Networks and Their Spectra

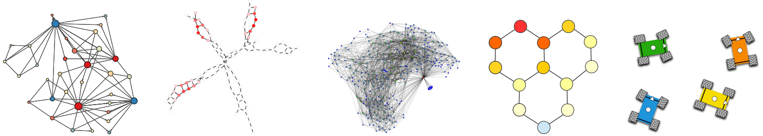
Victor Amelkin

University of California, Santa Barbara
Department of Computer Science
victor@cs.ucsb.edu

December 4, 2017

Introduction

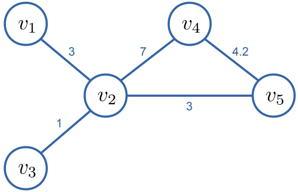
- Networks (= graphs) are everywhere.



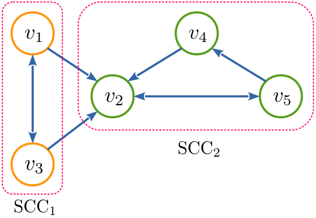
(a) Social network (b) Traffic network (c) Brain network (d) Material network (e) Robotic network

- Networks ≈ matrices.
- **Network's spectrum**—eigenvalues-eigenvectors of the network's "matrix".
- Network's matrices' spectra reveal lots of information about the network.
- **This talk's focus**—applications to walk analysis and spectral clustering.

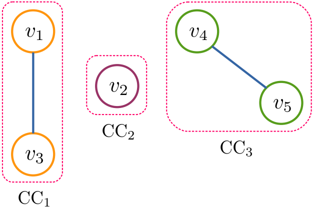
Undirected,
(weakly) connected



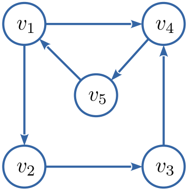
Directed,
(weakly) connected,
strongly disconnected



Undirected,
(weakly) disconnected



Directed,
strongly connected



Graphs – Standard Notation

- Graph $G = \langle V, E, w \rangle$, with $|V| = n$ nodes, and $|E| = m$ edges.
- **Adjacency matrix $\boldsymbol{A}$**:

$$A_{ij} = \begin{cases} w(\langle i, j \rangle) = w_{ij} & \text{if } \langle i, j \rangle \in E, \\ 0 & \text{otherwise.} \end{cases}$$

- **In-degree** d_i^{in} of node v_i is the sum of the weights on all of its incoming edges (similarly, **out-degree**).
- **Degree matrix:** $\boldsymbol{D} = diag(\{d_i\}_{1 \ldots n})$ (similarly, **in- and out-degree**).
- **Laplacian:** $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$.
- **Random-walk Laplacian:** $\boldsymbol{L}^{rw} = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{A}$.

Spectral Graph Theory

Graphs are usually represented with matrices. Spectral graph theory attempts to connect spectral properties of these matrices with the corresponding graphs' structural properties.

Limitations

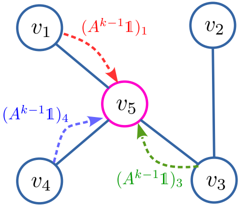
Most spectral graph theory's results are obtained for **undirected** graphs.

Spectrum of Adjacency Matrix – Walks in Graphs (I)

- $\boldsymbol{A} \in \{0,1\}^{n \times n}$ – adjacency matrix of an undirected unweighted graph G.
- A_{ij} – number of walks of length 1 in G between nodes v_i and v_j.

Spectrum of Adjacency Matrix – Walks in Graphs (I)

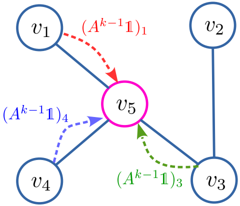
- $\boldsymbol{A} \in \{0,1\}^{n \times n}$ – adjacency matrix of an undirected unweighted graph G.
- A_{ij} – number of walks of length 1 in G between nodes v_i and v_j.
- $(\boldsymbol{A}^k)_{ij}$ – number of walks of length k in G between nodes v_i and v_j.
- $(\boldsymbol{A}^k \mathbb{1})_i$ – number of walks of length k ending at v_i.



$$\underset{\mathbf{A}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix}} \underset{\mathbf{A}^{k-1}\mathbb{1}}{\begin{bmatrix} (A^{k-1}\mathbb{1})_1 \\ (A^{k-1}\mathbb{1})_2 \\ (A^{k-1}\mathbb{1})_3 \\ (A^{k-1}\mathbb{1})_4 \\ (A^{k-1}\mathbb{1})_5 \end{bmatrix}} = \underset{\mathbf{A}^k\mathbb{1}}{\begin{bmatrix} \dots \\ \dots \\ \dots \\ \dots \\ (A^{k-1}\mathbb{1})_1 + (A^{k-1}\mathbb{1})_3 + (A^{k-1}\mathbb{1})_4 \end{bmatrix}}$$

