Aalto University
School of Science

Basic principles of algorithmic graph mining
Lecture 4 : Spectral graph analysis

Aristides Gionis
Aalto University

Estonian Summer School on Computer and System Science 2016
Aug 22–24, 2016

course agenda

- introduction to graph mining
- computing basic graph statistics
-
- spectral graph analysis
- additional topics and applications

spectral graph theory

spectral graph theory

objective :

- view the adjacency (or related) matrix of a graph with a linear algebra lens
- identify connections between spectral properties of such a matrix and structural properties of the graph
  - connectivity
  - bipartiteness
  - cuts
  - ...
- spectral properties = eigenvalues and eigenvectors
- in other words, what does the eigenvalues and eigenvectors of the adjacency (or related) matrix tell us about the graph?

background: eigenvalues and eigenvectors

- consider a real $n \times n$ matrix A, i.e., $A \in \mathbb{R}^{n \times n}$
- $\lambda \in \mathbb{C}$ is an eigenvalue of A
  if there exists $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x} \neq \mathbf{0}$
  such that
  $$A\mathbf{x} = \lambda\mathbf{x}$$
- such a vector $\mathbf{x}$ is called eigenvector of λ
- alternatively,
  $$(A - \lambda I)\mathbf{x} = \mathbf{0} \quad \text{or} \quad \det(A - \lambda I) = 0$$
- it follows that A has n eigenvalues
  (possibly complex and possibly with multiplicity > 1)

background: eigenvalues and eigenvectors

- consider a real and symmetric $n \times n$ matrix A
  (e.g., the adjacency matrix of an undirected graph)
- then
  - all eigenvalues of A are real
  - eigenvectors of different eigenvalues are orthogonal
    i.e., if $\mathbf{x}_1$ an eigenvector of λ_1
    and $\mathbf{x}_2$ an eigenvector of λ_2
    then $\lambda_1 \neq \lambda_2$ implies $\mathbf{x}_1 \perp \mathbf{x}_2$ (or $\mathbf{x}_1^T\mathbf{x}_2 = 0$)
- A is positive semi-definite if $\mathbf{x}^T A\mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$
- a symmetric positive semi-definite real matrix has real and non negative eigenvalues

background: eigenvalues and eigenvectors

- consider a real and symmetric $n \times n$ matrix A
- the eigenvalues $\lambda_1, \ldots, \lambda_n$ of A can be ordered

$$\lambda_1 \leq \ldots \leq \lambda_n$$

- theorem [variational characterization of eigenvalues]

$$\lambda_n = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T A \mathbf{x}}{\mathbf{x}^T \mathbf{x}}$$

$$\lambda_1 = \min_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T A \mathbf{x}}{\mathbf{x}^T \mathbf{x}}$$

$$\lambda_2 = \min_{\substack{\mathbf{x} \neq \mathbf{0} \\ \mathbf{x}^T \mathbf{x}_1 = 0}} \frac{\mathbf{x}^T A \mathbf{x}}{\mathbf{x}^T \mathbf{x}}$$ and "so on" for the other eigenvalues

- very useful way to think about eigenvalues

background: eigenvalues and eigenvectors

- the inverse holds, i.e.,

$$\lambda_1 = \min_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T A \mathbf{x}}{\mathbf{x}^T \mathbf{x}} = \min_{\mathbf{x} \neq \mathbf{0}} \frac{\sum_{ij} A_{ij} x_i x_j}{\sum_i x_i^2}$$

- and if $\mathbf{x}$ is an optimal vector, then $\mathbf{x}$ is eigenvector of λ_1
- similarly

$$\lambda_2 = \min_{\substack{\mathbf{x} \neq \mathbf{0} \\ \mathbf{x}^T \mathbf{x}_1 = 0}} \frac{\mathbf{x}^T A \mathbf{x}}{\mathbf{x}^T \mathbf{x}} = \min_{\substack{\mathbf{x} \neq \mathbf{0} \\ \mathbf{x}^T \mathbf{x}_1 = 0}} \frac{\sum_{ij} A_{ij} x_i x_j}{\sum_i x_i^2}$$

- and if $\mathbf{x}$ is an optimal vector, then $\mathbf{x}$ is eigenvector of λ_2

spectral graph analysis

- apply the eigenvalue characterization for graphs
- question : which matrix to consider ?
  - the adjacency matrix A of the graph
  - some matrix B so that $\mathbf{x}^T B \mathbf{x}$ is related to a structural property of the graph
- consider $G = (V, E)$ an undirected and d-regular graph (regular graph is used wlog for simplicity of expositions)
- let A be the adjacency matrix of G:
- *laplacian matrix* of G as

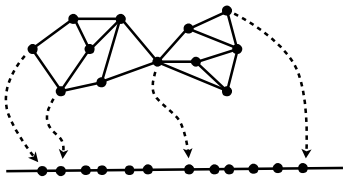
$$L = I - \frac{1}{d} A \quad \text{or} \quad L_{ij} = \begin{cases} 1 & \text{if } i = j \\ -1/d & \text{if } (i,j) \in E, i \neq j \\ 0 & \text{if } (i,j) \notin E, i \neq j \end{cases}$$

spectral graph analysis

- for the laplacian matrix $L = I - \frac{1}{d} A$ it is

$$\mathbf{x}^T L \mathbf{x} = \frac{1}{d} \sum_{(u,v) \in E} |x_u - x_v|^2$$

- here, x_u is the coordinate of the eigenvector $\mathbf{x}$ that corresponds to vertex $u \in V$
- eigenvector $\mathbf{x}$ is seen as a one-dimensional embedding
- i.e., mapping the vertices of the graph onto the real line



spectral graph analysis

high-level remark

- many graph problems can be modeled as mapping of vertices to a discrete space

  e.g., a cut is a mapping of vertices to $\{0, 1\}$

- eigenvector $\mathbf{x}$ is a relaxation of the discrete graph problem

  i.e., optimizes the same objective but without the integrality constraint

the smallest eigenvalue

apply the eigenvalue characterization theorem for L

- what is λ_1 ?

