Comp790-166: Computational Biology

Lecture 9

February 8, 2022

Good Morning Question

What is the intuition for how MAGIC fixes noise and / or dropout? If given a cell, how would I adjust its features, given the rest of the data?

Intermission for Announcements

- **Homework 1** : Now available online. Work together, discuss, but write up your own solution and send it to me by 11:59pm eastern time on February 23. `https://github.com/natalies-teaching/Comp790-166-CompBio-Spring2022/tree/main/Homework1`
- **Project Resource** : Open Single Cell Challenges (with benchmark datasets) `https://openproblems.bio/neurips_2021/`

Today

- Visualization for capturing complex patterns in cellular differentiation
- Single-Cell Data Augmentation
- Intro to GSP (graph signal processing)

A Dimension Reduction Method Well-Suited for Cellular Differentiation

PHATE aims to preserve between-cell population similarities according to the way that cells differentiate.

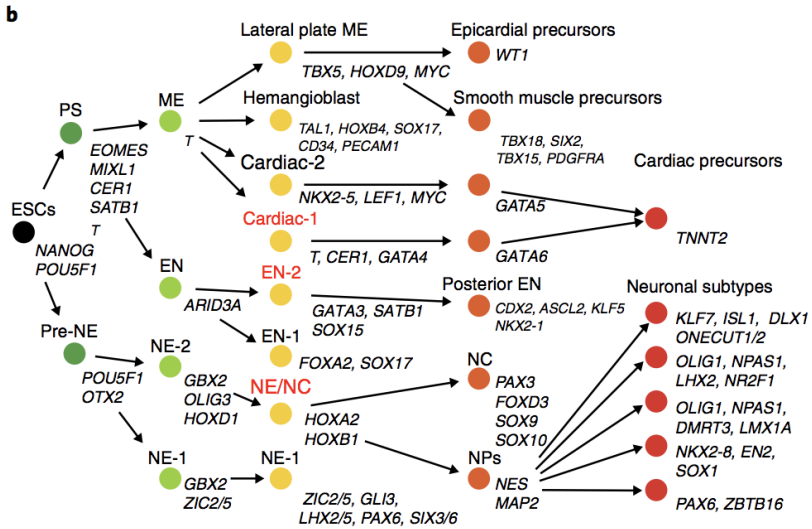


Figure: from Moon *et al.* Nature Biotechnology. 2019

Latent Structure

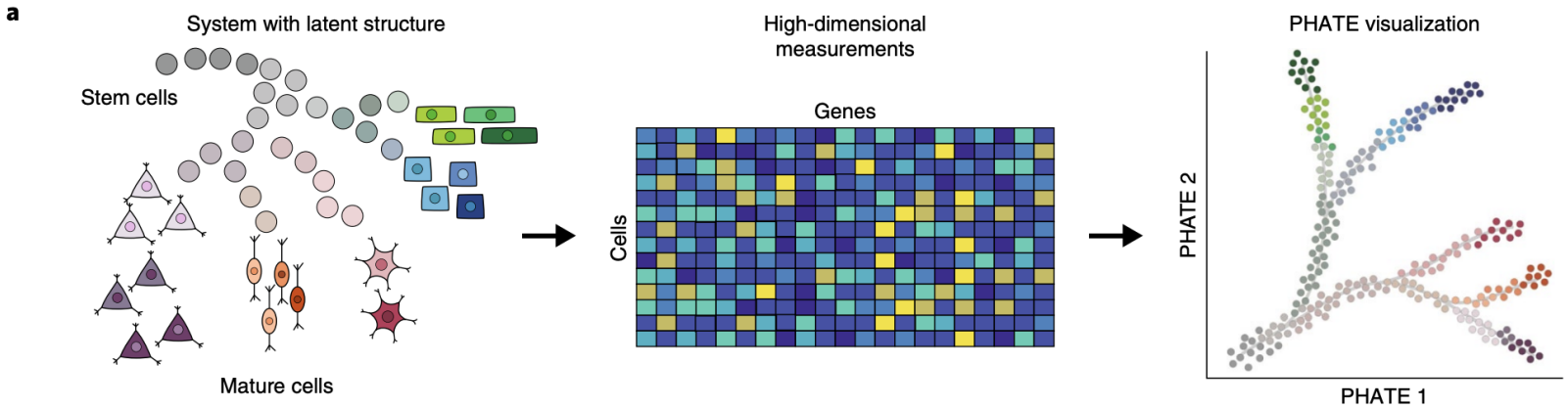


Figure: from Moon *et al.* Nature Biotechnology. 2019

PHATE Overview

Table 1 | General steps in the PHATE algorithm
Input: Data matrix, algorithm parameters (Methods)
Output: The PHATE visualization
(1) Compute the pairwise distances from the data matrix.
(2) Transform the distances to affinities to encode local information.
(3) Learn global relationships via the diffusion process.
(4) Encode the learned relationships using the potential distance.
(5) Embed the potential distance information into low dimensions for visualization.

Figure: from Moon *et al.* Nature Biotechnology. 2019

The First Few Steps are Identical to MAGIC

- **Step 1:** Cell $\times$ Cell distance matrix (based on Euclidean distance between cells)
- **Step 2:** Convert to Cell $\times$ Cell row-stochastic affinity matrix $\rightarrow$, $\mathbf{P}$, such that for the i-th row of $\mathbf{P}$, such that $\sum_j P_{ij} = 1$.
- **Step 3:** Power $\mathbf{P} \rightarrow \mathbf{P}^t$.

A Complementary Method to Determine Optimal t

- The choice of t will determine how much noise you want to filter.
- Let $[\eta(t)]_i = \lambda_i^t / \sum_{j=0}^{N-1} \lambda_j^t$ be the probability distribution defined by the non-negative eigenvalues of $\mathbf{P}^t$.

The von Neumann entropy $H(t)$ is then computed based on $[\eta(t)]_i$ and decreases as $t \to \infty$.

$$H(t) = -\sum_{i=1}^{N} [\eta(t)]_i \log([\eta(t)]_i) \tag{1}$$

Relationship Between t and $H(t)$

Higher t means less noise, but the implications of t are super powerful!

