IntroML Tutorial

Clustering and Dimensionality Reduction II

Nezihe Merve Gürel (nezihe.guerel@inf.ethz.ch)

April 29 - May 3, 2019

https://las.inf.ethz.ch/teaching/introml-s19

Table of Contents

1. Spectral Clustering

2. Kernel PCA

3. Autoencoders

Clustering

Clustering strategies

- Model-based clustering (*an underlying distribution that is a mixture of two or more clusters*),
- hierarchical clustering (*a tree based representation of objects*),
- density-based clustering (*identify the clusters with different shape and size*),
- **partitioning methods** (*subdivide the data sets into k groups*),

Global optimal: exhaustive enumeration of all partititons

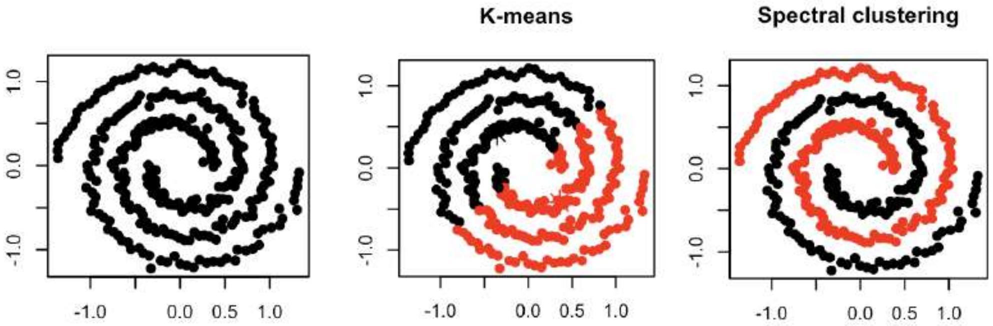
Heuristic methods: *K-means*, K-medoids, K-medians, etc.

Today's focus. Partitioning via Spectral Clustering

Spectral Clustering

K-means fails in clustering the manifolds with arbitrary shape but only compact ones!

Spectral clustering identifies communities of data points that are connected



- The data points are treated as nodes of a graph
- The partitioning of data points are based on the edges connecting them
- More specifically, spectrum (eigenvalues) of a graph-based dissimilarity matrix is exploited to learn partitioning
- Suitable for clustering arbitrary manifolds, e.g., intertwined spirals!

image credit: http://scalefreegan.github.io/Teaching/DataIntegration/practicals/p2.html

Spectral Clustering

Basic Stages.

1. Matrix Representation of a Graph

1.1 Construct an undirected similarity graph based on the similarity between nodes (data points). We represent the similarity between the nodes by a symmetric adjacency matrix A

1.2 Form Laplacian matrix L based on A

2. Embedding

Perform eigenvalue decomposition of the graph in order to embed data onto a low-dimensional space (*spectral embedding*) such that cluster are more obvious

3. Clustering

Apply a clustering algorithm to partition the embeddings, e.g., k-means

Spectral Clustering

Matrix representations of a graph

Given set of data points $x_1, \cdots, x_n$ and $w_{ij} \geq 0$ between x_i and x_j, form an undirected graph

$$\boxed{G = (V, E, A)}$$

with $V = \{v_1, \cdots, v_n\}$ where $v_i = x_i$ and the edge between two vertices v_i and v_j carries a non-negative weight $w_{ij} \geq 0$ and A is the affinity matrix that is built based on edge weights.

Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n \times n}$ symmetric matrix built upon the similarities between vertices

How to form adjacency graph?

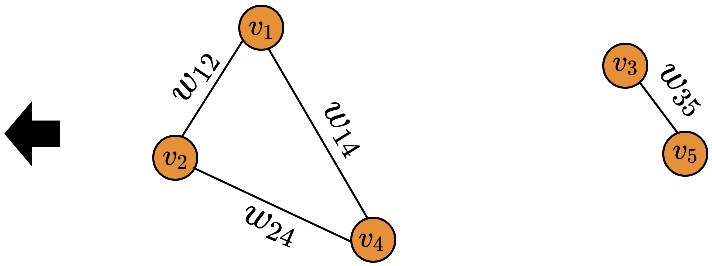
The ε-neighborhood graph. Connect all vertices whose pairwise distances are smaller than ε.

$$A_{ij} = \begin{cases} w_{ij} & \text{:if } w_{ij} \le \varepsilon \\ 0 & \text{:else} \end{cases}$$

$n = 5$

$A =$

	1	2	3	4	5
1	0	w_{12}	0	w_{14}	0
2	w_{12}	0	0	w_{24}	0
3	0	0	0	0	w_{35}
4	w_{14}	w_{24}	0	0	0
5	0	0	w_{35}	0	0



Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n\times n}$ symmetric matrix built upon the similarities between vertices

How to form adjacency graph?

