Optimization Methods for Graph Learning

Prof. Daniel P. Palomar

ELEC5470/IEDA6100A - Convex Optimization
The Hong Kong University of Science and Technology (HKUST)
Fall 2020-21

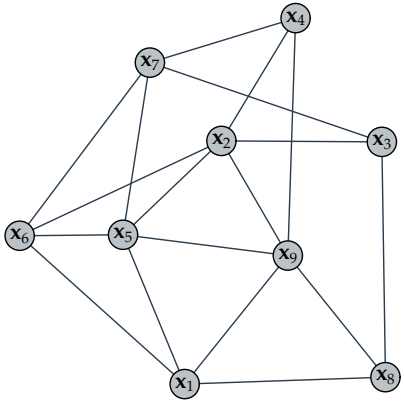
Outline

Outline

Graphical models

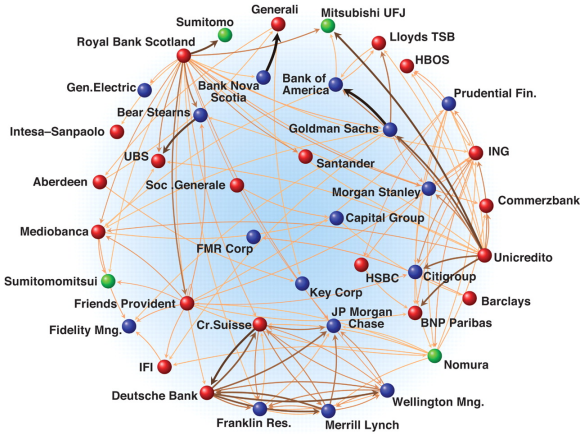
Graphical models are a way to represent knowledge:

- **nodes** correspond to the entities (variables);
- **edges** encode the relationships between entities (dependencies between the variables).



Examples

Financial graph: represents inter-dependencies of financial companies and the data are the economic indices (stock price, volume, etc.) of each entity.



Examples

Social graph: representa behavioral similarity/influence between people and the data are the online activities (tagging, liking, purchasing).

Graphical model importance

- Graphs are intuitive way of representing and visualising the **relationships** between entities.
- Graphs allow us to abstract out the **conditional independence** relationships and to answer questions like: "Is x_1 dependent on x_6 given that we know the value of x_8?" just by looking at the graph.
- Graph models constitute an effective **representation of data**, available across numerous domains in science and engineering.
- Graphs are widely used in a variety of applications in machine learning, graph CNN, graph signal processing, functional connectivity between brain regions, behavioral influence among groups of people, effect among stocks, etc.
- Graphs offer a **language** through which different **disciplines** can seamlessly interact with each other.
- It captures the actual geometry of data.
- It allows a visualization of high-dimensional data.

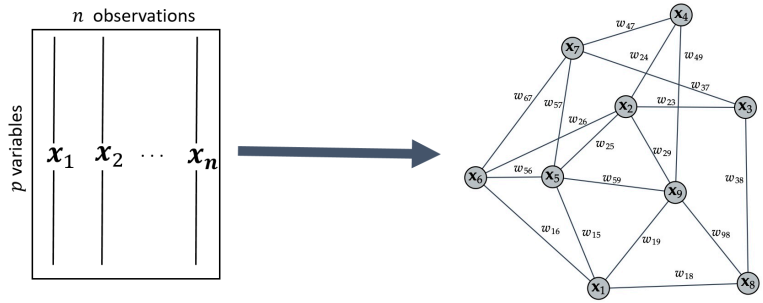
How to learn a graph?

- Graphical models are about having a graph representation that can encode **relationships** between entities.
- In many cases, the relationships between entities are **straightforward and direct**:
  - Are two people friends in a social network?
  - Are two researchers co-authors in a published paper?
- In many other cases, relationships are **not known and must be learned**:
  - Does one gene regulate the expression of others?
  - Which drug alters the pharmacologic effect of another drug?
- The **choice of graph representation** affects the subsequent analysis and eventually the performance of any graph-based algorithm.

👉 The goal is to learn a graph representation of data with specific properties (e.g., structures).

Graph learning from data

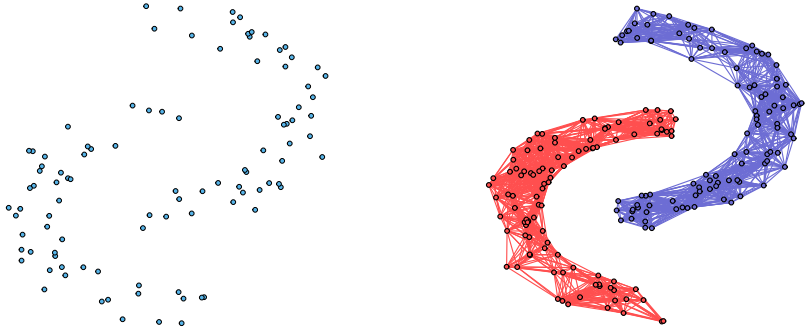
- Given a data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n] \in \mathbb{R}^{p \times n}$, each column $\mathbf{x}_i \in \mathbb{R}^p$ is a graph signal (one observation) and there are n observations.
- The goal is to obtain a graph representation of the data.



- A graph is a simple **mathematical structure** described by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, where
  - $\mathcal{V}$ contains the set of nodes $\mathcal{V} = \{1, 2, 3, \ldots, p\}$;
  - $\mathcal{E} = \{(1,2), (1,3), \ldots, (i,j), \ldots, (p, p-1)\}$ is the set of edges between pair of nodes (i,j);
  - the weight matrix $\mathbf{W}$ encodes the strength of the relationships.

Graph learning example

Example of transforming a dataset with points in $\mathbb{R}^2$ into a graph:



- Introductory references: (Mateos et al. 2019)[1] and (Dong et al. 2019)[2].

[1] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the dots," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 16–43, 2019.

[2] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Processing Magazine*, vol. 36, no. 3, pp. 44–63, 2019.

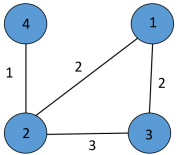
Outline

Graph and its matrix representation

- **Connectivity** matrix **C**, **Adjacency** matrix **W**, and **Laplacian** matrix **L**:

$$[\mathbf{C}]_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{E} \\ 0 & \text{if } (i,j) \notin \mathcal{E} \\ 0 & \text{if } i = j \end{cases}, \quad [\mathbf{W}]_{ij} = \begin{cases} w_{ij} & \text{if } (i,j) \in \mathcal{E} \\ 0 & \text{if } (i,j) \notin \mathcal{E} \\ 0 & \text{if } i = j \end{cases}, \quad [\mathbf{L}]_{ij} = \begin{cases} -w_{ij} & \text{if } (i,j) \in \mathcal{E} \\ 0 & \text{if } (i,j) \notin \mathcal{E} \\ \sum_{j=1}^{p} w_{ij} & \text{if } i = j \end{cases}$$

- Example: $\mathcal{V} = \{1,2,3,4\}$, $\mathcal{E} = \{(1,2),(1,3),(2,3),(2,4)\}$, and $\mathbf{W} = \{2,2,3,1\}$:



$$[\mathbf{C}]_{ij} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad [\mathbf{W}]_{ij} = \begin{bmatrix} 0 & 2 & 2 & 0 \\ 2 & 0 & 3 & 1 \\ 2 & 3 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad [\mathbf{L}]_{ij} = \begin{bmatrix} 4 & -2 & -2 & 0 \\ -2 & 6 & -3 & -1 \\ -2 & -3 & 5 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

Graph terminology

Graph matrices for a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$:

- **Weighted adjacency matrix** of a graph (or simply adjacency matrix), $\mathbf{W}$, is defined as $W_{ij} = w_{ij} \geq 0$. It is a symmetric matrix (undirected graph) and $W_{ii} = 0$ (no self-loops).
- **Connectivity matrix**: $\mathbf{C}$ is a particular case of the adjacency matrix with $0 - 1$ elements.
- The **degree matrix** is the diagonal matrix $\mathbf{D}$ that contains the degrees $d_1, \ldots, d_p$ along the diagonal:

$$\mathbf{D} = \text{Diag}\left(\mathbf{W1}\right).$$

  The degree d_i of the vertex i is defined as the row sum $d_i = \sum_{j \neq i} W_{i,j} = \sum_j w_{ij}$.
- **Laplacian** of a graph:

$$\mathbf{L} = \mathbf{D} - \mathbf{W}.$$

- **Normalized Laplacian** of a graph:

$$\mathbf{L}_{\text{norm}} = \mathbf{D}^{1/2}\mathbf{L}\mathbf{D}^{1/2} = \mathbf{I} - \mathbf{D}^{1/2}\mathbf{W}\mathbf{D}^{1/2}.$$

The Laplacian graph matrix

- The adjacency matrix $\mathbf{W}$ already contains all the graph information, so why do we need other graph matrices like the Laplacian matrix?
- The **Laplacian** $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} = \text{Diag}(\mathbf{W1})$, has a nice interesting physical meaning as well as many nice properties:
  - $\mathbf{L}$ is a symmetric and positive semidefinite matrix $\mathbf{L} \succeq \mathbf{0}$;
  - the number of zero eigenvalues denotes the number of connected components of the graph;
  - $\mathbf{L}$ is singular with eigenvector $\mathbf{1}$: $\mathbf{L1} = \mathbf{0}$.
- **Physical interpretation**: Denote with the vector $\mathbf{x} = (x_1, \ldots, x_p)$ the value of some quantity on all the p nodes. The Laplacian measures the **smoothness** or variance of that vector weighted with the graph weights:

$$\mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i,j} w_{ij}(x_i - x_j)^2.$$

Proof:

$$\mathbf{x}^T\mathbf{L}\mathbf{x} = \mathbf{x}^T\mathbf{D}\mathbf{x} - \mathbf{x}^T\mathbf{W}\mathbf{x} = \sum_i d_i x_i^2 - \sum_{i,j} w_{ij}x_i x_j = \sum_{i,j} w_{ij}x_i^2 - \sum_{i,j} w_{ij}x_i x_j.$$

Outline

Learning graphs from data

We will consider different ways to learn the graph **adjacency matrix**, $\mathbf{W}$, or **Laplacian matrix**, $\mathbf{L}$, with input either the data matrix $\mathbf{X}$ or the sample covariance matrix $\mathbf{S}$.

- **Similarity function based** methods for adjacency $\mathbf{W}$ (with input the data matrix $\mathbf{X}$). Basically, two nodes i and j are connected based on some similarity function, leading to:
  - thresholded Euclidean distance graph;
  - Gaussian graph;
  - thresholded Gaussian graph;
  - k-nearest neighbors (k-NN) graph;
  - feature correlation graph;
  - self-tuned Gaussian graph.

Learning graphs from data

- **Smooth signal based** methods for Laplacian $\mathbf{L}$:
  - graphs from smooth signals;
  - closest k-connected graph.

- **i.i.d. model based** methods for Laplacian $\mathbf{L}$ (with input the sample covariance matrix $\mathbf{S}$):
  - covariance matrix (with and without market factor, which applies to all the methods);
  - correlation matrix;
  - precision matrix;
  - graphical LASSO (GLASSO);
  - Laplacian-structured GLASSO;
  - Laplacian with spectral constraints (e.g., for k-connected graph);
  - shift operator based.

Outline

Basic similarity measures

Write the data matrix in terms of the node signals $\mathbf{x}_i \in \mathbb{R}^n$ along the rows: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p]^T \in \mathbb{R}^{p \times n}$.

To choose the edges and weights to form the **adjacency matrix W**, either weighted or 0-1 connectivity matrix, we can use different methods:

- **Thresholded Euclidean distance graph**: nodes i and j are connected ($w_{ij} = 1$) if the corresponding signals satisfy $\|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \gamma$ (where γ is a threshold); otherwise not connected ($w_{ij} = 0$).
- **Gaussian graph**: set every pair of points $i \neq j$ as connected with the following weights:

$$w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right),$$

  where σ^2 controls the size of the neighborhood.
- **Thresholded Gaussian graph**: we can combine the previous two graph constructions, i.e., set every pair of points $i \neq j$ satisfying $\|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \gamma$ with Gaussian weights; otherwise not connected ($w_{ij} = 0$).

Basic similarity measures

- k-**nearest neighbors (k-NN) graph**: nodes i and j are connected ($w_{ij} = 1$) if $\mathbf{x}_i$ is one of the k closest points (Euclidean distance) to $\mathbf{x}_j$ or viceversa; otherwise not connected ($w_{ij} = 0$).
- **Feature correlation graph**: simply use the pairwise feature correlation for $i \neq j$:

$$w_{ij} = \mathbf{x}_i^T \mathbf{x}_j.$$

- **Self-tuned Gaussian graph**: the Gaussian graph tends to be too densely connected, this approach automatically chooses a different σ for each point; in particular, it normalizes the distance from each node i to the other nodes with the distance to its k-NN:

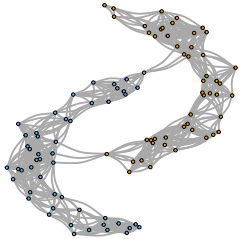
$$w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_i \sigma_j}\right),$$

where σ_i denotes the distance between the ith node and its k-NN.
Basic reference: (Manor and Perona 2004)[3].

