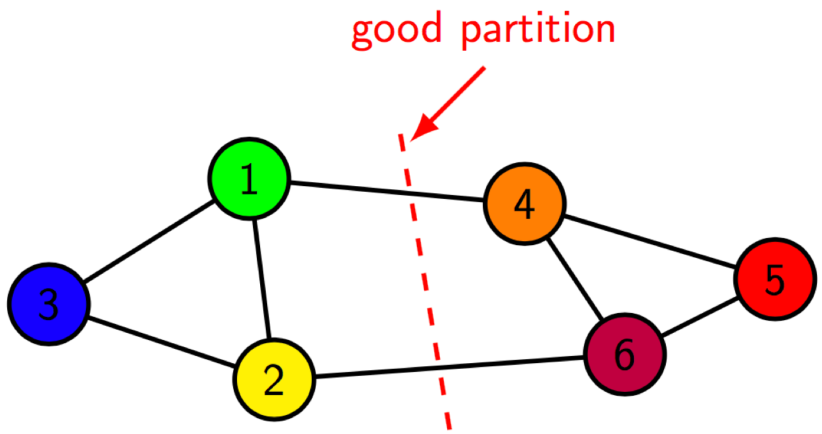
Network Science

#13 Conductance

MIME.

Conductance

Grouping as graph partitioning

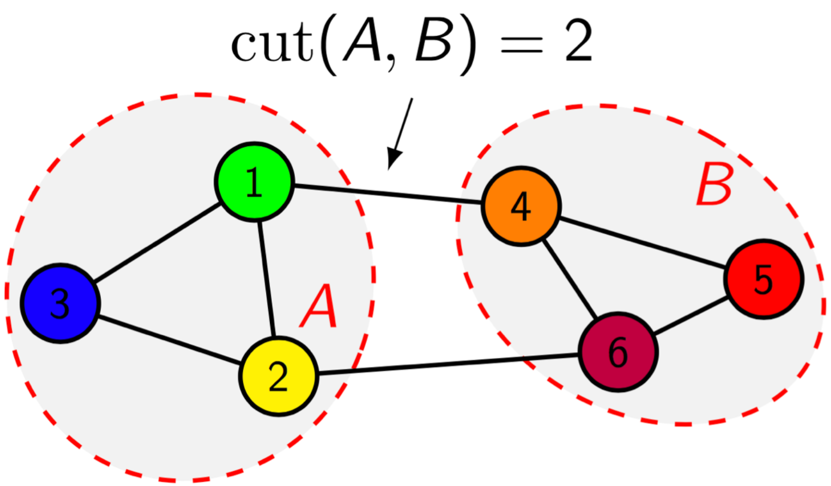


- We want to partition an (undirected) graph in two disjoint groups
- A good partition is one that

  maximizes the # of within-group connections

  minimizes the # of between-group connections

Grouping as graph partitioning

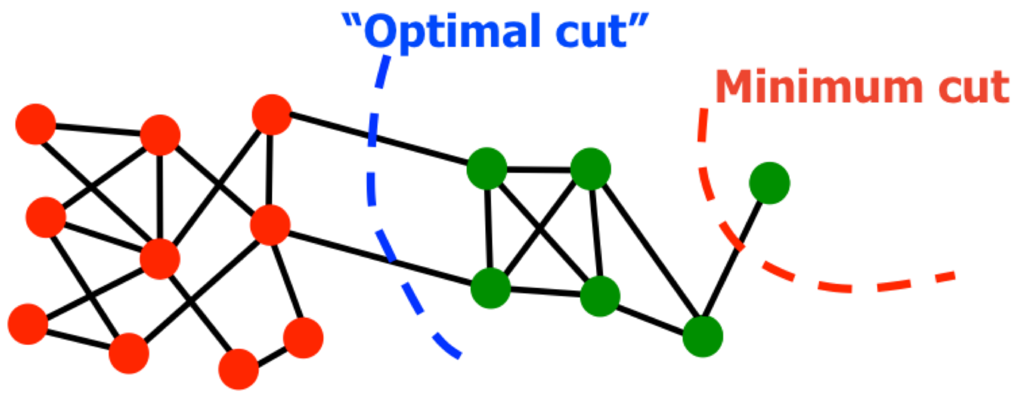


- We define an objective function that expresses the amount of edge cuts of a partition

$$\mathrm{cut}(A,B) = \sum_{i \in A,\, j \in B} a_{ij}$$

Minimum cut criterion

Minimize weight connections between groups by looking for partitions *A* and *B* that minimize cut(*A*,*B*)

