Winter Internship on Modularity Analysis on Complex Graphs

– Maitreya Sameer Ganu (IMS23099)

Indian Institute of Science Education and Research (IISER), Thiruvananthapuram

November 2024 - January 2025

CHAPTERS

1. Introduction.

In this project , I studied the Modularity Analysis on Graphs , which deals with finding and extracting sub-graphs of a given graph which are densely or sparsely connected and separating them into "Clusters" or "Communities" of high and low density respectively. It measures the "Quality (Q)" of the clusters by examining it's connectivity. This method differs slightly from Clustering Algorithms in the sense that clustering algorithms groups similar data points into a specific cluster , having some properties. I shall be referring to the research paper "Modularity and community structure in networks" by M. E. J. Newman.

2. Understanding Key Concepts.

2.1 : What is a graph partitioning?

Let's understand this in a simple way by an example:

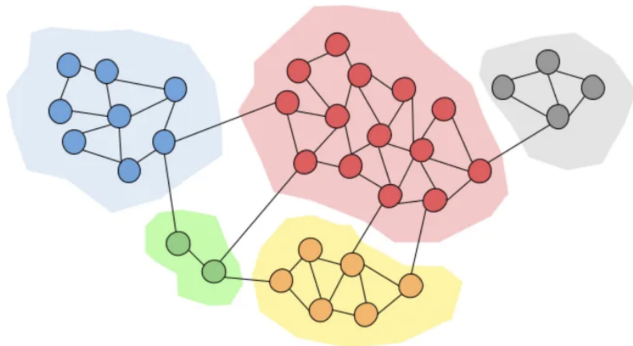

Figure: Communities in a graph.

A graph partition is the splitting of the graph into a mutually exclusive set of nodes or vertices. So , in our example the nodes in each colour form a community set. Note that there is no unique assignment of communities in a graph . It is also possible to make a single community consisting of all vertices/nodes of a graph.

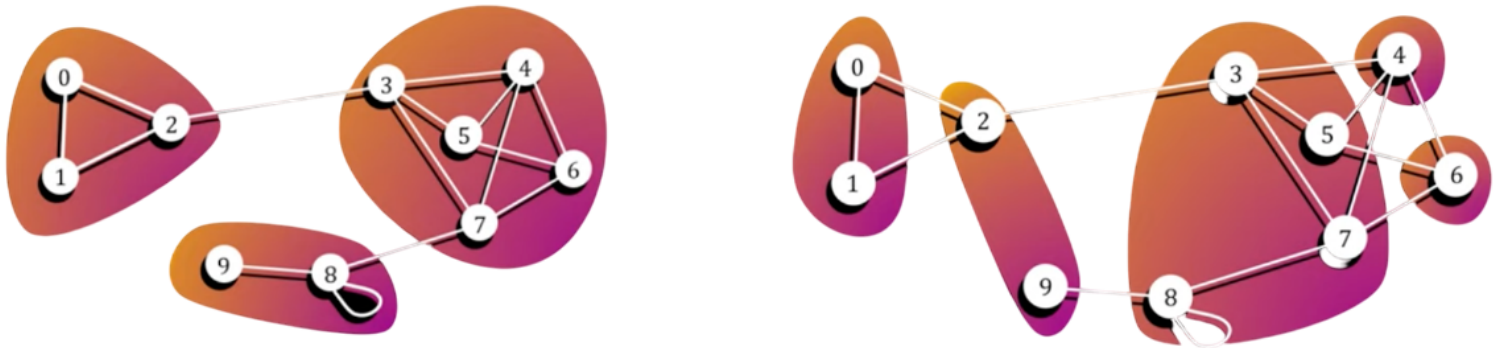

Figure: Different types of possible clusterings in a graph.

Let the figure on the left be labelled as figure-1 and the one on the right be labelled as figure-2. The cluster set of figure-1 is $C_1 =$ ((0,1,2),(3,4,5,6,7),(8,9)) while the cluster set of figure-2 is $C_2 =$ ((0,1),(2,9),(3,5,7,8),(4,6)). Now our task becomes to find total number of possible partitions of a given graph with 'n' nodes. These are given by something called as the "Bell numbers" or B_n

The first bell number B_1 is 1. For $n \geq 2$, we have:

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k$$

In real life problems, we will be dealing with very complex graphs consisting of millions or billions of nodes. In the next section, we shall tackle the modularity of a graph.