The ε-neighborhood graph. Connect all vertices whose pairwise distances are smaller than ε.

$$A_{ij} = \begin{cases} w_{ij} & \text{:if } w_{ij} \leq \varepsilon \\ 0 & \text{:else} \end{cases}$$

Disadvantage. Loss of similarity information

Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n \times n}$ symmetric matrix built upon the similarities between vertices

How to form adjacency graph?

k-**nearest neighbor graph**. Connect the vertex v_i with v_j such that

$$A_{ij} = \begin{cases} w_{ij} & \text{:if } v_i \text{ is among } k\text{-neighbors of } v_i \\ 0 & \text{:else} \end{cases}$$

Disadvantage. The graph is no more undirected

Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n \times n}$ symmetric matrix built upon the similarities between vertices

How to form adjacency graph?

k-**nearest neighbor graph**. Connect the vertex v_i with v_j such that

$$A_{ij} = \begin{cases} w_{ij} & \text{:if } v_i \text{ is among } k\text{-neighbors of } v_i \\ 0 & \text{:else} \end{cases}$$

Disadvantage. The graph is no more undirected

Solution. 1. Ignore the the directions of edges ☺ and take all neighbors into account

2. Connect them only if both are neighbors of each other (mutual k-nn)

weight the edges by the similarity of end points

Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n \times n}$ symmetric matrix built upon the similarities between vertices

How to form adjacency graph?

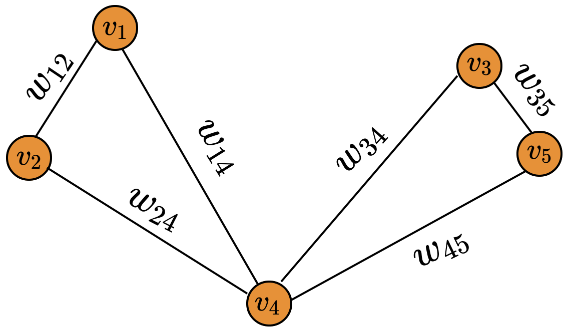
k-**nearest neighbor graph**. Connect the vertex v_i with v_j such that

$$A_{ij} = \begin{cases} w_{ij} & \text{:if } v_i \text{ is among } k\text{-neighbors of } v_i \\ 0 & \text{:else} \end{cases}$$

$A =$

	1	2	3	4	5
1	0	w_{12}	w_{13}	0	0
2	w_{12}	0	w_{23}	0	0
3	0	0	0	w_{34}	w_{35}
4	w_{14}	w_{24}	w_{34}	0	w_{35}
5	0	0	w_{35}	w_{45}	0

$n = 5, \ k = 2$



Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n \times n}$ symmetric matrix built upon the similarities between vertices

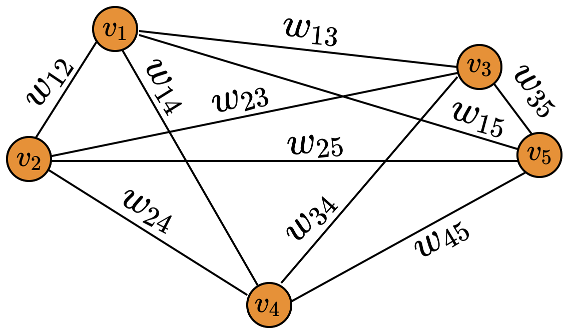
How to form adjacency graph?

The fully connected graph. Weight all edges such that

$$A_{ij} = w_{ij}$$

$A =$

	1	2	3	4	5
1	0	w_{12}	w_{13}	w_{14}	w_{15}
2	w_{12}	0	w_{23}	w_{24}	w_{25}
3	w_{13}	w_{23}	0	w_{34}	w_{35}
4	w_{14}	w_{24}	w_{34}	0	w_{35}
5	w_{15}	w_{25}	w_{35}	w_{45}	0



Spectral Clustering

Matrix representations of a graph.

Adjacency Matrix $A \in \mathbb{R}^{n \times n}$ symmetric matrix built upon the similarities between vertices

How to form adjacency graph?

The fully connected graph. Weight all edges such that

$$A_{ij} = w_{ij}$$

Gaussian kernel similarity function as an example:

$$w_{ij} = \exp\left(-\|x_i - x_j\|^2/(2\sigma^2)\right)$$

where σ is the width of neighborhoods.

See [Luxburg 2007] for further reading.

Spectral Clustering

Graph Laplacian

Definition

Let degree of a vertex $v_i \in V$ be given by $d_i = \sum_{j=1}^{n} A_{ij}$.

The degree matrix is a diagonal matrix with degrees $d_1, \cdots, d_n$ on the diagonal: $\boxed{D}$

Note: There is no unique description on "graph Laplacian". There are different variants with their own properties. We will first focus on the unnormalized graph Laplacian

The unnormalized graph Laplacian

$$\boxed{L = D - A}$$

$$L_{ij} = \begin{cases} A_{ij} - d_i & \text{:if } i = j \\ A_{ij} & \text{:else} \end{cases}$$

Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

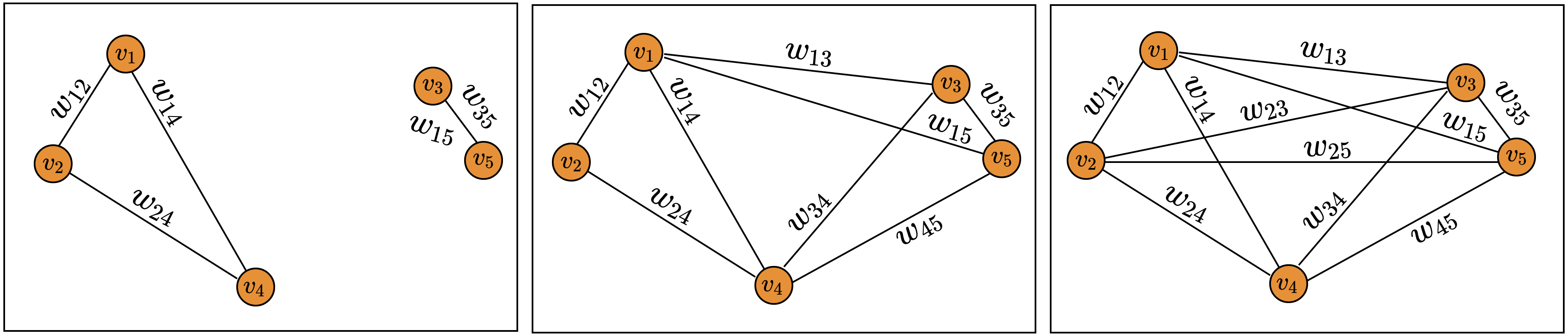
What spectral embedding tells us?

1. If 0 is the eigenvalue of L with k different eigenvectors, i.e., $0 = \lambda_1 = \lambda_2 = \cdots = \lambda_k$ then has k connected components

2. If the graph is connected, $\lambda_2 > 0$, the so-called *algebraic connectivity* of G.

The corresponding eigenvector is called *Fiedler vector*.

$0 = \lambda_1 = \lambda_2 < \lambda_3 \leq \cdots \leq \lambda_n$ | $0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_n$ | $0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_n$



G_1 | G_2 | G_3

The greater λ_2 the more connected G is.

$\boxed{\lambda_2(G_2) < \lambda_2(G_3)}$

Spectral Clustering

Resources

[Fiedler, 1973]

[de Abreu, 2006]

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

Bipartitioning via Spectral Decomposition

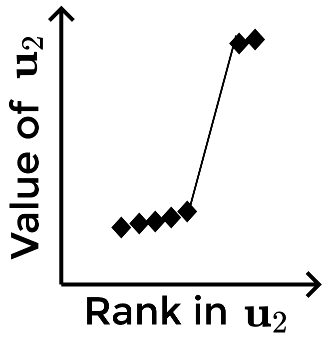
Take the second eigenvector $\mathbf{u}_2$ of graph Laplacian $\mathbf{L}$, the algebraic connectivity of G

➡ The smaller λ_2, the better quality of partitioning

For each node i in G assign $\mathbf{u}_2(v_i)$ to the respective node

Bipartition the graph into two clusters by choosing a splitting point.

➡ Naïve approaches: Split at 0 or median value



Split at 0

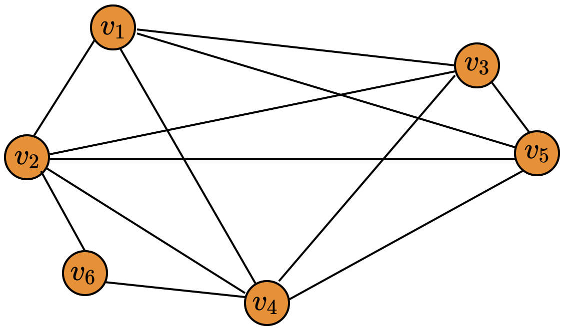
i	$\mathbf{u}_2(v_i)$
1	0.3
2	0.1
3	-0.2
4	0.15
5	-0.15
6	0.2

Cluster A

i	$\mathbf{u}_2(v_i)$
1	0.3
2	0.1
4	0.15
6	0.2

Cluster B

i	$\mathbf{u}_2(v_i)$
3	-0.2
5	-0.15



Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

Bipartitioning via Spectral Decomposition

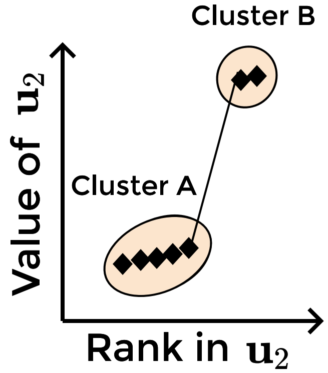
Take the second eigenvector $\mathbf{u}_2$ of graph Laplacian $\mathbf{L}$, the algebraic connectivity of G

➡ The smaller λ_2, the better quality of partitioning

For each node i in G assign $\mathbf{u}_2(v_i)$ to the respective node

Bipartition the graph into two clusters by choosing a splitting point.

➡ Naïve approaches: Split at 0 or median value



Split at 0

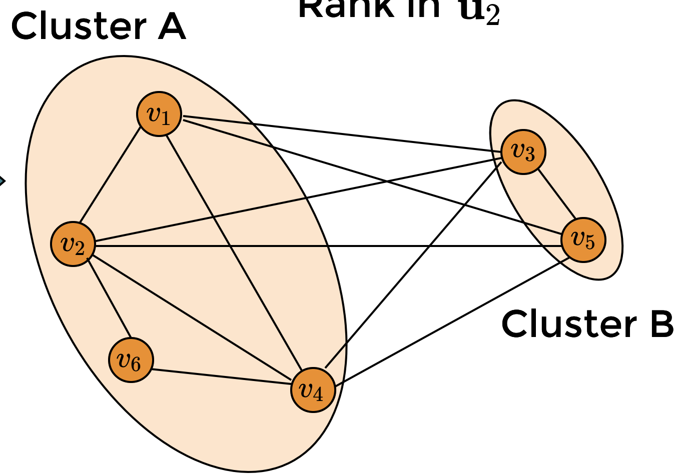
i	$\mathbf{u}_2(v_i)$
1	0.3
2	0.1
3	-0.2
4	0.15
5	-0.15
6	0.2

