Random Graph theory

Benjamin Chu

March 10, 2020

The electronic version of this document is available at:

`https://github.com/biona001/teaching`

under `preceptorship - biomath 203/random graph theory`. Code for chapter 3 is available as `kmeans.ipynb` for running the accompanying Julia code. If you do not know how to use Jupyter notebooks, you can view the code and figures from the accompanying `kmeans.html` file.

1 Basics of Graph Theory

- A graph G is a pair of sets $G = (V, E)$ where V is a set of vertices, E is a set of edges, and $e \in E$ can be written as $e = (x, y)$ with $x, y \in V$.
- It is comon to represent a graph by *drawing*. Each vertex $v \in V$ is represented as a point in the plane, while edges are lines connecting pairs of points.

There are a number of special graphs, which we will only mention.

- A graph with n nodes is **complete** (denoted by K_n) if every node forms an edge with every other node.
- A **cycle graph** (denoted by C_n) is a graph that consists of nodes connected in a closed chain. The degree of each vertex is 2.
- A **tree** is a connected graph with no cycles.

The following theorem will get you started with the basics of graph theory.

Theorem 1.1 First theorem of graph theory

A finite graph G has an even number of vertices with odd **degree** (i.e. the number of edges incident to it).

Proof. Since each edge connects 2 nodes,

$$2|E| = \sum_{v \in V} deg(v) = \sum_{\substack{v \in V \\ deg(v) \text{ even}}} deg(v) + \sum_{\substack{u \in V \\ deg(u) \text{ odd}}} deg(u) \implies \left(\sum_{\substack{u \in V \\ deg(u) \text{ odd}}} deg(u) \right) \text{ is even.}$$

If the sum is even, and each summand is odd, then there must be an even number of summands. □

2 Sharp Threshold for Connectivity in Erdos-Renyi Graph Model

Most materials for this section note is taken from [1, 4]. First some background:

- We use $G(n,p)$ to denote an undirected (Erdos-Renyi) graph with n nodes and probability of forming an edge $p(n)$.
- Each edge forms with probability $p \in (0,1)$ **independently** of other edges.
- An graph is **connected** if there is a path between any 2 pairs of nodes.
- When $p = p(n)$ is a function of n, we may be interested in the behavior of $G(n,p(n))$ as $n \to \infty$.

2.1 Warm-up

Q1. What is the probability that a vertex is isolated in $G(n,p)$? Ans: A given node i cannot form an edge with each of the remaining $n-1$ nodes. Thus the probability is $(1-p)^{n-1}$.

Q2. What is the probability that node 1 and node 2 are both isolated? Ans: Let I_1, I_2 be the indicator that node 1 and node 2 are isolated. Then $P(I_1 \cap I_2) = P(I_1)P(I_2 \mid I_1) = (1-p)^{n-1} * (1-p)^{n-2} = (1-p)^{2n-3}$.

Q3. What is the probability that a specific set of k nodes is not connected to the rest? Ans: Given the k nodes, each of them cannot connect to the remaining $n-k$ nodes with probability $(1-p)^{n-k}$. Sp the answer is $(1-p)^{(n-k)k}$

Theorem 2.1 Erdos-Renyi 1961

Consider a graph $g \sim G(n,p(n))$ where $p(n) = \lambda \frac{\ln(n)}{n}$. Then as $n \to \infty$,

$$P(g \text{ connected}) \to 0 \quad \text{if } \lambda < 1$$
$$P(g \text{ connected}) \to 1 \quad \text{if } \lambda > 1$$

Proof. Suppose $\lambda < 1$. Since $P(\text{connected}) = 1 - P(\text{disconnected})$, we will show $P(\text{disconnected}) \to 1$ by showing that **there is at least 1 isolated node**. Define

- X_n to be a random variable that counts the number of isolated nodes
- I_i to be a (Bernoulli) indicator random variable such that $I_i = 1$ when node i is isolated and is 0 otherwise
- Let $p = p(n)$ and $q = q(n) = (1-p(n))^{n-1}$ be the probability of a node being isolated