Spectrum of Adjacency Matrix – Walks in Graphs (I)

- $\boldsymbol{A} \in \{0,1\}^{n\times n}$ – adjacency matrix of an undirected unweighted graph G.
- A_{ij} – number of walks of length 1 in G between nodes v_i and v_j.
- $(\boldsymbol{A}^k)_{ij}$ – number of walks of length k in G between nodes v_i and v_j.
- $(\boldsymbol{A}^k\mathbb{1})_i$ – number of walks of length k ending at v_i.



$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} (A^{k-1}\mathbb{1})_1 \\ (A^{k-1}\mathbb{1})_2 \\ (A^{k-1}\mathbb{1})_3 \\ (A^{k-1}\mathbb{1})_4 \\ (A^{k-1}\mathbb{1})_5 \end{bmatrix}}_{\mathbf{A}^{k-1}\mathbb{1}} = \underbrace{\begin{bmatrix} \cdots \\ \cdots \\ \cdots \\ \cdots \\ (A^{k-1}\mathbb{1})_1 + (A^{k-1}\mathbb{1})_3 + (A^{k-1}\mathbb{1})_4 \end{bmatrix}}_{\mathbf{A}^k\mathbb{1}}$$

- $\mathbb{1}^\top \boldsymbol{A}^k\mathbb{1}$ – number of walks of length k in G.

- Connection to the largest eigenvalue μ_{max} of the adjacency matrix $\boldsymbol{A}$ of an undirected, unweighted, connected graph:

$$\mathbb{1}^\mathsf{T} \boldsymbol{A}^k \mathbb{1} = (\text{since } \boldsymbol{A} \text{ is real and symmetric}) = \mathbb{1}^\mathsf{T} (\boldsymbol{Q}\, \mathrm{diag}\,(\mu_i) \boldsymbol{Q}^\mathsf{T})^k \mathbb{1} =$$

$$= (\text{since } \boldsymbol{Q} \text{ is orthogonal}) = \mathbb{1}^\mathsf{T} \boldsymbol{Q}\, \mathrm{diag}\,(\mu_i^k) \boldsymbol{Q}^\mathsf{T} \mathbb{1} =$$

$$= \mathbb{1}^\mathsf{T} \left(\sum_{i=1}^{n} \mu_i^k \boldsymbol{q}_i \boldsymbol{q}_i^\mathsf{T} \right) \mathbb{1} = \sum_{i=1}^{n} \mu_i^k (\mathbb{1}^\mathsf{T} \boldsymbol{q}_i)(\mathbb{1}^\mathsf{T} \boldsymbol{q}_i)^\mathsf{T} = \sum_{i=1}^{n} \mu_i^k \langle \boldsymbol{q}_i, \mathbb{1} \rangle^2 =$$

$$= (\mathbb{1} = \alpha_1 \boldsymbol{q}_1 + \cdots + \alpha_n \boldsymbol{q}_n) = \sum_{i=1}^{n} \mu_i^k \left(\sum_{j=1}^{n} \alpha_j \langle \boldsymbol{q}_i, \boldsymbol{q}_j \rangle \right)^2 = \sum_{i=1}^{n} \mu_i^k (\alpha_i)^2;$$

$$\lim_{k\to\infty} \left(\mathbb{1}^\mathsf{T} \boldsymbol{A}^k \mathbb{1} \right)^{1/k} = \lim_{k\to\infty} \left(\sum_{i=1}^{n} \mu_i^k \alpha_i^2 \right)^{1/k} =$$

$$= \lim_{k\to\infty} \mu_{max} \left(\alpha_{max}^2 + \sum_{\text{“}i \neq max\text{''}} \left(\frac{\mu_i}{\mu_{max}} \right)^k \alpha_i^2 \right)^{1/k} = \mu_{max}.$$

- Thus, $\mathbb{1}^\mathsf{T} \boldsymbol{A}^k \mathbb{1} \sim \mu_{max}^k$, is, roughly, the number of walks of length k in G.

