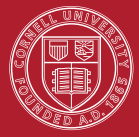


Cornell University

Lecture 6 - Graph partition and Node Centrality

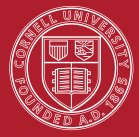
Graph-Based Data Science For Networked Systems
ECE 5260 – ORIE 5735

Anna Scaglione

Cornell Tech

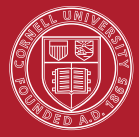
February 9, 2026

Content



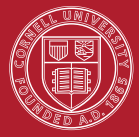
Cornell University

Outline



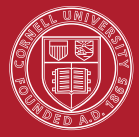
Cornell University

Summary



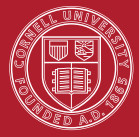
- We introduced basic graph theory concepts, defining weighted and unweighted graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and graph classes
- We studied these graphs properties using:
  - The adjacency matrix A:
    - the basic algebraic structrure to highlight nodes connectivity
  - The incidende matrix B:
    - the basic structure to differentiate nodal quantities and model the impact of edge flows
  - The Laplacian matrix L:
    - a combination of the two. Its spectrum is tied to graph connectivity
  - The Hodge Laplacian which explores the simplicial structure of the graph (model higher-order interactions)
  - We introduced algorithms to solve network flow problems and graph partitioning

Summary



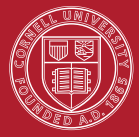
- We introduced basic graph theory concepts, defining weighted and unweighted graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and graph classes
- We studied these graphs properties using:
  - The adjacency matrix A:
    - the basic algebraic structrure to highlight nodes connectivity
  - The incidende matrix B:
    - the basic structure to differentiate nodal quantities and model the impact of edge flows
  - The Laplacian matrix L:
    - a combination of the two. Its spectrum is tied to graph connectivity
  - The Hodge Laplacian which explores the simplicial structure of the graph (model higher-order interactions)
  - We introduced algorithms to solve network flow problems and graph partitioning

Summary



- We introduced basic graph theory concepts, defining weighted and unweighted graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and graph classes
- We studied these graphs properties using:
  - The adjacency matrix $\boldsymbol{A}$:
    - the basic algebraic structrure to highlight nodes connectivity
  - The incidende matrix $\boldsymbol{B}$:
    - the basic structure to differentiate nodal quantities and model the impact of edge flows
  - The Laplacian matrix $\boldsymbol{L}$:
    - a combination of the two. Its spectrum is tied to graph connectivity
  - The Hodge Laplacian which explores the simplicial structure of the graph (model higher-order interactions)
  - We introduced algorithms to solve network flow problems and graph partitioning

Summary

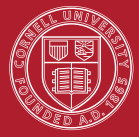


- We introduced basic graph theory concepts, defining weighted and unweighted graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and graph classes
- We studied these graphs properties using:
  - The adjacency matrix $\boldsymbol{A}$:
    - the basic algebraic structrure to highlight nodes connectivity
  - The incidende matrix $\boldsymbol{B}$:
    - the basic structure to differentiate nodal quantities and model the impact of edge flows
  - The Laplacian matrix $\boldsymbol{L}$:
    - a combination of the two. Its spectrum is tied to graph connectivity
  - The Hodge Laplacian which explores the simplicial structure of the graph (model higher-order interactions)
  - We introduced algorithms to solve network flow problems and graph partitioning

Summary

- We introduced basic graph theory concepts, defining weighted and unweighted graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and graph classes
- We studied these graphs properties using:
  - The adjacency matrix $\boldsymbol{A}$:
    - the basic algebraic structrure to highlight nodes connectivity
  - The incidende matrix $\boldsymbol{B}$:
    - the basic structure to differentiate nodal quantities and model the impact of edge flows
  - The Laplacian matrix $\boldsymbol{L}$:
    - a combination of the two. Its spectrum is tied to graph connectivity
  - The Hodge Laplacian which explores the simplicial structure of the graph (model higher-order interactions)
  - We introduced algorithms to solve network flow problems and graph partitioning

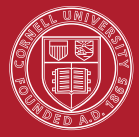
Introduction



Cornell University

- From these matrices we have derived through algebra a set of graph properties and explored models that apply to networked systems that use them.
- Last time in particular we talked about the Laplacian, some results in spectral graph theory and introduced the Hodge Laplacian and introduced the problem of clustering a network and the spectral clustering method.
- We also defined the Hodge-Laplacian

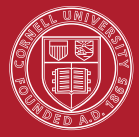
Introduction



Cornell University

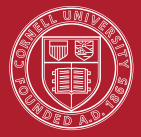
- From these matrices we have derived through algebra a set of graph properties and explored models that apply to networked systems that use them.
- Last time in particular we talked about the Laplacian, some results in spectral graph theory and introduced the Hodge Laplacian and introduced the problem of clustering a network and the spectral clustering method.
- We also defined the Hodge-Laplacian

Overview



- We will briefly see a key spectral property of the Hodge Laplacian $\boldsymbol{L}_k$
- We want to better explain the algorithms for graph partitioning using the Laplacian matrix $\boldsymbol{L}$
- We will take a first look first at network structural features (measures and metrics) through the notion of centrality

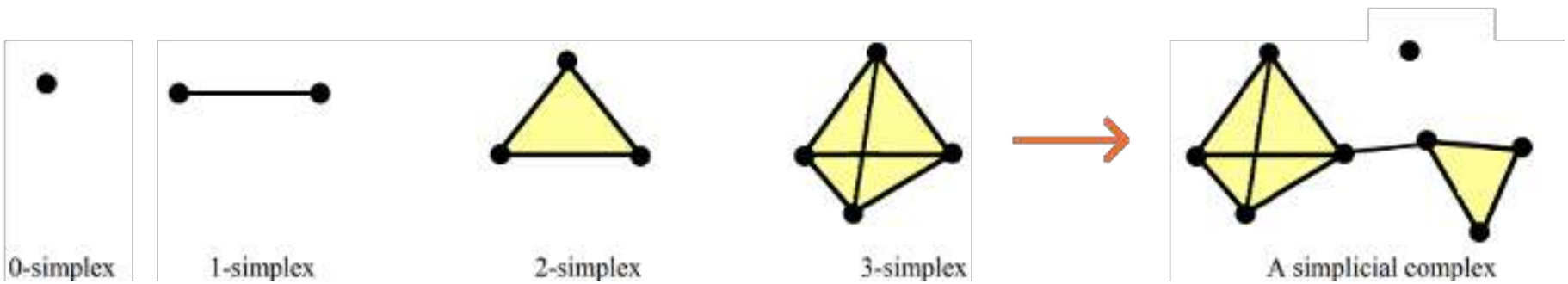
Outline



Cornell University

Simplicial Complexes

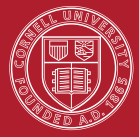
- A **simplicial complex** $\boldsymbol{x}_K$ of order K is a collection of simplices $\mathcal{S}_k$ for $k = 0, \ldots, K$.
- A k-simplex is a set of $k+1$ vertices and, if the nodes are embedded in space, it corresponds to a k-dimensional polytope that is the convex hull of its $k+1$ vertices. For example:
  - Nodes (0-simplices)
  - Edges (1-simplices)
  - Triangles (2-simplices)
  - Tetrahedrons (3-simplices)
- If $\mathcal{S}_k \in \boldsymbol{x}_K$, then all its faces $\mathcal{S}_{k-1}$ must also be in $\boldsymbol{x}_K$.



Betti Numbers and the Hodge Laplacian in Graphs

- **Betti numbers** are topological invariants that describe the number of independent cycles in a space.
- In graph theory, they help understand the topology of a network.
- The **Hodge Laplacian** provides a spectral decomposition of graph structure and relates to Betti numbers via the topology of the graph.
- While the spectrum of $\boldsymbol{L}_0$, the regular Laplacian, is tied to the number of connected components of the graphs, for higher orders $\boldsymbol{L}_k$ provides information about the presence of *holes* and cycles in the network fabric.

Betti Numbers: Definition



Given a simplicial complex X, the n-th **Betti number** is:

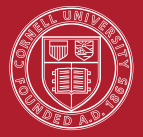
$$b_n = \text{rank } H_n(X),$$

where $H_n(X)$ is the n-th homology group.

Intuition:

- b_0: Number of connected components.
- b_1: Number of independent cycles.
- b_2: Number of independent voids / cavities (in higher dimensions)
- ....

Independent Cycles and the Cycle Space



Cycle space

Given an undirected graph $G = (V, E)$ with an arbitrary orientation of its edges, let $\boldsymbol{B}_1 \in \mathbb{R}^{|V| \times |E|}$ be the node–edge incidence matrix. The *cycle space* of the graph is defined as

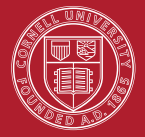
$$\mathcal{C} \;:=\; \ker \boldsymbol{B}_1.$$

where $\ker$ is the Kernel of the matrix $\boldsymbol{B}_1$. Vectors $\boldsymbol{f} \in \mathcal{C}$, i.e. such that

$$\boldsymbol{B}_1 \boldsymbol{f} = \mathbf{0},$$

represent edge flows that satisfy **flow conservation** at every node.

Independent Cycles and the Cycle Space



Independent cycles

A set of cycles is said to be *independent* if the corresponding vectors in $\ker \boldsymbol{B}_1$ are linearly independent. The number of independent cycles in the graph is therefore

$$b_1 = \dim(\ker \boldsymbol{B}_1),$$

which is the **first Betti** number of the graph.

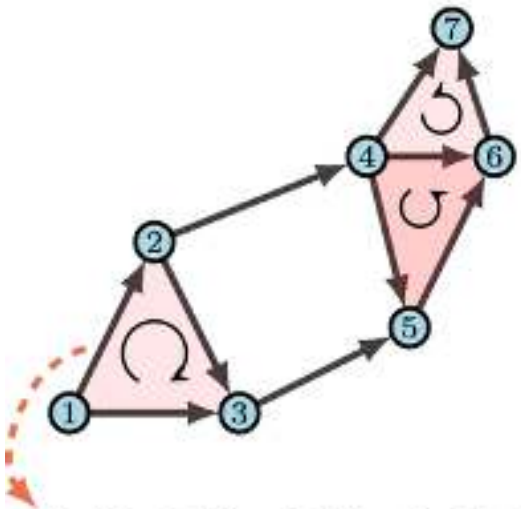
Interpretation

Each independent cycle corresponds to one degree of freedom for circulating flow that does not accumulate at any node.
Equivalently, each edge added to a spanning tree creates exactly one new independent cycle.

Incidence Matrices

The relationships between simplices are captured by incidence matrices:

- B_1: Node-to-edge incidence matrix.
- B_2: Edge-to-triangle incidence matrix.
- ...
- B_k: Edge-to-k-simplexes incidence matrix.



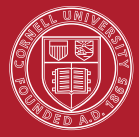
$$\partial_2([1,2,3]) = [1,2] - [1,3] + [2,3]$$

$$B_1 = \begin{array}{c|cccccccccc} & [1,2] & [1,3] & [2,3] & [2,4] & [3,5] & [4,5] & [4,6] & [4,7] & [5,6] & [6,7] \\ \hline [1] & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ [2] & 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ [3] & 0 & 1 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ [4] & 0 & 0 & 0 & 1 & 0 & -1 & -1 & -1 & 0 & 0 \\ [5] & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & -1 & 0 \\ [6] & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -1 \\ [7] & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

$$B_2 = \begin{array}{c|ccc} & [1,2,3] & [4,5,6] & [4,6,7] \\ \hline [1,2] & 1 & 0 & 0 \\ [1,3] & -1 & 0 & 0 \\ [2,3] & 1 & 0 & 0 \\ [2,4] & 0 & 0 & 0 \\ [3,5] & 0 & 0 & 0 \\ [4,5] & 0 & 1 & 0 \\ [4,6] & 0 & -1 & 1 \\ [4,7] & 0 & 0 & -1 \\ [5,6] & 0 & 1 & 0 \\ [6,7] & 0 & 0 & 1 \end{array}$$

They encode the boundary information between simplex orders.

From Cycles to Homology



Chains and boundary operators

For a simplicial complex X, let C_k denote the vector space of k-chains, i.e., formal linear combinations of k-simplices. The boundary operator

$$\partial_k : C_k \to C_{k-1}$$

maps a k-simplex to the signed sum of its $(k-1)$-dimensional faces. In matrix form, ∂_k is represented by the incidence matrix $\boldsymbol{B}_k$.

k-cycles and k-boundaries

- k-**cycles**: chains with no boundary

$$\boldsymbol{Z}_k := \ker \partial_k = \ker \boldsymbol{B}_k$$

- k-**boundaries**: chains that are boundaries of higher-order simplices

$$\boldsymbol{B}_k := \operatorname{im} \partial_{k+1} = \operatorname{im} \boldsymbol{B}_{k+1}$$

Hodge Laplacian

- The Hodge Laplacian at level k is defined as:

$$\boldsymbol{L}_k = \boldsymbol{B}_k^T \boldsymbol{B}_k + \boldsymbol{B}_{k+1} \boldsymbol{B}_{k+1}^T$$

  the regular Laplacian is $\boldsymbol{L}_0$ as $\boldsymbol{B}_0 = \boldsymbol{0}$
- Decomposed into:
  - Lower Laplacian: $\boldsymbol{L}_k^l = \boldsymbol{B}_k^T \boldsymbol{B}_k$
  - Upper Laplacian: $\boldsymbol{L}_k^u = \boldsymbol{B}_{k+1} \boldsymbol{B}_{k+1}^T$
- The kernel of $\boldsymbol{L}_k$ gives Betti numbers (the number of zero eigenvalues of $\boldsymbol{L}_k$):

$$b_k = \dim(\ker \boldsymbol{L}_k) = \nu(\boldsymbol{L}_k). \tag{1}$$

Homology Groups and Betti Numbers

Cornell University

The k-th homology group

The k-th homology group is defined as

$$H_k := \boldsymbol{Z}_k / \boldsymbol{B}_k,$$

i.e., the space of k-cycles modulo those that are boundaries of $(k+1)$-simplices.

