**Discovery Thread: Project 2**

In this project you will apply the techniques for random graphs model selection and community detection on a specific data set.

The following files are assigned to your team:

- sgb128Nodes*to*_coord.txt : Coordinates of a set of 40 points (cities) taken from SGB128 dataset; This text file has the following format:

  ```
  First line: X(1) Y(1) Z(1)
  Second line: X(2) Y(2) Z(2)
  ...
  Last line: X(n)  Y(n)  Z(n)
  ```

  Note: all Z coordinates are 0. You can discard them.

- sgb128Nodes*to*_weight.txt : A symmetric matrix of weights defined by $V(i,j) = exp(-6dist(i,j)/maxD)$, for $i \neq j$, where $dist(i,j)$ is the Euclidean distance between city $i$ and city $j$, and $maxD = max_{i,j}d(i,j)$ is the largest distance in the graph. This text file has the following format:

  ```
  First line: n
  Second line: V(1,1)  V(1,2)  V(1,3)  ...  V(1,n)
  Third line:  V(2,1)  V(2,2)  V(2,3)  ...  V(2,n)
  ...
  Line n+1: V(n,1)  V(n,2)  V(n,3)  ...  V(n,n)
  ```

- sgb128Nodes*to*_weight20.txt : a weight matrix $W$ obtained by threshlding $V$ to 20% of its maximum entry. Thus, if $V(i,j) \geq 0.2max(V)$ then $W(i,j) = V(i,j)$; otherwise $W(i,j) = 0$. Note: there are abut 40-45% non-zero entries. This text file has the following format:

  ```
  First line: n m
  Second line: W(1,1)  W(1,2)  W(1,3)  ...  W(1,n)
  Third line:  W(2,1)  W(2,2)  W(2,3)  ...  W(2,n)
  ...
  Line n+1: W(n,1)  W(n,2)  W(n,3)  ...  W(n,n)
  ```

- sgb128Nodes*to*_adj20.txt : The adjacency matrix $A$ associated to $W$: $A(i,j) = 1$ iff $W(i,j) > 0$. Note: the number of edges is equal to the number of non-zero entries in the upper traingle of $W$; This text file has the following format:

  ```
  First line: n m
  Second line: A(1,1)  A(1,2)  A(1,3)  ...  A(1,n)
  Third line:  A(2,1)  A(2,2)  A(2,3)  ...  A(2,n)
  ...
  Line n+1: A(n,1)  A(n,2)  A(n,3)  ...  A(n,n)
  ```

- sgb128_name.txt: List of names from the SGB128 file. Your cities are NodeX to NodeY where X and Y are taken from the file: sgb128NodesXtoY_coord.txt. Note: there are 128 names; your city names are only city X to city Y

On this dataset perform the following three tasks:
**I.** Random graph model testing: For this task use the full weight matrix $V$.

1. Order edges according to their weight. For this, create a matrix $E$ of size $n(n-1)/2$ x 2 that contains the ordered list of edges so that (E(1,1),E(1,2)) is an edge with the largest weight;
2. Loop over $k$ from 2 to $n(n-1)/2$ and for the set of edges $E(1:k, 1:2)$:
   (a) compute the actual number of 3-cliques $q3(k)$ and 4-cliques $q4(k)$;
   (b) Under the Erdos-Renyi random graph model, estimate the parameter $p$. Compute the estimated number of 3-cliques and 4-cliques (under the Erdos-Renyi model), say $ER3(k)$ and $ER4(k)$;
   (c) Under the SSBM random graph model, estimate the parameters $a$ and $b$ based on the number of vertices, edges, and 3-cliques, using the Modified Constrained Moment Matching Algorithm 2. Compute the estimated number of 3-cliques and 4-cliques (under the SSBM model), say $SSBM3(k)$ and $SSBM4(k)$;
3. Plot $q3$, $ER3$ and $SSBM3$ on the same plot. Estimate the amplitude $C$ and exponent $r$ from the power law $y(k) \sim Ck^r$ by a linear fit in the log-log plot, after you discard the first, say 10 entries. Call $C_{3,ER}$, $r_{3,ER}$ and $C_{3,SSBM}$, $r_{3,SSBM}$ the respective parameters.
4. Plot $log(q3)$, $log(ER3)$ and $log(C_{3,ER}) + r_{3,ER}log(k)$ on same figure over the range of $k$ utilized to estimate the exponent.
5. Plot $log(q3)$, $log(SSBM3)$ and $log(C_{3,SSBM}) + r_{3,SSBM}log(k)$ on same figure over the range of $k$ utilized to estimate the exponent.
6. Plot $q4$, $ER4$ and $SSBM4$ on the same plot. Estimate the exponent $r$ from the power law $y(k) \sim Ck^r$ by a linear fit in the log-log plot, after you discard, say 100 first entries. Call $C_{4,ER}$, $r_{4,ER}$ and $C_{4,SSBM}$, $r_{4,SSBM}$ the respective parameters.
7. Plot $log(q4)$, $log(ER4)$ and $log(C_{4,ER}) + r_{4,ER}log(k)$ on same figure over the range of $k$ utilized to estimate the exponent.
8. Plot $log(q4)$, $log(SSBM4)$ and $log(C_{4,SSBM}) + r_{4,SSBM}log(k)$ on same figure over the range of $k$ utilized to estimate the exponent.

Which of the two random graph model fits better the data? Why do you think I recommend to discard the first 10 or 100 entries?

**II.** Community detection: For this task use the weight matrix $W$ and the adjacency matrix $A$.

Implement the six community discovery algorithms (partition algorithms) and run them on your project data set.

Specifically, implement:

- Spectral methods using: $W$, $\Delta$, and $\tilde{\Delta}$
- SDP relaxation algorithms using: $W$, $\Delta$, and $\tilde{\Delta}$

1. For each of the six algorithms above, determine sets $S$ and $\bar{S} = \{1, 2, \ldots, n\} \setminus S$.
2. Compute the agreement matrix between these partitions: The output whould be a $6 \times 6$ matrix $Agr$ so that $Agr(k, l)$ represents the partition agreement between method $k$ and method $l$, $1 \leq k, l \leq 6$, the 6 methods above.
3. For visualization, for each of the six algorithms, map the two communities using two colors, say red and blue, using the coordinates $(X, Y)$ from from the coordinate file assigned to your project. For each algorithm produce two figures as follows:
   (a) Draw edges according to the adjacency matrix $A$, each edge with same color and same width;
   (b) Draw edges according to the weight matrix $W$, each edge with same color and but different width, the larger the weight, the thicker the edge.

**III.** Data Embedding For this task use the weight matrix $W$.

Implement the Laplacian Eigenmap and the Local Linear Embeding (LLE) algorithms using the weight matrix $W$, and run them on your project data set.

Specifically, implement and run:

1. Laplacian Eigenmap data embedding for target dimension $d = 2$;
2. LLE dimension reduction after Laplacian Eigenmap data embedding:
   (a) First run the Laplacian Eigenmap data embedding algorithm to create a geometric graph $\{x_1, \ldots, x_n\} \subset \mathbb{R}^N$ with $N = 10$;
   (b) Then implement and run the dimension reduction LLE algorithm with non-negativity constraints on the this geometric graph to reduce dimension to $d = 2$; use $K = 2d = 4$.

Plot both embeddings in two different figures, and then on the same figure using different colors.