Master MVA **Scribed By: Edwige Cyffers, Antoine Yang**
Graphs in Machine Learning **Lecture #2**
Instructor: Dr. Michal Valko **October 8, 2019**

Outline *In this lecture we will mention several public graphs datasets that might be useful for class projects. Then we will introduce a central tool of the lecture, the Laplacian matrix. We will establish some properties and recap the linear algebra that comes with it. Then we will move on variants of this matrix with normalized Laplacian and applications to random walks. Finally, we will apply this tool to the problem of clustering with different approximation schemes.*

Contents

Slides available on Michal Valko's INRIA page
`http://researchers.lille.inria.fr/~valko/projects/courses/graphsML/20192020/mlgraphs2.pdf`

1 Available public graphs datasets

You can find diverse graph examples used in state-of-the-art applications at the following links :

- `http://snap.stanford.edu/data/` Stanford Large Network Dataset Collection have a collection of social networks with the characteristics of the graphs.
- `http://www-personal.umich.edu/~mejn/netdata/` On the personal webpage of Mark Newman in GML format and freely avalaible.
- `http://proj.ise.bgu.ac.il/sns/datasets.html` Creted by a reasearch group from Ben Gurion University specializing in social networks research.
- `https://sparse.tamu.edu/` The SuiteSparse Matrix Collection is a large and actively growing set of sparse matrices that arise in real applications. The Collection is widely used by the numerical linear algebra community for the development and performance evaluation of sparse matrix algorithms.
- `http://vlado.fmf.uni-lj.si/pub/networks/data/default.html`

2 Laplacian : definitions and first properties

2.1 Definitions

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. We have some well-known matrices that derive from the graph. The adjacency matrix $\mathbf{A}$ is defined such that the coefficient $A_{i,j}$ is 1 if there is an edge between the corresponding vertices and 0 otherwise. We also have the weigth matrix $\mathbf{W}$ where the coefficient $W_{i,j}$ corresponds to the weight of the edge (i, j) and the degree matrix $\mathbf{D}$ that is diagonal and stores for each node its degree.

From these matrices, we can define the matrix that we will study in this lecture.
Definition (Laplacian matrix)
The Graph Laplacian matrix $\mathbf{L}$ is defined by $\mathbf{L} = \mathbf{D} - \mathbf{W}$. This matrix has the degrees of the corresponding nodes on the diagonal and the others coefficients are the opposite of the weights.

Let's fix the idea with an example of a simple graph.

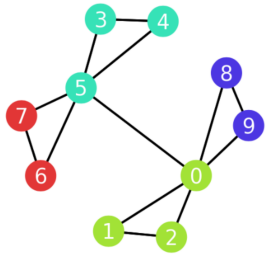

Figure 1: Example of graph

For this graph, the associated Laplacian will be the following matrix:

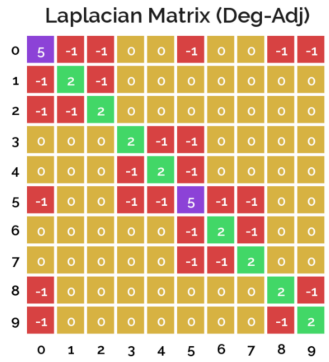

Figure 2: The graph Laplacian matrix. Other examples can be found on `https://dominikschmidt.xyz/spectral-clustering-exp/`

For instance, there is an edge between node 7 and 6, thus the coefficient is -1. The node 5 has five incoming edges, so the associated diagonal coefficient is 5.

This small example illustrates a first property of the Laplacian.
Lemma The graph Laplacian matrix is symmetric diagonal dominant.

Proof. Let's fix i, we have $|\mathbf{L}_{i,i}| = |d_i| = |-\sum_{j \neq i} W_{i,j}| = \sum_{j \neq i} |\mathbf{L}_{i,j}|$. □

This property is important because when a matrix is diagonal dominant, it is easier to invert it or to solve linear systems.

2.2 Eigenvalues and eigenvectors

This section reminds the reader some definitions of linear algebra.
Definition (Eigenvector, Eigenvalue)
A vector $v \neq 0$ is an eigenvector of matrix $\mathbf{M}$ of eigenvalue λ when $\mathbf{M}v = \lambda v$.

