Research Assignment
Modularity on the Karate Club Graph

DSC212: Graph Theory Module

What you will deliver

Submission mode (strict): An **IPython/Jupyter notebook (`.ipynb`)** that *runs top-to-bottom without manual edits*, containing your code, outputs, and required visualizations. You must upload the notebook and assets to a **GitHub repository you own** and submit the **repository URL**. **No other modes of submission will be accepted.**

Abstract

This handout introduces modularity as a quality function for community detection and guides you to implement a spectral modularity method *from scratch* on the classic Karate Club network. You will learn the story behind the data set, the motivation for modularity, the mathematics for the two-community case, and you will follow commented pseudocode to create your own Python implementation (using only basic scientific Python tools and NetworkX library).

1 The Karate Club Story

In the early 1970s, a graduate student in anthropology, Wayne W. Zachary, became interested in how friendships and alliances form within small social groups. To study this in detail, he spent two years observing the social interactions among 34 members of a university karate club in the United States. He carefully recorded who spent time with whom outside of formal classes: attending social events together, training together, and interacting as friends. These observations produced a network where each member of the club is a *node*, and an edge is drawn between two nodes if the members frequently interacted. This network, now called the *Karate Club graph*, is one of the most famous real-world social networks in network science.

A twist in the story makes this dataset especially interesting. During Zachary's fieldwork, the club's instructor and the club's administrator had a serious disagreement about whether club revenues should be spent on equipment or used for instructor salaries. This dispute escalated, eventually splitting the club into two rival factions. Members had to decide whom to follow: the instructor ("Mr. Hi") or the administrator (the club president). Remarkably, the friendships Zachary had recorded *before the split* were strong predictors of how the split actually happened: friends tended to stick together.

From a learning perspective, this network is a perfect case study. We know the "ground truth" outcome (two groups), but the network data were collected before the split occurred. This allows us to ask: *Can a community-detection method, applied only to the graph structure, recover the division that actually happened?* In other words, can we use mathematics to uncover the social fault lines that were about to fracture the club?

For these reasons, the Karate Club graph is the "hello world" of community detection: it is small (only 34 nodes, 78 edges), simple to visualize, and historically meaningful. It has been used in hundreds of scientific papers as a benchmark. In this assignment, you will use it as a testing ground to learn the idea of *modularity* and to implement a method that tries to rediscover the split between Mr. Hi's group and the club president's group.

2 Why Modularity?

At first glance, finding communities in a network might seem easy: simply cut as few edges as possible between two groups of nodes. However, this approach can lead to trivial and unhelpful solutions. For example, imagine we split off a single vertex from the rest of the network. That

cut only removes the edges attached to that single node, which could be a small number. Yet clearly this is not a meaningful "community"—it is just isolating one unlucky vertex.

This motivates the need for a better quality function, one that asks not just *how many edges cross the cut*, but *is the division surprising compared to chance?* This is exactly what **modularity** does.

The idea is to compare the observed number of within-community edges against a random baseline. The baseline comes from a *null model* that preserves the degree of each vertex (so highly connected nodes are still expected to connect to many others), but otherwise connects *stubs* or half-edges at random. If our observed division has *more within-group edges* (or equivalently, *fewer across-group edges*) than this random baseline, then the division is interesting. If not, then the split may just be noise.

Thus, modularity turns community detection into a statistically meaningful optimization problem: find the division of the network that maximizes the difference between "what we see" and "what we would expect if edges were random." This makes modularity one of the most widely used and conceptually intuitive measures of community structure in networks.

3 Mathematics of Modularity

Object	Symbol	Dimension
Global modularity matrix	$\mathbf{B}$	$n \times n$
Restricted modularity matrix	$\mathbf{B}^{(C)}$	$\lvert C\rvert \times \lvert C\rvert$
Eigenvalues	$\lambda_i^{(C)}$	Scalar
Global label vector	$\mathbf{t}$	$n \times 1$
Community label vector	$\mathbf{s}$	$\lvert C\rvert \times 1$
Generic vector	$\mathbf{x}$	$\lvert C\rvert \times 1$
All-ones vector	$\mathbf{1}$	$n \times 1$ (or $\lvert C\rvert \times 1$ when restricted)
Node degree vector	$\mathbf{k}$	$n \times 1$
Quadratic form	$\mathbf{s}^\top \mathbf{B}^{(C)} \mathbf{s}$	Scalar
Modularity gain	ΔQ	Scalar

Table 1: Notation and dimensions for modularity and eigenvalue stopping criterion. If confused about notations, check this table.

