Comp683: Computational Biology

Lecture 10

February 17, 2025

Announcement

Homework 1 is online and available in the git repository. You can use the LaTeX template I provided or just submit as a PDF in Canvas upload by 11:59pm **February 28.**

Today

- Finish single-cell data augmentation
- Graph signal processing (GSP) overview
- Start MELD, a GSP-based differential abundance method.

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.

Unpacking details

- 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.

$$\hat{\mathcal{K}}(\mathbf{y}_i, \mathbf{y}_j) = \sum_r \mathcal{K}(\mathbf{y}_i, \mathbf{x}_r)\mathcal{K}(\mathbf{x}_r, \mathbf{y}_j)s(r) \tag{1}$$

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

$$\mathbf{Y}_t = \mathbf{P}^t \times \mathbf{Y}_o \tag{2}$$

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 $\mathbf{P}^t$.

- 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 $\mathbf{P}^t$.

$$H(t) = -\sum_{i=1}^{N} [\eta(t)]_i \log([\eta(t)]_i) \tag{3}$$

Experiments : Synthetic and Biological

Stanford Bunny

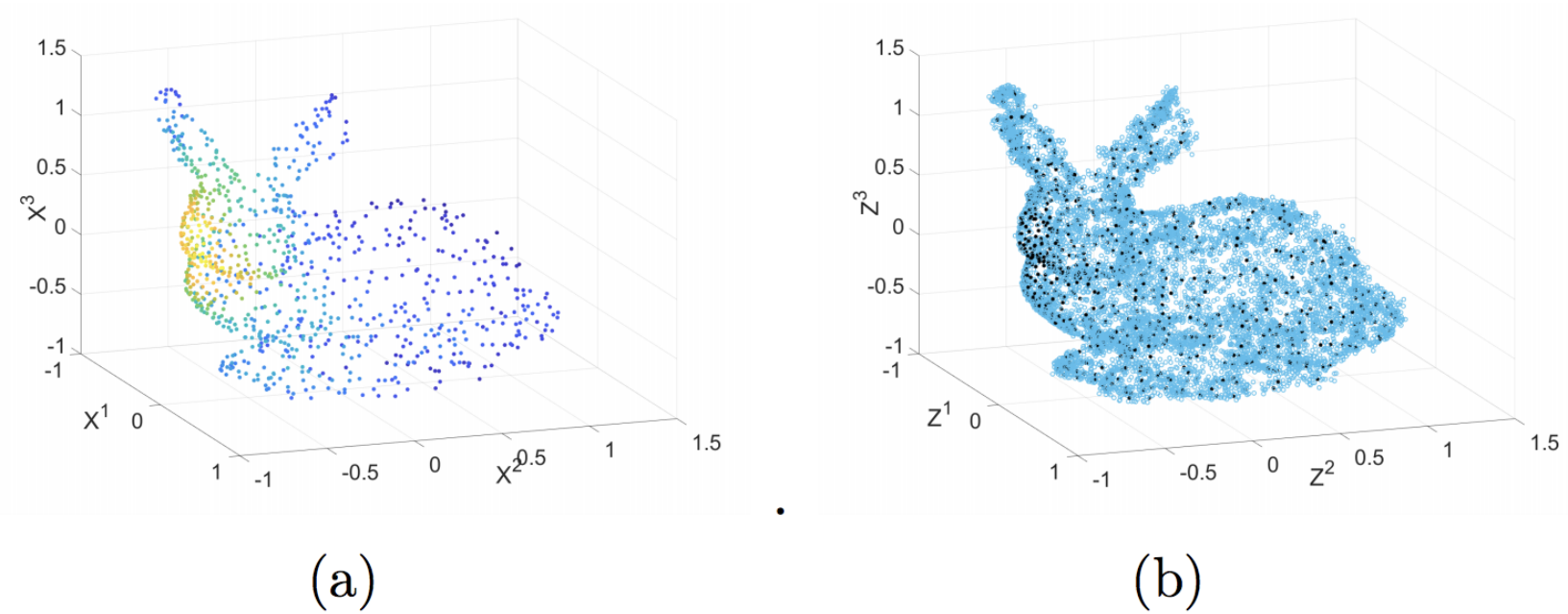


Figure: from Lindenbaum *et al.* NeurIPS. 2018. (**a**). The original set, $\mathbf{X}$ of points, colored by node degree. (**b**) $\mathbf{Y}_0$ are generated points (blue) and original points, $\mathbf{X}$ (black).