2.2 : Modularity and Quality of clustering: Let's consider a weighted graph . Clearly , we notice a lot of edges are contained within communities compared edges that run between the communities. Let 'm' be the total number of edges in a graph.

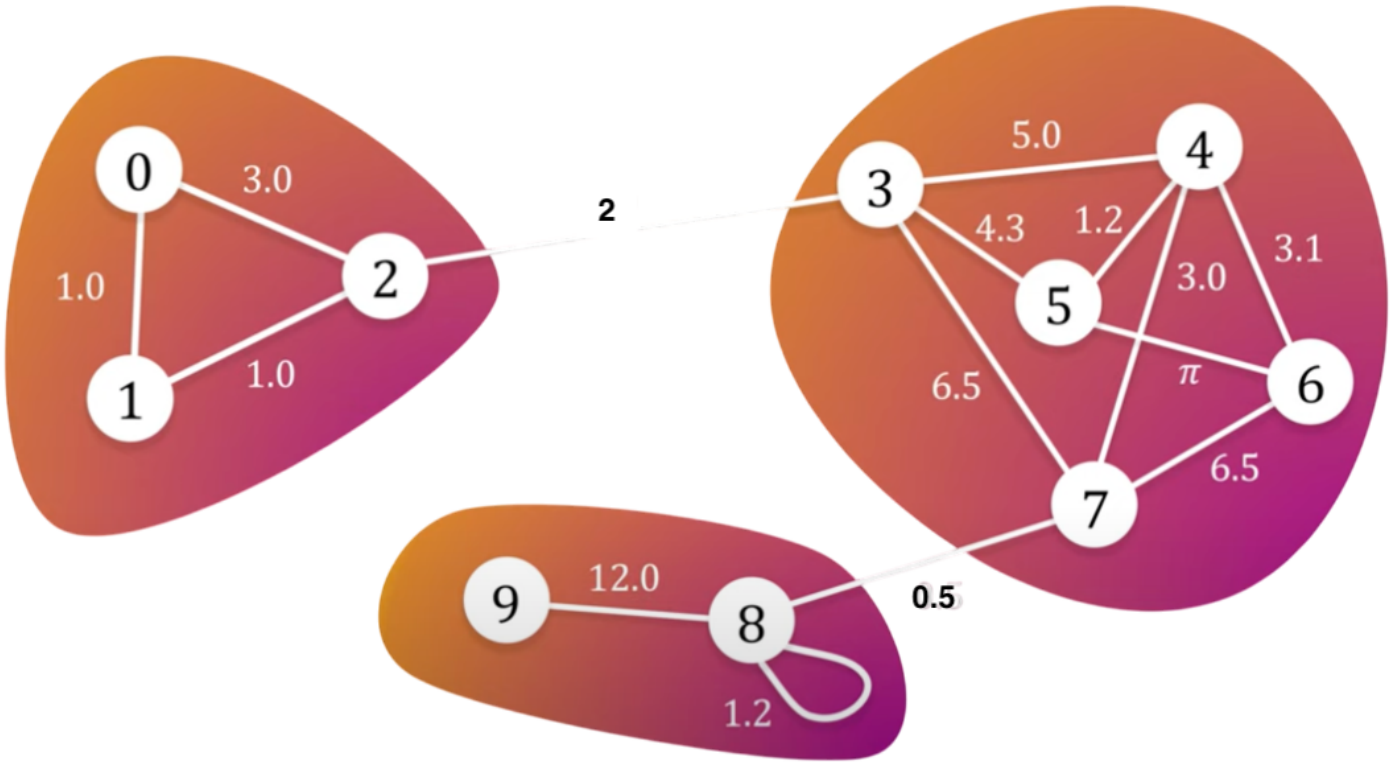

Figure: A weighted connected graph , divided into communities.