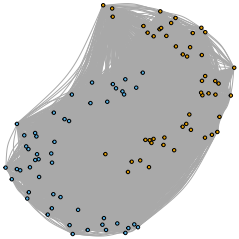
[3] L. Z. Manor and P. Perona, "Self-tuning spectral clustering," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2004.

Graphs from basic constructions

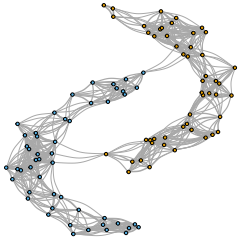
Thresholded distance 0−1 graph (two−moon data)



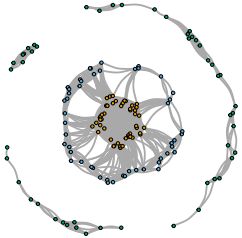
Gaussian graph (two−moon data)



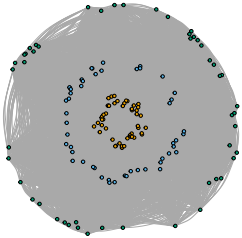
Thresholded Gaussian graph (two−moon data)



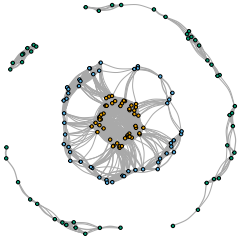
Thresholded distance 0−1 graph (three−circle data)



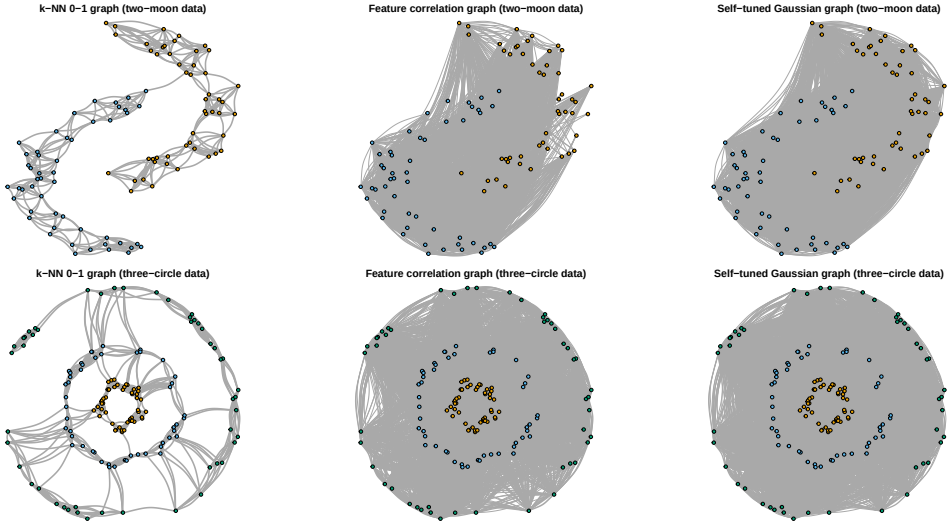
Gaussian graph (three−circle data)



Thresholded Gaussian graph (three−circle data)



Graphs from basic constructions



Outline

Learning graphs from data: Smooth signals

- Previously, we have learned different basic ways to construct the **adjacency matrix W**, either weighted or a 0-1 connectivity matrix, from a set of data points $\{\mathbf{x}_i\}_{i=1}^n$, where each p-dimensional vector $\mathbf{x}_i = (x_{1i}, \ldots, x_{pi})$ contains the signal from the p nodes of the graph.
- In particular, we have considered the following **similarity function based** methods:
  - thresholded Euclidean distance graph;
  - Gaussian graph;
  - thresholded Gaussian graph;
  - k-nearest neighbors (k-NN) graph;
  - feature correlation graph;
  - self-tuned Gaussian graph.
- We will now consider more sophisticated **smooth signal based** methods to learn the graph Laplacian matrix **L**:
  - graphs from smooth signals;
  - closest k-connected graph.

Learning graphs from smooth signals

- Recall that a measure of smoothness or variance of a graph-signal $\mathbf{x} \in \mathbb{R}^p$ is

$$\mathbf{x}^T\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i,j} w_{ij}(x_i - x_j)^2.$$

- Given p signals, each containing n observations along the rows, in the data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p]^T \in \mathbb{R}^{p\times n}$, we can measure its smoothness over the graph $\mathcal{G}$ as $\text{Tr}(\mathbf{X}^T\mathbf{L}\mathbf{X}) = \frac{1}{2}\sum_{i,j} w_{ij}\|\mathbf{x}_i - \mathbf{x}_j\|^2$.
- When the graph $\mathcal{G}$ is not available, we can learn it from the data $\mathbf{X}$ by finding the graph weights that minimize the variance term combined with some regularization term:

$$\underset{\mathbf{L}}{\text{minimize}} \quad \text{Tr}(\mathbf{X}^T\mathbf{L}\mathbf{X}) + \gamma h(\mathbf{L}).$$

  - Smaller distance $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ between data points $\mathbf{x}_i$ and $\mathbf{x}_j$ will force to learn a graph with larger affinity value w_{ij}, and vice versa.
  - Higher values of weight w_{ij} will imply the signals $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar and, hence, strongly connected.
  - $h(\mathbf{L})$ is a regularization function, e.g., $\|\mathbf{L}\|_1$, $\|\mathbf{L}\|_F^2$, and $-\text{logdet}(\mathbf{L})$.

Learning graphs from smooth signals

- The graph learning formulation from smooth signals can be formulated in convex form in terms of either the adjacency matrix $\mathbf{W}$ or the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} = \text{Diag}(\mathbf{d})$ and $\mathbf{d} = \mathbf{W}\mathbf{1}$ is the degree vector of the nodes.
- **Formulation in terms of Laplacian matrix**:

$$\begin{array}{ll} \underset{\mathbf{L}\succeq\mathbf{0}}{\text{minimize}} & \text{Tr}(\mathbf{X}^T\mathbf{L}\mathbf{X}) + \frac{\gamma}{2}\|\mathbf{L}\|_{F,\text{off}}^2 \\ \text{subject to} & \text{diag}(\mathbf{L}) = \mathbf{1} \\ & \mathbf{L}\mathbf{1} = \mathbf{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j \end{array}$$

where

- the regularization term $\|\mathbf{L}\|_{F,\text{off}}^2$ controls the energy of the off-diagonal elements;
- the constraint $\mathbf{d} = \text{diag}(\mathbf{L}) = \mathbf{1}$ controls the degrees of the nodes (to get a balanced graph); and
- we have the usual Laplacian constraints $\mathbf{L}\mathbf{1} = \mathbf{0}$ and $L_{ij} = L_{ji} \leq 0$ for $i \neq j$.

Learning graphs from smooth signals

- **Formulation in terms of adjacency matrix**:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \sum_{i,j=1}^{p} w_{ij}\|\mathbf{x}_i - \mathbf{x}_j\|^2 + \gamma \sum_{i,j=1}^{p} w_{ij}^2 \\ \text{subject to} & \sum_{j=1}^{p} w_{ij} = 1, \qquad \forall i \\ & w_{ii} = 0, \quad w_{ij} = w_{ji} \geq 0, \quad \forall i \neq j. \end{array}$$

- Defining $Z_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2$, we can rewrite the problem compactly:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \text{Tr}(\mathbf{WZ}) + \gamma\|\mathbf{W}\|_F^2 \\ \text{subject to} & \mathbf{W1} = \mathbf{1} \\ & \text{diag}(\mathbf{W}) = \mathbf{0}, \quad \mathbf{W} = \mathbf{W}^T \geq \mathbf{0}. \end{array}$$

  - With $\gamma = 0$ we would get just one connection per node!
  - With $\gamma \to \infty$ we get instead a fully dense graph (Cauchy-Schwartz).
  - So the term $\|\mathbf{W}\|_F^2$ makes the solution nonsparse!
- Both formulations (in terms of adjacency matrix or Laplacian matrix) are convex and can be easily solved with a solver. Interestingly, we can develop a simple tailored algorithm.

Learning graphs from smooth signals

- If we ignore the symmetry constraint in the adjacency matrix $\mathbf{W} = \mathbf{W}^T$, then the problem can be nicely solved separately for each row/column $\mathbf{w}_i$ as in (Nie et al. 2016)[4].
- Note that at a later stage we will have to symmetrize it:

$$\mathbf{W} \leftarrow \frac{1}{2}(\mathbf{W} + \mathbf{W}^T).$$

- The problem of the ith row is

$$\begin{array}{ll} \underset{\mathbf{w}_i}{\text{minimize}} & \sum_{j=1}^{p} w_{ij} z_{ji} + \gamma \sum_{j=1}^{p} w_{ij}^2 \\ \text{subject to} & \sum_{j=1}^{p} w_{ij} = 1,\ w_{ii} = 0,\ w_{ij} \geq 0 \end{array}$$

- More compactly

$$\begin{array}{ll} \underset{\mathbf{w}_i}{\text{minimize}} & \mathbf{z}_i^T \mathbf{w}_i + \gamma \|\mathbf{w}_i\|^2 \\ \text{subject to} & \mathbf{w}_i^T \mathbf{1} = 1,\ w_{ii} = 0,\ \mathbf{w}_i \geq \mathbf{0}. \end{array}$$

[4] F. Nie, X. Wang, M. Jordan, and H. Huang, "The constrained Laplacian rank algorithm for graph-based clustering," in *Proc. of the Thirtieth Conference on Artificial Intelligence (AAAI)*, Phoenix, Arizona, USA, 2016, pp. 1969–1976.

Learning graphs from smooth signals

- We can finally write our problem as

$$\begin{array}{ll} \underset{\mathbf{w}_i}{\text{minimize}} & \frac{1}{2}\|\mathbf{w}_i + \frac{\mathbf{z}_i}{2\gamma}\|^2 \\ \text{subject to} & \mathbf{w}_i^T\mathbf{1} = 1,\ w_{ii} = 0,\ \mathbf{w}_i \geq \mathbf{0}. \end{array}$$

- The Lagrangian is

$$\mathcal{L}(\mathbf{w}_i; \eta_i, \beta_i) = \frac{1}{2}\left\|\mathbf{w}_i + \frac{\mathbf{z}_i}{2\gamma}\right\|^2 - \eta_i(\mathbf{w}_i^T\mathbf{1} - 1) - \beta_i^T\mathbf{w}_i$$

  where η_i and $\beta_i \in \mathbb{R}^p$ are the Lagrangian multipliers with $\beta_{ji} \geq 0,\ \forall j \neq i$.

- From the KKT optimality conditions, the optimal solution (defining $z_{ii} = \infty$ so that $w_{ii} = 0$) is

$$\mathbf{w}_i = \left(\eta_i - \frac{\mathbf{z}_i}{2\gamma}\right)^+$$

  where η is found so that the constraint $\mathbf{w}_i^T\mathbf{1} = 1$ is satisfied.

