Graph Learning from Smooth Signals

Gurusha Juneja
2019EE10480

April 2022

Disclaimer: These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Course Coordinator

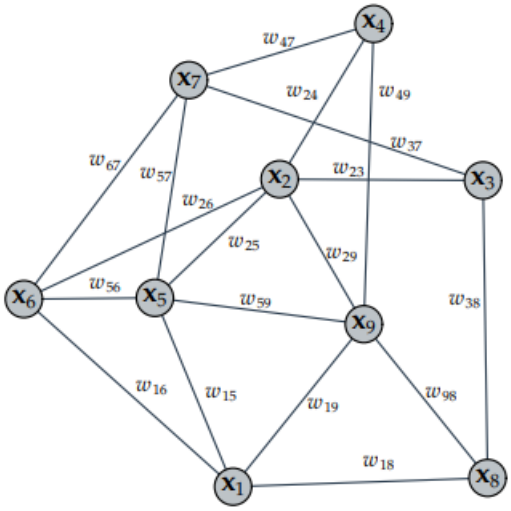
Contents

1 Introduction

Graph learning has become an important task with various applications. In these notes, we go through the algorithms to construct graph on a given dataset taking into consideration various constraints which help us make better connected graphs depicting real-life scenarios. One such way is learning graph from smooth signals, where we try to minimise the smoothness. Further in these notes, we explore how to approach problems that are highly combinatorial in nature, like learning structured graphs, and how using spectral properties of graphs can help us in doing so.

2 Schematic of Graph Learning

Given a data matrix $\mathbf{X} \in \mathbf{R}^{n*p} = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_p}]$, each column is assumed to represent one node on a graph and our goal is to obtain relationships between them as the weight of edges between these nodes as shown below,



Our goal is to obtain the weight matrix $\mathbf{W}$, where the weight of the edge between node i and node j is given by w_{ij}. The connectivity matrix $\mathbf{C}$, the weight matrix $\mathbf{W}$, the laplacian matrix $\mathbf{L}$ are defined as follows,

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

(1)

Relation between $\mathbf{W}$ and $\mathbf{L}$ can be given as, $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} = Diag(\mathbf{W}.\mathbf{1})$. Based on these matrices, various graph construction algorithms can be worked out. For example, sample correlation, where two nodes are connected if their pairwise correlation is greater than a given threshold, similarity based functions where $[\mathbf{W}]_{ij} = exp(-\frac{||\mathbf{x_i} - \mathbf{x_j}||_2^2}{2\sigma^2})$ and $w_{ii} = 0$.
However, some solutions can be degenerate values like $w_{ij} = 0 \, \forall i, j$, hence some sort of regularisation is required, which we do by learning graphs from smooth signals.

3 Quantifying Smoothness

The energy of a system is defined as $tr(\mathbf{XLX^T}) = \sum_{i,j} w_{ij} ||\mathbf{x_i} - \mathbf{x_j}||_2^2$. Smaller the value of the energy, smoother the signal X over the graph $\mathcal{G}$. When $\mathcal{G}$ is not available, we can learn it directly from $\mathbf{X}$ and simultaneously minimise the smoothness by adding some regularisation term denoted by $h(\mathbf{L})$. Hence, the optimal laplacian is given by,

$$\hat{\mathbf{L}} := \underset{\mathbf{L}}{argmin}\, tr(\mathbf{XLX^T}) + \lambda h(\mathbf{L}) \tag{2}$$

$h(\mathbf{L})$ can be any function like $||\mathbf{L}||_1, ||\mathbf{L}||_F^2, log(det(\mathbf{L}))$ etc.
Next we see how to formulate this problem as a convex optimisation problem and introduce some constraints.