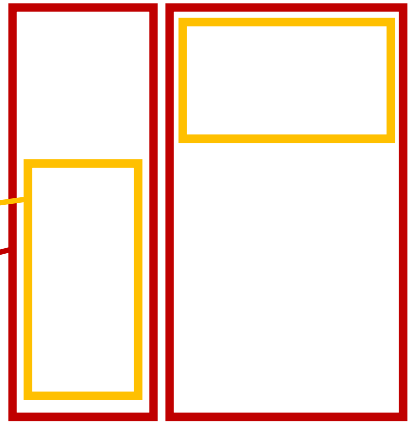


- Favours cutting small sets of isolated nodes (degenerate cuts)
- Only considers external cluster connections
- Does not consider internal cluster connections

Normalized cut

- Normalized cut

$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A)} + \frac{\text{cut}(B,A)}{\text{assoc}(B)}$$

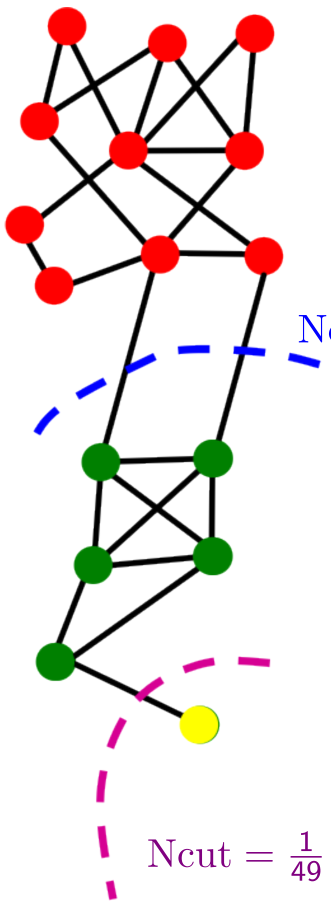


- Normalized cut criterion

  Minimize weight connections between groups by looking for partitions *A* and *B* that minimize Ncut(*A*,*B*)

- Produces more balanced partitions
- Avoids single nodes, Ncut = 1/1 + 1/(L-1) = L/(L-1) $\simeq$ 1
- But computing the optimum is NP-hard

Example



$$\mathrm{Ncut} = \frac{2}{30} + \frac{2}{20} = \frac{1}{6}$$

$$\mathrm{Ncut} = \frac{1}{49} + \frac{1}{1} = \frac{50}{49}$$

28 16

cut = 2

$$\boldsymbol{A} = \left[\begin{array}{cccccccccc|ccccc|c}
 & 1 & & 1 & & & & & & & & & & & & \\
1 & & 1 & 1 & & & & 1 & & & & & & & & \\
 & 1 & & 1 & & 1 & & & & & & & & & & \\
1 & 1 & 1 & & 1 & & 1 & & 1 & & & & & & & \\
 & & & 1 & & 1 & & & & & & & & & & \\
 & & 1 & & 1 & & & 1 & & & & & & & & \\
 & & & 1 & & & & 1 & & & 1 & & & & & \\
 & 1 & & & & 1 & 1 & & & 1 & & 1 & & & & \\
 & & & 1 & & & & & & 1 & & & & & & \\
 & & & & & & & 1 & 1 & & & & & & & \\
\hline
 & & & & & & 1 & & & & & 1 & 1 & 1 & & \\
 & & & & & & & 1 & & & 1 & & 1 & 1 & & \\
 & & & & & & & & & & 1 & 1 & & 1 & 1 & \\
 & & & & & & & & & & 1 & 1 & 1 & & 1 & \\
 & & & & & & & & & & & & 1 & 1 & & 1 \\
\hline
 & & & & & & & & & & & & & & 1 &
\end{array}\right]$$

1

Generalizations

- Multiple communities $$\mathrm{Ncut} = \sum_{i=1}^{K} \frac{\mathrm{cut}(A_i, A_i^c)}{\mathrm{assoc(A_i)}}$$

- Conductance $\phi(S) = \mathrm{cut}(S,S^c)\ /\ \min(\mathrm{assoc}(S), \mathrm{assoc}(S^c))$

MIME.

