GUSTAVE EIFFEL UNIVERSITY

MASTER 2 MATHEMATICS AND COMPUTER SCIENCES

DATA SCIENCES

Spectral clustering & Stellar data

Mamadou DIOUF

Years : 2024 / 2025

Contents

List of Figures

1 Spectral clustering

1.1 Introduction

Clustering is the process of grouping similar objects together. Spectral clustering is a powerful technique for partitioning data into clusters based on the connectivity information within the data(adjacency matrix). His techniques leverages the eigenvalues and eigenvectors of a graph Laplacian matrix constructed from pairwise similarities.

1.2 Similarity matrix

Given a set of data points $\mathbf{x_1}, ...\mathbf{x_n}$ and some notion of similarity $\mathbf{w_{ij} \geq 0}$ between all pairs of data points x_i and x_j , the intuitive goal of clustering is to divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other. If we do not have more information than similarities between data points, a nice way of representing the data is in form of the similarity graph $G = (V, E)$. Each vertex v_i in this graph represents a data point x_i. Two vertices are connected if the similarity w_{ij} between the corresponding data points x_i and x_j is positive or larger than a certain threshold, and the edge is weighted by w_{ij} .Common similarity measures include: euclidean distance(for continuous data),cosine similarity(for high-dimensional data) and RBF kernel(for non-linear relationships). It can be also the adjacency matrix $\mathbf{W}$ of the graph ,thus w_{ij} represents the similarity between nodes i and j.
The problem of clustering can now be reformulated using the similarity graph: we want to find a partition of the graph such that the edges between different groups have very low weights (which means that points in different clusters are dissimilar from each other) and the edges within a group have high weights (which means that points within the same cluster are similar to each other).

1.3 Graph Laplacian

Let $\mathbf{W}$ be a symmetric weight(similarity) matrix for a graph, where $w_{ij} = w_{ji} \geq 0$. Let $\mathbf{D} = diag(d_i)$ be a diagonal matrix containing the weighted degree of each node. We have $d_i = \sum_j w_{ij}$. We define the **graph Laplacian** as follows:

$$\mathbf{L} = \mathbf{D} - \mathbf{W}$$

This matrix has various important properties. Because each row sums to **zero**, we have that $\mathbf{1}$ is an eigenvector with eigenvalue $\mathbf{0}$. Furthermore, the matrix is symmetric and positive semi-definite. To see this, note that:

$$\mathbf{f^T L f} = \mathbf{f^T D f} - \mathbf{f^T W f} = \sum_i d_i f_i^2 - \sum_{i,j} w_{ij} f_i f_j = \frac{1}{2}\Big(\sum_i d_i f_i^2 - 2\sum_{i,j} w_{ij} f_i f_j + \sum_j d_j f_j^2\Big)$$
$$= \frac{1}{2}\Big(\sum_i \sum_j w_{ij} f_i^2 - 2\sum_{i,j} w_{ij} f_i f_j + \sum_j \sum_i w_{ij} f_j^2\Big) = \frac{1}{2}\sum_{i,j} w_{ij}(f_i - f_j)^2$$

Hence $\mathbf{f^T L f \geq 0}$ for all $f \in R^n$. Consequently we see that $\mathbf{L}$ has n non-negative, real-valued eigenvalues, $0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

1.3.1 Normalized graph Laplacian

In practice, it is important to normalize the graph Laplacian, to account for the fact that some nodes are more highly connected than others. There are two common ways to do this. One method od, used in e.g., (Shi and Malik 2000; Meila 2001), creates a stochastic matrix (random walk normalized Laplacian) where each row sums to one:

$$\mathbf{L_{rw} = D^{-1}L = I - D^{-1}W}$$

The eigenvalues and eigenvectors of $\mathbf{L}$ and $\mathbf{L_{rw}}$ are closely related to each other . And that the eigenvectors/ values of L_{rw} are equivalent to the generalized eigenvectors/ values of $\mathbf{L}$, which solve $\mathbf{Lu} = \lambda \mathbf{Du}$.
Another method, used in e.g., (Ng et al. 2001), creates a symmetric matrix(symmetric normalized Laplacian).

$$\mathbf{L_{sym} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}}$$

1.3.2 Theorem

The set of eigenvectors of $\mathbf{L}$ with eigenvalue 0 is spanned by the indicator vectors $1_{A_1}, ..., 1_{A_K}$, where A_k are the K connected components of the graph.