Let us now write the mathematics more carefully. Suppose we have an undirected graph $G = (V, E)$ with $n = |V|$ vertices and $m = |E|$ edges. The graph can be represented by its adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$, where

$$A_{ij} = \begin{cases} 1, & \text{if vertices } i \text{ and } j \text{ share an edge,} \\ 0, & \text{otherwise.} \end{cases}$$

The degree of vertex i is $k_i = \sum_j A_{ij}$, and the total degree across the graph is $\sum_i k_i = 2m$.

We now want to describe a division of the network into two groups. One convenient way is to assign each vertex i a label s_i:

$$s_i = \begin{cases} +1, & \text{if vertex } i \text{ is placed in group 1,} \\ -1, & \text{if vertex } i \text{ is placed in group 2.} \end{cases}$$

This produces a vector $\mathbf{s} \in \{-1, +1\}^n$ encoding the bipartition.

3.1 Understanding the modularity matrix

The key mathematical object is the *modularity matrix*, defined as

$$\mathbf{B} = \mathbf{A} - \frac{\mathbf{k}\,\mathbf{k}^\top}{2m},$$

where $\mathbf{k} = (k_1, \ldots, k_n)^\top$ is the degree vector, that is k_i is the degree of vertex v_i. The first term, $\mathbf{A}$, records the actual edges of the network. The second term, $\frac{\mathbf{k}\,\mathbf{k}^\top}{2m}$ (which is a matrix of the same dimension as $\mathbf{B}$), encodes the expected number of edges between vertices i and j under the random *null model*. This may look abstract, so let us unpack it carefully.

Step 1: What does the null model mean? The null model preserves each node's degree but randomizes the actual neighbors. Think of it as follows:

- Each node i has k_i "stubs" or "half-edges" coming out of it.
- There are $2m$ stubs in total, since each of the m edges contributes two ends.
- In the null model, these stubs are randomly paired to form edges.

Thus, the probability that one stub from i connects to one stub from j is roughly proportional to $k_i k_j$.

Step 2: Expected number of edges between i and j Formally, the expected number of edges between nodes i and j is

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

This is intuitive:

- Node i has k_i stubs to offer.
- Each of those stubs connects to one of the $2m$ available stubs uniformly at random.
- The probability of hitting a stub of node j is $\frac{k_j}{2m}$.
- Multiplying gives $k_i \cdot \frac{k_j}{2m}$.

So the expectation is exactly $\frac{k_i k_j}{2m}$.

Step 3: Putting it into the matrix If we arrange all these expectations into an $n \times n$ matrix, the (i, j) entry becomes

$$\left(\frac{\mathbf{k}\,\mathbf{k}^\top}{2m}\right)_{ij} = \frac{k_i k_j}{2m}.$$

That is what appears in the modularity matrix $\mathbf{B}$:

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

Since $\mathbf{B}$ is constructed symmetrically, one can show that it is always a real symmetric matrix; this fact is proved formally in Proposition 1 (see Section 5). Right after that, Theorem 1 tells us the the eigenvalues of the matrix $\mathbf{B}$ are real and one can choose an orthonormal eigenvectors of such a matrix. Interested students can go through the proofs.

The modularity score The modularity Q of a particular division $\mathbf{s}$ is then given by

$$Q = \frac{1}{4m}\,\mathbf{s}^\top \mathbf{B}\,\mathbf{s}. \tag{1}$$

Note that, actually Q is a function of $\mathbf{s}$. Here, $\mathbf{s}^\top \mathbf{B}\mathbf{s}$ measures how well the vertex partition induced by $\mathbf{s}$ (recall that the elements of $\mathbf{s}$ are $+1$ or -1 and induces a partition over the vertex set) aligns with positive values in $\mathbf{B}$ (i.e., pairs that are more connected than chance). A large positive value of Q indicates a meaningful split with significantly more within-community edges than expected at random.