Spectral clustering

Shi and Malik, “Normalized cuts and image segmentation," 2000
Ng, Jordan, Weiss, "On spectral clustering: analysis and an algorithm," 2002

Ncut with two communities

Find the best partition by minimizing

$$\frac{\boldsymbol{s}^{\mathsf{T}} \boldsymbol{L}\, \boldsymbol{s}}{D^2 - (\boldsymbol{s}^{\mathsf{T}} \boldsymbol{d})^2}$$

community membership (binary) signal

Laplacian matrix $\boldsymbol{L} = \mathrm{diag}(\boldsymbol{d}) - \boldsymbol{A}$

total degrees $D = \boldsymbol{1}^{\mathsf{T}} \boldsymbol{d} = 2L$

degree vector $\boldsymbol{d} = \boldsymbol{A}\, \boldsymbol{1}$

- Computing the optimum is NP-hard
- We need a suboptimum but simpler approach

MIME.

Ncut with two communities

Find the best partition by investigating

$$L_1 = D^{-½} \cdot L \cdot D^{-½} = I - D^{-½} \cdot A \cdot D^{-½}$$

Normalized Laplacian matrix

degree matrix
D = diag(d)

- We look for the eigen-structure of L_1
- L_1 is positive semidefinite with $2 \geq \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{N-1} \geq \lambda_N = 0$
- We assume a connected network where $\lambda_{N-1} > 0$
- We are interested in the algebraic connectivity λ_{N-1} and in the corresponding eigenvector x_{N-1} a.k.a. Fiedler vector
- Community membership corresponds to the signs of x_{N-1}

MiME.

Historical note – Fiedler's algorithm

Czechoslovak Mathematical Journal, 23 (98) 1973, Praha

- Fiedler, Algebraic connectivity of graphs, 1973

ALGEBRAIC CONNECTIVITY OF GRAPHS*)

MIROSLAV FIEDLER, Praha

(Received April 14, 1972)