Proof: Let us start with the case $K = 1$. If f is an eigenvector with eigenvalue $\mathbf{0}$, then $0 = \sum_{i,j} w_{ij}(f_i - f_j)^2$. If two nodes i, j are connected, so $w_{ij} > 0$, we must have that $f_i = f_j$.

Hence f is constant for all vertices which are connected by a path in the graph, we have only one component.

Now suppose $K > 1$. In this case, L will be block diagonal, each block will represent a component. A similar argument to the above shows that we will have K indicator functions, which "select out" the connected components. $\square$

1.4 Spectral clustering algorithm

Now we would like to state the most common spectral clustering algorithms. We assume that our data consists of $\mathbf{n}$ "points" $\mathbf{x_1}, ..., \mathbf{x_n}$ which can be arbitrary objects. We measure their pairwise similarities $\mathbf{w_{ij} = w(x_i, x_j)}$ by some similarity function which is symmetric and non-negative, and we denote the corresponding similarity matrix by $\mathbf{W = (w_{ij})_{i,j=1,\cdots,n}}$.

1.4.1 Unnormalized spectral clustering

This suggests the following algorithm. Compute the first K eigenvectors u_k of $\mathbf{L}$.
Let $\mathbf{U} = [\mathbf{u_1}, \cdots, \mathbf{u_K}]$ be an $N \times K$ matrix with the eigenvectors in its columns, with $N = n$.
Let $y_i \in R^K$ be the i^{th} row of U. Since these y_i will be piecewise constant, we can apply K-means clustering to them to recover the connected components. Now assign point i to cluster k if row i of $\mathbf{Y}$ was assigned to cluster k.

```
Input: data, number k of clusters to construct.
    • Construct a similarity graph by one of the ways described before.
      Let $\mathbf{W}$ be its weighted adjacency matrix.
    • Compute the unnormalized Laplacian $\mathbf{L}$
    • Compute the first $k$ eigenvectors $u_1, \cdots, u_k$ of $\mathbf{L}$
    • Let $U \in R^n$ be the matrix containing the vectors $u_1, \cdots, u_k$ as columns.
    • For $i = 1, ..., n$,let $y_i \in \mathbb{R}^k$ be the $i^{th}$ row of $U$
    • Cluster the points $(y_i)_{i=1,...,n}$ in $\mathbb{R}^k$ with the $k$-means algorithm into
      clusters $C_1, ..., C_k$.
Output: Clusters $A_1, ..., A_k$ with $A_i = \{j | y_j \in C_i\}$
```

1.4.2 Normalized spectral clustering

There are two different versions of normalized spectral clustering, depending which of the normalized graph Laplacians is used.
For $\mathbf{L_{rw}}$, the eigenspace of 0 is again spanned by the indicator vectors $\mathbf{1}_{A_k}$. This suggests the following algorithm: find the smallest K eigenvectors of $\mathbf{L_{rw}}$, create U, cluster the rows of U using K-means, then infer the partitioning of the original points (Shi and Malik 2000).

```
Normalized spectral clustering with $\mathbf{L_{rw}}$
Input: data, number k of clusters to construct.
    • Construct a similarity graph by one of the ways described before.
      Let $\mathbf{W}$ be its weighted adjacency matrix.
    • Compute the unnormalized Laplacian $\mathbf{L}$
    • Compute the first $k$ generalized eigenvectors $u_1, \cdots, u_k$ of
      the generalized eigenproblem $\mathbf{Lu} = \lambda\mathbf{Du}$
    • Let $U \in R^n$ be the matrix containing the vectors $u_1, \cdots, u_k$ as columns.
    • For $i = 1, ..., n$,let $y_i \in \mathbb{R}^k$ be the $i^{th}$ row of $U$
    • Cluster the points $(y_i)_{i=1,...,n}$ in $\mathbb{R}^k$ with the $k$-means algorithm into
      clusters $C_1, ..., C_k$.
Output: Clusters $A_1, ..., A_k$ with $A_i = \{j | y_j \in C_i\}$
```