The modularity matrix is defined as the weighted adjacency matrix of a graph. Consider the cluster ((0,1,2)). The adjacency matrix is given as :

$$\begin{bmatrix} 0 & 1 & 3 \\ 1 & 0 & 1 \\ 3 & 1 & 0 \end{bmatrix}$$

The adjacency matrix for the entire graph if one community enclosed all it's nodes will be as follows :

$$\begin{pmatrix} 0 & 1.0 & 3.0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.0 & 0 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3.0 & 1.0 & 0 & 2.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2.0 & 0 & 5.0 & 4.3 & 0 & 6.5 & 0 & 0 \\ 0 & 0 & 0 & 5.0 & 0 & 1.2 & 3.1 & 3.0 & 0 & 0 \\ 0 & 0 & 0 & 4.3 & 1.2 & 0 & \pi & 0 & 0 & 0 \\ 0 & 0 & 0 & 3.1 & \pi & 0 & 6.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6.5 & 3.0 & 0 & 6.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 1.2 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12 & 0 \end{pmatrix}$$

The above matrix is a symmetrical matrix because the graph is undirected . If it were a directed graph , we would have a symmetrical matrix . Now , our goal is to find such communities wherein there are maximum number of edges within the communities and minimum number of edges between the communities . Let's try to build a mathematical formula to evaluate "Quality" of the communities formed. Let C be the cluster set , consisting of a specific partition of nodes . Then , weighted number of edges within the communities is given by the equation :

$$\frac{1}{2}\sum_{c\in C}\left(\sum_{u\in C}\sum_{v\in C}A_{uv}\right) \tag{1}$$

Let's try to understand this formula step-by-step with an example . For this , we shall consider the graph in the section 2.2 and we shall select the cluster : (0,1,2)

Here , we have :

$$\sum_{u \in C} \sum_{v \in C} A_{uv} = \begin{array}{l} A_{00} + A_{01} + A_{02} \\ + A_{10} + A_{11} + A_{12} \\ + A_{20} + A_{21} + A_{22} \end{array}$$

This is nothing but the sum of all the elements in the adjacency matrix of the cluster $(0, 1, 2)$. This means we are counting the weighted number of edges twice because, for an undirected graph, the adjacency matrix is symmetric. So, the element $A_{ij} = A_{ji}$. That's why we are multiplying the term

$$\sum_{c \in C} \left(\sum_{u \in C} \sum_{v \in C} A_{uv} \right)$$

by the factor of $\frac{1}{2}$ so that we are correct with our counting of edges.

Now , using similar logic as above to avoid double counting of edges , it is obvious that the total number of edges in the graph are exactly :

$$\frac{1}{2}(\sum_{u}\sum_{v} A_{uv})$$

which is often denoted as 'm'. So , the fraction of edges within the communities is given by :

$$\frac{\frac{1}{2}\sum_{c\in C}\left(\sum_{u\in C}\sum_{v\in C} A_{uv}\right)}{\frac{1}{2}\sum_{c\in C}\left(\sum_{u\in C}\sum_{v\in C} A_{uv}\right)} \tag{2}$$

which can also be written as :

$$\frac{1}{2m}\sum_{c\in C}\left(\sum_{u\in C}\sum_{v\in C} A_{uv}\right) \tag{3}$$

Now , we can convert the above equations into an *"Optimisation"* problem by re-writing them as :

$$\max_{C}(\text{Fraction of edges within communities}) = \max_{C} \frac{1}{2m} \sum_{c \in C} \left(\sum_{u \in c} \sum_{v \in c} A_{uv} \right)$$

Now our task is to try out every possible clustering and see which clustering gives the highest value of equation (3) . We select the partition which gives the highest value. Let's see what partition gives the highest value for the graph in section 2.2 :