The optimization perspective. Notice that the entire problem of community detection (in this simplified two-community case) boils down to a single task:

Choose the components of $\mathbf{s} \in \{-1, +1\}^n$ *so that* $Q = \frac{1}{4m}\,\mathbf{s}^\top \mathbf{B}\,\mathbf{s}$ *is maximized.*

3.2 Spectral bipartition algorithm

The problem now boils down to finding the vector $\mathbf{s}$ that maximizes Q.

Step 1: Recall the modularity score. We want to maximize

$$Q = \frac{1}{4m}\,\mathbf{s}^\top \mathbf{B}\mathbf{s},$$

where $\mathbf{s} \in \{-1, +1\}^n$ is the community assignment vector.

Recall that here:

- $\mathbf{s}$ is an $n \times 1$ vector (one entry per node)
- $\mathbf{B}$ is an $n \times n$ modularity matrix
- $\mathbf{s}^\top$ is $1 \times n$,

so $\mathbf{s}^\top \mathbf{B}\mathbf{s}$ is a scalar.

Step 2: Why this is hard. Maximizing over all $\{-1, +1\}^n$ vectors means checking 2^n possible assignments — impossible even for moderate n. So we use a standard trick in optimization: *relax* the problem.

Step 3: The relaxation. Instead of forcing $s_i \in \{-1, +1\}$, suppose we allow s_i to take any real value, but constrain $\|\mathbf{s}\|_2 = 1$ (unit length, the subscript 2 below the norm means that the norm is the usual norm obtained by squaring the components of a vector adding them and then taking the square root of the sum.). Then the problem becomes:

$$\max_{\|\mathbf{s}\|_2=1} \mathbf{s}^\top \mathbf{B}\mathbf{s}.$$

A classic result from linear algebra says: **the maximizer of the above maximization problem is is the eigenvector of B corresponding to the largest eigenvalue, called the *leading eigenvector***. This expression $\max_{\|\mathbf{s}\|_2=1} \mathbf{s}^\top \mathbf{B}\mathbf{s}$ is an instance of a quadratic form, and its maximization can be analyzed using the Rayleigh–Ritz theorem (Theorem 2), whose proof appears in Section 5. Interested students can go through the proof.

Step 4: Interpreting $\mathbf{u}_1$. In the relaxation, $\mathbf{u}_1$ gives us a "soft assignment" of nodes: nodes with positive entries are pushed toward one side of the split, and nodes with negative entries are pushed toward the other. Intuitively:

- If $(\mathbf{u}_1)_i$ is strongly positive, node i "wants" to be in community 1.
- If $(\mathbf{u}_1)_i$ is strongly negative, node i "wants" to be in community 2.
- If $(\mathbf{u}_1)_i$ is close to zero, node i is not strongly tied to either side.

Step 5: Thresholding back to $\{-1, +1\}$. Since we need a hard partition, we simply assign

$$s_i = \begin{cases} +1 & \text{if } (\mathbf{u}_1)_i > 0, \\ -1 & \text{otherwise.} \end{cases}$$

This gives us a discrete $\mathbf{s} \in \{-1, +1\}^n$, i.e., a proper bipartition of the graph.

3.3 From Two Communities to Many using Recursive Bisection

So far, we have described how to split a network into two communities using the leading eigenvector of the modularity matrix $\mathbf{B}$. But real-world networks often contain more than just two communities. To extend the method, we use **recursive bisection**.

Step 1: Restricting to a subgraph

Suppose we have already found a community $C \subseteq V$. We want to test if C itself can be split further. The modularity matrix restricted to this group is simply

$$B_{ij}^{(C)} = A_{ij} - \frac{k_i k_j}{2m}, \quad \text{for } i, j \in C.$$

In other words, we take the block of the original modularity matrix $\mathbf{B}$ corresponding to nodes in C.

Step 2: Eigenvalue test

Compute the leading eigenpair $(\lambda_1^{(C)}, \mathbf{u}_1^{(C)})$ of $\mathbf{B}^{(C)}$.