Technical details: Derivation of solution

- To derive the optimal solution, setting the gradient of the Lagrangian w.r.t. $\mathbf{w}_i$ to zero gives

$$\mathbf{w}_i + \frac{\mathbf{z}_i}{2\gamma} - \eta_i \mathbf{1} - \beta_i = \mathbf{0}.$$

- Now, from the complementary slackness condition $w_{ji}\beta_{ji} = 0$, we get that, for $j \neq i$:
  - if $w_{ji} > 0$, then $w_{ji} = -\frac{z_{ji}}{2\gamma} + \eta_i$ and $-\frac{z_{ji}}{2\gamma} + \eta_i > 0$
  - if $w_{ji} = 0$, then $-\frac{z_{ji}}{2\gamma} + \eta_i = -\beta_{ji} \leq 0$
- In other words:
  - if $\eta_i - \frac{z_{ji}}{2\gamma} > 0$, then $w_{ji} = -\frac{z_{ji}}{2\gamma} + \eta_i$
  - if $\eta_i - \frac{z_{ji}}{2\gamma} \leq 0$, then $w_{ji} = 0$.
- More compactly (defining $z_{ii} = \infty$ so that $w_{ii} = 0$):

$$\mathbf{w}_i = \left(\eta_i - \frac{\mathbf{z}_i}{2\gamma}\right)^+.$$

Technical details: Obtaining sparsity via γ

- Let's find a choice of γ so that $\|\mathbf{w}_i\|_0 = m$ with $m \ll p$ (i.e., so that each node has exactly m neighbors). In fact, we will need a different γ for each $\mathbf{w}_i$.
- Without loss of generality, suppose $z_{1i}, z_{2i}, \ldots, z_{pi}$ are ordered in increasing order.
- The requirement $\|\mathbf{w}_i\|_0 = m$ implies $w_{mi} > 0$ and $w_{m+1,i} = 0$. Therefore, we have

$$\eta_i - \frac{z_{mi}}{2\gamma} > 0 \quad \text{and} \quad \eta_i - \frac{z_{m+1,i}}{2\gamma} \leq 0$$

- Combining the solution of $\mathbf{w}_i$ with the constraint $\mathbf{w}_i^T \mathbf{1} = 1$, we get

$$\sum_{j=1}^{m} \left(\eta_i - \frac{z_{ji}}{2\gamma} \right) = 1 \implies \eta_i = \frac{1}{m} + \frac{1}{2m\gamma} \sum_{j=1}^{m} z_{ji}.$$

- This leads to following inequality for γ:

$$\frac{m}{2} z_{mi} - \frac{1}{2} \sum_{j=1}^{m} z_{ji} < \gamma \leq \frac{m}{2} z_{m+1,i} - \frac{1}{2} \sum_{j=1}^{m} z_{ji}.$$

Technical details: Obtaining sparsity via γ

- Therefore, to obtain an optimal solution $\mathbf{w}_i$ with exactly m nonzero values ($\|\mathbf{w}_i\|_0 = m$), the maximal γ is

$$\gamma = \frac{m}{2} z_{m+1,i} - \frac{1}{2} \sum_{j=1}^{m} z_{ji}.$$

- Combining the previous results, we get

$$w_{ji} = \begin{cases} \frac{z_{m+1,i} - z_{ji}}{m z_{m+1,i} - \sum_{h=1}^{m} z_{hi}}, & j \leq m \\ 0, & j > m. \end{cases}$$

- We now drop the increasing ordering assumption on z_{ji} and we denote by $z_{(m+1),i}$ the $(m+1)$-th smallest element of $z_{1,i}, \ldots, z_{p,i}$.
- The final solution is written as:

$$\tilde{\mathbf{w}}_i = \left(1 - \frac{\mathbf{z}_i}{z_{(m+1),i}}\right)^+$$

$$\mathbf{w}_i = \tilde{\mathbf{w}}_i / \mathbf{1}^T \tilde{\mathbf{w}}_i.$$

Learning graphs from smooth signals revisited

- Recall the formulation for the graph learning from smooth signals:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \frac{1}{2}\text{Tr}(\mathbf{WZ}) + \frac{\gamma}{2}\|\mathbf{W}\|_F^2 \\ \text{subject to} & \mathbf{W1} = \mathbf{1} \\ & \text{diag}(\mathbf{W}) = \mathbf{0}, \quad \mathbf{W} = \mathbf{W}^T \geq \mathbf{0}. \end{array}$$

- If we move the hard constraint on the degrees of the nodes $\mathbf{W1} = \mathbf{1}$ to the objective as a penalty term to avoid the trivial solution $\mathbf{W} = \mathbf{0}$ (with vanishing degrees), we obtain (Kalofolias 2016)[5]:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \frac{1}{2}\text{Tr}(\mathbf{WZ}) - \alpha \mathbf{1}^T \log(\mathbf{W1}) + \frac{\beta}{2}\|\mathbf{W}\|_F^2 \\ \text{subject to} & \text{diag}(\mathbf{W}) = \mathbf{0}, \quad \mathbf{W} = \mathbf{W}^T \geq \mathbf{0}. \end{array}$$

- This formulation is convex and can be easily solved with a solver. Interestingly, we can again develop a simple tailored algorithm.

[5] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2016, pp. 920–929.

Learning graphs from smooth signals revisited

- If we ignore the symmetry constraint in the adjacency matrix $\mathbf{W} = \mathbf{W}^T$, then the problem can be nicely solved separately for each row/column $\mathbf{w}_i$:

$$\begin{array}{ll} \underset{\mathbf{w}_i}{\text{minimize}} & \frac{1}{2}\mathbf{z}_i^T\mathbf{w}_i - \alpha \log(\mathbf{1}^T\mathbf{w}_i) + \frac{\beta}{2}\|\mathbf{w}_i\|^2 \\ \text{subject to} & w_{ii} = 0,\ \mathbf{w}_i \geq \mathbf{0}. \end{array}$$

- Note that at a later stage we will have to symmetrize it:

$$\mathbf{W} \leftarrow \frac{1}{2}(\mathbf{W} + \mathbf{W}^T).$$

- From the KKT optimality conditions, the optimal solution (defining $z_{ii} = \infty$ so that $w_{ii} = 0$) is

$$\mathbf{w}_i = \left(\frac{\alpha/\beta}{t_i} - \frac{\mathbf{z}_i}{2\beta}\right)^+$$

where t_i is found so that $\mathbf{1}^T\mathbf{w}_i = t_i$.

Technical details: Derivation of solution

- The Lagrangian of the problem is

$$\mathcal{L}(\mathbf{w}_i; \lambda_i) = \frac{1}{2}\mathbf{z}_i^T\mathbf{w}_i - \alpha \log(\mathbf{1}^T\mathbf{w}_i) + \frac{\beta}{2}\|\mathbf{w}_i\|^2 - \boldsymbol{\lambda}_i^T\mathbf{w}_i$$

  where $\lambda_i \in \mathbb{R}^p$ is the Lagrangian multipliers with $\lambda_{ji} \geq 0,\ \forall j \neq i$.
- Setting the gradient of the Lagrangian w.r.t. $\mathbf{w}_i$ to zero gives $\frac{1}{2}\mathbf{z}_i - \frac{\alpha}{\mathbf{1}^T\mathbf{w}_i}\mathbf{1} + \beta\mathbf{w}_i - \boldsymbol{\lambda}_i = \mathbf{0}$.
- Now, from the complementary slackness condition $w_{ji}\lambda_{ji} = 0$, we get that, for $j \neq i$:
  - if $w_{ji} > 0$, then $w_{ji} = \frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} - \frac{z_{ji}}{2\beta}$ and $\frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} > \frac{z_{ji}}{2\beta}$
  - if $w_{ji} = 0$, then $\frac{z_{ji}}{2} - \frac{\alpha}{\mathbf{1}^T\mathbf{w}_i} = \lambda_{ji} \geq 0$
- In other words:
  - if $\frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} > \frac{z_{ji}}{2\beta}$, then $w_{ji} = \frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} - \frac{z_{ji}}{2\beta}$
  - if $\frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} \leq \frac{z_{ji}}{2\beta}$, then $w_{ji} = 0$.
- More compactly (defining $z_{ii} = \infty$ so that $w_{ii} = 0$): $\mathbf{w}_i = \left(\frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} - \frac{\mathbf{z}_i}{2\beta}\right)^+$.

Technical details: Obtaining sparsity via α and β

- Let's find a choice of α and β so that $\|\mathbf{w}_i\|_0 = m$ with $m \ll p$ (i.e., so that each node has exactly m neighbors). In fact, we will need a different α and β for each $\mathbf{w}_i$.
- Without loss of generality, suppose $z_{1i}, z_{2i}, \ldots, z_{pi}$ are ordered in increasing order.
- The requirement $\|\mathbf{w}_i\|_0 = m$ implies $w_{m,i} > 0$ and $w_{m+1,i} = 0$. Therefore, we have

$$\frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} > \frac{z_{m,i}}{2\beta} \quad \text{and} \quad \frac{\alpha/\beta}{\mathbf{1}^T\mathbf{w}_i} \leq \frac{z_{m+1,i}}{2\beta}.$$

- Denoting $\bar{\alpha} = \alpha/\beta$, we can write it as

$$\frac{z_{m,i}}{2\bar{\alpha}}\mathbf{1}^T\mathbf{w}_i < \beta \leq \frac{z_{m+1,i}}{2\bar{\alpha}}\mathbf{1}^T\mathbf{w}_i.$$

- Therefore, to obtain an optimal solution $\mathbf{w}_i$ with exactly m nonzero values ($\|\mathbf{w}_i\|_0 = m$), the maximal β is

$$\beta = \frac{z_{m+1,i}}{2\bar{\alpha}}\mathbf{1}^T\mathbf{w}_i.$$

Technical details: Obtaining sparsity via α and β

- Combining the previous results, the optimal solution is

$$w_{ji} = \frac{\bar{\alpha}}{\sum_{j=1}^{m} w_{ji}} \left(1 - \frac{z_{ji}}{z_{m+1,i}}\right)^+$$

- Now we denote $t_i = \sum_{j=1}^{m} w_{ji}$ and write

$$w_{ji} = \frac{\bar{\alpha}}{t_i} \left(1 - \frac{z_{ji}}{z_{m+1,i}}\right)^+$$

which leads to

$$t_i = \sum_{j=1}^{m} w_{ji} = \frac{\bar{\alpha}}{t_i} \sum_{j=1}^{p} \left(1 - \frac{z_{ji}}{z_{m+1,i}}\right)^+$$

or

$$t_i = \sqrt{\bar{\alpha} \sum_{j=1}^{p} \left(1 - \frac{z_{ji}}{z_{m+1,i}}\right)^+}.$$

Technical details: Obtaining sparsity via α and β

- The final solution for $\mathbf{w}_i$ can be compactly written as

$$\mathbf{w}_i = \frac{\bar{\alpha}}{t_i}\left(1 - \frac{\mathbf{z}_i}{z_{m+1,i}}\right)^+$$

with

$$t_i = \sqrt{\bar{\alpha}\sum_{j=1}^{p}\left(1 - \frac{z_{ji}}{z_{m+1,i}}\right)^+}.$$

- We now drop the increasing ordering assumption on z_{ji} and we denote by $z_{(m+1),i}$ the $(m+1)$-th smallest element of $z_{1,i}, \ldots, z_{p,i}$.
- The final solution is written as

$$\tilde{\mathbf{w}}_i = \bar{\alpha}\left(1 - \frac{\mathbf{z}_i}{z_{(m+1),i}}\right)^+$$

$$\mathbf{w}_i = \tilde{\mathbf{w}}_i / \sqrt{\mathbf{1}^T \tilde{\mathbf{w}}_i}.$$

Learning graphs from smooth signals revisited

- Recall the original convex formulation:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \frac{1}{2}\text{Tr}(\mathbf{WZ}) - \alpha \mathbf{1}^T \log(\mathbf{W1}) + \frac{\beta}{2}\|\mathbf{W}\|_F^2 \\ \text{subject to} & \text{diag}(\mathbf{W}) = \mathbf{0}, \quad \mathbf{W} = \mathbf{W}^T \geq \mathbf{0}. \end{array}$$

- Instead of solving it with a solver or with the previously derived columnwise closed-form solution (which ignores the symmetry constraint), we will attempt to derive a joint closed-form solution.
- Instead of dealing with matrix $\mathbf{W}$, denote with $\mathbf{w}$ the off-diagonal elements stacked in a vector.
- Define $\mathbf{z}$ such that $\text{Tr}(\mathbf{WZ}) = \mathbf{z}^T\mathbf{w}$.
- Also, define $\mathbf{G}$ to extract the row sums of $\mathbf{W}$: $\mathbf{W1} = \mathbf{G}^T\mathbf{w}$
- Note that $\frac{1}{2}\|\mathbf{W}\|_F^2 = \|\mathbf{w}\|^2$.
- The constraints $\text{diag}(\mathbf{W}) = \mathbf{0}$ and $\mathbf{W} = \mathbf{W}^T$ are unnecessary.

Learning graphs from smooth signals revisited

- The problem can be rewritten in terms of $\mathbf{w}$ as (for aesthetics we change two $1/2$ factors)

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{minimize}} & \mathbf{z}^T\mathbf{w} - \alpha \mathbf{1}^T \log(\mathbf{G}^T\mathbf{w}) + \frac{\beta}{2}\|\mathbf{w}\|^2 \\ \text{subject to} & \mathbf{w} \geq \mathbf{0}. \end{array}$$

- The Lagrangian of the problem is

$$\mathcal{L}(\mathbf{w}; \lambda) = \mathbf{z}^T\mathbf{w} - \alpha \mathbf{1}^T \log(\mathbf{G}^T\mathbf{w}) + \frac{\beta}{2}\|\mathbf{w}\|^2 - \boldsymbol{\lambda}^T\mathbf{w}$$

  where $\lambda \in \mathbb{R}^p_+$.