More results ((†) – applies to weighted graphs):

- (†) If graph G is connected, μ_{max} has multiplicity 1, and its eigenvector is positive (Perron-Frobenius).
- (†) $d_{avg} \leq \mu_{max} \leq d_{max}$ (d_{avg}, d_{max} – mean, max degrees).
- $\max\{d_{avg}, \sqrt{d_{max}}\} \leq \mu_{max} \leq d_{max}$.
- (†) If G is connected, and $\mu_{max} = d_{max}$, then $\forall i : d_i = d_{max}$.
- (†) A connected graph is bipartite iff $\mu_{min} = -\mu_{max}$.
- $\chi(G) \geq 1 + \mu_{min}/\mu_{max}$.

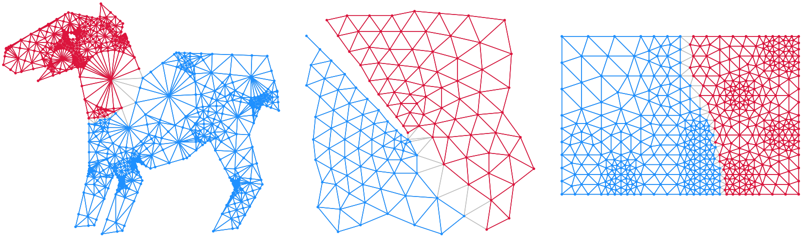
Spectral Graph Theory for Clustering

- **Clustering goal:** partition a network into k clusters so that the nodes being "close to each other" end up inside the clusters, and the nodes that are "far apart" belong to different clusters.
- **Select basic methods:** k-means, k-medoids, density-based clustering.
- **Our method:** cluster nodes based on the partial spectrum of Laplacian.

Spectral Bisection – Theory

- Goal: cut an undirected graph with $G(A)$ into $S \subset V$ and $\bar{S} = V \setminus S$.
- $\text{cut}(S, \bar{S}) = \sum_{i \in S, j \in \bar{S}} A_{ij}$.
- $\text{ratio-cut}(S, \bar{S}) = \frac{\text{cut}(S, \bar{S})}{|S||\bar{S}|}$.
- $\boldsymbol{x}^\mathsf{T} \boldsymbol{L} \boldsymbol{x} = \sum_{ij} A_{ij}(x_i - x_j)^2/2$, where $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$.
- Assume signed cluster indicator: $x_i = 1$ if $i \in S$; and $x_i = -1$ otherwise.
- $\text{ratio-cut}(S, \bar{S}) \to \min \sim \boldsymbol{x}^\mathsf{T} L \boldsymbol{x} \to \min$, with $\boldsymbol{x} \perp \mathbb{1}$ – NP-hard.
- Relax x_i to be real $\Rightarrow \boldsymbol{x}^* = \boldsymbol{q}_2$ – Fiedler vector.
- Thus, $\boldsymbol{q}_2 > \mathbb{0}$ – cluster indicator.
- If $\boldsymbol{L}^{rw} = D^{-1} L$ is used instead of $\boldsymbol{L}$, then we look for edge-balanced cut.

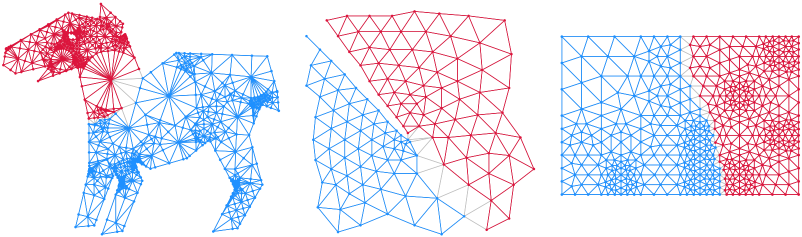
- Spectral bisection (splitting Fiedler vector around 0):



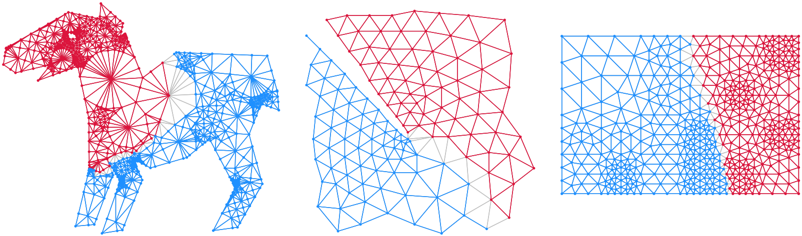
(Networks from `meshpart` MATLAB toolbox by John R. Gilbert and Shang-Hua Teng.)