We want to show $P(X_n > 0) \to 1$, or equivalently, $P(X_n = 0) \to 0$. To get a bound on $P(X_n = 0)$, we observe:

$$\begin{aligned}\text{Var}(X_n) &= E\left[(X_n - E(X_n))^2\right] \\ &= P(X_n = 0)(0 - E(X_n))^2 + P(X_n = 1)(1 - E(X_n))^2 + \ldots \\ &\geq P(X_n = 0)E(X_n)^2.\end{aligned}$$

Thus

$$\frac{\text{Var}(X_n)}{E(X_n)^2} \geq P(X_n = 0). \tag{2.1}$$

We will now calculate $\text{Var}(X_n)$ and $E(X_n)$ explicitly to show that the left hand side of (2.1) goes to 0. By linearity of expectation and applying definition of indicators,

$$E(X_n) = E\left(\sum_{i=1}^{n} I_i\right) = \sum_{i=1}^{n} E(I_i) = \sum_{i=1}^{n} P(I_i) = nq.$$

Since indicators I_i are **not independent** (why?), we use equation (1.10) in your book [3]:

$$\begin{aligned}\text{Var}(X_n) &= \text{Var}\left(\sum_{i=1}^{n} I_i\right) = \sum_{i=1}^{n} \text{Var}(I_i) + \sum_{i=1}^{n}\sum_{j\neq i} \text{Cov}(I_i, I_j) \\ &= \sum_{i=1}^{n} q(1-q) + \sum_{i=1}^{n}\sum_{j\neq i} [E(I_i I_j) - E(I_i)E(I_j)] \quad (\text{since Var(Bernoulli)} = p(1-p)) \\ &= nq(1-q) + \sum_{i=1}^{n}\sum_{j\neq i} [P(I_i \cap I_j) - P(I_i)P(I_j)] \\ &= nq(1-q) + \sum_{i=1}^{n}\sum_{j\neq i} \left[(1-p)^{n-1}(1-p)^{n-2} - (1-p)^{n-1}(1-p)^{n-1}\right] \\ &= nq(1-q) + \sum_{i=1}^{n}\sum_{j\neq i} \left[\frac{q^2}{1-p} - q^2\right] \\ &= nq(1-q) + n(n-1)q^2\frac{p}{1-p}.\end{aligned}$$

Thus

$$\frac{\text{Var}(X_n)}{E(X_n)^2} = \frac{nq(1-q) + n(n-1)q^2\frac{p}{1-p}}{(nq)^2} = \frac{1-q}{nq} + \frac{n-1}{n}\frac{p}{1-p}.$$

We will now show these terms approach 0 as $n \to \infty$, then eq (2.1) will give us what we need. The first term

is dominated by nq, and

$$\begin{aligned}
\lim_{n\to\infty} nq &= \lim_{n\to\infty} n(1-p)^{n-1} = \lim_{n\to\infty} \exp\{\ln(n) + (n-1)\ln(1-p)\} \\
&= \lim_{n\to\infty} \exp\left\{\ln(n) + (n-1)\ln\left(1 - \frac{\lambda\ln(n)}{n}\right)\right\} \\
&\approx \lim_{n\to\infty} \exp\left\{\ln(n) - \lambda\frac{n-1}{n}\ln(n)\right\} \quad \left(\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots \approx x + O(x^2) \text{ for small } x\right) \\
&= \lim_{n\to\infty} \exp\left\{\ln(n)\left(1 - \lambda\frac{n-1}{n}\right)\right\} \\
&= \infty \qquad (\text{since } \lambda < 1 \text{ and } n \to \infty)
\end{aligned}$$

For the second term, observe that $p = \lambda\frac{\ln(n)}{n} \to 0$ as $n \to \infty$. So $\frac{p}{1-p} \to 0$ as well. This completes the case for $\lambda < 1$.