Betti numbers

The k-th Betti number is the dimension of the homology group:

$$b_k := \dim H_k.$$

It counts the number of *independent k-dimensional holes* in the simplicial complex.

Betti Numbers and the Genus of a Surface

Two different but related notions

- **Betti numbers** are *intrinsic* topological invariants: they depend only on the connectivity of the simplicial complex.
- The **genus** g is an *extrinsic* geometric notion: it depends on how a graph is embedded on a surface.

For a graph viewed as a 1-dimensional simplicial complex:

- b_1 counts the total number of independent cycles.
- The genus g counts the minimum number of *handles* required to embed the graph without edge crossings.

Key message

A graph may have many independent cycles ($b_1 \gg 0$) and still be planar ($g = 0$). Genus measures how *entangled* the cycles are, not how many there are.

Euler Characteristic: Linking Cycles and Genus

Euler characteristic of a surface embedding

If a connected graph is embedded on a surface of genus g, then

$$\chi = |\mathcal{V}| - |\mathcal{E}| + |\mathcal{F}| = 2 - 2g,$$

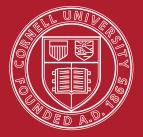
where $|\mathcal{F}|$ is the number of faces induced by the embedding and χ is called Euler characteristic. Since for a connected graph, $b_1 = \ker \boldsymbol{B}_1 = |\mathcal{E}| - |\mathcal{V}| + 1$. Combining the two expressions yields:

$$b_1 = |\mathcal{F}| + 2g - 1.$$

Interpretation:

- b_1 counts *all* independent cycles.
- Faces account for cycles that can be "filled in".
- Genus counts the cycles that *cannot* be removed without introducing crossings.

Betti Numbers in Graphs



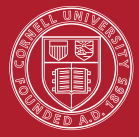
For a graph G:

- $b_0 = \#$ of connected components.
- $b_1 = \#$ of independent cycles (from cycle space of the graph).

Computation from Hodge Laplacian:

- $b_0 = \dim(\ker \boldsymbol{L}_0)$ (0th Laplacian, connected components).
- $b_1 = \dim(\ker \boldsymbol{L}_1)$ (1st Laplacian, cycle space).

Outline



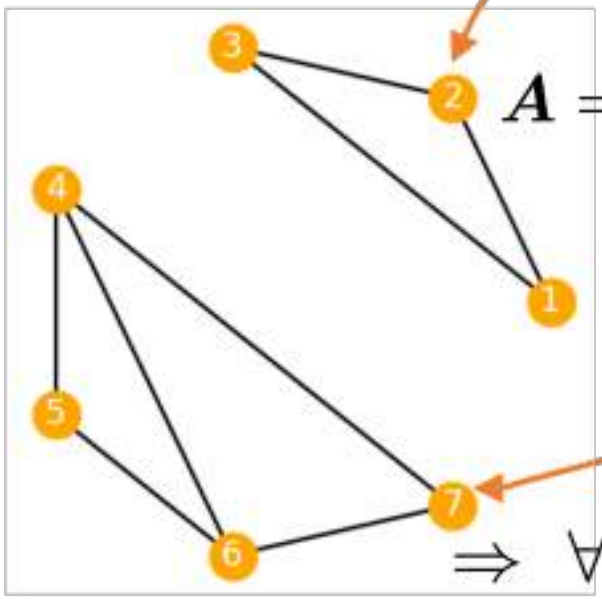
Cornell University

Recall what happens to disconnected graphs

The basic theorem of spectral graph theory is:

Theorem (M. Fiedler)

The nullity (number of zero eigenvalues) of $\boldsymbol{L}$ equals the number of connected components of a graph.

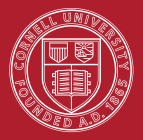


$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_2 \end{bmatrix} \Rightarrow \boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{L}_2 \end{bmatrix}$$

Another null vector

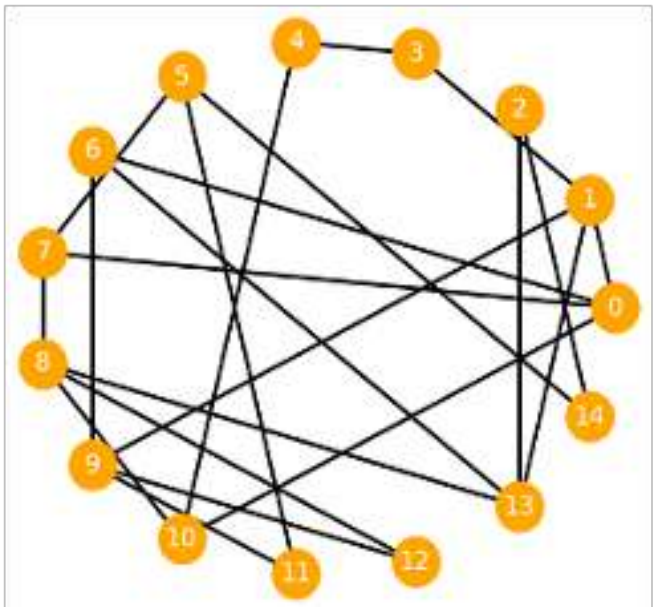
$$\Rightarrow \forall \alpha \quad \boldsymbol{L} \begin{bmatrix} \alpha \boldsymbol{1} \\ \boldsymbol{1} \end{bmatrix} = \begin{bmatrix} \alpha \boldsymbol{L}_1 \boldsymbol{1} \\ \boldsymbol{L}_2 \boldsymbol{1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{0}$$

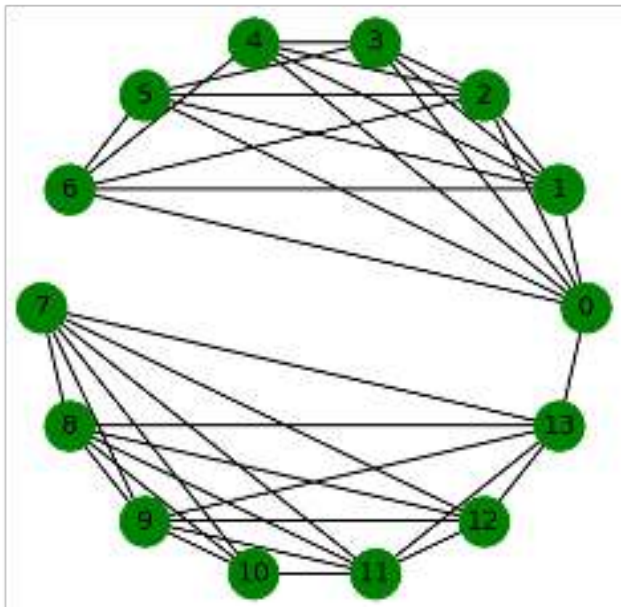
Graph partitioning and communities



Cornell University

- Even if the graph is connected, the second smallest eigenvalue of the Laplacian (called **algebraic connectivity** or **Fielder eigenvalue**) $\lambda_{N-1} > 0$ can be very small in two cases:
  1. When the network is sparsely connected throughout (it has low density $\delta = 2\frac{|\mathcal{E}|}{|\mathcal{V}|(|\mathcal{V}|-1)}$), e.g. cycle graph when $|\mathcal{V}| \gg 1$
  2. Networks that are dense, due to dense clusters, which are loosely connected among each other.

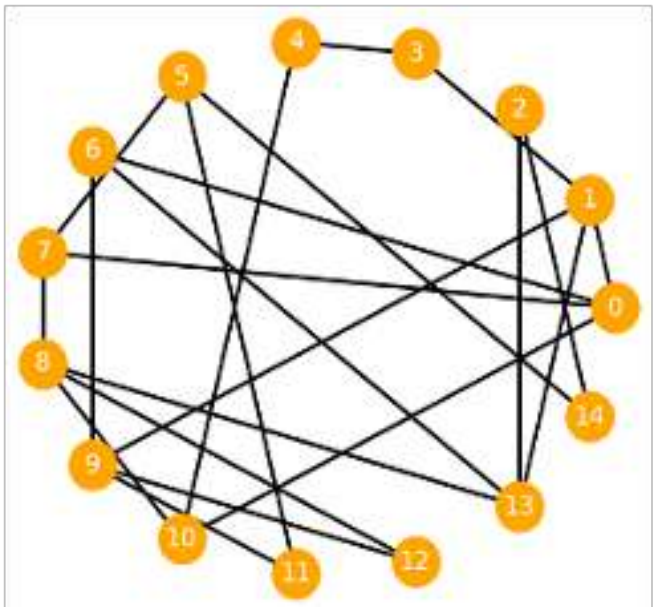


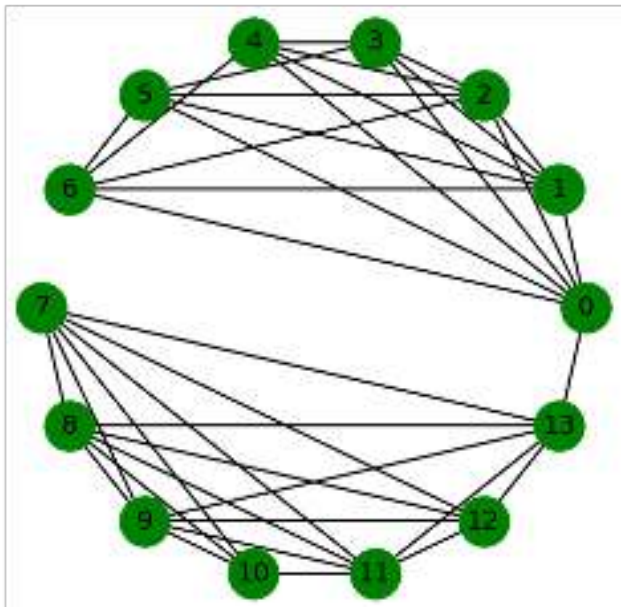


Graph partitioning and communities

Cornell University

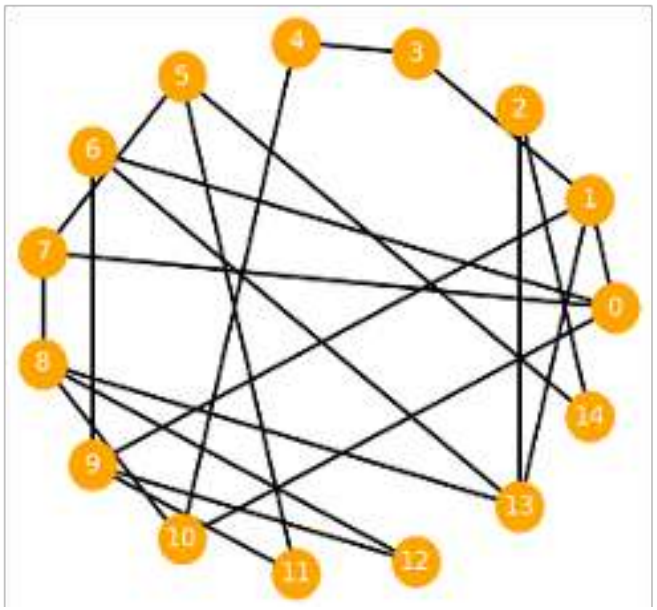
- Even if the graph is connected, the second smallest eigenvalue of the Laplacian (called **algebraic connectivity** or **Fielder eigenvalue**) $\lambda_{N-1} > 0$ can be very small in two cases:
  1. When the network is sparsely connected throughout (it has low density $\delta = 2\frac{|\mathcal{E}|}{|\mathcal{V}|(|\mathcal{V}|-1)}$), e.g. cycle graph when $|\mathcal{V}| \gg 1$
  2. Networks that are dense, due to dense clusters, which are loosely connected among each other.

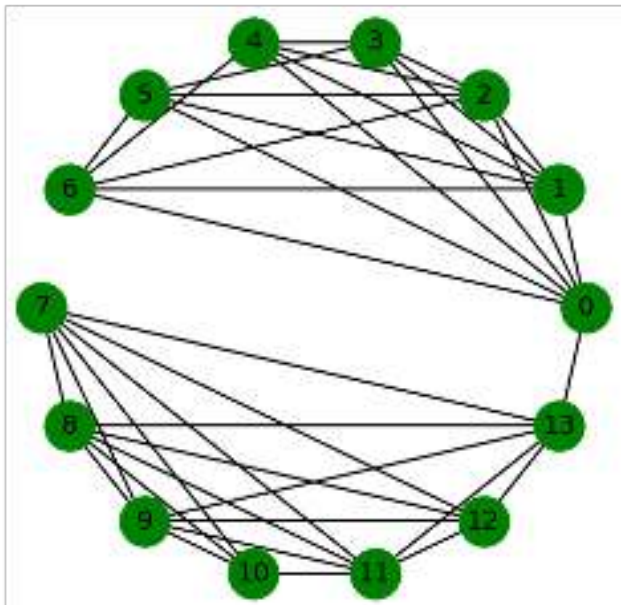




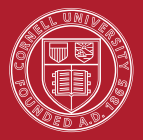
Graph partitioning and communities

- Even if the graph is connected, the second smallest eigenvalue of the Laplacian (called **algebraic connectivity** or **Fielder eigenvalue**) $\lambda_{N-1} > 0$ can be very small in two cases:
  1. When the network is sparsely connected throughout (it has low density $\delta = 2\frac{|\mathcal{E}|}{|\mathcal{V}|(|\mathcal{V}|-1)}$), e.g. cycle graph when $|\mathcal{V}| \gg 1$
  2. Networks that are dense, due to dense clusters, which are loosely connected among each other.