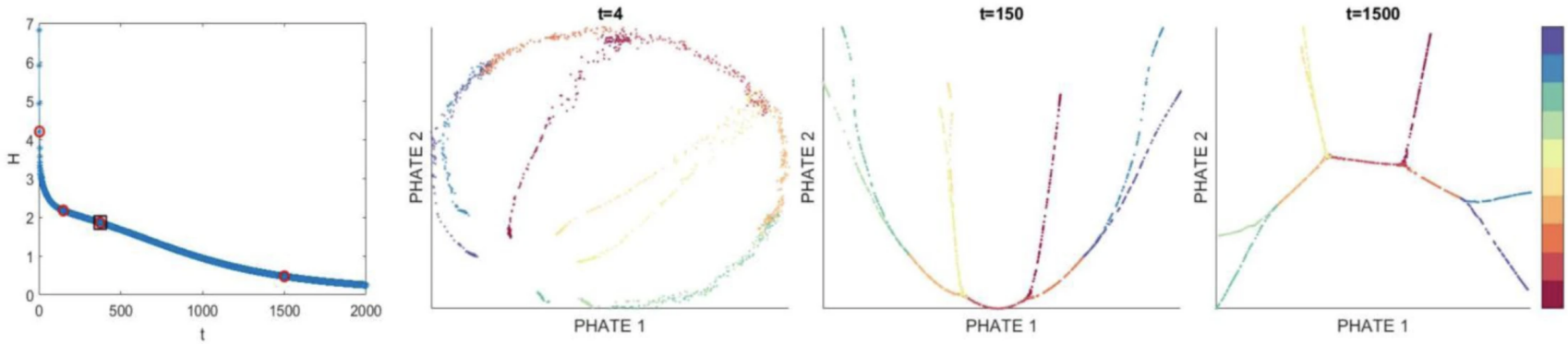


Figure: from Moon *et al.* Nature Biotechnology. 2019

Computing Potential Distances

- Potential distance is computed as a divergence between the associated diffusion probability distributions of the two cells to all other cells

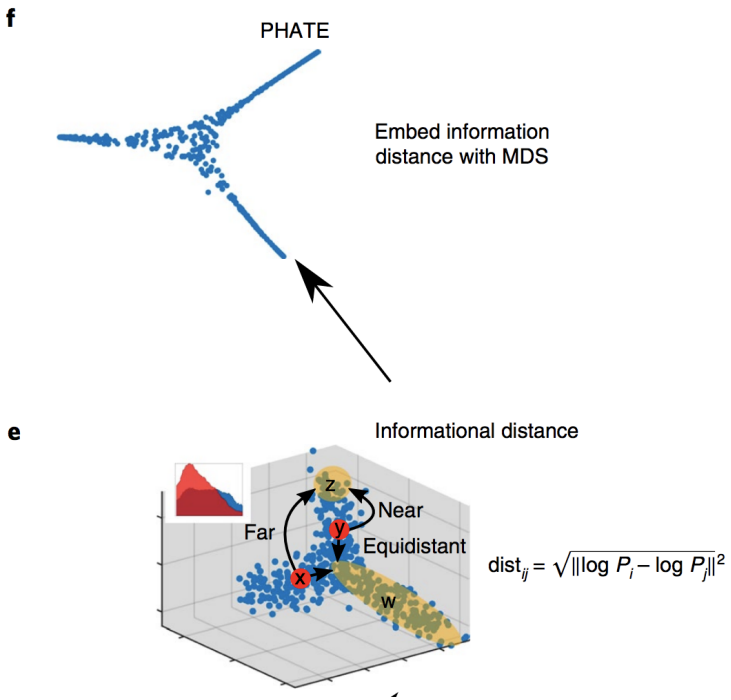


Figure: from Moon *et al.* Nature Biotechnology. 2019

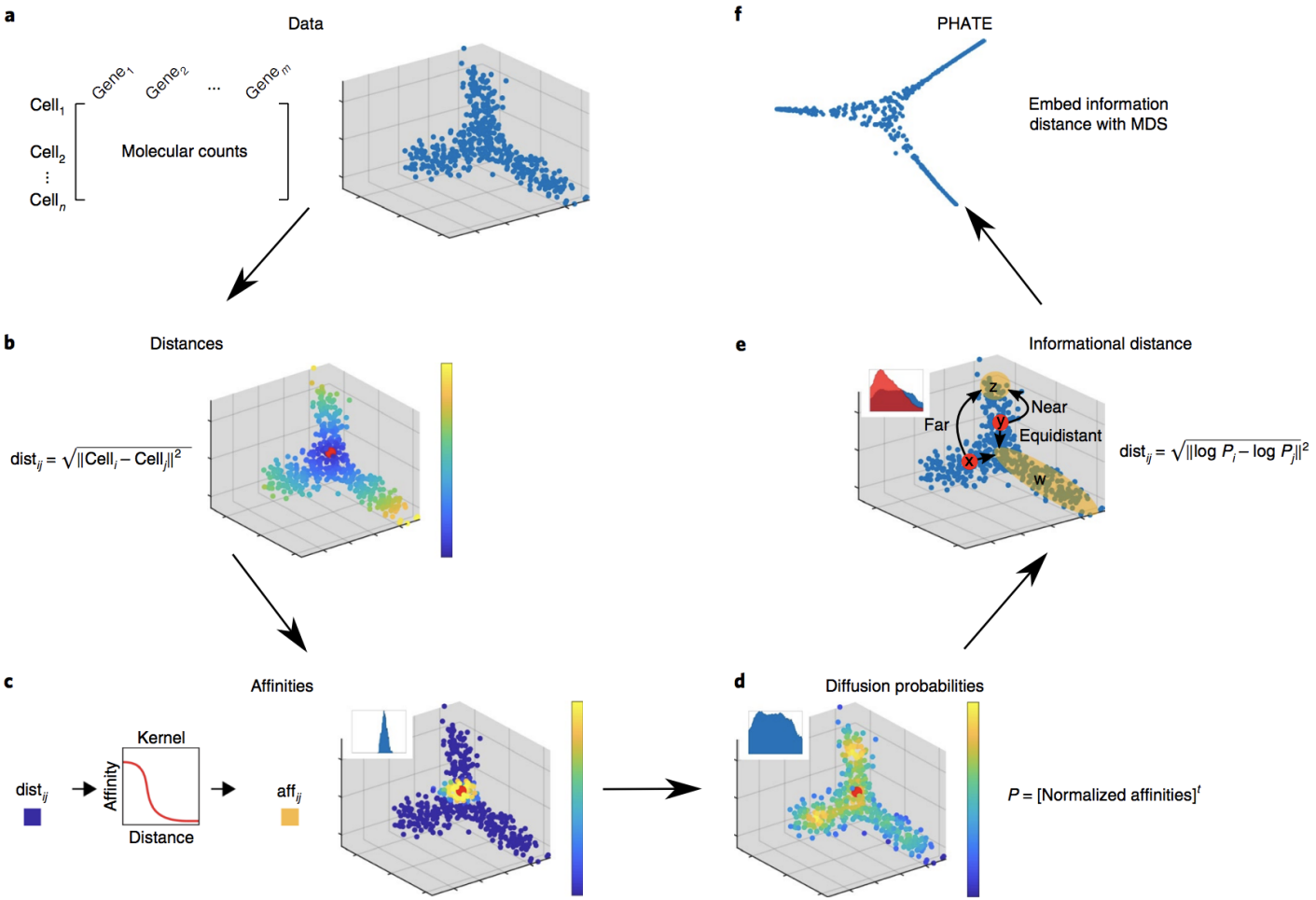
Potential Distances, Written Formally

- $U_{\mathbf{x}}^t = -\log(P_{\mathbf{x}}^t)$
- Then $\text{PD}(\mathbf{x}, \mathbf{y}) = ||U_{\mathbf{x}}^t - U_{\mathbf{y}}^t||_2$