Spectral Bisection – Practice

- Spectral bisection (splitting Fiedler vector around 0):

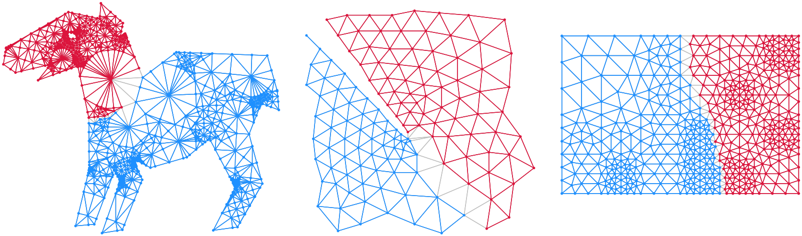


- Enforcing strict node-balance (splitting Fiedler vector around median):

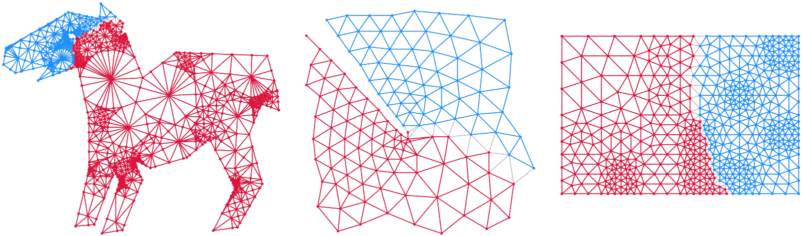


Spectral Bisection vs. K-Means

- Spectral bisection:

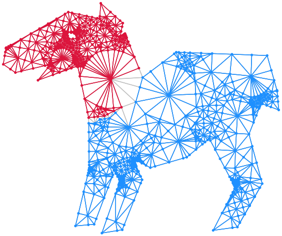


- K-Means:

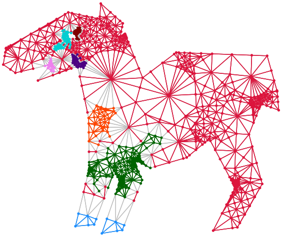


Spectral Bisection vs. HDBSCAN

- Spectral bisection:

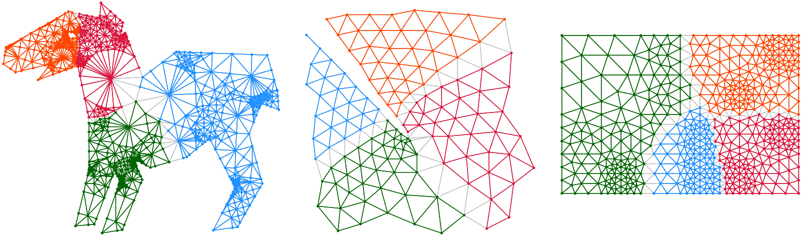


- HDBSCAN:



- Why not to use spectral bisection hierarchically, to partition a graph into an arbitrary number of clusters?

- Why not to use spectral bisection hierarchically, to partition a graph into an arbitrary number of clusters?



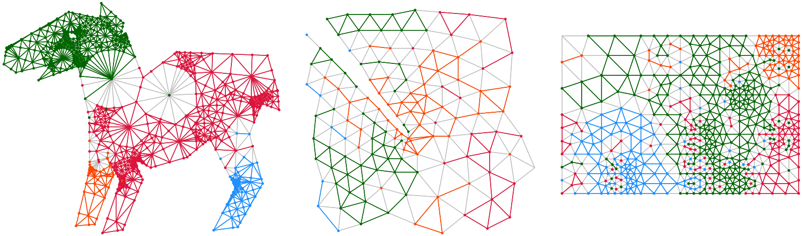
- Will see how it compares to “proper spectral clustering” in what follows.

Spectral Clustering – Theory

Conceptually similar to spectral bisection, with a few differences.

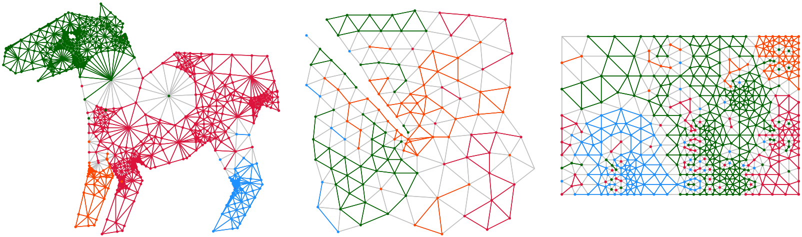
- Performing a k-way instead of a 2-way cut.
- Instead of using only $\boldsymbol{q}_2$, using $(k-1)$ smallest eigenvectors $\boldsymbol{Q}_{*,2:k}$.
- $\boldsymbol{Q}_{*,2:k}$ – discriminative $(k-1)$-dimensional embedding of the graph.
- Cluster map obtained via "splitting" rows of $\boldsymbol{Q}_{*,1:k}$.