Graph with dense sections (clusters)



Cornell University

- For instance, there could be two portions, corresponding to the blocks $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$, that are dense but not well connected:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{21} & \boldsymbol{A}_2 \end{bmatrix} \quad \|\boldsymbol{B}_{12}\|_0 \ll |\mathcal{V}_1|, \ \|\boldsymbol{B}_{21}\|_0 \ll |\mathcal{V}_2|$$

- In fact, we can view the adjacency matrix of this graph as a *perturbed version of that of a disconnected graph* A^{disc}:

$$A = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} + \overbrace{\begin{bmatrix} 0 & B_{12} \\ B_{21} & 0 \end{bmatrix}}^{\delta A \text{ which is sparse}} \approx \overbrace{\begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}}^{A^{\text{disc}}}$$

Graph with dense sections (clusters)

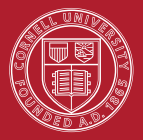
- For instance, there could be two portions, corresponding to the blocks $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$, that are dense but not well connected:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{21} & \boldsymbol{A}_2 \end{bmatrix} \quad \|\boldsymbol{B}_{12}\|_0 \ll |\mathcal{V}_1|, \ \ \|\boldsymbol{B}_{21}\|_0 \ll |\mathcal{V}_2|$$

- In fact, we can view the adjacency matrix of this graph as a *perturbed version of that of a disconnected graph* $\boldsymbol{A}^{\text{disc}}$:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_2 \end{bmatrix} + \overbrace{\begin{bmatrix} \boldsymbol{0} & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{21} & \boldsymbol{0} \end{bmatrix}}^{\delta \boldsymbol{A} \text{ which is sparse}} \approx \overbrace{\begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_2 \end{bmatrix}}^{\boldsymbol{A}^{\text{disc}}}$$

Null vectors of the Laplacian



- A similar partition can be done for the Laplacian:

$$\boldsymbol{L} = \boldsymbol{L}^{\text{disc}} + \delta \boldsymbol{L}$$

where $\delta \boldsymbol{L} = -\delta \boldsymbol{A}$, since $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$, so the off diagonal elements of $\boldsymbol{L}$ have simply a $-$ sign compared to those of $\boldsymbol{A}$.

- We know that $\boldsymbol{L}^{\text{disc}}$ has Fiedler eigenvalue $\lambda_{N-1}^{\text{disc}} = 0$ and a two dimensional null-space, because the corresponding graph has two components.

- Let $N = |\mathcal{V}|$. The null space of $\boldsymbol{L}^{\text{disc}}$ is spanned by two vectors:
  1. one vector is $\boldsymbol{u}_N = \frac{1}{\sqrt{N}}\mathbf{1}$ (where $\frac{1}{\sqrt{N}}$ makes $\|\boldsymbol{u}_N\| = 1$),
  2. the other one which we can choose to be[1] $\boldsymbol{u}_{N-1} \propto [\alpha \mathbf{1}^\top, \mathbf{1}^\top]^\top$ so that $\boldsymbol{u}_{N-1}^\top \boldsymbol{u}_N = \mathbf{0}$ and $\|\boldsymbol{u}_{N-1}\| = 1$.

[1] $\boldsymbol{a} \propto \boldsymbol{b}$ means $\boldsymbol{a}$ is proportional to $\boldsymbol{b}$, i.e. $\boldsymbol{a} = \gamma \boldsymbol{b}$

Null vectors of the Laplacian

- A similar partition can be done for the Laplacian:

$$\boldsymbol{L} = \boldsymbol{L}^{\text{disc}} + \delta \boldsymbol{L}$$

where $\delta \boldsymbol{L} = -\delta \boldsymbol{A}$, since $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$, so the off diagonal elements of $\boldsymbol{L}$ have simply a $-$ sign compared to those of $\boldsymbol{A}$.

- We know that $\boldsymbol{L}^{\text{disc}}$ has Fiedler eigenvalue $\lambda_{N-1}^{\text{disc}} = 0$ and a two dimensional null-space, because the corresponding graph has two components.

- Let $N = |\mathcal{V}|$. The null space of $\boldsymbol{L}^{\text{disc}}$ is spanned by two vectors:
  1. one vector is $\boldsymbol{u}_N = \frac{1}{\sqrt{N}}\mathbf{1}$ (where $\frac{1}{\sqrt{N}}$ makes $\|\boldsymbol{u}_N\| = 1$),
  2. the other one which we can choose to be[1] $\boldsymbol{u}_{N-1} \propto [\alpha \mathbf{1}^\top, \mathbf{1}^\top]^\top$ so that $\boldsymbol{u}_{N-1}^\top \boldsymbol{u}_N = \mathbf{0}$ and $\|\boldsymbol{u}_{N-1}\| = 1$.

[1] $\boldsymbol{a} \propto \boldsymbol{b}$ means $\boldsymbol{a}$ is proportional to $\boldsymbol{b}$, i.e. $\boldsymbol{a} = \gamma \boldsymbol{b}$

Null vectors of the Laplacian

- A similar partition can be done for the Laplacian:

$$\boldsymbol{L} = \boldsymbol{L}^{\text{disc}} + \delta \boldsymbol{L}$$

where $\delta \boldsymbol{L} = -\delta \boldsymbol{A}$, since $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$, so the off diagonal elements of $\boldsymbol{L}$ have simply a $-$ sign compared to those of $\boldsymbol{A}$.

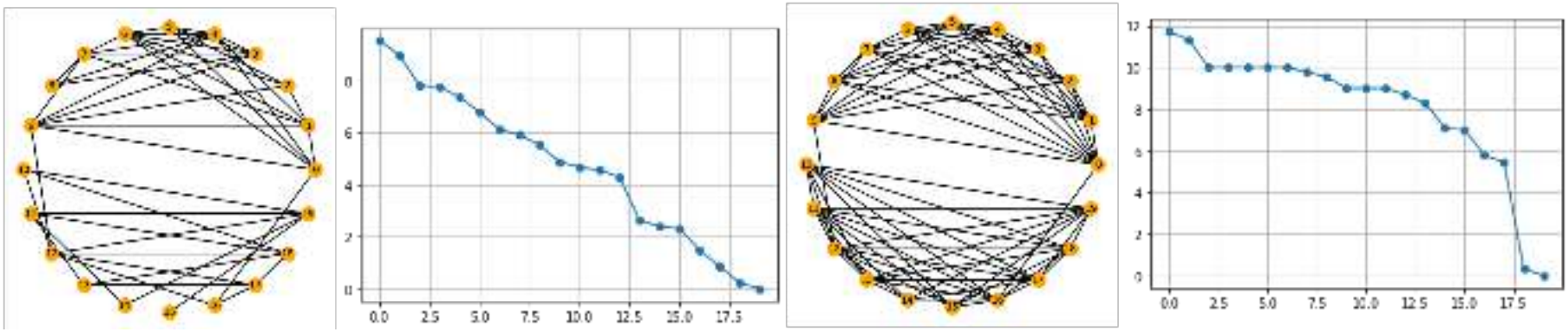
- We know that $\boldsymbol{L}^{\text{disc}}$ has Fiedler eigenvalue $\lambda_{N-1}^{\text{disc}} = 0$ and a two dimensional null-space, because the corresponding graph has two components.

- Let $N = |\mathcal{V}|$. The null space of $\boldsymbol{L}^{\text{disc}}$ is spanned by two vectors:
  1. one vector is $\boldsymbol{u}_N = \frac{1}{\sqrt{N}}\mathbf{1}$ (where $\frac{1}{\sqrt{N}}$ makes $\|\boldsymbol{u}_N\| = 1$),
  2. the other one which we can choose to be[1] $\boldsymbol{u}_{N-1} \propto [\alpha \mathbf{1}^\top, \mathbf{1}^\top]^\top$ so that $\boldsymbol{u}_{N-1}^\top \boldsymbol{u}_N = \mathbf{0}$ and $\|\boldsymbol{u}_{N-1}\| = 1$.

[1] $\boldsymbol{a} \propto \boldsymbol{b}$ means $\boldsymbol{a}$ is proportional to $\boldsymbol{b}$, i.e. $\boldsymbol{a} = \gamma \boldsymbol{b}$

Graph with dense sections (clusters)

- We said an indicator of the presence of dense communities loosely connected is the **spectral gap** of the Laplacian $\gamma_k = |\lambda_{N-k-1} - \lambda_{N-k}|$ when there are k clusters in the graph
- Next we give a mathematical reason why.



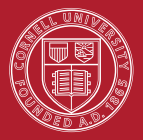
Eigenvector of perturbed matrices

- Denote the eigenvectors of $\boldsymbol{L}$ by $\boldsymbol{u}_n$, those of $\boldsymbol{L}^{\text{disc}}$ by $\boldsymbol{u}_n^{\text{disc}}$ (we continue to focus on the case where there are two dense clusters loosely connected).
- Both $\boldsymbol{L}$ and $\boldsymbol{L}^{\text{disc}}$ are such that $\boldsymbol{L}\mathbf{1} = \mathbf{0}$ and $\boldsymbol{L}^{\text{disc}}\mathbf{1} = \mathbf{0}$, so $\boldsymbol{u}_N = \boldsymbol{u}_N^{\text{disc}} = \frac{1}{\sqrt{N}}\mathbf{1}$.
- The Fiedler eigenvalue $\lambda_{N-1} > 0$ but a theorem, by Weyl shows that $\lambda_{N-1} \leq \|\delta \boldsymbol{L}\|_2$
- For the eigenvectors, a result for perturbed matrices in general:

$$\boldsymbol{u}_{N-1}^{\top}\boldsymbol{u}_n^{\text{disc}} \leq \gamma^{-1}\|\delta \boldsymbol{L}\|_2, \quad \gamma = \min_k |\lambda_{N-k}(\boldsymbol{L}^{\text{disc}}) - \lambda_{N-k-1}(\boldsymbol{L}^{\text{disc}})|$$

is the minimum spectral gap.

Eigenvector of perturbed matrices

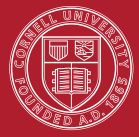


- Denote the eigenvectors of $\boldsymbol{L}$ by $\boldsymbol{u}_n$, those of $\boldsymbol{L}^{\text{disc}}$ by $\boldsymbol{u}_n^{\text{disc}}$ (we continue to focus on the case where there are two dense clusters loosely connected).
- Both $\boldsymbol{L}$ and $\boldsymbol{L}^{\text{disc}}$ are such that $\boldsymbol{L}\mathbf{1} = \mathbf{0}$ and $\boldsymbol{L}^{\text{disc}}\mathbf{1} = \mathbf{0}$, so $\boldsymbol{u}_N = \boldsymbol{u}_N^{\text{disc}} = \frac{1}{\sqrt{N}}\mathbf{1}$.
- The Fiedler eigenvalue $\lambda_{N-1} > 0$ but a theorem, by Weyl shows that $\lambda_{N-1} \leq \|\delta \boldsymbol{L}\|_2$
- For the eigenvectors, a result for perturbed matrices in general:

$$\boldsymbol{u}_{N-1}^{\top}\boldsymbol{u}_n^{\text{disc}} \leq \gamma^{-1}\|\delta \boldsymbol{L}\|_2, \quad \gamma = \min_k |\lambda_{N-k}(\boldsymbol{L}^{\text{disc}}) - \lambda_{N-k-1}(\boldsymbol{L}^{\text{disc}})|$$

is the minimum spectral gap.

Graph partitioning and communities



Cornell University

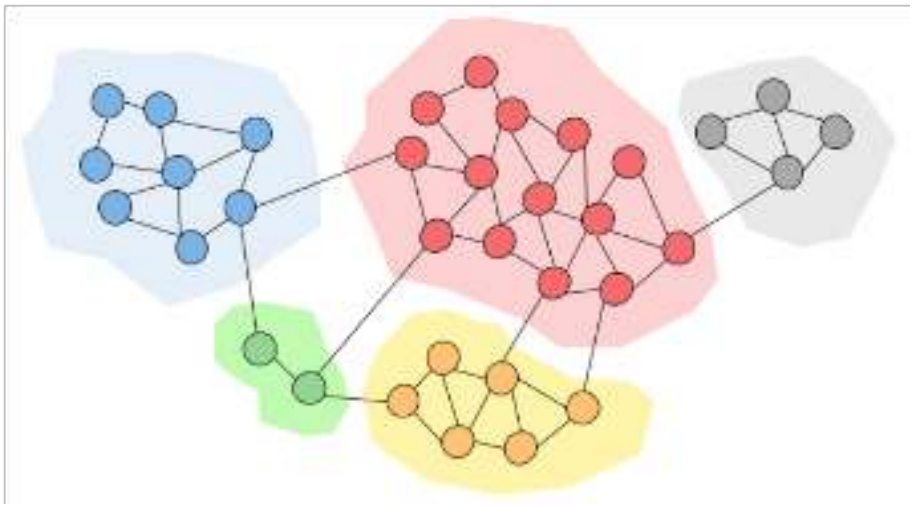
- A very common task in the analysis of graph data is to try to identify clusters that partition the network in communities.
- The most commonly used algorithm to do so is called Spectral Clustering and it works because of what we discussed.
- What we seek is a **partition** of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that cuts the network in parts.
- We will see that the previous observations are helpful in providing intuition on why the algorithm works.