Then use a classic algorithm called multi-dimensional scaling to project all points in two dimensions. Apply to $\mathfrak{V}^t$ (e.g. the potential distances from the powered to t $\mathbf{P}$).

$$\mathfrak{V}^t(\mathbf{x}, \mathbf{y}) = \left\| U_{\mathbf{x}}^t - U_{\mathbf{y}}^t \right\|_2, \mathbf{x}, \mathbf{y} \in \mathcal{X}$$

Recap


a
Data
Gene1
Gene2
...
Genem
Cell1
Cell2
Cellm
Molecular counts
b
Distances
distij = √||Celli − Cellj||2
c
Affinities
Kernel
distij
Affinity
Distance
affij
d
Diffusion probabilities
P = [Normalized affinities]t
e
Informational distance
z
Near
Far
y
Equidistant
x
w
distij = √||log Pi − log Pj||2
f
PHATE
Embed information
distance with MDS

PHATE preserves branch structure

As expected tSNE is preserving local similarities, but not necessarily distances between groups of points.

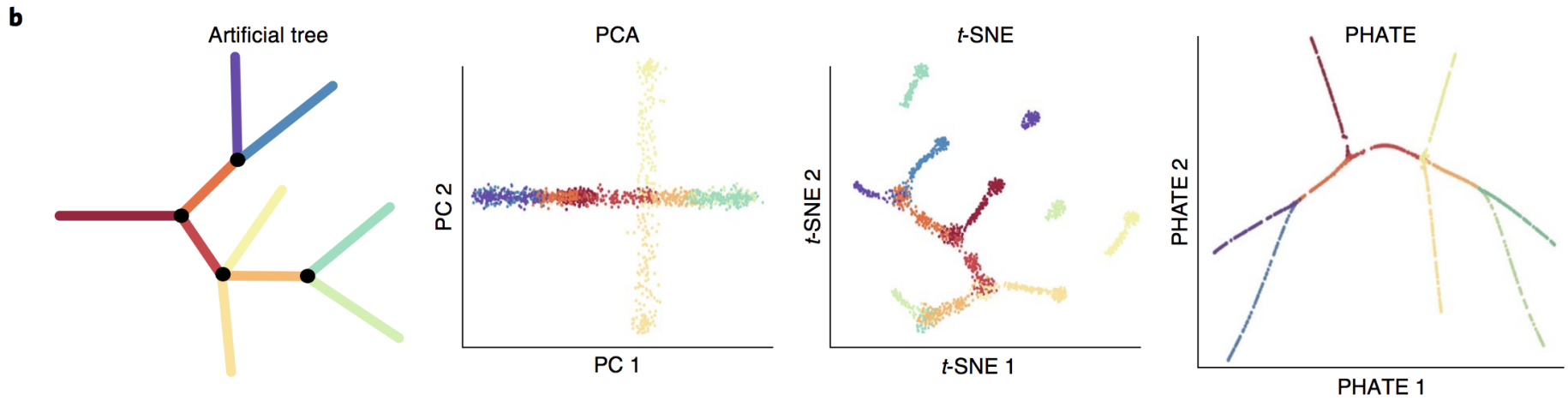


Figure: from Moon *et al.* Nature Biotechnology. 2019

And Preserves Cellular Differentiation Patterns!

