SPECTRAL CLUSTERING

Overview

Spectral clustering is a method for clustering that proceeds as follows:

1. Given unlabeled patterns $x_1, \ldots, x_n$, construct a graph where the nodes are the data and the edge weights reflect similarity
2. Form an $n \times n$ matrix L from the graph called the graph Laplacian
3. Infer a partition of the graph from the eigenvalue (or spectral) decomposition of L.

Unlike k-means and GMMs, spectral clustering can give rise to nonconvex clusters.

Similarity Graphs

Similarity graphs are defined by a graph structure and edge weights. The information is captured

by an $n \times n$ weighted adjacency matrix

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \\ w_{n1} & & w_{nn} \end{bmatrix}$$

where $w_{ij} \geq 0$, $w_{ij} > 0$ iff there is an edge between x_i & x_j. If $w_{ij} > 0$ we say x_i and x_j are adjacent. We also assume W is symmetric, i.e., $w_{ij} = w_{ji}$ $\forall i,j$. Here are some examples of common graph structures and edge weights.

Graph structures

- k-nearest neighbor graph: every x_i is adjacent to its k neighest neighbors
- ε-ball graph: every x_i is adjacent to every x_j within a radius of ε
- complete graph: every x_i is adjacent to every

other x_j

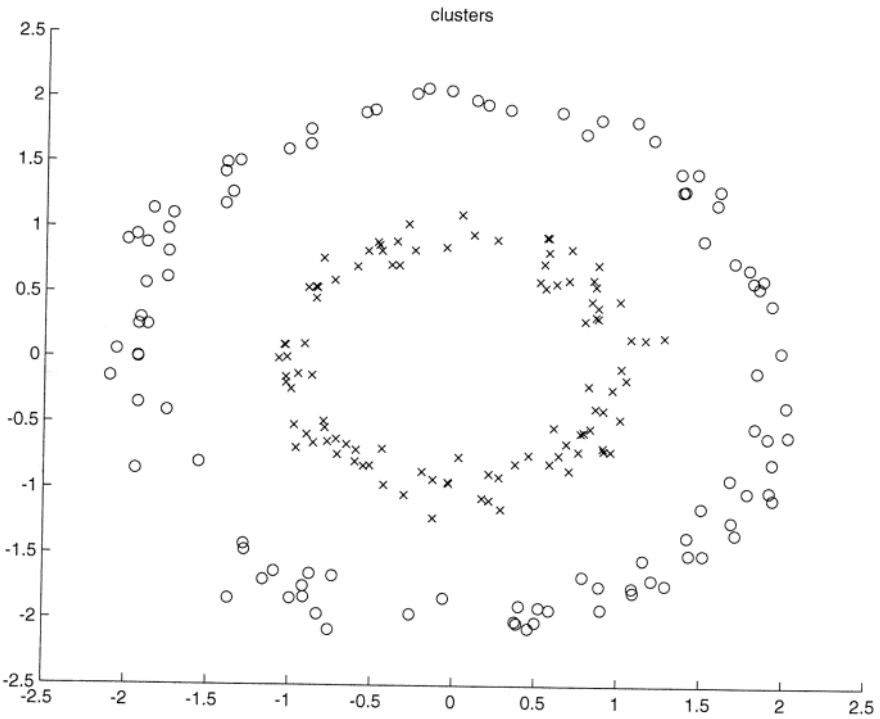
Edge weights

- Constant: $w_{ij} = \begin{cases} 1 & \text{if } x_i, x_j \text{ connected} \\ 0 & \text{otherwise} \end{cases}$