$$\lambda_1 = \min_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T L \mathbf{x}}{\mathbf{x}^T \mathbf{x}} = \min_{\mathbf{x} \neq \mathbf{0}} \frac{\sum_{(u,v) \in E} |x_u - x_v|^2}{d \sum_{u \in V} x_u^2}$$

- observe that $\lambda_1 \geq 0$
- can it be $\lambda_1 = 0$?
- yes : take $\mathbf{x}$ to be the constant vector

the second smallest eigenvalue

apply the eigenvalue characterization theorem for L

- what is λ_2 ?

$$\lambda_2 = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}^T\mathbf{x}_1=0}} \frac{\mathbf{x}^T L\,\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}^T\mathbf{x}_1=0}} \frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d\sum_{u\in V} x_u^2}$$

- can it be $\lambda_2 = 0$?
- $\lambda_2 = 0$ if and only if the graph is disconnected

  map the vertices of each connected component to a different constant

the k-th smallest eigenvalue

- alternative characterization for λ_k

$$\lambda_k = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}\in\mathbb{S} \\ \mathbb{S}:k\text{-dim}}} \max \frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d\sum_{u\in V} x_u^2}$$

- $\lambda_k = 0$ if and only if the graph has at least k connected components

the largest eigenvalue

- what about λ_n ?

$$\lambda_n = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\mathbf{x}^T L\,\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d\sum_{u\in V} x_u^2}$$

- consider a boolean version of this problem
- restrict mapping to $\{-1,+1\}$

$$\lambda_n \geq \max_{\mathbf{x}\in\{-1,+1\}^n} \frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d\sum_{u\in V} x_u^2}$$

the largest eigenvalue

- mapping of vertices to $\{-1,+1\}$ corresponds to a cut S then

$$\begin{aligned}
\lambda_n &\geq \max_{\mathbf{x}\in\{-1,+1\}^n} \frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d\sum_{u\in V} x_u^2} \\
&= \max_{S\subseteq V} \frac{4\,E(S, V\setminus S)}{d\,n} \\
&= \max_{S\subseteq V} \frac{4\,E(S, V\setminus S)}{2\,|E|} \\
&= \frac{2\,\text{maxcut}(G)}{|E|}
\end{aligned}$$

- it follows that if G bipartite then $\lambda_n \geq 2$
  (because if G bipartite exists S that cuts all edges)

the largest eigenvalue

- on the other hand

$$\begin{aligned}
\lambda_n &= \max_{\mathbf{x}\neq\mathbf{0}} \frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d\sum_{u\in V} x_u^2} \\
&= \max_{\mathbf{x}\neq\mathbf{0}} \frac{2d\sum_{u\in V} x_u^2 - \sum_{(u,v)\in E}(x_u + x_v)^2}{d\sum_{u\in V} x_u^2} \\
&= 2 - \min_{\mathbf{x}\neq\mathbf{0}} \frac{\sum_{(u,v)\in E}(x_u + x_v)^2}{d\sum_{u\in V} x_u^2}
\end{aligned}$$

- $\lambda_n \leq 2$
- $\lambda_n = 2$ iff there is $\mathbf{x}$ s.t. $x_u = -x_v$ for all $(u,v) \in E$
- $\lambda_n = 2$ iff G has a bipartite connected component

summary so far

eigenvalues and structural properties of G :

- $\lambda_2 = 0$ iff G is disconnected
- $\lambda_k = 0$ iff G has at least k connected components
- $\lambda_n = 2$ iff G has a bipartite connected component

robustness

- how robust are these results ?
- for instance, what if $\lambda_2 = \epsilon$?

  is the graph G almost disconnected ?

  i.e., does it have small cuts ?
- or, what if $\lambda_n = 2 - \epsilon$?

  does it have a component that is "close" to bipartite ?

the second eigenvalue

$$\lambda_2 = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}^T\mathbf{x}_1=0}} \frac{\sum_{(u,v)\in E}(x_u - x_v)^2}{d\sum_{u\in V} x_u^2} = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}^T\mathbf{x}_1=0}} \frac{\sum_{(u,v)\in E}(x_u - x_v)^2}{\frac{d}{n}\sum_{(u,v)\in V^2}(x_u - x_v)^2}$$

where V^2 is the set of ordered pairs of vertices

why?

