Clustering of Unsigned and Signed Graphs Using Normalized Graph Cuts (Normalized Cuts 15+ years later) Part I

Jean Gallier

CIS Department
University of Pennsylvania

jean@cis.upenn.edu

September 29, 2015

Special thanks to Jocelyn Quaintance, Joao Cedoc, Jianbo Shi, and Stella Yu.

Special thanks to Jocelyn Quaintance, Joao Cedoc,
Jianbo Shi, and Stella Yu.

Complete details are in

Spectral Theory of Unsigned and Signed Graphs Applications to Graph Clustering: a Survey.

http://www.cis.upenn.edu/~jean/spectral-graph-notes.pdf

Special thanks to Jocelyn Quaintance, Joao Cedoc,
Jianbo Shi, and Stella Yu.

Complete details are in

Spectral Theory of Unsigned and Signed Graphs
Applications to Graph Clustering: a Survey.

http://www.cis.upenn.edu/∼jean/spectral-graph-notes.pdf

Detailed slides in Web page for CIS515:

http://www.cis.upenn.edu/∼cis515/cis515-notes-15.html

Figure 1: Dog Logic

1. Graph Clustering

Given a set of data, the goal of clustering is to *partition* the data into different groups according to their *similarities*.

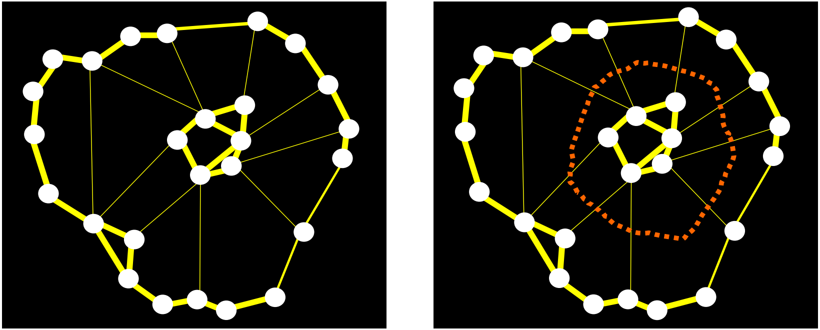


Figure 2: A weighted graph and its partition into two clusters.

When the data is given in terms of a *similarity graph* G, where the *weight* $w_{i\,j}$ between two nodes v_i and v_j is a measure of similarity of v_i and v_j, the problem can be stated as follows:

When the data is given in terms of a *similarity graph* G, where the *weight* w_{ij} between two nodes v_i and v_j is a measure of similarity of v_i and v_j, the problem can be stated as follows:

Find a partition $(A_1, \ldots, A_K)$ of the set of nodes V into different groups such that the *edges between different groups have very low weight* (which indicates that the points in different clusters are dissimilar), and the *edges within a group have high weight* (which indicates that points within the same cluster are similar).

When the data is given in terms of a *similarity graph* G, where the *weight* $w_{i\,j}$ between two nodes v_i and v_j is a measure of similarity of v_i and v_j, the problem can be stated as follows:

Find a partition $(A_1, \ldots, A_K)$ of the set of nodes V into different groups such that the *edges between different groups have very low weight* (which indicates that the points in different clusters are dissimilar), and the *edges within a group have high weight* (which indicates that points within the same cluster are similar).

The above graph clustering problem can be formalized as an optimization problem, using the notion of *cut*.

Typically, the following steps are followed:

Typically, the following steps are followed:

1. Formulate the *discrete optimization problem* in matrix form.

Typically, the following steps are followed:

1. Formulate the *discrete optimization problem* in matrix form.
2. The discrete problem is often very hard (NP-hard, ...). *Relax* the problem (drop some constraints and look for *continuous solutions*).

Typically, the following steps are followed:

1. Formulate the *discrete optimization problem* in matrix form.
2. The discrete problem is often very hard (NP-hard, ...). *Relax* the problem (drop some constraints and look for *continuous solutions*).
3. Find a *discrete solution* as close as possible to a continuous solution.

Typically, the following steps are followed:

1. Formulate the *discrete optimization problem* in matrix form.
2. The discrete problem is often very hard (NP-hard, ...). *Relax* the problem (drop some constraints and look for *continuous solutions*).
3. Find a *discrete solution* as close as possible to a continuous solution.

Step (2) often reduces to some kind of eigenvalue problem.

Typically, the following steps are followed:

1. Formulate the *discrete optimization problem* in matrix form.
2. The discrete problem is often very hard (NP-hard, ...). *Relax* the problem (drop some constraints and look for *continuous solutions*).
3. Find a *discrete solution* as close as possible to a continuous solution.

Step (2) often reduces to some kind of eigenvalue problem.

Step (3) is usually the hardest step.

2. Weigted Graphs, Cuts, Laplacians

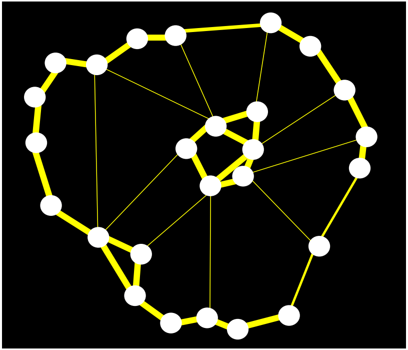


Figure 3: A weighted graph.

The thickness of an edge corresponds to the magnitude of its weight.

Given the weight matrix

$$W = \begin{pmatrix} 0 & 3 & 6 & 3 \\ 3 & 0 & 0 & 3 \\ 6 & 0 & 0 & 3 \\ 3 & 3 & 3 & 0 \end{pmatrix},$$

Given the weight matrix

$$W = \begin{pmatrix} 0 & 3 & 6 & 3 \\ 3 & 0 & 0 & 3 \\ 6 & 0 & 0 & 3 \\ 3 & 3 & 3 & 0 \end{pmatrix},$$

the corresponding graph G is:

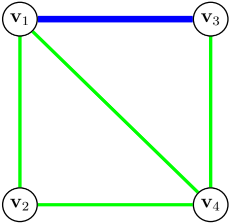


Figure 4: The weighted graph corresponding to W.

Definition 1

A *weighted graph* is a pair $G = (V, W)$, where $V = \{v_1, \ldots, v_m\}$ is a set of *nodes* or *vertices*, and W is a symmetric matrix called the *weight matrix*, such that $w_{i\,j} \geq 0$ for all $i, j \in \{1, \ldots, m\}$, and $w_{i\,i} = 0$ for $i = 1, \ldots, m$. We say that a set $\{v_i, v_j\}$ is an *edge* iff $w_{i\,j} > 0$. The corresponding (undirected) graph (V, E) with $E = \{\{v_i, v_j\} \mid w_{i\,j} > 0\}$, is called the *underlying graph* of G.