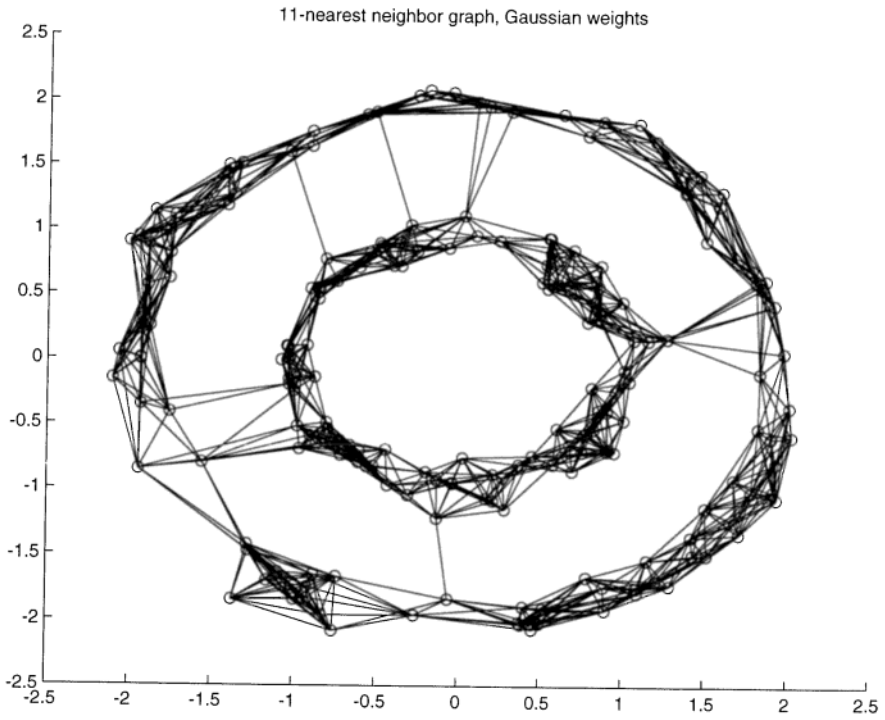
(not appropriate for complete graph)

- Gaussian: $w_{ij} = \begin{cases} \exp\left(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2\right), & x_i, x_j \text{ connected} \\ 0, & \text{otherwise} \end{cases}$

Here's a simple data set:



Here's a similarity graph:



Ideally, the similarity graph would only have edges between points in the same cluster. In practice, however, this is usually difficult to achieve. Model selection for similarity graphs (e.g., setting k, ε, or σ) is especially challenging.

There are at least four ways to derive spectral clustering:

1. Properties of graph Laplacian
2. Optimal graph cuts
3. Laplacian eigenmaps
4. Random walks

} we'll cover these three

Graph Laplacians

The (weighted) degree of a node x_i is

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

The degree matrix is the diagonal matrix

$$D = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix}$$

The unnormalized graph Laplacian is

$$L := D - W$$

Note that L is independent of the self-similarity weights w_{ii} because

$$L_{ii} = d_i - w_{ii} = \sum_{j \neq i} w_{ij}$$

Properties of L

1. For every $f \in \mathbb{R}^n$

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

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

2. L is symmetric and PSD

3. The smallest eigenvalue of L is 0, and $\underline{1} = [1 \; 1 \; \cdots \; 1]^T \in \mathbb{R}^n$ is a corresponding eigenvector.

Hence L has n nonnegative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

Proof

1. $$f^T L f = f^T D f - f^T W f$$

$$= \sum_{i=1}^{n} d_i f_i^2 - \sum_{i,j} w_{ij} f_i f_j$$

$$= \frac{1}{2} \left(\sum_{i=1}^{n} d_i f_i^2 - 2 \sum_{i,j} w_{ij} f_i f_j + \sum_{j=1}^{n} d_j f_j^2 \right)$$

$$= \frac{1}{2} \sum_{i,j} w_{ij} \left(f_i^2 - 2 f_i f_j + f_j^2 \right)$$

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

The second property follows from the first and

symmetry of W. To see the third property,

$$L\underline{1} = D\underline{1} - W\underline{1} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = 0 \cdot \underline{1}.$$

L encodes many properties of the similarity graph. The following is one such property that is relevant for clustering.

For $A \subseteq \{x_1, \ldots, x_n\}$, define the indicator vector

$$\underline{1}_A = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \in \mathbb{R}^n \quad \text{where} \quad f_i = \begin{cases} 1 & \text{if } x_i \in A \\ 0 & \text{if } x_i \notin A. \end{cases}$$

Also, note that the nullspace of L is also the 0-eigenspace of L: the subspace of all eigenvector associated with the eigenvalue 0.