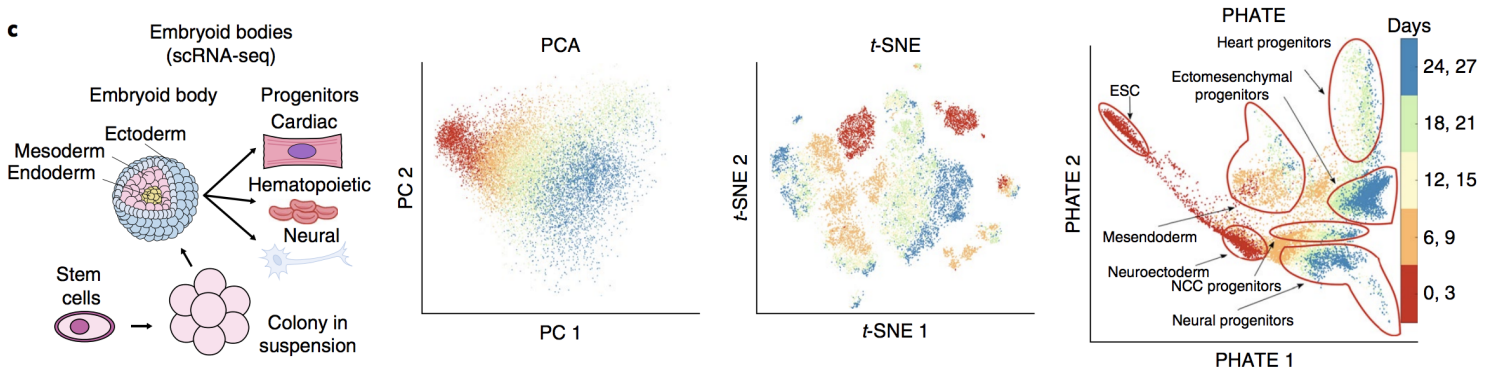


Figure: from Moon *et al.* Nature Biotechnology. 2019

Compare Phate to Other Dimension Reduction Techniques

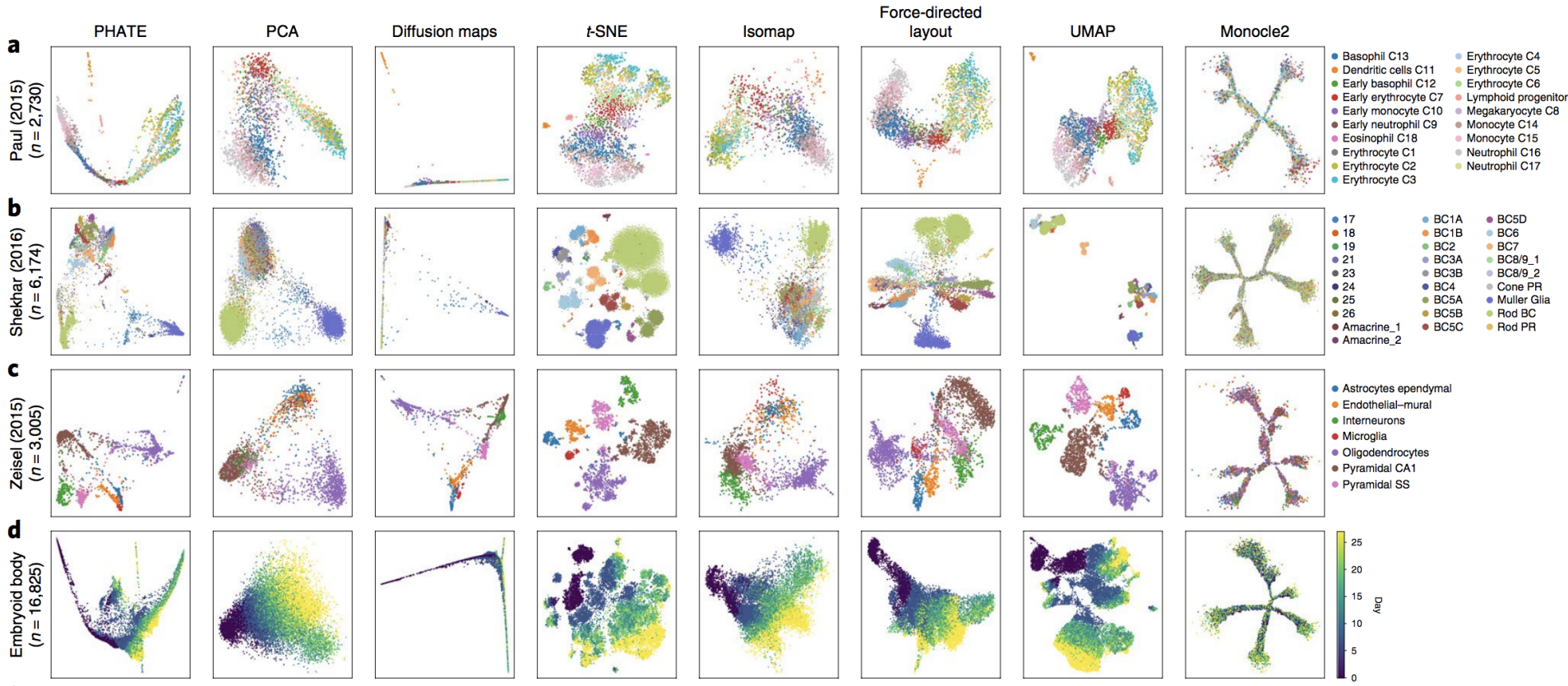


Figure: ffrom Moon *et al.* Nature Biotechnology. 2019

Rare + Sparse Cell Populations

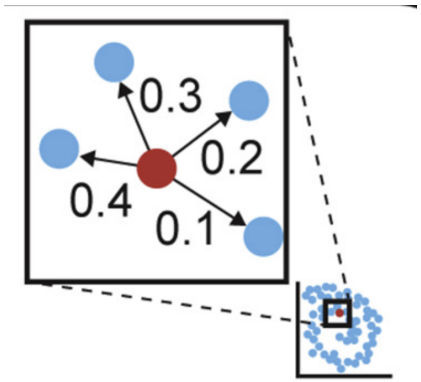


Figure: from MAGIC paper

- ‘Dense‘ cell-populations vs ‘Sparse‘ cell-populations.
- Low-frequency cell-populations and data artifacts can fail to be under-represented and not accurately reflect the underlying biology.
- Are members of a particular cell-type not there or are they just very infrequent?

General Problems with Downstream Tasks from Sparse Data

- Imbalanced classes can affect classification accuracy
- In clustering, imbalanced ‘ground-truth‘ clusters can cause distortion or mis-representation of the clusters in the data
- Too few samples/observations can cause mis-representation of the dependencies or correlations between features.

Welcome SUGAR

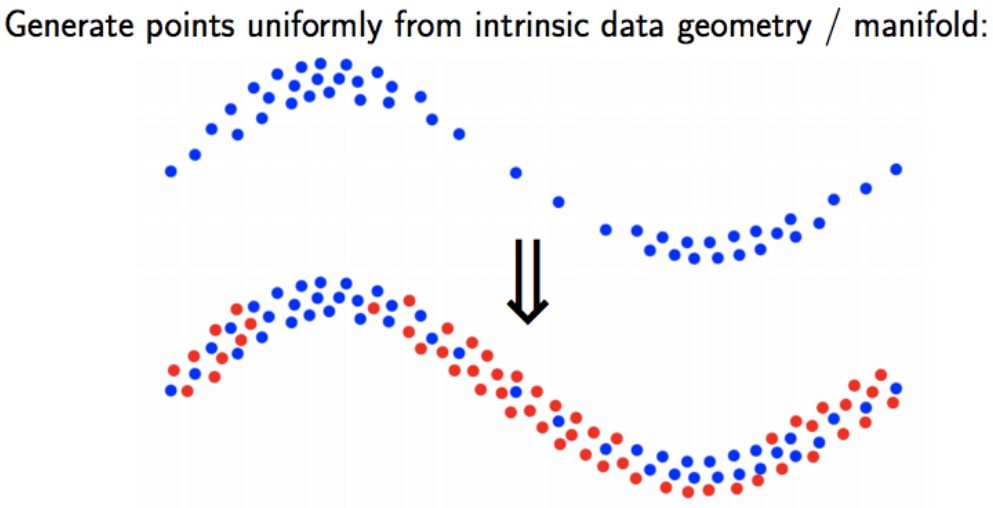


Figure: from Lindenbaum *et al.* NeurIPS. 2018

- Most generative modeling approaches seek to learn and replicate the data density
- SUGAR is a data-generation approach that uses the underlying geometry of the data (e.g. a random walk based approach)

Common Data Generation Approaches

- Older Techniques
  - **Parametric Models :** Specify a model and optimize the parameters through maximum likelihood optimization.
  - Use the learned model to generate new data
  - Use a histogram or kernel to estimate generating distributions
- Newer techniques to generate additional points from complicated data distributions
  - GAN (generative adversarial network) $\rightarrow$ `https://papers.nips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf`
  - VAE (variational autoencoders) $\rightarrow$ `https://arxiv.org/abs/1606.05908`

Remembering our Good Friend Gaussian Kernel

For a pair of nodes, i and j, the strength of their connection can be computed as $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$, or

$$K_{ij} = \exp(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}) \tag{2}$$

We also remember our row-stochastic markov matrix, $\mathbf{P}$ computed from $\mathbf{K}$.

Measure Based Gaussian Correlation

- Given your data, $\mathbf{X}$, define a set of reference points, $\mathbf{r}$ with $r \in \mathbf{X}$.
- Define $\mu(\mathbf{r})$ as some measure over the reference points

Then define a new kernel, $\hat{\mathcal{K}}(\mathbf{x}_i, \mathbf{x}_j)$ as,

$$\hat{\mathcal{K}}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{\mathbf{r} \in \mathbf{X}} \mathcal{K}(\mathbf{x}_i, \mathbf{r})(\mathbf{x}_j, \mathbf{r})\mu(\mathbf{r}) \tag{3}$$

For our purposes, $\mu(\mathbf{r})$ will be some value that relates to sparsity and is the inverse of node r's degree.