These eigenvectors verify several properties. First they are linearly independent, we have the stronger result:
Lemma If (λ_1, v_1) and (λ_2, v_2) are eigenpairs for symmetric M with $\lambda_1 \neq \lambda_2$ then $v_1^\top v_2 = 0$.

Proof. $\lambda_1 v_1^\top v_2 = v_1^\top M v_2 = v_1^\top \lambda_2 v_2$. □

The same eigenvalue can correspond to several independent eigenvectors, and we call *multiplicity* the dimension of the space of eigenvectors corresponding to it.

For a symmetric matrix, we have the Spectral Theorem.

Theorem (Spectral Theorem) A $N \times N$-symmetric matrix has N eigenvalues with multiplicity. It ensures the existence of an eigendecomposition of a symmetric matrix $\mathbf{M}$:

$$\mathbf{M} = \mathbf{M}\mathbf{Q}\mathbf{Q}^\top = \mathbf{Q}\mathbf{\Gamma}\mathbf{Q}^\top = \sum_i \lambda_i v_i v_i^\top$$

2.3 First properties

We can assume that all the weights are non-negative. Indeed we can suppose that the coefficients are non-negative : two objects represented by nodes with nothing in common will be assigned with edge weight 0.
Lemma L is symmetric.

Proof. Follows from the symmetry of similarity. If the similarity is not symmetric $w_{i,j} = \frac{1}{2}(s_{i,j} + s_{j,i})$. □

Lemma L is semi-definite.
Theorem The smallest eigenvalue of $\mathbf{L}$ is 0 and a corresponding eigenvector is $\mathbf{1}_N$.

Proof. $(\mathbf{L}1_N)_i = \sum_k \mathbf{L}_{k,i} = d_i - \sum_{k \neq i} w_{k,i} = 0$ □

Lemma All eigenvalues are non-negative reals $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$

Proof. This comes from the fact that $\mathbf{L}$ is definite positive. □

Lemma Self-edges do not change the value of $\mathbf{L}$.

Proof. The contribution to the degree matrix is cancelled by the contribution to the weight matrix. □

These properties will be reused for establishing results on graph functions.
Definition (Graph function)
A graph function is a vector $f \in \mathbb{R}^N$ assigning values to nodes:

$$\mathbf{f} : \mathcal{V}(\mathcal{G}) \to \mathbb{R}$$

An example of graph function is the coloring of a graph, the labels corresponds to the values of the nodes. Another good example is clustering : the assignation to the cluster is the graph function.
Definition (Smoothness)
For a graph function $\mathbf{f}$ and a Laplacian $\mathbf{L}$ that can be decomposed as $\mathbf{L} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$ where $\mathbf{Q}$ have for columns the eigenvectors of $\mathbf{L}$ and $\mathbf{\Lambda}$ is diagonal, we define the *smoothness* $S_G(\mathbf{f})$ by:

$$S_G(\mathbf{f}) = \mathbf{f}^\top \mathbf{L}\mathbf{f} = \mathbf{f}^\top \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top \mathbf{f} = \|\alpha\|_\Lambda = \sum_{i=1}^{N} \lambda_i \alpha_i^2$$

where $\alpha_i = \mathbf{Q}^\top \mathbf{f}$

Lemma $\mathbf{f}^\top \mathbf{L}\mathbf{f} = \frac{1}{2}\sum_{i,j} w_{i,j}(f_i - f_j)^2 = S_G(\mathbf{f})$.

We prove it by computation :

$\mathbf{f}^\top \mathbf{L}\mathbf{f} = \mathbf{f}^\top (D - W)\mathbf{f} = \mathbf{f}^\top D\mathbf{f} - \mathbf{f}^\top W\mathbf{f} = \sum_i d_i f_i^2 - \sum_{i,j} w_{i,j} f_i f_j$.

Thus, we have $\mathbf{f}^\top \mathbf{L}\mathbf{f} = \frac{1}{2}\left(\sum_i \sum_j w_{i,j} f_i^2 - 2\sum_{i,j} w_{i,j} f_i f_j + \sum_i \sum_j f_j^2\right)$.