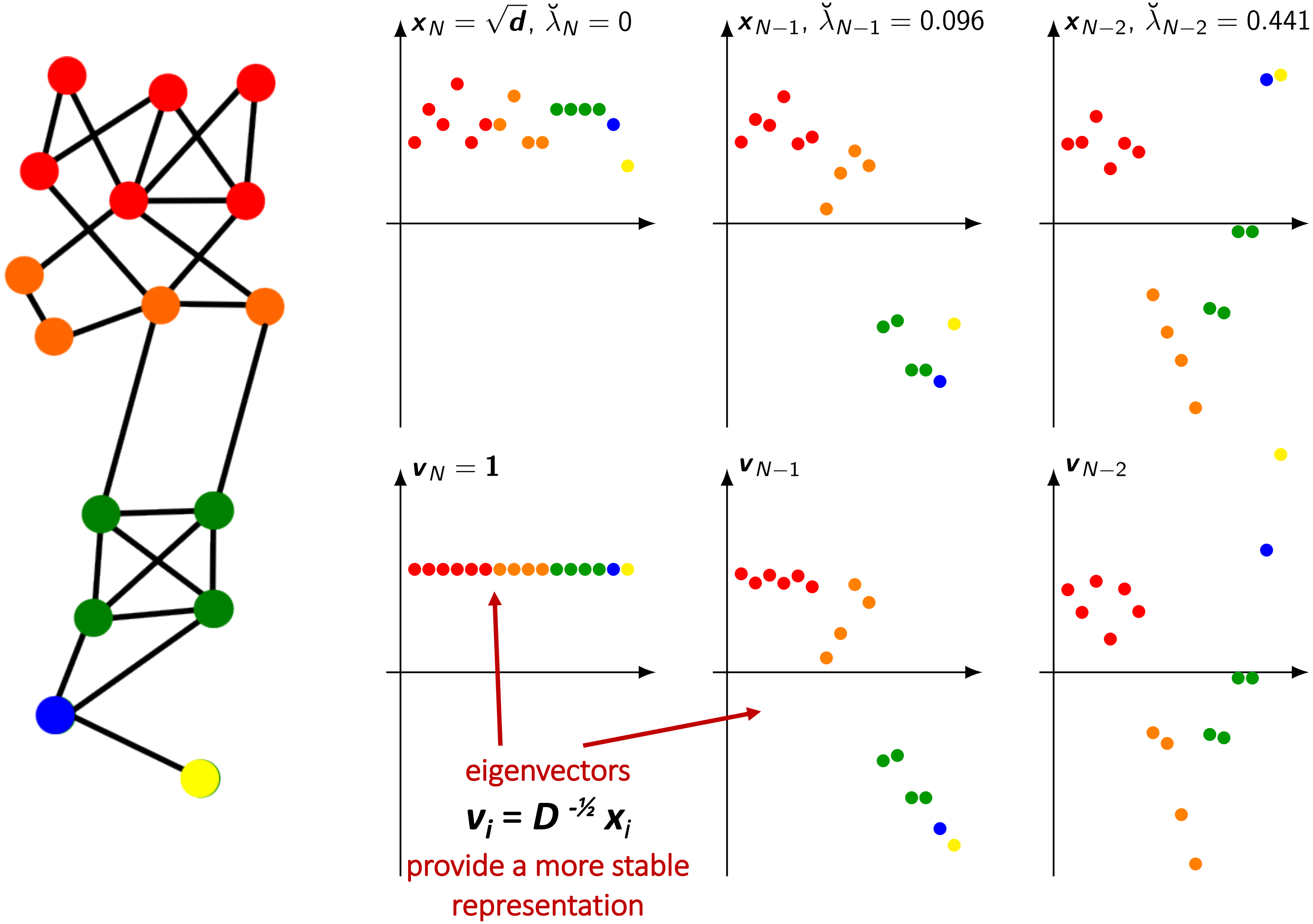
1. INTRODUCTION

Let $G = (V, E)$ be a non-directed finite graph without loops and multiple edges. Having chosen a fixed ordering $w_1, w_2, \ldots, w_n$ of the set V, we can form a square n-rowed matrix $A(G)$ whose off-diagonal entries are $a_{ik} = a_{ki} = -1$ if $(w_i, w_k) \in E$ and $a_{ik} = 0$ otherwise and whose diagonal entries a_{ii} are equal to the valencies of the vertices w_i. This matrix $A(G)$, which is frequently used to enumerate the spanning

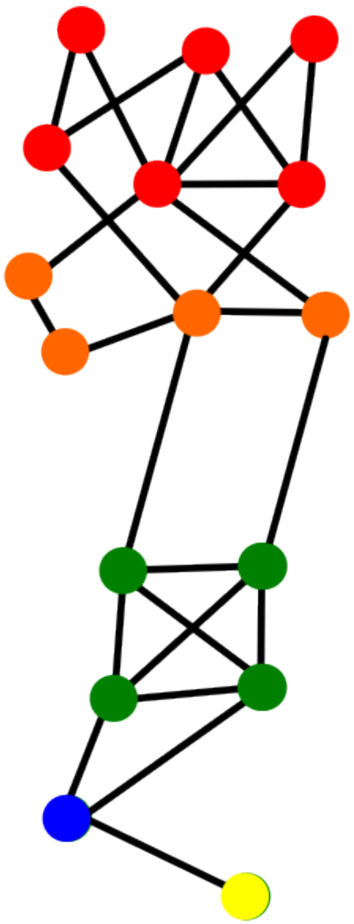
- Identify the Fiedler vector $\boldsymbol{x}_{N-1}$ of $\boldsymbol{L}$ corresponding to algebraic connectivity $\lambda_{N-1} > 0$ and derive two communities according to the signs of x_{N-1}
- But the normalized Laplacian matrix works much better