For $\mathbf{L_{sym}}$, the eigenspace of 0 is spanned by $D^{\frac{1}{2}} A_k$. This suggest the following algorithm: find the smallest K eigenvectors of $\mathbf{L_{sym}}$, create U, normalize each row to unit norm by creating $\mathbf{t_{ij}} = \mathbf{u_{ij}} / (\sum_\mathbf{k} \mathbf{u_{ik}^2})^{\mathbf{1/2}}$, cluster the rows of T using K-means, then infer the partitioning of the original points (Ng et al. 2001).

```
Normalized spectral clustering with $\mathbf{L_{sym}}$
Input:  data, number k of clusters to construct.
    • Construct a similarity graph by one of the ways described before.
      Let $\mathbf{W}$ be its weighted adjacency matrix.
    • Compute the unnormalized Laplacian $\mathbf{L}$
    • Compute the first $k$ eigenvectors $u_1, \cdots, u_k$ of $\mathbf{L_{sym}}$
    • Let $U \in R^{n \times k}$ be the matrix containing the vectors $u_1, \cdots, u_k$ as columns.
    • Form the matrix $T \in R^{n \times k}$ from $U$ by normalizing the rows to norm 1,
      that is $t_{ij} = u_{ij}/(\sum_k u_{ik}^2)^{1/2}$
    • For $i = 1, ..., n$,let $y_i \in \mathbb{R}^k$ be the $i^{th}$ row of $T$
    • Cluster the points $(y_i)_{i=1,...,n}$ in $\mathbb{R}^k$ with the $k$-means algorithm into
      clusters $C_1, ..., C_k$.
Output:  Clusters $A_1, ..., A_k$ with $A_i = \{j | y_j \in C_i\}$
```

1.5 Graph-cut algorithm

An alternative view of **clustering** is in terms of **graph cuts**. The idea is we create a weighted undirected graph **W** from the similarity matrix **S**, typically by using the nearest neighbors of each point; this ensures the graph is sparse, which speeds computation. If we want to find a partition into K clusters, say $A_1, ..., A_K$, one natural criterion is to minimize:

$$cut(A_1, ..., A_K) := \frac{1}{2} \sum_{i=1}^{K} W(A_i, \bar{A}_i)$$

where $\bar{A}_k = V \backslash \bar{A}_k$ is the complement of A_k, and $W(A, B) = \sum_{i \in A, j \in B} w_{ij}$.

Here we introduce the factor $\frac{1}{2}$ for notational consistency, otherwise we would count each edge twice in the cut. In particular for $k = 2$, **mincut** is a relatively easy problem and can be solved efficiently, see Stoer and Wagner (1997) and the discussion therein. However, in practice it often does not lead to satisfactory partitions. The problem is that in many cases, the solution of **mincut** simply separates *one individual vertex* from the rest of the graph. Of course this is not what we want to achieve in clustering, as clusters should be reasonably *large groups of points*. One way to circumvent this problem is to explicitly request that the sets $\mathbf{A_1}, ..., \mathbf{A_k}$ are "*reasonably large*". The two most common objective functions to encode this are **RatioCut** (Hagen and Kahng, 1992) and the *normalized cut* **Ncut** (Shi and Malik, 2000). In **RatioCut**, the size of a subset A of a graph is measured by its number of vertices $|\mathbf{A}|$, while in **Ncut** the size is measured by the weights of its edges $\mathbf{vol}(|\mathbf{A}|)$. The definitions are:

$$|A| := \text{the number of vertices in } A$$

$$vol(A) := \sum_{i \in A} d_i$$

$$cut(A_1, ..., A_K) := \frac{1}{2} \sum_{i=1}^{K} \frac{W(A_i, \bar{A}_i)}{|A_i|} = \sum_{i=1}^{K} \frac{cut(A_i, \bar{A}_i)}{|A_i|}$$

$$Ncut(A_1, ..., A_K) := \frac{1}{2} \sum_{i=1}^{K} \frac{W(A_i, \bar{A}_i)}{vol(|A_i|)} = \sum_{i=1}^{K} \frac{cut(A_i, \bar{A}_i)}{vol(|A_i|)}$$

Note that both objective functions take a small value if the clusters Ai are not too small. In particular, the minimum of the function $\sum_{i=1}^{k}(1/|A_i|)$ is achieved if all $|A_i|$ coincide, and the minimum of $\sum_{i=1}^{k}(1/vol(A_i)$ is achieved if all $vol(A_i)$ coincide. So what both objective functions try to achieve is that the clusters are "*balanced*", as measured by the number of vertices or edge weights, respectively. Unfortunately, introducing balancing conditions makes the previously simple to solve *mincut* problem become NP hard. Spectral clustering is a way to solve relaxed versions of those problems. We will see that relaxing *Ncut* leads to normalized spectral clustering, while relaxing *RatioCut* leads to unnormalized spectral clustering .