3.1 Learning graph under constraints

Here we will introduce some constraints on the weight matrix so as to ensure that our solution becomes realistic, introducing the following constraints,

- Bounded weights: $0 \leq w_i \leq 1$
- sum of weights: $\sum_{i=1}^{p} w_{ij} = 1$
- Bounded energy: $\sum_{i=1}^{p} w_{ij}$

Hence the problem becomes,

$$\underset{w_{ij}}{Minimise} \sum_{i,j} w_{ij} ||\mathbf{x_i} - \mathbf{x_j}||_2^2 + \frac{\lambda}{2} \sum_{j=1}^{p} w_{ij}^2 \tag{3}$$

$$Subject\, to \sum_{j=1}^{p} w_{ij} = 1 \; w_{ij} \geq 0 \; w_{ii} = 0 \; \forall i, j \tag{4}$$

Since the objective function is convex and the constraints are all linear, this is a convex optimisation problem, we will get a closed form update rule.

When to use negative weights and when they don't make sense
They make sense when dealing with biology or chemistry of molecules, they don't make sense in real life scenarios.

3.2 How to approach an optimisation problem?

For the above optimisation problem, let us construct a matrix $\mathbf{e}$ s.t. $e_{ij} = ||\mathbf{x_i} - \mathbf{x_j}||_2^2$ where $\mathbf{e_i}$ as a vector with e_{ij} as the j^{th} element of the vector. This transforms the problem into:

$$\underset{\mathbf{w}_i}{Minimise}\ \frac{1}{2}||\mathbf{w}_i + \frac{\mathbf{e}_i}{\lambda}||_2^2 \tag{5}$$

$$subject\,to\ \mathbf{w}_i^T\mathbf{1} = 1, w_{ij} \geq 0, w_{ii} = 0 \tag{6}$$

Lagrangian function for this problem is given by the following expression,

$$\mathcal{L}(\mathbf{w_i}, \eta_i, \beta_i) = \frac{1}{2}||\mathbf{w}_i + \frac{\mathbf{e}_i}{\lambda}||_2^2 - \eta_i(\mathbf{w_i}^T \cdot \mathbf{1} - 1) - \beta_i^T \cdot \mathbf{w_i} \tag{7}$$

Where $\eta_i > 0, \beta_i \in \mathbf{R}^p$. Writing the KKT conditions,

1. Derivative of Lagrangian w.r.t. $\mathbf{w_i}$. so we have

$$\hat{\mathbf{w}_\mathbf{i}} + \frac{\mathbf{e_i}}{\lambda} - \eta_i\mathbf{1} - \beta_i = 0 \tag{8}$$

2. Setting derivative of the j_th element of w_i, we have

$$\hat{\mathbf{w}_\mathbf{ij}} + \frac{\mathbf{e_{ij}}}{\lambda} - \eta_i - \beta_{ij} = 0 \tag{9}$$

3. $\hat{w_{ij}}\beta_{ij} = 0$

We get,

$$\hat{w_{ij}} = (-\frac{e_{ij}}{\lambda} + \eta_i)^+ \tag{10}$$

where $a^+ = \max(0,a)$.

3.3 How to enforce sparsity?

Uptill now, we have got a graph that minimises the regularised errors subject to some constraints on the weights of the edges, but what if we want to enforce some sparsity conditions? This can be achieved in two ways,

1. Use l_1 norm.
2. choose λ s.t. each node has exactly $m(<< p)$ neighbours i.e. $||\mathbf{w_i}||_0 = m$

Exploring the second path, let us assume $e_{i1}, e_{i2}...e_{in}$ are arranged in increasing order.
Imposing the constraint $||\mathbf{w_i}||_0 = m$, we get $\hat{w}_{im} \geq 0$ and $\hat{w}_{im+1} = 0$, hence we have ,

$$-\frac{e_{im}}{\lambda_i} + 2\eta_i > 0\ and\ -\frac{e_{i(m+1)}}{\lambda_i} + \eta_i \leq 0 \tag{11}$$

combining the solution of $\hat{w}$ with the constraint $\hat{\mathbf{w}} \cdot \mathbf{1} = 1$, we have ,

$$\sum_{j=1}^{m}(-\frac{e_{im}}{\lambda_i} + 2\eta_i) = 1 \implies \eta_i = \frac{1}{m} + \frac{1}{2m\lambda_i}\sum_{j=1} m e_{ij} \tag{12}$$