- Spectral clustering (4 clusters, combinatorial Laplacian $L = D - A$):

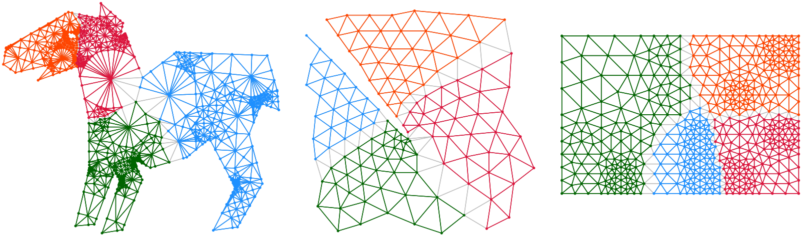


Spectral Clustering – Practice

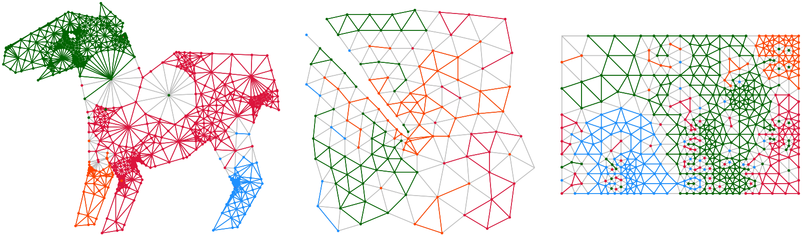
- Spectral clustering (4 clusters, combinatorial Laplacian $L = D - A$):



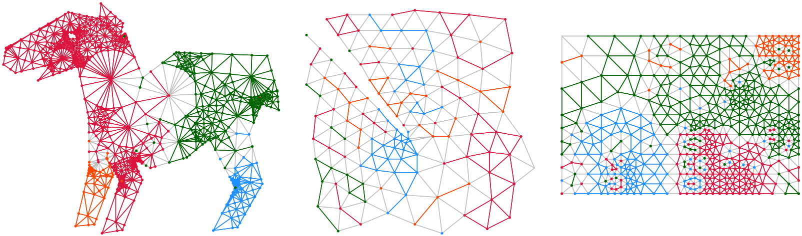
- Hierarchical spectral bisection:



- Spectral clustering (4 clusters, combinatorial Laplacian $L = D - A$):

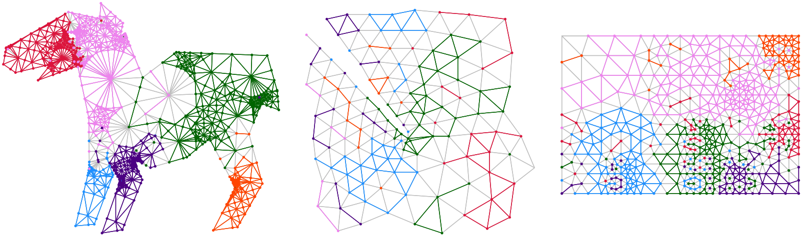


- Spectral clustering (4 clusters, random-walk Laplacian $L^{rw} = I - D^{-1}A$):

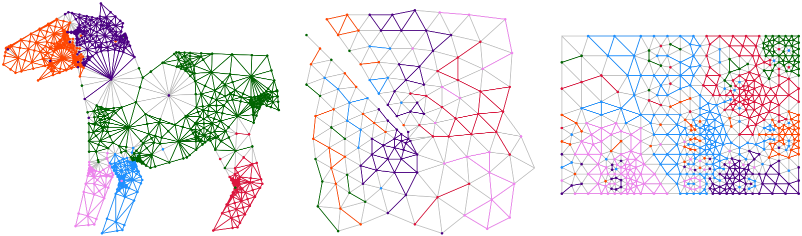


Spectral Clustering – Practice

- Spectral clustering (6 clusters, combinatorial Laplacian $L = D - A$):

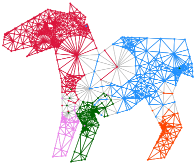


- Spectral clustering (6 clusters, random-walk Laplacian $L^{rw} = I - D^{-1}A$):



Summary

- Spectrum of a graph provides plenty of information about the graph.
- Spectral bisection is superior to k-means-like and density-based clustering[1].
- Hierarchical spectral bisection is hard to use for multi-way cutting.
- Spectral clustering is effective at discovering clusters of complex shape.
- L for node-balanced (ratio) cut; L^{rw} for edge-balanced (normalized) cut.
- Spectral clustering is relatively cheap for sparse networks.



[1] When we need "connectivity-based" rather than "geometric" clustering.

$\sim$ Thanks $\sim$