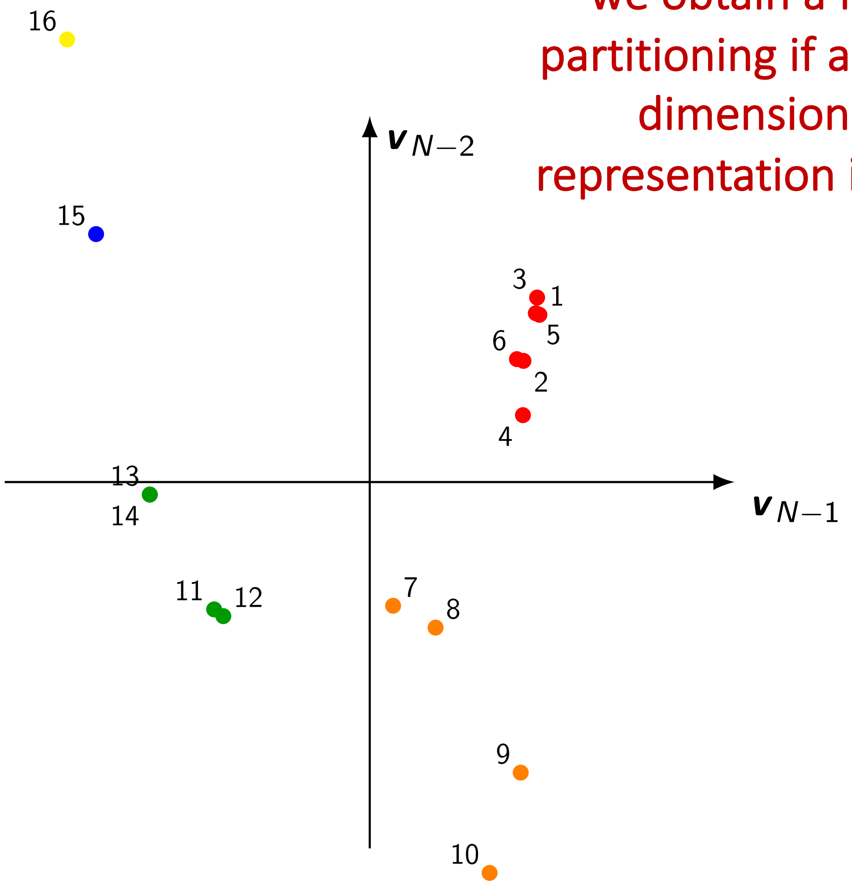
MIME.

Example


$\boldsymbol{x}_N = \sqrt{\boldsymbol{d}}$, $\breve{\lambda}_N = 0$
$\boldsymbol{x}_{N-1}$, $\breve{\lambda}_{N-1} = 0.096$
$\boldsymbol{x}_{N-2}$, $\breve{\lambda}_{N-2} = 0.441$
$\boldsymbol{v}_N = \mathbf{1}$
$\boldsymbol{v}_{N-1}$
$\boldsymbol{v}_{N-2}$
eigenvectors
$\boldsymbol{v}_i = \boldsymbol{D}^{-½}\, \boldsymbol{x}_i$
provide a more stable
representation

