Università della Svizzera italiana | Facoltà di scienze della comunicazione | Istituto di tecnologie della comunicazione ITC

Notes on Spectral Clustering

Politecnico di Milano, 28/05/2012

Davide Eynard
Institute of Computational Sciences - Faculty of Informatics
Università della Svizzera Italiana
davide.eynard@usi.ch

Talk outline

- Spectral Clustering
  - Distances and similarity graphs
  - Graph Laplacians and their properties
  - Spectral clustering algorithms
  - SC under the hood

Similarity graph

- The objective of a clustering algorithm is partitioning data into groups such that:
  - Points in the **same** group are **similar**
  - Points in **different** groups are **dissimilar**

- **Similarity graph** $G=(V,E)$ (*undirected* graph)
  - Vertices v_i and v_j are connected by a **weighted** edge if their similarity is above a given threshold
  - GOAL: find a partition of the graph such that:
    - edges **within** a group have **high weights**
    - edges **across** different groups have **low weights**

Weighted adjacency matrix

- Let $G(V,E)$ be an undirected graph with vertex set $V=\{v_1,\ldots,v_n\}$
- **Weighted adjacency matrix** $\boldsymbol{W}=(w_{ij})_{i,j=1,\ldots,n}$
  - $w_{ij} \geq 0$ is the weight of the edge between v_i and v_j
  - $w_{ij}=0$ means that v_i and v_j are not connected by an edge
  - $w_{ij}=w_{ji}$
- **Degree of a vertex** $v_i \in V$: $\boldsymbol{d_i}=\sum_{j=1..n} w_{ij}$
- **Degree matrix** $\boldsymbol{D}=diag(d_1,\ldots,d_n)$

Different similarity graphs

- ε-neighborhood
  - Connect all points whose pairwise distance is less than ε
- *k*-nearest neighbors
  - if $v_i \in knn(v_j)$ ***OR*** $v_j \in knn(v_i)$
  - if $v_i \in knn(v_j)$ ***AND*** $v_j \in knn(v_i)$ (*mutual knn*)
  - after connecting edges, use similarity as weight
- fully connected
  - *all* points with similarity $s_{ij}>0$ are connected
  - To control neighborhoods to be *local*, use a similarity function like the ***Gaussian***: $s(x_i,x_j)=exp(-\|x_i-x_j\|^2/(2\sigma^2))$

Graph Laplacians

- **Graph Laplacian:**
  - $\boldsymbol{L} = D - W$ *(symmetric and positive semi-definite)*
- Properties
  - Smallest eigenvalue $\lambda_1=0$ with eigenvector = $\mathbb{1}$
  - n non-negative, real-valued eigenvalues $0=\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$
  - the multiplicty k of the eigenvalue 0 of L equals the number of connected components $A_1,\ldots,A_k$ in the graph

Spectral Clustering algorithm (1)

Spectral Clustering algorithm

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

1. Construct a similarity graph as previously described. Let W be its weighted adjacency matrix.
2. Compute the unnormalized Laplacian L
3. Compute the first k eigenvectors $u_1,\dots,u_k$ of L
4. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1,\dots,u_k$ as columns
5. For $i=1,\dots,n$ let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U
6. Cluster the points $(y_i)_{i=1,\dots,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1,\dots,C_k$.

Output: Clusters $A_1,\dots,A_k$ with $A_i = \{j \mid y_j \in C_i\}$.

Normalized Graph Laplacians

- **Normalized graph Laplacians**
  - *Symmetric*: $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$
  - *Random Walk*: $L_{rw} = D^{-1} L = I - D^{-1} W$
- Properties
  - λ is an eigenvalue of L_{rw} with eigenvector u iff λ is an eigenvalue of L_{sym} with eigenvector $w = D^{1/2} u$
  - λ is an eigenvalue of L_{rw} with eigenvector u iff λ and u solve the **generalized eigenproblem** $Lu = \lambda Du$

Normalized Graph Laplacians

- **Normalized graph Laplacians**
  - *Symmetric*: $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$
  - *Random Walk*: $L_{rw} = D^{-1} L = I - D^{-1} W$
- Properties (follow)
  - 0 is an eigenvalue of L_{rw} with $\mathbb{1}$ as eigenvector, and an eigenvalue of L_{sym} with eigenvector $D^{1/2}\mathbb{1}$.
  - L_{sym} and L_{rw} are positive semi-definite and have n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$
  - the multiplicty k of the eigenvalue 0 of both L_{sym} *and* L_{rw} equals the number of connected components $A_1, \ldots, A_k$

Spectral Clustering algorithm (2)