Cluster A

i	$\mathbf{u}_2(v_i)$
1	0.3
2	0.1
4	0.15
6	0.2

Cluster B

i	$\mathbf{u}_2(v_i)$
3	-0.2
5	-0.15



Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

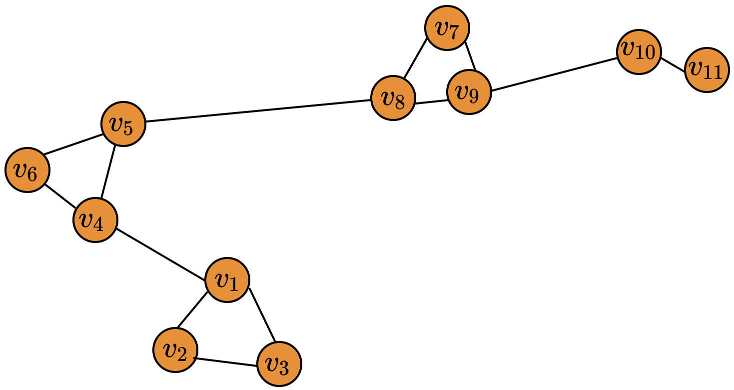
Partitioning via Spectral Decomposition

How to partition a graph into k clusters?

Two approaches:

1. Recursive bi-partitioning [Hagen et al. 1992]

   How does $\mathbf{u}_2(v_i)$ look when there are more than two clusters?



2. Cluster multiple eigenvectors [Shi-Malik, 2000; Ng-Jordan-Weiss, 2002]

   Next topic

Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

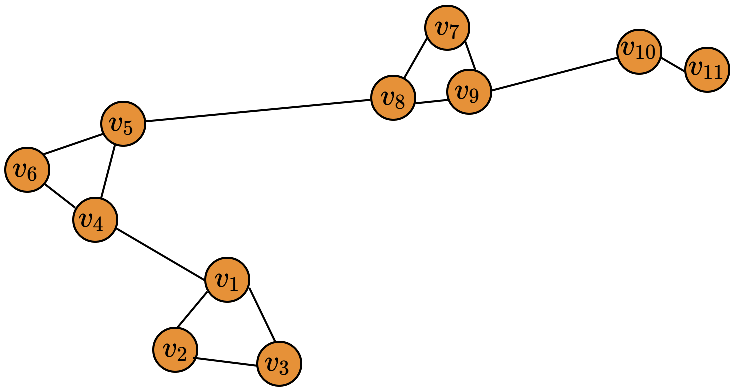
Partitioning via Spectral Decomposition

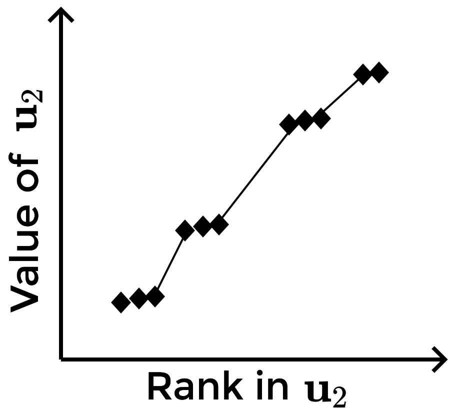
How to partition a graph into k clusters?

Two approaches:

1. Recursive bi-partitioning [Hagen et al. 1992]

How does $\mathbf{u}_2(v_i)$ look when there are more than two clusters?





2. Cluster multiple eigenvectors [Shi-Malik, 2000; Ng-Jordan-Weiss, 2002]

Next topic

Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

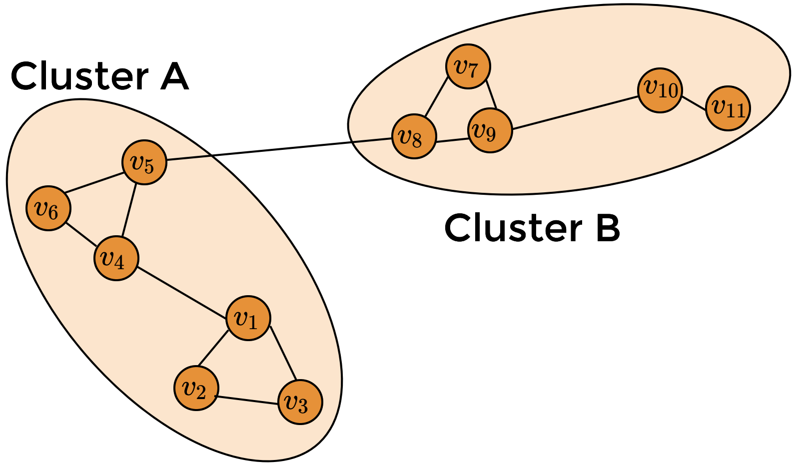
Partitioning via Spectral Decomposition

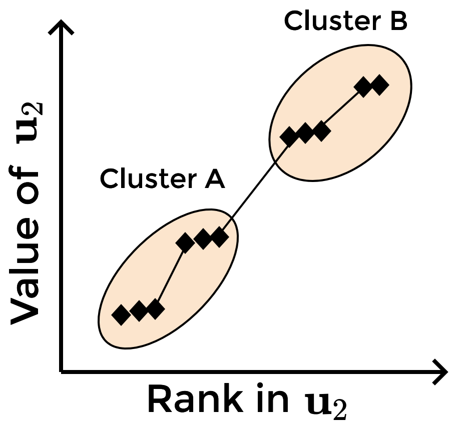
How to partition a graph into k clusters?