With this formula, we can see what matters to minimize the function. The intuition is that when nodes are close from each other, they should have the same treatments by $\mathbf{f}$, for instance being assigned to the same cluster.

The smoothness can be rewritten according to eigendecomposition of the Laplacian.

If $\mathbf{f}$ is smooth, we expect a small value, which means small values for the biggest eigenvalues. Of course the eigenvalues are not chosen by us, but we chose the α.

Example : The smoothness of an eigenvector is the associate eigenvalue. This notion is convenient to prove an important property of the Laplacian:
Theorem The multiplicity of the eigenvalue 0 of $\mathbf{L}$ is equal to the number of connected components of the graph.

Proof. If $(0, \mathbf{f})$ is an eigenpair of $\mathbf{L}$, the previous formula gives

$$0 = \sum_{i,j} w_{i,j}(f_i - f_j)^2$$

which means that all the terms of the sum are equals to 0, so $w_{i,j} > 0$ implies $f_i = f_j$, which means that the function is constant on connected components. Let fix k the number of connected components. The previous constraints give a decomposition of $\mathbf{L}$ as a k-block-diagonal matrix. However, for block-diagonal matrices, the spectrum is the union of the spectra of the blocks. For each block, we have an eigenpairs given by:

$$(0, \mathbf{1}_{|V_i|}) \text{ where } V_i \text{ is the } i\text{-th connected component}$$

This proves that we have k eigenpairs of eigenvalue 0. □

Exemple : Laplacian of the complete graph K_N

For the complete graph, the Laplacian matrix is

$$\mathbf{L} = \begin{pmatrix} N-1 & -1 & \cdots & -1 \\ -1 & N-1 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & N-1 \end{pmatrix}$$

One eigenpair is $(0, (1, \dots, 1)^\top)$. It means that the other eigenvectors have to verify $(1, \dots, 1)v = 0$ which means that $\sum_i v_i = 0$. . By definition, an eigenvector v must verify

$$(Lv)_i = (N-1)v_i - \sum_{j \neq i}^{N} v_j = Nv_i - \sum_j v_j = Nv_i$$

Thus, the other eigenvalues are equal to N, and every vector of the hyperplan $\{x | \sum_i x_i = 0\}$ is an eigenvector.

3 Normalized Laplacians and random walks

The Laplacian graph matrix we introduced is one of the most popular representations. However, alternative representations are sometimes proposed to obtain better results [1] [2] under specific assumptions. Here we present two variants that are quite popular.

3.1 Normalized Laplacians

Definition (Normalized Laplacians)
We define symmetric and random walk laplacians as followed: $\mathbf{L}_{sym} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$, $\mathbf{L}_{rw} = \mathbf{D}^{-1}\mathbf{L}$.

We deduce from the previous properties on $\mathbf{L}$ the following properties on $\mathbf{L}_{sym}$ and $\mathbf{L}_{rw}$.

A first one is given by culaculation:
Lemma $\mathbf{f}^\top \mathbf{L}_{sym}\mathbf{f} = \frac{1}{2}\sum_{i,j \leq N} w_{i,j}(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}})^2$

Then we deduce several useful properties:
Lemma By using $\mathbf{D}$ for the degree matrix, we have the following properties:

- $(\lambda, \mathbf{u})$ is an eigenpair for $\mathbf{L}_{rw}$ if and only if $(\lambda, \mathbf{D}^{1/2}\mathbf{u})$ is an eigenpair for $\mathbf{L}_{sym}$.
- $\mathbf{L}_{sym}$ and $\mathbf{L}_{rw}$ are positive semi-definite.
- $(0, 1_N)$ is an eigenpair for $\mathbf{L}_{rw}$ and $(0, \mathbf{D}^{1/2}1_N)$ is an eigenpair for $\mathbf{L}_{sym}$.
- $(\lambda, \mathbf{u})$ is an eigenpair for $\mathbf{L}_{rw}$ if and only if $(\lambda, \mathbf{u})$ solves the generalized eigenproblem $\mathbf{L}\mathbf{u} = \lambda\mathbf{D}\mathbf{u}$.
- Multiplicity of eigenvalue 0 of $\mathbf{L}_{rw}$ or $\mathbf{L}_{sym}$ equals to the number of connected components.