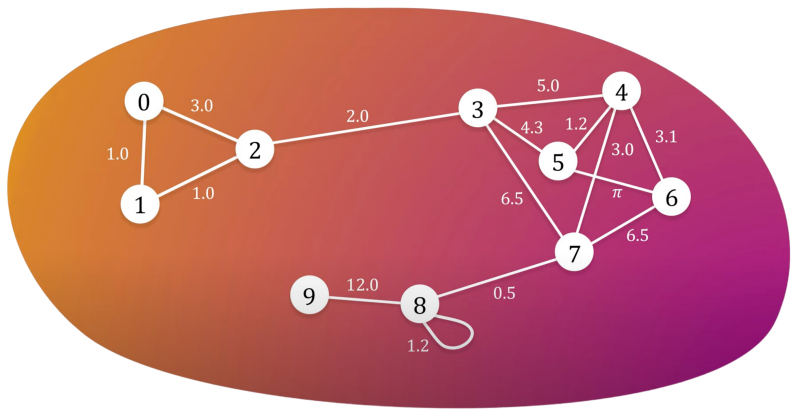

Figure: Partition giving the highest value in equation (3)

The Problem with the Formula

The problem with this is that for any graph, there would be only one community consisting of all the nodes. This happens because if every node is placed in a single community, then the term

$$\sum_{c \in C}$$

becomes useless, and hence our formula reduces to:

$$\frac{1}{2m}\left(\sum_{u \in C}\sum_{v \in C} A_{uv}\right)$$

which is nothing but

$$\frac{2m}{2m} = 1$$

And thus , we shall always get only one community consisting of all nodes which gives highest value in equation (3).

Solution to Solve the Above Problem

"A good division of a network into communities is not merely one in which there are few edges between communities; it is one in which there are **fewer than expected edges** *between communities."*

- *If the number of edges between two groups is only what one would expect on the basis of random chance, then few thoughtful observers would claim this constitutes evidence of meaningful community structure.*
- *On the other hand, if the number of edges between groups is* **significantly less than we expect by chance**, *or equivalently if the number within groups is significantly more, then it is reasonable to conclude that something interesting is going on.*

– M. E. J. Newman

Redefining Terms

Let's define Modularity as difference between Fraction of edges that fall within the communities and Fraction of edges that fall within the communities *if edges were distributed at random*. But some things should be kept in mind whilst considering "if edges were distributed at random" . Properties of graph that must be preserved in the random model of the graph :

- It should have same number of nodes and edges.
- The node degree of each node should be same as that of the actual graph.

Let's perform an algorithm to write everything mathematically : There are 'm' edges in the graph . Cut each edge so that every node has "edge stubs" . So , there are '2m' edge stubs . The probability that any two edge stubs will share an edge is

$$\frac{1}{2m-1}$$

because an edge stub can't connect with itself .

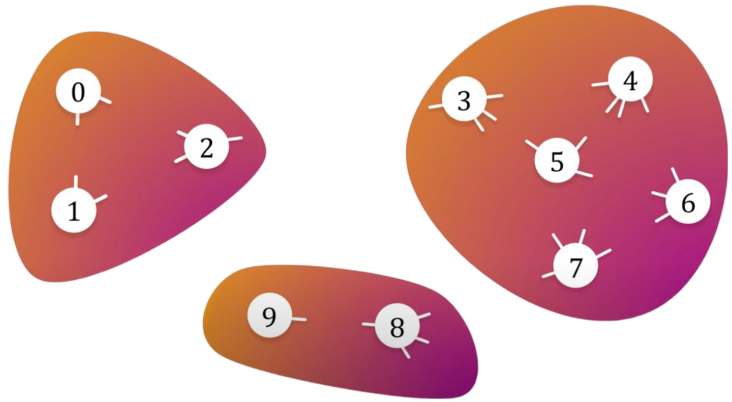

Figure: Edge Stubs

The term

$$\frac{1}{2m}\sum_{c\in C}\left(\sum_{u\in c}\sum_{v\in c}\mathbb{E}\right)$$

Gives the fraction of edges that fall within the communities if edges were distributed at random. We have to find '$\mathbb{E}$', which is the expected number of edges between any two nodes u and v, such that $u \neq v$.