Stanford Bunny Part II

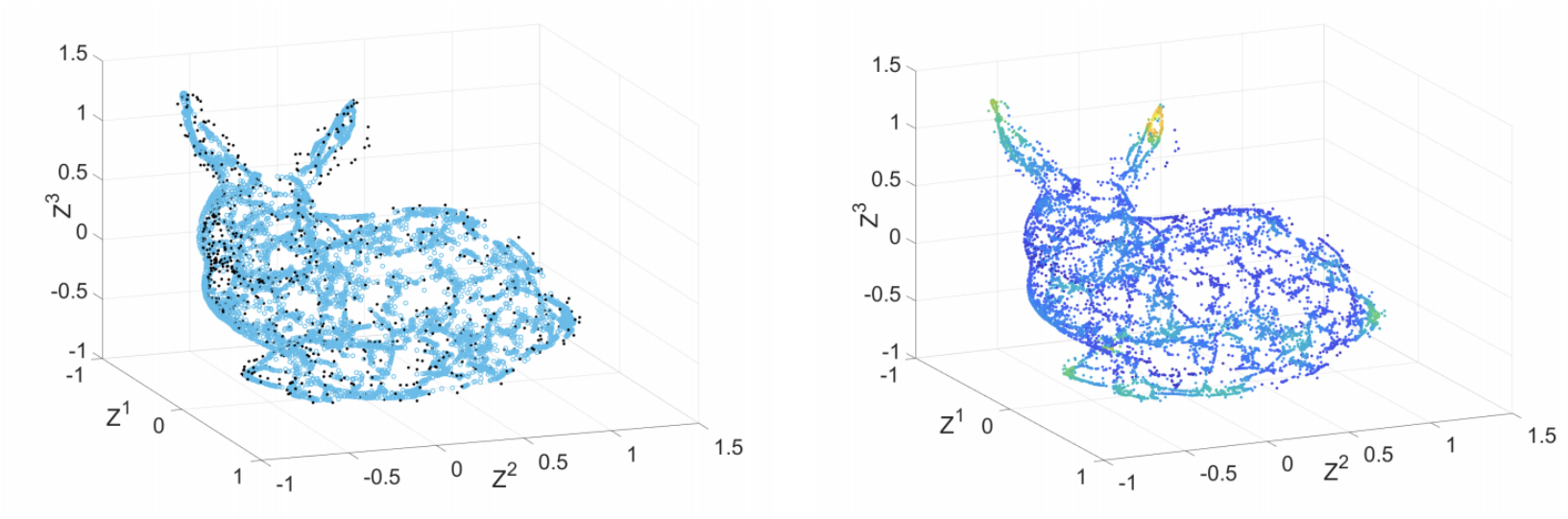


Figure: from Lindenbaum *et al.* NeurIPS. 2018. (**c**). Original and generated points, before MGC diffusion. (**d**) Generated points (**Y**) after MGC diffusion. Points are colored by degree.

Application : Augmented Clustering

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

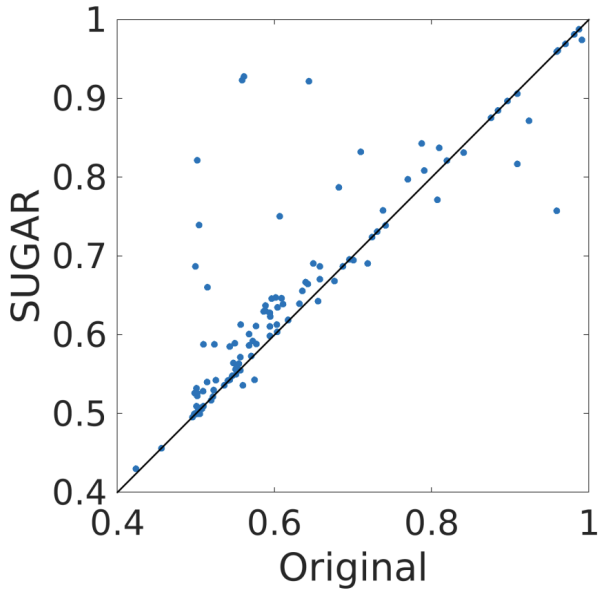


Figure: from Lindenbaum *et al.* NeurIPS. 2018. In 119 datasets, the data were augmented with additional datapoints using sugar. Adjusted Rand Index was computed for the original data vs data + SUGAR.