Definition 1

A *weighted graph* is a pair $G = (V, W)$, where $V = \{v_1, \ldots, v_m\}$ is a set of *nodes* or *vertices*, and W is a symmetric matrix called the *weight matrix*, such that $w_{i\,j} \geq 0$ for all $i, j \in \{1, \ldots, m\}$, and $w_{i\,i} = 0$ for $i = 1, \ldots, m$. We say that a set $\{v_i, v_j\}$ is an *edge* iff $w_{i\,j} > 0$. The corresponding (undirected) graph (V, E) with $E = \{\{v_i, v_j\} \mid w_{i\,j} > 0\}$, is called the *underlying graph* of G.

We can think of the weight $w_{i\,j}$ of an edge $\{v_i, v_j\}$ as a degree of similarity (or affinity) in an image, or a cost in a network.

For every node $v_i \in V$, the *degree* $d(v_i)$ of v_i is the sum of the weights of the edges adjacent to v_i:

$$d(v_i) = \sum_{j=1}^{m} w_{i\,j}.$$

For every node $v_i \in V$, the *degree* $d(v_i)$ of v_i is the sum of the weights of the edges adjacent to v_i:

$$d(v_i) = \sum_{j=1}^{m} w_{i\,j}.$$

Note that in the above sum, only nodes v_j such that there is an edge $\{v_i, v_j\}$ have a nonzero contribution. Such nodes are said to be *adjacent* to v_i.

For every node $v_i \in V$, the *degree* $d(v_i)$ of v_i is the sum of the weights of the edges adjacent to v_i:

$$d(v_i) = \sum_{j=1}^{m} w_{i\,j}.$$

Note that in the above sum, only nodes v_j such that there is an edge $\{v_i, v_j\}$ have a nonzero contribution. Such nodes are said to be *adjacent* to v_i.

The *degree matrix* D is defined by $D = \mathrm{diag}(d(v_1), \ldots, d(v_m))$.

Given any subset of nodes $A \subseteq V$, we define the *volume* $\mathrm{vol}(A)$ of A as the sum of the weights of all edges adjacent to nodes in A:

$$\mathrm{vol}(A) = \sum_{v_i \in A} d(v_i) = \sum_{v_i \in A} \sum_{j=1}^{m} w_{i\,j}.$$

Given any subset of nodes $A \subseteq V$, we define the *volume* $\mathrm{vol}(A)$ of A as the sum of the weights of all edges adjacent to nodes in A:

$$\mathrm{vol}(A) = \sum_{v_i \in A} d(v_i) = \sum_{v_i \in A} \sum_{j=1}^{m} w_{i\,j}.$$

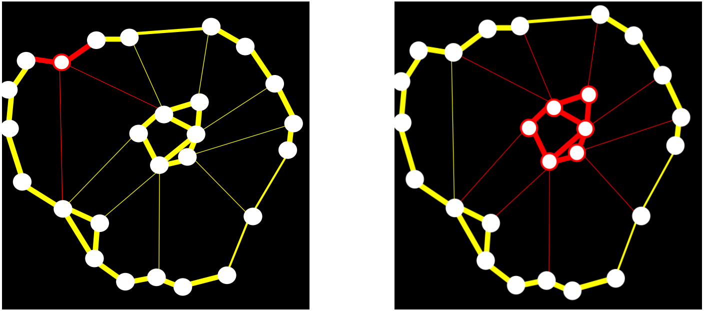


Figure 5: Degree and volume.

Observe that $\mathrm{vol}(A) = 0$ if A consists of isolated vertices ($w_{i\,j} = 0$ for all $v_i \in A$). Thus, it is best to assume that G does not have isolated vertices.

Observe that $\mathrm{vol}(A) = 0$ if A consists of isolated vertices ($w_{i\,j} = 0$ for all $v_i \in A$). Thus, it is best to assume that G does not have isolated vertices.

Given any two subset $A, B \subseteq V$ (not necessarily distinct), we define $\mathrm{links}(A, B)$ by

$$\mathrm{links}(A, B) = \sum_{v_i \in A, v_j \in B} w_{i\,j}.$$

Observe that $\mathrm{vol}(A) = 0$ if A consists of isolated vertices ($w_{i\,j} = 0$ for all $v_i \in A$). Thus, it is best to assume that G does not have isolated vertices.

Given any two subset $A, B \subseteq V$ (not necessarily distinct), we define $\mathrm{links}(A, B)$ by

$$\mathrm{links}(A, B) = \sum_{v_i \in A, v_j \in B} w_{i\,j}.$$

Since the matrix W is symmetric, we have

$$\mathrm{links}(A, B) = \mathrm{links}(B, A).$$

The quantity $\mathrm{links}(A, \overline{A}) = \mathrm{links}(\overline{A}, A)$, where $\overline{A} = V - A$ denotes the complement of A in V, *measures how many links escape from A* (and $\overline{A}$), and the quantity $\mathrm{links}(A, A)$ *measures how many links stay within A itself*.

The quantity $\mathrm{links}(A, \overline{A}) = \mathrm{links}(\overline{A}, A)$, where $\overline{A} = V - A$ denotes the complement of A in V, *measures how many links escape from A* (and $\overline{A}$), and the quantity $\mathrm{links}(A, A)$ *measures how many links stay within A itself*.

The quantity

$$\mathrm{cut}(A) = \mathrm{links}(A, \overline{A})$$

is often called the *cut* of A.

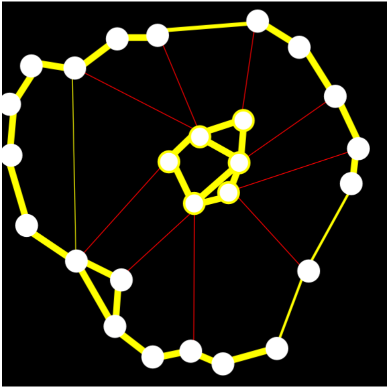


Figure 6: A Cut involving the set of nodes in the center and the nodes on the perimeter.

We now define the most important concept of this talk: The *Laplacian matrix* of a graph. Actually, as we will see, it comes in several flavors.

We now define the most important concept of this talk: The *Laplacian matrix* of a graph. Actually, as we will see, it comes in several flavors.

