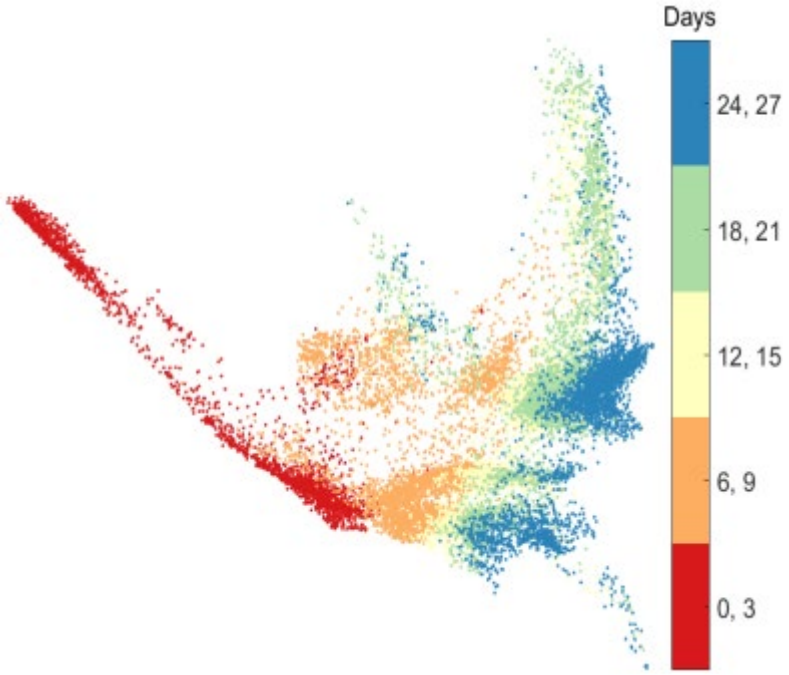


PHATE applied to the 10x megacell mouse brain data (>1 million cells)

Noisy Visualizations

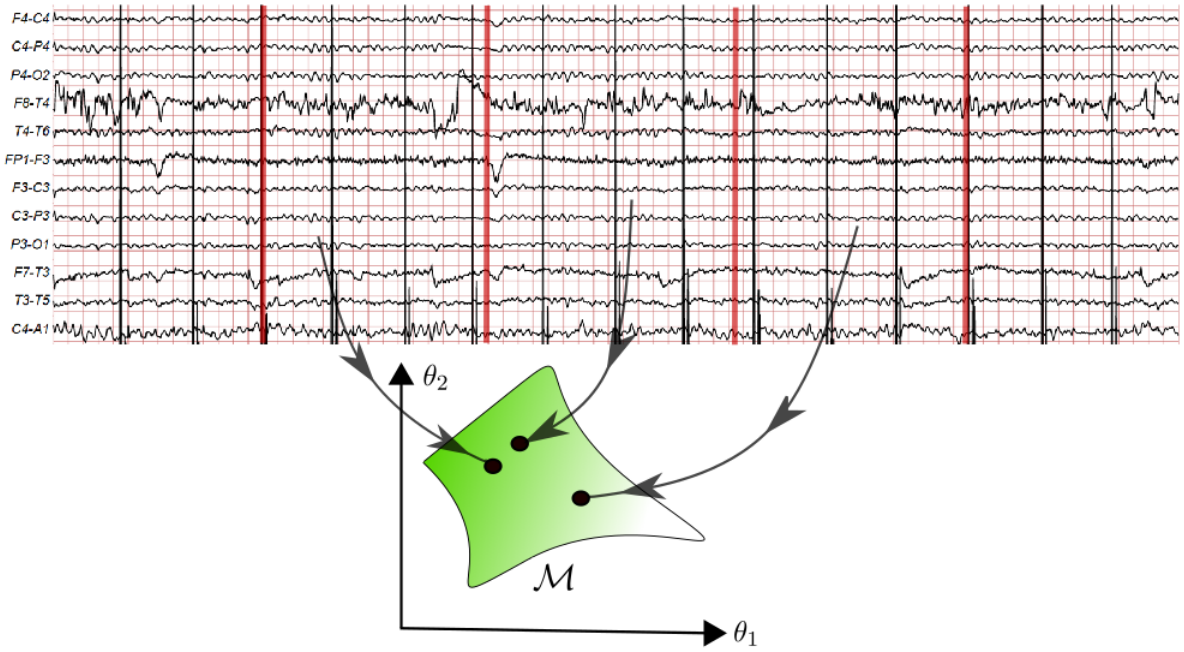
- The visualizations are still pretty noisy when the data are noisy
- Can we just diffuse more to denoise more?
- Increasing the amount of diffusion can oversmooth
  - Loss of true signal
- Can we do better than this?