SUGAR on Single-Cell

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

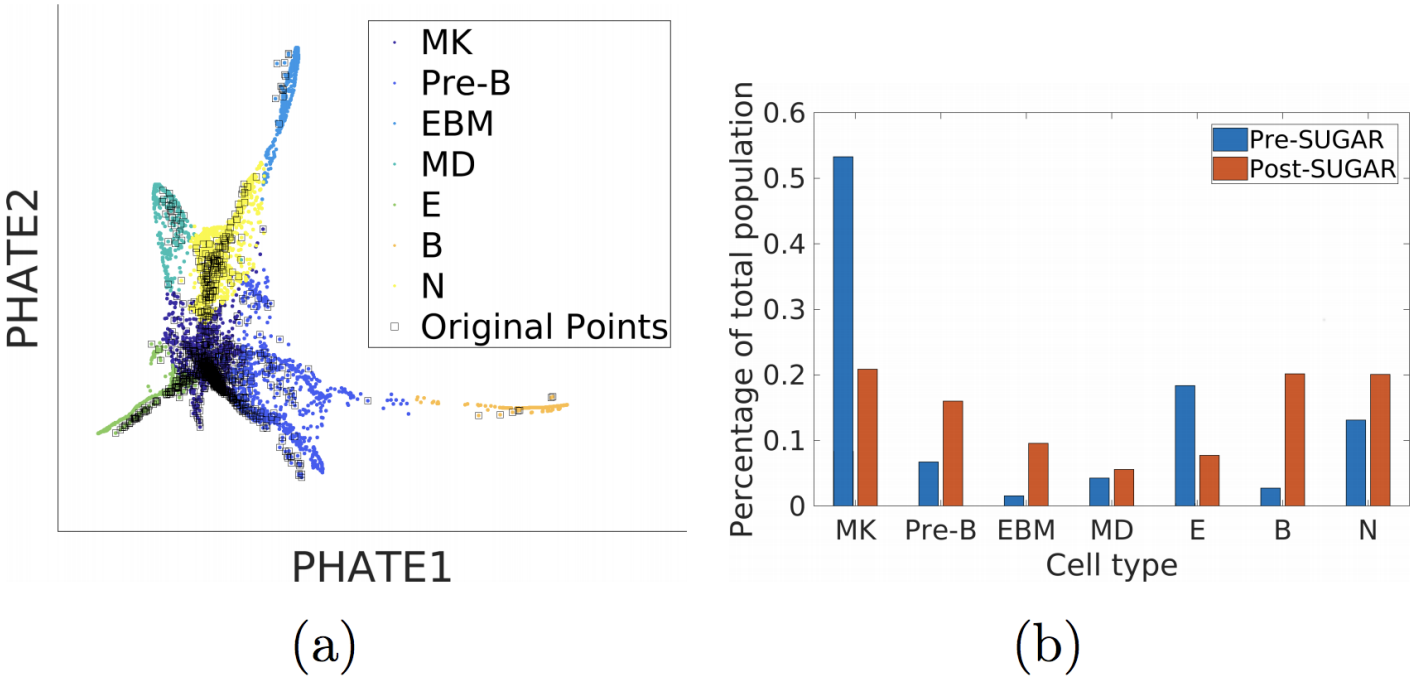


Figure: from Lindenbaum *et al.* NeurIPS. 2018

Maintaining Intra-Module Marker Co-Expression

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

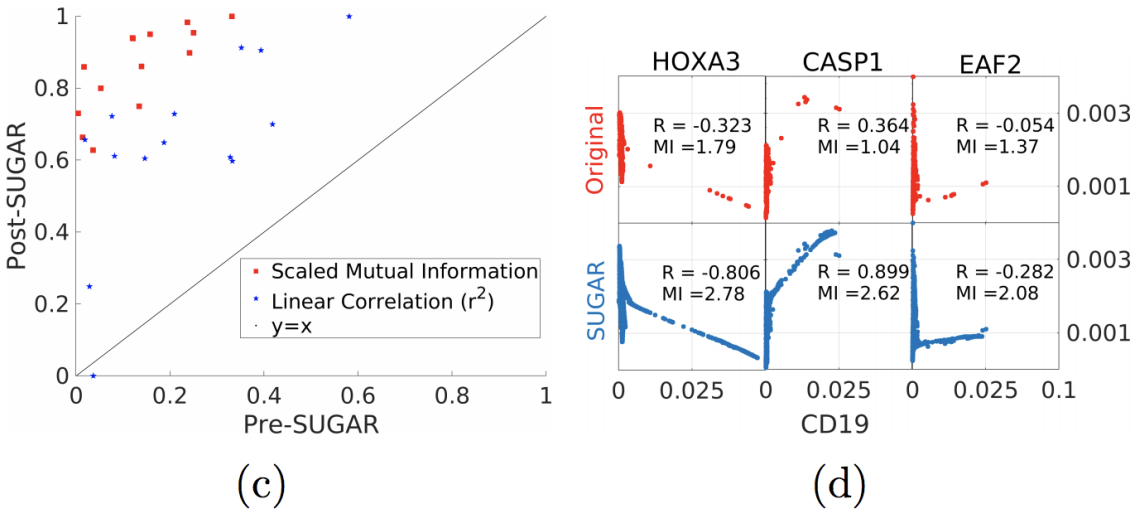


Figure: from Lindenbaum *et al.* NeurIPS. 2018

Graph Signal Processing

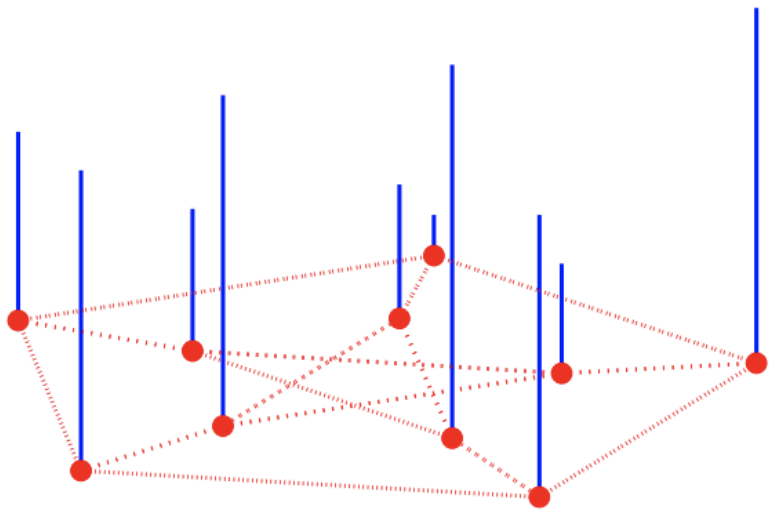


Figure: from Shuman *et al.* ArXiv. The purpose is to study the interplay between some signal (e.g. a clinical outcome or gene expression) and graph connectivity.

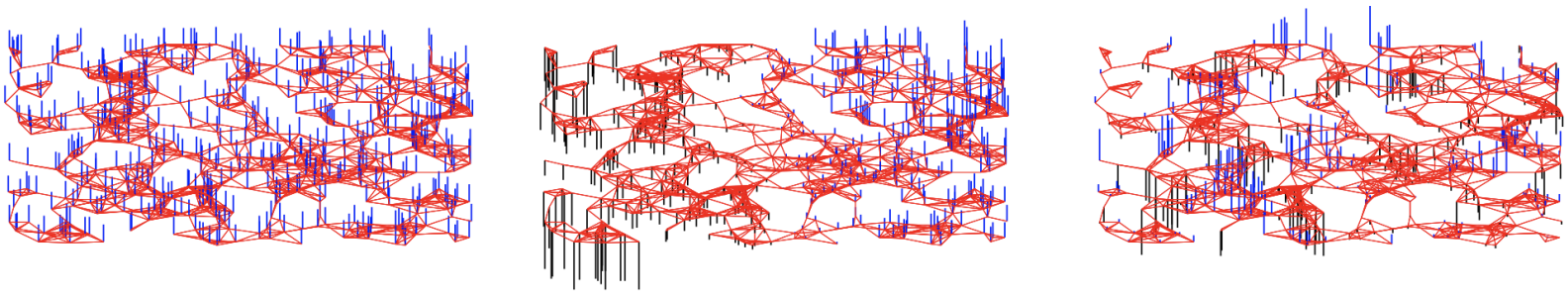
The Interplay Between Signal and Graph Structure

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

- We will use spectra of the graph Laplacian to link signal, $\mathbf{f}$ to graph structure and modulate as necessary. For example $\mathbf{f}$ could be a clinical label of a cell.
  - First re-write $\mathbf{f}$ in terms of eigenvectors of the Laplacian
  - The eigenvectors corresponding to the first *smallest* 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 larger 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