Let the nodes u and v have k_u and k_v edge stubs respectively. So , $\mathbb{E}$ equals the product of all possible pairs of stubs between the nodes u and v and the probability that any two distinct nodes will share an edge. Mathematically ,

$$\mathbb{E} = \frac{k_u k_v}{2m - 1}$$

Hence , Fraction of edges that fall within the communities if edges were distributed at random now becomes :

$$\frac{1}{2m} \sum_{c \in C} \left(\sum_{u \in c} \sum_{v \in c} \left(\frac{k_u k_v}{2m - 1} \right) \right)$$

So , We define modularity as difference between Fraction of edges that fall within the communities and Fraction of edges that fall within the communities if edges were distributed at random which now can be mathematically be represented as :

$$\frac{1}{2m}\sum_{c\in C}\left(\sum_{u\in C}\sum_{v\in C}A_{uv}\right)-\frac{1}{2m}\sum_{c\in C}\left(\sum_{u\in c}\sum_{v\in c}\frac{k_u k_v}{2m-1}\right)$$

which can be simplified as:

$$\frac{1}{2m}\sum_{c\in C}\sum_{u\in c}\sum_{v\in c}\left(A_{uv}-\frac{k_u k_v}{2m-1}\right) \tag{4}$$

Equation (4) is denoted by $\mathbb{Q}(C)$, also called "quality." Whilst dealing with real-life examples, the quantity $2m$ is much, much greater than 1, and so $2m-1\approx 2m$. Thus, we get:

$$\mathbb{Q}(C)=\frac{1}{2m}\sum_{c\in C}\sum_{u\in c}\sum_{v\in c}\left(A_{uv}-\frac{k_u k_v}{2m}\right) \tag{5}$$

Now, we can rewrite Equation (5) as follows:

$$\mathbb{Q}(C) = \frac{1}{2m} \sum_{c \in \mathcal{C}} \left(\Sigma_c - \frac{(\hat{\Sigma}_c)^2}{2m} \right) \tag{6}$$

where:

$$\Sigma_c := \sum_{u \in c} \sum_{v \in c} A_{uv},$$

Which is twice the sum of the weights of edges inside community c.

$$\hat{\Sigma}_c := \sum_{v \in c} K_v,$$

Which is the sum of the weights of edges incident to vertices in c.

I shall explain what we have done so far with a real life example of a graph and finding the quality of each cluster.

3. Spectral Modularity Maximization

Consider a graph with n nodes. We now define the quality of clusters as:

$$\mathbb{Q}(C) = \frac{1}{2m} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j)$$

where $\delta(c_i, c_j)$ is the *Kronecker delta*, which equals 1 if $i = j$ and 0 if $i \neq j$. The quantity

$$A_{ij} - \frac{k_i k_j}{2m} = B_{ij}$$

is called as "Modularity matrix".

$$\mathbb{Q}(C) = \begin{cases} > 0 & : \text{implies assortative mixing,} \\ < 0 & : \text{implies disassortative mixing,} \\ = 0 & : \text{implies perfect randomness.} \end{cases}$$

Assortative Mixing:

Assortative mixing refers to a tendency for nodes in a network to be connected to other nodes that are similar to them, typically in terms of degree or other properties. Specifically, a network exhibits assortative mixing when high-degree nodes are more likely to connect with other high-degree nodes, and low-degree nodes are more likely to connect with other low-degree nodes.

Disassortative Mixing:

Disassortative mixing, on the other hand, refers to a tendency for nodes in a network to be connected to other nodes that are different from them, in terms of degree or other properties. In a disassortative network, high-degree nodes tend to connect with low-degree nodes, and vice versa.

We shall re-define the *Kronecker delta* as follows:

$$\delta(c_i, c_j) = \frac{1}{2}(s_i s_j + 1)$$

Property of B_{ij} :

$$\sum_j B_{ij} = 0$$

So, our formula is now modified as:

$$\mathbb{Q} = \frac{1}{4m} \sum_{i,j} B_{ij}(s_i s_j + 1)$$

$$= \frac{1}{4m} \sum_{i,j} B_{ij} s_i s_j = \frac{1}{4m} \sum_{i=1}^{n} \sum_{j=1}^{n} B_{ij} s_i s_j$$

$$= \frac{1}{4m}(\mathbf{s}^T \mathbf{B} \mathbf{s}) \ldots\ldots (7)$$

Why does $\sum_j B_{ij} = 0$?