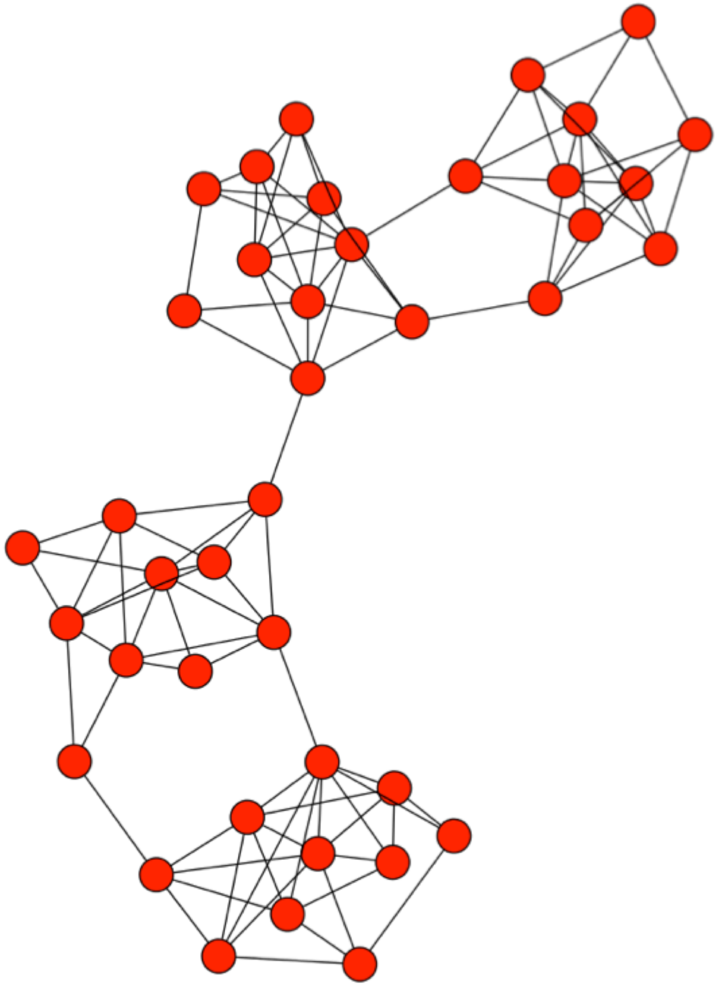
Example

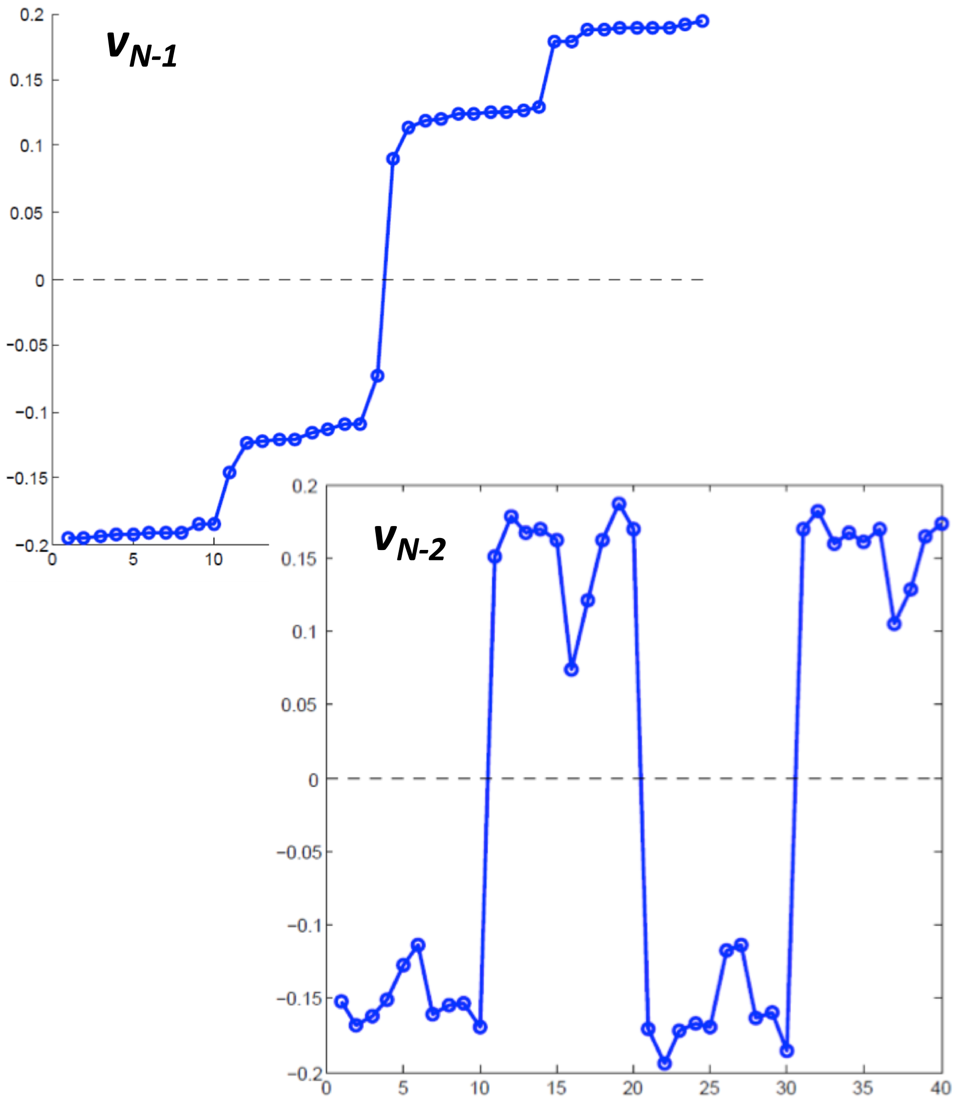




we obtain a finer partitioning if a multi-dimensional representation is used

Another example





Spectral clustering

On vectors $\boldsymbol{v}_i$

- $\boldsymbol{x}_i$ are eigenvectors of $\boldsymbol{L}_1$
- As such they satisfy $\boldsymbol{D}^{-½} \boldsymbol{L} \boldsymbol{D}^{-½} \boldsymbol{x}_i = \lambda_i \boldsymbol{x}_i$
- We replace $\boldsymbol{v}_i = \boldsymbol{D}^{-½} \boldsymbol{x}_i$
- We obtain $\boldsymbol{D}^{-1} \boldsymbol{L} \boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$
- $\boldsymbol{v}_i$ are the (right) eigenvectors of matrix $\boldsymbol{D}^{-1} \boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1} \boldsymbol{A}$
- It is $\lambda_N = 0$ and $\boldsymbol{v}_N = \boldsymbol{1}$ by construction, but we are not interested in it