Proof. All the properties can be established in the same way than for the Laplacian. □

3.2 Random Walks

We focus on the random walk where we jump from a vertex to one of its neighbors with probability proportional to the weight of the corresponding edge. The transition matrix is given by $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$. This process can appear for several applications. For instance, for the shuffling of a deck of cards,

we construct a graph whose nodes are all permutations of the deck, and two of them are adjacent if we obtain one from the other with one shuffle move. Then repeated shuffle move correspond to a random walk on this graph. Another application is the Brownian motion of a dust particle [3].

If G is connected and non-bipartite, there exists a unique stationary distribution $\pi = (\pi_1 \dots \pi_N)$ where $\pi_i = \frac{d_i}{vol(d)}$. Indeed we can check that this distribution verifies $\pi \mathbf{P} = \pi$.

Remark This setting is different from PageRank, because the graph is undirected. This makes the problem easier to solve.

4 Application to spectral clustering

4.1 Reminder about clustering

It is an unsupervised learning issue. It is often a rather ill-defined problem, despite a good capacity for human to recognize it in small dimensions : people recognize it when they see it.

Basically, there are three properties that we would like to have for a clustering algorithm :

Scale invariance The cluster should remain identical if we strech equally all the data.

Richness The clustering function should be able to take any value (in the absence of knowing the distance between points).

Consistency If we diminish the distance within a cluster, or increase it between clusters, the clustering should not change.

However, it has been proven that it is impossible to build such a clustering function is the general case[4].

The k-means algorithm remains a decent baseline, but suffers from a high sensitivity to the starting point and requires the numbers of clusters to be defined a priori. Furthermore, because it essentially optimizes for compactness, it cannot provide good answers for some datasets (e.g. circles).

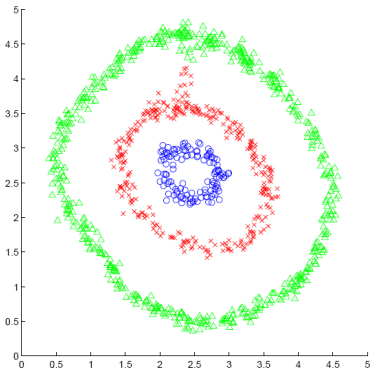

Figure 3: An example where k-means cannot provide a good solution: the clusters are not compact

4.2 From cut to clustering

For a graph, a clustering can be obtained thanks to cuts. In this situation, the clustering is a consequence of the cut objective.

Definition

We can define the minCut problem as

$$\min cut(A, B) = \sum_{i \in A, j \in B} w_{i,j}$$

This problem can be solved in polynomial time. Indeed, using the max-flow/min-cut theorem, we can solve the max flow associated problem. The Ford-Fulkerson algorithm or Dinic's algorithm provide polynomial time solutions to it.

However, finding these minimal cuts is not enough, because we are looking for rather balanced clusters. Hence we might get very small clusters using a mincut approach.

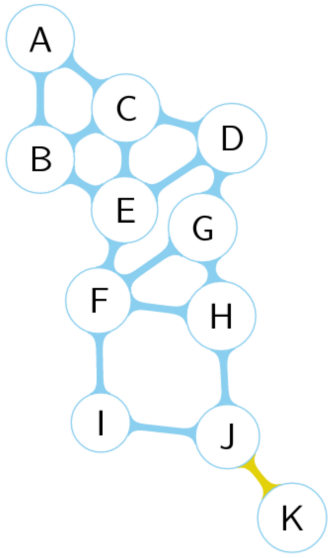

Figure 4: An example for which mincut does not provide a satisfying clustering solution

That is why we introduce two other objective functions that correct this behaviour.