Definition 2

Given any weighted graph $G = (V, W)$ with $V = \{v_1, \ldots, v_m\}$, the *(unnormalized) graph Laplacian $L(G)$ of G* is defined by

$$L(G) = D(G) - W,$$

where $D(G) = \mathrm{diag}(d_1, \ldots, d_m)$ is the degree matrix of G (a diagonal matrix), with

$$d_i = \sum_{j=1}^{m} w_{i\,j}.$$

Consider the weight matrix

$$W = \begin{pmatrix} 0 & 3 & 6 & 3 \\ 3 & 0 & 0 & 3 \\ 6 & 0 & 0 & 3 \\ 3 & 3 & 3 & 0 \end{pmatrix},$$

Consider the weight matrix

$$W = \begin{pmatrix} 0 & 3 & 6 & 3 \\ 3 & 0 & 0 & 3 \\ 6 & 0 & 0 & 3 \\ 3 & 3 & 3 & 0 \end{pmatrix},$$

and the corresponding graph G

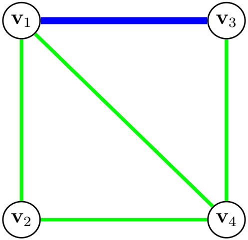


Figure 7: The weighted graph corresponding to W.

The degree matrix (`diag(sum(W))`) is

$$D(G) = \begin{pmatrix} 12 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 9 \end{pmatrix},$$

The degree matrix (`diag(sum(W))`) is

$$D(G) = \begin{pmatrix} 12 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 9 \end{pmatrix},$$

and the Laplacian is

$$L = D(G) - W = \begin{pmatrix} 12 & -3 & -6 & -3 \\ -3 & 6 & 0 & -3 \\ -6 & 0 & 9 & -3 \\ -3 & -3 & -3 & 9 \end{pmatrix}.$$

The degree matrix (`diag(sum(W))`) is

$$D(G) = \begin{pmatrix} 12 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 9 \end{pmatrix},$$

and the Laplacian is

$$L = D(G) - W = \begin{pmatrix} 12 & -3 & -6 & -3 \\ -3 & 6 & 0 & -3 \\ -6 & 0 & 9 & -3 \\ -3 & -3 & -3 & 9 \end{pmatrix}.$$

The eigenvalues of L are: $0, 6.8038, 12, 17.1962$.

The vector $\mathbf{1}$ is the nullspace of L, but it is less obvious that L is positive semidefinite.

The vector $\mathbf{1}$ is the nullspace of L, but it is less obvious that L is positive semidefinite.

Proposition 1

For any $m \times m$ symmetric matrix W, if we let $L = D - W$ where D is the degree matrix of $W = (w_{ij})$, then we have

$$x^\top L x = \frac{1}{2} \sum_{i,j=1}^{m} w_{i\,j}(x_i - x_j)^2 \quad \text{for all } x \in \mathbb{R}^m.$$

Consequently, $x^\top L x$ does not depend on the diagonal entries in W, and if $w_{i\,j} \geq 0$ for all $i, j \in \{1, \ldots, m\}$, then L is positive semidefinite.

Proposition 1 immediately implies the following facts: For any weighted graph $G = (V, W)$, Proposition 1 immediately implies the following facts: For any weighted graph $G = (V, W)$,

1. The eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_m$ of L are real and nonnegative, and there is an orthonormal basis of eigenvectors of L.
2. The smallest eigenvalue λ_1 of L is equal to 0, and $\mathbf{1}$ is a corresponding eigenvector.

Normalized variants of the graph Laplacian are needed, especially in applications to graph clustering.

These variants make sense only if G has no isolated vertices. In this case, the degree matrix D contains positive entries, so it is invertible and $D^{-1/2}$ makes sense; namely

$$D^{-1/2} = \mathrm{diag}(d_1^{-1/2}, \ldots, d_m^{-1/2}).$$

These variants make sense only if G has no isolated vertices. In this case, the degree matrix D contains positive entries, so it is invertible and $D^{-1/2}$ makes sense; namely

$$D^{-1/2} = \mathrm{diag}(d_1^{-1/2}, \ldots, d_m^{-1/2}).$$

Definition 3

Given any weighted directed graph $G = (V, W)$ with no isolated vertex and with $V = \{v_1, \ldots, v_m\}$, the *(normalized) graph Laplacians L_{sym} and L_{rw} of G* are defined by

$$\begin{aligned} L_{\mathrm{sym}} &= D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2} \\ L_{\mathrm{rw}} &= D^{-1} L = I - D^{-1} W. \end{aligned}$$

Proposition 2

Let $G = (V, W)$ be a weighted graph without isolated vertices. The graph Laplacians, L, L_{sym}, and L_{rw} satisfy the following properties:

(1) *The normalized graph Laplacians L_{sym} and L_{rw} have the same spectrum $(0 = \nu_1 \leq \nu_2 \leq \ldots \leq \nu_m \leq 2)$, and a vector $u \neq 0$ is an eigenvector of L_{rw} for λ iff $D^{1/2}u$ is an eigenvector of L_{sym} for λ.*

(2) *The graph Laplacians, L, L_{sym}, and L_{rw} are symmetric, positive, semidefinite.*

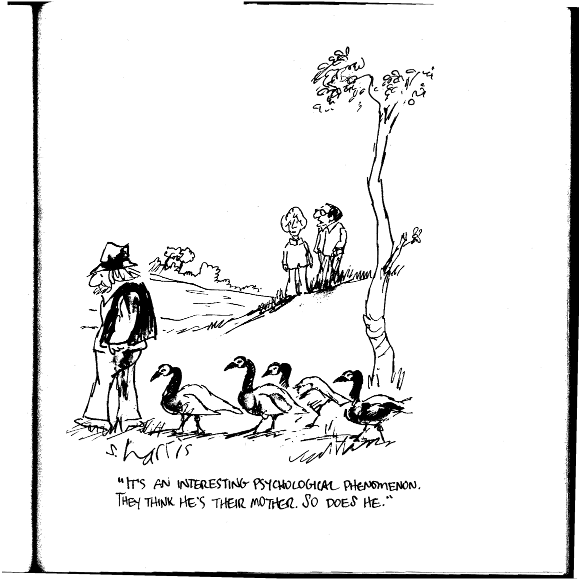


Figure 8: Are you my mother?

3. Back to Graph Clustering

If we want to partition V into K clusters, we can do so by finding a partition $(A_1, \ldots, A_K)$ that minimizes the quantity

$$\mathrm{cut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{1=1}^{K} \mathrm{cut}(A_i).$$

3. Back to Graph Clustering

If we want to partition V into K clusters, we can do so by finding a partition $(A_1, \ldots, A_K)$ that minimizes the quantity

$$\mathrm{cut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{1=1}^{K} \mathrm{cut}(A_i).$$

For $K = 2$, the mincut problem is a classical problem that can be solved efficiently, but in practice, it does not yield satisfactory partitions.