- Setting the gradient of the Lagrangian w.r.t. $\mathbf{w}$ to zero gives

$$\mathbf{z} - \alpha \sum_{j=1}^{p} \frac{\mathbf{g}_j}{\mathbf{g}_j^T\mathbf{w}} + \beta\mathbf{w} - \boldsymbol{\lambda} = \mathbf{0}.$$

Learning graphs from smooth signals revisited

- Now, from the complementary slackness condition $w_i \lambda_i = 0$, we get:
  - if $w_i > 0$, then $w_i = \frac{\alpha}{\beta} \sum_{j=1}^{p} \frac{g_{ij}}{\mathbf{g}_j^T \mathbf{w}} - \frac{z_i}{\beta}$ and $\frac{\alpha}{\beta} \sum_{j=1}^{p} \frac{\mathbf{g}_{ij}}{\mathbf{g}_j^T \mathbf{w}} > \frac{z_i}{\beta}$
  - if $w_i = 0$, then $z_i - \alpha \sum_{j=1}^{p} \frac{g_{ij}}{\mathbf{g}_j^T \mathbf{w}} = \lambda_i \geq 0$.
- In other words:
  - if $\frac{\alpha}{\beta} \sum_{j=1}^{p} \frac{\mathbf{g}_{ij}}{\mathbf{g}_j^T \mathbf{w}} > \frac{z_i}{\beta}$, then $w_i = \frac{\alpha}{\beta} \sum_{j=1}^{p} \frac{g_{ij}}{\mathbf{g}_j^T \mathbf{w}} - \frac{z_i}{\beta}$
  - if $\frac{\alpha}{\beta} \sum_{j=1}^{p} \frac{g_{ij}}{\mathbf{g}_j^T \mathbf{w}} \leq \frac{z_i}{\beta}$, then $w_{ji} = 0$.
- More compactly (defining $z_{ii} = \infty$ so that $w_{ii} = 0$):

$$\mathbf{w} = \left(\frac{\alpha}{\beta} \sum_{j=1}^{p} \frac{\mathbf{g}_j}{\mathbf{g}_j^T \mathbf{w}} - \frac{\mathbf{z}}{\beta} \right)^+ .$$

- This expression is a fixed-point equation in $\mathbf{w}$, which is not easy to compute in practice, although algorithms can be devised.

Learning graphs from smooth signals revisited

- Denoting explicitly the degrees of the nodes by $d_i = \mathbf{g}_i^T \mathbf{w}$, we can write a double fixed-point equation in $(\mathbf{d}, \mathbf{w})$:

$$\mathbf{d} = \mathbf{G}^T \mathbf{w}$$

$$\mathbf{w} = \left(\frac{\alpha}{\beta} \mathbf{G} \mathbf{d}^{-1} - \frac{\mathbf{z}}{\beta} \right)^+ .$$

- We can also write the fixed-point equation in terms of $\mathbf{d}$ only:

$$\mathbf{d} = \mathbf{G}^T \left(\frac{\alpha}{\beta} \mathbf{G} \mathbf{d}^{-1} - \frac{\mathbf{z}}{\beta} \right)^+$$

- Alternatively, one would devise a gradient projection method:

$$\mathbf{w}^{k+1} = \left(\mathbf{w}^k - \mu^k \left(\mathbf{z} - \alpha \mathbf{G} (\mathbf{G}^T \mathbf{w}^k)^{-1} + \beta \mathbf{w}^k \right) \right)^+$$

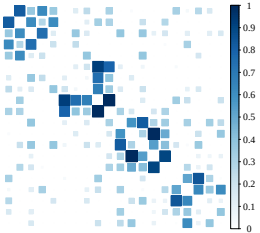
where μ^k is the step size.

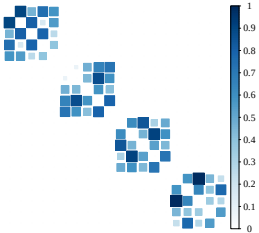
- To avoid numerical issues in some degree d_i becoming zero, one can use the heuristic:

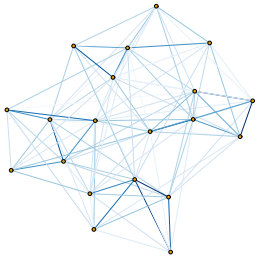
$$\mathbf{w}^{k+1} = \left(\mathbf{w}^k - \mu^k \left(\mathbf{z} - \alpha \mathbf{G} (\mathbf{G}^T \mathbf{w}^k + \epsilon)^{-1} + \beta \mathbf{w}^k \right) \right)^+$$

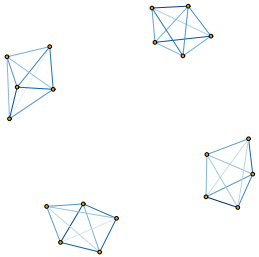
Learning a k-connected graph

Learn a clean graph from a noisy graph by imposing k-connected structure:









Eigenvalue property of Laplacian matrix

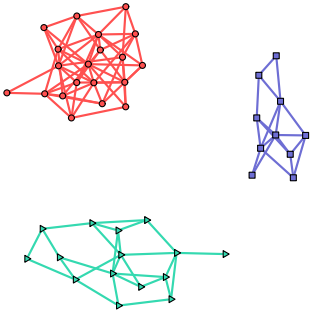
- Consider the eigenvalue decomposition of the Laplacian matrix:

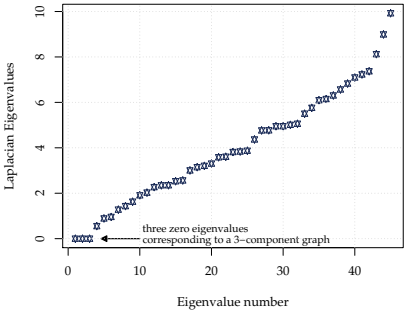
$$\mathbf{L} = \mathbf{U}\text{Diag}(\lambda_1, \lambda_2, \ldots, \lambda_p)\mathbf{U}^T,$$

where $\mathbf{U}$ contains the eigenvectors columnwise and $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_p$ are the eigenvalues in increasing order.

- For a k-connected graph, the k smallest eigenvalues are zero:

$$\lambda_1 = \cdots = \lambda_k = 0.$$





Learning a k-connected graph

- **Goal**: Given an initial noisy adjacency matrix $\mathbf{W}_0$, infer a k-connected graph.
- In other words, we want to learn $\mathbf{W}$ as close as possible to $\mathbf{W}_0$ but satisfying the property of a k-connected graph: zero k smallest eigenvalues of the corresponding Laplacian matrix.
- Define the Laplacian operator: $\mathbf{L}(\mathbf{W}) = \text{Diag}(\mathbf{W1}) - \mathbf{W}$.

- Rank constrained nonconvex formulation (Nie et al. 2016)[6]:

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \|\mathbf{W} - \mathbf{W}_0\|_F^2 \\ \text{subject to} & \mathbf{W1} = \mathbf{1},\ \text{diag}(\mathbf{W}) = \mathbf{0},\ \mathbf{W} = \mathbf{W}^T \geq \mathbf{0} \\ & \text{rank}(\mathbf{L}(\mathbf{W})) = p - k. \end{array}$$

[6] F. Nie, X. Wang, M. Jordan, and H. Huang, "The constrained Laplacian rank algorithm for graph-based clustering," in *Proc. of the Thirtieth Conference on Artificial Intelligence (AAAI)*, Phoenix, Arizona, USA, 2016, pp. 1969–1976.

Learning a k-connected graph

- If we relax the low-rank constraint $\lambda_1(\mathbf{L}(\mathbf{W})) = \cdots = \lambda_k(\mathbf{L}(\mathbf{W})) = 0$, we can rewrite the problem as

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \|\mathbf{W} - \mathbf{W}_0\|_F^2 + \beta \sum_{i=1}^{k} \lambda_i(\mathbf{L}(\mathbf{W})) \\ \text{subject to} & \mathbf{W}\mathbf{1} = \mathbf{1},\ \text{diag}(\mathbf{W}) = \mathbf{0},\ \mathbf{W} = \mathbf{W}^T \geq \mathbf{0} \end{array}$$

- We now use the variational interpretation of the smallest eigenvalues known as Ky Fan's theorem (Fan 1949):

$$\sum_{i=1}^{k} \lambda_i(\mathbf{X}) = \min_{\mathbf{F} \in \mathbb{R}^{p \times k}} \quad \text{Tr}(\mathbf{F}^T \mathbf{X} \mathbf{F}) \quad \text{s.t.} \quad \mathbf{F}^T \mathbf{F} = \mathbf{I}.$$

- The problem can then be rewritten in a more manageable form (still nonconvex) as

$$\begin{array}{ll} \underset{\mathbf{W},\mathbf{F}}{\text{minimize}} & \|\mathbf{W} - \mathbf{W}_0\|_F^2 + \beta \text{Tr}(\mathbf{F}^T \mathbf{L}(\mathbf{W}) \mathbf{F}) \\ \text{subject to} & \mathbf{W}\mathbf{1} = \mathbf{1},\ \text{diag}(\mathbf{W}) = \mathbf{0},\ \mathbf{W} = \mathbf{W}^T \geq \mathbf{0} \\ & \mathbf{F}^T \mathbf{F} = \mathbf{I}. \end{array}$$

Learning a k-connected graph

- To solve this nonconvex problem in the variables $(\mathbf{W}, \mathbf{F})$, we will use the block coordinate descent (BCD) method, which in this particular case with two blocks it becomes a simple sequential optimization with respect to $\mathbf{W}$ and $\mathbf{F}$ alternatively.
- Optimization w.r.t. $\mathbf{F}$:

$$\begin{array}{ll} \underset{\mathbf{F}}{\text{minimize}} & \text{Tr}(\mathbf{F}^T\mathbf{L}(\mathbf{W})\mathbf{F}) \\ \text{subject to} & \mathbf{F}^T\mathbf{F} = \mathbf{I}. \end{array}$$

  whose solution is trivially given by the eigenvectors corresponding to the k smallest eigenvalues of $\mathbf{L}(\mathbf{W})$.
- Optimization w.r.t. $\mathbf{W}$ is a quadratic problem (QP):

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \|\mathbf{W} - \mathbf{W}_0\|_F^2 + \beta\text{Tr}(\mathbf{F}^T\mathbf{L}(\mathbf{W})\mathbf{F}) \\ \text{subject to} & \mathbf{W}\mathbf{1} = \mathbf{1},\ \text{diag}(\mathbf{W}) = \mathbf{0},\ \mathbf{W} = \mathbf{W}^T \geq \mathbf{0}. \end{array}$$

Learning a k-connected graph

- From the property of the Laplacian matrix we can write:

$$\text{Tr}(\mathbf{F}^T\mathbf{L}(\mathbf{W})\mathbf{F}) = \frac{1}{2}\sum_{i,j} w_{ij}\|\mathbf{f}_i - \mathbf{f}_j\|^2$$

- The problem w.r.t. $\mathbf{W}$ can then be rewritten as

$$\begin{array}{ll} \underset{\mathbf{W}}{\text{minimize}} & \|\mathbf{W} - \mathbf{W}_0\|_F^2 + \frac{\beta}{2}\sum_{i,j} w_{ij}\|\mathbf{f}_i - \mathbf{f}_j\|^2 \\ \text{subject to} & \mathbf{W}\mathbf{1} = \mathbf{1},\ \text{diag}(\mathbf{W}) = \mathbf{0},\ \mathbf{W} = \mathbf{W}^T \geq \mathbf{0}. \end{array}$$

- If we ignore the symmetry constraint in the adjacency matrix $\mathbf{W} = \mathbf{W}^T$, then the problem can be nicely solved separately for each row/column $\mathbf{w}_i$ as in (Nie et al. 2016)[7]:

$$\begin{array}{ll} \underset{\mathbf{w}_i}{\text{minimize}} & \|\mathbf{w}_i - \mathbf{w}_{0,i}\|^2 + \frac{\beta}{2}\mathbf{w}_i^T\mathbf{v}_i \\ \text{subject to} & \mathbf{w}_i^T\mathbf{1} = 1,\ w_{ii} = 0,\ \mathbf{w}_i \geq \mathbf{0}, \end{array}$$

where $v_{ij} = \|\mathbf{f}_i - \mathbf{f}_j\|^2$.

[7] F. Nie, X. Wang, M. Jordan, and H. Huang, "The constrained Laplacian rank algorithm for graph-based clustering," in *Proc. of the Thirtieth Conference on Artificial Intelligence (AAAI)*, Phoenix, Arizona, USA, 2016, pp. 1969–1976.

