Lecture 11: Review of nonlinear geometric graph modeling

Radu Balan

Department of Mathematics, AMSC, CSCAMM and NWC
University of Maryland, College Park, MD

May 5, 2020

Main Problem

Main Problem

Given a weighted graph $G = (\mathcal{V}, W)$ with n nodes, find a dimension d and a set of n points $\{y_1, \cdots, y_n\} \subset \mathbb{R}^d$ such that $W_{i,j} = \varphi(\|y_i - y_j\|)$ for some monotonically decreasing function φ. Additionally test how a random graph model explains the data.

Main Problem

Main Problem

Given a weighted graph $G = (\mathcal{V}, W)$ with n nodes, find a dimension d and a set of n points $\{y_1, \cdots, y_n\} \subset \mathbb{R}^d$ such that $W_{i,j} = \varphi(\|y_i - y_j\|)$ for some monotonically decreasing function φ. Additionally test how a random graph model explains the data.

Thus we look for a dimension $d > 0$ and a set of points $\{y_1, y_2, \cdots, y_n\} \subset \mathbb{R}^d$ so that all $d_{i,j} = \|y_i - y_j\|$'s are compatible with weighted graph.

Typical weight functions:

1. *Exponential model*: $\varphi(t) = Ce^{-t^2}$, for some $C > 0$.
2. *Power law*: $\varphi(t) = \frac{C}{t^p}$, for some $C > 0$ and $p > 0$.

Analysis

Three studies need to be done:

1. *Random graph hypothesis*: Sort edges by weight: from the largest weight to the smallest weight. Then compare sample statistics of 3-cliques, 4-cliques and spectral gap to the expected ones for $\mathcal{G}_{n,p}$ and $\Gamma^{n,m}$.

Analysis

Three studies need to be done:

1. *Random graph hypothesis*: Sort edges by weight: from the largest weight to the smallest weight. Then compare sample statistics of 3-cliques, 4-cliques and spectral gap to the expected ones for $\mathcal{G}_{n,p}$ and $\Gamma^{n,m}$.
2. *SDP optimization approach*: Solve for the Gram matrix G that optimizes a Semi-Definite Program; Find its effective rank and then perform the SVD of $G = Y^T Y$ to find the geometric graph.

Analysis

Three studies need to be done:

1. *Random graph hypothesis*: Sort edges by weight: from the largest weight to the smallest weight. Then compare sample statistics of 3-cliques, 4-cliques and spectral gap to the expected ones for $\mathcal{G}_{n,p}$ and $\Gamma^{n,m}$.
2. *SDP optimization approach*: Solve for the Gram matrix G that optimizes a Semi-Definite Program; Find its effective rank and then perform the SVD of $G = Y^T Y$ to find the geometric graph.
3. *Laplacian eigenmaps*: The geometric graph is obtained by solving the bottom $d+1$ eigenproblems for the normalized symmetric Laplacian $\tilde{\Delta} = I - D^{-1/} W D^{-1/2}$.

Distribution of Cliques
Expected Values

Let X_q denote the number of q-cliques in a random graph G. Then the expectation of X_q in $\mathcal{G}_{n,p}$ class is

$$\mathbb{E}[X_q] = \left(\begin{array}{c} n \\ q \end{array}\right) p^{q(q-1)/2}$$

The expectation of X_q in the class $\Gamma^{n,m}$ is approximated by the above formula for $p = \frac{2m}{n(n-1)}$:

$$\mathbb{E}[X_q] \approx \left(\begin{array}{c} n \\ q \end{array}\right) \left(\frac{2m}{n(n-1)}\right)^{q(q-1)/2} \sim \theta_q \frac{m^{q(q-1)/2}}{n^{q(q-2)}}$$

$$\mathbb{E}[X_3] \sim \theta \frac{m^3}{n^3} \quad , \quad \mathbb{E}[X_4] \sim \theta \frac{m^6}{n^8}$$

3-Cliques and 4-cliques

Thresholds

Theorem

Let $m = m(n)$ be the number of edges in $\Gamma^{n,m}$.