Same Concept Visualized A Bit Differently

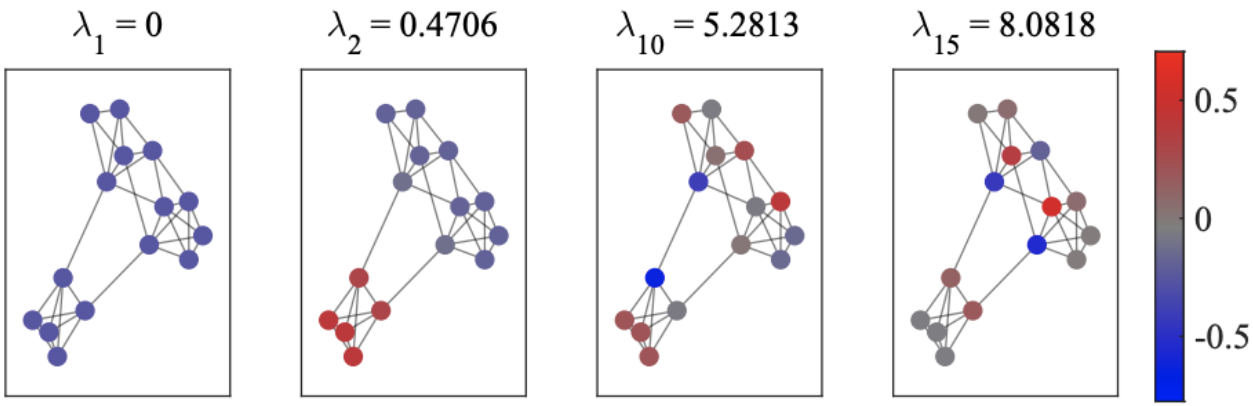


Figure: Notes are colored by their corresponding eigenvector component. From `https://arxiv.org/pdf/2008.01305.pdf`

Similarly

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

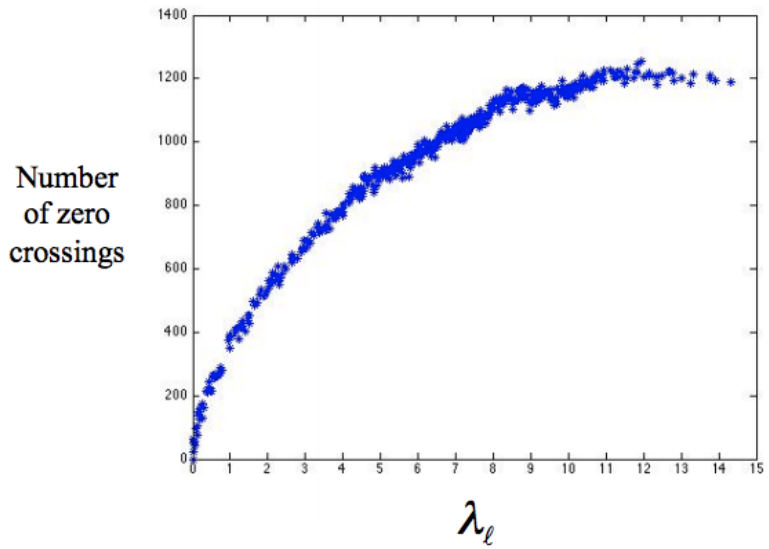


Figure: from GSP Review https://arxiv.org/abs/1211.0053

What is Graph Fourier Transform (on a high level?)

- Explain frequency content of the graph signal (e.g. experimental measurements/labels/etc) as a weighted sum of the eigenvectors of the Graph Laplacian
- The eigenvectors of the Graph Laplacian comprise the **Graph Fourier Basis and can help to decouple high and low frequency signals**

Local Variation of a Signal

The local variation of a signal or the sum of differences around a node can be written as,

$$(\mathcal{L}\mathbf{f})(i) = ([\mathbf{D} - \mathbf{A}]\mathbf{f})(i) \tag{4}$$

$$= d(i)\mathbf{f}(i) - \sum_j A_{ij}\mathbf{f}(j) \tag{5}$$

$$= \sum_j A_{ij}(\mathbf{f}(i) - \mathbf{f}(j)) \tag{6}$$

Local Variation Leads to Total Variation

The total variation of a signal on a graph is defined as follows and is also known as the Laplacian Quadratic Form

$$TV(\mathbf{f}) = \sum_{i,j} A_{ij}(\mathbf{f}(i) - \mathbf{f}(j))^2 \tag{7}$$

$$= \mathbf{f}^T \mathcal{L} \mathbf{f} \tag{8}$$

- Note here I have been assuming that we have an unweighted graph, but you could certainly substitute A_{ij} with a weighted version, W_{ij}

Getting to Graph Fourier Basis

- Start with the eigendecomposition of of $\mathbf{L}$ as $\mathbf{L} = \mathbf{\Psi \Lambda \Psi}^T$
- We can look at eigenvectors, $\mathbf{\Psi} = [\psi_1, \psi_2, \ldots, \psi_N]$ of $\mathcal{L}$
- and eigenvalues, $\mathbf{\Lambda} = [0 = \lambda_1 \leq \cdots \leq \lambda_N]$ of $\mathcal{L}$

The Graph Fourier Transform of a Signal

The ith frequency component of a signal, $\mathbf{f}$ is the inner product between $\boldsymbol{\psi}_i$ and $\mathbf{f}$ and can be written as,