Learning a k-connected graph

- The problem can be finally rewritten as

$$\begin{array}{ll} \underset{\mathbf{w}_i}{\text{minimize}} & \left\| \mathbf{w}_i - \left(\mathbf{w}_{0,i} - \frac{\beta}{2} \mathbf{v}_i \right) \right\|^2 \\ \text{subject to} & \mathbf{w}_i^T \mathbf{1} = 1, \ w_{ii} = 0, \ \mathbf{w}_i \geq \mathbf{0}, \end{array}$$

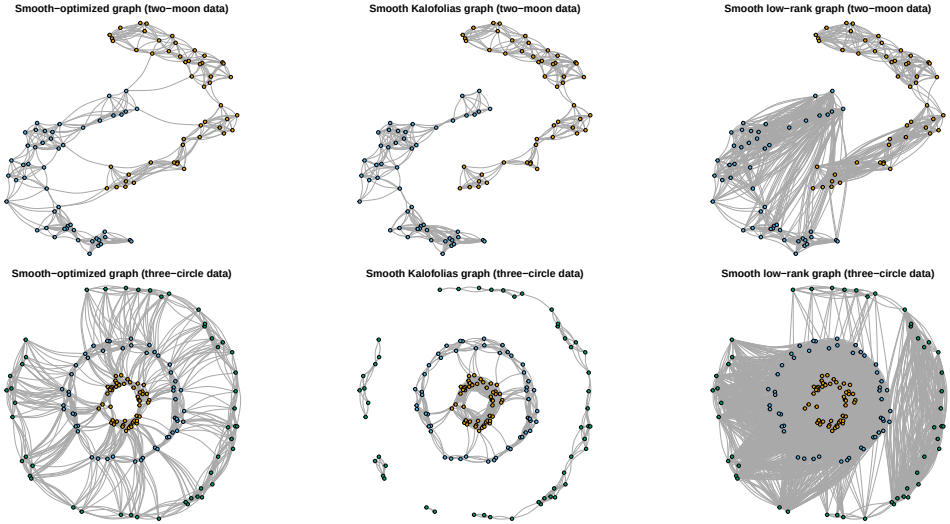
  where $v_{ij} = \|\mathbf{f}_i - \mathbf{f}_j\|^2$.

- We have seen this problem before and it has the closed-form solution for $j \neq i$ (recall $w_{ii} = 0$)

$$w_{ji} = \left(\mathbf{w}_{0,i} - \frac{\beta}{2} \mathbf{v}_i + \eta_i \right)^+$$

  where η is found so that the constraint $\mathbf{w}_i^T \mathbf{1} = 1$ is satisfied.

Graphs based on smooth signals



Outline

Notation for time series

- Our convention for graphs is to form the data matrix as $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{p \times n}$, where p is the number of nodes and n is the number of observations per node. Note that each $\mathbf{x}_i$ denotes one graph observation.
- In the context of time series, we usually denote the number of series by N and the number of time observations by T (with observation along rows instead of columns). So the correspondence of the two conventions is
  - $p = N$: number of nodes/series
  - $n = T$: number of observations/features
- The way to construct our graph data matrix $\mathbf{X}$ from a time series is by transposing the usual time series matrix (where each column represents the T observations of one series). In our graph notation, we want each series along a row of $\mathbf{X}$ and each column is one vector observation, so that $\mathbf{X}$ is $N \times T$ ($p \times n$):

$$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T] \in \mathbb{R}^{N \times T}.$$

Modeling as i.i.d. Random Variables

- Until now the data matrix $\mathbf{X}$ was composed of some (columnwise) observations for each of the nodes without any statistical modeling.
- Examples of observations $\mathbf{x}_i$:
  - the x-y coordinates of some points (e.g., two-moon dataset, three-circle dataset)
  - feature vector with arbitrary features (e.g., the animal dataset, where the features include attributed such as whether the animal is a mammal, has wings, has feathers, weight, size, etc.)
- Now we will assume a **statistical model** where the observations are multivariate i.i.d. random variables and each multivariate observation $\mathbf{x}_i$ is distributed according to some multivariate distribution.
- Example: Gaussian distribution

$$\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

where $\boldsymbol{\mu}$ is the mean and $\boldsymbol{\Sigma}$ the covariance matrix of the observations.

Graph from covariance matrix

- If our dataset is a collection of multivariate i.i.d. random variables $\{\mathbf{x}_i\}$, we may look for more interesting similarity measures.
- Let $\mathbf{\Sigma} = \mathbb{E}[\mathbf{x}_i \mathbf{x}_i^T]$ be the covariance matrix of the multivariate random i.i.d. variables. The correlation matrix is a normalized version (equivalent to normalizing the variance of each stock):

$$\mathbf{C} = \mathbf{D}^{-1/2} \mathbf{\Sigma} \mathbf{D}^{-1/2}$$

  where $\mathbf{D} = \text{Diag}(\mathbf{\Sigma})$.
- The correlation matrix can be used for graph construction (Heimo et al. 2007)[8].
- We can consider a modification of the correlation matrix as an adjacency matrix:

$$\mathbf{W} = |\mathbf{C}| - \mathbf{I}$$

  so that $\mathbf{W}$ has nonnegative elements and zero along the diagonal.

[8] T. Heimo, J. Saramaki, J.-P. Onnela, and K. Kaski, "Spectral and network methods in the analysis of correlation matrices of stock returns," *Physica A: Statistical Mechanics and its Applications*, vol. 383, no. 1, pp. 147–151, 2007.

Graph from precision matrix

- The covariance/correlation matrix measures the direct dependency between two nodes but ignores the other nodes.
- Can we improve that? Answer: **conditional dependence graph**.
- **Definition**: We say that two random variables X and Y are conditionally independent given a third variable Z if their conditional probability distributions given Z are independent. We denote it by $X \perp Y \mid Z$.
  - **Example**: Height and vocabulary of kids are not independent, but they are conditionally independent if you also consider age.
- **Definition**: The **precision matrix** is defined as $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$.
- **Theorem**: Suppose our random variables are Gaussian with zero mean and covariance $\mathbf{\Sigma}$. Then

$$\mathbf{\Theta}_{ij} = 0 \iff X_i \perp X_j \mid \{X_l\}_{l \neq i,j}.$$

- In words: the non-zero entries of $\mathbf{\Theta}$ indicate **conditional dependence** between the two random variables given all the other random variables.

Historical timeline of Markov graphical models

- Suppose the columns of the data matrix $\mathbf{X}$ follow $\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and let $\mathbf{S}$ be the sample covariance matrix.
- Covariance selection (Dempster 1972): graph from the elements of $\mathbf{S}^{-1}$ inverse sample covariance matrix.
- Neighborhood regression (Meinshausen and Bühlmann 2006):

$$\arg\min_{\beta_1} \; |\mathbf{x}^{(1)} - \beta_1^T \mathbf{X}_{/\mathbf{x}^{(1)}}|^2 + \alpha \|\beta\|_1.$$

- ℓ_1-regularized MLE (Banerjee et al. 2008; Friedman et al. 2008):

$$\underset{\boldsymbol{\Theta} \succ \mathbf{0}}{\text{maximize}} \; \log\det(\boldsymbol{\Theta}) - \text{Tr}(\boldsymbol{\Theta}\mathbf{S}) - \alpha \|\boldsymbol{\Theta}\|_1.$$

- Ising model: ℓ_1-regularized logistic regression (Ravikumar et al. 2010).
- Attractive IGMRF (Slawski and Hein 2015).
- Laplacian structure in $\boldsymbol{\Theta}$ (Lake and Tenenbaum 2010).
- ℓ_1-regularized MLE with Laplacian structure (Egilmez et al. 2017; Zhao et al. 2019).

Graphical LASSO (GLASSO)

- In real life, interactions are typically **local** and **sparse**.
- Normally the ideal precision matrix will be **sparse**.
- But the sample covariance matrix is **noisy**.
- **Graphical LASSO** tries to learn a sparse precision matrix:

$$\underset{\boldsymbol{\Theta} \succ \mathbf{0}}{\text{maximize}} \quad \log\det(\boldsymbol{\Theta}) - \text{Tr}(\mathbf{S}\boldsymbol{\Theta}) - \rho\|\boldsymbol{\Theta}\|_{1,\text{off}}$$

where $\mathbf{S}$ is the sample covariance matrix and $\|\cdot\|_{1,\text{off}}$ denotes the ℓ_1 elementwise of the off-diagonal elements.

👉 If $\rho = 0$, the solution is given precisely by the inverse sample covariance matrix $\boldsymbol{\Theta} = \mathbf{S}^{-1}$.

- Seminal reference (Friedman et al. 2008)[9].

[9] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

Derivation of GLASSO algorithm

- The optimality condition for the problem

$$\underset{\boldsymbol{\Theta} \succ \mathbf{0}}{\text{maximize}} \quad \log\det(\boldsymbol{\Theta}) - \text{Tr}(\mathbf{S}\boldsymbol{\Theta}) - \rho\|\boldsymbol{\Theta}\|_{1,\text{off}}$$

is

$$\boldsymbol{\Theta}^{-1} - \mathbf{S} - \alpha\boldsymbol{\Gamma} = \mathbf{0},$$

where $\boldsymbol{\Gamma}$ is a matrix of component-wise signs of $\boldsymbol{\Theta}$:

$$[\boldsymbol{\Gamma}]_{jk} = \begin{cases} \gamma_{jk} = \text{sign}(\Theta_{jk}), & \text{if } \Theta_{jk} \neq 0 \\ \gamma_{jk} \in [-1, 1], & \text{if } \Theta_{jk} \neq 0. \end{cases}$$

- The equation for optimality condition is also known as the normal equation.
- Furthermore, the constraint requires Θ_{jj} to be positive, this implies that

$$\hat{\Sigma}_{ii} = S_{ii} + \alpha, \quad i = 1, \ldots, p,$$

where $\hat{\boldsymbol{\Sigma}} = \boldsymbol{\Theta}^{-1}$.

Derivation of GLASSO algorithm

- GLASSO uses a block-coordinate method for solving the problem.
- Consider a partitioning of $\mathbf{\Theta}$ and $\hat{\mathbf{\Sigma}}$:

$$\mathbf{\Theta} = \begin{pmatrix} \mathbf{\Theta}_{11} & \boldsymbol{\theta}_{12} \\ \boldsymbol{\theta}_{12}^T & \theta_{22} \end{pmatrix}, \quad \hat{\mathbf{\Sigma}} = \begin{pmatrix} \hat{\mathbf{\Sigma}}_{11} & \hat{\boldsymbol{\sigma}}_{12} \\ \hat{\boldsymbol{\sigma}}_{12}^T & \hat{\sigma}_{22} \end{pmatrix},$$

  where $\mathbf{\Theta}_{11} \in \mathbb{R}^{(p-1)\times(p-1)}$, $\hat{\boldsymbol{\theta}}_{12} \in \mathbb{R}^{p-1}$ and θ_{22} is a scalar, and similarly for the other partitions.
- Then, $\hat{\mathbf{\Sigma}} = \mathbf{\Theta}^{-1}$ ($\mathbf{\Theta}\hat{\mathbf{\Sigma}} = \mathbf{I}$) can be expressed as

$$\hat{\mathbf{\Sigma}} = \begin{pmatrix} \mathbf{\Theta}_{11}^{-1} + \frac{\mathbf{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}\boldsymbol{\theta}_{12}^T\mathbf{\Theta}_{11}^{-1}}{\theta_{22}-\boldsymbol{\theta}_{12}^T\mathbf{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}} & \frac{\mathbf{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}}{\theta_{22}-\boldsymbol{\theta}_{12}^T\mathbf{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}} \\ \cdot & \frac{1}{\theta_{22}-\boldsymbol{\theta}_{12}^T\mathbf{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}} \end{pmatrix}$$

- GLASSO solves for a row/column of $\mathbf{\Theta}$ at a time, holding the rest fixed. Considering the pth column of the normal equation, we get

$$-\hat{\boldsymbol{\sigma}}_{12} + \mathbf{s}_{12} + \alpha\boldsymbol{\gamma}_{12} = \mathbf{0}.$$

Derivation of GLASSO algorithm

- Consider reading off $\boldsymbol{\theta}_{12}$ from the partitioned expression:

$$\frac{\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}}{\theta_{22} - \boldsymbol{\theta}_{12}^T\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}} + \mathbf{s}_{12} + \alpha\boldsymbol{\gamma}_{12} = \mathbf{0}.$$

- The above also simplifies to

$$\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}\hat{\sigma}_{22} + \mathbf{s}_{12} + \alpha\boldsymbol{\gamma}_{12} = \mathbf{0}$$

with $\boldsymbol{\nu} = \boldsymbol{\theta}_{12}\hat{\sigma}_{22}$ (with $\hat{\sigma}$ fixed), $\boldsymbol{\Theta}_{11} \succ \mathbf{0}$ is equivalent to the stationary condition for (Mazumder and Hastie 2012)

$$\underset{\boldsymbol{\nu}\in\mathbb{R}^{p-1}}{\text{minimize}} \quad \frac{1}{2}\boldsymbol{\nu}^T\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\nu} + \boldsymbol{\nu}^T\mathbf{s}_{12} + \alpha\|\boldsymbol{\nu}\|_1$$

- Let $\boldsymbol{\nu}^\star$ be the minimizer, then

$$\boldsymbol{\theta}_{12}^\star = \boldsymbol{\nu}^\star\hat{\sigma}_{22}$$

$$\theta_{22}^\star = \frac{1}{\hat{\sigma}_{22}} + (\boldsymbol{\theta}_{12}^\star)^T\boldsymbol{\Theta}_{11}^{-1}\boldsymbol{\theta}_{12}^\star$$