Definition

We define the RatioCut objective function by

$$RatioCut(A, B) = \sum_{i \in A, j \in B} w_{i,j}(\frac{1}{|A|} + \frac{1}{|B|})$$

We also define the normalized cut function by

$$NCut(A, B) = \sum_{i \in A, j \in B} w_{i,j}(\frac{1}{vol(A)} + \frac{1}{vol(B)})$$

We can naturally generalize this idea to $k \geq 2$.

However, these last two objective functions are hard to compute (NP-hard problem), which will motivate further approximations.

4.3 Approximations : spectral clustering in practice

From now, we will add the constraint that all clusters' cardinals are equal : $\min_{|A|=|B|} cut(A, B)$.
If we consider the graph function $\mathbf{f}$ such that $f_i(V_i) = \mathbb{1}_{V_i \in A} - \mathbb{1}_{V_i \in B}$, we can observe that $cut(A, B) = \sum_{i \in A, j \in B} w_{i,j} = \frac{1}{4} \sum_{i \in A, j \in B} w_{i,j}(f_i - f_j)^2 = \frac{1}{2}\mathbf{f}^\top \mathbf{L}\mathbf{f} = \frac{S_G(\mathbf{f})}{2}$.
Also note that $|A| = |B|$ implies $\sum_i f_i = 0$ i.e. $\mathbf{f} \perp 1_N$. We also have $\|\mathbf{f}\| = \sqrt{N}$.
Thus, we can reformulate the problem as followed :

$$\min_{f_i = \pm 1, \mathbf{f} \perp 1_N, \|\mathbf{f}\| = \sqrt{N}} \mathbf{f}^\top \mathbf{L}\mathbf{f}$$

However, this is still a NP-hard problem ; there is still a difficulty that inherently comes from the discrete values taken by $\mathbf{f}$, which is why we relax it to real numbers and consider the following problem:

$$\min_{f_i \in \mathbb{R}, \mathbf{f} \perp 1_N, \|\mathbf{f}\| = \sqrt{N}} \mathbf{f}^\top \mathbf{L}\mathbf{f}$$

Theorem (Rayleigh-Ritz) If $\lambda_1 \leq \cdots \leq \lambda_N$ are the eigenvalues of real symmetric L then

$$\lambda_1 = \min_{\mathbf{x} \neq 0} \frac{\mathbf{x}^\top \mathbf{L}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} = min_{\|\mathbf{x}\|=1} \mathbf{x}^\top \mathbf{L}\mathbf{x}$$

$$\lambda_N = \max_{\mathbf{x} \neq 0} \frac{\mathbf{x}^\top \mathbf{L}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} = max_{\|\mathbf{x}\|=1} \mathbf{x}^\top \mathbf{L}\mathbf{x}$$

Proof. We do the computation:

$$\mathbf{x}^\top \mathbf{L}\mathbf{x} = \mathbf{x}\mathbf{Q}\Lambda\mathbf{Q}^\top \mathbf{x} = (\mathbf{Q}^\top \mathbf{x})^\top \Lambda(\mathbf{Q}^\top \mathbf{x}) = \sum_{i=1}^{N} \lambda_i (\mathbf{Q}^\top \mathbf{x})_i^2 \leq \lambda_N \sum_{i=1}^{N} (\mathbf{Q}^\top \mathbf{x})_i^2 = \lambda_N \|\mathbf{x}\|^2$$

The proof for λ_1 is analogous.

Another proof of the Rayleigh-Ritz theorem relies on the use of partial derivatives. □

We can naturally generalize the Rayleigh-Ritz theorem for all eigenvalues.
Theorem (Generalized Rayleigh-Ritz) If $\lambda_1 \leq \cdots \leq \lambda_N$ are the eigenvalues of real symmetric L, $\mathbf{v}_1 \dots \mathbf{v}_N$ the corresponding eigenvectors, then for $k = 1 : N - 1$:

$$\lambda_{k+1} = \min_{\mathbf{x} \neq 0, \mathbf{x} \perp \mathbf{v}_1 \dots \mathbf{v}_k} \frac{\mathbf{x}^\top \mathbf{L}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} = min_{\|\mathbf{x}\|=1, \mathbf{x} \perp \mathbf{v}_1 \dots \mathbf{v}_k} \mathbf{x}^\top \mathbf{L}\mathbf{x}$$

$$\lambda_{N-k} = \max_{\mathbf{x} \neq 0, \mathbf{x} \perp \mathbf{v}_N \dots \mathbf{v}_{N-k+1}} \frac{\mathbf{x}^\top \mathbf{L}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} = max_{\|\mathbf{x}\|=1, \mathbf{x} \perp \mathbf{v}_N \dots \mathbf{v}_{N-k+1}} \mathbf{x}^\top \mathbf{L}\mathbf{x}$$