Problem Formulation for Data Generation Problem

- Let $\mathcal{M}$ be a d-dimensional manifold such that $\mathcal{M} \in \mathbb{R}^d$
- Let $\mathbf{X} \subset \mathcal{M}$ be a subset of points samples from $\mathcal{M}$ with $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\} \in \mathbf{X}$
- Assuming that instances, $\mathbf{X}$ are unevenly sampled from $\mathcal{M}$, we seek a set of new points, $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_M\}$ that also lie of $\mathcal{M}$ and that $\mathbf{X} \cup \mathbf{Y}$ is uniform.

Synthesis Using Geometrically Aligned Random Walks

Algorithm 1 SUGAR: Synthesis Using Geometrically Aligned Random-walks

Input: Dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}, \boldsymbol{x}_i \in \mathbb{R}^D$.

Output: Generated set of points $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_M\}, \boldsymbol{y}_i \in \mathbb{R}^D$.

1: Compute the diffusion geometry operators $\boldsymbol{K}$, $\boldsymbol{P}$, and degrees $\hat{d}(i)$, $i = 1, ..., N$ (see Sec. 3)
2: Define a sparsity measure $\hat{s}(i), i = 1, ..., N$ (Eq. 2).
3: Estimate a local covariance $\boldsymbol{\Sigma}_i$, $i = 1, ..., N$, using k nearest neighbors around each $\boldsymbol{x}_i$.
4: For each point $i = 1, ..., N$ draw $\hat{\ell}(i)$ vectors (see Sec. 4.3) from a Gaussian distribution $\mathcal{N}(\boldsymbol{x}_i, \boldsymbol{\Sigma}_i)$. Let $\hat{\boldsymbol{Y}}_0$ be a matrix with these $M = \sum_{i=1}^{N} \hat{\ell}(i)$ generated vectors as its rows.

Figure: from Lindenbaum *et al.* NeurIPS. 2018. Let's walk through steps 1-4 for now.

Synthesis Using Geometrically Aligned Random Walks

- **Specify Kernel:** Initialized by forming a Gaussian Kernel over the input data, $\mathbf{X}$, $\mathbf{G}_{\mathbf{x}}$.
  - Use $\mathbf{G}_{\mathbf{x}}$ to estimate the degree of each node i, $d(i)$.
  - The sparsity of each point, $s(i)$ is defined as the inverse degree of node i or $s(i) = 1/d(i)$.
- **Sample According to Each Point** Next, sample $\ell(i)$ points, $\mathbf{h}_j \in \mathbf{H}_i$ for $j = 1 \ldots \ell_i$ around each $\mathbf{x}_i \in \mathbf{X}$ from a localized gaussian distribution (e.g. k-nearest points around i).
  - $G_i = \mathcal{N}(\mathbf{x}_i, \Sigma_i)$

Practical : Sampling from MV Gaussians

numpy.random.multivariate_normal

random.multivariate_normal(*mean, cov, size=None, check_valid='warn', tol=1e-8*)

Draw random samples from a multivariate normal distribution.

The multivariate normal, multinormal or Gaussian distribution is a generalization of the one-dimensional normal distribution to higher dimensions. Such a distribution is specified by its mean and covariance matrix. These parameters are analogous to the mean (average or "center") and variance (standard deviation, or "width," squared) of the one-dimensional normal distribution.

Back to SUGAR

- Let $\mathbf{Y}_0$ be the set of all new $M = \sum_i \ell(i)$ points generated around each i, with $\mathbf{Y}_0 = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_M\}$
- **MGC Kernel:** Now use affinities between points in $\mathbf{X}$ and $\mathbf{Y}_0$. Here, points in $\mathbf{X}$ are used as reference.

$$\hat{\mathcal{K}}(\mathbf{y}_i, \mathbf{y}_j) = \sum_r \mathcal{K}(\mathbf{y}_i, \mathbf{x}_r)\mathcal{K}(\mathbf{x}_r, \mathbf{y}_j)s(r) \tag{4}$$

Pulling $\mathbf{Y}_0$ towards sparser regions of $\mathcal{M}$

Pasted psuedo code for the second part of SUGAR

5: Compute the sparsity based diffusion operator $\hat{\boldsymbol{P}}$ (see Sec 4.2).
6: Apply the operator $\hat{\boldsymbol{P}}$ at time instant t to the new generated points in $\hat{\boldsymbol{Y}}_0$ to get diffused points as rows of $\boldsymbol{Y}_t = \hat{\boldsymbol{P}}^t \cdot \boldsymbol{Y}_0$.
7: Rescale $\boldsymbol{Y}_t$ to get the output $\boldsymbol{Y}[\cdot, j] = \boldsymbol{Y}_t[\cdot, j] \cdot \frac{\text{percentile}(\boldsymbol{X}_{[\cdot,j]},.99)}{\max \boldsymbol{Y}_{t[\cdot,j]}}$, $j = 1, \ldots, D$, in order to fit the original range of feature values in the data.

Figure: from Lindenbaum *et al.* NeurIPS. 2018.

Diffusion Operator Again

- Take affinities from $\hat{\mathcal{K}}$ and convert them to $\mathbf{P}$, the row-normalized Markov matrix.
- This will allow us to correct points in $\mathbf{Y}_0$ according to neighborhood regions

As you will all recognize, powering $\mathbf{P}$ as $\mathbf{P}^t$ estimates the probability of successfully traveling between nodes with t steps. The transformed matrix, $\mathbf{Y}_t$ is computed as

$$\mathbf{Y}_t = \mathbf{P}^t \times \mathbf{Y}_o \tag{5}$$

Same Story Regarding t

Remember in PHATE, t was chosen according to the knee point of the Von-Neumann entropy of the normalized eigenvalues of $\mathbf{P}^t$.