Derivation of GLASSO algorithm

Algorithm Graphical LASSO

initialize: $\hat{\mathbf{\Sigma}} = \text{Diag}(\mathbf{S}) + \alpha\mathbf{I}$ and $\mathbf{\Theta} = \hat{\mathbf{\Sigma}}^{-1}$. **repeat**

- Rearrange rows and columns such that the target column is the last.
- Compute $\mathbf{\Theta}_{11}^{-1} = \hat{\mathbf{\Sigma}}_{11} - \frac{\hat{\boldsymbol{\sigma}}_{12}\hat{\boldsymbol{\sigma}}_{12}^T}{\hat{\sigma}_{22}}$
- Obtain $\boldsymbol{\nu}$ and update $\boldsymbol{\theta}_{12}^\star$ and $\theta_{22}^\star$.
- Update $\mathbf{\Theta}$ and $\hat{\mathbf{\Sigma}}$ using the second partition function, ensuring $\mathbf{\Theta}\hat{\mathbf{\Sigma}} = \mathbf{I}$.

until convergence

return $\mathbf{\Theta}$

Laplacian-structured GLASSO

- We can compute a precision-like matrix with the **Laplacian structure** by adding the following constraints:

$$\mathbf{L} \succeq \mathbf{0},\ \mathbf{L1} = \mathbf{0},\ L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j.$$

- **Laplacian-structured GLASSO**: precision estimation with Laplacian constraints and ℓ_1-norm regularization:

$$\begin{array}{ll} \underset{\mathbf{L} \succeq \mathbf{0}}{\text{maximize}} & \log \text{gdet}\,(\mathbf{L}) - \text{Tr}\,(\mathbf{SL}) - \rho \|\mathbf{L}\|_{1,\text{off}} \\ \text{subject to} & \mathbf{L1} = \mathbf{0},\ L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j \end{array}$$

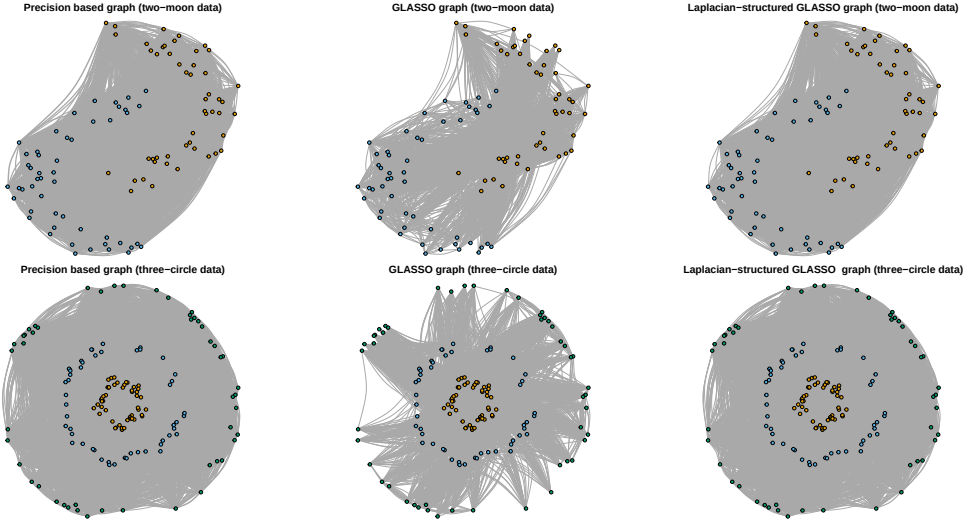
- Basic references (Lake and Tenenbaum 2010)[10], (Egilmez et al. 2017)[11], and (Zhao et al. 2019)[12].

[10] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graphs," in *Proc. the 33rd Annual Cognitive Science Conference*, 2010.

[11] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, 2017.

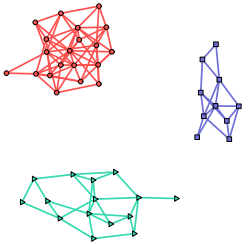
[12] L. Zhao, Y. Wang, S. Kumar, and D. P. Palomar, "Optimization algorithms for graph Laplacian estimation via ADMM and MM," *IEEE Trans. on Signal Processing*, vol. 67, no. 16, pp. 4231–4244, 2019.

Graphs based on the i.i.d. model

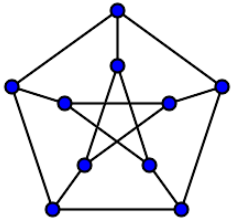


Outline

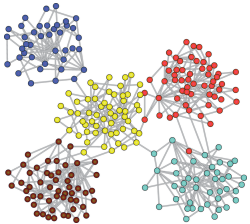
Structured graphs



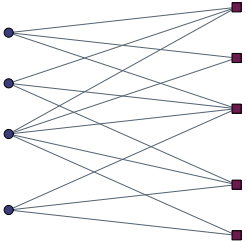
Multi-component graph



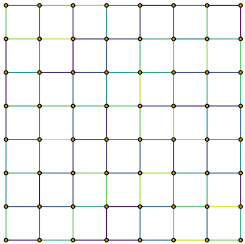
Regular graph



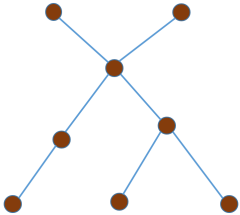
Modular graph



Bipartite graph



Grid graph



Tree graph

Structured graphs: Importance and challenges

Useful graph structures:

- Multi-component: for clustering, classification.
- Bipartite: for matching and constructing two-channel filter banks.
- Multi-component bipartite: for co-clustering.
- Tree: for sampling algorithms.
- Modular: for social network analysis.
- Connected sparse: for graph signal processing applications.

Structured graph learning from data

- involves both the estimation of structure (graph connectivity) and parameters (graph weights),
- parameter estimation is well explored (e.g., maximum likelihood),
- but structure is a combinatorial property which makes structure estimation very challenging.

👉 Structure learning is NP-hard for a general class of graphical models (Bogdanov et al. 2008).

Eigenvalue property of Laplacian matrix

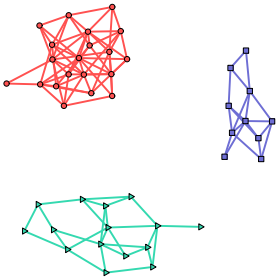
- Consider the eigenvalue decomposition of the Laplacian matrix:

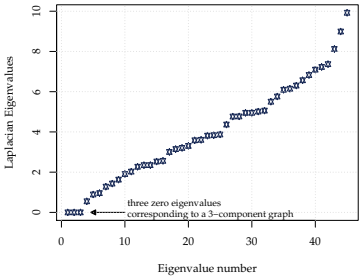
$$\mathbf{L} = \mathbf{U}\text{Diag}(\lambda_1, \lambda_2, \ldots, \lambda_p)\mathbf{U}^T,$$

  where $\mathbf{U}$ contains the eigenvectors columnwise and $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_p$ are the eigenvalues in increasing order.

- For a **multi-component graph** (aka *k*-**connected graph**), the k smallest eigenvalues are zero:

$$\lambda_1 = \cdots = \lambda_k = 0.$$





Eigenvalue property of adjacency matrix

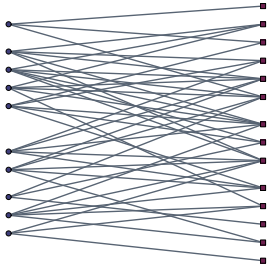
- Consider the eigenvalue decomposition of the adjacency matrix:

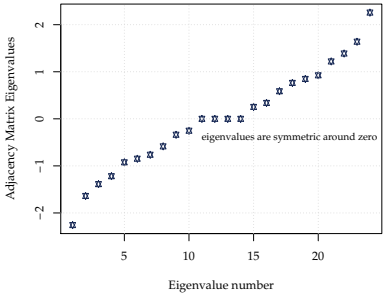
$$\mathbf{W} = \mathbf{V}\text{Diag}(\psi_1, \psi_2, \ldots, \psi_p)\mathbf{V}^T,$$

where $\mathbf{V}$ contains the eigenvectors columnwise and $\psi_1, \psi_2, \ldots, \psi_p$ are the eigenvalues in increasing order.

- For a bipartite graph, the eigenvalues are symmetric:

$$\psi_i = -\psi_{p-i} \quad \forall i.$$





Laplacian with spectral constraints

- The Laplacian-structured GLASSO is useful, but in many practical situations, graphs of more complex structures need to be estimated, e.g., k-component graph, bipartite graph, etc.
- Such classes of graphs can be enforced via spectral constraints on the graph matrices $\mathbf{W}$ and $\mathbf{L}$.
- The (nonconvex) formulation of the **Laplacian estimation with spectral constraints** is

$$\begin{array}{ll} \underset{\mathbf{L}\succeq\mathbf{0}}{\text{maximize}} & \log \text{gdet}\,(\mathbf{L}) - \text{Tr}\,(\mathbf{S}\mathbf{L}) - \rho h(\mathbf{L}) \\ \text{subject to} & \mathbf{L}\mathbf{1} = \mathbf{0},\ L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j \\ & \boldsymbol{\lambda}(\mathbf{L}) \in \mathcal{S}_{\lambda}. \end{array}$$

- References: (Kumar et al. 2019)[13] and (Kumar et al. 2020)[14].

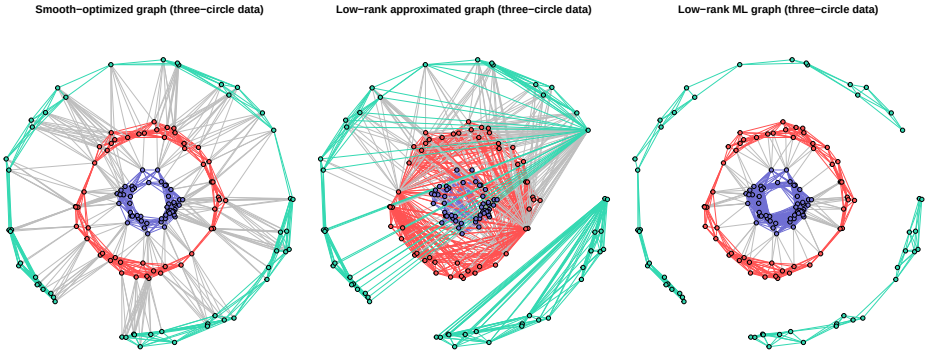
[13] S. Kumar, J. Ying, J. V. de M. Cardoso, and D. P. Palomar, "Structured graph learning via laplacian spectral constraints," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2019.

[14] S. Kumar, J. Ying, J. V. de M. Cardoso, and D. P. Palomar, "A unified framework for structured graph learning via spectral constraints," *Journal of Machine Learning Research (JMLR)*, pp. 1–60, 2020.

Graphs with spectral constraints

Consider learning a graph with $k = 3$ clusters:

- without imposing the clusters, we get a fully connected graph;
- further approximating the previous graph with a low-rank one, we don't get the best results;
- learning directly a low-rank graph is the best option.



Derivation of Structured Graph Learning (SGL) algorithm

- Recall the original problem

$$\begin{array}{ll} \underset{\mathbf{L}\succeq\mathbf{0}}{\text{maximize}} & \log\text{gdet}\left(\mathbf{L}\right) - \text{Tr}\left(\mathbf{SL}\right) - \rho\|\mathbf{L}\|_{1,\text{off}} \\ \text{subject to} & \mathbf{L1} = \mathbf{0},\ L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j \\ & \boldsymbol{\lambda}(\mathbf{L}) \in \mathcal{S}_{\boldsymbol{\lambda}}. \end{array}$$

- To properly control the eigenvalues of $\mathbf{L}$, let's include them explicitly as variables:

$$\begin{array}{ll} \underset{\mathbf{L}\succeq\mathbf{0},\boldsymbol{\lambda},\mathbf{U}}{\text{maximize}} & \log\text{gdet}\left(\mathbf{L}\right) - \text{Tr}\left(\mathbf{SL}\right) - \rho\|\mathbf{L}\|_{1,\text{off}} \\ \text{subject to} & \mathbf{L1} = \mathbf{0},\ L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j \\ & \mathbf{L} = \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T, \quad \boldsymbol{\lambda} \in \mathcal{S}_{\boldsymbol{\lambda}}, \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}, \end{array}$$

  where $\boldsymbol{\lambda}$ denote the eigenvalues and $\mathbf{U}$ the eigenvectors columnwise.

- This formulation seems intractable with so many nonconvex constraints coupling the different variables.

  👉 To simplify it, we will introduct a linear operator $\mathcal{L}$ that transforms the Laplacian structural constraints into simple algebraic constraints.

