CSE 590
Data Science Fundamentals

Cluster Analysis II

Klaus Mueller

Computer Science Department
Stony Brook University and SUNY Korea

Lecture	Topic	Projects
1	Intro, schedule, and logistics	
2	Data Science components and tasks	
3	Data types	Project #1 out
4	Introduction to R, statistics foundations	
5	Introduction to D3, visual analytics	
6	Data preparation and reduction	
7	Data preparation and reduction	Project #1 due
8	Similarity and distances	Project #2 out
9	Similarity and distances	
10	Cluster analysis	
11	Cluster analysis	
12	Pattern mining	Project #2 due
13	Pattern mining	
14	Outlier analysis	
15	Outlier analysis	Final Project proposal due
16	Classifiers	
17	Midterm	
18	Classifiers	
19	Optimization and model fitting	
20	Optimization and model fitting	
21	Causal modeling	
22	Streaming data	Final Project preliminary report due
23	Text data	
24	Time series data	
25	Graph data	
26	Scalability and data engineering	
27	Data journalism	
	Final project presentation	Final Project slides and final report due

Overview

We will discuss

- Spectral clustering (Shi and Malik, 2000)
- DBSCAN (Ester et al., 1996)
- t-SNE (van der Maaten and Hinton, 2008)

Spectral Clustering

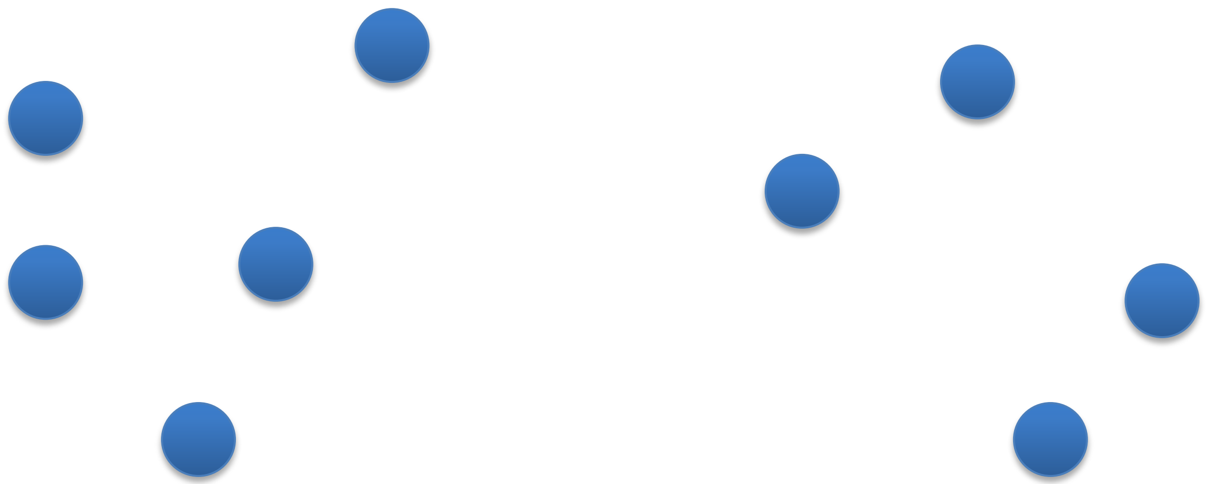
Some clustering don’t lend themselves to a “centroid” based definition of a cluster

These kinds of clusters are defined by points that are close **any** member in the cluster, rather than the **average** member of the cluster