Experiments : Synthetic and Biological

Stanford Bunny

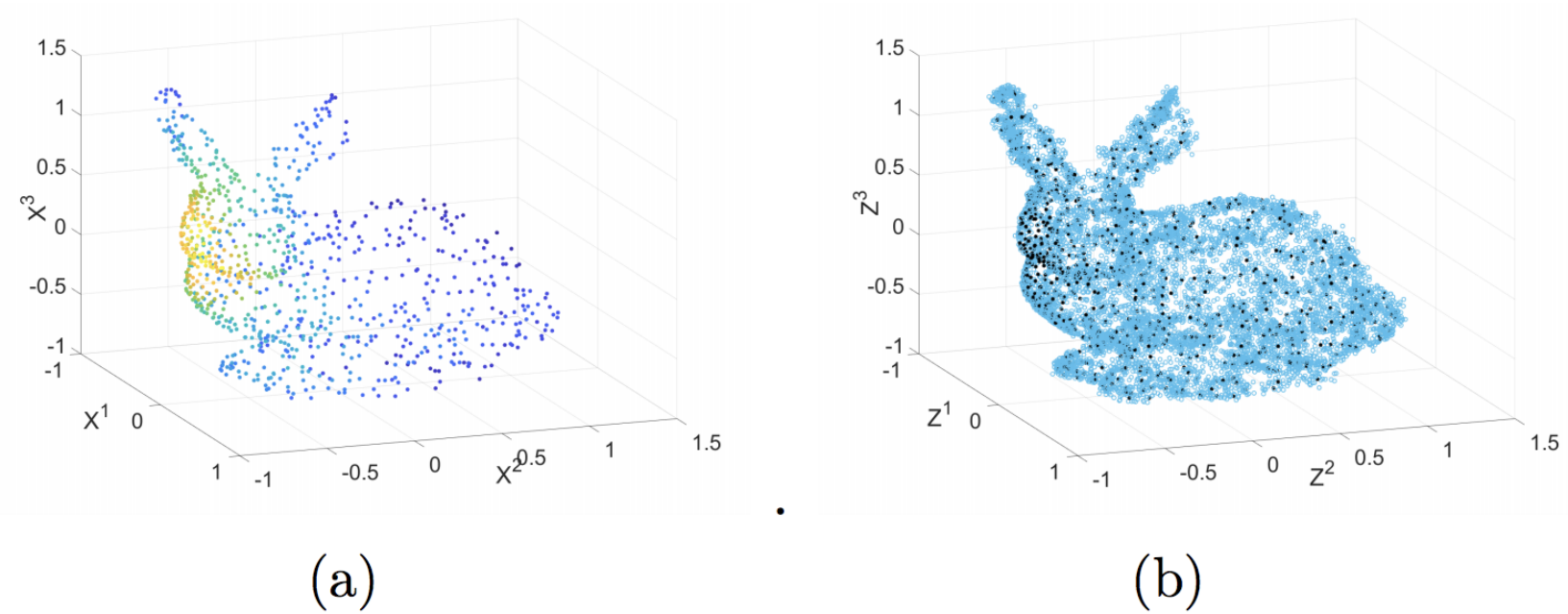


Figure: from Lindenbaum *et al.* NeurIPS. 2018. (**a**). The original set, $\mathbf{X}$ of points, colored by node degree. (**b**) $\mathbf{Y}_0$ are generated points (blue) and original points, $\mathbf{X}$ (black).

Stanford Bunny Part II

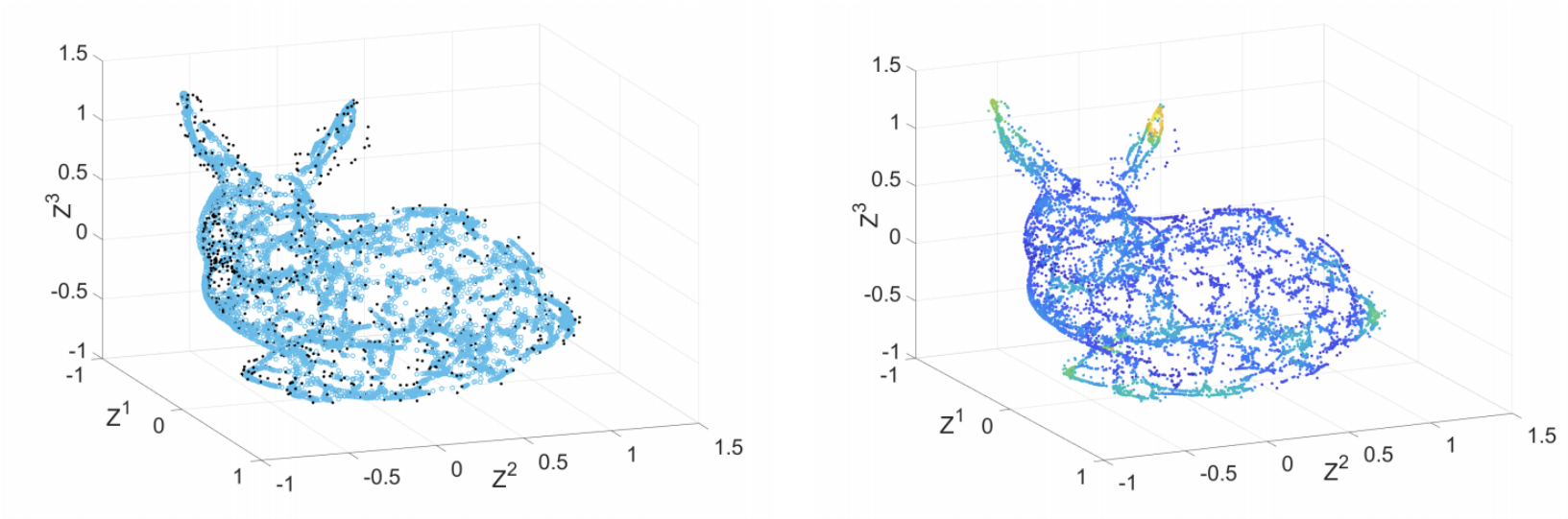


Figure: from Lindenbaum *et al.* NeurIPS. 2018. (**c**). Original and generated points, before MGC diffusion. (**d**) Generated points (**Y**) after MGC diffusion. Points are colored by degree.

Application : Augmented Clustering

SUGAR was used to generate additional data points to improve the quality of clusters identified with k-means.

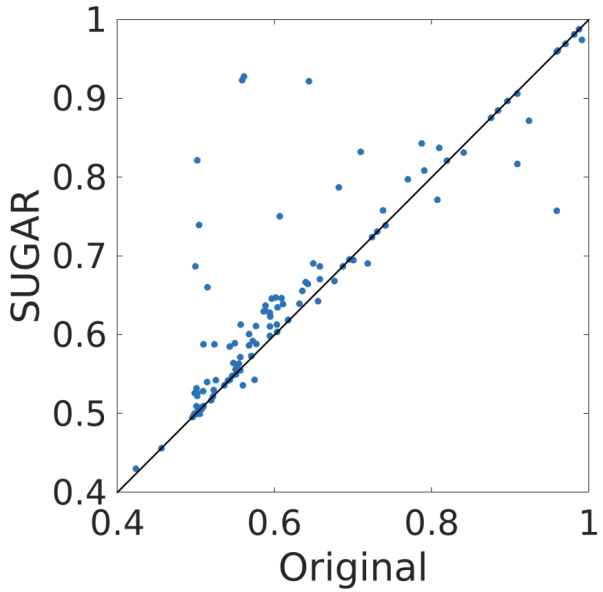


Figure: from Lindenbaum *et al.* NeurIPS. 2018. In 119 datasets, the data were augmented with additional datapoints using sugar. Adjusted Rand Index was computed for the original data vs data + SUGAR.