Dynamical Systems Approach

- Let's add some more structural assumptions
- Let's assume that the data are generated by a dynamical system (e.g. EEG measurements)
  - I.e., we now have a time component to the data
- **Goal**: learn a low-dimensional representation of the data



Diffusion with Dynamical Systems

- State-space formalism:

$$\boldsymbol{x}_t = \boldsymbol{y}_t(\boldsymbol{\theta}_t) + \boldsymbol{\xi}_t$$
$$d\theta_t^i = a^i\left(\theta_t^i\right)dt + dw_t^i, \qquad i = 1, \dots, m$$

  - $\boldsymbol{x}_t$ is the observed time series while $\boldsymbol{\theta}_t$ represents the unobserved states that drive the process
  - $\boldsymbol{\xi}_t$ is a stationary process independent of $\boldsymbol{y}_t$ (noise)
  - $\boldsymbol{x}_t$ is a corrupted version of a clean process $\boldsymbol{y}_t$ that is driven by $\boldsymbol{\theta}_t$
  - The unknown drift functions a^i are independent of θ^j when $j \neq i$
    - $\Rightarrow$ we assume local independence between θ_t^i and θ_t^j when $j \neq i$
  - w_t^i are independent white noise
- Can we recover $\boldsymbol{\theta}_t$ from $\boldsymbol{x}_t$?

Diffusion with Dynamical Systems

- State-space formalism:

$$\boldsymbol{x}_t = \boldsymbol{y}_t(\boldsymbol{\theta}_t) + \boldsymbol{\xi}_t$$
$$d\theta_t^i = a^i\left(\theta_t^i\right)dt + dw_t^i, \qquad i = 1, \dots, m$$

- Can view $\boldsymbol{y}_t$ as being drawn from the conditional pdf $p(\boldsymbol{y}|\boldsymbol{\theta})$
- **Important insight 1** (Talmon & Coifman, 2015): the pdf $p(\boldsymbol{x}|\boldsymbol{\theta})$ is a linear transformation of $p(\boldsymbol{y}|\boldsymbol{\theta})$
- New feature space: histograms $\boldsymbol{h}_t$ taken within a time window centered at $\boldsymbol{x}_t$
- **Important insight 2**: the expected value of the histograms, e.g. $\mathbb{E}[\boldsymbol{h}_t]$, is a linear transformation of $p(\boldsymbol{x}|\boldsymbol{\theta})$

Diffusion with Dynamical Systems

- **Important insight 1** (Talmon & Coifman, 2015): the pdf $p(\boldsymbol{x}|\boldsymbol{\theta})$ is a linear transformation of $p(\boldsymbol{y}|\boldsymbol{\theta})$
- **Important insight 2**: the expected value of the histograms, e.g. $\mathbb{E}[\boldsymbol{h}_t]$, is a linear transformation of $p(\boldsymbol{x}|\boldsymbol{\theta})$
- **Important insight 3**: the Mahalanobis distance is invariant under linear transformations
- **Therefore**, the following distance is noise resilient (Talmon & Coifman, 2015):

$$D^2(\boldsymbol{x}_t, \boldsymbol{x}_s) = (\mathbb{E}[\boldsymbol{h}_t] - \mathbb{E}[\boldsymbol{h}_s])^T C_t^{-1} (\mathbb{E}[\boldsymbol{h}_t] - \mathbb{E}[\boldsymbol{h}_s])$$