$$\sum_{(u,v)\in V^2}(x_u - x_v)^2 = n\sum_v x_v^2 - 2\sum_{u,v} x_u x_v = n\sum_v x_v^2 - 2\left(\sum_u x_u\right)^2$$

and $\sum_u x_u = 0$ since $\mathbf{x}^T\mathbf{x}_1 = 0$

the second eigenvalue

$$\lambda_2 = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}^T\mathbf{x}_1=0}} \frac{\sum_{(u,v)\in E}(x_u - x_v)^2}{\frac{d}{n}\sum_{(u,v)\in V^2}(x_u - x_v)^2} = \min_{\substack{\mathbf{x}\neq\mathbf{0} \\ \mathbf{x}^T\mathbf{x}_1=0}} \frac{n}{d}\frac{\mathbb{E}_{(u,v)\in E}[(x_u - x_v)^2]}{\mathbb{E}_{(u,v)\in V^2}[(x_u - x_v)^2]}$$

consider again discrete version of the problem, $x_u \in \{0, 1\}$

$$\min_{\substack{\mathbf{x}\neq\{0,1\}^n \\ \mathbf{x} \text{ non const}}} \frac{n}{d}\frac{\mathbb{E}_{(u,v)\in E}[(x_u - x_v)^2]}{\mathbb{E}_{(u,v)\in V^2}[(x_u - x_v)^2]} = \min_{S\subseteq V} \frac{n}{d}\frac{E(S,\overline{S})}{|S|\,|\overline{S}|} = \mathrm{usc}(G)$$

$\mathrm{usc}(G)$: uniform sparsest cut of G

uniform sparsest cut

- it can be shown that

  $$\lambda_2 \leq \mathrm{usc}(G) \leq \sqrt{8\lambda_2}$$
- relaxation
- second inequality is constructive :
- if $\mathbf{x}$ is an eigenvector of λ_2

  then there is some $t \in V$ such that

  the cut $(S, V\setminus S) = (\{u \in V \mid x_u \leq x_t\}, \{u \in V \mid x_u > x_t\})$

  has cost $\mathrm{usc}(S) \leq \sqrt{8\lambda_2}$

conductance

- conductance : another measure for cuts
- the conductance of a set $S \subseteq V$

  $$\phi(S) = \frac{E(S, V\setminus S)}{d|S|}$$
- expresses the probability to "move out" of S by following a random edge from S
- we are interested in sets of small conductance
- the conductance of the graph G

  $$\phi(G) = \min_{\substack{S\subseteq V \\ 0\leq S\leq |V|/2}} \phi(S)$$

Cheeger's inequality

- Cheeger's inequality:

  $$\frac{\lambda_2}{2} \leq \frac{\mathrm{usc}(G)}{2} \leq \phi(G) \leq \sqrt{2\lambda_2}$$

  $\Rightarrow$ conductance is small if and only if λ_2 is small
- the two leftmost inequalities are "easy" to show
- relaxation
- the second follows by

  $$\frac{\mathrm{usc}(S)}{2} = \frac{n}{2d}\frac{E(S, V\setminus S)}{|S||V\setminus S|} \leq \frac{E(S, V\setminus S)}{d|S|} = \phi(S)$$

  since $|V\setminus S| \geq n/2$

Cheeger's inequality

$$\frac{\lambda_2}{2} \le \frac{\mathrm{usc}(G)}{2} \le \phi(G) \le \sqrt{2\lambda_2}$$

-
- proof sketch (three steps):

1. consider a vector $\mathbf{y} \ge \mathbf{0}$
   - $S \subseteq \{v \in V \mid y_v > 0\}$ such that

$$\phi(S) \le \frac{\sum_{(u,v)\in E} |y_u - y_v|}{d \sum_{u\in V} |y_u|} \qquad \text{(no squares)}$$

   - pick random $t \in [0, \max_v y_v]$ $\qquad S = \{v \mid y_v \ge t\}$
   - then $\phi(S) \le$ r.h.s on expectation
   - thus, there is some t that the property holds

Cheeger's inequality

$$\frac{\lambda_2}{2} \le \frac{\mathrm{usc}(G)}{2} \le \phi(G) \le \sqrt{2\lambda_2}$$

- proof sketch (three steps):

2. given a vector $\mathbf{x}$ $\qquad$ $\mathbf{y}$ such that

$$\frac{\sum_{(u,v)\in E} |y_u - y_v|}{d \sum_{u\in V} |y_u|} \le \sqrt{2\frac{\sum_{(u,v)\in E} |x_u - x_v|^2}{d \sum_{u\in V} |x_u|^2}}$$

   and $\quad |\{v \mid y_v > 0\}| \le \frac{n}{2}$
   - proof of this claim is constructive; uses Cauchy-Schwarz
3. take $\mathbf{x}$ to be the eigenvector of λ_2

generalization to non-regular graphs

- $G = (V, E)$ is undirected and non-regular
- let d_u be the degree of vertex u
- D to be a diagonal matrix whose u-th diagonal element is d_u
- the *normalized laplacian matrix* of G

$$L = I - D^{-1/2} A D^{-1/2}$$

or

$$L_{uv} = \begin{cases} 1 & \text{if } u = v \\ -1/\sqrt{d_u d_v} & \text{if } (u,v) \in E, u \ne v \\ 0 & \text{if } (u,v) \notin E, u \ne v \end{cases}$$

generalization to non-regular graphs

- with the *normalized laplacian*

  the eigenvalue expressions become (e.g., λ_2)

$$\lambda_2 = \min_{\substack{\mathbf{x}\ne\mathbf{0} \\ \langle \mathbf{x}, \mathbf{x}_1 \rangle_D = 0}} \frac{\sum_{(u,v)\in E} (x_u - x_v)^2}{\sum_{u\in V} d_u x_u^2}$$

  where we use weighted inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle_D = \sum_{u\in V} d_u x_u y_u$$

summary so far

eigenvalues and structural properties of G :