Graph partitioning and communities

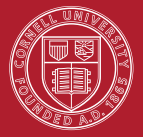
Cornell University

Partition

- A partition of the nodes in the network, denoted by the subsets $\mathcal{V}_k \subseteq \mathcal{V}$ such that:
  1) $\bigcup_{k=1}^{K} \mathcal{V}_k = \mathcal{V}$;
  2) $\mathcal{V}_k \bigcap \mathcal{V}_{k'} = \emptyset$.



Graph partitioning

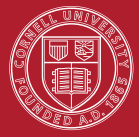


- We can consider the general case of weighted networks w_{uv}
- Let the complement of a subset $\mathcal{A}$ be denoted by $\overline{\mathcal{A}} = \mathcal{V}/\mathcal{A}$.
- All the measures of a *good partition* use the following definitions:

$$W(\mathcal{A}, \mathcal{B}) = \sum_{u \in \mathcal{A}} \sum_{v \in \mathcal{B}} w_{uv}$$

- When we partition the graph we want minimize the number (or weight) of the edges going outside $\mathcal{V}_k$, i.e. $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ as small as possible, but also have $\mathcal{V}_k$ with similar sizes.

Graph partitioning

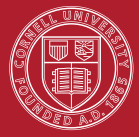


- We can consider the general case of weighted networks w_{uv}
- Let the complement of a subset $\mathcal{A}$ be denoted by $\overline{\mathcal{A}} = \mathcal{V}/\mathcal{A}$.
- All the measures of a *good partition* use the following definitions:

$$W(\mathcal{A}, \mathcal{B}) = \sum_{u \in \mathcal{A}} \sum_{v \in \mathcal{B}} w_{uv}$$

- When we partition the graph we want minimize the number (or weight) of the edges going outside $\mathcal{V}_k$, i.e. $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ as small as possible, but also have $\mathcal{V}_k$ with similar sizes.

Graph partitioning

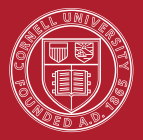


- We can consider the general case of weighted networks w_{uv}
- Let the complement of a subset $\mathcal{A}$ be denoted by $\overline{\mathcal{A}} = \mathcal{V}/\mathcal{A}$.
- All the measures of a *good partition* use the following definitions:

$$W(\mathcal{A}, \mathcal{B}) = \sum_{u \in \mathcal{A}} \sum_{v \in \mathcal{B}} w_{uv}$$

- When we partition the graph we want minimize the number (or weight) of the edges going outside $\mathcal{V}_k$, i.e. $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ as small as possible, but also have $\mathcal{V}_k$ with similar sizes.

Partitions and the Laplacian



Two facts allow us to write $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ in terms of the Laplacian $\boldsymbol{L}$:

1. The degree d_u of a node $u \in \mathcal{V}_k$ is the sum of the edges $(u, v), v \in \mathcal{V}_k$ and those with $v \in \overline{\mathcal{V}}_k$, i.e.

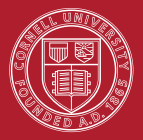
$$d_u = \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} + \sum_{v \in \overline{\mathcal{V}}_k} w_{u,v} \ ,$$

2. Recall that $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$. The diagonal elements of $\boldsymbol{L}$ are the degrees d_u and the off-diagonal are $-w_{uv}$.

$$W(\mathcal{V}_k, \overline{\mathcal{V}}_k) = \sum_{u \in \mathcal{V}_k} \sum_{v \in \overline{\mathcal{V}}_k} w_{uv} \overbrace{=}^{(1)} \sum_{u \in \mathcal{V}_k} \left(d_u - \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} \right)$$

$$\overbrace{=}^{(2)} \sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv}$$

Partitions and the Laplacian



Two facts allow us to write $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ in terms of the Laplacian $\boldsymbol{L}$:

1. The degree d_u of a node $u \in \mathcal{V}_k$ is the sum of the edges $(u, v), v \in \mathcal{V}_k$ and those with $v \in \overline{\mathcal{V}}_k$, i.e.

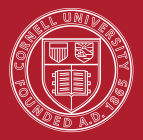
$$d_u = \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} + \sum_{v \in \overline{\mathcal{V}}_k} w_{u,v} \ ,$$

2. Recall that $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$. The diagonal elements of $\boldsymbol{L}$ are the degrees d_u and the off-diagonal are $-w_{uv}$.

$$W(\mathcal{V}_k, \overline{\mathcal{V}}_k) = \sum_{u \in \mathcal{V}_k} \sum_{v \in \overline{\mathcal{V}}_k} w_{uv} \overbrace{=}^{(1)} \sum_{u \in \mathcal{V}_k} \left(d_u - \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} \right)$$

$$\overbrace{=}^{(2)} \sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv}$$

Partitions and the Laplacian



Two facts allow us to write $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ in terms of the Laplacian $\boldsymbol{L}$:

1. The degree d_u of a node $u \in \mathcal{V}_k$ is the sum of the edges $(u, v), v \in \mathcal{V}_k$ and those with $v \in \overline{\mathcal{V}}_k$, i.e.

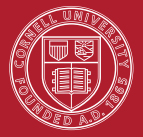
$$d_u = \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} + \sum_{v \in \overline{\mathcal{V}}_k} w_{u,v} \ ,$$

2. Recall that $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$. The diagonal elements of $\boldsymbol{L}$ are the degrees d_u and the off-diagonal are $-w_{uv}$.

$$W(\mathcal{V}_k, \overline{\mathcal{V}}_k) = \sum_{u \in \mathcal{V}_k} \sum_{v \in \overline{\mathcal{V}}_k} w_{uv} \overbrace{=}^{(1)} \sum_{u \in \mathcal{V}_k} \left(d_u - \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} \right)$$

$$\overbrace{=}^{(2)} \sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv}$$

Partitions and the Laplacian



Two facts allow us to write $W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$ in terms of the Laplacian $\boldsymbol{L}$:

1. The degree d_u of a node $u \in \mathcal{V}_k$ is the sum of the edges $(u, v), v \in \mathcal{V}_k$ and those with $v \in \overline{\mathcal{V}}_k$, i.e.

$$d_u = \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} + \sum_{v \in \overline{\mathcal{V}}_k} w_{u,v} \ ,$$

2. Recall that $\boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$. The diagonal elements of $\boldsymbol{L}$ are the degrees d_u and the off-diagonal are $-w_{uv}$.

$$W(\mathcal{V}_k, \overline{\mathcal{V}}_k) = \sum_{u \in \mathcal{V}_k} \sum_{v \in \overline{\mathcal{V}}_k} w_{uv} \overbrace{=}^{(1)} \sum_{u \in \mathcal{V}_k} \left(d_u - \sum_{v \in \mathcal{V}_k \setminus \{u\}} w_{u,v} \right)$$

$$\overbrace{=}^{(2)} \sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv}$$

Balanced graph clustering

Balanced graph clustering times at identifying communities by *minimizing* the following metric:

$$\text{RatioCut}(\mathcal{V}_1, ..., \mathcal{V}_K) := \sum_{k=1}^{K} \frac{1}{|\mathcal{V}_k|} W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$$

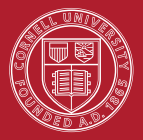
- We can define the following vectors that are associated with a partition $\mathcal{V}_1, ..., \mathcal{V}_K$:

$$\boldsymbol{p}_k \quad \text{such that} \quad [\boldsymbol{p}_k]_u = \begin{cases} \frac{1}{\sqrt{|\mathcal{V}_k|}} & u \in \mathcal{V}_k \\ 0 & \text{else} \end{cases} \qquad k = 1, \dots, K$$

- Then, we can write

$$W(\mathcal{V}_k, \overline{\mathcal{V}}_k) = \sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv} = |\mathcal{V}_k| \boldsymbol{p}_k^\top \boldsymbol{L} \boldsymbol{p}_k$$

Balanced graph clustering



Balanced graph clustering times at identifying communities by *minimizing* the following metric:

$$\text{RatioCut}(\mathcal{V}_1, ..., \mathcal{V}_K) := \sum_{k=1}^{K} \frac{1}{|\mathcal{V}_k|} W(\mathcal{V}_k, \overline{\mathcal{V}}_k)$$

- We can define the following vectors that are associated with a partition $\mathcal{V}_1, ..., \mathcal{V}_K$:

$$\boldsymbol{p}_k \quad \text{such that} \quad [\boldsymbol{p}_k]_u = \begin{cases} \frac{1}{\sqrt{|\mathcal{V}_k|}} & u \in \mathcal{V}_k \\ 0 & \text{else} \end{cases} \qquad k = 1, \dots, K$$

- Then, we can write

$$W(\mathcal{V}_k, \overline{\mathcal{V}}_k) = \sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv} = |\mathcal{V}_k| \boldsymbol{p}_k^\top \boldsymbol{L} \boldsymbol{p}_k$$

Balanced graph clustering

- Note that $\|\boldsymbol{p}_k\|^2 = \boldsymbol{p}_k^\top \boldsymbol{p}_k = 1$, and for $m \neq k$, $\boldsymbol{p}_k^\top \boldsymbol{p}_m = \mathbf{0}$.
- Introducing the matrix $\boldsymbol{P} := (\boldsymbol{p}_1, \dots, \boldsymbol{p}_K)$ we note that

$$\boldsymbol{p}_k^\top \boldsymbol{p}_m = \begin{cases} 1 & k = m \\ 0 & \text{else} \end{cases} \quad \Rightarrow \quad \boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}$$

- In addition we can write the ratio cut as follows:

$$\begin{aligned} \text{RatioCut}(\mathcal{V}_1, \dots, \mathcal{V}_K) &= \sum_{k=1} \frac{1}{|\mathcal{V}_k|} \overbrace{\sum_{u \in \mathcal{V}_k} \sum_{v \in \mathcal{V}_k} [\boldsymbol{L}]_{uv}}^{|\mathcal{V}_k| \boldsymbol{p}_k^\top \boldsymbol{L} \boldsymbol{p}_k} \\ &= \sum_{k=1}^{K} \boldsymbol{p}_k^\top \boldsymbol{L} \boldsymbol{p}_k = \text{Tr}(\boldsymbol{P}^\top \boldsymbol{L} \boldsymbol{P}) \end{aligned}$$

  where $\text{Tr}(\boldsymbol{A}) := \sum_{v=1} A_{vv}$ is the trace of matrix $\boldsymbol{A}$
- Note that $\text{Tr}(\boldsymbol{A}) = \text{Tr}(\boldsymbol{U}_{\boldsymbol{A}} \boldsymbol{\Lambda}_{\boldsymbol{A}} \boldsymbol{U}_{\boldsymbol{A}}^\top) = \text{Tr}(\boldsymbol{\Lambda}_{\boldsymbol{A}}) \equiv \sum_{v=1} \lambda_v(\boldsymbol{A})$, where $\lambda_v(\boldsymbol{A})$ are the eigenvalues of $\boldsymbol{A}$.

Minimum Ratio Cut criterion

Minimum Ratio Cut clustering

The optimum partition using the RatioCut metric

$$\min_{\mathcal{V}_1,\ldots,\mathcal{V}_K} \text{RatioCut}(\mathcal{V}_1,\ldots,\mathcal{V}_K)$$

can be formulated in terms of the Laplacian as follows:

$$\min_{\boldsymbol{P}} \text{Tr}(\boldsymbol{P}^\top \boldsymbol{L}\boldsymbol{P}), \quad \text{subject to} \quad \boldsymbol{P}^\top\boldsymbol{P} = \boldsymbol{I}, \quad \boldsymbol{P} \in \mathcal{P}$$

where $\mathcal{P}$ is the set of $N \times K$ matrices $\boldsymbol{P}$ whose entries are $[\boldsymbol{P}]_{uk} = p_{uk} \in \{0, \alpha_k\}$, $\alpha_k^2 \|\boldsymbol{p}_k\|_0 = 1$

Unnormalized spectral clustering: min RatioCut

- The min RatioCut problem is a **combinatorial problem** because of the this constraint expressed by $\boldsymbol{P} \in \mathcal{P}$ that requires searching for the right support of each vector $\boldsymbol{p}_k$, $k = 1, \ldots, K$
- Consider a constrained optimization:

$$\min_{\boldsymbol{x}} C(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{x} \in \mathcal{X}$$

- **Relaxation**: Relaxing the optimization problem means loosening either completely or in part the constraint $\boldsymbol{x} \in \mathcal{X}$ by finding an easier constraint to deal with.

Unnormalized spectral clustering: min RatioCut

- The min RatioCut problem is a **combinatorial problem** because of the this constraint expressed by $\boldsymbol{P} \in \mathcal{P}$ that requires searching for the right support of each vector $\boldsymbol{p}_k$, $k = 1, \ldots, K$
- Consider a constrained optimization:

$$\min_{\boldsymbol{x}} C(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{x} \in \mathcal{X}$$

- **Relaxation**: Relaxing the optimization problem means loosening either completely or in part the constraint $\boldsymbol{x} \in \mathcal{X}$ by finding an easier constraint to deal with.

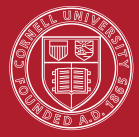
Unnormalized spectral clustering: min RatioCut

- The min RatioCut problem is a **combinatorial problem** because of the this constraint expressed by $\boldsymbol{P} \in \mathcal{P}$ that requires searching for the right support of each vector $\boldsymbol{p}_k$, $k = 1, \ldots, K$
- Consider a constrained optimization:

$$\min_{\boldsymbol{x}} C(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{x} \in \mathcal{X}$$

- **Relaxation**: Relaxing the optimization problem means loosening either completely or in part the constraint $\boldsymbol{x} \in \mathcal{X}$ by finding an easier constraint to deal with.