The modularity matrix B_{ij} is defined as:

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2m},$$

where:

- A_{ij}: The actual connection between nodes i and j,
- $\frac{k_i k_j}{2m}$: The expected connection between nodes i and j in a random graph.

To show why $\sum_j B_{ij} = 0$, we calculate it step by step:

$$\sum_j B_{ij} = \sum_j A_{ij} - \sum_j \frac{k_i k_j}{2m}.$$

Step 1: First Term

First term: $\sum_j A_{ij}$ is the sum of all connections of node i to other nodes. This is simply the degree of node i, k_i:

$$\sum_j A_{ij} = k_i.$$

Step 2: Second Term

Second term: $\sum_j \frac{k_i k_j}{2m}$. Here:

$$\sum_j \frac{k_i k_j}{2m} = \frac{k_i}{2m} \sum_j k_j.$$

The sum $\sum_j k_j$ equals $2m$, the total degree of the graph. Thus:

$$\sum_j \frac{k_i k_j}{2m} = \frac{k_i \cdot 2m}{2m} = k_i.$$

Step 3: Combine the Two Terms

Now, combine the two terms:

$$\sum_j B_{ij} = \sum_j A_{ij} - \sum_j \frac{k_i k_j}{2m}.$$

Substituting the results:

$$\sum_j B_{ij} = k_i - k_i = 0.$$

Conclusion

The sum $\sum_j B_{ij} = 0$ because the modularity matrix measures the difference between actual and expected connections.

- The first term $\sum_j A_{ij}$ accounts for the actual connections.
- The second term $\sum_j \frac{k_i k_j}{2m}$ balances it with expected connections in a random graph.

These differences are designed to cancel each other out.

Understanding s_i: Community Assignment

What is s_i?

In Newman's modularity framework, s_i represents the community assignment of node i. It works as:

- $s_i = +1$: if node i belongs to community 1.
- $s_i = -1$: if node i belongs to community 2.

Where s_i Appears in Modularity:

$$\mathbb{Q} = \frac{1}{4m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j$$

- A_{ij}: adjacency matrix, 1 if there is an edge between i and j, 0 otherwise.
- k_i, k_j: degrees of nodes i and j.
- m: total number of edges.
- $s_i s_j$: represents the product of community memberships:
  - $s_i s_j = +1$: when i and j are in the same community.
  - $s_i s_j = -1$: when i and j are in different communities.

Why Only Two Communities? Simplification of the Problem

1. **Simplification of the Problem:**
   - Dividing a network into **two communities** is the simplest non-trivial grouping.
   - It is easier to analyze and compute than dividing into three or more communities.
   - The binary nature of community membership ($s_i = +1$ or -1) simplifies the mathematics.

2. **Spectral Methods Work Naturally for Two Communities:**
   - Newman's modularity optimization uses the **eigenvector of the modularity matrix**.
   - The **leading eigenvector** (associated with the largest eigenvalue) provides the best division of the network into two groups.
   - For more than two communities, additional eigenvectors and more complex algorithms are required.

Why Only Two Communities? Iterative Splitting and Efficiency

3. **Iterative Splitting to Handle More Communities:**
   - Newman's method extends to multiple communities through:
     1. Splitting the network into two communities.
     2. Recursively splitting each community further if needed.

4. **Computational Efficiency:**
   - Directly dividing into many communities is computationally expensive.
   - Starting with two communities and refining them step-by-step reduces complexity.

Why Only Two Communities? Real-World Analogy and Summary

5. Real-World Analogy:

- Imagine organizing a party and splitting guests into smaller groups.
- Starting with just two groups (e.g., those who know each other well vs. those who don't) is simpler than figuring out several groups at once.
- Once split, you can refine the groups further.

Summary: Newman starts with two communities because:

- It is mathematically simpler.
- Spectral methods work naturally for binary splits.
- Iterative splitting can extend the method to handle multiple communities efficiently.

Relaxation of the Integer Optimization Problem

$$\max_{s_i \in \{-1,1\}} \frac{1}{4m} s^T B s \approx \max_{s \in \mathbb{R}} \frac{1}{4m} s^T B s$$

This is a relaxation of the original integer optimization problem, where we allow the variables s_i to take on real values instead of just -1 or 1. This relaxation leads to a new constraint:

$$s^T s = n$$

This constraint ensures that the sum of the squares of the elements of s is equal to n.