This and the next 22 slides are due to Andrew Rosenberg, CUNY

Graph Representation

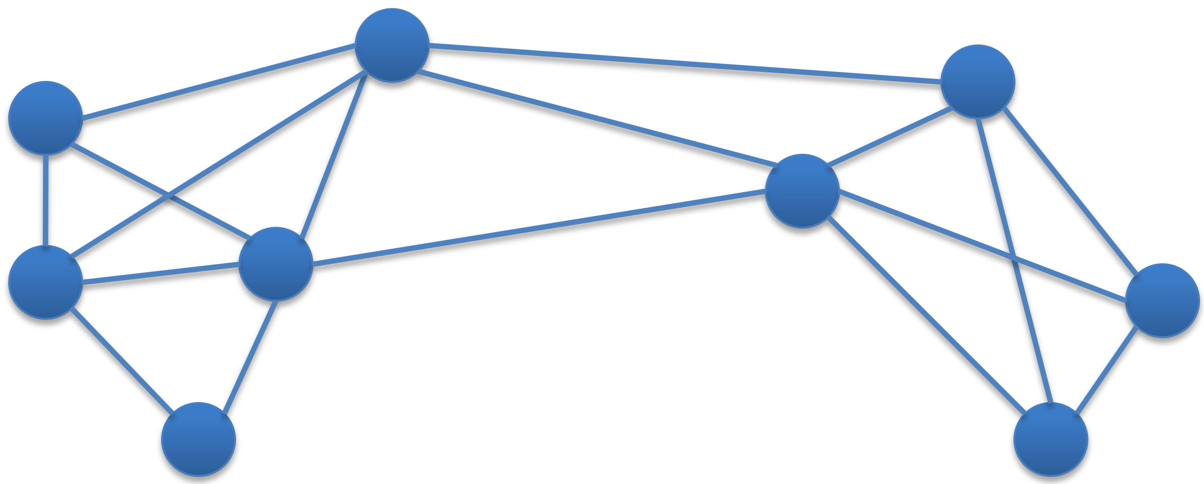
We can represent the relationships between data points in a graph.



Graph Representation

We can represent the relationships between data points in a graph.

Weight the edges by the similarity between points

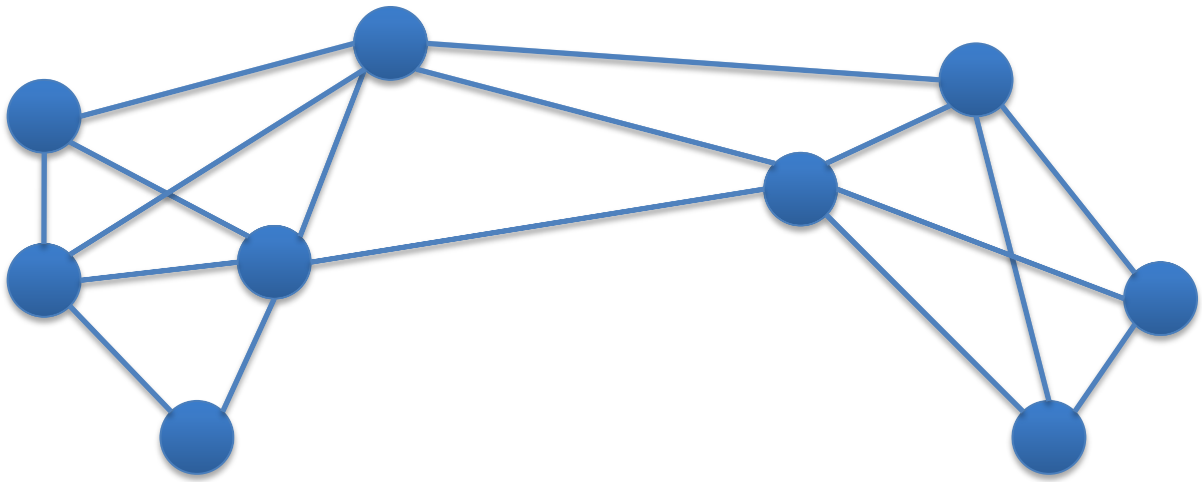


Representing data in a graph

What is the best way to calculate similarity between two data points?