3. Back to Graph Clustering

If we want to partition V into K clusters, we can do so by finding a partition $(A_1, \ldots, A_K)$ that minimizes the quantity

$$\mathrm{cut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{1=1}^{K} \mathrm{cut}(A_i).$$

For $K = 2$, the mincut problem is a classical problem that can be solved efficiently, but in practice, it does not yield satisfactory partitions.

Indeed, in many cases, the mincut solution separates one vertex from the rest of the graph. What we need is to design our cost function in such a way that it keeps the subsets A_i "reasonably large" (reasonably balanced).

A way to get around this problem is to normalize the cuts by *dividing by some measure of each subset A_i*.

A way to get around this problem is to normalize the cuts by *dividing by some measure of each subset A_i*.

One possibility is to use the size (the number of elements) of A_i.

A way to get around this problem is to normalize the cuts by *dividing by some measure of each subset A_i*.

One possibility is to use the size (the number of elements) of A_i.

Another is to use the *volume* $\mathrm{vol}(A_i)$ of A_i. A solution using the second measure (the volume) (for $K = 2$) was proposed and investigated in a seminal paper of Shi and Malik.

A way to get around this problem is to normalize the cuts by *dividing by some measure of each subset A_i*.

One possibility is to use the size (the number of elements) of A_i.

Another is to use the *volume* $\mathrm{vol}(A_i)$ of A_i. A solution using the second measure (the volume) (for $K = 2$) was proposed and investigated in a seminal paper of Shi and Malik.

Subsequently, Stella Yu (in her dissertation) and Yu and Shi extended the method to $K > 2$ clusters.

The idea is to minimize the cost function

$$\mathrm{Ncut}(A_1, \ldots, A_K) = \sum_{i=1}^{K} \frac{\mathrm{links}(A_i, \overline{A_i})}{\mathrm{vol}(A_i)} = \sum_{i=1}^{K} \frac{\mathrm{cut}(A_i, \overline{A_i})}{\mathrm{vol}(A_i)}.$$

The idea is to minimize the cost function

$$\mathrm{Ncut}(A_1, \ldots, A_K) = \sum_{i=1}^{K} \frac{\mathrm{links}(A_i, \overline{A_i})}{\mathrm{vol}(A_i)} = \sum_{i=1}^{K} \frac{\mathrm{cut}(A_i, \overline{A_i})}{\mathrm{vol}(A_i)}.$$

We proceed directly to the case $K > 2$ which is the most interesting case, and is harder to handle.

Figure 9: Newton goes to Wharton

4. K-Way Clustering Using Normalized Cuts

Two crucial issues need to be addressed:

4. K-Way Clustering Using Normalized Cuts

Two crucial issues need to be addressed:

1. The choice of a matrix representation for partitions on the set of vertices.

4. K-Way Clustering Using Normalized Cuts

Two crucial issues need to be addressed:

1. The choice of a matrix representation for partitions on the set of vertices.

   It is important that such a representation be *scale-invariant*.

4. K-Way Clustering Using Normalized Cuts

Two crucial issues need to be addressed:

1. The choice of a matrix representation for partitions on the set of vertices.

   It is important that such a representation be *scale-invariant*.

   It is also necessary to state necessary and sufficient conditions for such matrices to represent a partition.

4. K-Way Clustering Using Normalized Cuts

Two crucial issues need to be addressed:

1. The choice of a matrix representation for partitions on the set of vertices.

   It is important that such a representation be *scale-invariant*.

   It is also necessary to state necessary and sufficient conditions for such matrices to represent a partition.

2. The choice of a *metric* to compare solutions.

We describe a partition $(A_1, \ldots, A_K)$ of the set of nodes V by an $N \times K$ matrix $X = [X^1 \cdots X^K]$ whose columns $X^1, \ldots, X^K$ are indicator vectors of the partition $(A_1, \ldots, A_K)$.

We describe a partition $(A_1, \ldots, A_K)$ of the set of nodes V by an $N \times K$ matrix $X = [X^1 \cdots X^K]$ whose columns $X^1, \ldots, X^K$ are indicator vectors of the partition $(A_1, \ldots, A_K)$.

We assume that the vector X^j is of the form

$$X^j = (x_1^j, \ldots, x_N^j),$$

where $x_i^j \in \{a_j, 0\}$ for $j = 1, \ldots, K$ and $i = 1, \ldots, N$, and with $a_j \neq 0$.

When $N = 10$ and $K = 4$, an example of a matrix X representing the partition of $V = \{v_1, v_2, \ldots, v_{10}\}$ into the four blocks

$$\{A_1, A_2, A_3, A_4\} = \{\{v_2, v_4, v_6\}, \{v_1, v_5\}, \{v_3, v_8, v_{10}\}, \{v_7, v_9\}\},$$

is shown below:

When $N = 10$ and $K = 4$, an example of a matrix X representing the partition of $V = \{v_1, v_2, \ldots, v_{10}\}$ into the four blocks

$$\{A_1, A_2, A_3, A_4\} = \{\{v_2, v_4, v_6\}, \{v_1, v_5\}, \{v_3, v_8, v_{10}\}, \{v_7, v_9\}\},$$

is shown below:

$$X = \begin{pmatrix} 0 & a_2 & 0 & 0 \\ a_1 & 0 & 0 & 0 \\ 0 & 0 & a_3 & 0 \\ a_1 & 0 & 0 & 0 \\ 0 & a_2 & 0 & 0 \\ a_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_4 \\ 0 & 0 & a_3 & 0 \\ 0 & 0 & 0 & a_4 \\ 0 & 0 & a_3 & 0 \end{pmatrix}.$$

Let $d = \mathbf{1}^\top D\mathbf{1}$ and $\alpha_j = \mathrm{vol}(A_j)$, so that $\alpha_1 + \cdots + \alpha_K = d$.

Let $d = \mathbf{1}^\top D \mathbf{1}$ and $\alpha_j = \mathrm{vol}(A_j)$, so that $\alpha_1 + \cdots + \alpha_K = d$.

Then, $\mathrm{vol}(\overline{A_j}) = d - \alpha_j$, and we have

$$
\begin{aligned}
(X^j)^\top L X^j &= a_j^2 \,\mathrm{cut}(A_j, \overline{A_j}),\\
(X^j)^\top D X^j &= \alpha_j a_j^2,
\end{aligned}
$$

Let $d = \mathbf{1}^\top D \mathbf{1}$ and $\alpha_j = \mathrm{vol}(A_j)$, so that $\alpha_1 + \cdots + \alpha_K = d$.

