A Mathematical Introduction to Data Science **December 2, 2014**

Homework 7. Multiple Spectral Clustering

Instructor: Yuan Yao *Due: Tuesday December 16, 2014*

The problem below marked by * is optional with bonus credits.

1. *Degree Corrected Stochastic Block Model (DCSBM)*: A random graph is generated from a DCSBM with respect to partition $\Omega = \{\Omega_k : k = 1, \ldots, K\}$ if its adjacency matrix $A \in \{0,1\}^{N\times N}$ has the following expectation

$$\mathbb{E}[A] = \mathcal{A} = \Theta Z B Z^T \Theta$$

where $Z^{N\times k}$ has row vectors $\in \{0,1\}^K$ as the block membership function $z : V \to \Omega$.

$$z_{ik} = \begin{cases} 1, & i \in \Omega_k, \\ 0, & otherwise. \end{cases}$$

B is a K-by-K symmetric stochastic block probability matrix, here we make it positive definite. You can choose a diagonally dominant matrix, which is often satisfied in real data.

$\Theta = \mathrm{diag}(\theta_i)$ is the expected degree satisfying,

$$\sum_{i\in\Omega_k} \theta_i = |\Omega_k| = n, \quad \forall k = 1, \ldots, K.$$

When $\Theta = I$, it becomes stochastic block model.

Also you need to verify your construction $\mathcal{A}$ is indeed a probability matrix, i.e.

$$\max_{i,j} \theta_i \theta_j B_{z_i z_j} \leq 1$$

The following matlab codes simulate a DCSBM of nK nodes, written by Kaizheng Wang,

`http://www.math.pku.edu.cn/teachers/yaoy/data/DCSBM.m`

Construct a DCSBM yourself($K \geq 3, n \geq 30$), and simulate random graphs of 10 times. Then try to compare the following two spectral clustering methods in finding the K blocks (communities).

Algorithm I [1] Compute the *top* K generalized eigenvector

$$(D - A)\phi_i = \lambda_i D \phi_i,$$

construct a K-dimensional embedding of V using $\Phi^{N\times K} = [\phi_1, \ldots, \phi_K]$;

[2] Run k-means algorithm (call `kmeans` in matlab) on Φ to find K clusters.

Algorithm II [1] Compute the *bottom* K eigenvector of

$$L_n = D^{-1/2}(D-A)D^{-1/2} = U\Lambda U^T,$$

construct an embedding of V using $U^{N\times K}$;

[2] Normalized the row vectors u_{i*} on to the sphere: $\hat{u}_{i*} = u_{i*}/\|u_{i*}\|$;

[3] Run k-means algorithm (call `kmeans` in matlab) on $\hat{U}$ to find K clusters.

You may run it multiple times with a stabler clustering. Suppose the estimated membership function is $\hat{z} : V \to \{1, \ldots, K\}$ in either methods. Compare the performance using mutual information between membership function z and estimate $\hat{z}$,

$$I(z, \hat{z}) = \sum_{s,t=1}^{K} Prob(z_i = s, \hat{z}_i = t) \log \frac{Prob(z_i = s, \hat{z}_i = t)}{Prob(z_i = s)Prob(\hat{z}_i = t)}. \tag{1}$$

A reference matlab code can be found at (thanks to Kaizheng Wang for pointing out this)

`http://www.cse.ust.hk/~weikep/notes/NormalizedMI.m`

2. *A Dream of Red Mansions*: Try the spectral clustering algorithms above to character-character cooccurance network, where the 376-by-475 matrix X of character-event is given, e.g. in .txt file:

   `http://www.math.pku.edu.cn/teachers/yaoy/data/hongloumeng/HongLouMeng374.txt`

   or in the Matlab format

   `http://www.math.pku.edu.cn/teachers/yaoy/data/hongloumeng/hongloumeng376.mat`

   with a readme file:

   `http://www.math.pku.edu.cn/teachers/yaoy/data/hongloumeng/readme.m`

   $*$ How do you decide K in this real-world example?

   Note: all the 'NaN's in the matlab file refer to 0*'s.*