- $\lambda_2 = 0$ iff G is disconnected
- $\lambda_k = 0$ iff G has at least k connected components
- $\lambda_n = 2$ iff G has a bipartite connected component
- small λ_2 iff G is "almost" disconnected (small conductance)

random walks

random walks

- consider random walk on the graph G by following edges
- from vertex i move to vertex j with prob. $1/d_i$ if $(i,j) \in E$
- $\mathbf{p}_i^{(t)}$ probability of being at vertex i at time t
- process is described by equation $\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} P$, where $P = D^{-1} A$ is row-stochastic
- process converges to stationary distribution $\pi = \pi P$ (under certain irreducibility conditions)
- for undirected and connected graphs

$$\pi_i = \frac{d_i}{2m} \quad \text{(stationary distribution} \quad \text{degree)}$$

random walks — useful concepts

- hitting time $H(i,j)$: expected number of steps before visiting vertex j, starting from i
- commute time $\kappa(i,j)$: expected number of steps before visiting j and i again, starting at i : $\kappa(i,j) = H(i,j) + H(j,i)$
- cover time: expected number of steps to reach every node
- mixing time $\tau(\epsilon)$: a measure of how fast the random walk approaches its stationary distribution

$$\tau(\epsilon) = \min\{t \mid d(t) \leq \epsilon\}$$

where

$$d(t) = \max_i ||\mathbf{p}^t(i,\cdot) - \pi|| = \max_i \left\{ \sum_j |\mathbf{p}^t(i,j) - \pi_j| \right\}$$

random walks vs. spectral analysis

- consider the normalized laplacian $L = I - D^{-1/2} A D^{-1/2}$

$$\begin{aligned} L\mathbf{u} &= \lambda \mathbf{u} \\ (I - D^{-1/2} A D^{-1/2})\mathbf{u} &= \lambda \mathbf{u} \\ (D - A)\mathbf{u} &= \lambda D \mathbf{u} \\ D\mathbf{u} &= A\mathbf{u} + \lambda D \mathbf{u} \\ (1-\lambda)\mathbf{u} &= D^{-1} A \mathbf{u} \\ \mu \mathbf{u} &= P \mathbf{u} \end{aligned}$$

- $(\lambda, \mathbf{u})$ is an eigenvalue–eigenvector pair for L if and only if $(1-\lambda, \mathbf{u})$ is an eigenvalue–eigenvector pair for P
- the eigenvector with smallest eigenvalue for L is the eigenvector with largest eigenvalue for P

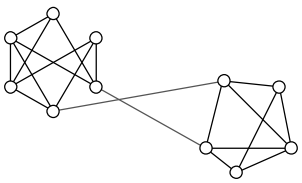
random walks vs. spectral analysis

- stochastic matrix P, describing the random walk
- eigenvalues: $-1 < \mu_n \leq \ldots \leq \mu_2 < \mu_1 = 1$
- spectral gap: $\gamma_* = 1 - \mu_2 = \lambda_2$
- relaxation time: $\tau_* = \frac{1}{\gamma_*}$
- theorem: for an aperiodic, irreducible, and reversible random walk, and any ϵ

$$(\tau_* - 1) \log\left(\frac{1}{2\epsilon}\right) \leq \tau(\epsilon) \leq \tau_* \log\left(\frac{1}{2\epsilon\sqrt{\pi_{\min}}}\right)$$

random walks vs. spectral analysis

- intuition: fast mixing related to graph being an expander



small spectral gap $\Leftrightarrow$ large mixing time $\Leftrightarrow$ bottlenecks $\Leftrightarrow$

$\Leftrightarrow$ clusters $\Leftrightarrow$ low conductance $\Leftrightarrow$ small λ_2

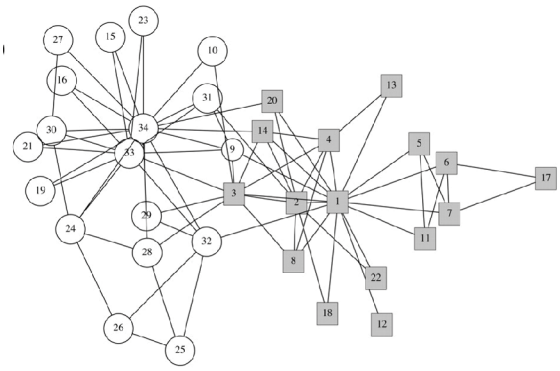
graph partitioning

graph partitioning and community detection

motivation

- knowledge discovery
  - partition the web into sets of related pages (web graph)
  - find groups of scientists who collaborate with each other (co-authorship graph)
  -
    (query graph)
- performance
  - partition the nodes of a large social network into different machines so that, to a large extent, friends are in the same machine (social networks)

graph partitioning



[Newman and Girvan, 2004])

basic spectral-partition algorithm

1. form normalized Laplacian $L' = I - D^{-1/2} A D^{-1/2}$
2. compute eigenvector $\mathbf{x}_2$ (Fielder vector)
3. $\mathbf{x}_2$
4. consider only sweeping cuts: splits that respect the order
5. take the sweeping cut S that minimizes $\phi(S)$

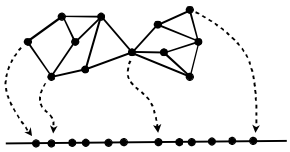
theorem
a cut S such that $\phi(S) \leq 2\sqrt{\phi(G)}$

proof: by Cheeger inequality

$$\phi(S) \leq \sqrt{2 \cdot \lambda_2} \leq \sqrt{2 \cdot 2 \cdot \phi(G)}$$

spectral partitioning rules

1. conductance: $\phi(G)$
2. bisection: split in two equal parts
3. sign: separate positive and negative values
4. gap: separate according to the largest gap