Two approaches:

1. Recursive bi-partitioning [Hagen et al. 1992]

How does $\mathbf{u}_2(v_i)$ look when there are more than two clusters?





2. Cluster multiple eigenvectors [Shi-Malik, 2000; Ng-Jordan-Weiss, 2002]

Next topic

Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

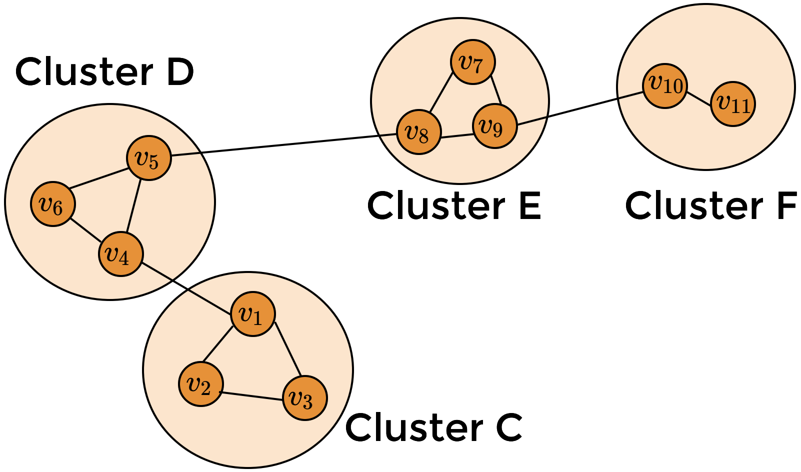
Partitioning via Spectral Decomposition

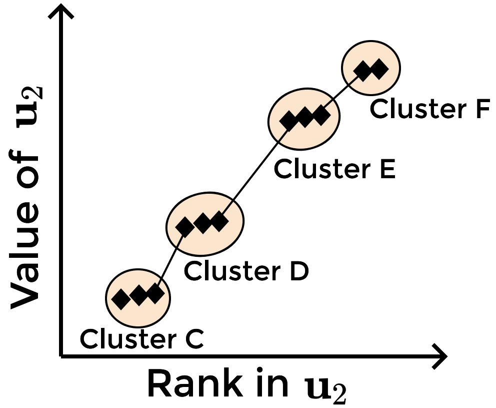
How to partition a graph into k clusters?

Two approaches:

1. Recursive bi-partitioning [Hagen et al. 1992]

How does $\mathbf{u}_2(v_i)$ look when there are more than two clusters?





2. Cluster multiple eigenvectors [Shi-Malik, 2000; Ng-Jordan-Weiss, 2002]

Next topic

Spectral Clustering

Spectrum of the Laplacian
$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

Partitioning via Spectral Decomposition

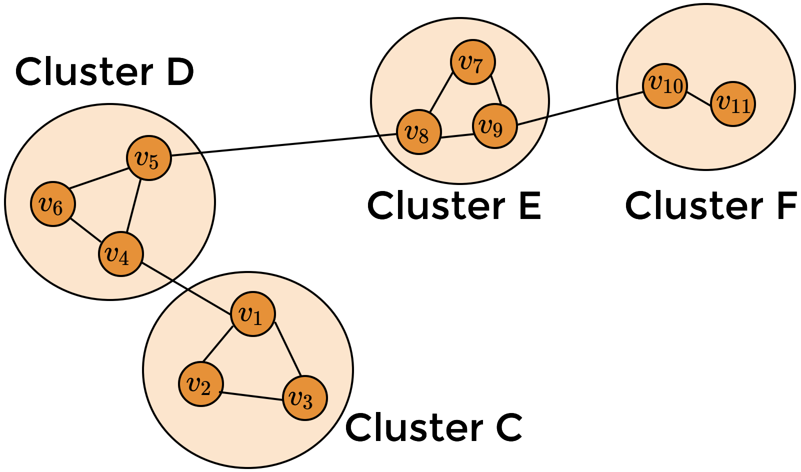
How to partition a graph into k clusters?

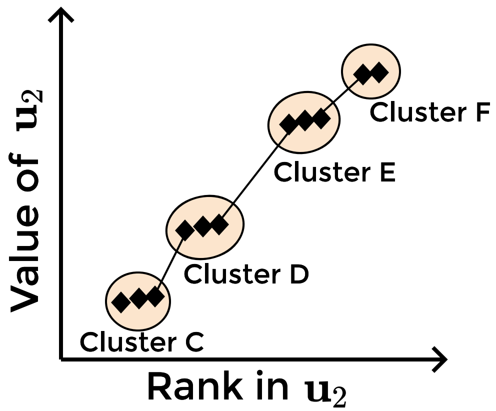
Two approaches:

Disadvantages: unstable & inefficient

1. Recursive bi-partitioning [Hagen et al. 1992]

How does $\mathbf{u}_2(v_i)$ look when there are more than two clusters?