Part II. Now suppose $\lambda > 1$. We want to show $P(\text{connected}) \to 1$, or equivalently $P(\text{disconnected}) \to 0$. A graph is disconnected if there is a subgraph of k nodes that does not connect to any of the other $n-k$ nodes (draw a picture). By symmetry, we only have to consider $k \in \{1, 2, \ldots \lfloor n/2 \rfloor\}$. So

$$\begin{aligned}
P(\text{disconnected}) &= P\left(\bigcup_{k=1}^{\lfloor n/2\rfloor} \text{some set of } k \text{ nodes not connected to the rest}\right) \\
&\leq \sum_{k=1}^{\lfloor n/2\rfloor} P(\text{some set of } k \text{ nodes not connected to the rest}) \quad (\text{inclusion-exclusion picture}) \\
&\leq \sum_{k=1}^{\lfloor n/2\rfloor} \binom{n}{k} P(\text{a specific set of } k \text{ nodes not connected to the rest}) \\
&= \sum_{k=1}^{\lfloor n/2\rfloor} \binom{n}{k} \left[(1-p)^{(n-k)}\right]^k \\
&\leq \sum_{k=1}^{\lfloor n/2\rfloor} \binom{n}{k} e^{-p(n-k)k} \quad \left(e^{-x} = 1 - x + \frac{x^2}{2} - \ldots \approx 1 - x + O(x^2) \text{ for small } x\right) \\
&= \sum_{k=1}^{\lfloor n/2\rfloor} \binom{n}{k} \exp\left\{\frac{-\lambda\ln(n)(n-k)k}{n}\right\} \\
&= \sum_{k=1}^{\lfloor n/2\rfloor} \binom{n}{k} n^{\frac{-\lambda}{n}(n-k)k} \\
&= \sum_{k=1}^{n^*} \binom{n}{k} n^{\frac{-\lambda}{n}(n-k)k} + \sum_{k=n^*+1}^{\lfloor n/2\rfloor} \binom{n}{k} n^{\frac{-\lambda}{n}(n-k)k} \quad \left(\text{Choose } n^* s.t. \frac{\lambda(n-n^*)}{n} > 1 \iff n^* = \lfloor n(1 - \frac{1}{\lambda})\rfloor\right)
\end{aligned}$$

For the first term,

$$\begin{aligned}
\sum_{k=1}^{n^*}\binom{n}{k}n^{\frac{-\lambda}{n}(n-k)k} &\le \sum_{k=1}^{n^*} n^k n^{\frac{-\lambda}{n}(n-k)k} = \sum_{k=1}^{n^*}\left[n^{1-\frac{\lambda}{n}(n-k)}\right]^k \\
&\le \sum_{k=1}^{n^*}\left[n^{1-\frac{\lambda}{n}(n-n^*)}\right]^k \qquad \text{(judiciously bound inner } k \text{ with something bigger)} \\
&= \sum_{k=1}^{n^*} r^k \qquad \left(\text{define } r = n^{1-\frac{\lambda}{n}(n-n^*)}\right) \\
&= \left(\sum_{k=0}^{n^*} r^k\right) - 1 \\
&= \frac{r}{1-r} \qquad \left(\text{geometric series. } r<1 \text{ since } 1-\frac{\lambda}{n}(n-n^*) < 0\right) \\
&= \frac{1}{n^{\frac{\lambda}{n}(n-n^*)-1}-1} \\
&\longrightarrow 0 \qquad (\text{since } n\to\infty \text{ and exponent} > 0)
\end{aligned}$$

For the second term, we use a better bound than before (see homework):