Relaxation



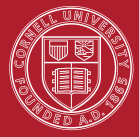
- More formally, the relaxed set $\mathcal{X}'$ is such that $\mathcal{X} \subset \mathcal{X}'$.

$$\min_{\boldsymbol{x}} C(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{x} \in \mathcal{X}'$$

- If $\mathcal{X}' \equiv \mathbb{R}^N$ the constraint is completely relaxed, i.e. becomes unconstrained.
- It holds that the minimum objective of the relaxed problem has always a minimum cost that is no larger than the minimum cost of the original problem.
- So if $\boldsymbol{x}_r$ is the solution of the relaxed problem and $\boldsymbol{x}_o$ that of the original problem can achieve a lower cost (it includes all previous possible values and more)

$$\min_{\boldsymbol{x} \in \mathcal{X}'} C(\boldsymbol{x}) \leq \min_{\boldsymbol{x} \in \mathcal{X}} C(\boldsymbol{x}).$$

Relaxation



- More formally, the relaxed set $\mathcal{X}'$ is such that $\mathcal{X} \subset \mathcal{X}'$.

$$\min_{\boldsymbol{x}} C(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{x} \in \mathcal{X}'$$

- If $\mathcal{X}' \equiv \mathbb{R}^N$ the constraint is completely relaxed, i.e. becomes unconstrained.
- It holds that the minimum objective of the relaxed problem has always a minimum cost that is no larger than the minimum cost of the original problem.
- So if $\boldsymbol{x}_r$ is the solution of the relaxed problem and $\boldsymbol{x}_o$ that of the original problem can achieve a lower cost (it includes all previous possible values and more)

$$\min_{\boldsymbol{x} \in \mathcal{X}'} C(\boldsymbol{x}) \leq \min_{\boldsymbol{x} \in \mathcal{X}} C(\boldsymbol{x}).$$

Min RatioCut relaxation

- The spectral clustering algorithm comes from relaxing the constraint $\boldsymbol{P} \in \mathcal{P}$. In other words, by solving instead

$$\min_{\boldsymbol{P}} \mathrm{Tr}(\boldsymbol{P}^\top \boldsymbol{L}\boldsymbol{P}), \quad \text{subject to} \quad \boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}.$$

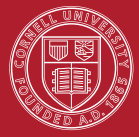
  where $\boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}$ means that $\boldsymbol{P}$ is simply orthonormal, does not have to be necessarily a diagonally scaled permutation matrix.

- This problem is also non convex, because it of the orthogonality constraint, but has a solution:

$$\boldsymbol{P}_{\text{opt}} = \boldsymbol{U}_{\text{min}}$$

  where the columns of $\boldsymbol{U}_{\text{min}}$ are the K least significant eigenvectors of $\boldsymbol{L}$.

Min RatioCut relaxation



- The spectral clustering algorithm comes from relaxing the constraint $\boldsymbol{P} \in \mathcal{P}$. In other words, by solving instead

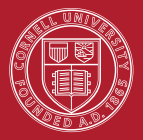
$$\min_{\boldsymbol{P}} \mathrm{Tr}(\boldsymbol{P}^\top \boldsymbol{L} \boldsymbol{P}), \quad \text{subject to} \quad \boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}.$$

  where $\boldsymbol{P}^\top \boldsymbol{P} = \boldsymbol{I}$ means that $\boldsymbol{P}$ is simply orthonormal, does not have to be necessarily a diagonally scaled permutation matrix.
- This problem is also non convex, because it of the orthogonality constraint, but has a solution:

$$\boldsymbol{P}_{\text{opt}} = \boldsymbol{U}_{\text{min}}$$

  where the columns of $\boldsymbol{U}_{\min}$ are the K least significant eigenvectors of $\boldsymbol{L}$.

Minimum RatioCut bound



- With $\boldsymbol{P}_{\text{opt}} = \boldsymbol{U}_{\text{min}}$ the minimum objective becomes

$$\min_{\boldsymbol{P}} \text{Tr}(\boldsymbol{U}_{\text{min}}^{\top} \boldsymbol{L} \boldsymbol{U}_{\text{min}}) = \text{Tr}(\boldsymbol{\Lambda}_{\text{min}})$$

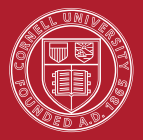
  where $\boldsymbol{\Lambda}_{\text{min}}$ is the diagonal matrix of $\lambda_{N-K} \geq \cdots \geq \lambda_{N-1} \geq \lambda_N = 0$ are the smallest K eigenvalues of the Laplacian matrix $\boldsymbol{L}$.

- Since the relaxation results in a minimum that is no larger than original problem (i.e.$\min_{x \in \mathcal{X}'} C(\boldsymbol{x}) \leq \min_{x \in \mathcal{X}} C(\boldsymbol{x})$.):

$$\min_{\mathcal{V}_1, \ldots, \mathcal{V}_K} \text{RatioCut}(\mathcal{V}_1, \ldots, \mathcal{V}_K) \geq \sum_{k=1}^{K-1} \lambda_{N-k}$$

- Now, how do we go from $\boldsymbol{U}_{\text{min}}$ to the partition $(\mathcal{V}_1, \ldots, \mathcal{V}_K)$?

Minimum RatioCut bound



- With $\boldsymbol{P}_{\text{opt}} = \boldsymbol{U}_{\text{min}}$ the minimum objective becomes

$$\min_{\boldsymbol{P}} \text{Tr}(\boldsymbol{U}_{\text{min}}^{\top} \boldsymbol{L} \boldsymbol{U}_{\text{min}}) = \text{Tr}(\boldsymbol{\Lambda}_{\text{min}})$$

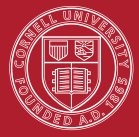
  where $\boldsymbol{\Lambda}_{\text{min}}$ is the diagonal matrix of $\lambda_{N-K} \geq \cdots \geq \lambda_{N-1} \geq \lambda_N = 0$ are the smallest K eigenvalues of the Laplacian matrix $\boldsymbol{L}$.

- Since the relaxation results in a minimum that is no larger than original problem (i.e.$\min_{\boldsymbol{x} \in \mathcal{X}'} C(\boldsymbol{x}) \leq \min_{\boldsymbol{x} \in \mathcal{X}} C(\boldsymbol{x})$.):

$$\min_{\mathcal{V}_1, \ldots, \mathcal{V}_K} \text{RatioCut}(\mathcal{V}_1, \ldots, \mathcal{V}_K) \geq \sum_{k=1}^{K-1} \lambda_{N-k}$$

- Now, how do we go from $\boldsymbol{U}_{\text{min}}$ to the partition $(\mathcal{V}_1, \ldots, \mathcal{V}_K)$?

Minimum RatioCut bound



- With $\boldsymbol{P}_{\text{opt}} = \boldsymbol{U}_{\text{min}}$ the minimum objective becomes

$$\min_{\boldsymbol{P}} \text{Tr}(\boldsymbol{U}_{\text{min}}^{\top} \boldsymbol{L} \boldsymbol{U}_{\text{min}}) = \text{Tr}(\boldsymbol{\Lambda}_{\text{min}})$$

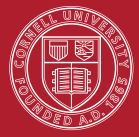
  where $\boldsymbol{\Lambda}_{\text{min}}$ is the diagonal matrix of $\lambda_{N-K} \geq \cdots \geq \lambda_{N-1} \geq \lambda_N = 0$ are the smallest K eigenvalues of the Laplacian matrix $\boldsymbol{L}$.

- Since the relaxation results in a minimum that is no larger than original problem (i.e.$\min_{\boldsymbol{x} \in \mathcal{X}'} C(\boldsymbol{x}) \leq \min_{\boldsymbol{x} \in \mathcal{X}} C(\boldsymbol{x})$.):

$$\min_{\mathcal{V}_1, \ldots, \mathcal{V}_K} \text{RatioCut}(\mathcal{V}_1, \ldots, \mathcal{V}_K) \geq \sum_{k=1}^{K-1} \lambda_{N-k}$$

- Now, how do we go from $\boldsymbol{U}_{\text{min}}$ to the partition $(\mathcal{V}_1, \ldots, \mathcal{V}_K)$?

Unnormalized spectral clustering

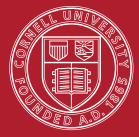


- The most common approach is to use the $K-$ **means** algorithm to the **rows** of $\boldsymbol{U}_{\text{min}}$, to now cluster the entries of the vectors.
- Why on earth would that work?
- First we need to understand what $K-$means does

K-means

K-means partition N observations into K clusters in which each observation belongs to the cluster with the nearest mean.

Unnormalized spectral clustering



- The most common approach is to use the $K-$ **means** algorithm to the **rows** of $\boldsymbol{U}_{\text{min}}$, to now cluster the entries of the vectors.
- Why on earth would that work?
- First we need to understand what $K-$means does

K-means

K-means partition N observations into K clusters in which each observation belongs to the cluster with the nearest mean.

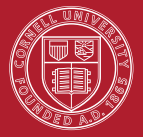
Unnormalized spectral clustering

- The most common approach is to use the $K-$ **means** algorithm to the **rows** of $\boldsymbol{U}_{\text{min}}$, to now cluster the entries of the vectors.
- Why on earth would that work?
- First we need to understand what $K-$means does

K-means

K-means partition N observations into K clusters in which each observation belongs to the cluster with the nearest mean.

K-means algorithm

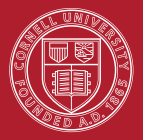


- Given a set of points $\mathcal{X} = \boldsymbol{x}_1, \dots$ $K-$means seeks a partition of $\mathcal{X}$, in K disjoint subsets of points $\{\mathcal{X}_1, \dots, \mathcal{X}_K\}$ that targets the following objective:

$$\min_{\{\mathcal{X}_1,\dots,\mathcal{X}_N\}} \sum_{k=1}^{K} \sum_{i:\boldsymbol{x}_i \in \mathcal{X}_i} \|\boldsymbol{x}_i - \boldsymbol{\mu}_i\|^2, \quad \text{where} \quad \boldsymbol{\mu}_i = \frac{1}{|\mathcal{X}_i|} \sum_{i:\boldsymbol{x}_i \in \mathcal{X}_i} \boldsymbol{x}_i$$

where $\boldsymbol{\mu}_i$ are called the centroids.

K-means algorithm on $\boldsymbol{U}_{\min}$



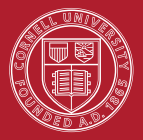
- Let us denote the rows of $\boldsymbol{U}_{\min}$ as follows

$$\boldsymbol{U}_{\min} = \begin{bmatrix} \boldsymbol{u}_1^{\text{row}} \\ \vdots \\ \boldsymbol{u}_N^{\text{row}} \end{bmatrix}$$

  we just apply K-means to $\boldsymbol{u}_n^{\text{row}}, r = 1, \ldots, N$ as the set of points to cluster.

- The if $\boldsymbol{u}_n^{\text{row}}$, $n = 1, \ldots, N$ is in cluster k then node n is in the graph cluster $\mathcal{V}_k$

K-means algorithm on $\boldsymbol{U}_{\min}$



- Let us denote the rows of $\boldsymbol{U}_{\min}$ as follows

$$\boldsymbol{U}_{\min} = \begin{bmatrix} \boldsymbol{u}_1^{\text{row}} \\ \vdots \\ \boldsymbol{u}_N^{\text{row}} \end{bmatrix}$$

  we just apply K-means to $\boldsymbol{u}_n^{\text{row}}, r = 1, \ldots, N$ as the set of points to cluster.

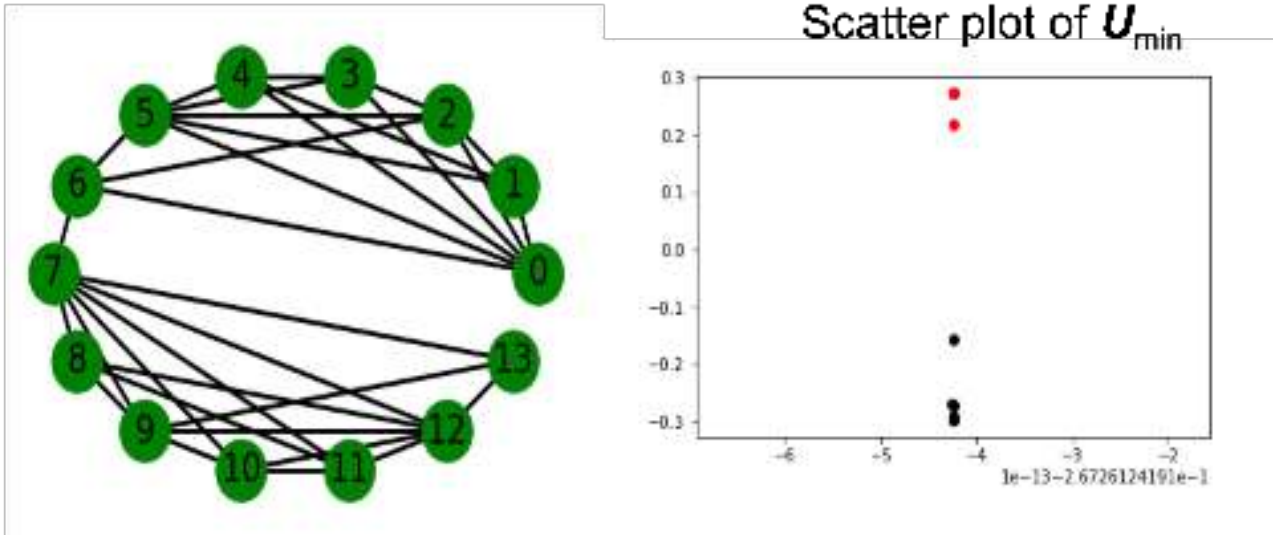
- The if $\boldsymbol{u}_n^{\text{row}}$, $n = 1, \ldots, N$ is in cluster k then node n is in the graph cluster $\mathcal{V}_k$