Lagrange Multiplier Method

To solve the relaxed optimization problem, we use the Lagrange multiplier method for adding constraints:

$$\frac{\partial}{\partial s}\left(s^T B s\right) = 0 \quad \Rightarrow \quad \frac{\partial}{\partial s}\left[s^T B s + \beta\left(n - s^T s\right)\right] = 0$$

Simplifying the derivative:

$$2Bs - 2\beta s = 0 \quad \Rightarrow \quad Bs = \beta s$$

This implies that s is an eigenvector of B, with β as the eigenvalue.

Maximization of the Objective Function

To maximize the relaxed objective function:

$$\max \frac{1}{4m} s^T B s = \frac{1}{4m} s^T (\beta s) = \frac{\beta}{4m} s^T s$$

Since $s^T s = n$, the maximization depends on picking the largest β, which corresponds to the largest eigenvalue of B. Therefore, we choose:

$$s = v_1 \quad \text{(eigenvector corresponding to the largest eigenvalue)}$$

Conversion to Integer Solution

To convert back to the original problem where $s_i \in \{-1, 1\}$, we use the sign of the entries of v_1:

$$s_i = \begin{cases} +1 & [v_1]_i > 0 \\ -1 & [v_1]_i < 0 \end{cases}$$

Or equivalently:

$$s = \text{sign}(v_1)$$

Summary

- The original integer optimization problem was relaxed to a continuous optimization problem.
- The solution to the relaxed problem involves finding the eigenvector corresponding to the largest eigenvalue of B.
- The solution was mapped back to the original integer domain using the sign function.

This method is widely used in spectral graph theory, including modularity maximization for community detection.

Dividing Networks into More than Two Communities :
In the previous section, we described a matrix-based method for dividing a network into two parts. However, many networks contain more than two communities. To handle this, the method is extended by using repeated division:

- First, divide the network into two parts using the algorithm.
- Then, further divide these parts iteratively.

Key Note: It is incorrect to delete edges between two parts after the initial division and then apply the algorithm to subgraphs. This changes the degrees in the modularity definition, leading to incorrect results.
Instead, we calculate the additional contribution ΔQ to the modularity for dividing a group g of size n_g into two parts.

Modularity Contribution for a Group g:
The modularity contribution for a group g is given by:

$$Q = \frac{1}{2m} \left[\frac{1}{2} \sum_{i,j \in g} B_{ij} \left(s_i s_j + 1 \right) - \sum_{i,j \in g} B_{ij} \right],$$

where:

- B_{ij} is the modularity matrix element,
- s_i, s_j are the spin variables indicating whether nodes i and j are in the same group ($s_i, s_j \in \{+1, -1\}$),
- m is the total number of edges in the network.

Simplification of the Expression:
Expanding $(s_i s_j + 1)$ gives:

$$Q = \frac{1}{2m} \left[\frac{1}{2} \sum_{i,j \in g} (B_{ij} s_i s_j + B_{ij}) - \sum_{i,j \in g} B_{ij} \right].$$

Simplifying further:

$$Q = \frac{1}{4m} \left[\sum_{i,j \in g} B_{ij} s_i s_j - \sum_{i,j \in g} B_{ij} \right].$$

Generalized Modularity Matrix for Group g:
The summation $\sum_{i,j \in g} B_{ij}$ can be split into diagonal and off-diagonal terms:

$$\sum_{i,j \in g} B_{ij} = \sum_{i \in g} \sum_{k \in g} B_{ik}.$$

Thus, the expression for modularity becomes:

$$Q = \frac{1}{4m} \sum_{i,j \in g} \left[B_{ij} - \delta_{ij} \sum_{k \in g} B_{ik} \right] s_i s_j,$$

where δ_{ij} is the Kronecker delta.