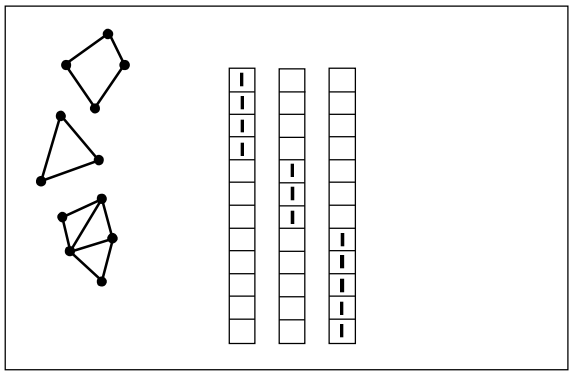
other common spectral-partitioning algorithms

1. utilize more eigenvectors than just the Fielder vector
   use k eigenvectors
2. different versions of the Laplacian matrix

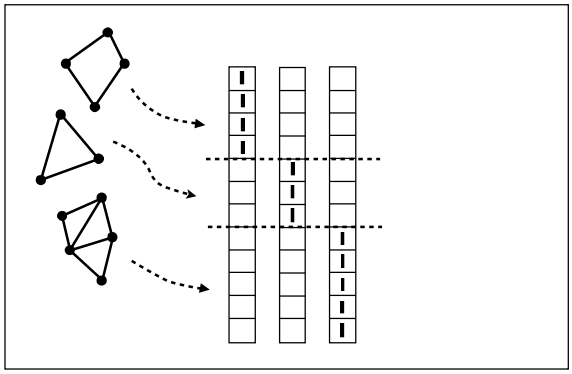
using k eigenvectors

- ideal scenario: the graph consists of k disconnected components (perfect clusters)
- then: eigenvalue 0 of the Laplacian has multplicity k
  the eigenspace of eigenvalue 0 is spanned by indicator vectors of the graph components

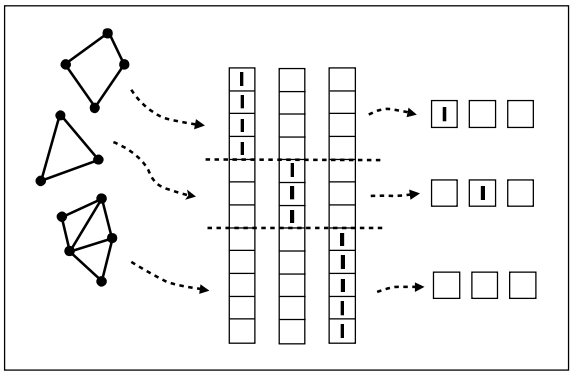
using k eigenvectors



using k eigenvectors



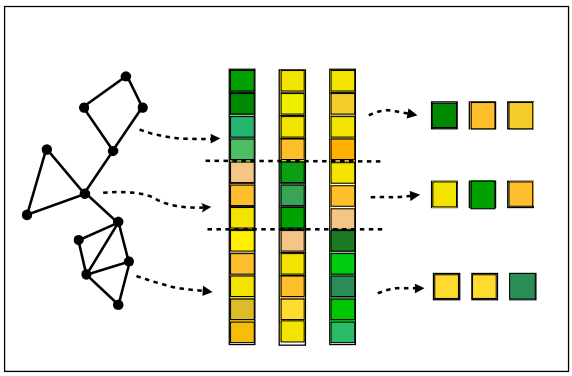
using k eigenvectors



using k eigenvectors

- robustness under perturbations: if the graph has less well-separated components the previous structure holds approximately
- clustering of Euclidean points can be used to separate the components

using k eigenvectors



laplacian matrices

- normalized laplacian: $L = I - D^{-1/2} A D^{-1/2}$
- unormalized laplacian: $L_u = D - A$
- normalized "random-walk" laplacian: $L_{\text{rw}} = I - D^{-1} A$

all laplacian matrices are related

- unormalized Laplacian: $\lambda_2 = \min_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{u}_1=0}} \sum_{(i,j)\in E}(x_i - x_j)^2$
- normalized Laplacian:

$$\lambda_2 = \min_{\substack{||\mathbf{x}||=1 \\ \mathbf{x}^T\mathbf{u}_1=0}} \sum_{(i,j)\in E}\left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}}\right)^2$$

- $(\lambda, \mathbf{u})$ is an eigenvalue/vector of L_{rw} if and only if $(\lambda, D^{1/2}\,\mathbf{u})$ is an eigenvalue/vector of L
- $(\lambda, \mathbf{u})$ is an eigenvalue/vector of L_{rw} if and only if $(\lambda, \mathbf{u})$ solve the generalized eigen-problem $L_u\,\mathbf{u} = \lambda\, D\,\mathbf{u}$

algorithm 1: unormalized spectral clustering

input graph adjacency matrix A, number k

1. form diagonal matrix D
2. form unormalized Laplacian $L = D - A$
3. k eigenvectors $u_1, \ldots, u_k$ of L
4. form matrix $U \in \mathbb{R}^{n\times k}$ with columns $u_1, \ldots, u_k$
5. consider the i-th row of U as point $y_i \in \mathbb{R}^k, i = 1, \ldots, n$,
6. cluster the points $\{y_i\}_{i=1,\ldots,n}$ into clusters $C_1, \ldots, C_k$
   e.g., with k-means clustering