Then, $\mathrm{vol}(\overline{A_j}) = d - \alpha_j$, and we have

$$\begin{aligned} (X^j)^\top L X^j &= a_j^2 \,\mathrm{cut}(A_j, \overline{A_j}), \\ (X^j)^\top D X^j &= \alpha_j a_j^2, \end{aligned}$$

so

$$\frac{\mathrm{cut}(A_j, \overline{A_j})}{\mathrm{vol}(A_j)} = \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \quad j = 1, \ldots, K.$$

Let $d = \mathbf{1}^\top D\mathbf{1}$ and $\alpha_j = \mathrm{vol}(A_j)$, so that $\alpha_1 + \cdots + \alpha_K = d$.

Then, $\mathrm{vol}(\overline{A_j}) = d - \alpha_j$, and we have

$$(X^j)^\top L X^j = a_j^2 \,\mathrm{cut}(A_j, \overline{A_j}),$$
$$(X^j)^\top D X^j = \alpha_j a_j^2,$$

so

$$\frac{\mathrm{cut}(A_j, \overline{A_j})}{\mathrm{vol}(A_j)} = \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \quad j = 1, \ldots, K.$$

von Luxburg and Yu and Shi pick

$$a_j = \frac{1}{\sqrt{\alpha_j}} = \frac{1}{\sqrt{\mathrm{vol}(A_j)}}, \quad j = 1, \ldots, K.$$

If we let

$$\mathcal{X} = \left\{ [X^1 \ \ldots \ X^K] \mid X^j = a_j(x_1^j, \ldots, x_N^j),\ x_i^j \in \{1, 0\}, a_j \in \mathbb{R},\ X^j \neq 0 \right\}$$

then our optimization problem is:

If we let

$$\mathcal{X} = \Big\{[X^1 \ \dots \ X^K] \mid X^j = a_j(x_1^j, \dots, x_N^j),\ x_i^j \in \{1, 0\}, a_j \in \mathbb{R},\ X^j \neq 0\Big\}$$

then our optimization problem is:

***K*-way Clustering of a graph using Normalized Cut, Version 1: Problem PNC1**

$$\begin{aligned}
&\text{minimize} && \sum_{j=1}^{K} \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \\
&\text{subject to} && (X^i)^\top D X^j = 0, \quad 1 \leq i, j \leq K,\ i \neq j, \\
& && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}, \qquad X \in \mathcal{X}.
\end{aligned}$$

The solutions that we are seeking are K-tuples $(\mathbb{P}(X^1), \ldots, \mathbb{P}(X^K))$ of points in $\mathbb{RP}^{N-1}$ determined by their homogeneous coordinates $X^1, \ldots, X^K$.

The solutions that we are seeking are K-tuples $(\mathbb{P}(X^1), \ldots, \mathbb{P}(X^K))$ of points in $\mathbb{RP}^{N-1}$ determined by their homogeneous coordinates $X^1, \ldots, X^K$.

Our original formulation (PNC1) can be converted to a more convenient form, by chasing the denominators in the Rayleigh ratios, and by expressing the objective function in terms of the *trace* of a certain matrix.

K-way Clustering of a graph using Normalized Cut, Version 1: Problem PNC1

$$
\begin{aligned}
&\text{minimize} && \sum_{j=1}^{K} \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \\
&\text{subject to} && (X^i)^\top D X^j = 0, \quad 1 \leq i, j \leq K,\ i \neq j, \\
& && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}, \qquad\qquad X \in \mathcal{X}.
\end{aligned}
$$

K-way Clustering of a graph using Normalized Cut, Version 1: Problem PNC1

$$
\begin{aligned}
&\text{minimize} && \sum_{j=1}^{K} \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \\
&\text{subject to} && (X^i)^\top D X^j = 0, \quad 1 \leq i, j \leq K,\ i \neq j, \\
& && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}, \qquad X \in \mathcal{X}.
\end{aligned}
$$

K-way Clustering of a graph using Normalized Cut, Version 2: Problem PNC2

$$
\begin{aligned}
&\text{minimize} && \operatorname{tr}(X^\top L X) \\
&\text{subject to} && X^\top D X = I, \\
& && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}, \qquad X \in \mathcal{X}.
\end{aligned}
$$

Problem PNC2 is equivalent to problem PNC1 if we view the solutions as homogeneous coordinates (up to a nonzero scalar).

Problem PNC2 is equivalent to problem PNC1 if we view the solutions as homogeneous coordinates (up to a nonzero scalar).

The main problem in finding a good relaxation of problem PNC2 is that it is very difficult to enforce the condition $X \in \mathcal{X}$.

Problem PNC2 is equivalent to problem PNC1 if we view the solutions as homogeneous coordinates (up to a nonzero scalar).

The main problem in finding a good relaxation of problem PNC2 is that it is very difficult to enforce the condition $X \in \mathcal{X}$.

The first natural relaxation of problem PNC2 is to drop the condition that $X \in \mathcal{X}$, and we obtain

Problem $(*_2)$

$$\begin{aligned} &\text{minimize} && \mathrm{tr}(X^\top L X) \\ &\text{subject to} && X^\top D X = I, \\ & && X(X^\top X)^{-1}X^\top \mathbf{1} = \mathbf{1}. \end{aligned}$$

Problem $(*_2)$

$$
\begin{aligned}
&\text{minimize} && \mathrm{tr}(X^\top L X)\\
&\text{subject to} && X^\top D X = I,\\
& && X(X^\top X)^{-1}X^\top \mathbf{1} = \mathbf{1}.
\end{aligned}
$$

Actually, since the discrete solutions $X \in \mathcal{X}$ that we are ultimately seeking are solutions of problem PNC1, the preferred relaxation is the one obtained from problem PNC1 by dropping the condition $X \in \mathcal{X}$, and simply requiring that $X^j \neq 0$, for $j = 1, \ldots, K$:

Problem $(*_1)$

$$
\begin{aligned}
&\text{minimize} && \sum_{j=1}^{K} \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \\
&\text{subject to} && (X^i)^\top D X^j = 0, X^j \neq 0 \qquad 1 \leq i,j \leq K,\ i \neq j, \\
& && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}.
\end{aligned}
$$

Problem $(*_1)$

$$\begin{aligned} &\text{minimize} && \sum_{j=1}^{K} \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j} \\ &\text{subject to} && (X^i)^\top D X^j = 0, X^j \neq 0 \qquad 1 \leq i, j \leq K,\; i \neq j, \\ & && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}. \end{aligned}$$