1. *If $m \gg n$ (i.e. $\lim_{n\to\infty} \frac{m}{n} = \infty$) then* $\lim_{n\to\infty} Prob[G \in \Gamma^{n,m}\ has\ a\ 3-clique] \to 1$.
2. *If $m \ll n$ (i.e. $\lim_{n\to\infty} \frac{m}{n} = 0$) then* $\lim_{n\to\infty} Prob[G \in \Gamma^{n,m}\ has\ a\ 3-clique] \to 0$.

Theorem

Let $m = m(n)$ be the number of edges in $\Gamma^{n,m}$.

1. *If $m \gg n^{4/3}$ (i.e. $\lim_{n\to\infty} \frac{m}{n^{4/3}} = \infty$) then* $\lim_{n\to\infty} Prob[G \in \Gamma^{n,m}\ has\ a\ 4-clique] \to 1$.
2. *If $m \ll n^{4/3}$ (i.e. $\lim_{n\to\infty} \frac{m}{n^{4/3}} = 0$) then* $\lim_{n\to\infty} Prob[G \in \Gamma^{n,m}\ has\ a\ 4-clique] \to 0$.

3-Cliques and 4-Cliques

Behavior at the threshold

In general we obtain a "coarse threshold". Recall a Poisson process X with parameter λ has p.m.f. $Prob[X = k] = e^{-\lambda}\frac{\lambda^k}{k!}$.

Theorem

In $\mathcal{G}_{n,p}$,

1. *For $p = \frac{c}{n}$, X_3 is asymptotically Poisson with parameter $\lambda = c^3/6$.*
2. *For $p = \frac{c}{n^{2/3}}$, X_4 is asymptotically Poisson with parameter $\lambda = c^6/24$.*

3-Cliques and 4-Cliques

Behavior at the threshold

In general we obtain a "coarse threshold". Recall a Poisson process X with parameter λ has p.m.f. $Prob[X=k]=e^{-\lambda}\frac{\lambda^k}{k!}$.

Theorem

In $\mathcal{G}_{n,p}$,

1. *For $p=\frac{c}{n}$, X_3 is asymptotically Poisson with parameter $\lambda=c^3/6$.*
2. *For $p=\frac{c}{n^{2/3}}$, X_4 is asymptotically Poisson with parameter $\lambda=c^6/24$.*

Theorem

In $\Gamma^{n,m}$,

1. *For $m=cn$, X_3 is asymptotically Poisson with parameter $\lambda=4c^3/3$.*
2. *For $m=cn^{4/3}$, X_4 is asymptotically Poisson with parameter $\lambda=8c^6/3$.*

Eigenvalues of Laplacians

$\Delta, L, \tilde{\Delta}$

What do we know about the set of eigenvalues of these matrices for a graph G with n vertices?

1. $\Delta = \Delta^T \geq 0$ and hence its eigenvalues are non-negative real numbers.
2. $eigs(\tilde{\Delta}) = eigs(L) \subset [0, 2]$.
3. 0 is always an eigenvalue and its multiplicity equals the number of connected components of G,

$$dim\, ker(\Delta) = dim\, ker(L) = dim\, ker(\tilde{\Delta}) = \#connected\ \ components.$$

Eigenvalues of Laplacians

$\Delta, L, \tilde{\Delta}$

What do we know about the set of eigenvalues of these matrices for a graph G with n vertices?

1. $\Delta = \Delta^T \geq 0$ and hence its eigenvalues are non-negative real numbers.
2. $eigs(\tilde{\Delta}) = eigs(L) \subset [0, 2]$.
3. 0 is always an eigenvalue and its multiplicity equals the number of connected components of G,

$$dim\, ker(\Delta) = dim\, ker(L) = dim\, ker(\tilde{\Delta}) = \#connected\ \ components.$$

Let $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{n-1}$ be the eigenvalues of $\tilde{\Delta}$. Denote

$$\lambda(G) = \max_{1 \leq i \leq n-1} |1 - \lambda_i|.$$

Note $\sum_{i=1}^{n-1} \lambda_i = trace(\tilde{\Delta}) = n$. Hence the average eigenvalue is about 1. $\lambda(G)$ is called *the absolute gap* and measures the spread of eigenvalues away from 1.