Derivation of SGL algorithm

- We will now derive the linear operator $\mathcal{L}$ to characterize the Laplacian constraints.
- Recall that the valid Laplacian are defined in the set

$$\mathcal{S}_{\mathbf{L}} = \{\mathbf{L} \succeq \mathbf{0} \mid \mathbf{L}\mathbf{1} = \mathbf{0},\ L_{ij} = L_{ji} \leq 0, \quad \forall i \neq j\},$$

- It's not difficult to see that due to the constraints $\mathbf{L}\mathbf{1} = \mathbf{0}$ and $L_{ij} = L_{ji}$ the degrees of freedom of the Laplacian $\mathbf{L} \in \mathbb{R}^{p \times p}$ are actually $p(p-1)/2$.
- We can define a linear operator $\mathcal{L} : \mathbf{w} \in \mathbb{R}_+^{p(p-1)/2} \to \mathcal{L}\mathbf{w} \in \mathbb{R}^{p \times p}$ that maps a weight vector $\mathbf{w}$ to a valid Laplacian matrix satisfying all the required properties.
- Eaxmple for $p = 4$: $\mathbf{w} = [w_1, w_2, \ldots, w_6] \in \mathbb{R}^6$ and

$$\mathcal{L}\mathbf{w} = \begin{bmatrix} \sum_{i=1,2,3} w_i & -w_1 & -w_2 & -w_3 \\ -w_1 & \sum_{i=1,4,5} w_i & -w_4 & -w_5 \\ -w_2 & -w_4 & \sum_{i=2,4,6} w_i & -w_6 \\ -w_3 & -w_5 & -w_6 & \sum_{i=3,5,6} w_i \end{bmatrix}.$$

Derivation of SGL algorithm

- Using $\mathbf{L} = \mathcal{L}\mathbf{w}$ and $\text{gdet}(\mathbf{L}) = \text{gdet}(\text{Diag}(\boldsymbol{\lambda}))$, we can rewrite the problem as

$$\begin{array}{ll} \underset{\mathbf{w},\boldsymbol{\lambda},\mathbf{U}}{\text{maximize}} & \log \text{gdet}\left(\text{Diag}(\boldsymbol{\lambda})\right) - \text{Tr}\left(\mathbf{S}\mathcal{L}\mathbf{w}\right) - \rho\|\mathcal{L}\mathbf{w}\|_{1,\text{off}} \\ \text{subject to} & \mathcal{L}\mathbf{w} = \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T, \quad \boldsymbol{\lambda} \in \mathcal{S}_\lambda, \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}, \end{array}$$

- Noting that $\|\mathcal{L}\mathbf{w}\|_{1,\text{off}} = \frac{1}{2}\|\mathbf{w}\|_1 = \frac{1}{2}\mathbf{1}^T\mathbf{w}$, we can group together the two linear terms on $\mathbf{w}$ (i.e., $\text{Tr}\left(\mathbf{S}\mathcal{L}\mathbf{w}\right)$ and $\|\mathcal{L}\mathbf{w}\|_{1,\text{off}}$) into the single linear term $\mathbf{k}^T\mathbf{w}$ with a properly defined $\mathbf{k}$.
- Finally, relaxing the hard constraint $\mathcal{L}\mathbf{w} = \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T$ to the objective, we get the approximate formulation:

$$\begin{array}{ll} \underset{\mathbf{w},\boldsymbol{\lambda},\mathbf{U}}{\text{minimize}} & \mathbf{k}^T\mathbf{w} - \log \text{gdet}\left(\text{Diag}(\boldsymbol{\lambda})\right) + \frac{\beta}{2}\|\mathcal{L}\mathbf{w} - \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T\|_F^2 \\ \text{subject to} & \mathbf{w} \geq \mathbf{0}, \quad \boldsymbol{\lambda} \in \mathcal{S}_\lambda, \quad \mathbf{U}^T\mathbf{U} = \mathbf{I}. \end{array}$$

Derivation of SGL algorithm for k-connected graph

- To solve the problem in the three variables $(\mathbf{w}, \boldsymbol{\lambda}, \mathbf{U})$ we will use the block **majorization-minimization (MM)** method, which updates each variable sequentially while keeping the others fixed based on simple surrogate functions (Sun et al. 2017)[15], (Razaviyayn et al. 2013)[16].
- For illustration purposes we will give the algorithm for a k-connected graph.
- The constraints for each of the three variables are:
  - Spectral constraint: $\mathcal{S}_{\boldsymbol{\lambda}} = \{\{\lambda_j = 0\}_{j=1}^{k},\ c_1 \leq \lambda_{k+1} \leq \cdots \leq \lambda_p \leq c_2\}$.
  - Nonnegativity constraint: $\mathbf{w} \geq \mathbf{0}$.
  - Orthogonality constraint: $\mathbf{U}^T\mathbf{U} = \mathbf{I}$.

[15] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Trans. Signal Processing*, vol. 65, no. 3, pp. 794–816, 2017.

[16] M. Razaviyayn, M. Hong, and Z. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM J. Optim.*, vol. 23, no. 2, pp. 1126–1153, 2013.

Derivation of SGL algorithm: Update for w

- Sub-problem for $\mathbf{w}$:

$$\underset{\mathbf{w}\geq\mathbf{0}}{\text{minimize}} \quad \mathbf{k}^T\mathbf{w} + \tfrac{\beta}{2}\|\mathcal{L}\mathbf{w} - \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T\|_F^2.$$

- It can be rewritten as

$$\underset{\mathbf{w}\geq\mathbf{0}}{\text{minimize}} \quad f(\mathbf{w}) \triangleq \|\mathcal{L}\mathbf{w}\|_F^2 - \mathbf{c}^T\mathbf{w}$$

  which is clearly a convex quadratic program (QP).
- The QP does not have a closed-form solution due to the constraint $\mathbf{w} \geq \mathbf{0}$ and we will use MM.
- The function $f(\mathbf{w})$ is majorized at $\mathbf{w}^t$ by

$$g(\mathbf{w}|\mathbf{w}^t) = f(\mathbf{w}^t) + (\mathbf{w} - \mathbf{w}^t)^T\nabla f(\mathbf{w}^t) + \frac{L}{2}\|\mathbf{w} - \mathbf{w}^t\|^2$$

- The majorized problem has now a simple closed-form solution:

$$\mathbf{w}^{t+1} = \left(\mathbf{w}^t - \frac{1}{2p}\nabla f(\mathbf{w}^t)\right)^+.$$

Derivation of SGL algorithm: Update for U

- Sub-problem for $\mathbf{U}$:

$$\begin{array}{ll} \underset{\mathbf{U}}{\text{minimize}} & \frac{\beta}{2}\|\mathcal{L}\mathbf{w} - \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T\|_F^2 \\ \text{subject to} & \mathbf{U}^T\mathbf{U} = \mathbf{I}. \end{array}$$

- It can be rewritten as

$$\begin{array}{ll} \underset{\mathbf{U}}{\text{maximize}} & \text{Tr}(\mathbf{U}^T\mathcal{L}\mathbf{w}\mathbf{U}\text{Diag}(\boldsymbol{\lambda})) \\ \text{subject to} & \mathbf{U}^T\mathbf{U} = \mathbf{I}. \end{array}$$

- This sub-problem is an optimization on the orthogonal Stiefel manifold (Absil et al. 2009; Benidis et al. 2016).
- From the KKT optimality conditions the solution is given by

$$\mathbf{U}^{t+1} = \text{eigenvectors}(\mathcal{L}\mathbf{w}^{t+1})[k+1:p],$$

that is, the $p-k$ principal eigenvectors of the matrix $\mathcal{L}\mathbf{w}^{t+1}$ in increasing order of eigenvalue magnitude.

Derivation of SGL algorithm: Update for λ

- Sub-problem for $\boldsymbol{\lambda}$:

$$\underset{\boldsymbol{\lambda}\in\mathcal{S}_{\boldsymbol{\lambda}}}{\text{minimize}} \quad -\log \text{gdet}\left(\text{Diag}(\boldsymbol{\lambda})\right) + \frac{\beta}{2}\|\mathcal{L}\mathbf{w} - \mathbf{U}\text{Diag}(\boldsymbol{\lambda})\mathbf{U}^T\|_F^2.$$

- It can be rewritten for the k-component graph as

$$\underset{c_1\leq\lambda_{k+1}\leq\cdots\leq\lambda_p\leq c_2}{\text{minimize}} \quad -\sum_{i=1}^{p-k}\log(\lambda_{k+i}) + \frac{\beta}{2}\|\boldsymbol{\lambda} - \mathbf{d}\|^2.$$

- This sub-problem is popularly known as a regularized isotonic regression problem. It is a convex optimization problem and the solution can be obtained from the KKT optimality conditions.
- An efficient algorithm with a fast convergence to the global optimum can be derived with a maximum of $p-k$ iterations (Kumar et al. 2020).

Derivation of SGL algorithm: Summary

Algorithm SGL for k-connected graph

Input: $\mathbf{S}, k, c_1, c_2, \beta$

Output: $\mathcal{L}\mathbf{w}$

$t \leftarrow 0$

repeat

- $\mathbf{w}^{t+1} = \left(\mathbf{w}^t - \frac{1}{2p}\nabla f(\mathbf{w}^t)\right)^+$
- $\mathbf{U}^{t+1} \leftarrow$ eigenvectors$(\mathbf{L}\mathbf{w}^{t+1})$, suitably ordered.
- Update $\boldsymbol{\lambda}^{t+1}$ via isotonic regression.
- $t \leftarrow t+1$

until convergence

return $\mathcal{L}\mathbf{w}^t$

Derivation of SGL algorithm: Convergence

The following gives the convergence of the SGL algorithm to a stationary point (Kumar et al. 2019)[17] and (Kumar et al. 2020)[18].

Theorem

The limit point $(\mathbf{w}^\star, \boldsymbol{\lambda}^\star, \mathbf{U}^\star)$ generated by the SGL algorithm converges to the set of KKT points of the optimization problem.

Furthermore, the worst-case computational complexity of the proposed algorithm is $O(p^3)$.

[17] S. Kumar, J. Ying, J. V. de M. Cardoso, and D. P. Palomar, "Structured graph learning via laplacian spectral constraints," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2019.

[18] S. Kumar, J. Ying, J. V. de M. Cardoso, and D. P. Palomar, "A unified framework for structured graph learning via spectral constraints," *Journal of Machine Learning Research (JMLR)*, pp. 1–60, 2020.

Outline

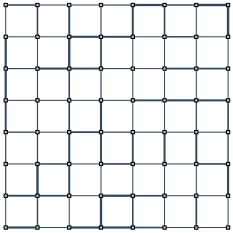
Synthetic experiment setup

- Generate a graph with desired structure.
- Generate weights for the graph edges.
- Obtain true Laplacian $\mathbf{L}_{\text{true}}$.
- Sample data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ with $\mathbf{x}_i \in \mathbb{R}^p \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma} = \mathbf{L}_{\text{true}}^{\dagger})$.
- $\mathbf{S} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$
- Use $\mathbf{S}$ as input for the learning and some prior spectral information, if available, to estimate the graph $\hat{\mathbf{L}}$
- Performance metric:

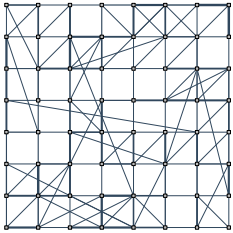
$$\text{Relative Error} = \frac{\|\hat{\mathbf{L}} - \mathbf{L}_{\text{true}}\|_F}{\|\mathbf{L}_{\text{true}}\|_F} \quad \text{and} \quad \text{F-Score} = \frac{2\text{tp}}{2\text{tp} + \text{fp} + \text{fn}},$$

  where tp, fp, fn correspond to true positives, false positives, and false negatives, respectively.
- SGL refers to structured graph learning.