The intuition

- We already discussed this, we said that impact $\boldsymbol{L}$ can be represented as follows

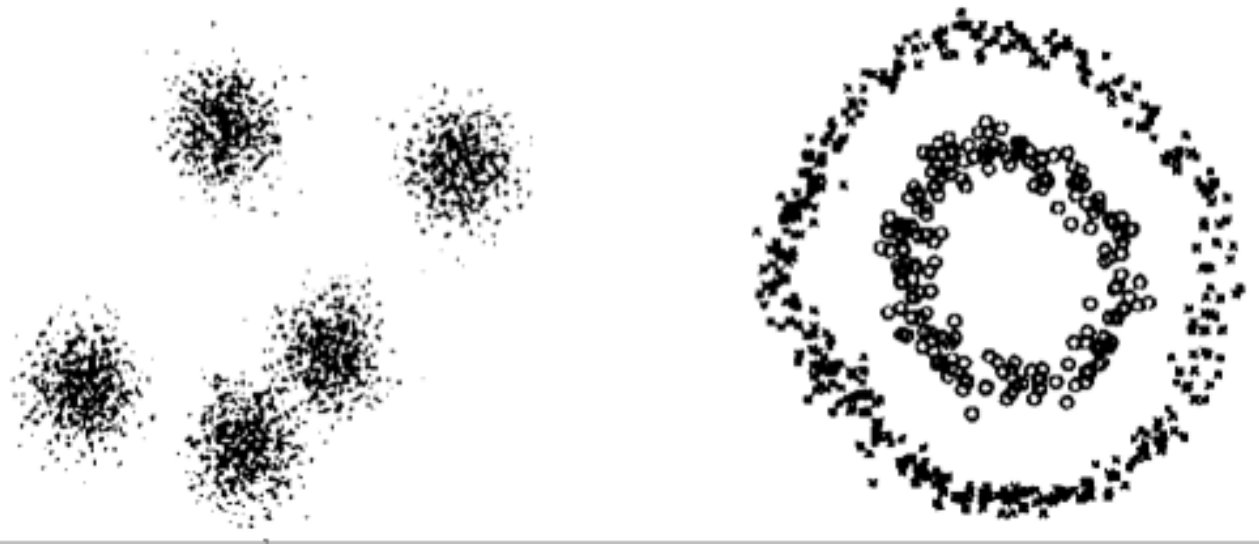
$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{L}_2 \end{bmatrix} + \delta\boldsymbol{L}, \qquad \|\delta\boldsymbol{L}\| \ll \|\boldsymbol{L}\|$$

- We already had marked in red and black the points that correspond to the differnet clusters and said that the same can be generalized for a graph that has $K > 2$ clusters.



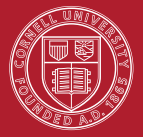
Normalized laplacian matrix of weighted graphs

- Normalized Laplacians are also used to understand the topological structure of data and identify if they can be partitioned.
- In both figures the points scattered are two dimensional real vectors $\boldsymbol{x}_i$ and clustering algorithms are *unsupervised* method to identify the indexes of these sets

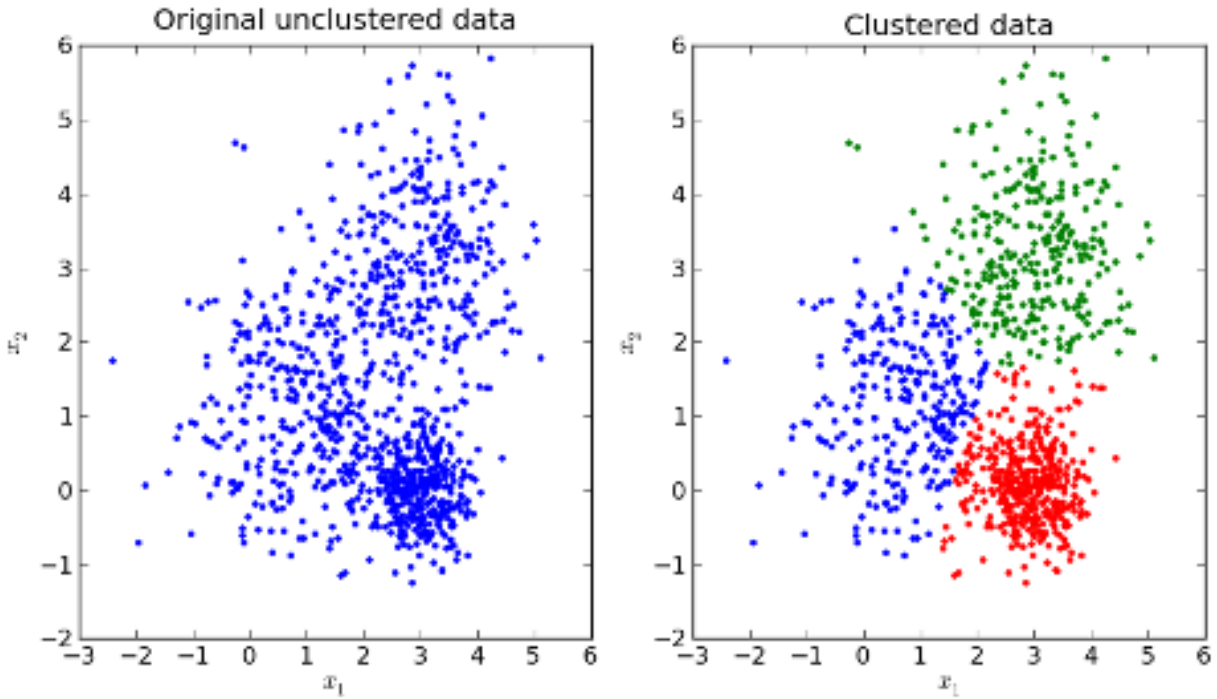


- Spectral clustering is much more effective for data scattered in groups around non convex shapes, like the ones on the right figure.

K-Means vs. Spectral Clustering

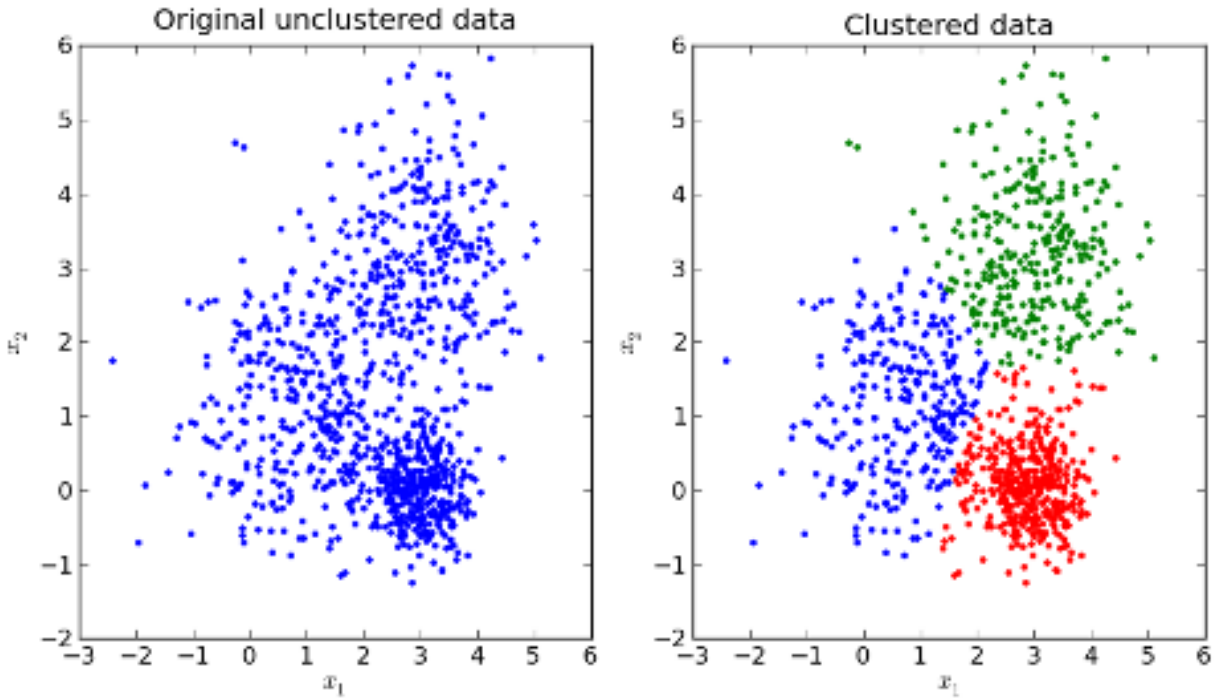


- The left case happens to be a mixture of Gaussian: You have a cathegorical distribution $P(k) : \sum_{k=1}^{K} P(k) = 1$ and with probabilty $P(k)$ you draw a sample out of $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{R}_k)$.
- For the points obtained, you can apply K-means directly to the points $\boldsymbol{x}_i$



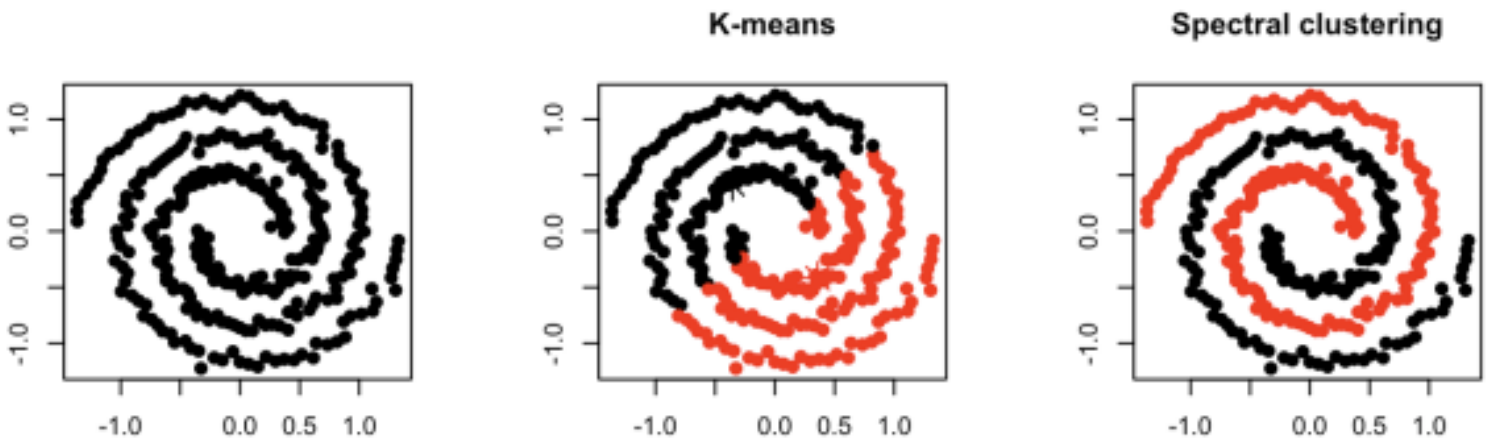
K-Means vs. Spectral Clustering

- The left case happens to be a mixture of Gaussian: You have a cathegorical distribution $P(k) : \sum_{k=1}^{K} P(k) = 1$ and with probabilty $P(k)$ you draw a sample out of $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{R}_k)$.
- For the points obtained, you can apply K-means directly to the points $\boldsymbol{x}_i$



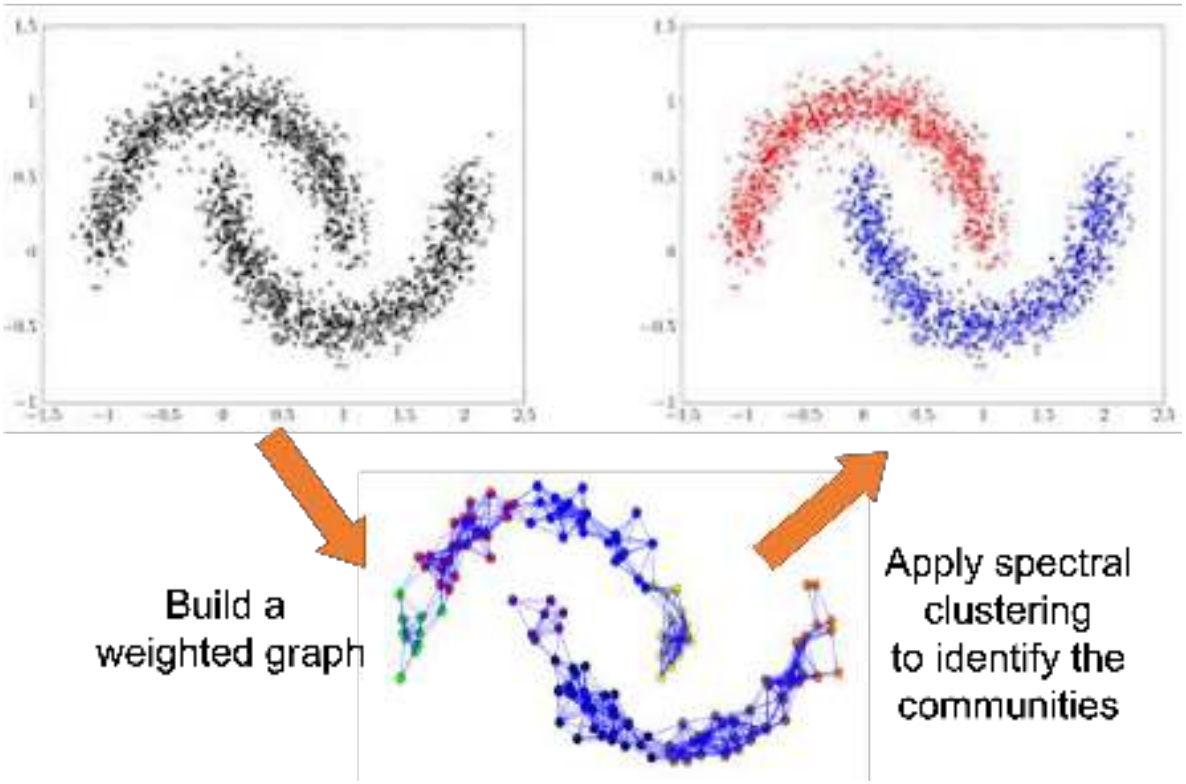
K-Means vs. Spectral Clustering

- If you apply K-means directly to the points $\boldsymbol{x}_i$ for the other case you do not separate correctly the data



Normalized laplacian matrix of weighted graphs

- What is done is to first define weights that rapidly monotonically *decrease* with the data points distance, denoting weak affinity. You threshold those that are too small.
- Non convex clusters of points will form tight nit communities



Normalized laplacian matrix of weighted graphs

- Let the data points be $\boldsymbol{x}_i, i = 1, \ldots, n$, and $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ a measure of their distance, affinity matrices (e.g. for points $\boldsymbol{x}_i \in \mathbb{R}^D$ we can use the Frobenious norm $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$). One option often used is the so called Gaussian kernel:

$$[\boldsymbol{A}]_{ij} = w_{ij} = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2}} \quad \Rightarrow \quad \boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$$

- The weights w_{ij} measure of *affinity* and can be used to define a weighted adjacency matrix.
- Instead of applying K-means to the points $\boldsymbol{x}_i$ spectral clustering can be applied to separate these points, by applying K-means to the K least significant eigenvectors $\boldsymbol{U}_{\min}$ of this Laplacian.