The spectral absolute gap
$\lambda(G)$

The main result in [9]) says that for connected graphs w/h.p.:

$$\lambda_1 \geq 1 - \frac{C}{\sqrt{\text{Average Degree}}} = 1 - \frac{C}{\sqrt{p(n-1)}} = 1 - C\sqrt{\frac{n}{2m}}.$$

Theorem (For class $\mathcal{G}_{n,p}$)

Fix $\delta > 0$ and let $p > (\frac{1}{2} + \delta)log(n)/n$. Let $d = p(n-1)$ denote the expected degree of a vertex. Let $\tilde{G}$ be the giant component of the Erdös-Rényi graph. For every fixed $\varepsilon > 0$, there is a constant $C = C(\delta, \varepsilon)$, so that

$$\lambda(\tilde{G}) \leq \frac{C}{\sqrt{d}}$$

with probability at least $1 - Cn\exp(-(2-\varepsilon)d) - C\exp(-d^{1/4}log(n))$.

Connectivity threshold: $p \sim \frac{log(n)}{n}$.

The spectral absolute gap
$\lambda(G)$

The main result in [9] says that for connected graphs w/h.p.:

$$\lambda_1 \geq 1 - \frac{C}{\sqrt{\text{Average Degree}}} = 1 - \frac{C}{\sqrt{p(n-1)}} = 1 - C\sqrt{\frac{n}{2m}}.$$

Theorem (For class $\Gamma^{n,m}$)

Fix $\delta > 0$ and let $m > \frac{1}{2}(\frac{1}{2}+\delta)n\log(n)$. Let $d = \frac{2m}{n}$ denote the expected degree of a vertex. Let $\tilde{G}$ be the giant component of the Erdös-Rényi graph. For every fixed $\varepsilon > 0$, there is a constant $C = C(\delta,\varepsilon)$, so that

$$\lambda(\tilde{G}) \leq \frac{C}{\sqrt{d}}$$

with probability at least $1 - Cn\exp(-(2-\varepsilon)d) - C\exp(-d^{1/4}\log(n))$.

Connectivity threshold: $m \sim \frac{1}{2}n\log(n)$.

Isometric Embeddings with Partial Data

Linear constraints

Given any set of vectors $\{y_1, \cdots, y_n\}$ and their associated matrix $Y = [y_1|\cdots|y_n]$ their invariant to the action of the rigid transformations (translations, rotations, and reflections) is the Gram matrix of the centered system:

$$G = (I - \frac{1}{n}1 \cdot 1^T)Y^T Y(I - \frac{1}{n}1 \cdot 1^T) =: LY^T YL \quad , \quad L = I - \frac{1}{n}1 \cdot 1^T.$$

On the other hand, the distance between points i and j can be computed by:

$$d_{i,j}^2 = \|y_i - y_j\|^2 = G_{i,i} - G_{i,j} + G_{j,j} - G_{j,i} = e_{ij}^T G e_{ij}$$

where

$$e_{ij} = \delta_i - \delta_j = [0 \cdots 0\,1 \cdots -1\,0 \cdots 0]^T$$

where 1 is on position i, -1 is on position j, and 0 everywhere else.

Almost Isometric Embeddings with Partial Data

The SDP Problem

Reference [10] proposes to find the matrix G by solving the following Semi-Definite Program:

$$\begin{array}{cc} \min\limits_{\substack{G = G^T \geq 0 \\ G \cdot 1 = 0 \\ |\langle Ge_{ij}, e_{ij}\rangle - \tilde{d}^2_{i,j}| \leq \varepsilon \ , \ (i,j) \in \Theta}} & trace(G) \end{array}$$

where $\tilde{d}^2_{i,j}$ are noisy estimates $d_{i,j}$ and ε is the maximum noise level. The trace promotes low rank in this optimization. However, this is basically a feasibility problem: Decrease ε to the minimum value where a feasible solution exists. With probability 1 that is unique.