2. Cluster multiple eigenvectors [Shi-Malik, 2000; Ng-Jordan-Weiss, 2002]

Next topic

Spectral Clustering

Dimensionality reduction

Cluster using multiple eigenvectors

- Embed the data into a low dimensional space using eigenvectors
- Apply a clustering method, i.e., k-means

Spectral Embedding via graph Laplacian (unnormalized)

1. Compute the eigendecomposition $L = D - A$

2. Select the k smallest eigenvalues $\lambda_1 \leq \cdots \leq \lambda_k$

3. Form $n \times k$ matrix $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_k]$ such that

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1(v_1) & \cdots & \mathbf{u}_k(v_1) \\ \vdots & & \vdots \\ \mathbf{u}_1(v_n) & \cdots & \mathbf{u}_k(v_n) \end{bmatrix}$$

4. Cluster the normalized rows of $\mathbf{U}$ into k clusters using k-means

Spectral Clustering

Cluster using multiple eigenvectors

- Embed the data into a low dimensional space using eigenvectors
- Apply a clustering method, i.e., k-means

Spectral Embedding via graph Laplacian (normalized & symmetric)

1. Compute the eigendecomposition $L_{norm} = D^{-1/2} L D^{-1/2}$

2. Select the k smallest eigenvalues $\lambda_1 \leq \cdots \leq \lambda_k$

3. Form $n \times k$ matrix $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_k]$ such that

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1(v_1) & \cdots & \mathbf{u}_k(v_1) \\ \vdots & & \vdots \\ \mathbf{u}_1(v_n) & \cdots & \mathbf{u}_k(v_n) \end{bmatrix}$$

4. Normalize each row of $\mathbf{U}$ to norm 1

5. Cluster the normalized rows of $\mathbf{U}$ into k clusters using k-means

Preferable & commonly used in recent papers

See tutorial slides by [Ding 2004]

Spectral Clustering

Graph cut point of view [Luxburg 2007]

How to cut a graph to partition data points into clusters?

Let $W(A,B) := \sum_{i\in A, j\in B} w_{ij}$ and $\bar{A}$ is the complement of A

Also let k be the number of clusters we want to partition our data into

Mincut minimizes $cut(A_1,\cdots,A_k) := \sum_{i=1}^{k} W(A_i,\bar{A}_i)$ → **problems?**

RatioCut minimizes $RatioCut(A_1,\cdots,A_k) := \sum_{i=1}^{k} \frac{W(A_i,\bar{A}_i)}{|A_i|} = \sum_{i=1}^{k} \frac{cut(A_i,\bar{A}_i)}{|A_i|}$, $|A_i|$ is #vertices in A_i

Ncut minimizes $Ncut(A_1,\cdots,A_k) := \sum_{i=1}^{k} \frac{W(A_i,\bar{A}_i)}{vol(A_i)} = \sum_{i=1}^{k} \frac{cut(A_i,\bar{A}_i)}{vol(A_i)}$ $vol(A_i)$ is weights of edges in A_i

Unnormalized Laplacian ≈ RatioCut

Normalized Laplacian ≈ NCut

Spectral Clustering

Advantages

1. No strong assumption on shape and statistics of clusters, hence resulting better clustering performance for connected clusters than k-means.
2. Easy to implement
3. Reasonably fast for sparse data sets of several thousand elements.

Disadvantages

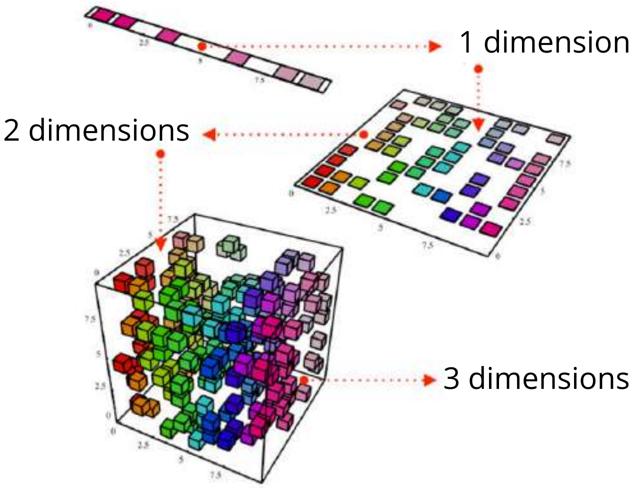
1. Computationally expensive for large data set due to eigendecomposition
2. Use of k-means in the last step may lead unstability due to the sensitivity of k-means to the initial centroids *(Homework 5, Problem 2)*

Recap (Dimensionality Reduction)

Suppose $x_i \in \mathbb{R}^d, i \in \{1, \cdots, n\}$ and we want to learn a mapping $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ with $k << d$

where we can reconstruct the data with little loss of information

Motivation: Visualization, compression, unsupervised feature discovery



Key question: How to choose the mapping f ?

linear? nonlinear?

Linear dimensionality reduction?