Normalized laplacian matrix of weighted graphs

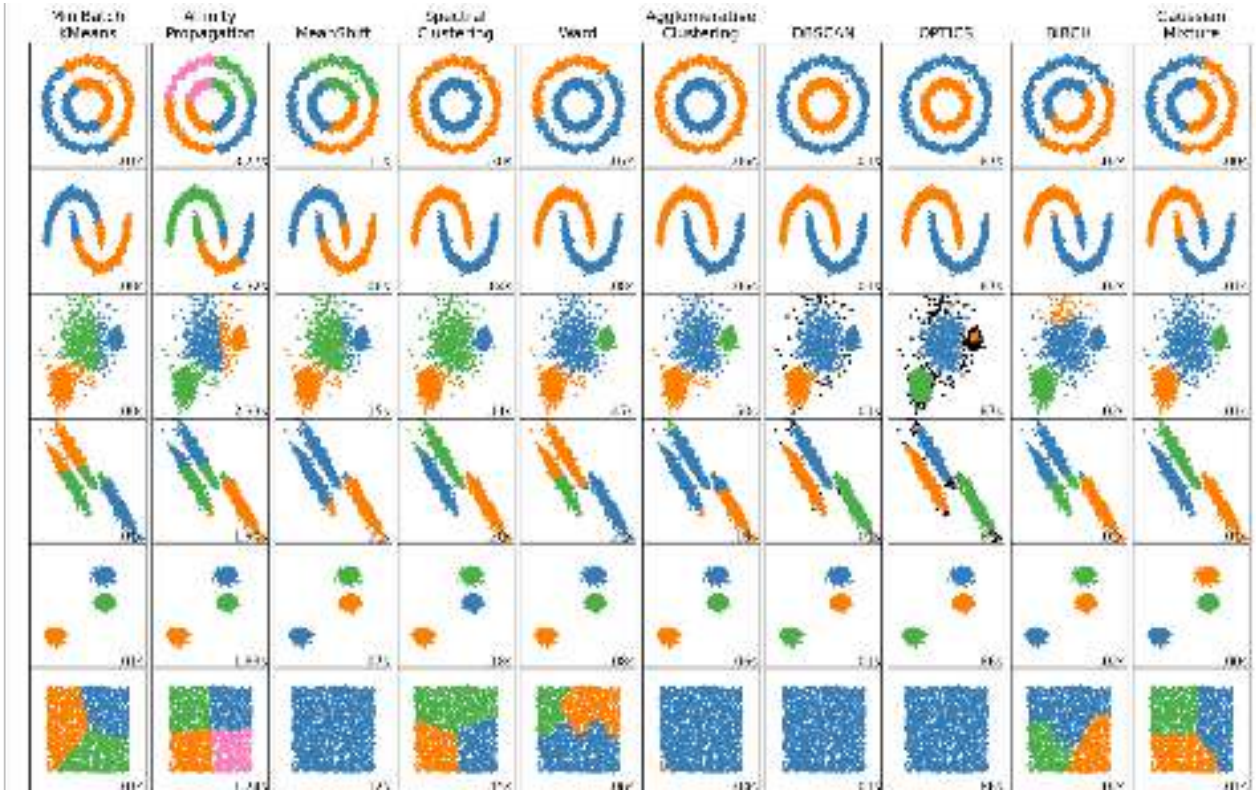
- Let the data points be $\boldsymbol{x}_i, i = 1, \ldots, n$, and $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ a measure of their distance, affinity matrices (e.g. for points $\boldsymbol{x}_i \in \mathbb{R}^D$ we can use the Frobenious norm $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$). One option often used is the so called Gaussian kernel:

$$[\boldsymbol{A}]_{ij} = w_{ij} = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2}} \quad \Rightarrow \quad \boldsymbol{L} = \text{diag}(\boldsymbol{d}) - \boldsymbol{A}$$

- The weights w_{ij} measure of *affinity* and can be used to define a weighted adjacency matrix.
- Instead of applying K-means to the points $\boldsymbol{x}_i$ spectral clustering can be applied to separate these points, by applying K-means to the K least significant eigenvectors $\boldsymbol{U}_{\min}$ of this Laplacian.

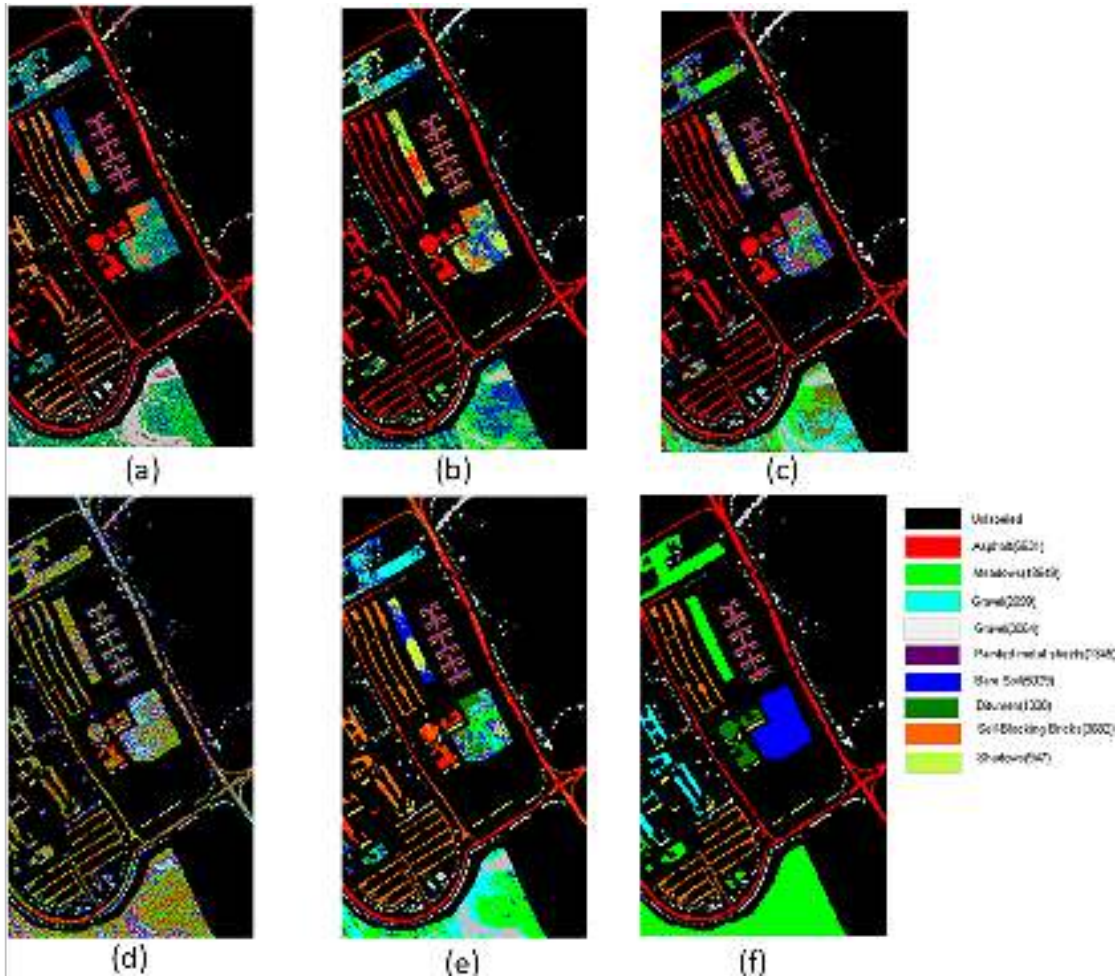
Clustering methods

- Of course K-means is also a clustering method and there are many variations, but spectral clustering is very popular and effective.



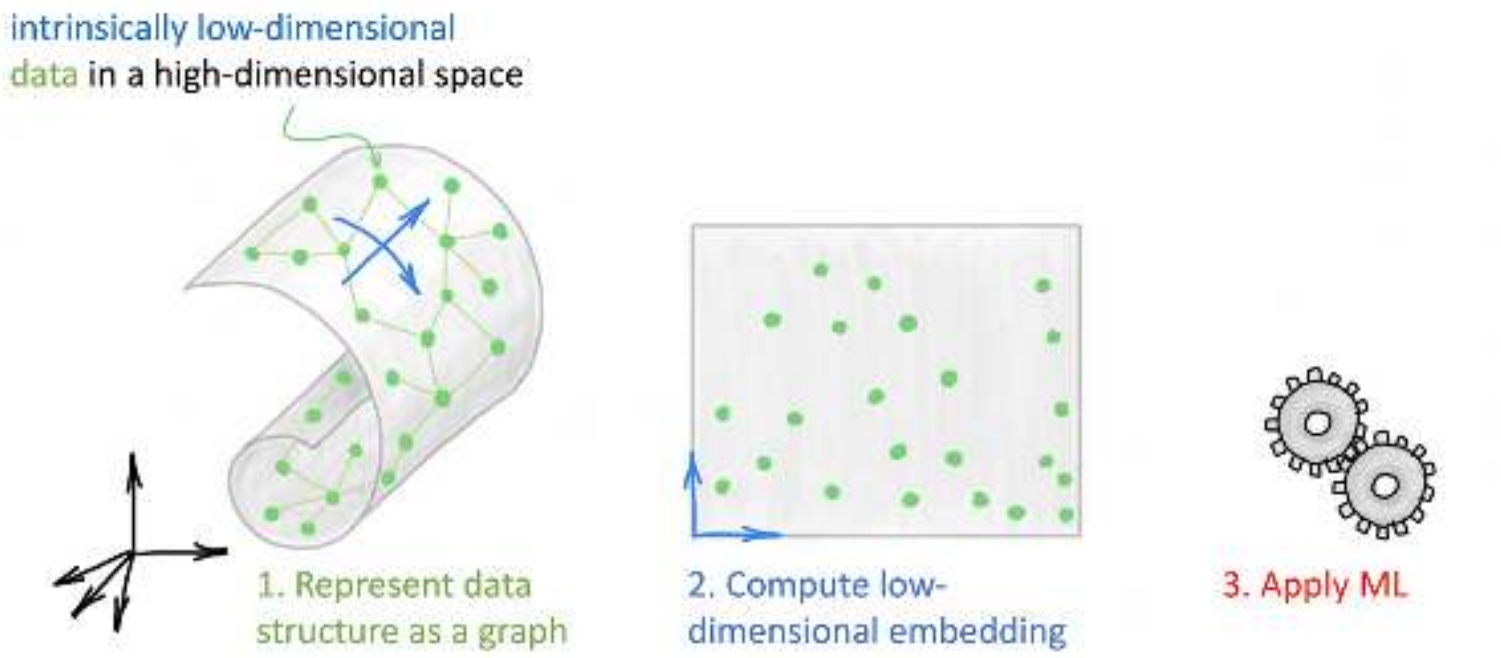
Applications

- There are many interesting applications



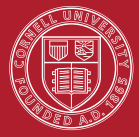
Applications of spectral clustering

- For many data, not necessarily 2-D, that are expected to have a low dimensional structure that is non convex, spectral clustering is the go-to method for - unsupervised classification



- In some cases, one seeks to map them in a lower dimensions first through what is called an embedding

