L23-SNA

Spectral Clustering Algorithm

- 3 BASIC STEPS
  - Pre-processing-matrix construction
  - Decomposition-Eigen vector finding
  - Grouping—Assign points to two or more clusters based on the new representation

Laplacian matrix for *simple graphs*

Given a simple graph *G* with *n* vertices, its Laplacian matrix $L_{n\times n}$ is defined as:[1]

$$L = D - A,$$

where *D* is the degree matrix and *A* is the adjacency matrix of the graph. Since G is a simple graph, A only contains 1s or 0s and its diagonal elements are all 0s. In the case of directed graphs, either the indegree or outdegree might be used, depending on the application. The elements of L are given by

$$L_{i,j} := \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

where deg(v_i) is the degree of the vertex *i*.

Symmetric normalized Laplacian

The symmetric normalized Laplacian matrix is defined as:[1]

$$L^{\text{sym}} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2},$$

The elements of L^{sym} are given by

$$L^{\text{sym}}_{i,j} := \begin{cases} 1 & \text{if } i = j \text{ and } \deg(v_i) \neq 0 \\ -\frac{1}{\sqrt{\deg(v_i)\deg(v_j)}} & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise.} \end{cases}$$

Properties of Laplacian matrix:

- For an undirected graph G and it's Laplacian matrix L with Eigen values $\lambda_0 <= \lambda_1 <= \ldots \lambda_{n-1}$;
- L is symmetric and diagonally dominant
- L is positive semidefinite($\lambda_i >= 0$ for all i)
- L is an M-Matrix (its off diagonal entries are non-positive, yet the real parts of its Eigen values are non-negative)
- Every row sum and column sum of Lis zero. Indeed, in the sum the degree of the vertex is summed with a "1" for each algebraic
- The number of connected components in the graph is the dimension of the null space & the algebraic multiplicity of the 0 Eigen value
- Laplacian matrix singular
- When G is k-regular, the normalized Laplacian in
- L=(1/k)*L= I-(1/k)*A
- Where A is the adjacency matrix, I is the identity matrix

Summary of Spectral Algorithms

- Algorithms that assign nodes to communities based on Eigen vector of matrices , such as the adjacency matrix of the network or related matrices
- Given a graph you build the Laplacian matrix L, find Eigen values lambda λ and Eigen vectors x of the matrix L, then map vertices to corresponding components of λ_2.
- K means can be used at the end for clustering ie to assign the node to the components.

- Disadvantage of spectral algorithm is its computational complexity
- modern implementations for eigenvector computation use iterative algorithms such as the Lanczos algorithm, where at each stage a series of matrix vector multiplications are performed to obtain successive approximations to the eigenvector currently being computed.
- The complexity for computing the top eigenvector is $O(kM(m))$, where k is the number of matrix-vector multiplications and$M(m)$ is the complexity of each such multiplication,
- dependent primarily on the number of non-zeros m in the matrix.
- k depends on the specific properties of the matrix at hand - such as the spectral gap i.e. the difference between the current eigenvalue and the next eigenvalue;
- the smaller this gap, the more number of matrix-vector multiplications are required for convergence.

- In practice, spectral clustering is hard to scale up to networks with more than tens of thousands of vertices without employing parallel algorithms.
- The weighted cut measures such as normalized cut that are often optimized using spectral clustering can also be optimized using an equivalent weighted kernel k-means algorithm.
- This is the core idea behind their algorithm **Graclus**, which can cluster graphs at a comparable quality to spectral clustering without paying the same computational cost, since k-means is much faster compared to eigenvector computation