1.5.1 Approximating RatioCut

RatioCut for k=2

Let us start with the case of *RatioCut* and $k = 2$, because the relaxation is easiest to understand in this setting. Our goal is to solve the optimization problem:

$$\min_{\mathbf{A}\subset\mathbf{V}} \mathbf{RatioCut}(\mathbf{A}, \bar{\mathbf{A}})$$

We first rewrite the problem in a more convenient form. Given a subset $A \subset V$ we define the vector $f = (f_1, ..., f_n)^T \in \mathbb{R}^n$ with entries

$$f_i = \begin{cases} \sqrt{|\bar{A}|/|A|} & \text{if } v_i \in A \\ \sqrt{|A|/|\bar{A}|} & \text{if } v_i \in \bar{A} \end{cases} \qquad (D1)$$

Now the RatioCut objective function can be conveniently rewritten using the unnormalized graph Laplacian. This is due to the following calculation:

$$f^T Lf = \frac{1}{2}\sum_{i,j} w_{ij}(f_i - f_j)^2$$

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

$$= \frac{1}{2}\left(\sqrt{\frac{|\bar{A}|}{|A|}} + \sqrt{\frac{|A|}{|\bar{A}|}}\right)^2\left(\sum_{i\in A, j\in\bar{A}} w_{ij} + \sum_{i\in\bar{A}, j\in A} w_{ij}\right)$$

$$= cut(A, \bar{A})\left(\frac{|\bar{A}|}{|A|} + \frac{|A|}{|\bar{A}|} + 2\right)$$

$$= cut(A, \bar{A})\left(\frac{|A| + \bar{A}|}{|A|} + \frac{|A| + |\bar{A}|}{|\bar{A}|}\right) = |V| \cdot Ratiocut(A, \bar{A})$$

Additionally, we have

$$\sum_{i=1}^{n} f_i = \sum_{i\in A}\sqrt{\frac{|\bar{A}|}{|A|}} - \sum_{i\in\bar{A}}\sqrt{\frac{|A|}{|\bar{A}|}} = |A|\sqrt{\frac{|\bar{A}|}{|A|}} - |\bar{A}|\sqrt{\frac{|A|}{|\bar{A}|}} = |A||\bar{A}| - |\bar{A}||A| = 0$$

In other words, the vector f as defined in Equation $(D1)$ is orthogonal to the constant one vector $\mathbf{1}$. Finally, note that f satisfies

$$||f||^2 = \sum_{i=1}^{n} f_i^2 = |A|\frac{|\bar{A}|}{|A|} - |\bar{A}|\frac{|A|}{|\bar{A}|} = |\bar{A}| + |A| = |V| = n$$

Altogether we can see that the problem of minimizing (1) can be equivalently rewritten as

$$\min_{A\subset V} f^T L f \text{ subject to } f \perp \mathbf{1}, f_i \text{ as defined in Eq. (2)}, \ \|f\| = \sqrt{n}.$$

This is a discrete optimization problem as the entries of the solution vector f are only allowed to take two particular values, and of course it is still $NP - hard$. The most obvious relaxation in this setting is to discard the discreteness condition and instead allow that f_i takes arbitrary values in . This leads to the relaxed optimization problem

$$\min_{f\in\mathbb{R}^n} f^T L f \text{ subject to } f \perp \mathbf{1}, \ \|f\| = \sqrt{n}$$

Rayleigh theorem:

The *Rayleigh quotient* of a matrix $\mathbb{A}$ at $f \neq 0 \in \mathbb{R}^n$ is

$$R(\mathbb{A}, f) = \frac{\langle \mathbb{A} f, f\rangle}{\langle f, f\rangle}.$$

And we have :

$$\lambda_k = \min_{f\neq 0, f\perp span(f_1,\cdots,f_{k-1})} R(\mathbb{A}, f).$$

Since

$$\min_{f\in\mathbb{R}^n, \|f\|=\sqrt{n}} f^T L f \quad \text{subject to } f \perp \mathbf{1} = \min_{f\neq 0, f\perp\mathbf{1}} R(\mathbb{A}, f)\sqrt{n} = \min_{f\neq 0, f\perp\mathbf{1}} R(\mathbb{A}, f) = \lambda_2$$

Because $\mathbf{1}$ is a eigenvector of the eigenvalue $\lambda_1 = 0$ of L.