$$\hat{f}_i = \psi_i^T \mathbf{f} \tag{9}$$

The Graph Fourier Transform (GFT) is written as,

$$\hat{\mathbf{f}} = \boldsymbol{\Psi}^T \mathbf{f} \tag{10}$$

Example on Three Different Graphs

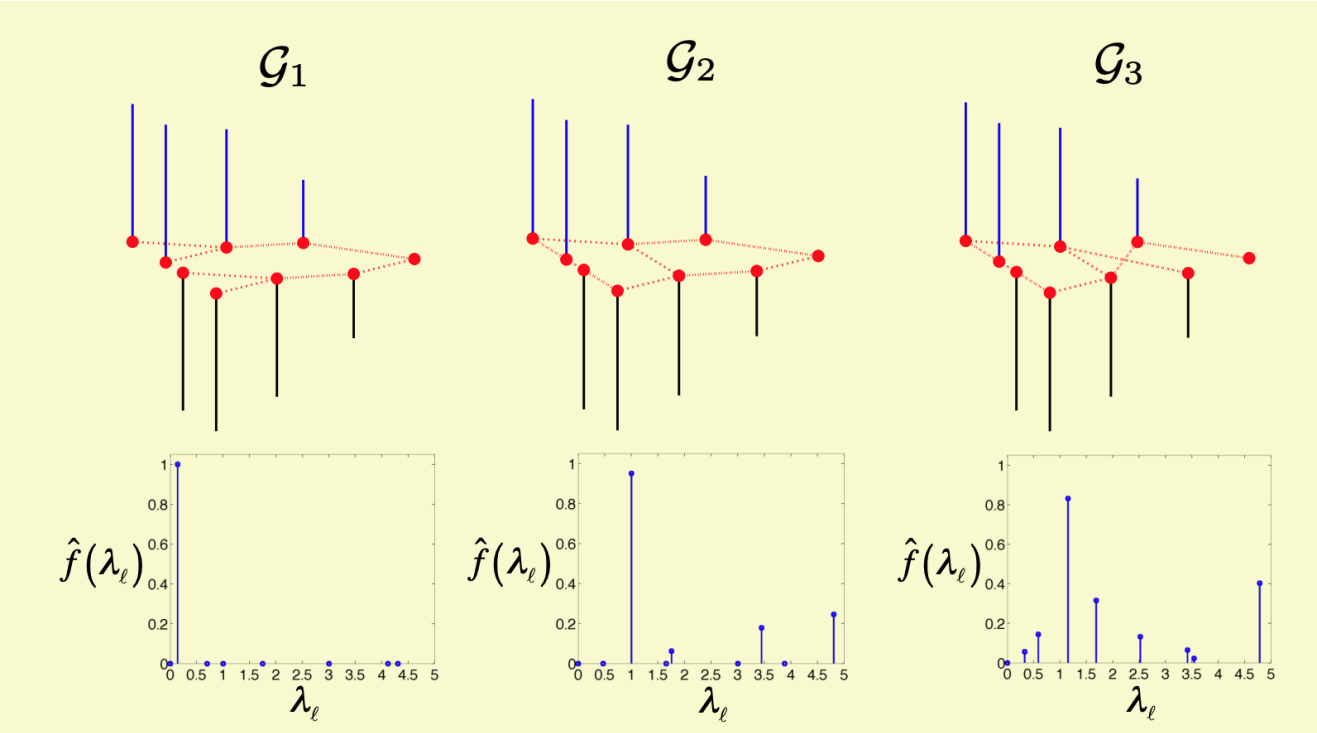


Figure: Vertex domain (top) vs spectral domain (bottom). Recall $\hat{f}(\lambda_l)$ is $\psi_l^T \mathbf{f}$. $\mathcal{G}_1$ has signals corresponding to graph structure and hence highest $\hat{f}(\lambda_l)$ for $l = 0$. The opposite is true for $\mathcal{G}_3$.

How Does This Relate to Quadratic Form

Recall quadratic form is $\mathbf{f}^T\mathbf{L}\mathbf{f}$

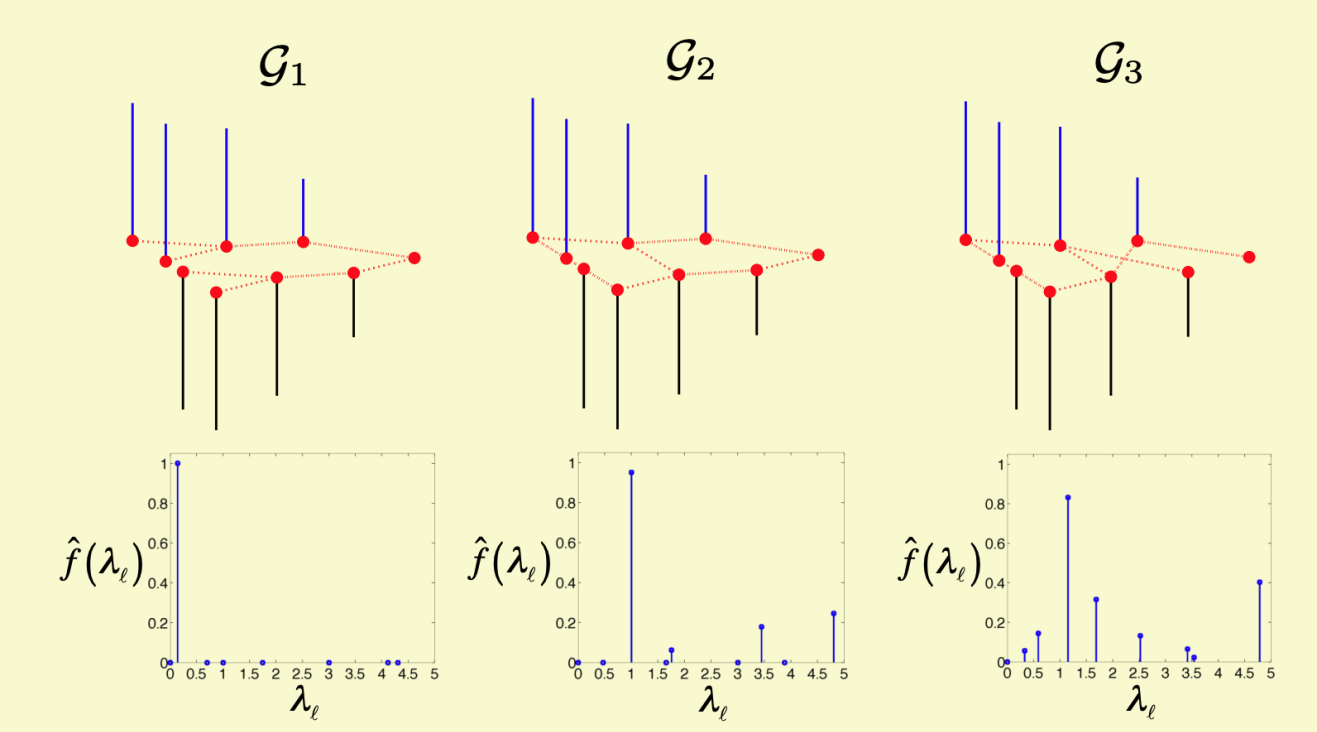


Figure: Here, $\mathbf{f}^T\mathbf{L}_1\mathbf{f} = 0.14$, $\mathbf{f}^T\mathbf{L}_2\mathbf{f} = 1.31$, $\mathbf{f}^T\mathbf{L}_3\mathbf{f} = 1.81$