- I.e., under certain assumptions,

$$\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_s\|^2 \approx D^2(\boldsymbol{x}_t, \boldsymbol{x}_s)$$

Dynamical Information Geometry (Duque et al., 2019)

- Compute the Mahalanobis between expected values:

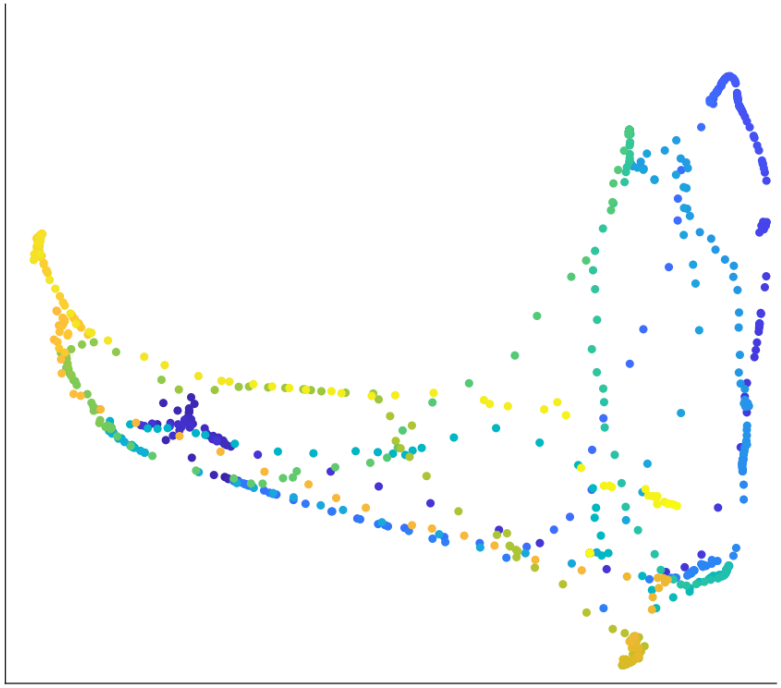
$$d^2(\boldsymbol{z}_t, \boldsymbol{z}_s) = (\mathbb{E}(\boldsymbol{h}_t) - \mathbb{E}(\boldsymbol{h}_s))^T (\boldsymbol{C}_t + \boldsymbol{C}_s)^{-1} (\mathbb{E}(\boldsymbol{h}_t) - \mathbb{E}(\boldsymbol{h}_s)),$$

  - Expected values and covariance matrices computed in a time window of length L_2
- Input these distances into diffusion maps to get EIG (Talmon and Coifman, 2015)
  - A noise-resilient dimensionality reduction method
- Input these distances into PHATE to get DIG
  - A noise resilient visualization method