Proposition If the graph has connected components $A_1, \ldots, A_k$, then the nullspace of L has dimension k and is spanned by $\underline{1}_{A_1}, \ldots, \underline{1}_{A_k}$.

Proof The nullspace of L is $N(L) = \{f : Lf = 0\}$.

It suffices to show

- $\mathbb{1}_{A_k} \in N(L)$ for each $k = 1, \ldots, K$
- If $f \in N(L)$, then

$$f \in \sum_{k=1}^{K} \alpha_k \mathbb{1}_{A_k}$$

for some $\alpha_1, \ldots, \alpha_K \in \mathbb{R}$.

Since $\mathbb{1}_{A_1}, \ldots, \mathbb{1}_{A_K}$ are clearly linearly independant, it follows that $\dim(N(L)) = K$.

First, consider the case $K = 1$. In this case, we have established $\underline{1} \in N(L)$ previously. To show the spanning property, suppose $f \in N(L)$. Then $Lf = 0$, and so

$$0 = f^T L f = \frac{1}{2} \sum_{i,j} w_{ij} (f_i - f_j)^2.$$

If x_i and x_j are adjacent, then $w_{ij} > 0$ which implies $f_i = f_j$. More generally, since $K = 1$, and two points x_i and x_j are connected by a path, and therefore $f_i = f_j$. Thus all f_i are equal to a

constant, and therefore f is a multiple of $\underline{1}$.

If $K > 1$, let us suppose the data are enumerated such that L is block diagonal:

$$L = \begin{bmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_K \end{bmatrix}.$$

Notice that L_k is the graph Laplacian on A_k.

Applying the previous case, we deduce that

- $L \underline{1}_{A_k} = 0$ for each k
- If $Lf = 0$, then f is piecewise constant on each A_k

$$\Longrightarrow f = \sum_{k=1}^{K} \alpha_k \underline{1}_{A_k}$$ ∎

A simple corollary leads to a procedure that forms

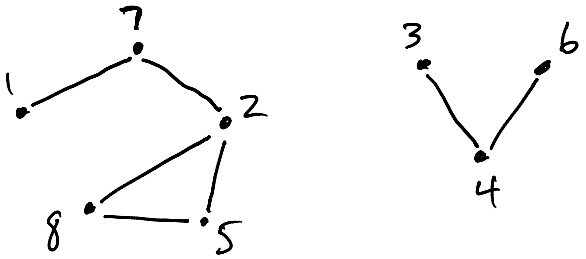
the basis of spectral clustering:

Corollary If $\{u_1, \ldots, u_k\} \subset \mathbb{R}^n$ is a basis of $N(L)$ and we define

$$y_i = (u_1^{(i)}, \ldots, u_k^{(i)}) \in \mathbb{R}^k,$$

then $y_i = y_j$ iff x_i and x_j are in the same connected component.

Proof by example



$$L = \begin{bmatrix} \bullet & & & & & & & \bullet \\ & \bullet & & & & & \bullet & \bullet \\ & & \bullet & \bullet & & & & \\ & & \bullet & \bullet & & \bullet & & \\ & \bullet & & & \bullet & & & \bullet \\ & & & \bullet & & \bullet & & \\ \bullet & \bullet & & & & & \bullet & \\ & \bullet & & & & & & \bullet \end{bmatrix}$$

$\bullet$ = nonzero entry

$$\underline{1}_{A_1} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad \underline{1}_{A_2} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$\mathbb{1}_{A_1} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbb{1}_{A_2} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Suppose u_1, u_2 are a basis of $N(L)$. Write

$$u_1 = \alpha_1 \mathbb{1}_{A_1} + \beta_1 \mathbb{1}_{A_2}$$

$$u_2 = \alpha_2 \mathbb{1}_{A_1} + \beta_2 \mathbb{1}_{A_2}$$

Then

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_2 \\ \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \\ \beta_1 & \beta_2 \\ \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \\ \alpha_1 & \alpha_2 \\ \alpha_1 & \alpha_2 \end{bmatrix}$$