- If $\lambda_1^{(C)} > 0$, then splitting C by the signs of the entries of $\mathbf{u}_1^{(C)}$ increases modularity.
- If $\lambda_1^{(C)} \leq 0$, then no further split improves modularity and C should remain as a single community.

Step 3: Recursive splitting

We repeat this procedure:

1. Start with the full vertex set V.
2. If V can be split (largest eigenvalue > 0), divide it into two groups.
3. Apply the same test recursively to each resulting group.
4. Stop when all groups have nonpositive leading eigenvalues.

At the end, we obtain a partition of V into multiple communities, each of which is "stable" in the sense that no further modularity-improving split exists.

The rule "stop when the leading eigenvalue is nonpositive" is justified formally in Theorem 3 (see Section 5). Intuitively, if the largest eigenvalue of $\mathbf{B}$ is zero or negative, then the quadratic form $\mathbf{s}^\top \mathbf{B}\mathbf{s}$ can never be positive for any choice of assignment vector $\mathbf{s}$. In other words, no partition will increase modularity beyond the trivial value. Therefore, once the leading eigenvalue is nonpositive, further splitting the community cannot improve modularity and should not be attempted.

Key intuition

- A **positive eigenvalue** means there exists a split that improves modularity.
- A **nonpositive eigenvalue** means all possible quadratic forms $\mathbf{s}^\top \mathbf{B}^{(C)} \mathbf{s}$ are ≤ 0, so no improvement is possible.
- Modularity is **additive across communities**, so splitting one block without harming others always increases the total score.
- This process naturally stops, leaving us with multiple communities.

Attribution

The modularity objective, modularity matrix $\mathbf{B}$, and spectral bipartition method are based on standard results; see Newman (2006), "Modularity and community structure in networks," *PNAS* 103(23):8577–8582. We cite it in the References and you must cite it in your report when you describe your method.

4 Multi-Community Detection and Network Metrics

Recursive bisection (multi-community detection)

We extend the two-way spectral split into a multi-community detection algorithm by applying the procedure recursively:

1. Start with the full node set V.
2. Compute the modularity matrix $\mathbf{B}$ for the induced subgraph.
3. Find the leading eigenpair $(\lambda_1, \mathbf{u}_1)$.
4. If $\lambda_1 \leq 0$: stop; the group is indivisible.
5. Else: split nodes into C^+ and C^- by the sign of $\mathbf{u}_1$.
6. Recurse on C^+ and C^- separately.

At the end, we obtain a set of communities $\{C_1, C_2, \ldots, C_r\}$.

Visualization task

At each iteration:

- Draw the graph using a fixed spring layout.
- Assign a unique color to each community.
- Label nodes by ID so that community changes are visible.

Node metrics to compute

After each split, we compute standard centrality and cohesion measures for every node using NetworkX built-in functions:

1. **Degree centrality:**

$$C_D(i) = \frac{k_i}{n-1}, \quad \texttt{networkx.degree\_centrality(G).}$$

   Measures the fraction of nodes that are directly connected to i. Nodes with high degree centrality are immediate hubs with many direct links.

2. **Betweenness centrality:**

$$C_B(i) = \sum_{s \neq i \neq t} \frac{\sigma_{st}(i)}{\sigma_{st}}, \quad \texttt{networkx.betweenness\_centrality(G).}$$

   Counts how often node i lies on shortest paths between other nodes. Nodes with high betweenness act as bridges or brokers, controlling information flow between different parts of the network.

3. **Closeness centrality:**

$$C_C(i) = \frac{n-1}{\sum_{j \neq i} d(i,j)}, \quad \texttt{networkx.closeness\_centrality(G).}$$

   Inverse of the average shortest-path distance from i to all other nodes. A high closeness score indicates that a node can reach all others quickly, making it a good "spreader" in the network.

4. **Clustering coefficient:**

$$C(i) = \frac{2T(i)}{k_i(k_i-1)}, \quad \texttt{networkx.clustering(G).}$$

   Fraction of a node's neighbors that are also connected to each other. High clustering indicates that i is embedded in a tightly-knit group (a local community or clique).

Visualization of metric evolution

For each metric above:

- After each iteration, store the metric values in a dictionary keyed by iteration number.
- Plot how each node's values evolve across iterations (e.g., line plots or bar plots).
- Use consistent node labeling and legends so that trends are clear.