Problem $(*_2)$

$$\begin{aligned} &\text{minimize} && \mathrm{tr}(X^\top L X) \\ &\text{subject to} && X^\top D X = I, \\ & && X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}. \end{aligned}$$

Let

$$\mu(X^1, \dots, X^K) = \sum_{j=1}^{K} \frac{(X^j)^\top L X^j}{(X^j)^\top D X^j}.$$

Proposition 3

For any orthogonal $K \times K$ matrix R, any symmetric $N \times N$ matrix A, and any $N \times K$ matrix $X = [X^1 \ \cdots \ X^K]$, the following properties hold:

(1) $\mu(X) = \mathrm{tr}(\Lambda^{-1} X^\top L X)$, *where*

$$\Lambda = \mathrm{diag}((X^1)^\top D X^1, \ldots, (X^K)^\top D X^K).$$

(2) *If* $(X^1)^\top D X^1 = \cdots = (X^K)^\top D X^K = \alpha^2$, *then*

$$\mu(X) = \mu(XR) = \frac{1}{\alpha^2} \mathrm{tr}(X^\top L X).$$

(3) *The condition* $X^\top A X = \alpha^2 I$ *is preserved if* X *is replaced by* XR.

(4) *The condition* $X(X^\top X)^{-1} X^\top \mathbf{1} = \mathbf{1}$ *is preserved if* X *is replaced by* XR.

Every solution Z of problem $(*_2)$ yields a *family of solutions* of problem $(*_1)$; namely, all matrices of the form $ZR\Lambda$, where $R \in \mathbf{O}(K)$ and Λ is a diagonal invertible matrix.

Every solution Z of problem $(*_2)$ yields a *family of solutions* of problem $(*_1)$; namely, all matrices of the form $Z R \Lambda$, where $R \in \mathbf{O}(K)$ and Λ is a diagonal invertible matrix.

We will take advantage of this fact in looking for a discrete solution X "close" to a solution Z of the relaxed problem $(*_2)$.

Every solution Z of problem $(*_2)$ yields a *family of solutions* of problem $(*_1)$; namely, all matrices of the form $ZR\Lambda$, where $R \in \mathbf{O}(K)$ and Λ is a diagonal invertible matrix.

We will take advantage of this fact in looking for a discrete solution X "close" to a solution Z of the relaxed problem $(*_2)$.

Observe that a matrix is of the form $R\Lambda$ with $R \in \mathbf{O}(K)$ and Λ a diagonal invertible matrix iff its columns are nonzero and pairwise orthogonal.

If we make the change of variable $Y = D^{1/2}X$ or equivalently $X = D^{-1/2}Y$, we get

If we make the change of variable $Y = D^{1/2}X$ or equivalently $X = D^{-1/2}Y$, we get

Problem $(**_2)$

$$\begin{aligned} &\text{minimize} && \mathrm{tr}(Y^\top D^{-1/2} L D^{-1/2} Y) \\ &\text{subject to} && Y^\top Y = I, \\ & && YY^\top D^{1/2}\mathbf{1} = D^{1/2}\mathbf{1}. \end{aligned}$$

If we make the change of variable $Y = D^{1/2}X$ or equivalently $X = D^{-1/2}Y$, we get

Problem $(**_2)$

$$\begin{aligned} &\text{minimize} && \mathrm{tr}(Y^\top D^{-1/2} L D^{-1/2} Y) \\ &\text{subject to} && Y^\top Y = I, \\ & && YY^\top D^{1/2}\mathbf{1} = D^{1/2}\mathbf{1}. \end{aligned}$$

We pass from a solution Y of problem $(**_2)$ to a solution Z of problem $(*_2)$ by $Z = D^{-1/2}Y$.

It is not a priori obvious that the minimum of $\mathrm{tr}(Y^\top L_{\mathrm{sym}} Y)$ over all $N \times K$ matrices Y satisfying $Y^\top Y = I$ is equal to the sum $\nu_1 + \cdots + \nu_K$ of the first K eigenvalues of $L_{\mathrm{sym}} = D^{-1/2} L D^{-1/2}$.

It is not a priori obvious that the minimum of $\mathrm{tr}(Y^\top L_{\mathrm{sym}} Y)$ over all $N \times K$ matrices Y satisfying $Y^\top Y = I$ is equal to the sum $\nu_1 + \cdots + \nu_K$ of the first K eigenvalues of $L_{\mathrm{sym}} = D^{-1/2} L D^{-1/2}$.

Fortunately, the Poincaré separation theorem guarantees that the sum of the K smallest eigenvalues of L_{sym} is a lower bound for $\mathrm{tr}(Y^\top L_{\mathrm{sym}} Y)$.

It is not a priori obvious that the minimum of $\mathrm{tr}(Y^\top L_{\mathrm{sym}} Y)$ over all $N \times K$ matrices Y satisfying $Y^\top Y = I$ is equal to the sum $\nu_1 + \cdots + \nu_K$ of the first K eigenvalues of $L_{\mathrm{sym}} = D^{-1/2} L D^{-1/2}$.

Fortunately, the Poincaré separation theorem guarantees that the sum of the K smallest eigenvalues of L_{sym} is a lower bound for $\mathrm{tr}(Y^\top L_{\mathrm{sym}} Y)$.

Furthermore, if we temporarily ignore the second constraint, the minimum of problem $(**_2)$ is achieved by any K unit eigenvectors $(u_1, \ldots, u_K)$ associated with the smallest eigenvalues

$$0 = \nu_1 \leq \nu_2 \leq \ldots \leq \nu_K$$

of L_{sym}.

We may assume that $\nu_2 > 0$, namely that the underlying graph is connected (otherwise, we work with each connected component), in which case $Y^1 = D^{1/2}\mathbf{1}/\left\|D^{1/2}\mathbf{1}\right\|_2$, because $\mathbf{1}$ is in the nullspace of L.

We may assume that $\nu_2 > 0$, namely that the underlying graph is connected (otherwise, we work with each connected component), in which case $Y^1 = D^{1/2}\mathbf{1}/\left\|D^{1/2}\mathbf{1}\right\|_2$, because $\mathbf{1}$ is in the nullspace of L.

Then, $Z = D^{-1/2}Y$ with $Y = [u_1 \ \ldots \ u_K]$ yields a minimum of our relaxed problem $(*_1)$ (the second constraint is satisfied because $\mathbf{1}$ is in the range of Z).