GFT Will Be Used to Filter

- A filter on the graph will take in a signal and attenuate it according to a frequency response function.
- **Low-Pass Filter:** We filter or preserve only frequencies corresponding to eigenvalues below some threshold, λ_k. So, consider frequencies λ_b, with $\lambda_b < \lambda_k$
- **High-Pass Filters**: Preserve only frequencies corresponding to eigenvalues above some threshold, λ_k. So, consider frequencies λ_b, with $\lambda_b \geq \lambda_{k+1}$

A Simple Low-Pass Filter

Define some filter h as,

$$h : [0, \max(\mathbf{\Lambda})] \rightarrow [0, 1] \tag{11}$$

Assuming the cutoff is λ_k,
$h(x) > 0$, for $x < \lambda_k$ and $h(x) = 0$, otherwise

Defining Notation and Applying Filter to GFT

Define $h(\mathbf{\Lambda})$ as a diagonal matrix of eigenvalues with the filter applied. Based on what we computed with GFT, the filtered signal, $\hat{\mathbf{f}}_{filt}$ can be computed as,

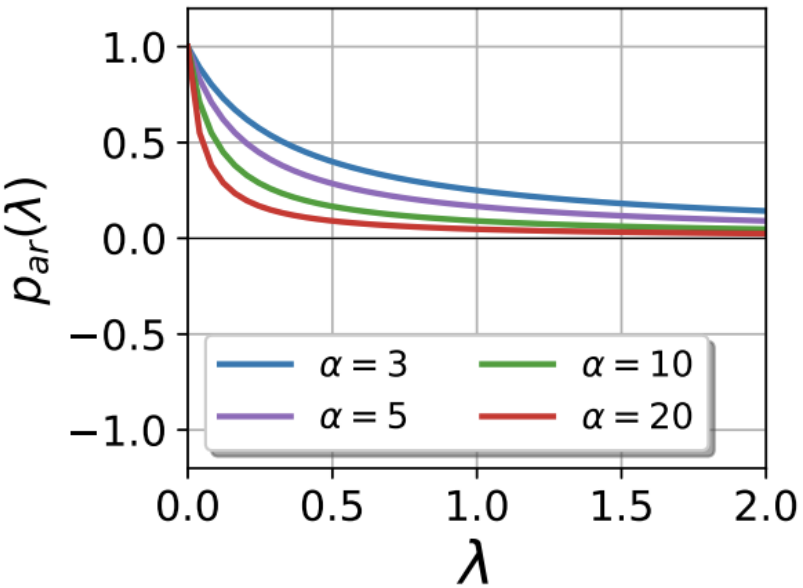
$$\hat{\mathbf{f}}_{filt} = h(\mathbf{\Lambda})\hat{\mathbf{f}} \tag{12}$$

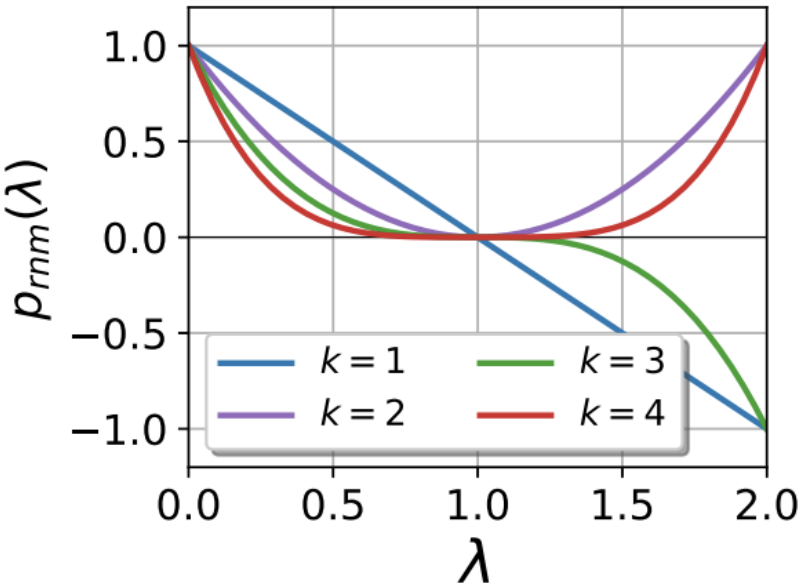
Applying a Filter in General

In general, you can filter an original signal, $\mathbf{f}$ in general as,

$$\underbrace{\boldsymbol{\Psi}(\mathbf{I} + \alpha\boldsymbol{\Lambda})^{-1}\boldsymbol{\Psi}^T}_{\text{Filtered Graph Laplacian}} \mathbf{f}. \tag{13}$$

Example Filters





(a) $p_{\text{ar}}(\lambda) = (1 + \alpha\lambda)^{-1}$ (b) $p_{\text{rnm}}(\lambda) = (1 - \lambda)^{k}$

Figure: from `https://openaccess.thecvf.com/content_CVPR_2019/papers/Li_Label_Efficient_Semi-Supervised_Learning_via_Graph_Filtering_CVPR_2019_paper.pdf`

Example in PyGSP

- Access PyGSP here, `https://pygsp.readthedocs.io/en/stable/tutorials/intro.html`

$$g(x) = \frac{1}{1+\tau x},$$

let's define and plot that low-pass filter:

```
>>> tau = 1
>>> def g(x):
...     return 1. / (1. + tau * x)
>>> g = filters.Filter(G, g)
>>>
>>> fig, ax = plt.subplots()
>>> g.plot(plot_eigenvalues=True, ax=ax)
>>> _ = ax.set_title('Filter frequency response')
```