How to do this: Use CVX with Matlab.

Geometric Graph Embedding

Gram matrix factorization: The Algorithm

Algorithm

Input: Symmetric $n \times n$ Gram matrix G.

1. *Compute the eigendecomposition of G, $G = Q\Lambda Q^T$ with diagonal of Λ sorted in a descending order;*
2. *Determine the number d of significant positive eigevalues;*
3. *Partition*

$$Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \text{ , and } \Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix}$$

   where Q_1 contains the first d columns of Q, and Λ_1 is the $d \times d$ diagonal matrix of significant positive eigenvalues of G.

4. *Compute:*

$$Y = \Lambda_1^{1/2} Q_1^T$$

Output: Dimension d and $d \times n$ matrix Y of vectors $Y = [y_1 | \cdots | y_n]$

Nearly Isometric Embeddings with Partial Data

Stability to Noise

[10] proves the following stability result in the case of partial measurements. Here we denote $\Theta_r = \{(i,j)\ ,\ \|y_i - y_j\| \leq r\}$ the set of all pairs of points at distance at most r.

Theorem

Let $\{y_1, \cdots, y_n\}$ be n nodes distributed uniformly at random in the hypercube $[-0.5, 0.5]^d$. Further, assume that we are given noisy measurement of all distances in Θ_r for some $r \geq 10\sqrt{d}(\log(n)/n)^{1/d}$ and the induced geometric graph of edges is connected. Let $\tilde{d}^2_{i,j} = d^2_{i,j} + \nu_{i,j}$ with $|\nu_{i,j}| \leq \varepsilon$. Then with high probability, the error distance between the estimated $\hat{Y} = [\hat{y}_1, | \cdots | \hat{y}_n]$ returned by the SDP-based algorithm and the correct coordinate matrix $Y = [y_1 | \cdots | y_n]$ is upper bounded as

$$\|L\hat{Y}^T\hat{Y}L - LY^TYL\|_1 \leq C_1(nr^d)^5 \frac{\varepsilon}{r^4}.$$

Optimization Criterion

Assume $\mathcal{G} = (\mathcal{V}, W)$ is a undirected weighted graph with n nodes and weight matrix W.
We interpret $W_{i,j}$ as the *similarity* between nodes i and j. The larger the weight the more similar the nodes, and the closer they are in a geometric graph embedding.
Thus we look for a dimension $d > 0$ and a set of points $\{y_1, y_2, \cdots, y_n\} \subset \mathbb{R}^d$ so that $d_{i,j} = \|y_i - y_j\|$'s is small for large weight $W_{i,j}$. This means we want to minimize

$$J(y_1, y_2, \cdots, y_n) = \sum_{1 \leq i,j \leq n} W_{i,j} \|y_i - y_j\|^2,$$

To avoid trivial solution $Y = 0$ we impose a normalization condition:

$$YDY^T = I_d.$$

The Optimization Problem

Combining the criterion with the constraint:

$$(LE) \; : \quad \begin{array}{ll} \text{minimize} & trace\left\{Y\Delta Y^T\right\} \\ \text{subject to} & YDY^T = I_d \end{array}$$

we obtained the *Laplacian Eigenmap* problem.

Good news: The optimizer Y is obtaind by solving an eigenproblem.

Laplacian Eigenmaps Embedding

Algorithm

Algorithm (Laplacian Eigenmaps)

Input: Weight matrix W, target dimension d

1. *Construct the diagonal matrix* $D = diag(D_{ii})_{1 \leq i \leq n}$*, where* $D_{ii} = \sum_{k=1}^{n} W_{i,k}$.
2. *Construct the normalized Laplacian* $\tilde{\Delta} = I - D^{-1/2} W D^{-1/2}$.
3. *Compute the bottom* $d+1$ *eigenvectors* $e_1, \cdots, e_{d+1}$, $\tilde{\Delta} e_k = \lambda_k e_k$, $0 = \lambda_1 \cdots \lambda_{d+1}$.