Figure 10: Try and try again

The conditions $(Z^i)^\top DZ^j = 0$ do not necessarily imply that Z^i and Z^j are orthogonal (w.r.t. the Euclidean inner product), but we can obtain a solution of Problems $(*_2)$ and $(*_1)$ achieving the same minimum for which distinct columns Z^i and Z^j are simultaneously orthogonal and D-orthogonal, by multiplying Z by some $K \times K$ orthogonal matrix R on the right.

Indeed, if Z is a solution of $(*_2)$ obtained as above, the $K \times K$ symmetric matrix $Z^\top Z$ can be diagonalized by some orthogonal $K \times K$ matrix R as

$$Z^\top Z = R\Sigma R^\top,$$

where Σ is a diagonal matrix,

Indeed, if Z is a solution of $(*_2)$ obtained as above, the $K \times K$ symmetric matrix $Z^\top Z$ can be diagonalized by some orthogonal $K \times K$ matrix R as

$$Z^\top Z = R\Sigma R^\top,$$

where Σ is a diagonal matrix,

and thus,

$$R^\top Z^\top ZR = (ZR)^\top ZR = \Sigma,$$

which shows that the columns of ZR are orthogonal.

6. What if the weight matrix is very large?

Given a $n \times n$ matrix W for $n \approx 10^5$, we want to find to compute a rank-k approximation, with $k \ll n$ (where $k \sim$ the number of clusters),

$$\underset{n\times n}{W} \approx \underset{n\times k}{E} \; \underset{k\times n}{F^{\top}}.$$

6. What if the weight matrix is very large?

Given a $n \times n$ matrix W for $n \approx 10^5$, we want to find to compute a rank-k approximation, with $k \ll n$ (where $k \sim$ the number of clusters),

$$\underset{n \times n}{W} \approx \underset{n \times k}{E} \quad \underset{k \times n}{F^\top}.$$

The columns of E and F are required to be orthogonal.

6. What if the weight matrix is very large?

Given a $n \times n$ matrix W for $n \approx 10^5$, we want to find to compute a rank-k approximation, with $k \ll n$ (where $k \sim$ the number of clusters),

$$\underset{n \times n}{W} \approx \underset{n \times k}{E} \; \underset{k \times n}{F^\top}.$$

The columns of E and F are required to be orthogonal.

This problem requires algorithms for computing the *Singular Value Decomposition (SVD)*.

What if the weight matrix is very large?

We use *randomized algorithms* that compute partial matrix decompositions (Halko, Martinsson, Tropp).

What if the weight matrix is very large?

We use *randomized algorithms* that compute partial matrix decompositions (Halko, Martinsson, Tropp).

- For a dense input matrix, randomized algorithms require $O(mn\log(k))$ floating-point operations in contrast with $O(mnk)$ for classical algorithms.
- randomized techniques require only a constant number of passes over the data.
- probabilistic bound on accuracy.

What if the weight matrix is very large?

We use *randomized algorithms* that compute partial matrix decompositions (Halko, Martinsson, Tropp).

- For a dense input matrix, randomized algorithms require $O(mn\log(k))$ floating-point operations in contrast with $O(mnk)$ for classical algorithms.
- randomized techniques require only a constant number of passes over the data.
- probabilistic bound on accuracy.

We want to find $A \approx U\Sigma_k V^\top$.

Fast SVD

Given an $m \times n$ matrix A and integers ℓ and q, this algorithm computes an $m \times \ell$ orthonormal matrix Q whose range approximates the range of A.

Fast SVD

Given an $m \times n$ matrix A and integers ℓ and q, this algorithm computes an $m \times \ell$ orthonormal matrix Q whose range approximates the range of A.

Algorithm 2 Computing SVD using Randomized Algorithm

1: Draw an $n \times \ell$ Gaussian random matrix Ω.
2: Form the $m \times \ell$ matrix $Y = (AA^\top)^q A\Omega$ via alternating application of A and $A^\top$.
3: Construct an $m \times \ell$ matrix Q whose columns form an orthonormal basis for the range of Y, e.g., via the QR factorization $Y = QR$.
4: Form $B = Q^\top A$.
5: Compute an SVD of the small matrix: $B = \tilde{U}\Sigma V^\top$.
6: Set $U = Q\tilde{U}$.

Probabilistic Bounds

What is the error $e_k = \left\| A - U\Sigma_k V^\top \right\|$?

Probabilistic Bounds

What is the error $e_k = \left\| A - U\Sigma_k V^\top \right\|$?

Theorem 4

Eckart-Young Theorem: *e_k is bounded from below by the $(k+1)$th singular value σ_{k+1} of A.*

Probabilistic Bounds

What is the error $e_k = \left\|A - U\Sigma_k V^\top\right\|$?

Theorem 4

Eckart-Young Theorem: *e_k is bounded from below by the $(k+1)th$ singular value σ_{k+1} of A.*

We want e_k to be close to σ_{k+1}, but this is not true.
The expectation of $\frac{e_k}{\sigma_{k+1}}$ is large with high variance.

Probabilistic Bounds

What is the error $e_k = \left\| A - U\Sigma_k V^\top \right\|$?

Theorem 4

Eckart-Young Theorem: *e_k is bounded from below by the $(k+1)$th singular value σ_{k+1} of A.*

We want e_k to be close to σ_{k+1}, but this is not true.
The expectation of $\frac{e_k}{\sigma_{k+1}}$ is large with high variance.

However, using oversampling where we compute $k + p$ where $p = k$ solves this issue.

Probabilistic Bounds

Theorem 5

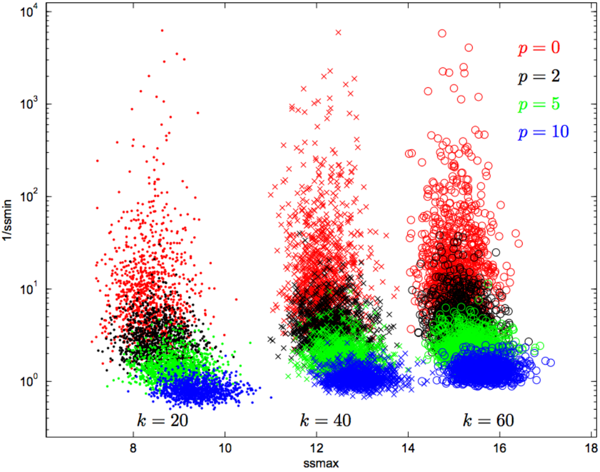
Suppose that A is a real $m \times n$ matrix. Select an exponent q and a target number k of singular vectors, where $2 \leq k \leq 0.5 \min\{m, n\}$. Randomized SVD algorithm to obtain a rank-2k factorization $U\Sigma V^\top$. Then

$$\mathbb{E}\left[\|A - U\Sigma_k V^\top\|\right] \leq \left[1 + 4\sqrt{\frac{2\min\{m,n\}}{k-1}}\right]^{1/(2q+1)} \sigma_{k+1},$$

where $\mathbb{E}$ denotes expectation with respect to the random test matrix and σ_{k+1} is the $(k+1)$th singular value of A. (Halko, Martinsson, Tropp 2011)

Bound Example

Scatter plot showing distribution of $k \times (k + p)$ Gaussian matrices.