$$\binom{n}{k} < \left(\frac{en}{k}\right)^k.$$

Thus

$$\begin{aligned}
\sum_{k=n^*+1}^{\lfloor n/2\rfloor}\binom{n}{k}n^{\frac{-\lambda}{n}(n-k)k} &< \sum_{k=n^*+1}^{\lfloor n/2\rfloor}\left(\frac{en}{k}\right)^k n^{\frac{-\lambda(n-k)k}{n}} = \sum_{k=n^*+1}^{\lfloor n/2\rfloor}\left[\frac{en^{1-\frac{\lambda(n-k)}{n}}}{k}\right]^k \\
&\le \sum_{k=n^*+1}^{\lfloor n/2\rfloor}\left[\frac{en^{1-\frac{\lambda(n-\frac{n}{2})}{n}}}{n^*+1}\right]^k \qquad \text{(bound inner } k \text{ with something from above)} \\
&= \sum_{k=n^*+1}^{\lfloor n/2\rfloor}\left[\frac{en^{1-\frac{\lambda}{2}}}{n(1-\frac{1}{\lambda})+1}\right]^k \le \sum_{k=n^*+1}^{\lfloor n/2\rfloor}\left[\frac{en^{\frac{-\lambda}{2}}}{1-\frac{1}{\lambda}}\right]^k \\
&\le \sum_{k=n^*+1}^{\lfloor n/2\rfloor} r^k \qquad \left(r=\frac{en^{\frac{-\lambda}{2}}}{1-\frac{1}{\lambda}}, 0<r<1 \text{ for large } n\right) \\
&\le \sum_{k=n^*+1}^{\infty} r^k = \sum_{k=0}^{\infty} r^k - \sum_{k=0}^{n^*} r^k \\
&= \frac{1}{1-r} - \frac{1-r^{n^*+1}}{1-r} \qquad \left(\text{finite geometric series } \sum_{k=0}^{m} r^k = \frac{1-r^{m+1}}{1-r}\right) \\
&= \frac{r^{n^*+1}}{1-r} \longrightarrow 0 \qquad \text{since } n^*\to\infty.
\end{aligned}$$

□

3 Clustering graphs

Sometimes it is useful to **cluster** a graph, which lumps a graph's nodes into several groups so that there are much more edges within groups than between groups. There are many algorithms to do this (e.g. K-means, hierarchical), which is not our focus. Rather, we will combine random graph theory with (discrete time) Markov chains to define a new distance measure that can be used together with various clustering algorithms. Most material is based on [7].

3.1 Euclidean distance for clustering in k-means algorithm

Clustering algorithms require some measures of similarity (i.e. distance) between 2 nodes. One common distance measure is the Euclidean distance: $d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$. This defines the traditional k-means and hierarchical clustering algorithms.

(review) k-means algorithm:

(1) Randomly initialize k cluster centers $c_1, ...c_k$.

(2) Assign each node to some cluster based on the smallest **Euclidean distance** to each cluster center c_i.

(3) Update center location $c_1, ..., c_k$ by computing the means of the nodes in the cluster.

(4) Repeat 2 and 3 until assignments no longer change

3.2 Random Walk and Euclidean Commute Time (ETC) Distance

We setup a Markov chain as follows:

- Consider a connected **weighted graph** with N nodes where each edge connecting nodes i and j has a weight $w_{ij} > 0$ that is symmetric $w_{ij} = w_{ji}$.
- Let each node of a graph be a state in a Markov chain.
- For node i, the probability of jumping to an adjacent node j is $p_{ij} = \frac{a_{ij}}{a_{i\cdot}}$ where $a_{i\cdot} = \sum_j a_{ij}$. Hence large w_{ij} values means easier communication through the edge.
- Connectivity implies the Markov chain is irreducible.

Based on this Markov chain, we define 2 important quantities:

- Starting at state i, the **average first passage time**

$$\begin{cases} m(k|k) = 0 \\ m(k|i) = 1 + \sum_{j=1}^{N} p_{ij} m(k|j) \quad \text{if } i \neq k. \end{cases} \tag{3.1}$$

  is the average number of steps a random walker needs to enter state k.

- Starting at state i, the **average commute time**

$$n(i,j) = m(j|i) + m(i|j)$$

is the number of steps a random walker take to enter $j \neq i$ for the first time, then go back to i. One can show that this function is a distance measure.

$\sqrt{n(i,j)}$ is called the **Euclidean Commute Time (ETC) Distance**. Intuitively, $n(i,j)$ decreases when 2 nodes are highly connected, or when the length of any path decrease. The fact that it takes "connectivity" between nodes into account sets it apart from shortest path distances.