output clusters $A_1, \ldots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$

algorithm 2: normalized spectral clustering

[Shi and Malik, 2000]

input graph adjacency matrix A, number k

1. form diagonal matrix D
2. form unormalized Laplacian $L = D - A$
3. k eigenvectors $u_1, \ldots, u_k$ of the generalized eigenproblem $L\,\mathbf{u} = \lambda\, D\,\mathbf{u}$ (eigvctrs of L_{rw})
4. form matrix $U \in \mathbb{R}^{n\times k}$ with columns $u_1, \ldots, u_k$
5. consider the i-th row of U as point $y_i \in \mathbb{R}^k, i = 1, \ldots, n$,
6. cluster the points $\{y_i\}_{i=1,\ldots,n}$ into clusters $C_1, \ldots, C_k$
   e.g., with k-means clustering

output clusters $A_1, \ldots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$

algorithm 3: normalized spectral clustering

[Ng et al., 2001]

input graph adjacency matrix A, number k

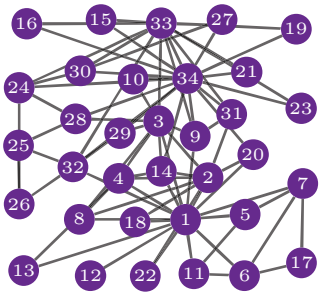
1. form diagonal matrix D
2. form normalized Laplacian $L' = I - D^{-1/2} A\, D^{-1/2}$
3. k eigenvectors $u_1, \ldots, u_k$ of L'
4. form matrix $U \in \mathbb{R}^{n\times k}$ with columns $u_1, \ldots, u_k$
5. normalize U so that rows have norm 1
6. consider the i-th row of U as point $y_i \in \mathbb{R}^k, i = 1, \ldots, n$,
7. cluster the points $\{y_i\}_{i=1,\ldots,n}$ into clusters $C_1, \ldots, C_k$
   e.g., with k-means clustering

output clusters $A_1, \ldots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$

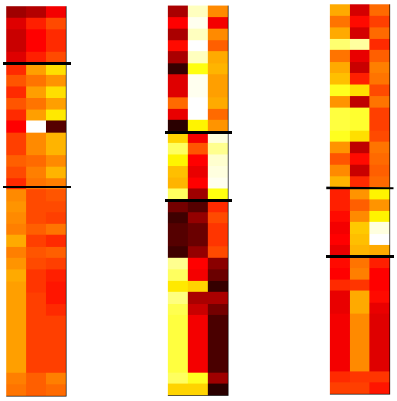
notes on the spectral algorithms

- quite similar except for using different Laplacians
- can be used to cluster any type of data, not just graphs
  form all-pairs similarity matrix and use as adjacency matrix
- computation of the first eigenvectors of sparse matrices can be done efficiently using the Lanczos method

Zachary's karate-club network

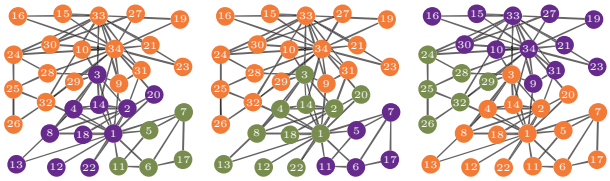


Zachary's karate-club network



unormalized Laplacian | normalized symmetric Laplacian | normalized random walk Laplacian

Zachary's karate-club network



unormalized Laplacian | normalized symmetric Laplacian | normalized random walk Laplacian

which Laplacian to use?

[von Luxburg, 2007]

- when graph vertices have about the same degree all laplacians are about the same
- for skewed degree distributions normalized laplacians tend to perform better
- normalized laplacians are associated with conductance, which is a good objective
  (conductance involves vol(S) rather than $|S|$ and captures better the community structure)

modularity

- one component
- many components ?
- related question: what is the optimal number of partitions ?
- modularity has been used to answer those questions [Newman and Girvan, 2004]
- in hierarchical graph partitioning

modularity

- intuition: compare actual subgraph density with expected subgraph density, if vertices were attached regardless of community structure

$$
\begin{aligned}
Q &= \frac{1}{2m}\sum_{ij}(A_{ij} - P_{ij})\delta(C_i, C_j) \\
&= \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{d_i d_j}{2m})\delta(C_i, C_j) \\
&= \sum_c \left[\frac{m_c}{2m} - \left(\frac{d_c}{2m}\right)^2\right]
\end{aligned}
$$

$P_{ij} = 2mp_ip_j = 2m(d_i/2m)(d_j/2m) = (d_id_j/2m)$
m_c: edges within cluster c
d_c: total degree of cluster c

values of modularity

- 0 random structure; 1 strong community structure; [0.3..0.7]; typical good structure; can be negative, too
- Q measure is not monotone with k

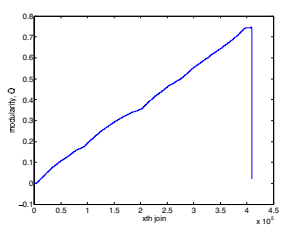


FIG. 1: The modularity Q over the course of the algorithm (the x axis shows the number of joins). Its maximum value is $Q = 0.745$, where the partition consists of 1684 communities.

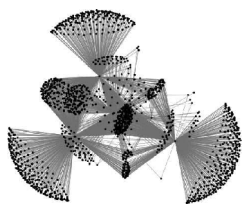


FIG. 2: A visualization of the community structure at maximum modularity. Note that the some major communities have a large number of "satellite" communities connected only to them (top, lower left, lower right). Also, some pairs of major communities have sets of smaller communities that act as "bridges" between them (e.g., between the lower left and lower right, near the center).