Your Tasks

1. Implement recursive spectral modularity partitioning to detect multiple communities.
2. Visualize the graph after each split, coloring communities differently.
3. After each split, use `networkx` functions to compute:
   - `degree_centrality`,
   - `betweenness_centrality`,
   - `closeness_centrality`,
   - `clustering`.
4. Plot the evolution of each metric across iterations for every node.
5. Write a short discussion: which nodes consistently remain central across splits, and how community structure influences these metrics.

5 Supporting Mathematical Proofs (for complete understanding)

Recall the modularity matrix (for the whole graph) is

$$\mathbf{B} \;=\; \mathbf{A} \;-\; \frac{\mathbf{k}\,\mathbf{k}^\top}{2m},$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the adjacency matrix, $\mathbf{k} \in \mathbb{R}^n$ is the degree vector ($k_i = \sum_j A_{ij}$), and m is the number of edges (a scalar). Below we collect and prove several elementary but important facts about $\mathbf{B}$.

Proposition 1. *[Reality and symmetry of $\mathbf{B}$] The matrix $\mathbf{B}$ is a real symmetric $n \times n$ matrix.*

Proof. Dimension and reality checks:

- $\mathbf{A}$ is an $n \times n$ matrix with entries $A_{ij} \in \{0, 1\} \subset \mathbb{R}$, hence $\mathbf{A} \in \mathbb{R}^{n\times n}$.
- $\mathbf{k} \in \mathbb{R}^n$ (each k_i is an integer), so $\mathbf{k}\,\mathbf{k}^\top$ is an $n \times n$ real matrix.
- $2m$ is a positive real scalar, so $\dfrac{\mathbf{k}\,\mathbf{k}^\top}{2m} \in \mathbb{R}^{n\times n}$.

Symmetry:

$$\left(\frac{\mathbf{k}\,\mathbf{k}^\top}{2m}\right)^\top = \frac{(\mathbf{k}\,\mathbf{k}^\top)^\top}{2m} = \frac{\mathbf{k}\,\mathbf{k}^\top}{2m},$$

because $(\mathbf{k}\,\mathbf{k}^\top)^\top = \mathbf{k}\,\mathbf{k}^\top$ (rank-one symmetric). Also $\mathbf{A}^\top = \mathbf{A}$ since the graph is undirected. Therefore

$$\mathbf{B}^\top = \mathbf{A}^\top - \frac{(\mathbf{k}\,\mathbf{k}^\top)^\top}{2m} = \mathbf{A} - \frac{\mathbf{k}\,\mathbf{k}^\top}{2m} = \mathbf{B}.$$

Thus $\mathbf{B}$ is symmetric and (by the reality of all entries) a real symmetric matrix. □

Theorem 1. *[Eigenvalues of a real symmetric matrix are real; orthonormal eigenbasis] Let $\mathbf{B} \in \mathbb{R}^{n\times n}$ be real and symmetric. Then:*

1. *All eigenvalues of $\mathbf{B}$ are real numbers.*
2. *There exists an orthonormal basis of $\mathbb{R}^n$ consisting of eigenvectors of $\mathbf{B}$.*

Proof. **(1) Eigenvalues are real.** Suppose $\mathbf{v} \in \mathbb{C}^n$ (possibly complex) and $\lambda \in \mathbb{C}$ satisfy the eigenrelation

$$\mathbf{B}\mathbf{v} = \lambda\mathbf{v}.$$

Take the conjugate-transpose (Hermitian transpose) $\mathbf{v}^*$ of $\mathbf{v}$ and multiply on the left:

$$\mathbf{v}^*\mathbf{B}\mathbf{v} = \lambda\,\mathbf{v}^*\mathbf{v}.$$

Because $\mathbf{B}$ is real and symmetric we have $\mathbf{B} = \mathbf{B}^*$. Therefore the scalar $\mathbf{v}^*\mathbf{B}\mathbf{v}$ satisfies

$$(\mathbf{v}^*\mathbf{B}\mathbf{v})^* = \mathbf{v}^*\mathbf{B}^*(\mathbf{v}^*)^* = \mathbf{v}^*\mathbf{B}^*\mathbf{v} = \mathbf{v}^*\mathbf{B}\mathbf{v},$$