3.3 Computation of Euclidean Commute Time (ETC) Distance

3.3.1 Closed form solution involving pseudoinverse

Define

- The **adjacency matrix** $\mathbf{A} = (a_{ij})$ where $a_{ij} = w_{ij}$ if node i is connected to j, otherwise $a_{ij} = 0$
- $\mathbf{D} = \text{diag}(a_{i\cdot})$ where $a_{i\cdot} = \sum_j a_{ij}$
- The **Laplacian matrix** of the graph is $\mathbf{L} = \mathbf{D} - \mathbf{A}$

Note the Laplacian $\mathbf{L}$ is not full rank because $\mathbf{1}$ (vector of 1s) is in its null space. Hence the following theorem involves a pseudoinverse $\mathbf{L}^+$:

Theorem 3.1 Computaton of average commute time $n(i,j)$

We can compute average commute time between nodes i and j by:

$$n(i,j) = V_G(\mathbf{e}_i - \mathbf{e}_j)^t \mathbf{L}^+ (\mathbf{e}_i - \mathbf{e}_j)$$

where $\mathbf{L}^+$ is the Moore-Penrose pseudoinverse of $\mathbf{L}, V_G = \sum_{i,j} a_{ij}$ is the volume of the graph, and $\mathbf{e}_i$'s are the standard basis vectors that is 0 everywhere and is 1 at position i.

Proof. See appendix of [2]. □

Observe that:

(1) $\mathbf{L}^+$ is symmetric since $\mathbf{L}$ is.

(2) $\mathbf{L}^+$ is positive semidefinite, since $\mathbf{L}$ psd $\iff$ $\mathbf{L}^+$ psd, and $\mathbf{L}$ is psd since it is diagonally dominant.

Here (1) + (2) above implies that $\mathbf{L}^+$ defines an inner product between $\mathbf{x}$ and $\mathbf{y}$ as $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^t \mathbf{L}^+ \mathbf{y}$ in $\mathbb{R}^n$. This induces a norm: $||\mathbf{x}|| = (\mathbf{x}^t \mathbf{L}^+ \mathbf{x})^{1/2}$. Therefore, the quantity $[n(i,j)]^{1/2}$ is called the **Euclidean Commute Time** (ETC) **Distance**.

One way to compute the pseudoinverse is to do $\mathbf{L}^+ = (\mathbf{L} - \mathbf{1}\mathbf{1}^y/n)^{-1} - \mathbf{1}\mathbf{1}^t/n$ via sparse (low rank) Cholesky factorization [2] (homework). For large graphs, [7] suggests Markov chain Monte Carlo.

3.3.2 Iterative methods for ETC distance

An alternative is to use iterative methods. Observe that the recursive relation for $m(k|i)$ in equation (3.1) can be interpreted as a matrix-matrix multiplication:

$$m_{ik} = \sum_{j=1}^{N} p_{ij} m_{jk} + 1 \iff \begin{cases} \mathbf{M}^{(n+1)} = \mathbf{P}\mathbf{M}^{(n)} + \mathbf{1}\mathbf{1}^t \\ diag(\mathbf{M}^{(n+1)}) = \mathbf{0} \end{cases} \tag{3.2}$$

There are many virtues to this recurrence relations:

- If the degree for each node is small, then $\mathbf{P}$ is a sparse matrix, so the multiplication $\mathbf{P}\mathbf{M}^{(n)}$ is linear in the number of non-zero entries
- It maintains correct diagonal entries
- It is monotonic in the sense that $\mathbf{M}^{(n+1)} \geq \mathbf{M}^{(n)}$ elementwise
- It converges to the minimal solution of the equations.

We prove monotonicity and convergence below.

Theorem 3.2 Properties of the iterative formula

Let T_{ik} be a random variable that counts the number of steps taken to reach k from i, that is, $E(T_{ik}) = m_{ik}$. Initialize $\mathbf{M}$ to be a matrix of 0's. Then the iteration in equation (3.2) satisfies:

(1) $0 \leq m_{ik}^{(1)} \leq \ldots \leq m_{ik}^{(n)} \leq m_{ik}^{\text{true}}$

(2) $\lim_{n\to\infty} m_{ik}^{(n)} = m_{ik}^{\text{true}}$

(3) The iterate $m_{ij}^{(n)}$ equals $E(T_{ij} 1_{\{T_{ij} \leq n\}})$ and converges to $E(T_{ij})$ whenever the latter is finite.