Distance based:

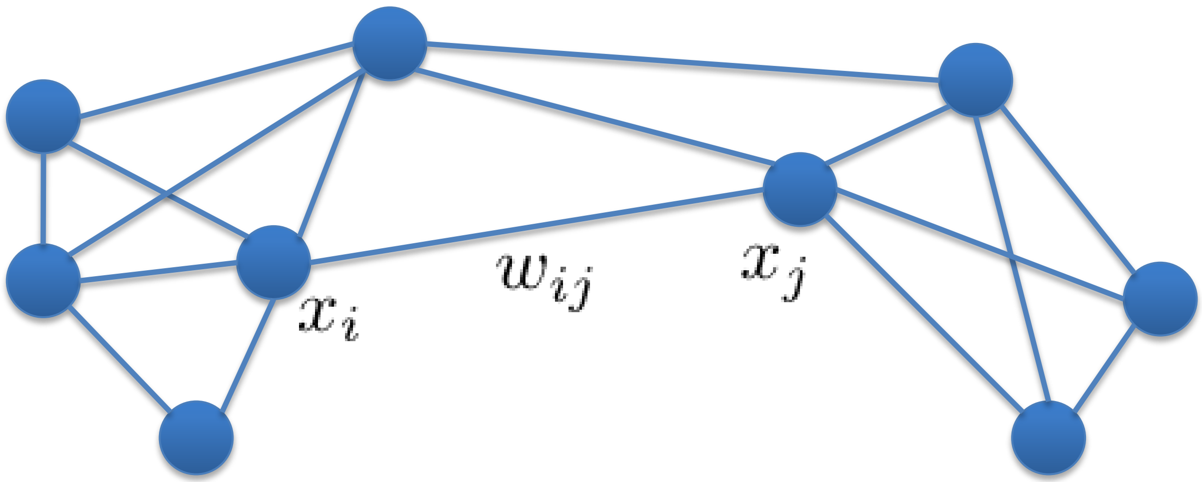
$$d(x_i, x_j) = \exp\left\{\frac{\|x_i - x_j\|}{\sigma^2}\right\}$$



Graphs

Nodes and Edges

Edges can have weights associated with them



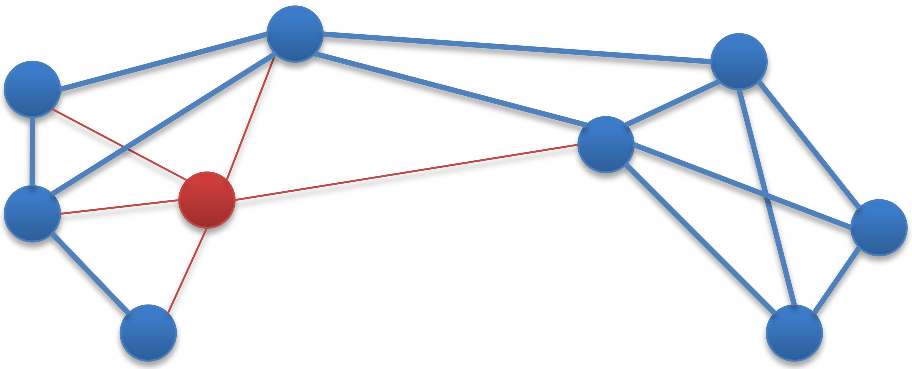
Here the weights correspond to **pairwise affinity**