Spectral Clustering

Let's apply the above ideas to develop a clustering

algorithm. In practice, we cannot hope for the connected components of the similarity graph to coincide with the clusters. Instead, we're more likely to have a situation like this:

$$L = \begin{bmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_k \end{bmatrix} + \begin{bmatrix} & & & \\ & \text{noise} & & \\ & & & \\ & & & \end{bmatrix}$$

$$= L_{ideal} + \Delta$$

where Δ accounts for edges between nodes in different clusters. If we have constructed a decent similarity graph, the entries of Δ should be much smaller than the entries of $L_1, \ldots, L_k$. We can then appeal to matrix perturbation theory, which (loosely speaking) says that the k smallest eigenvalues of L should still be close to zero, and their

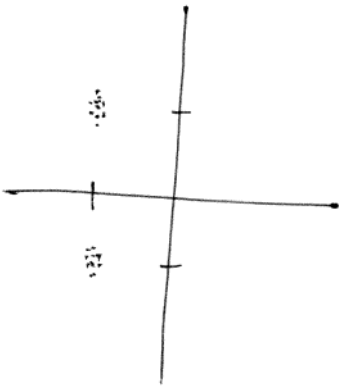
corresponding eigenvectors should still roughly span the nullspace of L_{ideal}. This motivates the spectral clustering algorithm:

- Input: $x_1, \ldots, x_n$
- Construct a similarity graph, form L
- Determine K smallest eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_K$ of L and corresponding eigenvectors $u_1, \ldots, u_K \in \mathbb{R}^n$
- Set $y_i = (u_1^{(i)}, \ldots, u_K^{(i)})$, $i = 1, \ldots, n$
- Cluster $\{y_i\}_{i=1}^n$ using k-means clustering, and assign $\{x_i\}_{i=1}^n$ to corresponding clusters

Here's what happens if we apply this to the data and similarity graph shown earlier:

-0.070711	-0.07071
-0.070711	-0.07071
-0.070711	-0.070709
-0.070711	-0.07071
-0.070711	-0.070708
-0.070711	-0.070708
-0.070711	-0.070709
-0.070711	0.071153
-0.070711	0.071682
-0.070711	0.070093
-0.070711	0.071059
-0.070711	-0.070708
-0.070711	-0.070709
-0.070711	0.070098
-0.070711	-0.07071
-0.070711	0.071489
-0.070711	-0.070709
-0.070711	-0.07071
-0.070711	0.070098
-0.070711	-0.070708
-0.070711	0.070098

a few randomly selected y_i's



The mapping $x_i \longmapsto y_i$ is actually a form of nonlinear dimensionality reduction s.t. in the y_i space, a simple algorithm like k-means is sufficient.

Normalized Spectral Clustering

The normalized graph Laplacian is

$$\hat{L} = D^{-1} L.$$

It's an easy exercise to check the following:

Proposition

1. $\hat{L}$ is symmetric PSD
2. 0 is an eigenvalue of $\hat{L}$ and $\underline{1} = [1\ 1\ \dots\ 1]^T \in \mathbb{R}^n$ is a corresponding eigenvector.

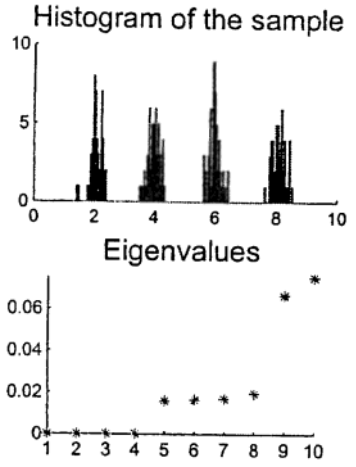
3. If the similarity graph has K connected components $A_1, \ldots, A_K$, then $\mathbb{1}_{A_1}, \ldots, \mathbb{1}_{A_K}$ is a basis for $N(\tilde{L})$.