Proof. The first claim uses induction. For the base case, we initialize $m_{ij}^{(0)} = 0$. Thus

$$m_{ik}^{(1)} = 1 + \sum_{j \neq i}^{N} p_{ij} m_{jk}^{(0)} = \begin{cases} 1 & \text{off diagonal (i.e. } k = i) \\ 0 & \text{on diagonal (i.e. } k \neq i) \end{cases}$$

$$m_{ik}^{(2)} = 1 + \sum_{j \neq i}^{N} p_{ij} m_{jk}^{(1)} \geq 1 \geq m_{ik}^{(1)}.$$

Now assume $m_{ik}^{(n-1)} \geq m_{ik}^{(n-2)}$, we want to show $m_{ik}^{(n)} \geq m_{ik}^{(n-1)}$. But this is clear:

$$m_{ik}^{(n)} = 1 + \sum_{j \neq i}^{N} p_{ij} m_{jk}^{(n-1)} \geq 1 + \sum_{j \neq i}^{N} p_{ij} m_{jk}^{(n-2)} = m_{ik}^{(n-1)}.$$

To show the sequence is upper bounded, suppose $\mathbf{M} = (m_{ij})$ solves the system of equations such that $m_{ik} \geq$

$m_{ik}^{(n-1)}$. Then again by induction, we have

$$m_{ik} = 1 + \sum_{j \neq i}^{N} p_{ij} m_{jk} \geq 1 + \sum_{j \neq i}^{N} p_{ij} m_{jk}^{(n-1)} = m_{ik}^{(n)}.$$

Part (2). The second claim follows from the monotone convergence theorem, since $m_{ik}^{(n)}$ is monotonically increasing in n and upper bounded by m_{ik}.

Part (3). For the last point, note that the case for $n = 0$ is true since $m_{ii}^{(0)} = T_{ii} = 0$. If we start from $X_0 = i$ and transition to $X_1 = j$ in the first step, we have

$$\begin{aligned}
m_{ik}^{(n+1)} &= E\left(T_{ik} 1_{\{T_{ik} \leq n+1\}}\right) \\
&= \sum_j p_{ij} E\left(T_{ik} 1_{\{T_{ik} \leq n+1\}} \mid X_1 = j\right) \quad \text{(law of total expectation)} \\
&= \sum_j p_{ij} E\left((T'_{jk} + 1) 1_{T'_{jk}+1 \leq n+1}\right) \\
&= 1 + \sum_j p_{ij} E\left(T'_{jk} 1_{T'_{jk} \leq n}\right) \\
&= 1 + \sum_j p_{ij} m_{jk}^{(n)}.
\end{aligned}$$

The monotone convergence theorem again implies that $m_{ij}^{(n)}$ converges to $E(T_{ij})$ given the later is finite. □

3.4 Revised k-means clustering using ETC Distance

(1) Connect each node to its h nearest neighbors, then add the edges of a minimum spanning tree. Then add weights w_{ij} using your favorite method (e.g. inverse Euclidean distance).

(2) Choose number of clusters k and its respective cluster centers $\mathbf{p}_1, ..., \mathbf{p}_k$ where each $\mathbf{p}_i$ is a node.

(3) Assign each node $\mathbf{x}_i$ to the nearest cluster C_l by finding which of $\mathbf{p}_1, ..., \mathbf{p}_k$ is closest to $\mathbf{x}_i$, where closeness is measured as a function of the ETC distance: $\text{dist}(\mathbf{x}_i, \mathbf{p}_j) = n(i,j)^2$

(4) Recompute new cluster centers $\mathbf{p}_1, ..., \mathbf{p}_k$ by minimizing the within-cluster distance:

$$\mathbf{p}_l = \text{argmin}_{\mathbf{x}_j} \left\{ \sum_{\mathbf{x}_k \in C_l} n(k,j)^2 \right\}$$

(5) Repeat (3) and (4) until convergence.