Let's go back to the original problem. We did not approximate the clustering. From the second eigenvector, we can use a natural threshold heuristic. However, this does not give good results which is why in practice, we rather perform k-means on the eigenvectors to get the assignments.

However, we wanted to approximate RatioCut and Normalized Cut. If we consider the graph function $\mathbf{f}$ such that $f_i(V_i) = \mathbb{1}_{V_i \in A} \times \sqrt{\frac{|B|}{|A|}} - \mathbb{1}_{V_i \in B} \times \sqrt{\frac{|A|}{|B|}}$, we observe that $S_G(\mathbf{f}) = \frac{1}{2} \sum_{i \in A, j \in B} w_{i,j}(f_i - f_j)^2 = (|A| + |B|) RatioCut(A, B)$.

Indeed, $\frac{1}{2} \sum_{i \in A, j \in B} w_{i,j}(f_i - f_j)^2 = \frac{1}{2} \left(\sum_{i \in A} \sum_{j \in B} w_{i,j} \left(\left(\sqrt{\frac{|B|}{|A|}} + \sqrt{\frac{|A|}{|B|}} \right)^2 + \left(\sqrt{\frac{|B|}{|A|}} - \sqrt{\frac{|A|}{|B|}} \right)^2 \right) \right)$

$= \frac{1}{2} \left(\sum_{i \in A, j \in B} w_{i,j} \left(2\frac{|B|}{|A|} + 2\frac{|A|}{|B|} + 4 \right) \right) = (|A| + |B|) \sum_{i \in A, j \in B} w_{i,j} (\frac{1}{|A|} + \frac{1}{|B|})$.

We also have $\sum_i f_i = 0$ and $\sum_i {f_i}^2 = N$.

Thus, it comes down to the same optimization problem:

$$\min_{f_i \in \mathbb{R}, \mathbf{f} \perp 1_N, \|\mathbf{f}\| = \sqrt{N}} \mathbf{f}^\top \mathbf{L} \mathbf{f}$$

The Rayleigh-Ritz solution of this problem is still the second eigenvector of $\mathbf{L}$ from which we deduce a clustering as done previously.

Finally we can do something similar to approximate Normalized Cut. Using the graph function $\mathbf{f}$ such that $f_i(V_i) = \mathbb{1}_{V_i \in A} \times \sqrt{\frac{vol(B)}{vol(A)}} - \mathbb{1}_{V_i \in B} \times \sqrt{\frac{vol(A)}{vol(B)}}$, we show that $\mathbf{f}^\top \mathbf{L} \mathbf{f} = vol(\mathcal{V}) NCut(A, B)$. Thus, it comes down to a slightly different optimization problem :

$$\min_{f_i \in \mathbb{R}, \mathbf{D}\mathbf{f} \perp 1_N, \mathbf{f}^\top \mathbf{D} \mathbf{f} = vol(\mathcal{V})} \mathbf{f}^\top \mathbf{L} \mathbf{f}$$

If we now define $\mathbf{w} = \mathbf{D}^{1/2} \mathbf{f}$, we can rewrite this optimization problem in two different ways:

$$\min_{\mathbf{w}_i \in \mathbb{R}, \mathbf{w} \perp \mathbf{D}^{1/2} 1_N, \|\mathbf{w}\|^2 = vol(\mathcal{V})} \mathbf{w}^\top \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} \mathbf{w}$$

$$\min_{\mathbf{w}_i \in \mathbb{R}, \mathbf{w} \perp \mathbf{v_{1, L_{sym}}}, \|\mathbf{w}\|^2 = vol(\mathcal{V})} \mathbf{w}^\top \mathbf{L}_{sym} \mathbf{w}$$

The Rayleigh-Ritz solution of this problem is $\mathbf{w} = \mathbf{v_{2, L_{sym}}}$ hence $\mathbf{f} = \mathbf{D}^{-1/2} \mathbf{w} = \mathbf{v_{2, L_{rv}}}$, i.e. we can take the second eigenvector of $\mathbf{L_{rv}}$ from which we can deduce a clustering as done previously.