Generalized Modularity Matrix Definition:
We define the generalized modularity matrix $\mathbf{B}^{(g)}$ as:

$$B_{ij}^{(g)} = B_{ij} - \delta_{ij} \sum_{k \in g} B_{ik}.$$

The modularity contribution is now written as:

$$Q = \frac{1}{4m} s^T \mathbf{B}^{(g)} s,$$

where s is the vector of community assignment for the nodes in group g.

Properties of the Generalized Modularity Matrix:
The generalized modularity matrix $\mathbf{B}^{(g)}$ has the following properties:

- The rows and columns of $\mathbf{B}^{(g)}$ sum to zero:

$$\sum_j B_{ij}^{(g)} = 0.$$

- When g is undivided, the contribution $\Delta Q = 0$.
- For a complete network, $\mathbf{B}^{(g)}$ reduces to the original modularity matrix $\mathbf{B}$.

Generalized Modularity Matrix:
The contribution ΔQ to modularity is given by:

$$\Delta Q = \frac{1}{4m} \sum_{i,j \in g} B_{ij}^{(g)} s_i s_j,$$

where:

- δ_{ij} is the Kronecker delta,
- $s_i \in \{+1, -1\}$,
- $B^{(g)}$ is the generalized modularity matrix of size $n_g \times n_g$,
- The elements of $B^{(g)}$ are indexed by the labels i, j of vertices in group g, and have values:

$$B_{ij}^{(g)} = A_{ij} - \frac{k_i k_j}{2m}.$$

When to Halt the Subdivision Process ?:

The algorithm halts the subdivision process under the following conditions:

- If there is no division of a subgraph that increases modularity ($\Delta Q \leq 0$).
- This occurs when there are no positive eigenvalues of $\mathbf{B}^{(g)}$.

The leading eigenvalue of $\mathbf{B}^{(g)}$ provides a simple check:

- If the leading eigenvalue is zero, the subgraph is indivisible.
- However, small positive eigenvalues may still result in negative ΔQ when combined with large negative eigenvalues.

Algorithm Summary:
The steps of the algorithm are as follows:

1. Construct the modularity matrix $\mathbf{B}$ for the network.
2. Find the leading eigenvalue and eigenvector of $\mathbf{B}$.
3. Divide the network into two parts based on the signs of the eigenvector elements.
4. Repeat the process for each part using the generalized modularity matrix $\mathbf{B}^{(g)}$.
5. Stop when no division increases modularity ($\Delta Q \leq 0$).

Result:
The network is decomposed into indivisible subgraphs. Each "community" is defined as an indivisible subgraph.

4. Code for Modularity Analysis

To download the project PDF, click here

5. Summary of Newman's Paper.

- Newman revolutionized visualization of networks in complex connected graphs . His primary approach was to divide a graph into two communities and then applying his algorithm to each community iteratively . This was , however not good because it lead to a lot of incorrect results which contradicted with the basic definitions of modularity. Hence he used the sub-division process as discussed in chapter 3 . It depends heavily on the eigenvalue decomposition on the modularity matrix .
- His algorithm is used in a wide variety of applications from social networks , AI , protein interactions , technological networks , etc.
- A higher $\mathbb{Q}$ indicates a stronger community structure.

6. Key Insights.

- Overall , this was a fun project to do. It made me use softwares which I had never used before . It helped me understand key concepts of network analysis and visualization.
- This project has upgraded my skill set in many ways and has inspired me to learn more about graph theory and clustering algorithms.
- I really want to thank and express my gratefulness and gratitude to *Prof . Saptarshi Bej* , School of Data Science , IISER Thiruvananthapuram for giving me internship under his kind guidance and mentorship. He was really helpful and an excellent teacher throughout my project.

7. References.

- Newman's paper on modularity.
  -  Newman, M. E. J., **Modularity and Community Structure in Networks**. Available at: https://www.pnas.org/content/103/23/8577 (Accessed: December 15, 2024).
-  Channel Name: "*Splience*" Available at: https://www.youtube.com/watch?v=Xt0vBtBY2BU (Accessed: December 15, 2024).
-  Channel Name: "*Justin Ruths*" Available at: https://www.youtube.com/watch?v=IRX5CvK3JpY (Accessed: December 15, 2024).

THANK YOU!!