so $\mathbf{v}^*\mathbf{B}\mathbf{v}$ is equal to its complex conjugate and thus is a real number. Also $\mathbf{v}^*\mathbf{v} = \|\mathbf{v}\|_2^2$ is a nonnegative real number (and positive if $\mathbf{v} \neq \mathbf{0}$). From

$$\mathbf{v}^*\mathbf{B}\mathbf{v} = \lambda\, \mathbf{v}^*\mathbf{v}$$

and the fact that the left-hand side and $\mathbf{v}^*\mathbf{v}$ are both real, we deduce that λ must be real (otherwise the product would not be real). Hence every eigenvalue λ is real.

(2) Existence of an orthonormal eigenbasis. This is the standard spectral theorem for real symmetric (Hermitian) matrices: eigenvectors corresponding to distinct eigenvalues are orthogonal, and one can choose an orthonormal basis of eigenvectors for $\mathbb{R}^n$. A short demonstration of orthogonality: if $\mathbf{B}\mathbf{u} = \lambda\mathbf{u}$ and $\mathbf{B}\mathbf{v} = \mu\mathbf{v}$ with $\lambda \neq \mu$, then

$$\lambda(\mathbf{u}^\top\mathbf{v}) = \mathbf{u}^\top(\mathbf{B}\mathbf{v}) = (\mathbf{u}^\top\mathbf{B})\mathbf{v} = (\mathbf{B}\mathbf{u})^\top\mathbf{v} = \lambda(\mathbf{u}^\top\mathbf{v}),$$

and also

$$\mu(\mathbf{u}^\top\mathbf{v}) = \mathbf{u}^\top(\mathbf{B}\mathbf{v}).$$

Subtracting gives $(\lambda - \mu)(\mathbf{u}^\top\mathbf{v}) = 0$, so $\mathbf{u}^\top\mathbf{v} = 0$. Using Gram–Schmidt within eigenspaces provides an orthonormal set spanning $\mathbb{R}^n$. □

Proposition 2 (The all-ones vector is an eigenvector of $\mathbf{B}$ with eigenvalue 0)**.** *Let $\mathbf{1} \in \mathbb{R}^n$ denote the vector whose entries are all ones. Then*

$$\mathbf{B}\,\mathbf{1} \;=\; \mathbf{0},$$

so $\mathbf{1}$ is an eigenvector of $\mathbf{B}$ with eigenvalue 0.

Proof. Compute the product explicitly (dimension check: $\mathbf{B}$ is $n \times n$, $\mathbf{1}$ is $n \times 1$):

$$\mathbf{B}\,\mathbf{1} \;=\; \Big(\mathbf{A} - \frac{\mathbf{k}\,\mathbf{k}^\top}{2m}\Big)\mathbf{1} = \mathbf{A}\mathbf{1} - \frac{\mathbf{k}\,(\mathbf{k}^\top\mathbf{1})}{2m}.$$

By definition of the degree vector, $\mathbf{A}\mathbf{1} = \mathbf{k}$, and

$$\mathbf{k}^\top\mathbf{1} = \sum_{i=1}^{n} k_i = 2m.$$

Substituting gives

$$\mathbf{B}\,\mathbf{1} = \mathbf{k} - \frac{\mathbf{k}\,(2m)}{2m} = \mathbf{k} - \mathbf{k} = \mathbf{0}.$$

Therefore $\mathbf{B}\mathbf{1} = \mathbf{0}$ and 0 is always an eigenvalue of $\mathbf{B}$. □

This fact has a direct consequence for modularity itself.

Corollary 1 (Trivial partition has modularity zero)**.** *If we take the trivial assignment vector $\mathbf{s} = \mathbf{1}$ (all vertices in the same community), then the modularity is*

$$Q = \frac{1}{4m}\,\mathbf{s}^\top\mathbf{B}\mathbf{s} = \frac{1}{4m}\,\mathbf{1}^\top\mathbf{B}\mathbf{1} = 0.$$

Thus the partition that places every node into a single community produces $Q = 0$.