Therefore we can substitute $\tilde{L}$ for L and get another spectral clustering algorithm.

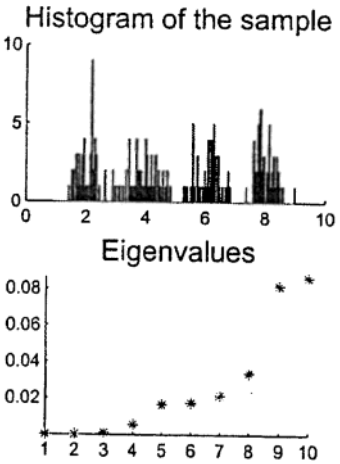
Normalized spectral clustering is recommended over unnormalized.

Selecting K

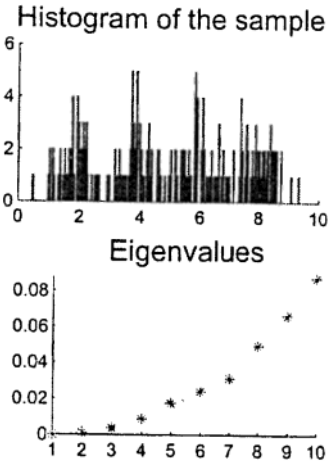
Here's three simple one-dimensional data sets with four clusters:



easy



medium



hard

If there is a jump in the sorted eigenvalues, from "near zero" to "not near zero", that probably indicates the number of clusters.

Graph Cuts

Given a similarity graph, we would like to find a partition $A_1, \ldots, A_K$ of $\{1, 2, \ldots, n\}$ such that

- w_{ij} large if x_i, x_j in same cluster
- w_{ij} small if x_i, x_j in different cluster.

We can approach this problem directly by trying to solve the mincut problem, which tries to minimize

$$\text{cut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{k=1}^{K} W(A_k, \bar{A}_k)$$

where

$$W(A, B) = \sum_{i \in A} \sum_{j \in B} w_{ij}.$$

and $\bar{A}$ = complement of A.

mincut unfortunately leads to small and often singleton clusters. Therefore some modifications

have been proposed:

- Ratio Cut (Hagen and Kahng, 1992)

$$\text{RatioCut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{k=1}^{k} \frac{W(A_k, \bar{A}_k)}{|A_k|}$$

where

$$|A| = \#\text{ of nodes in } A$$

- Normalized Cut (Shi and Malik, 2000)

$$\text{Ncut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{k=1}^{k} \frac{W(A_k, \bar{A}_k)}{\text{vol}(A_k)}$$

where

$$\text{vol}(A) = \sum_{i \in A} \sum_{j \in V} w_{ij}.$$

Unfortunately, introducing these "balancing" terms causes these problems to be NP-hard.

However, relaxed versions of these problems can be solved by spectral clustering. In particular,

- unnormalized spectral clustering solves a relaxed version of Ratio Cut.
- normalized spectral clustering solves a relaxed

version of Ncut.

Approximating Ratio Cut, K=2

We wish to solve

$$\min_A \text{Ratio Cut}(A,\bar{A}) = \min_A \left[\frac{\text{cut}(A,\bar{A})}{|A|} + \frac{\text{cut}(A,\bar{A})}{|\bar{A}|}\right]$$

Given $A \subseteq \{1,2,\ldots,n\}$, define $f_A = (f_{A1}, \ldots, f_{An})^T$ by

$$f_{Ai} := \begin{cases} \sqrt{|\bar{A}|/|A|} & \text{if } i \in A \\ -\sqrt{|A|/|\bar{A}|} & \text{if } i \notin A. \end{cases}$$