This leads to the following inequality on λ ,

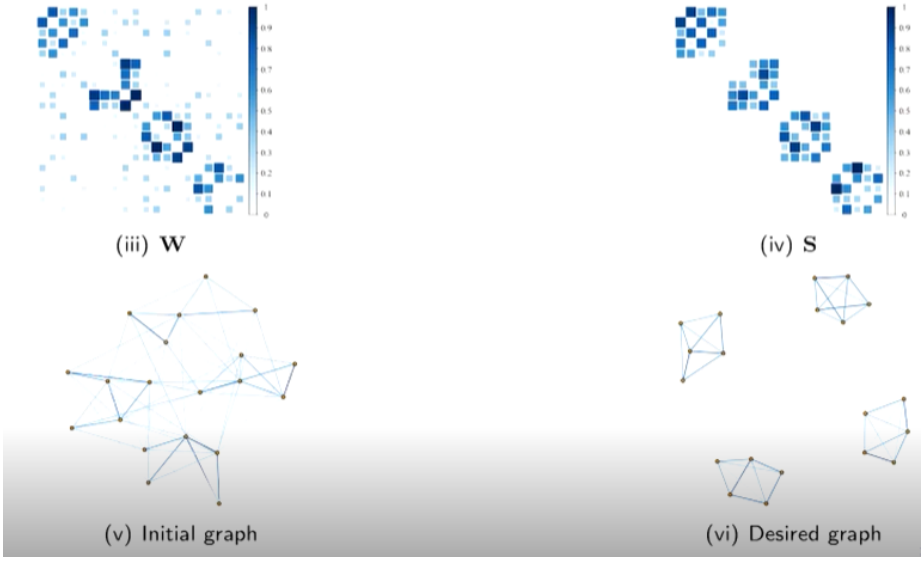
$$\frac{m}{2}e_{im} - \frac{1}{2}\sum_{j=1}^{m} e_{ij} < \lambda_i \leq \frac{m}{2}e_{i(m+1)} - \frac{1}{2}\sum_{j=1}^{m} e_{ij} \tag{13}$$

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

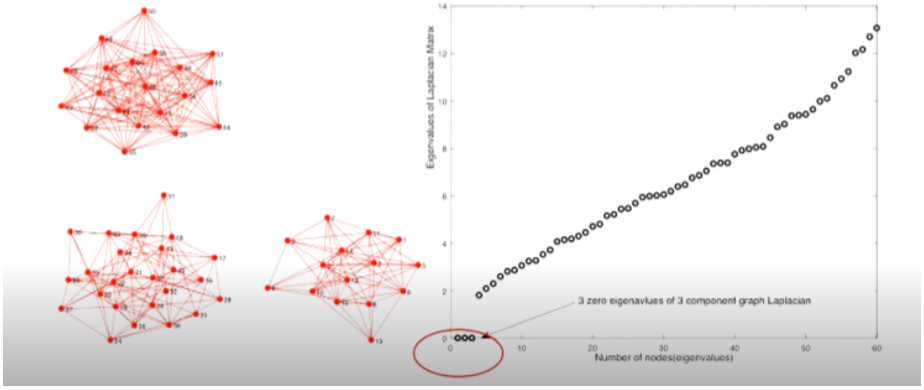
$$\lambda_i = \frac{m}{2}e_{i(m+1)} - \frac{1}{2}\sum_{j=1}^{m} e_{ij} \tag{14}$$

4 Graph Clustering

Our aim here is to learn a clustered graph from a raw graph. As shown in the figure below, on the left is the unclustured raw graph and on right is the clustured graph.



We can use the spectral properties of the graph. One property that can be used here is that the number of eigenvalues of the Laplacian is equal to the number of components present in the graph. As can be seen below, there are three components and there are three zero eigen values shown in the graph on the left.