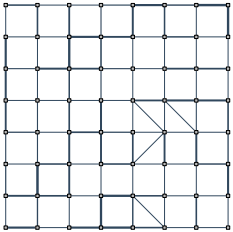
Grid graph



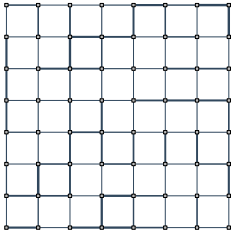
True



[Egilmez et al., 2017]

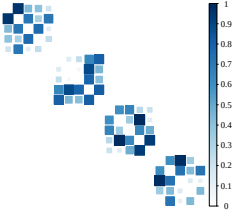


SGL with ℓ_1-norm

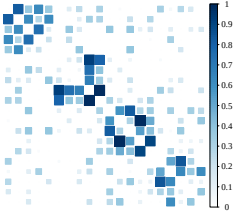


SGL with reweighted ℓ_1-norm

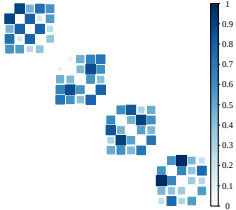
Noisy multi-component graph



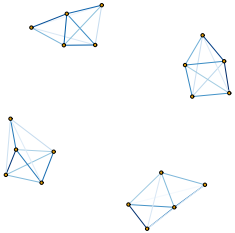
True $\mathbf{W}$



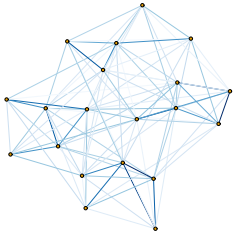
Noisy $\mathbf{W}$



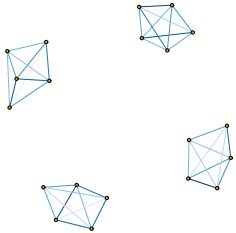
Learned $\mathbf{W}$



True graph

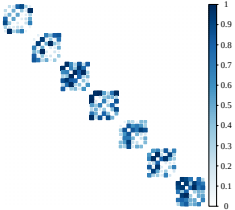


Noisy graph

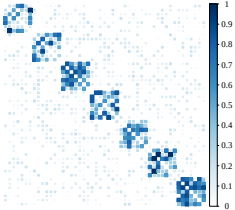


Learned graph

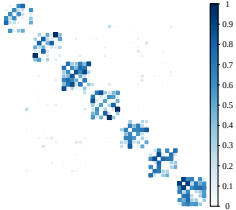
Model mismatch



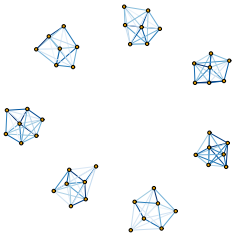
True $\mathbf{W}$



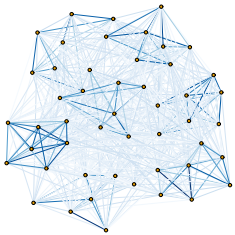
Noisy $\mathbf{W}$



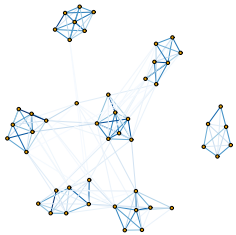
Learned $\mathbf{W}$ (with $k = 2$)



True graph

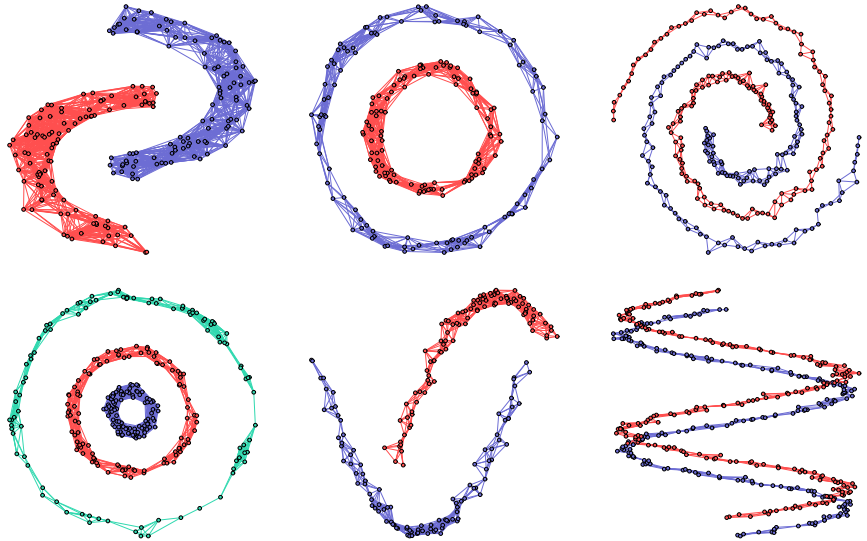


Noisy graph

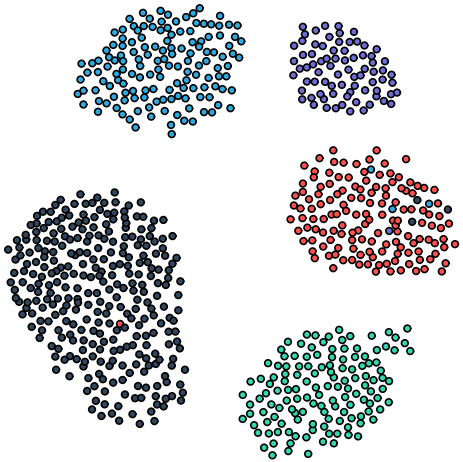


Learned graph (with $k = 2$)

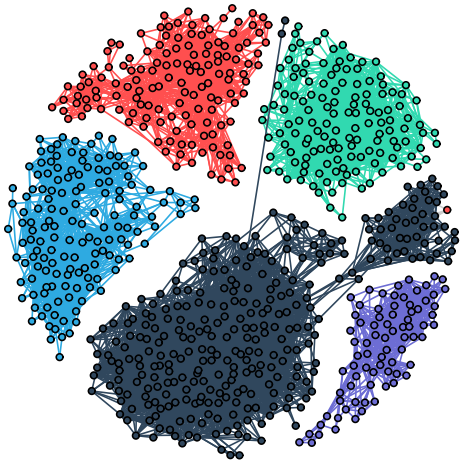
Popular multi-component structures



Real data: cancer dataset (Weinstein et al. 2013)



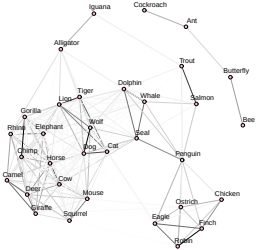
CLR (Nie et al., 2016)



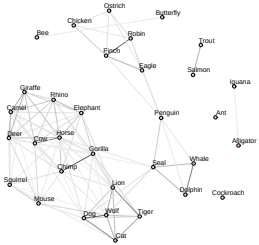
SGL with $k = 5$

- Clustering accuracy (ACC): CLR $= 0.9862$ and SGL $= 0.99875$.

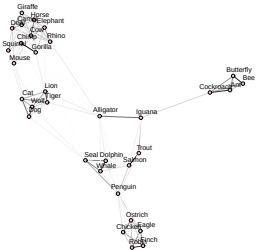
Animal dataset (Osherson et al. 1991)



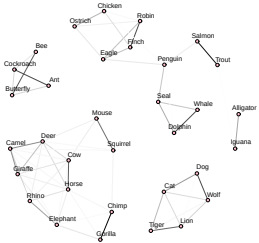
GGL [Egilmez et al., 2017]



GLasso [Friedman et al., 2008]

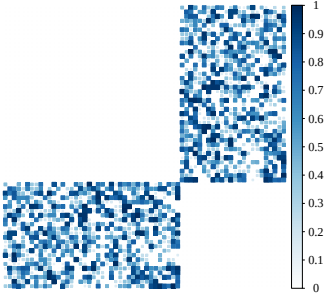


SGL with $k = 1$

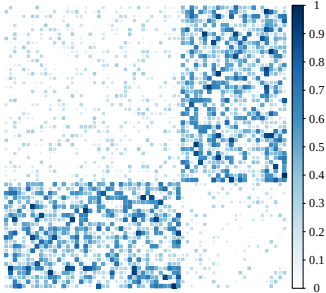


SGL with $k = 4$

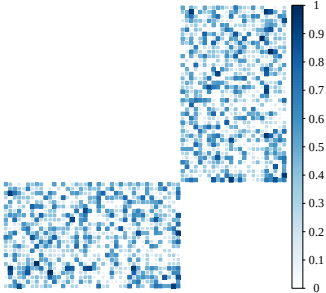
Bipartite structure via adjacency spectral constraints



True

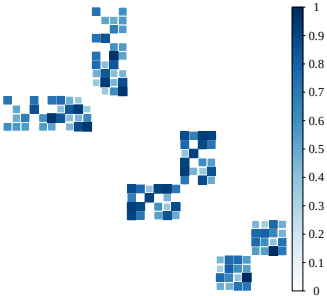


Noisy

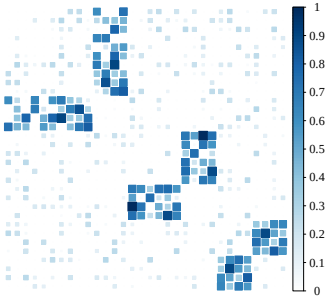


Learned

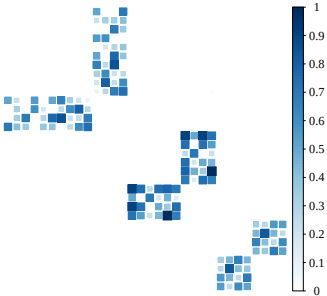
Multi-component bipartite structure via joint spectral constraints



True



Noisy



Learned

Thanks

For more information visit:

https://www.danielppalomar.com

References I

Absil, P. A., Mahony, R., & Sepulchre, R. (2009). *Optimization algorithms on matrix manifolds*. Princeton University Press.

Banerjee, O., Ghaoui, L., & d'Aspremont, A. (2008). Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, *9*(Mar), 485–516.

Benidis, K., Sun, Y., Babu, P., & Palomar, D. P. (2016). Orthogonal sparse PCA and covariance estimation via Procrustes reformulation. *IEEE Trans. Signal Processing*, *64*(23), 6211–6226.

Bogdanov, A., Mossel, E., & Vadhan, S. (2008). The complexity of distinguishing markov random fields. In *Approximation, randomization and combinatorial optimization. Algorithms and techniques* (pp. 331–342). Springer.

Dempster, A. P. (1972). Covariance selection. *Biometrics*, 157–175.

References II

Dong, X., Thanou, D., Rabbat, M., & Frossard, P. (2019). Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, *36*(3), 44–63.

Egilmez, H. E., Pavez, E., & Ortega, A. (2017). Graph learning from data under Laplacian and structural constraints. *IEEE Journal of Selected Topics in Signal Processing*, *11*(6), 825–841.

Fan, K. (1949). On a theorem of weyl concerning eigenvalues of linear transformations i. *Proceedings of the National Academy of Sciences of the United States of America*, *35*(11), 652.

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441.

Heimo, T., Saramaki, J., Onnela, J.-P., & Kaski, K. (2007). Spectral and network methods in the analysis of correlation matrices of stock returns. *Physica A: Statistical Mechanics and its Applications*, *383*(1), 147–151.

References III

Kalofolias, V. (2016). How to learn a graph from smooth signals. In *Proc. Int. Conf. Artif. Intell. Statist.* (pp. 920–929).

Kumar, S., Ying, J., M. Cardoso, J. V. de, & Palomar, D. P. (2019). Structured graph learning via laplacian spectral constraints. In *Proc. Advances in neural information processing systems (neurips)*. Vancouver, Canada.

Kumar, S., Ying, J., M. Cardoso, J. V. de, & Palomar, D. P. (2020). A unified framework for structured graph learning via spectral constraints. *Journal of Machine Learning Research (JMLR)*, 1–60.

Lake, B., & Tenenbaum, J. (2010). Discovering structure by learning sparse graphs. In *Proc. The 33rd annual cognitive science conference*.

Manor, L. Z., & Perona, P. (2004). Self-tuning spectral clustering. In *Proc. Advances in neural information processing systems (neurips)*.

References IV

Mateos, G., Segarra, S., Marques, A. G., & Ribeiro, A. (2019). Connecting the dots. *IEEE Signal Processing Magazine*, *36*(3), 16–43.

Mazumder, R., & Hastie, T. (2012). The graphical lasso: New insights and alternatives. *Electronic journal of statistics*, *6*, 2125.

Meinshausen, N., & Bühlmann, P. (2006). High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, *34*(3), 1436–1462.

Nie, F., Wang, X., Jordan, M., & Huang, H. (2016). The constrained Laplacian rank algorithm for graph-based clustering. In *Proc. Of the thirtieth conference on artificial intelligence (aaai)* (pp. 1969–1976). Phoenix, Arizona, USA.

Osherson, D. N., Stern, J., Wilkie, O., Stob, M., & Smith, E. E. (1991). Default probability. *Cognitive Science*, *15*(2), 251–269.

References V

Ravikumar, P., Wainwright, M., Lafferty, J. D., & others. (2010). High-dimensional ising model selection using ℓ_1-regularized logistic regression. *The Annals of Statistics*, *38*(3), 1287–1319.

Razaviyayn, M., Hong, M., & Luo, Z. (2013). A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM J. Optim.*, *23*(2), 1126–1153.

Slawski, M., & Hein, M. (2015). Estimation of positive definite m-matrices and structure learning for attractive gaussian markov random fields. *Linear Algebra and its Applications*, *473*, 145–179.

Sun, Y., Babu, P., & Palomar, D. P. (2017). Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Trans. Signal Processing*, *65*(3), 794–816.

References VI

Weinstein, J. N., Collisson, E. A., Mills, G. B., Shaw, K. R. M., Ozenberger, B. A., Ellrott, K., et al. (2013). The cancer genome atlas pan-cancer analysis project. *Nature Genetics*, *45*(10), 1113.

Zhao, L., Wang, Y., Kumar, S., & Palomar, D. P. (2019). Optimization algorithms for graph Laplacian estimation via ADMM and MM. *IEEE Trans. on Signal Processing*, *67*(16), 4231–4244.