$1/\sigma_{\min}$ *is plotted against* $\sigma_{\max}$.

7. Signed Graphs

Intuitively, in a weighted graph, an edge with a positive weight denotes similarity or proximity of its endpoints.

7. Signed Graphs

Intuitively, in a weighted graph, an edge with a positive weight denotes similarity or proximity of its endpoints.

For many reasons, it is desirable to allow edges labeled with *negative weights*, the intuition being that *a negative weight indicates dissimilarity or distance*.

7. Signed Graphs

Intuitively, in a weighted graph, an edge with a positive weight denotes similarity or proximity of its endpoints.

For many reasons, it is desirable to allow edges labeled with *negative weights*, the intuition being that *a negative weight indicates dissimilarity or distance*.

Weighted graphs for which the weight matrix is a symmetric matrix in which negative and positive entries are allowed are called *signed graphs*.

Given the signed matrix

$$W = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & -1 & 1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & -1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & -1 & 0 & 1 & 0 & -1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & -1 & -1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & -1 & 0 \end{pmatrix}$$

the corresponding signed graph is

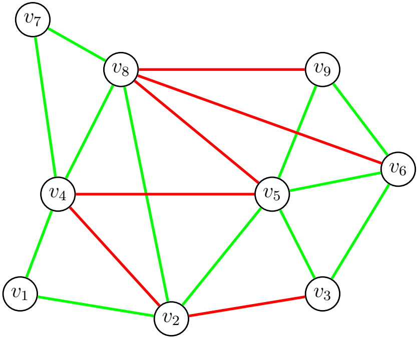


Figure 11: A signed graph G.

The first obstacle is that the degree matrix may now contain *zero or negative entries*.

The first obstacle is that the degree matrix may now contain *zero or negative entries*.

As a consequence, the Laplacian L may no longer be positive semidefinite, and worse, $D^{-1/2}$ may not exist.

The first obstacle is that the degree matrix may now contain *zero or negative entries*.

As a consequence, the Laplacian L may no longer be positive semidefinite, and worse, $D^{-1/2}$ may not exist.

A simple remedy is to use the *absolute values of the weights* in the degree matrix!

The first obstacle is that the degree matrix may now contain *zero or negative entries*.

As a consequence, the Laplacian L may no longer be positive semidefinite, and worse, $D^{-1/2}$ may not exist.

A simple remedy is to use the *absolute values of the weights* in the degree matrix!

This idea applied to signed graph with weights $(-1, 0, 1)$ occurs in Hou. Kolluri, Shewchuk and O'Brien take the natural step of using absolute values of weights in the degree matrix in their original work on surface reconstruction from noisy point clouds.

Kunegis et al. appear to be the first to make a systematic study of spectral methods applied to signed graphs.

Kunegis et al. appear to be the first to make a systematic study of spectral methods applied to signed graphs.

However, it should be noted that only 2-clustering is considered in the above papers.

Kunegis et al. appear to be the first to make a systematic study of spectral methods applied to signed graphs.

However, it should be noted that only 2-clustering is considered in the above papers.

The trick of using absolute values of weights in the degree matrix allows the whole machinery that we have presented to be used to attack the problem of clustering signed graphs using normalized cuts.

Kunegis et al. appear to be the first to make a systematic study of spectral methods applied to signed graphs.

However, it should be noted that only 2-clustering is considered in the above papers.

The trick of using absolute values of weights in the degree matrix allows the whole machinery that we have presented to be used to attack the problem of clustering signed graphs using normalized cuts.

This requires a modification of the notion of normalized cut.

Definition 6

The *signed normalized cut* $\mathrm{sNcut}(A_1, \ldots, A_K)$ of the partition $(A_1, \ldots, A_K)$ is defined as

$$\mathrm{sNcut}(A_1, \ldots, A_K) = \sum_{j=1}^{K} \frac{\mathrm{cut}(A_j, \overline{A_j})}{\mathrm{vol}(A_j)} + 2 \sum_{j=1}^{K} \frac{\mathrm{links}^{-}(A_j, A_j)}{\mathrm{vol}(A_j)}.$$

Then, we can show that

$$\mathrm{sNcut}(A_1, \ldots, A_K) = \sum_{j=1}^{K} \frac{(X^j)^\top \overline{L} X^j}{(X^j)^\top \overline{D} X^j}.$$

where X is the $N \times K$ matrix whose jth column is X^j and $\overline{L}$ is the signed Laplacian of W.

Then, we can show that

$$\mathrm{sNcut}(A_1, \ldots, A_K) = \sum_{j=1}^{K} \frac{(X^j)^\top \overline{L} X^j}{(X^j)^\top \overline{D} X^j}.$$

where X is the $N \times K$ matrix whose jth column is X^j and $\overline{L}$ is the signed Laplacian of W.

Therefore, *this is the same problem as in the unsigned case, with L replaced by $\overline{L}$ and D replaced by $\overline{D}$*.

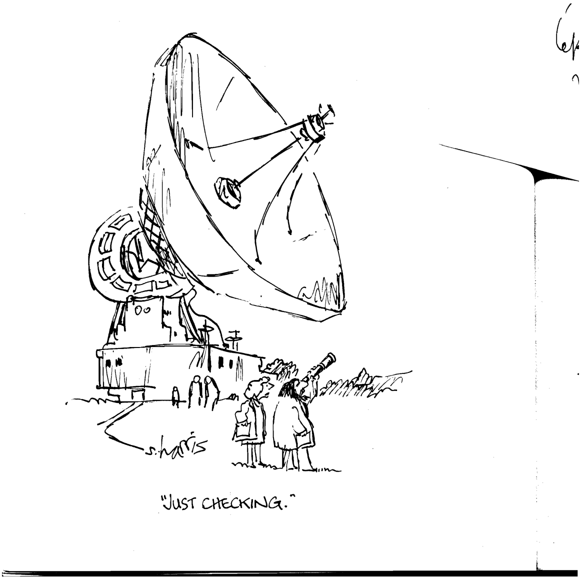


Figure 12: Just Checking!