4.1 Constrained Laplacian Rank for Clustering

In this algorithm, we first construct a weight matrix, where $w_{ij} = exp(-\frac{||\mathbf{x_i}-\mathbf{x_j}||_2^2}{2\sigma^2})$. This might not be a clustered graph matrix, but we can learn a clustered graph matrix with the help of this weight matrix. Let us define a matrix L_s as $(Diag(\mathbf{S1}) - \mathbf{S})$, hence the problem formulation is as follows:

$$\underset{\mathbf{S}=[\mathbf{s_1},\ldots,\mathbf{s_p}]}{Minimise}\ ||\mathbf{S}-\mathbf{W}||_F^2 \tag{15}$$

$$subject\ to\ \mathbf{s_i^T 1} = 1, s_i \geq 0, s_{ii} = 0, Rank(\mathbf{L}_s) = p - k$$

4.2 Kay-Fan Theorem

To solve such optimisation problems, Kay Fan theorem says that the sum of top k eigen values is the minimum value of the trace of $F^T L_s F$ subject to $F^T F = 1$. Formally stating,

$$\underset{\{\lambda_i(L_s)\}_{i=1}^k}{minimise} \sum_{i=1}^{k} \lambda_i(L_s) = \underset{F}{minimise}\, tr(\mathbf{F^T L_s F})\ s.t. \mathbf{F^T F} = 1 \tag{16}$$

Using the Kay Fan theorem, we can force the first k eigen values to be zero using the followinf formulation,

$$\underset{\mathbf{S},\mathbf{F}}{minimise} ||\mathbf{S}-\mathbf{W}||_2^2 + \beta tr(\mathbf{F^T L_s F}) \tag{17}$$

$$subject\ to\ \mathbf{s_i^T 1} = 1, \mathbf{s_{ij}} \geq 0, \mathbf{s_{ii}} = 0, \mathbf{F^T F} = I \tag{18}$$

4.3 Solving for S and F

To solve for S and F, we alternatively solve two subproblems, which are as follows,

1. For S,

$$\underset{s_{ij}}{minimise} \sum_j ||s_{ij} - w_{ij}||_2^2 + \beta \sum_{i,j} ||f_i - ff_j||_2^2 s_{ij} \tag{19}$$

$$subject\ to\ \mathbf{s_i^T 1} = 1, \mathbf{s_{ij}} \geq 0, \mathbf{s_{ii}} = 0, \tag{20}$$

2. For F

$$\underset{F}{minimise}\ tr(\mathbf{F^T L_s F})\ s.t. \mathbf{F^T F} = 1 \tag{21}$$

since the two problems are independent, we can solve them independently.

5 Spectral clustering Algorithm

This algorithm was devised by Ng. et Al. Here they present a spectral clusturing based algorithm, that is,
Given a set of points, $S = \{s_1, ...s_n\}$, we want to cluster them in k components,

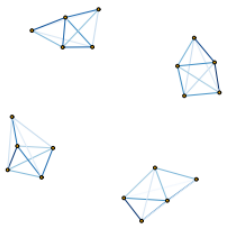
1. Form a matrix A s.t. $A_{ij} = exp(-||s_i - s_j||_2^2/2\sigma^2)$ is $i \neq j$ and 0 if $i = j$.
2. Define a diagonal matrix D and construct the matrix $L = D^{-1/2} A D^{1/2}$
3. Find $x_1, ...x_k$, the k largest Eigen values of L matrix and make a vector $X = [x_1, x_2...x_k]^T$ by stacking these k values.
4. Normalise X
5. Treating each row of X as a point in $\mathbf{R}^k$, clusture them into k components by K-means.
6. Assign each s_i to clusture j if and only if row i of the matrix X was assigned to clusture j.

6 Structure Learning

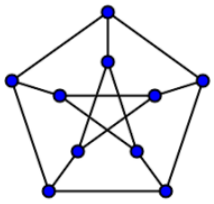
In many scenaro=ios, knowing the structure provides us useful insights and helps us learn better graphs, in this section we explore how the structural properties can help us provide constraints for a combinatorial problem and solve it with help of spectral properties.