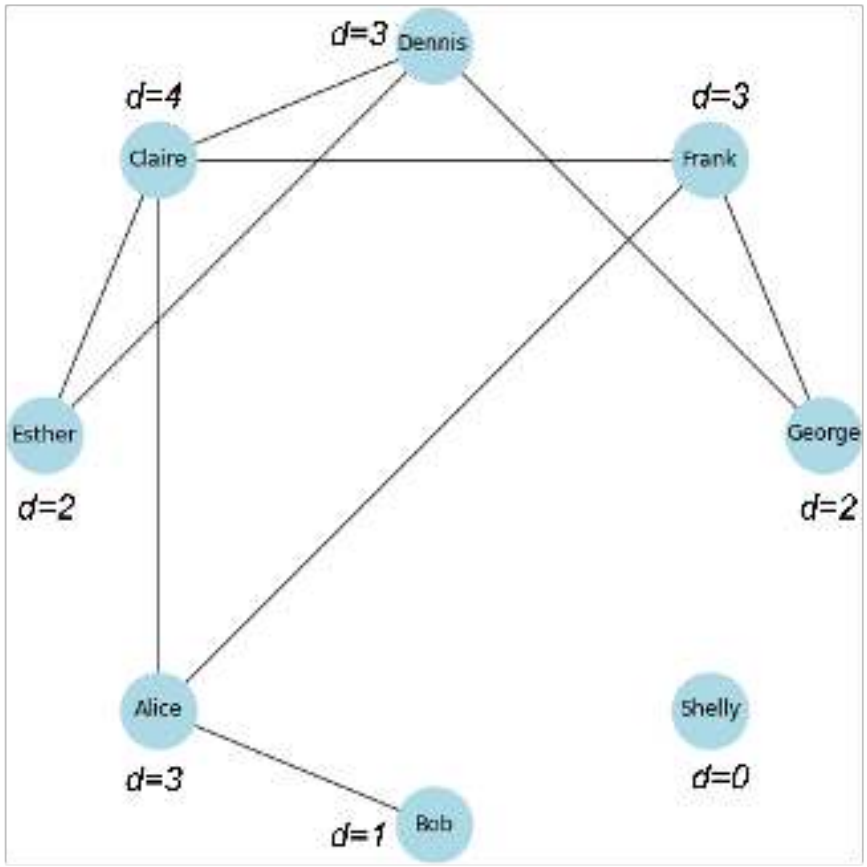
Outline



Cornell University

Nodal degree

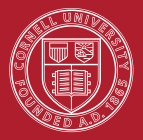
Feature extraction



- We already defined the degree $d_v = |\mathcal{N}_u|$ is the number of neighbors of a node. The degree sequence is a feature of the graph.
- For a directed graph we have in degree and out degrees d_v^+ and d_v^-, which provide for each node a 2D feature component.
- For simplicity we will focus on undirected graphs.

Degree distribution

Feature extraction



- For the whole graph the empirical distribution of the nodal degree is a feature vector: it is the fraction of nodes that has a given nodal degree. Mathematically:

$$p_k = \hat{P}(d_v = k) = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \delta(d_v - k), \quad k = 1, \dots, \bar{d}$$

where $\bar{d} = \max_{u \in \mathcal{V}}(d_u)$ and $\delta(k)$ is the Kronecker delta, i.e the indicator function defined as:

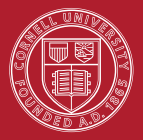
$$\delta(k) := \begin{cases} 1 & \text{for } k = 0 \\ 0 & \text{else} \end{cases}$$

- Note that the average degree:

$$d_{\text{av}} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} d_v = \sum_{k=1}^{\bar{d}} k p_k$$

Degree distribution

Feature extraction



- For the whole graph the empirical distribution of the nodal degree is a feature vector: it is the fraction of nodes that has a given nodal degree. Mathematically:

$$p_k = \hat{P}(d_v = k) = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \delta(d_v - k), \quad k = 1, \ldots, \bar{d}$$

where $\bar{d} = \max_{u \in \mathcal{V}}(d_u)$ and $\delta(k)$ is the Kronecker delta, i.e the indicator function defined as:

$$\delta(k) := \begin{cases} 1 & \text{for } k = 0 \\ 0 & \text{else} \end{cases}$$

- Note that the average degree:

$$d_{\text{av}} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} d_v = \sum_{k=1}^{\bar{d}} k p_k$$

Node centrality measures

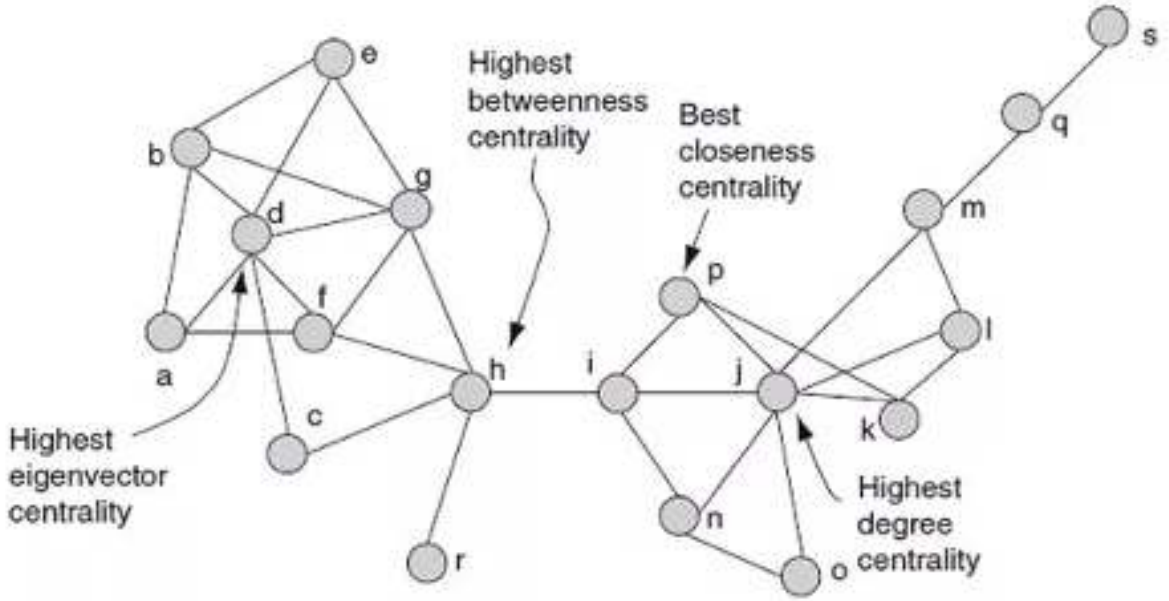
graph analysis

Cornell University

Centrality

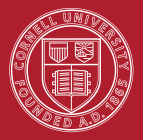
This measure is used to establish the **most important node in the network**

- They find different nodes, since the form of *importance* is different and has a functional meaning



Degree centrality

graph analysis



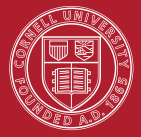
- Degree centrality is one measure. The central node is the one that has maximum degree, i.e.

$$v^{\text{deg}} = \text{argmax}_v \, d_v.$$

- Is very simple but often very indicative of importance in social networks and of congestion in intrastructures

Degree centrality

graph analysis



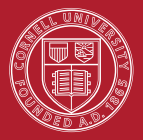
- Degree centrality is one measure. The central node is the one that has maximum degree, i.e.

$$v^{\text{deg}} = \text{arg}\max_{v} d_v.$$

- Is very simple but often very indicative of importance in social networks and of congestion in intrastructures

Eigenvector centrality

graph analysis



- Another popular measure is called Eigenvector Centrality that is defined with a recurrence equation, which captures the fact that an central node must have **central neighbors**:

$$z_u \propto \sum_{v \in \mathcal{N}_u} z_v \quad u \in \mathcal{V} \quad \Rightarrow \quad \lambda \boldsymbol{z} = \boldsymbol{A}\boldsymbol{z}$$

- Which eigenvector would we use, given that the score is proportional to λ and there are N of them?

Eigenvector centrality

graph analysis

- Another popular measure is called Eigenvector Centrality that is defined with a recurrence equation, which captures the fact that an central node must have **central neighbors**:

$$z_u \propto \sum_{v \in \mathcal{N}_u} z_v \quad u \in \mathcal{V} \quad \Rightarrow \quad \lambda \boldsymbol{z} = \boldsymbol{A}\boldsymbol{z}$$

- Which eigenvector would we use, given that the score is proportional to λ and there are N of them?

Eigenvector centrality

graph analysis

- Note that $\boldsymbol{A}$ has non negative entries: Perron Frobenious Theorem guarantees that the **maximum eigenvector has positive entries.**
- Hence the eigenvector centrality of a node a node is the corresponding entry of the leading eigenvector, which will assign a positive score to every node
- Let $\boldsymbol{z}_1$ be the leading eigenvector, i.e. $\lambda_1(\boldsymbol{A})\boldsymbol{z}_1 = \boldsymbol{A}\boldsymbol{z}_1$. The most central node is[2]:

$$v^{\text{eig}} = \arg\max_v [\boldsymbol{z}_1]_u$$

- Again, for disconnected networks is more meaningful to have a central node per component.

[2] Any scaling of the leading eigenvector will give the same result.

Eigenvector centrality

graph analysis

- Note that $\boldsymbol{A}$ has non negative entries: Perron Frobenious Theorem guarantees that the **maximum eigenvector has positive entries.**
- Hence the eigenvector centrality of a node a node is the corresponding entry of the leading eigenvector, which will assign a positive score to every node
- Let $\boldsymbol{z}_1$ be the leading eigenvector, i.e. $\lambda_1(\boldsymbol{A})\boldsymbol{z}_1 = \boldsymbol{A}\boldsymbol{z}_1$. The most central node is[2]:

$$v^{\text{eig}} = \text{argmax}_v [\boldsymbol{z}_1]_u$$

- Again, for disconnected networks is more meaningful to have a central node per component.

[2] Any scaling of the leading eigenvector will give the same result.

Eigenvector centrality

graph analysis

- Note that $\boldsymbol{A}$ has non negative entries: Perron Frobenious Theorem guarantees that the **maximum eigenvector has positive entries.**
- Hence the eigenvector centrality of a node a node is the corresponding entry of the leading eigenvector, which will assign a positive score to every node
- Let $\boldsymbol{z}_1$ be the leading eigenvector, i.e. $\lambda_1(\boldsymbol{A})\boldsymbol{z}_1 = \boldsymbol{A}\boldsymbol{z}_1$. The most central node is[2]:

$$v^{\text{eig}} = \text{arg}\max_v [\boldsymbol{z}_1]_u$$

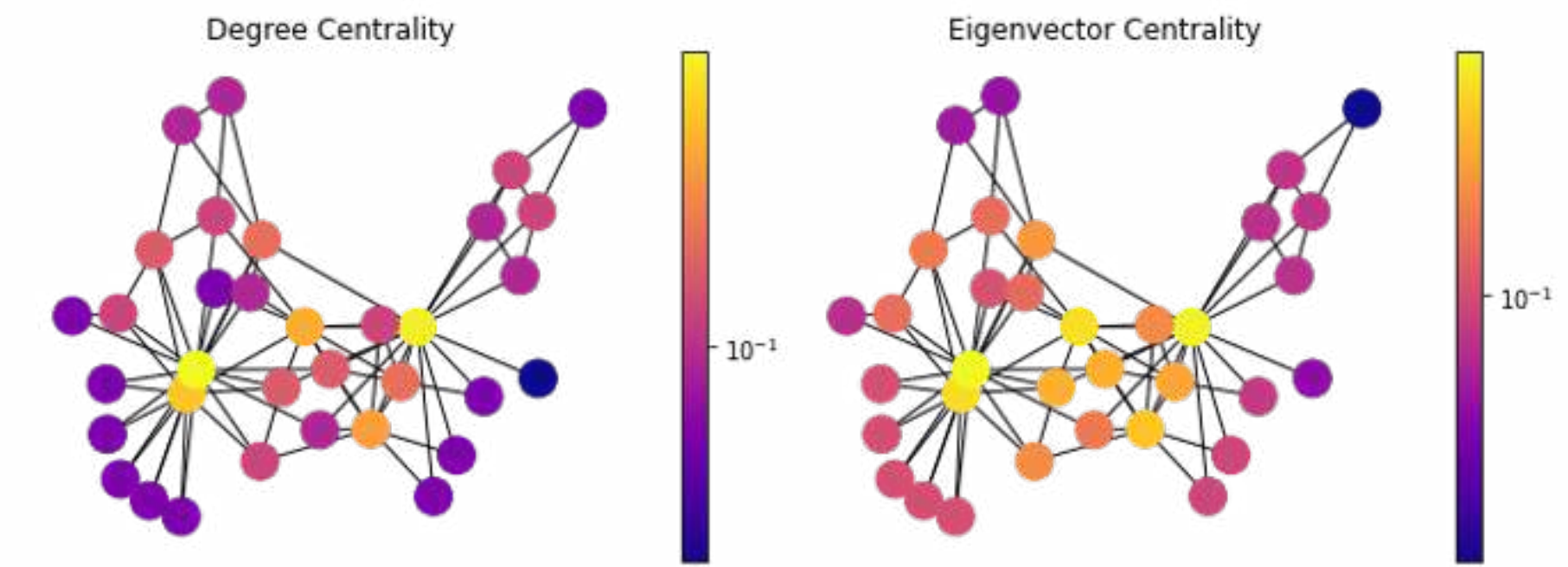
- Again, for disconnected networks is more meaningful to have a central node per component.

[2] Any scaling of the leading eigenvector will give the same result.

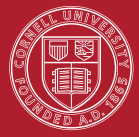
Eigenvector centrality example

graph analysis

- The example of the Zachary's Karate Club for both measures illustrates similarties and diferences between the two metrics

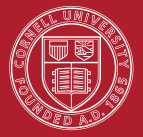


Outline



Cornell University

Conclusions



- We studied in detail the problem of graph partitioning and saw how the Laplacian plays a central role in it
- We introduced some metrics to characterize them and compare nodes.
- In the next lecture we will continue to refine the notion of centrality, and introduce the general notion of graph embedding and introduce other graph features that are used to identify structural characteristics.