Theorem 2. *[Rayleigh–Ritz characterization] Let $\mathbf{B} \in \mathbb{R}^{n\times n}$ be a real symmetric matrix with ordered eigenvalues*

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$$

and corresponding orthonormal eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_n$. Then

$$\max_{\|\mathbf{s}\|_2=1} \;\mathbf{s}^\top\mathbf{B}\mathbf{s} \;=\; \lambda_1,$$

and the maximum is achieved when $\mathbf{s} = \mathbf{u}_1$ (the leading eigenvector).

Proof. Take any vector $\mathbf{s} \in \mathbb{R}^n$ with $\|\mathbf{s}\|_2 = 1$. Because $\{\mathbf{u}_i\}$ is an orthonormal basis, we can write

$$\mathbf{s} \;=\; \sum_{i=1}^{n} \alpha_i \mathbf{u}_i, \qquad \text{with } \alpha_i = \mathbf{u}_i^\top \mathbf{s}.$$

(Each α_i is a scalar. Dimension check: $\mathbf{s}$ and each $\mathbf{u}_i$ are $n \times 1$ vectors; the scalars α_i make the linear combination valid.)

Since the $\mathbf{u}_i$ are orthonormal and $\|\mathbf{s}\|_2 = 1$, we have

$$\|\mathbf{s}\|_2^2 \;=\; \mathbf{s}^\top \mathbf{s} \;=\; \sum_{i=1}^{n} \alpha_i^2 \;=\; 1.$$

So the coefficients $\{\alpha_i\}$ satisfy the unit-sum constraint $\sum_i \alpha_i^2 = 1$.

Use $\mathbf{B}\mathbf{u}_i = \lambda_i \mathbf{u}_i$ and linearity:

$$\mathbf{s}^\top \mathbf{B}\mathbf{s} = \Big(\sum_{i=1}^{n} \alpha_i \mathbf{u}_i\Big)^\top \mathbf{B} \Big(\sum_{j=1}^{n} \alpha_j \mathbf{u}_j\Big) = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j \mathbf{u}_i^\top \mathbf{B}\mathbf{u}_j.$$

Because $\mathbf{u}_j$ is an eigenvector,

$$\mathbf{u}_i^\top \mathbf{B}\mathbf{u}_j = \mathbf{u}_i^\top (\lambda_j \mathbf{u}_j) = \lambda_j (\mathbf{u}_i^\top \mathbf{u}_j).$$

But $\mathbf{u}_i^\top \mathbf{u}_j = 0$ for $i \neq j$ and $= 1$ for $i = j$. Therefore all cross terms vanish, leaving

$$\mathbf{s}^\top \mathbf{B}\mathbf{s} = \sum_{i=1}^{n} \alpha_i^2 \lambda_i.$$

Since λ_1 is the largest eigenvalue and $\alpha_i^2 \geq 0$ with $\sum_i \alpha_i^2 = 1$,

$$\mathbf{s}^\top \mathbf{B}\mathbf{s} = \sum_{i=1}^{n} \alpha_i^2 \lambda_i \leq \sum_{i=1}^{n} \alpha_i^2 \lambda_1 = \lambda_1 \sum_{i=1}^{n} \alpha_i^2 = \lambda_1.$$

Thus for any unit vector $\mathbf{s}$ we have $\mathbf{s}^\top \mathbf{B}\mathbf{s} \leq \lambda_1$.

Choose $\mathbf{s} = \mathbf{u}_1$. Then $\alpha_1 = 1$, $\alpha_{i \neq 1} = 0$, so

$$\mathbf{u}_1^\top \mathbf{B}\mathbf{u}_1 = \lambda_1.$$

Therefore the maximum value λ_1 is attained and the maximizer is $\mathbf{u}_1$. If λ_1 has multiplicity greater than one, any unit vector in the eigenspace corresponding to λ_1 attains the same maximum.