SUGAR on Single-Cell

SUGAR was using to augment cells in a single-cell RNAseq dataset.

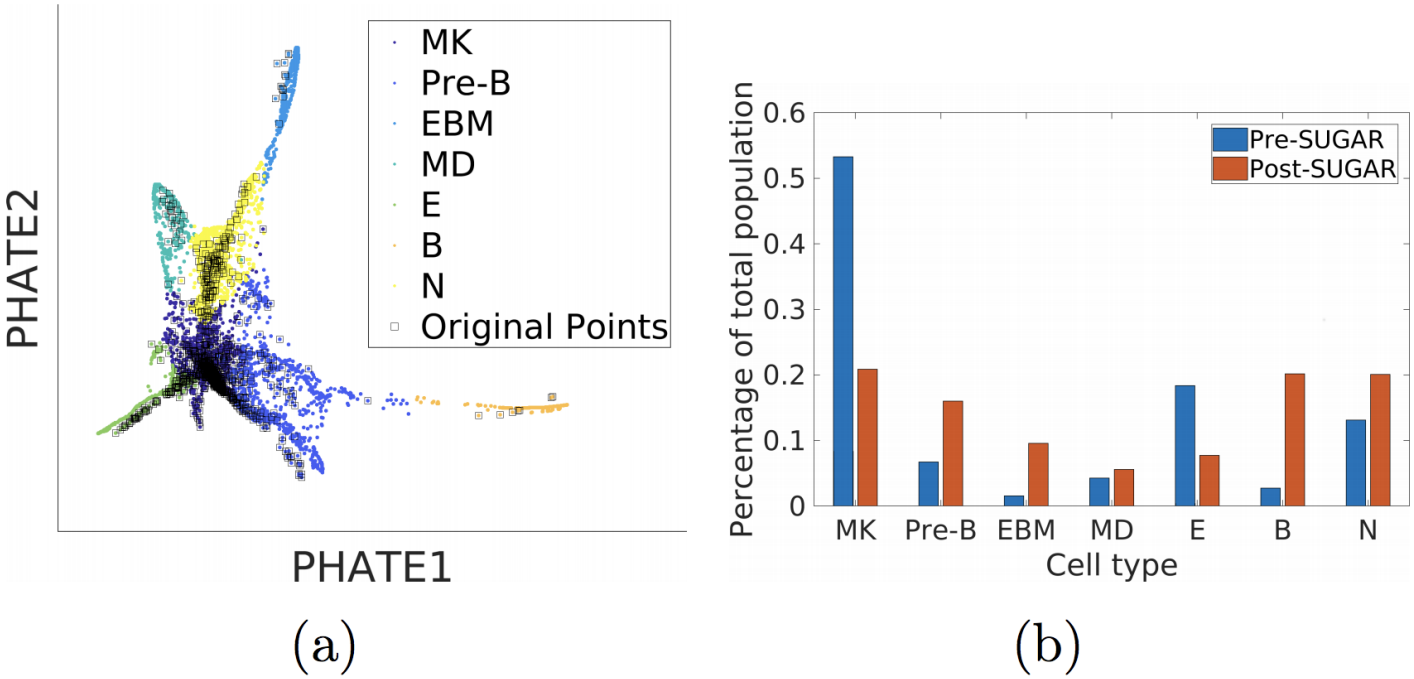


Figure: from Lindenbaum *et al.* NeurIPS. 2018

Maintaining Intra-Module Marker Co-Expression

Among cells assigned to the same module or cluster, after SUGAR lead to higher intra-module between-marker correlation.

