Lecture 7:
Spectral Clustering;
Linear Dimensionality Reduction via Principal Component Analysis

Lester Mackey

April 21, 2014
Blackboard discussion

- See lecture notes
Spectral clustering example: GMM

Data generated from a mixture of 4 Gaussians in 1D

W from 10-nearest neighbors

Top row: normalized $L_{rw}$

Bottom row: unnormalized $L_{un}$

- Data generated from a mixture of 4 Gaussians in 1D
- W from 10-nearest neighbors
- Top row: normalized $L_{rw}$
- Bottom row: unnormalized $L_{un}$
Spectral clustering example: GMM

- Data generated from a mixture of 4 Gaussians in 1D
- \( W = S \)
- Top row: normalized \( L_{rw} \)
- Bottom row: unnormalized \( L_{un} \)

**Diagram:**
- Odd rows: fully connected graph
- Even rows: \( k \)-means

**Images:**
- Eigenvalues and eigenvectors for different similarity functions.
- Histogram of the sample.
Spectral clustering example: circles

\[ \text{Eigenvectors of } L \]

\[ \text{Eigenvalues of } L \]

\( k \)-means, 2 clusters

Spectral clustering, 2 clusters

Courtesy: Sriram Sankararaman
Spectral clustering example: noisy circles

\[ \text{Eigenvalues of } L \]

\[ \text{Eigenvectors of } L \]

\( k \)-means, 2 clusters

Spectral clustering, 2 clusters

Courtesy: Sriram Sankararaman
Spectral clustering image segmentation

- Spectral clustering widely used in image segmentation

Shi and Malik, 2001
How to construct graph weights W?

- **Goal:** capture local neighborhood relationships between points / focus on very similar points

- **Most common constructions**
  - **$\epsilon$-neighborhood graph:** connect all points with similarity $> \epsilon$
    - Use same weight for all connected points
  - **$k$-nearest neighbor graph:** connect $i$ and $j$ if $j$ is among the $k$-most similar vertices to $i$ **or** vice-versa
    - Weight retained edges according to similarity
  - **Mutual $k$-nearest neighbor graph:** connect $i$ and $j$ if $j$ is among the $k$-most similar vertices to $i$ **and** vice-versa
    - Weight retained edges according to similarity
  - **Fully connected graph:** connect all nodes
    - Only useful when “local” similarity measure used like $s_{ij} = \exp(-||x_i - x_j||^2 / (2\sigma^2))$, which decays rapidly
known to guide us in this task. In general, if the similarity graph contains more connected components than the number of clusters we ask the algorithm to detect, then spectral clustering will trivially return connected components as clusters. Unless one is perfectly sure that those connected components are the correct clusters, one should make sure that the similarity graph is connected, or only consists of “few” connected components and very few or no isolated vertices. There are many theoretical results on how connectivity of random graphs can be achieved, but all those results only hold in the limit for the sample size \( n \to \infty \). For example, it is known that for \( n \) data points drawn i.i.d. from some underlying density with a connected support in \( \mathbb{R}^d \), the \( k \)-nearest neighbor graph and the mutual \( k \)-nearest neighbor graph will be connected if we choose \( k \) on the order of \( \log(n) \) (e.g., Brito et al. 1997). Similar arguments show that the parameter \( \varepsilon \) in the \( \varepsilon \)-neighborhood graph has to be chosen as \( (\log(n)/n)^d \) to guarantee connectivity in the limit (Penrose 1999). While being of theoretical interest, all those results do not really help us for choosing \( k \) on a finite sample.

Now let us give some rules of thumb. When working with the \( k \)-nearest neighbor graph, then the connectivity parameter should be chosen such that the resulting graph is connected, or at least has significantly fewer connected components than clusters we want to detect. For small or medium-sized graphs this can be tried out “by foot”. For very large graphs, a first approximation could be to choose \( k \) in the order of \( \log(n) \), as suggested by asymptotic connectivity results.

For the mutual \( k \)-nearest neighbor graph, we have to admit that we are a bit lost for rules of thumb. The advantage of the mutual \( k \)-nearest neighbor graph compared to the standard \( k \)-nearest neighbor graph is that it tends not to connect areas of different density. While this can be good if there are clear clusters induced by separate high-density areas, this can hurt in less obvious situations as disconnected parts in the graph will always be chosen to be clusters by spectral clustering. Very generally, one can observe that the mutual \( k \)-nearest neighbor graph has much fewer edges than the standard \( k \)-nearest neighbor graph for the same parameter \( k \). This suggests to choose \( k \) significantly larger for the mutual \( k \)-nearest neighbor graph than one would do for the standard \( k \)-nearest neighbor graph. However, to take advantage of the property that the mutual \( k \)-nearest neighbor graph does not connect regions of different density, it would be necessary to allow for several “meaningful” disconnected parts of the graph. Unfortunately, we do not know of any general heuristic to choose the parameter \( k \) such that this can be achieved.

For the \( \varepsilon \)-neighborhood graph, we suggest to choose \( \varepsilon \) such that the resulting graph is safely connected. To determine the smallest value of \( \varepsilon \) where the graph is connected is very simple: one has to choose \( \varepsilon \) as the length of the longest edge in a minimal spanning tree of the fully connected graph on the data points. The latter can be determined easily by any minimal spanning tree algorithm. However, note that when...
Spectral clustering and optimality

- Is spectral clustering optimal in any sense? If so, for what objective?
  - One variant minimizes a relaxation of the normalized cut graph partitioning criterion (Shi and Malik, 2000)
    - Same variant, based on $L_{rw}$, approximately minimizes probability that a random walk on the weighted graph transitions from one cluster to another
  - Consistency studied under certain statistical models (e.g., Rohe/Chatterjee/Yu, 2010 - Spectral clustering and the high-dimensional stochastic blockmodel)
Dimensionality reduction

- **Goal:** Find a low-dimensional representation that captures the “essence” of higher-dimensional data points
  - Also known as **latent feature modeling**

- **Motivation**
  - **Compression** for improved storage and computational complexity
  - **Visualization** for improved human understanding of data
    - Difficult to plot / interpret data in more than 3 dimensions
  - **Noise reduction**
    - Ameliorates noisy and infrequent measurements, missingness
  - **Preprocessing** for supervised learning task
    - Reduced / denoised representations may lead to better performance or act as regularization for reduced overfitting
  - **Anomaly detection**
    - Characterize normal data and distinguish from outliers
**Linear dimensionality reduction**

- **Given:** High-dimensional datapoints \( x_i \in \mathbb{R}^p \)
  - e.g., images of faces in \( \mathbb{R}^{361} \)

- **Goal:** Assign useful representations \( z_i = U^T x_i \in \mathbb{R}^k \), where a \( U^T \in \mathbb{R}^{k \times p} \) is a linear mapping into a low-dimensional space

- How to choose a useful \( U \)?
Blackboard discussion

- See lecture notes