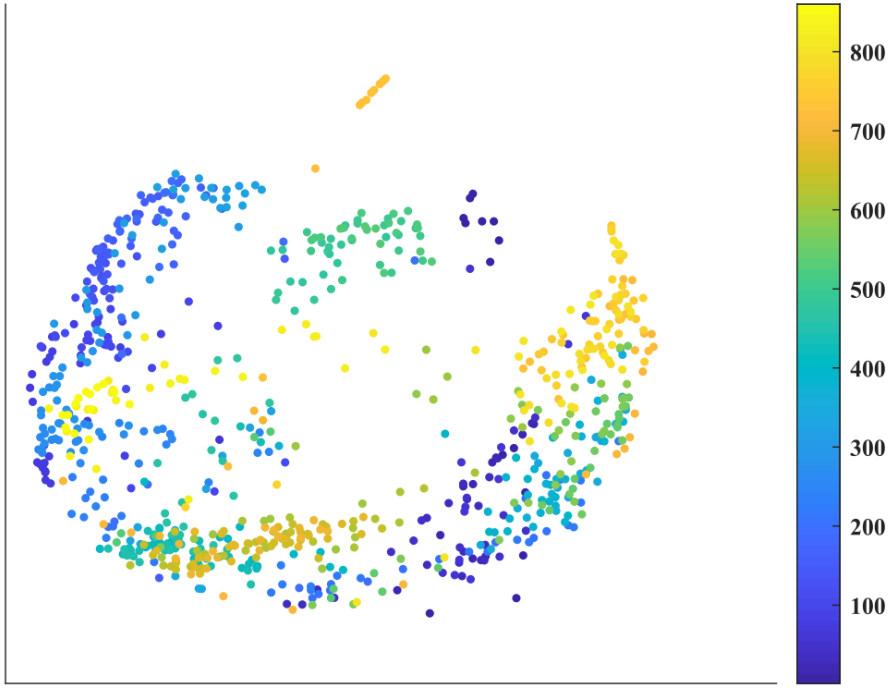
EEG Results: Information Distances

- Applied DIG to EEG sleep data (Terzano et al., 2002; Goldberger et al., 2000)
- Visualizations colored by time

DIG (Mahalanobis Distance)



An alternative geodesic distance



Information Distances

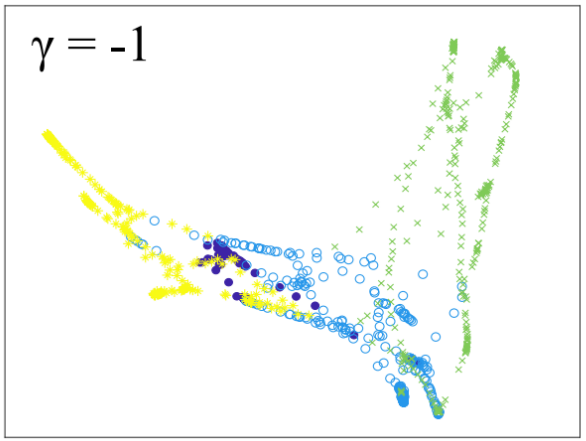
- We also explored other information distances (applied to the diffusion operator) besides the potential distance

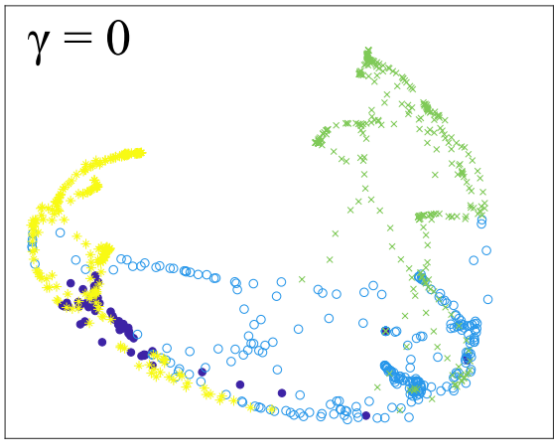
$$D^2_{\gamma,t}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \begin{cases} \sum_{k=1}^{N} \frac{(\log P^t_{ki} - \log P^t_{kj})^2}{\phi_0(k)}, & \gamma = 1 \\ \sum_{k=1}^{N} \frac{(P^t_{ki} - P^t_{kj})^2}{\phi_0(k)}, & \gamma = -1 \\ \sum_{k=1}^{N} \frac{2((P^t_{ki})^{\frac{1-\gamma}{2}} - (P^t_{kj})^{\frac{1-\gamma}{2}})^2}{(1-\gamma)\phi_0(k)}, & \text{-1} < \gamma < 1. \end{cases}$$