Laplacian Eigenmaps Embedding

Algorithm-cont's

Algorithm (Laplacian Eigenmaps - cont'd)

4. *Construct the $d \times n$ matrix Y,*

$$Y = \begin{bmatrix} e_2 \\ \vdots \\ e_{d+1} \end{bmatrix} D^{-1/2}$$

5. *The new geometric graph is obtained by converting the columns of Y into n d-dimensional vectors:*

$$\begin{bmatrix} y_1 & | & \cdots & | & y_n \end{bmatrix} = Y$$

Output: Set of points $\{y_1, y_2, \cdots, y_n\} \subset \mathbb{R}^d$.

Problem Formulation

Given: It is assumed that we are given a set of points $\{x_1, \cdots, x_n\} \subset \mathbb{R}^N$, or a weight matrix W, where $W_{i,j}$ is inverse monotonically dependent to distances $\|x_i - x_j\|$; the smaller the distance $\|x_i - x_j\|$ the larger the weight $W_{i,j}$.
Target: We look for a dimension $d > 0$ and a set of points $\{y_1, y_2, \cdots, y_n\} \subset \mathbb{R}^d$ so that all $d_{i,j} = \|y_i - y_j\|$'s are compatible with the raw data.
Approaches:

1. Principal Component Analysis
2. Independent Component Analysis
3. Laplacian Eigenmaps
4. Local Linear Embeddings (LLE)
5. Isomaps

Principal Component Analysis

Algorithm

Algorithm (Principal Component Analysis)

Input: Data vectors $\{x_1, \cdots, x_n\} \in \mathbb{R}^N$*; dimension* d*.*

0. *If affine subspace is the goal, append '1' at the end of each data vector.*
1. *Compute the sample covariance matrix*

$$R = \sum_{k=1}^{n} x_k x_k^T$$

2. *Solve the eigenproblems* $Re_k = \sigma_k^2 e_k$, $1 \leq k \leq N$, *order eigenvalues* $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_N^2$ *and normalize the eigenvectors* $\|e_k\|_2 = 1$.

Principal Component Analysis

Algorithm - cont'ed

Algorithm (Principal Component Analysis)

3. *Construct the co-isometry*

$$U = \begin{bmatrix} e_1^T \\ \vdots \\ e_d^T \end{bmatrix}.$$

4. *Project the input data*

$$y_1 = Ux_1 \ , \ y_2 = Ux_2 \ , \ \cdots \ , \ y_n = Ux_n.$$

Output: Lower dimensional data vectors $\{y_1, \cdots, y_n\} \in \mathbb{R}^d$.

The orthogonal projection is given by $P = \sum_{k=1}^{d} e_k e_k^T$ and the optimal subspace is $V = Ran(P)$.

Dimension Reduction using Laplacian Eigenmaps

Algorithm

Algorithm (Dimension Reduction using Laplacian Eigenmaps)

Input: A geometric graph $\{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}^N$. *Parameters: threshold* τ, *weight coefficient* α, *and dimension* d.

1. *Compute the set of pairwise distances* $\|x_i - x_j\|$ *and convert them into a set of weights:*
$$W_{i,j} = \begin{cases} exp(-\alpha\|x_i - x_j\|^2) & if \quad \|x_i - x_j\| \leq \tau \\ 0 & if \quad otherwise \end{cases}$$
2. *Compute the* $d+1$ *bottom eigenvectors of the normalized Laplacian matrix* $\tilde{\Delta} = I - D^{-1/2} W D^{-1/2}$, $\tilde{\Delta} e_k = \lambda_k e_k$, $1 \leq k \leq d+1$, $0 = \lambda_0 \leq \cdots \leq \lambda_{d+1}$, *where* $D = diag(\sum_{k=1}^n W_{i,k})_{1 \leq i \leq n}$.