We now have to evaluate whether these approximations lead to unwanted behaviors. Actually, they can have disastrous consequences. One example is cockroach graphs. Other relaxations are possible, but with every relaxation has its limits.

4.4 Choice of the number of clusters

Determining the optimal number of clusters in a data set is a fundamental issue in partitioning clustering. In this method, we can propose an efficient method to choose it by observing the

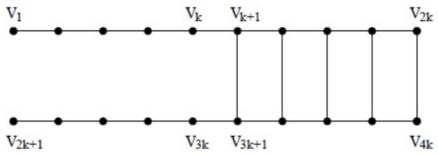

Figure 5: An example of graph where the several approximations we made lead to disastrous results

distribution of the eigenvalues and look for elbows. Because the multiplicity of 0 equals the number of connected components, we can choose for the number of clusters the number of small eigenvalues, and thus select the number where the eigenvalues distribution marks an elbow.

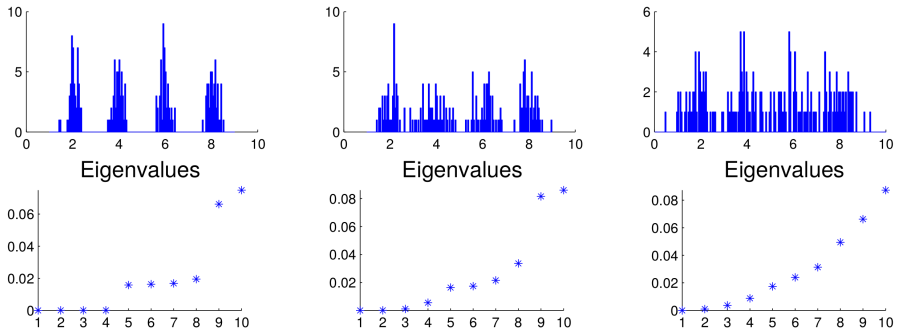

Figure 6: Elbow-rule. For the first case, 4 is exactly the number of connected components and the multiplicity of the eigenvalue 0. With a bit more noise (middle) we still clearly see a discontinuity for 4.

Spectral clustering in the following example would also behave well, because it is connected. Note that k-means would also work nicely, as it is also compact.

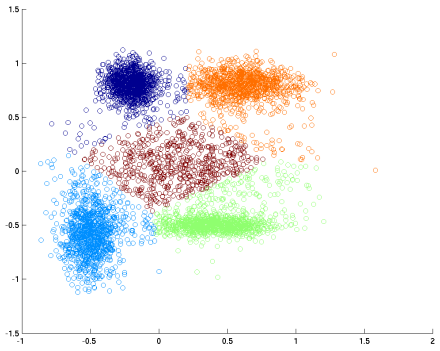

Figure 7: A compact and connected graph

References

[1] Alaa Saade, Florent Krzakala, and Lenka Zdeborová. Spectral clustering of graphs with the bethe hessian. In *NIPS*, 2014.

[2] Rakesh Shivanna and Chiranjib Bhattacharyya. Learning on graphs using orthonormal representation is statistically consistent. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3635–3643. Curran Associates, Inc., 2014.

[3] L. Lovász. Random walks on graphs: A survey. In D. Miklós, V. T. Sós, and T. Szőnyi, editors, *Combinatorics, Paul Erdős is Eighty*, volume 2, pages 353–398. János Bolyai Mathematical Society, Budapest, 1996.

[4] J. M Kleinberg. *An impossibility theorem for clustering.* Advances in neural information processing systems, 2003.