6.1 What is structured graph learning?

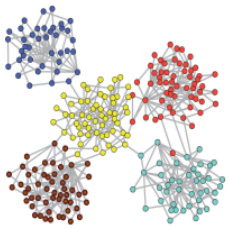
In many cases, we already know the structure of graph to be constructed for example, a multi component graph is required when we wish to do clustering. The figure below shows different types of graph structures.



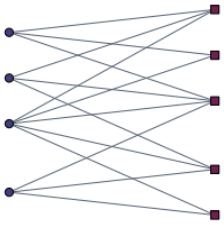
(iii) Multi-component graph



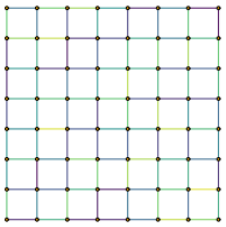
(iv) Regular graph



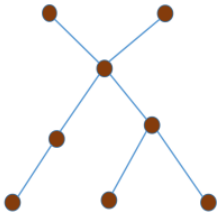
(v) Modular graph



(vi) Bipartite graph



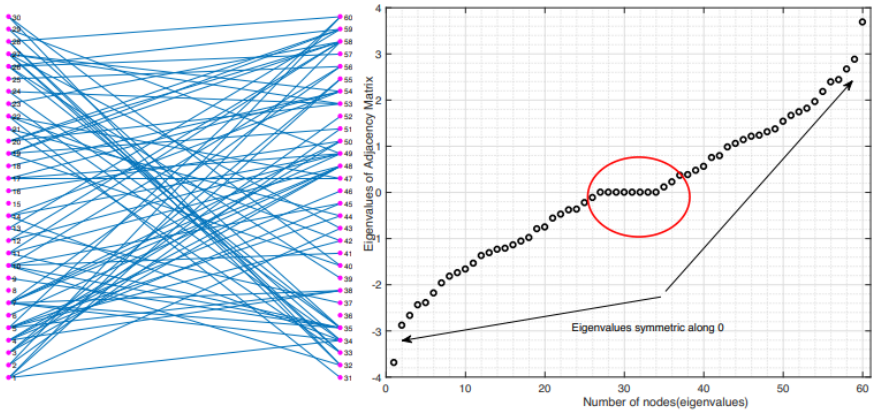
(vii) Grid graph



(viii) Tree graph

6.2 Spectral properties of structured graphs

Learning from the structure of a graph can be highly combinatorial in nature and hence is NP-Hard, but many properties of graphs are stored in their eigen values. For example, in any bipartite graph the adjacency matrix would have symmetric zero centered Eigenvalues.



can be seen above, the eigenvalues are symmetric about zero. This is a very important combinatorial property encoded in the spectral properties of the graph.

6.3 Incorporating Spectral Properties into graph learning framework

These properties of structured graphs can be used as constraints in the graph learning framework. Hence the problem formulation becomes,
Given $\mathbf{X} = \{\mathbf{x^{(i)}}\}_{i=1}^{n}$, we compute the sample co variance matrix $\mathbf{S} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x^{(i)}}\mathbf{x^{(i)T}}$. We want to estimate Θ which satisfies the laplacian constraints.

$$\mathbf{S_\Theta} = \{\Theta \in \mathbf{S_+^p} | \Theta_{ij} \leq 0 \; i \neq j \; \Theta.\mathbf{1} = 0\} \tag{22}$$

And the eigenvalues $\lambda(\Theta)$ which satisfy the spectral properties.

6.4 Proposed Unified framework for Graph structure Learning

The below proposed formulation has converted the combinatorial constraints into analytical constraints.

$$\underset{\Theta}{maximise}\; log\, gdet(\Theta) - tr(\mathbf{\Theta S}) - \alpha h(\Theta)$$

$$subject\, to\; \Theta \in S_\Theta, \lambda(T(\Theta)) \in S_T$$

Here, we are forcing Θ to belong to the set that satisfies the laplacian constraints and λ to belong to the set satisfying the spectral properties, but solving this problem is not easy, hence three formulations have been suggested below.