(figures from [Clauset et al., 2004])

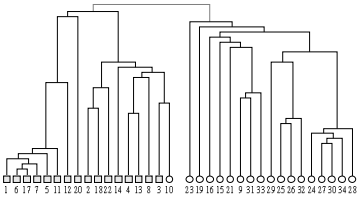
optimizing modularity

- problem optimizes modularity
- **NP**-hard problem [Brandes et al., 2006]
- top-down approaches [Newman and Girvan, 2004]
- spectral approaches [Smyth and White, 2005]
- mathematical-programming [Agarwal and Kempe, 2008]

top-down algorithms for optimizing modularity

[Newman and Girvan, 2004]

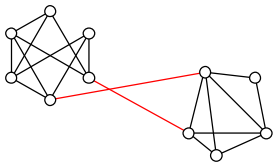
- a set of algorithms based on removing edges from the graph, one at a time
- the graph gets progressively disconnected, creating a hierarchy of communities



[Newman, 2004])

top-down algorithms

- select edge to remove based on *"betweenness"*



- shortest-path betweenness: number of shortest paths that the edge belongs to
- random-walk betweenness: expected number of paths for a random walk from u to v
- current-flow betweenness: resistance derived from considering the graph as an electric circuit

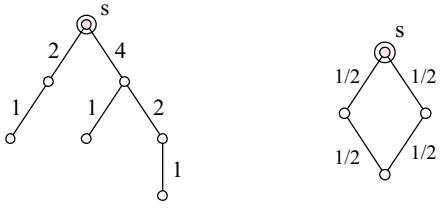
top-down algorithms

general scheme

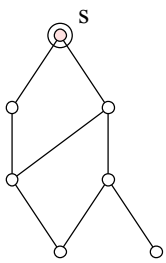
1. TOP-DOWN
2. compute betweenness value of all edges
3. remove the edge with the highest betweenness
4. recompute betweenness value of all remaining edges
5. repeat until no edges left

shortest-path betweenness

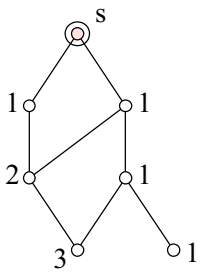
- how to compute shortest-path betweenness?
- BFS from each vertex
- leads to $O(mn)$ for all edge betweenness
- OK if there are single paths to all vertices



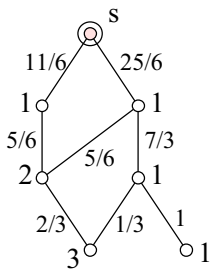
shortest-path betweenness



shortest-path betweenness



shortest-path betweenness



overall time of TOPDOWN is $O(m^2 n)$

random-walk betweenness

- stochastic matrix of random walk is $P = D^{-1} A$
- $\mathbf{s}$ is the vector with 1 at position s and 0 elsewhere
- probability distribution over vertices at time n is $\mathbf{s} P^n$
- expected number of visits at each vertex given by

$$\sum_n \mathbf{s} P^n = \mathbf{s} (1 - P)^{-1}$$

$$c_u = \text{E[\# times passing from } u \text{ to } v] = \left[\mathbf{s} (1 - P)^{-1}\right]_u \frac{1}{d_u}$$

$$\mathbf{c} = \mathbf{s} (1 - P)^{-1} D^{-1} = \mathbf{s} (D - A)^{-1}$$

- *random-walk betweenness* at (u, v) as $|c_u - c_v|$

random-walk betweenness

- *random-walk betweenness* at (u, v) is $|c_u - c_v|$ with $\mathbf{c} = \mathbf{s} (D - A)^{-1}$
- one matrix inversion $O(n^3)$
- in total $O(n^3 m)$ time with recalculation
- not scalable
- *current-flow betweenness* is equivalent!

  [Newman and Girvan, 2004] recommend shortest-path betweenness

other modularity-based algorithms

spectral approach [Smyth and White, 2005]

$$\begin{aligned} Q &= \sum_{c=1}^{k} \left[\frac{m_c}{2m} - \left(\frac{d_c}{2m} \right)^2 \right] \propto \sum_{c=1}^{k} \left[(2m)\, m_c - d_c^2 \right] \\ &= \sum_{c=1}^{k} \left[(2m) \sum_{i,j=1}^{n} w_{ij} x_{ic} x_{jc} - \left(\sum_{i=1}^{n} d_i x_{ic} \right)^2 \right] \\ &= \sum_{c=1}^{k} \left[(2m)\, \mathbf{x}_c^T W \mathbf{x}_c - \mathbf{x}_c^T D \mathbf{x}_c \right] \\ &= \text{tr}(X^T (W' - D)\, X) \end{aligned}$$

where $X = [\mathbf{x}_1 \ldots \mathbf{x}_k] = [x_{ic}]$ point-cluster assignment matrix

spectral-based modularity optimization

maximize	$\text{tr}(X^T (W' - D)\, X)$
such that	X is an assignment matrix

solution:

$$L_Q X = X \Lambda$$

where $L_Q = W' - D$, Q-Laplacian

- standard eigenvalue problem
- but solution is fractional, we want integral
- treat rows of X as vectors and cluster graph vertices using k-means
- [Smyth and White, 2005] propose two algorithms, based on this idea

spectral-based modularity optimization

spectral algorithms perform almost as good as the agglomerative, but they are more efficient

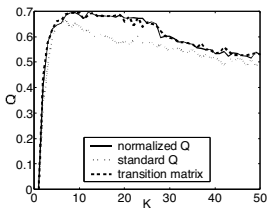


Figure 3: Q versus k for the WordNet data.