3.5 Some simulations

It is well known that K-means algorithm assumes Gaussian clusters with the same variance. Below illustrates 2 scenarios, one with 2 Gaussian clusters, another with non-convex clusters. Here the graph is formed by connecting each node to 3 of its nearest neighbors. Also, each node i is connected to node $i+1$ except for the

transition node between the 2 clusters. Edge weights are chosen to be equal to the inverse Euclidean distance $w_{ij} = \frac{1}{\text{euclidean}(\mathbf{x}_i, \mathbf{x}_j)}$.

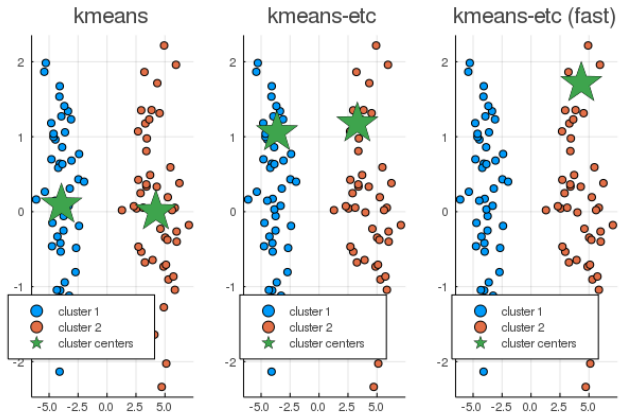


Figure 1: Two unit variance Gaussian clusters with mean -4 and 4.

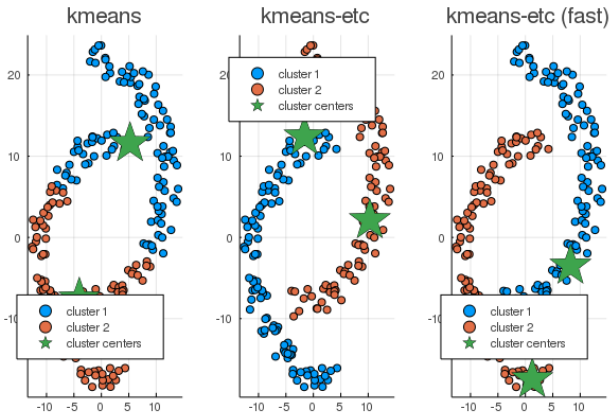
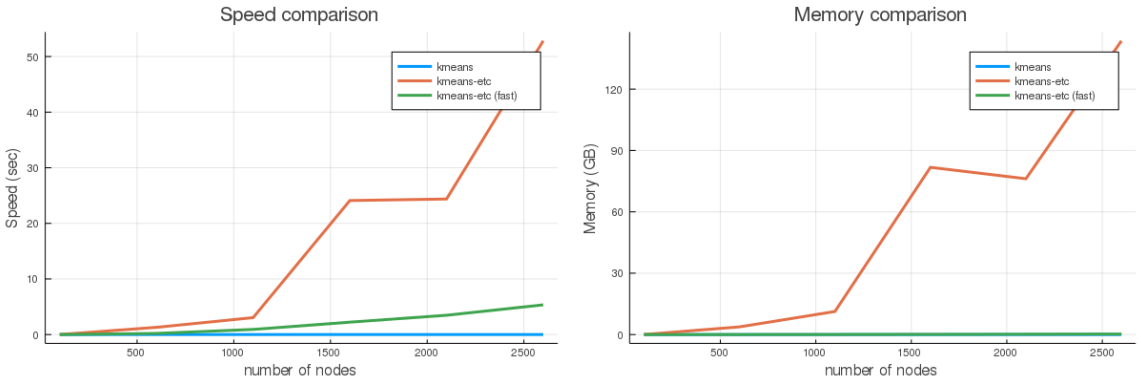


Figure 2: Two non-convex clusters.

Below compares scalability of the k-means, k-means using closed form pseudoinverse of ETC distance (kmeans-etc), and k-means using the iterative algorithm to estimate ETC distance (kmeans-etc (fast)). One should not take it too seriously because I did not try very hard to optimize this code.



4 Problems

In honor of Ken Lange, you are required to do 2 problems. If you do more I will grade your top 2 problems. Every problem is worth the same number of points. If a problem has subproblems, each subproblem is worth the same number of points.