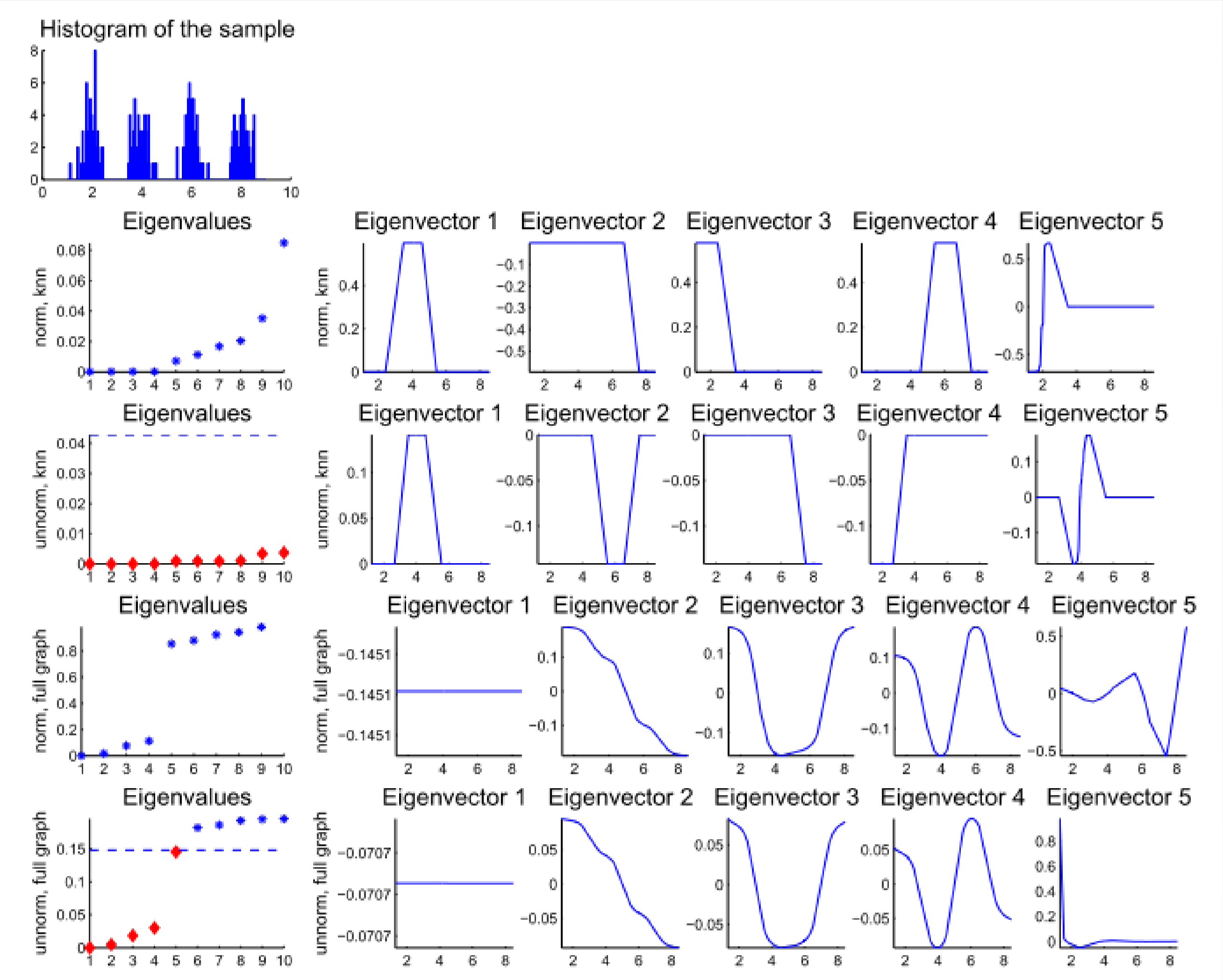
Normalized Spectral Clustering

Input: Similarity matrix $S \in \mathbb{R}^{n\times n}$, number k of clusters to construct.

- L_{rw}:
  3. Compute the first *k* **generalized** eigenvectors $u_1,...,u_k$ of the generalized eigenproblem $Lu=\lambda Du$
- L_{sym}:
  2. Compute the **normalized** Laplacian L_{sym}
  3. Compute the first *k* eigenvectors $u_1,...,u_k$ of L_{sym}
  4. normalize the eigenvectors

Output: Clusters $A_1,...,A_k$ with $A_i = \{j \mid y_j \in C_i\}$.

A spectral clustering example



Under the hood

- *0-eigenvalues* in the ideal case
- *parameters* are crucial:
  - *k* in k nearest neighbors
  - σ in Gaussian kernel
  - *k* (another one!) in k-means

Random Walk point of view

- Random walk: stochastic process which randomly jumps from one vertex to another
  - Clustering: finding a partition such that a random walk stays long within a cluster and seldom jumps between clusters
- Transition probability $p_{ij} := w_{ij} / d_i$
- Transition matrix: $P = D^{-1}W \quad \Rightarrow \quad L_{rw} = I - P$
  - λ is an eigenvalue of L_{rw} with eigenvector u iff $1-\lambda$ is an eigenvalue of P with eigenvector u

References

- Von Luxburg, U. (2007). *A tutorial on spectral clustering.* Statistics and Computing, 17(4), 395-416. Springer.

Thank you!

Thanks for your attention!

Questions?