Hence we can approximate a minimizer of *RatioCut* by the second eigenvector of L, λ_2. However, in order to obtain a partition of the graph we need to re-transform the real-valued solution vector f of the relaxed problem into a discrete indicator vector. The simplest way to do this is to use the sign of f as indicator function, that is to choose

$$\begin{cases} \mathrm{v}_i \in A \text{ if } f_i \geq 0 \\ \mathrm{v}_i \in \bar{A} \text{ if } f_i < 0 \end{cases}$$

Ratiocut k>2

The relaxation of the *RatioCut* minimization problem in the case of a general value $k > 2$ follows a similar principle as the one above. Given a partition of V into k sets $A_1, \cdots, A_k$, we define k indicator vectors $h_j = (h_{1,j}, ..., h_{n,j})^T$ by

$$h_{i,j} = \begin{cases} \dfrac{1}{\sqrt{|A_j|}} & \text{if } v_i \in A_j \\ 0 & \text{otherwise} \end{cases} \qquad (i = 1, \cdots, n; j = 1, \cdots, k). \qquad (E1)$$

Then we set the matrix $H \in \mathbb{R}^{n\times k}$ as the matrix containing those k indicator vectors as columns. Observe that the columns in H are orthonormal to each other, that is $H^T H = I$. Similar to the calculations in the last section we can see that

$$h_i^T L h_i = \frac{cut(A_i, \bar{A}_i)}{|A_i|}$$

Moreover, one can check that

$$h_i^T L h_i = (H^T L H)_{ii}$$

Combining those facts we get

$$RatioCut(A_1, ..., A_k) = \sum_{i=1}^{k} h_i^T L h_i = \sum_{i=1}^{k} (H^T L H)_{ii} = Tr(H^T L H),$$

where Tr denotes the trace of a matrix. So the problem of minimizing $RatioCut(A_1, \cdots, A_k)$ can be rewritten as

$$\min_{A_1, \cdots, A_k} = Tr(H^T L H) \text{ subject to } H^T H = I, H \text{ as defined in } E1$$

Similar to above we now relax the problem by allowing the entries of the matrix H to take arbitrary real values. Then the relaxed problem becomes:

$$\min_{H \in \mathbb{R}^{n \times k}} Tr(H^T L H) \text{ subject to } H^T H = I.$$

This is the standard form of a trace minimization problem, and again a version of the Rayleigh theorem tells us that the solution is given by choosing H as the matrix which contains the first k eigenvectors of L as columns. We can see that the matrix H is in fact the matrix U used in the unnormalized spectral clustering algorithm . Again we need to re-convert the real valued solution matrix to a discrete partition. As above, the standard way is to use the k-means algorithms on the rows of U.

1.5.2 Approximating Ncut

RatioCut can be used to derive normalized spectral clustering as relaxation of minimizing $Ncut$.

Ncut k=2

In the case $k = 2$ we define the cluster indicator vector f by

$$f_i = \begin{cases} \sqrt{vol(\bar{A})/vol(A)} & \text{if } v_i \in A \\ -\sqrt{vol(A)/vol(\bar{A})} & \text{if } v_i \in \bar{A} \end{cases} \qquad (D2)$$

We have $Df = (d_1 f_1, \cdots, d_n f_n)^T$ then:

$$(Df)^T = (d_1 f_1, \cdots, d_n f_n)\mathbf{1} = \sum_{i=1}^{n} d_i f_i = \sum_{i \in A} d_i f_i + \sum_{i \in \bar{A}} d_i f_i$$

$$= \sum_{i \in A} d_i \sqrt{\frac{vol(\bar{A}}{vol(A)}} - \sum_{i \in \bar{A}} d_i \sqrt{\frac{vol(A)}{vol(\bar{A})}} = vol(A)\sqrt{\frac{vol(\bar{A}}{vol(A)}} - vol(\bar{A})\sqrt{\frac{vol(A)}{vol(\bar{A})}} = 0$$

Thus $Df \perp \mathbf{1}$.

$$f^T D f = \sum_{i=1}^{n} d_i f_i^2 = \sum_{i\in A} d_i \frac{vol(\bar{A}}{vol(A)} + \sum_{i\in \bar{A}} d_i \frac{vol(A)}{vol(\bar{A})} = vol(A) + vol(\bar{A}) = vol(V)$$

$f^T D f = vol(V)$.