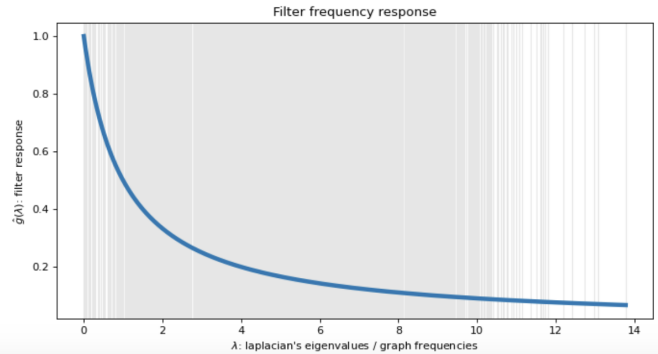
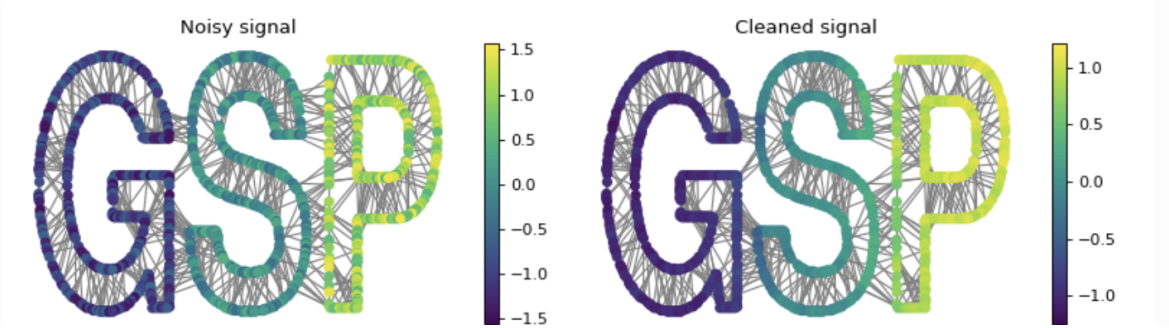


Figure: A simple filter for eigenvalues of **L**

Low-Pass Filtering a Noisy SIgnal

```
>>> s2 = g.filter(s)
>>>
>>> fig, axes = plt.subplots(1, 2, figsize=(10, 3))
>>> G.plot_signal(s, vertex_size=30, ax=axes[0])
>>> _ = axes[0].set_title('Noisy signal')
>>> axes[0].set_axis_off()
>>> G.plot_signal(s2, vertex_size=30, ax=axes[1])
>>> _ = axes[1].set_title('Cleaned signal')
>>> axes[1].set_axis_off()
>>> fig.tight_layout()
```



Linking Single Cell Data to External Information

A Question for You

If you were just given a bunch of cells and someone told you to find 200 cells that were associated with an experiment or a condition of interest, how would you choose those cells? What would cause you to trust that a particular cell was indeed representative of the experimental label?

Treatment as a Signal on a Graph

After creating a graph of cells, an indicator of treatment or control can be viewed as the signal on the graph. Interpretations of ‘signal‘ in relation to graph structure should help to inform treatment associated relative likelihood.

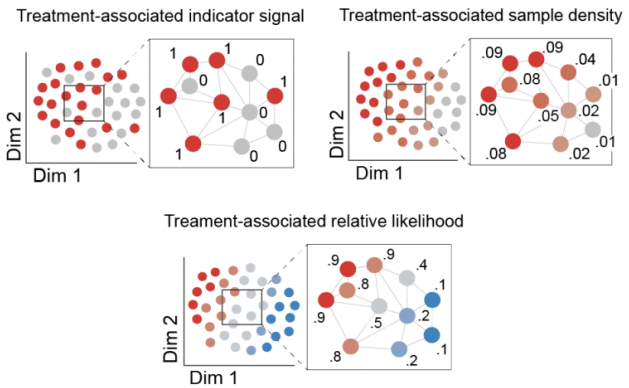


Figure: Burkhardt *et al.*, Nature Biotechnology. 2021.

What on Earth is a Cell-State

We have seen cell states already!

- Frequency $\rightarrow$ how many of a particular cell-type are there in a sample?
- Function $\rightarrow$ Which proteins are activated in a particular cell-type?

General Overview of the Steps of MELD

- Build a graph between cells based on gene or protein expression measurements
- **Graph Signals:** Experimental label (a binary indicator) is used to label each cell according to experimental condition
- Using GSP techniques, MELD filters biological and technical noise to look at how much the experimental signal of a cell matches the true experimental label. This quantifies how prototypical each cell is in its condition.
- Relate back to cell-types and features that differ between experimental conditions

RES vs EES

EES represents the enhanced experimental signal, in comparison to RES, which was the raw, binary signal.

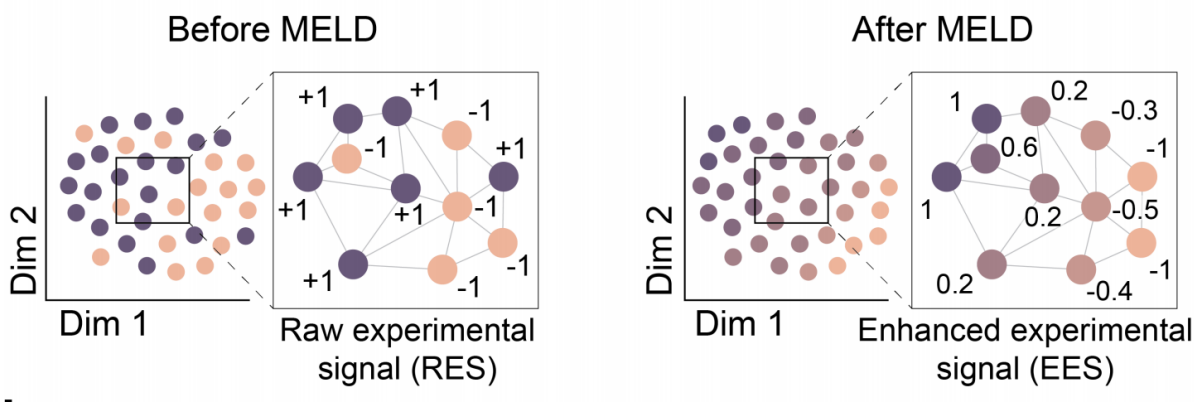


Figure: from Burkhardt *et al.*, Nature Biotechnology. 2021

Sources of Noise

- Cells with similar feature measurements are said to be in the same state (biologically)
- **High Frequency Noise** : High frequency noise is when the labels of neighboring cells are rapidly fluctuating.
- **Graph Fourier Transform** is used to study the frequency of a signal over an irregular domain, like a graph.

Incorporating these ideas into meld

- Define a latent variable $\mathbf{z}$ that gives a score for how prototypical a cell is for a specific experimental or clinical condition.
- $\mathbf{z}$ will be computed using low-pass graph filters.
- Defining more specific variables
  - $\mathbf{x}$ is the vector of original labels (RES) for each cell
  - $\mathbf{z}$ is the vector of enhanced experimental signals (EES) for each cell.