Principal Component Analysis (PCA)

image credit: https://bigsnarf.wordpress.com/

Recap (Pricipal Component Analysis)

Recall from the lecture that PCA is a linear dimensionality reduction technique

$$\mathbf{z}_i = \mathbf{W}^T \mathbf{x}_i, \quad \mathbf{W} \in \mathbb{R}^{d \times k}$$

which minimizes the reconstruction error $\sum_i \|\mathbf{W}\mathbf{z}_i - \mathbf{x}_i\|_2^2$

Solution to PCA. For centered data: $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$

$$\mathbf{W}^* = (\mathbf{v}_1 | \cdots | \mathbf{v}_k) \quad \text{and} \quad \mathbf{z}_i = \mathbf{W}^* \mathbf{x}_i \quad \text{where} \quad \Sigma = \sum_{i=1}^{d} \lambda_i \mathbf{v}_i \mathbf{v}_i^T, \quad \lambda_1 \geq \cdots \geq \lambda_d \geq 0$$

(4 points) (iii) *Short questions on dimensionality reduction.* Assume we apply PCA with k principal components to a data set $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ where $\mathbf{x}_i \in \mathbb{R}^d$.

1. If $k < d$ we can *exactly* reconstruct x_i from k principal components.

   ☐ True ☑ False

2. If $k = d$ we can *exactly* reconstruct x_i from k principal components.

   ☑ True ☐ False

3. If $k > n$ we can *exactly* reconstruct x_i from k principal components.

   ☑ True ☐ False

4. Suppose $\mathbf{X}$ is the $n \times d$ data matrix. Then, the eigenvectors of $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$ are the same.

   ☐ True ☑ False

Final Exam 2018

Non-linear Dimensionality Reduction - Kernel PCA

Motivation. How to capture non-linear manifold structures?

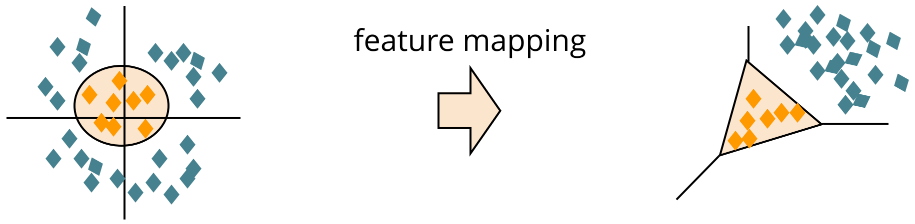
$$k(\mathbf{x},\mathbf{z}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)^T(z_1^2, \sqrt{2}z_1z_2, z_2^2)$$
$$= (\mathbf{x}^T\mathbf{z})^2$$

Kernel PCA. Apply Kernel method to PCA!

Map data to higher dimensions where contain linear patterns

Data becomes linearly separable in the new feature space

Example. Feature mapping function $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ $\quad (x_1, x_2) \rightarrow (z_1, z_2, z_3) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$



Data in low dimensional space

Data in high dimensional space

The feature mapping ϕ is not necessary to know! We deal with kernel functions instead ☺

Recall from the class that kernel principal components $\alpha^{(1)}, \cdots, \alpha^{(k)} \in \mathbb{R}^n$ are given by

$\alpha^{(i)} = \frac{1}{\sqrt{\lambda_i}}\mathbf{v}_i$ where $\lambda_i, \mathbf{v}_i, i = \{1, \cdots, n\}$ are obtained by eigendecomposition of $\mathbf{K} = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^T$

A new point x is projected as $\boxed{z_i = \sum_{j=1}^{n} \alpha_j^{(i)} k(x, x_j)}$

Kernel PCA vs. Spectral Clustering

Consider the Kernel matrix $\mathbf{K}_{ij} = k(x_i, x_j)$

In Kernel-PCA, we compute the eigenvector $\mathbf{Kv} = \lambda \mathbf{v}$

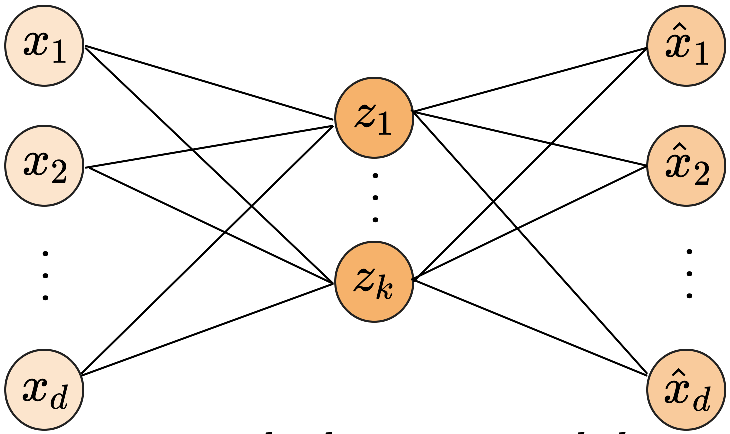
Generalized eigenvector $\mathbf{Kz} = \lambda \mathbf{Dz}, \;\; \mathbf{D} = diag(\sum_j k(x_1, x_j), \cdots, \sum_j k(x_n, x_j))$ $\boxed{spectral\ clustering}$

"There is a clear equivalence between the spectral embedding methods used in spectral clustering and Laplacian eigenmaps with the projection computed by the kernel PCA method." [Bengio et al. 2004]

Linear Autoencoder

Given data points $\mathbf{x}_i \in \mathbb{R}^d, i = 1, \cdots, n$ compress data into k-dimensional representation $k \leq d$.

Linear auto-encoding with a single hidden layer



How to choose E and D?