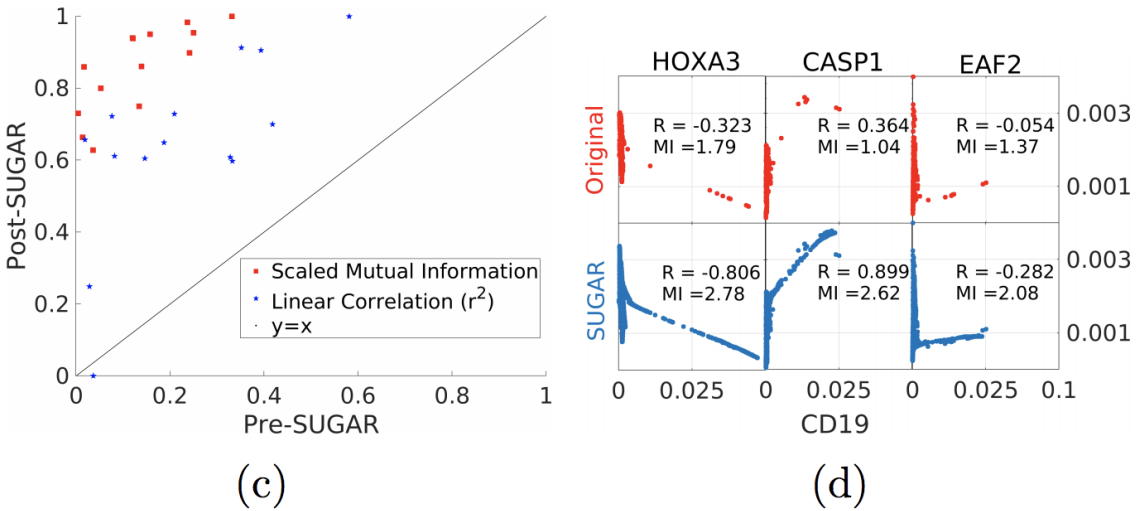


Figure: from Lindenbaum *et al.* NeurIPS. 2018

Graph Signal Processing

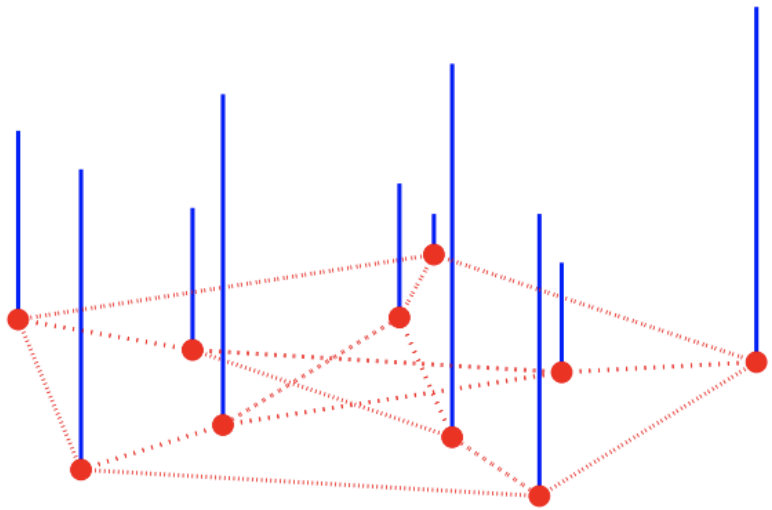
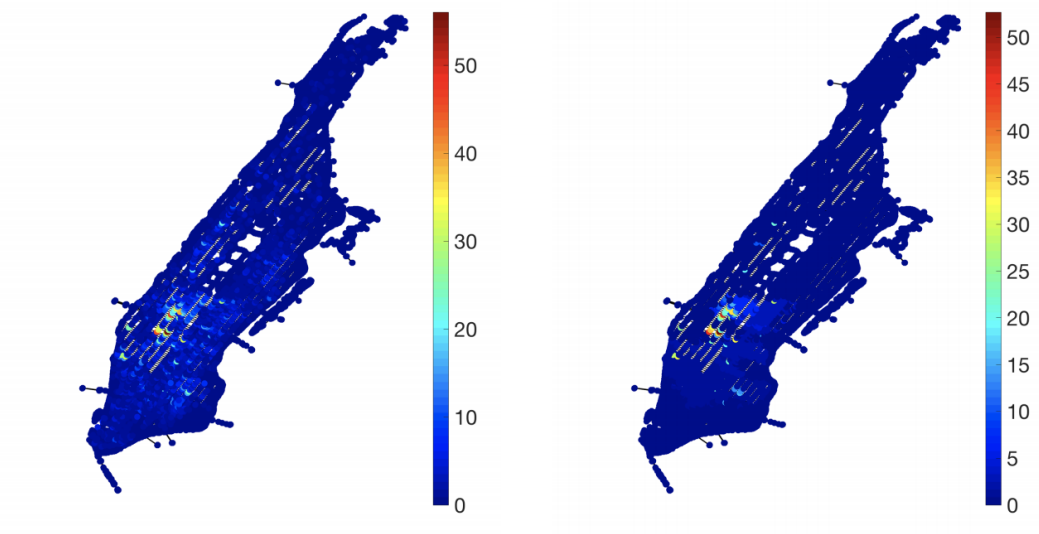


Figure: from Shuman *et al.* ArXiv. The purpose is to study the interplay between some signal and graph connectivity.

Piecewise Smooth Assumption

Translation: Nodes that are close (in terms of geodesic distance) on the graph should have similar signals. You can approximate the signal of a node, based on its neighbors.



(a) Actual data (b) Approximation

Figure: from `https://arxiv.org/abs/1712.00468`. Here the graph is street intersections in Manhattan and the signal is taxi pickups.

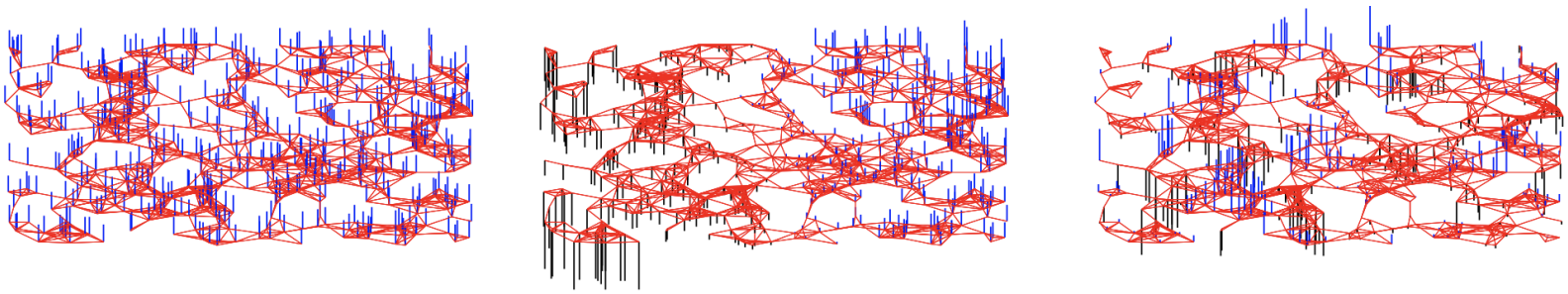
How Localized is the Signal?

Remember, our friend Graph Laplacian ($\mathbf{L} = \mathbf{D} - \mathbf{A}$),

- Some very nice theory falls out about the eigenvalues of the Laplacian matrix in terms of how 'localized' a graph signal, $\mathbf{f}$, is. For example $\mathbf{f}$ could be an expression of some protein.
  - First re-write $\mathbf{f}$ in terms of eigenvectors of the Laplacian
  - The eigenvectors corresponding to the first few eigenvalues of $\mathbf{L}$ are considered **low frequency**, and hence entries of the eigenvector entries corresponding to nodes that are connected should be similar
  - For higher **high frequencies** corresponding to 'later' eigenvalues, the values of the eigenvectors of adjacent nodes will be more different.

Signal Specificity

Here we visualize eigenvector entries at nodes ($\mathbf{u}_0$,$\mathbf{u}_1$,$\mathbf{u}_{50}$)



$\mathbf{u}_0$ $\mathbf{u}_1$ $\mathbf{u}_{50}$

Figure: from GSP Review https://arxiv.org/abs/1211.0053

Similarly

Zero crossings mean that eigenvector entries are neighboring nodes will be different.

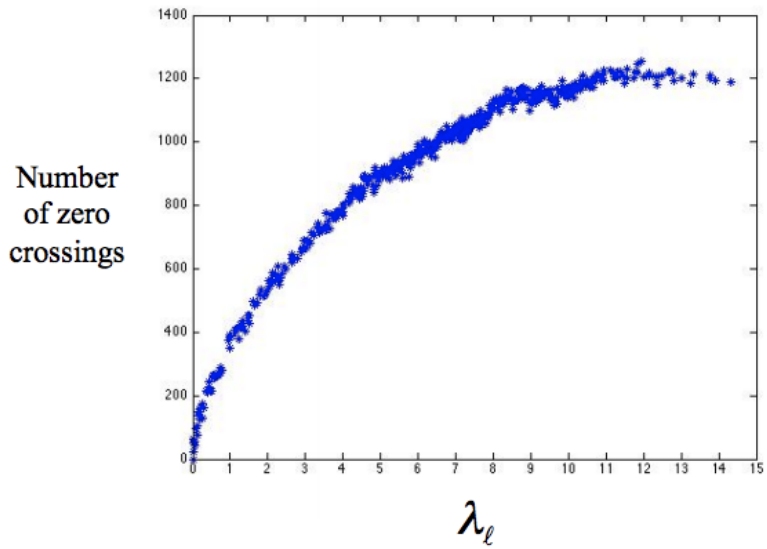


Figure: from GSP Review https://arxiv.org/abs/1211.0053