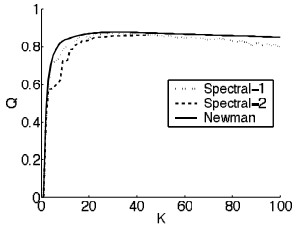


Figure 7: Q versus k for NIPS coauthorship data.

[Smyth and White, 2005]

other modularity-based algorithms

mathematical programming [Agarwal and Kempe, 2008]

$$Q \propto \sum_{i,j=1}^{n} B_{ij}(1 - x_{ij})$$

where

$$x_{ij} = \begin{cases} 0 & \text{if } i \text{ and } j \text{ get assigned to the same cluster} \\ 1 & \text{otherwise} \end{cases}$$

it should be

$$x_{ik} \leq x_{ij} + x_{jk} \quad \text{for all vertices } i, j, k$$

solve the integer program with triangle inequality constraints

mathematical-programming approach for modularity optimization

[Agarwal and Kempe, 2008]

- integer program is **NP**-hard
- relax integrality constraints
  replace $x_{ij} \in \{0, 1\}$ with $0 \leq x_{ij} \leq 1$
- corresponding linear program can be solved in polynomial time
- solve linear program and round the fractional solution
- place in the same cluster vertices i and j if x_{ij} is small (pivot algorithm [Ailon et al., 2008])

Results

Network	size n	GN	DA	EIG	VP	LP	UB
KARATE	34	0.401	0.419	0.419	0.420	0.420	0.420
DOLPH	62	0.520	-	-	0.526	0.529	0.531
MIS	76	0.540	-	-	0.560	0.560	0.561
BOOKS	105	-	-	0.526	0.527	0.527	0.528
BALL	115	0.601	-	-	0.605	0.605	0.606
JAZZ	198	0.405	0.445	0.442	0.445	0.445	0.446
COLL	235	0.720	-	-	0.803	0.803	0.805
META	453	0.403	0.434	0.435	0.450	-	-
EMAIL	1133	0.532	0.574	0.572	0.579	-	-

Table 2. The modularity obtained by many of the previously published methods and by the methods introduced in this paper, along with the upper bound.

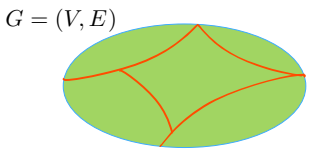
(table from [Agarwal and Kempe, 2008])

need for scalable algorithms

- spectral, agglomerative, LP-based algorithms
- not scalable to very large graphs
- handle datasets with billions of vertices and edges
  - `facebook`: 1 billion users with avg degree 130
  - `twitter`: 1.5 billion social relations
  - `google`: web graph more than a trillion edges (2011)
- design algorithms for streaming scenarios
  - `twitter` posts
  - election trends, `twitter` as election barometer

graph partitioning

- graph partitioning is a way to split the graph vertices in multiple machines
- graph partitioning objectives guarantee low communication overhead among different machines
- additionally balanced partitioning is desirable



- each partition contains $\approx n/k$ vertices

off-line k-way graph partitioning

METIS algorithm [Karypis and Kumar, 1998]

- popular family of algorithms and software
- multilevel algorithm
- coarsening phase in which the size of the graph is successively decreased
- followed by bisection (based on spectral)
- followed by uncoarsening phase in which the bisection is

summary

- spectral analysis reveals structural properties of a graph
- used for graph partitioning, but also for other problems
- well-studied area, many results and techniques
- for graph partitioning and community detection many other methods are available

acknowledgements

Luca Trevisan

references

Agarwal, G. and Kempe, D. (2008).
Modularity-maximizing graph communities via mathematical programming.
The European Physical Journal B, 66(3).

Ailon, N., Charikar, M., and Newman, A. (2008).
Aggregating inconsistent information: ranking and clustering.
Journal of the ACM (JACM), 55(5).

Brandes, U., Delling, D., Gaertler, M., Görke, R., Höfer, M., Nikoloski, Z., and Wagner, D. (2006).
Maximizing modularity is hard.
Technical report, DELIS – Dynamically Evolving, Large-Scale Information Systems.

Clauset, A., Newman, M., and Moore, C. (2004).
Finding community structure in very large networks.
arXiv.org.

references (cont.)

Karypis, G. and Kumar, V. (1998).
A fast and high quality multilevel scheme for partitioning irregular graphs.
SIAM J. Sci. Comput., 20(1):359–392.

Newman, M. (2004).
Fast algorithm for detecting community structure in networks.
Physical review E, 69(6).

Newman, M. E. J. and Girvan, M. (2004).
Finding and evaluating community structure in networks.
Physical Review E, 69(2).

Ng, A., Jordan, M., and Weiss, Y. (2001).
On spectral clustering: Analysis and an algorithm.
NIPS.

Shi, J. and Malik, J. (2000).
Normalized cuts and image segmentation.
IEEE transactions on Pattern Analysis and Machine Intelligence, 22(8).

references (cont.)

Smyth, P. and White, S. (2005).
A spectral clustering approach to finding communities in graphs.
SDM.

von Luxburg, U. (2007).
A Tutorial on Spectral Clustering.
arXiv.org.