Optimal solution satisfies:

$$\boxed{\min \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{D}\mathbf{E}\mathbf{x}_i\|_2^2}$$

$\boxed{\mathbf{E} = \mathbf{U}_k^T}$ $E \in \mathbb{R}^{k \times d}$ ***encoder*** $D \in \mathbb{R}^{d \times k}$ ***decoder***

PCA

$\boxed{\mathbf{D} = \mathbf{U}_k} \longrightarrow \boxed{\mathbf{D}\mathbf{E}\mathbf{X} = \mathbf{U}_k \Lambda_k \mathbf{V}_k^T}$

Eckart-Young theorem: Let $\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ and SVD of $\mathbf{X} = \mathbf{U}\Lambda\mathbf{V}$. For $k \leq \min(n, d)$

$$\underset{\hat{\mathbf{X}}:\text{rank}(\hat{\mathbf{X}})=\mathbf{k}}{\arg\min} \|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 = \mathbf{U}_k \Lambda_k \mathbf{V}_k^T$$

More details in Deep Learning class

Non-linear Autoencoder

Use neural network autoencoders to learn the nonlinear mapping for dimensionality reduction through an identity function

$$x \approx f(x;\theta)$$

Properties of $f(\cdot)$: approximates the identitity function & performs compression

How to pick $f(\cdot)$: Composition of two nonlinear functions $f_1(\cdot)$ and $f_2(\cdot)$ such that

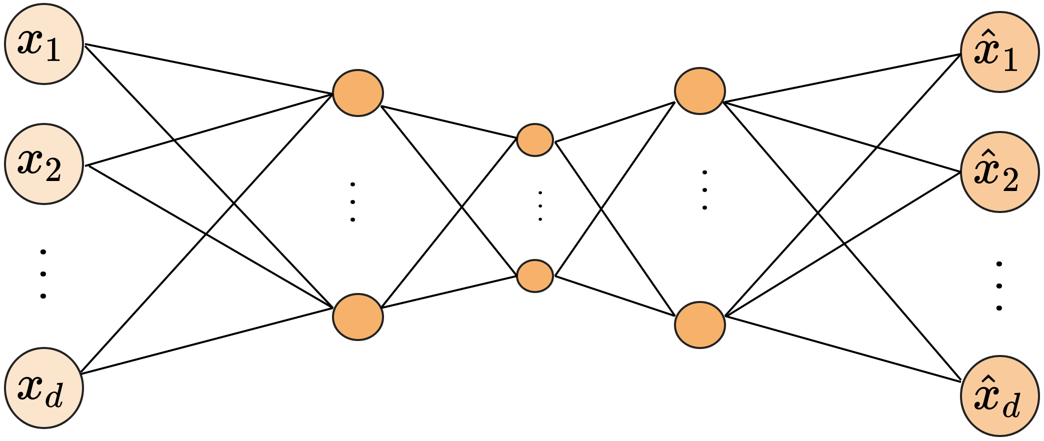
$$\boxed{f(x;\theta) = f_2(f_1(x;\theta_1);\theta_2)}$$

where $\underbrace{f_1(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^k}_{\text{encoder}}$ and $\underbrace{f_2(\cdot) : \mathbb{R}^k \rightarrow \mathbb{R}^d}_{\text{decoder}}$

How to learn $f_1(\cdot)$ and $f_2(\cdot)$? Use Neural Networks!

Non-linear generalization of PCA.

Non-linear Autoencoder



$f_1 = F_1 \circ \cdots \circ F_l : \mathbb{R}^d \to \mathbb{R}^k, \mathbf{x} \to \mathbf{z}$

encoder

$f_2 = F_L \circ \cdots \circ F_{l+1} : \mathbb{R}^k \to \mathbb{R}^d, \mathbf{z} \to \hat{\mathbf{x}}$

decoder

How to train autoencoders?

Optimize the weights such that $\boxed{\hat{\mathbf{x}} = f(\mathbf{x}; \mathbf{w}^{(1)}, \mathbf{w}^{(2)}) = f_2\big(f_1(\mathbf{x}; \mathbf{w}^{(1)}); \mathbf{w}^{(2)}\big) \approx \mathbf{x}}$

e.g., $\boxed{\min_{\mathbf{W}} \sum_{i=1}^{n} \|\mathbf{x}_i - f(\mathbf{x}_i; \mathbf{W})\|_2^2}$ via backpropagation.

Autoencoders vs. PCA

original autoencoder PCA

image credit: http://nghiaho.com

See js demo for digit images:

https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoenc

Questions?

References

[Bengio et al. 2003] Spectral Clustering and Kernel PCA are Learning Eigenfunctions, *Technical report, Département d'informatique et recherche opérationnelle, Université de Montréal.*

[Ding 2004] A Tutorial on Spectral Clustering. *Talk presented at ICML (Slides are available: http://crd.lbl.gov/~cding/Spectral/)*

[Luxburg 2007] A Tutorial on Spectral Clustering. *Statistics and Computing, 17 (4)*

[Fiedler 1973] Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal, vol. 23 (1973), issue 2, pp. 298-305*

[de Abreu 2006] Old and new results on algebraic connectivity of graphs. *Linear Algebra and its Applications Volume 423, Issue 1, Pages 53-73*

[Hagen et al. 1992] New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Computer Aided Design, 11(9), 1074-1085.*

[Shi-Malik 2000] Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8), 888 – 905.*

[Ng-Jordan-Weiss 2002] On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems 14 (pp. 849 – 856)*