$$f^T L f = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 = \sum_{i\in A, j\in\bar{A}} w_{ij}(f_i - f_j)^2 + \sum_{j\in A, i\in\bar{A}} w_{ij}(f_i - f_j)^2$$

$$= \sum_{i\in A, j\in\bar{A}} w_{ij}\left(\sqrt{\frac{vol(\bar{A})}{vol(A)}} + \sqrt{\frac{vol(A)}{vol(\bar{A})}}\right)^2 + \sum_{j\in A, i\in\bar{A}} w_{ij}\left(-\sqrt{\frac{vol(\bar{A})}{vol(A)}} - \sqrt{\frac{vol(A)}{vol(\bar{A})}}\right)^2$$

$$= cut(A,\bar{A})\left(\frac{vol(\bar{A})}{vol(A)} + \frac{vol(A)}{vol(\bar{A})} + 2\right)$$

$$= cut(A,\bar{A})\left(\frac{vol(\bar{A}) + vol(A)}{vol(A)} + \frac{vol(A) + vol(\bar{A})}{vol(\bar{A})}\right)$$

$$= vol(V) Ncut(A,\bar{A})$$

Thus we can rewrite the problem of minimizing **Ncut** by the equivalent problem

$$\min_{A} f^T L f \text{ subject to } f \text{ as in } (D2) \quad Df \perp \mathbf{1}, f^T D f = vol(V).$$

we relax the problem by allowing f to take arbitrary real values:

$$\min_{f\in\mathbb{R}^n} f^T L f \text{ subject to } Df \perp \mathbf{1}, f^T D f = vol(V).$$

Now we substitute $g := D^{1/2} f$. After substitution, the problem is

$$\min_{g\in\mathbb{R}^n} g^T D^{-1/2} L D^{-1/2} g \quad \text{subject to } g \perp D^{1/2}\mathbf{1}, \|g\|^2 = vol(V).$$

Observe that $D^{1/2} L D^{1/2} = L_{sym}$,$D^{1/2}$ is the first eigenvector of L_{sym}, and $vol(V)$ is a constant. Hence, the problem is in the form of the standard Rayleigh-Ritz theorem, and its solution g is given by the second eigenvector of L_{sym}. Re-substituting$f = D^{1/2} g$ and using we see that f is the second eigenvector of L_{rw}, or equivalently the generalized eigenvector of $Lu = \lambda Du$.

Ncut k>2

For the case of finding k > 2 clusters, we define k indicator vectors $h_j = (h_{1,j}, ..., h_{n,j})^T$ by

$$h_{i,j} = \begin{cases} \frac{1}{\sqrt{vol(A_j)}} & \text{if } v_i \in A_j \\ 0 & \text{otherwise} \end{cases} \qquad (i = 1, \cdots, n; j = 1, \cdots, k). \qquad (E2)$$

Then we set the matrix $H \in \mathbb{R}^{n\times k}$ as the matrix containing those k indicator vectors as columns. Observe that the columns in H are orthonormal to each other, that is $H^T D H = I$. Similar to the calculations in the last section we can see that

$$h_i^T L h_i = \frac{cut(A_i, \bar{A}_i)}{vol(A_i)}$$

Moreover, one can check that

$$h_i^T L h_i = (H^T L H)_{ii}$$

Combining those facts we get

$$NCut(A_1, ..., A_k) = \sum_{i=1}^{k} h_i^T L h_i = \sum_{i=1}^{k} (H^T L H)_{ii} = Tr(H^T L H),$$

where Tr denotes the trace of a matrix. So the problem of minimizing $NCut(A_1, \cdots, A_k)$ can be rewritten as

$$\min_{A_1, \cdots, A_k} = Tr(H^T L H) \text{ subject to } H^T D H = I, H \text{ as defined in } (E2)$$

Relaxing the discreteness condition and substituting $M = D^{-1/2}H$ Then the relaxed problem becomes:

$$\min_{M \in \mathbb{R}^{n \times k}} Tr(M^T D^{-1/2} L D^{-1/2} M) \text{ subject to } M^T M = I.$$

Thus we have the standard trace minimization problem which is solved by the matrix M which contains the first k eigenvectors of L_{sym} as columns. Resubstituting $H = D^{1/2}M$ and we see that the solution H consists of the first k eigenvectors of the matrix L_{rw}, or the first k generalized eigenvectors of $Lu = \lambda Du$. This yields the normalized spectral clustering algorithm.

2 Stellar data

2.1 Introduction

3 Annex

Machine Learning, A Probabilistic Perspective, Kevin P. Murphy

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17, 395–416. Springer.