Dimension Reduction using Laplacian Eigenmaps

Algorithm - cont'd

Algorithm (Dimension Reduction using Laplacian Eigenmaps-cont'd)

3. *Construct the* $d \times n$ *matrix* Y*,*

$$Y = \begin{bmatrix} e_2^T \\ \vdots \\ e_{d+1}^T \end{bmatrix} D^{-1/2}$$

4. *The new geometric graph is obtained by converting the columns of* Y *into* n d*-dimensional vectors:*

$$\begin{bmatrix} y_1 & | & \cdots & | & y_n \end{bmatrix} = Y$$

Output: $\{y_1, \cdots, y_n\} \subset \mathbb{R}^d$.

Dimension Reduction using Isomaps
Algorithm

Algorithm (Dimension Reduction using Isomap)

Input: A geometric graph $\{x_1, x_2, \cdots, x_n\} \subset \mathbb{R}^N$. *Parameters: neighborhood size* K *and dimension* d.

1. *Construct the symmetric matrix* S *of squared pairwise distances:*
   1. *Construct the sparse matrix* T, *where for each node* i *find the nearest* K *neighbors* $\mathcal{V}_i$ *and set* $T_{i,j} = \|x_i - x_j\|_2$, $j \in \mathcal{V}_i$.
   2. *For any pair of two nodes* (i, j) *compute* $d_{i,j}$ *as the length of the shortest path,* $\sum_{p=1}^{L} T_{k_{p-1},k_p}$ *with* $k_0 = i$ *and* $k_L = j$, *using e.g. Dijkstra's algorithm.*
   3. *Set* $S_{i,j} = d_{i,j}^2$.

Dimension Reduction using Isomaps

Algorithm - cont'd

Algorithm (Dimension Reduction using Isomap - cont'd)

2. *Compute the Gram matrix G:*

$$\rho = \frac{1}{2n} 1^T \cdot S \cdot 1 \ , \quad \nu = \frac{1}{n}(S \cdot 1 - \rho 1)$$

$$G = \frac{1}{2}\nu \cdot 1^T + \frac{1}{2} 1 \cdot \nu^T - \frac{1}{2} S$$

3. *Find the top d eigenvectors of G, say $e_1, \cdots, e_d$ so that $GE = E\Lambda$, form the matrix Y and then collect the columns:*

$$Y = \Lambda^{1/2} \begin{bmatrix} e_1^T \\ \vdots \\ e_d^T \end{bmatrix} = \begin{bmatrix} y_1 & | & \cdots & | & y_n \end{bmatrix}$$

Output: $\{y_1, \cdots, y_n\} \subset \mathbb{R}^d$.

References

- B. Bollobás, **Graph Theory. An Introductory Course**, Springer-Verlag 1979. **99**(25), 15879–15882 (2002).
- S. Boyd, L. Vandenberghe, **Convex Optimization**, available online at: http://stanford.edu/ boyd/cvxbook/
- F. Chung, **Spectral Graph Theory**, AMS 1997.
- F. Chung, L. Lu, The average distances in random graphs with given expected degrees, Proc. Nat.Acad.Sci. 2002.
- F. Chung, L. Lu, V. Vu, The spectra of random graphs with Given Expected Degrees, Internet Math. **1**(3), 257–275 (2004).
- R. Diestel, **Graph Theory**, 3rd Edition, Springer-Verlag 2005.
- P. Erdös, A. Rényi, On The Evolution of Random Graphs

- G. Grimmett, **Probability on Graphs. Random Processes on Graphs and Lattices**, Cambridge Press 2010.
- C. Hoffman, M. Kahle, E. Paquette, Spectral Gap of Random Graphs and Applications to Random Topology, arXiv: 1201.0425 [math.CO] 17 Sept. 2014.
- A. Javanmard, A. Montanari, Localization from Incomplete Noisy Distance Measurements, arXiv:1103.1417, Nov. 2012; also ISIT 2011.
- J. Leskovec, J. Kleinberg, C. Faloutsos, Graph Evolution: Densification and Shrinking Diameters, ACM Trans. on Knowl.Disc.Data, **1**(1) 2007.
- S.T. Roweis, L.K. Saul, Locally linear embedding, Science 290, 2323–2326 (2000).
- J.B. Tenenbaum, V. de Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290, 2319–2323 (2000).