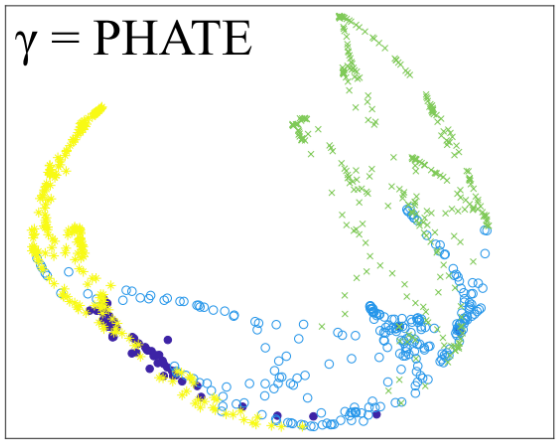
- Top corresponds to potential distance (PHATE), middle to diffusion maps

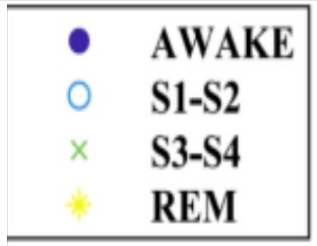
EEG Results: Information Distances

- Visualizations colored by sleep stage

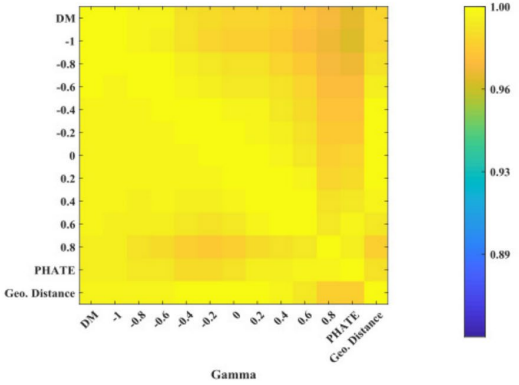




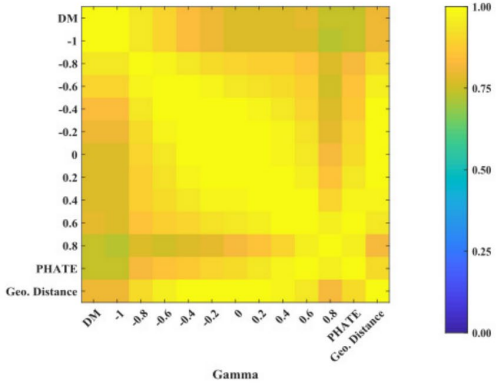




Local distortion

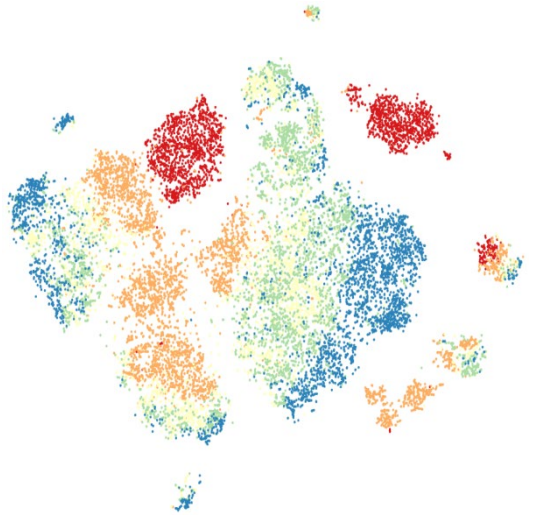


Global distortion



PHATE Summary

- Data have structure at different scales
  - Local branching structure
  - Global relationship between branches
- Existing visualization methods fail to account for all scales
- PHATE captures both global and local structure
- PHATE reveals new biology
- DIG extends this to better denoise the data



Further reading

- PHATE
  - Paper: https://doi.org/10.1101/120378
  - Code: https://github.com/KrishnaswamyLab/PHATE
- DIG
  - Paper: https://doi.org/10.1109/MLSP.2019.8918875
  - Code: https://github.com/KevinMoonLab/DIG
- T-SNE
  - https://lvdmaaten.github.io/tsne/
  - https://distill.pub/2016/misread-tsne/
- UMAP
  - Damrich and Hamprecht (2021): https://openreview.net/pdf?id=DKRciknd MGC
  - Kobak and Linderman (2021): https://www.nature.com/articles/s41587-020-00809-z