normalized adjacency matrix *M* -- **looks like** the random walk of PageRank

Spectral clustering algorithm

Ng, Jordan, Weiss, «On spectral clustering: analysis and an algorithm» [2002]

1. Pre-processing
   - Construct a matrix representation $\boldsymbol{A}$ of the graph
   - $\boldsymbol{A}$ can have non-binary weights
   - Derive the normalized Laplacian $\boldsymbol{L}_1$
2. Decomposition
   - Identify K eigenvectors $\boldsymbol{x}_k$ of $\boldsymbol{L}_1$ corresponding to the smallest K eigenvalues $\lambda_{N-1}, \lambda_{N-2}, \ldots, \lambda_{N-K}$
   - Realign eigenvectors $\boldsymbol{v}_k = \boldsymbol{D}^{-1/2} \boldsymbol{x}_k$ and normalize them *to* $||\boldsymbol{v}_k|| = 1$
   - Map each vertex to a lower dimensional representation given by the rows of matrix $[\boldsymbol{v}_{N-1}, \boldsymbol{v}_{N-2}, \ldots, \boldsymbol{v}_{N-K}]$
3. Grouping
   - Assign points to one or more clusters based on the new representation, e.g., using signs, or more elaborate clustering techniques (k-means, EM, etc.)

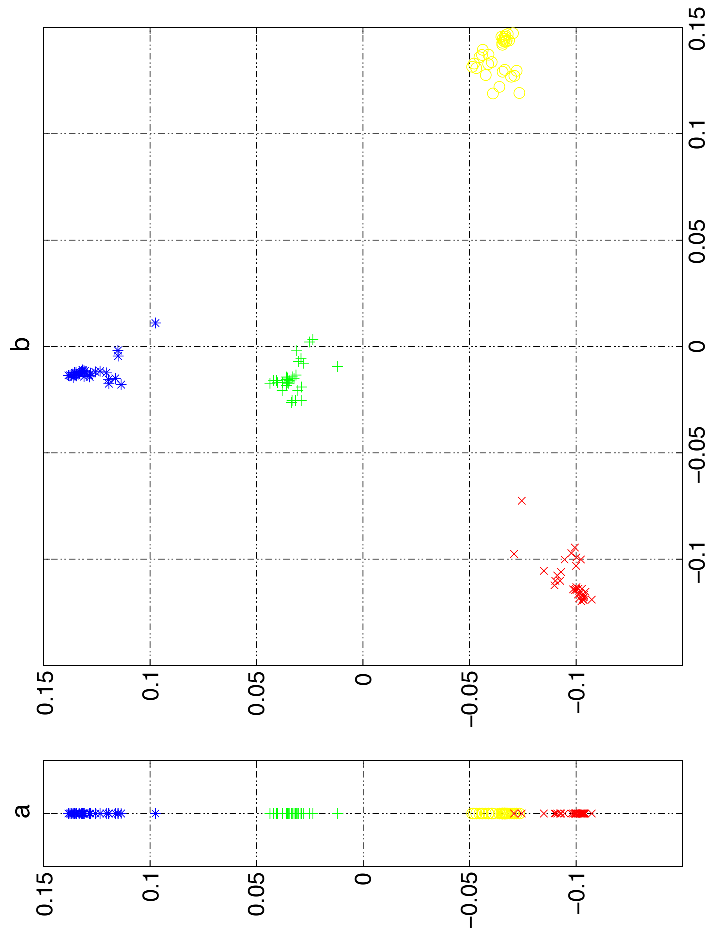
Why multiple eigenvectors?