The relaxed problem

$$\max_{\|\mathbf{s}\|_2 = 1} \mathbf{s}^\top \mathbf{B}\mathbf{s}$$

is solved by the leading eigenvector $\mathbf{u}_1$ and the maximum value equals the largest eigenvalue λ_1. This is the Rayleigh–Ritz characterization of eigenvalues. It justifies using the leading eigenvector as the continuous solution to the modularity maximization problem; rounding its entries by sign then produces a discrete ± 1 partition. □

Theorem 3. *[Eigenvalue stopping criterion for recursive bisection] Let $C \subseteq V$ be a group of nodes (a candidate community), and let $\mathbf{B}^{(C)}$ denote the restriction of the global modularity matrix $\mathbf{B}$ to the rows and columns indexed by C (so $\mathbf{B}^{(C)}$ is a $|C| \times |C|$ real symmetric matrix). For any split of C described by a vector $\mathbf{s} \in \{-1, +1\}^{|C|}$, the change in modularity produced by splitting C into two parts according to $\mathbf{s}$ equals*

$$\Delta Q \;=\; \frac{1}{4m}\, \mathbf{s}^\top \mathbf{B}^{(C)} \mathbf{s}.$$

Consequently, if the largest eigenvalue of $\mathbf{B}^{(C)}$ satisfies $\lambda_1^{(C)} \leq 0$, then $\Delta Q \leq 0$ for every possible split $\mathbf{s}$, so no *partition of C increases the modularity. Therefore recursion should stop on C.*

Proof. **(Change-in-modularity identity).** The modularity expression for a partition of the whole graph is

$$Q \;=\; \frac{1}{4m}\,\mathbf{t}^\top \mathbf{B}\mathbf{t},$$

where $\mathbf{t} \in \{-1,+1\}^n$ is the global label vector (one entry per node). If we only change the labels of nodes inside C (leaving labels outside C unchanged), the contribution to Q that changes comes only from pairs (i,j) with both $i,j \in C$. Pulling out those terms gives precisely the restricted quadratic form $\mathbf{s}^\top \mathbf{B}^{(C)}\mathbf{s}$, where $\mathbf{s}$ is the $|C| \times 1$ vector of labels for nodes in C. (This is the same identity used in Newman's derivation; algebraically, it follows by writing the double sum defining $\mathbf{s}^\top \mathbf{B}\mathbf{s}$ and restricting to indices inside C.)

Thus the modularity gain from splitting C as specified equals $\Delta Q = \frac{1}{4m}\,\mathbf{s}^\top \mathbf{B}^{(C)}\mathbf{s}$.

(Use Rayleigh–Ritz to obtain the criterion). Because $\mathbf{B}^{(C)}$ is real symmetric it has real eigenvalues. Let $\lambda_1^{(C)}$ be its largest eigenvalue. The Rayleigh–Ritz theorem (see earlier) implies that for any vector $\mathbf{x} \in \mathbb{R}^{|C|}$,

$$\mathbf{x}^\top \mathbf{B}^{(C)}\mathbf{x} \;\le\; \lambda_1^{(C)}\,\|\mathbf{x}\|_2^2.$$

Apply this with $\mathbf{x} = \mathbf{s} \in \{\pm 1\}^{|C|}$: then $\|\mathbf{s}\|_2^2 = |C|$ and

$$\mathbf{s}^\top \mathbf{B}^{(C)}\mathbf{s} \;\le\; \lambda_1^{(C)}\,|C|.$$

If $\lambda_1^{(C)} \le 0$, the right-hand side is ≤ 0, so $\mathbf{s}^\top \mathbf{B}^{(C)}\mathbf{s} \le 0$ for every $\mathbf{s}$. Hence every possible split of C yields $\Delta Q \le 0$, i.e., no split increases modularity. Therefore it is safe to stop splitting C. □

Remarks and intuition.

- For the *full* matrix $\mathbf{B}$ (the whole graph) we always have the all-ones vector $\mathbf{1}$ as an eigenvector with eigenvalue 0 (Proposition above). Thus the global $\mathbf{B}$ always has at least one eigenvalue equal to 0.
- When we consider a restricted block $\mathbf{B}^{(C)}$ (the submatrix for a subset C), the row-sum property need not hold and 0 is not guaranteed to be an eigenvalue of $\mathbf{B}^{(C)}$. That is why $\lambda_1^{(C)}$ can be positive, negative, or zero.
- The stopping rule "stop when $\lambda_1^{(C)} \le 0$" is exactly the algebraic expression of the idea "no split of C will yield a positive modularity gain."