6.4.1 Formulation 1: Structure via Laplacian spectral constraints

$$\underset{\Theta}{maximise}\; log\,(gdet(\Theta)) - tr(\mathbf{\Theta S}) - \alpha h(\Theta)$$

$$subject\, to\; \Theta \in S_\Theta, \Theta = UDiag(\lambda)U^T, \lambda(T(\Theta)) \in S_T,\; U^TU = I$$

where $\lambda = [\lambda_1 ... \lambda_p]$ and U is the matrix of eigenvectors. This problem formulation can solve various graph structures depending upon the $h(\Theta)$ function. Some of those graphs are,

1. Connected Sparse Graphs with $S_\lambda = [\lambda_1 = 0, \lambda_2 \geq \lambda_1 ... \lambda_n \geq \lambda_{n-1}]$
2. k-component graph with $S_\lambda = [\{\lambda\}_1^k = 0, \lambda_{k+1} \geq \lambda_k ... \lambda_n \geq \lambda_{n-1}]$
3. d-regular graph $S_\lambda = [\{\lambda\}_1^k = 0, c_1 \leq \lambda_{k+1} \leq \lambda_k ... \lambda_n \leq c_2]$ and $Diag(\Theta) = d.I$
4. Connected unweighted graph, cycle graph, star graph, minimum edge weighted tree, hypercube graph, maximum spanning tree, etc.

6.4.2 Formulation 2: Structure via Adjacency spectral constraints

We define an operator A that maps the laplacian matrix Θ to the corresponding adjacency matrix Θ_A.

$$[A(\Theta)]_{ij} = -[\Theta_{ij}]\, i \neq j\ 0\ i = j$$

$$\underset{\Theta,\psi,V}{maximise}\ log\, gdet(\Theta) - tr(\mathbf{\Theta S}) - \alpha h(\Theta)$$

$$subject\, to\ \Theta \in S_\Theta,\ A(\Theta) = V Diag(\psi) V^T,\ \psi \in S_\psi,\ V^T V = I$$

$S_\psi = \{\psi_i = -\psi_{p-i+1} \forall i = 1...p\}$. The graphs that can be solved using the ψ formualtion are,

1. Bipartite Graph
2. Connected Bipartite Graph

6.4.3 Formulation 3: Structure via joint spectral constraints

$$\underset{\Theta,\lambda,\psi,U,V}{maximise}\ log\, gdet(\Theta) - tr(\mathbf{\Theta S}) - \alpha h(\Theta)$$

$$subject\, to\ \Theta \in S_\Theta,\ A(\Theta) = V Diag(\psi) V^T,\ \psi \in S_\psi,\ V^T V = I,$$
$$\Theta = U Diag(\lambda) U^T,\ \lambda(T(\Theta)) \in S_T,\ U^T U = I$$

The structures that can be solved using this kind of approach are,

1. Regular Graph $Diag(\lambda) = Diag(d.\mathbf{1}) - Diag(\psi)$
2. k-component bipartite graph $S_\lambda = [\{\lambda\}_1^k = 0, \lambda_{k+1} \geq \lambda_k ... \lambda_n \geq \lambda_{n-1}]$ and $S_\psi = \{\psi_i = -\psi_{p-i+1} \forall i = 1...p\}$
3. Bipartite Regular Graph $S_\lambda = [\{\lambda\}_1^k = 0, c_1 \leq \lambda_{k+1} \leq \lambda_k ... \lambda_n \leq c_2]$ and$S_\psi = \{\psi_i = -\psi_{p-i+1} \forall i = 1...p\}$

7 References

1. On Spectral Clustering: Analysis and an algorithm by Andrew Ng, Michael Jordan, Yair Weiss, https://ai.stanford.edu/ ang/papers/nips01-spectral.pdf

2. Lecture 31 taught by Prof Sandeep Kumar

3. A Unified Framework for Structured Graph Learning via Spectral Constraints With: J. Ying, Ze. Vinicius, and D. Palomar , Journal of Machine Learning Research (JMLR), Vol. 21, no. 22, pp. 1-60, January 2020