Visualizing **x** and **z**

The left is RES (**x**) and the right is EES (**z**). **z** is what is being optimized.

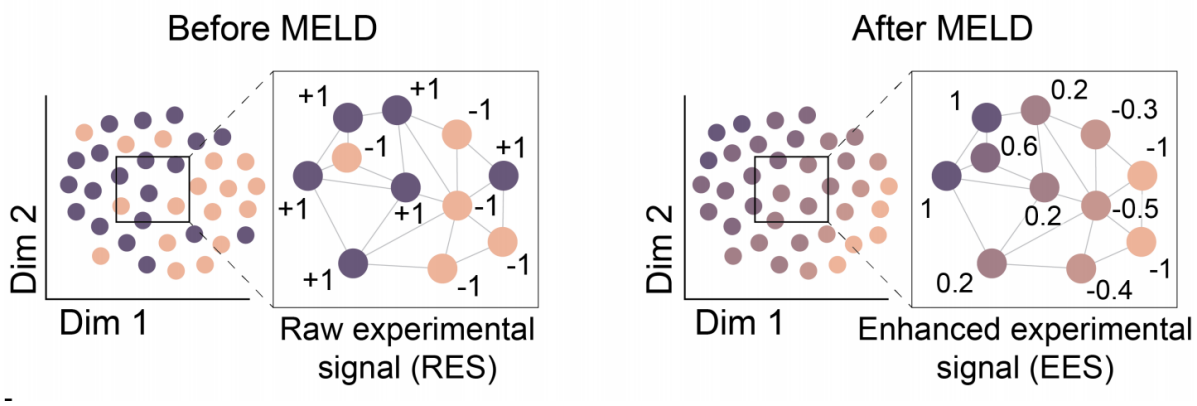


Figure: from Burkhardt *et al.*, Nature Biotechnology. 2021

MELD Optimization Problem

To find an appropriate $\mathbf{z}$, an optimization problem can be defined as,

$$\mathbf{y} = \underset{\mathbf{z}}{\operatorname{argmin}} \underbrace{\|\mathbf{x} - \mathbf{z}\|_2^2}_{\mathbf{a}} + \underbrace{\beta \mathbf{z}^T \mathcal{L} \mathbf{z}}_{\mathbf{b}} \tag{14}$$

Unpacking

z is the EES or Enhanced Experimental Signal

$$\mathbf{y} = \underset{\mathbf{z}}{\operatorname{argmin}} \underbrace{\|\mathbf{x} - \mathbf{z}\|_2^2}_{\mathbf{a}} + \underbrace{\beta \mathbf{z}^T \mathcal{L} \mathbf{z}}_{\mathbf{b}} \tag{15}$$

- The Laplacian Regularization (term b) initially encourages smoothness for an input graph signal, **x**
- (**a**) Term a represents reconstruction between **x** and **z**
- (**b**) Term b represents Laplacian regularization or a measure of smoothness on the graph. Recall this looks a lot like total variation.

$$\beta \mathbf{z}^T \mathcal{L} \mathbf{z} = \beta \sum_{i,j} A_{ij} (\mathbf{z}(i) - \mathbf{z}(j))^2 \tag{16}$$

Introducing the MELD Filter

They adjust the filer a bit as follows. The following allows also for a flexible notion of figure order, ρ,

$$\begin{aligned} \mathbf{y} &= \underset{z}{\text{argmin}} \|\mathbf{x} - \mathbf{z}\|_2^2 + \mathbf{z}^T \mathcal{L}_* \mathbf{z} \\ \text{where } \mathcal{L}_* &= [\beta \mathcal{L} - \alpha \mathbf{I}]^\rho \end{aligned} \tag{17}$$

Takeaway

They show that their Laplacian Regularization is a filter with the following frequency response,

$$h_{\mathrm{MELD}}(\lambda) = \frac{1}{1 + (\beta\lambda - \alpha)^{\rho}} \tag{18}$$

This was a lot to unpack. I recommend staring at the details (if you are interested) in
`https://www.biorxiv.org/content/10.1101/532846v1.full.pdf`

Filter Variety

Here are some experiments showing what parameters on the MELD filter will do to the frequency response, $h(\lambda)$.

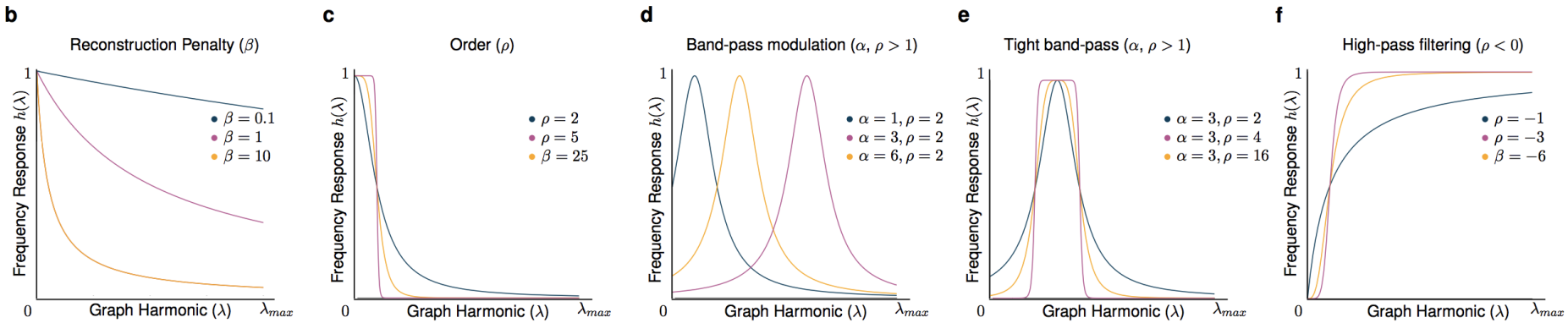


Figure: from Burkhardt *et al.*, Nature Biotechnology. 2021. Negative values of ρ, for example, can produce a high-pass filter.