Problem 4.1 Bounds of binomial coefficients

For integers n and k, prove the following inequalities

$$\frac{n^k}{k^k} \leq \binom{n}{k} \leq \frac{n^k}{k!} < \left(\frac{ne}{k}\right)^k$$

which is used in part 2 of our sharp threshold proof. For the strict inequality, rewrite $\frac{n^k}{k!} = \left(\frac{n}{k}\right)^k \frac{k^k}{k!}$ and use Taylor expansion on e^k.

Problem 4.2 Verify this unproved claim

Prove that $\mathbf{L}^+$ in theorem 3.1 can be computed via

$$\mathbf{L}^+ = \left(\mathbf{L} - \frac{\mathbf{1}\mathbf{1}^t}{n}\right)^{-1} + \frac{\mathbf{1}\mathbf{1}^t}{n}$$

where $\mathbf{1}$ is a vector of 1s. This is equation (3) in [2], but it came without a proof.

Problem 4.3 Mandatory computational problem

Starting with my code, do one of the computational problems below:

- Compute the pseudoinverse $\mathbf{L}^+$ via Cholesky factorization. Recall Cholesky for $\mathbf{A}\mathbf{x} = \mathbf{b}$ is solved via back substitution: $\mathbf{A}\mathbf{x} = \mathbf{b} \iff \mathbf{L}\mathbf{L}^t\mathbf{x} = \mathbf{b} \iff \mathbf{L}^t\mathbf{x} = \mathbf{y}$ and $\mathbf{L}\mathbf{y} = \mathbf{b}$. Solving for $\mathbf{y}$ is easy because $\mathbf{L}$ is lower triangular.
- Recall that my code adds edges between nodes i and $i+1$ as an adhoc tatic for making the graph connected. For a better alternative, [7] suggests to add edges from a **minimal spanning tree**.

In my code for computing the adjacency matrix **A**, compute the edges from a minimal spanning tree using your favoriate algorithm. Hopefully this solves most of our stability problem.

Problem 4.4 Colorings of graphs

Let K_z be a **complete graph** where all $z \in \mathbb{Z}_+$ nodes forms an edge with every other node. With equal probability, each edge is colored with red or green. Prove that $z = 6$ is the minimal number of nodes needed to guarantee the existance of a **monochromatic triangle** (i.e. triangle with all edges the same color). This type of problem is what Ramsey theory studies, which we almost did.

We have just shown $R(3,3) = 6$. Similarly, $R(3,4)$ is the minimal number of nodes to guarantee a red triangle or complete green rectangle (i.e. green K_4). Function R obviously generalizes to more colors and shapes. Using Erdos' probabilistic method, Ramsey's theorem (see [6] or theorem 3.3 of [5]) says this number is finite but exponential. This takes us to Erdos' famous quote:

> Suppose aliens invade earth and threaten to obliterate us within a year unless human beings can find $R(5,5)$. We could marshal the world's best minds and fastest computers, and within a year we could probably calculate the value. However, if the aliens demanded $R(6,6)$, we would have no choice but to launch a preemptive attack.

If you want to be famous, find $R(5,5)$.

References

[1] Acemoglu, D. and Ozdaglar, A. (2009). Lecture 3: Erdos-Renyi graphs and Branching Processes. http://economics.mit.edu/files/4621.

[2] Fouss, F., Pirotte, A., Renders, J.-M., and Saerens, M. (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on knowledge and data engineering*, 19(3):355–369.

[3] Lange, K. (2010). *Applied probability*. Springer Science & Business Media.

[4] Ramchandran, K. (2009). Random Graphs. https://inst.eecs.berkeley.edu/~ee126/sp18/random-graphs.pdf.

[5] Van Lint, J. H., Wilson, R. M., and Wilson, R. M. (2001). *A course in combinatorics*. Cambridge university press.

[6] Vasey, S. (2018). The Probabilistic Method and Ramsey Theory. http://people.math.harvard.edu/~sebv/probability-spring-2018/probabilistic-notes.pdf.

[7] Yen, L., Vanvyve, D., Wouters, F., Fouss, F., Verleysen, M., and Saerens, M. (2005). clustering using a random walk based distance measure. In *ESANN*, pages 317–324.