Then

$$\boxed{f_A^T L f_A = n \,\text{Ratio Cut}(A,\bar{A}).}$$

To see this, observe

$$f_A^T L f_A = \frac{1}{2}\sum_{i,j} w_{ij}(f_{Ai} - f_{Aj})^2$$

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

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

$$i \in \bar{A}, j \in A \quad w_{ij} \left(\frac{-}{\sqrt{|\bar{A}|}} - \sqrt{\frac{\cdot}{|A|}} \right)$$

$$= \frac{1}{2} \operatorname{cut}(A, \bar{A}) \left(\frac{|\bar{A}|}{|A|} + \frac{|A|}{|\bar{A}|} + 2 \right)$$

$$+ \frac{1}{2} \operatorname{cut}(A, \bar{A}) \left(\frac{|A|}{|\bar{A}|} + \frac{|\bar{A}|}{|A|} + 2 \right)$$

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

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

$$= n \left[\frac{\operatorname{cut}(A, \bar{A})}{|\bar{A}|} + \frac{\operatorname{cut}(\bar{A}, A)}{|A|} \right]$$

$$= n \operatorname{Ratio Cut}(A, \bar{A}).$$

Furthermore, f_A satisfies

$$\underline{1}^T f_A = \sum_{i=1}^{n} f_{A_i}$$

$$= \sum_{i \in A} \sqrt{\frac{|\bar{A}|}{|A|}} - \sum_{i \notin A} \sqrt{\frac{|A|}{|\bar{A}|}}$$

$$= \sqrt{|A| \cdot |\bar{A}|} - \sqrt{|\bar{A}| \cdot |A|}$$

$$= 0$$

and

and

$$\|f_A\|^2 = \sum_{i=1}^{n} f_{Ai}^2$$

$$= \sum_{i \in A} \frac{|\bar{A}|}{|A|} + \sum_{i \notin A} \frac{|A|}{|\bar{A}|}$$

$$= |\bar{A}| + |A|$$

$$= n.$$

Therefore, RatioCut can be written as the following combinatorial optimization problem:

$$\min_{A \subset \{1,\dots,n\}} \quad f_A^T L f_A$$

$$\text{s.t.} \quad \mathbf{1}^T f_A = 0$$

$$\|f_A\| = \sqrt{n}$$

} It's still RatioCut without these, but we include them to keep the relaxation closer to the original problem

A relaxation of this problem is

$$\min_{f \in \mathbb{R}^n} \quad f^T L f$$

$$\text{s.t.} \quad \mathbf{1}^T f = 0$$

We no longer require $f = f_A$ for some A.

$$\|f\| = \sqrt{n}$$

The solution is an eigenvector of L corresponding to the second smallest eigenvalue of L.

To recover a solution to RatioCut, we can form

$$y_i = (1 \quad f_i)$$

and cluster these using k-means. The clusters determine the estimate of A. Therefore, the relaxation is solved by spectral clustering.

Important note: The gap between the optimal value of RatioCut and the optimal value of its relaxation can be arbitrarily large.

A similar analysis applies to $K > 2$ and Ncut.

Final Remarks

There is yet a third graph Laplacian defined as

$$\tilde{L} = D^{-1/2} L D^{-1/2}$$

It has properties similar to L and $\tilde{L}$, but the spectral clustering algorithm needs to be tweaked.

For a very thorough tutorial on spectral clustering see

U. von Luxburg, "A Tutorial on Spectral Clustering," 2007.

Semi-Supervised Learning

Graph Laplacians also come up in semi-supervised learning.

Consider a regression problem where you have both

- labeled data $(x_i, y_i)_{i=1}^{m}$
- unlabeled data $(x_i)_{i=m+1}^{m+n}$

The unlabeled data can potentially help to improve a regression estimate. If f denotes a regression function and we solve

$$\min_f \quad \frac{1}{2}\sum_{i=1}^{m}(y_i - f(x_i))^2 + \underbrace{\frac{\lambda}{2}\sum_{i,j=m+1}^{m+n} w_{ij}(f(x_i)-f(x_j))^2}_{= \lambda \cdot f_u^T L f_u}$$

$$f_u = [f(x_{m+1}) \cdots f(x_{m+n})]^T$$

where w_{ij} come from a similarity graph defined on the unlabeled data, then the second term regularizes the

solution to be more smooth. This improves the interpolation between the labeled data points.