- Much more stable (and less expensive) than recursive bisection methods
- Approximate the optimal cut

$$\text{Ncut} = \sum_{i=1}^{K} \frac{\text{cut}(A_i, A_i^c)}{\text{assoc}(\mathrm{A_i})}$$

- Communities are better separated
- Emphasizes cohesive clusters
  - ✓ Increases the unevenness in the distribution of data
  - ✓ Associations with similar points are amplified
  - ✓ The data begins to approximate a clustering
  - ✓ Well separated spaces

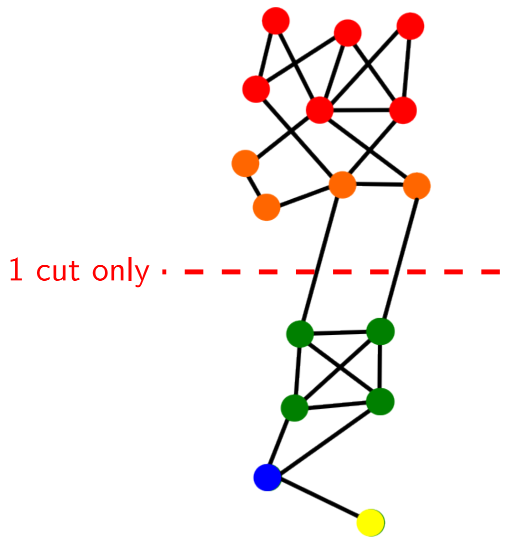
Donetti and Munõz [2004]

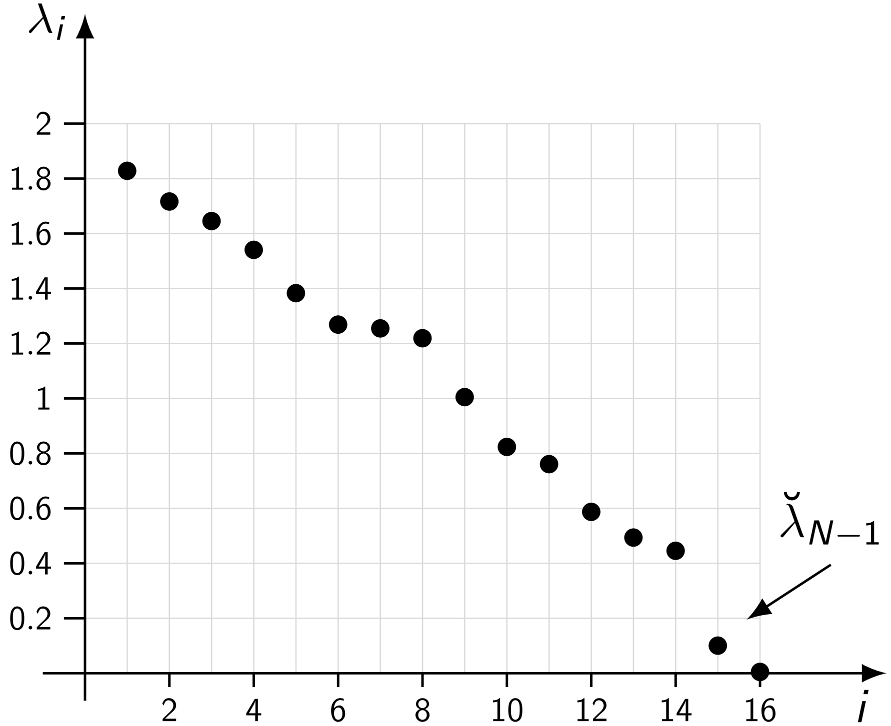


How to select the # of eigenvectors?

The eigengap heuristic

- Choose the number *K* of eigenvectors such that
  - The eigenvalues $\lambda_{N-1}, \lambda_{N-2}, \ldots, \lambda_{N-K}$ are small
  - The eigengap $|\lambda_{N-K+1} - \lambda_{N-K}|$ is large





How good is the clustering result?

Chung, "Laplacians of graphs and Cheeger's inequalities," 1996

Some basic inequalities

- $\lambda_{N-1} \leq N/(N-1)$ with equality for a complete graph
- $\lambda_{N-1} \leq 1$ when the graph is not complete
- $\lambda_1 \geq N/(N-1)$

Cheeger's inequality (1970)

- $\frac{1}{2}\lambda_{N-1} \leq h_G \leq (2\lambda_{N-1})^{1/2}$
- Cheeger's constant $h_G = \min_A \phi(A)$

small algebraic connectivity = small h_G

approximate value (upper bound) of the target function – small h_G means good clustering

Questions ?