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

Graphs

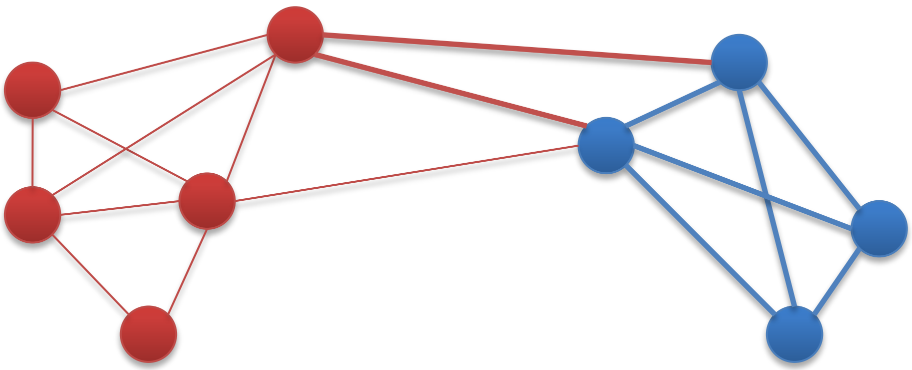
Degree

$$D(x_i) = \sum_{j \in V} w_{ij}$$



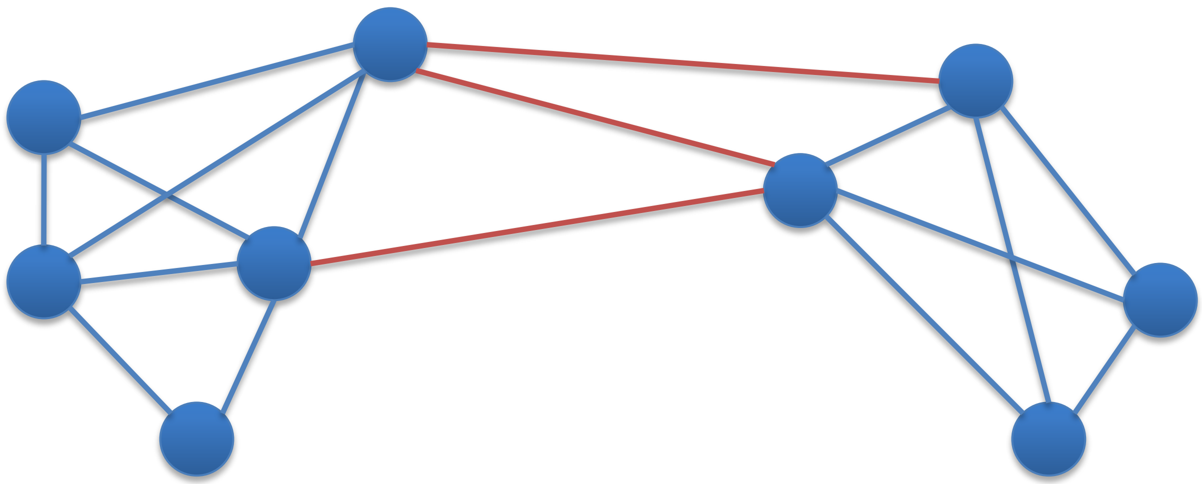
Volume of a set

$$Vol(C) = \sum_{i \in C} D(x_i)$$



Graph Cuts

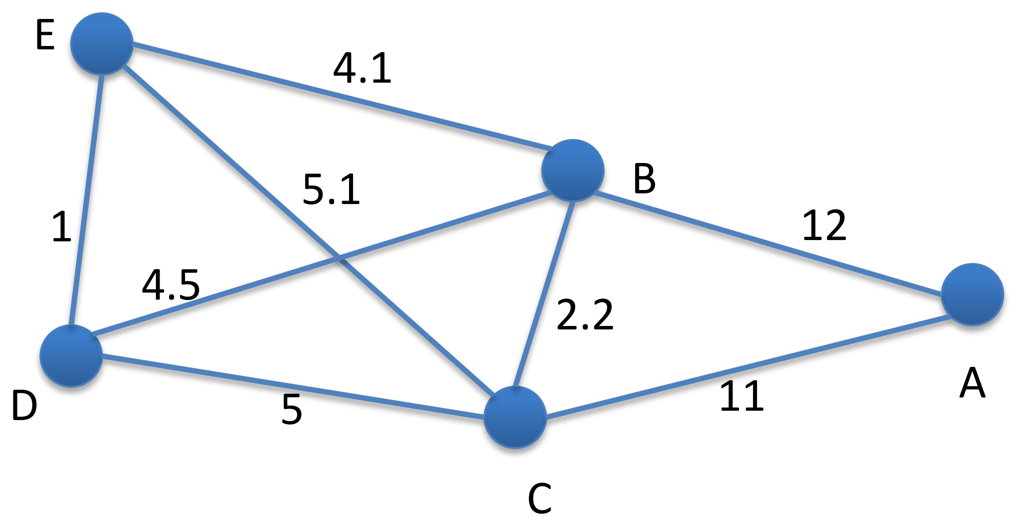
The **cut** between two subgraphs is calculated as follows



$$Cut(C_1, C_2) = \sum_{i \in C_1} \sum_{j \in C_2} w_{ij}$$

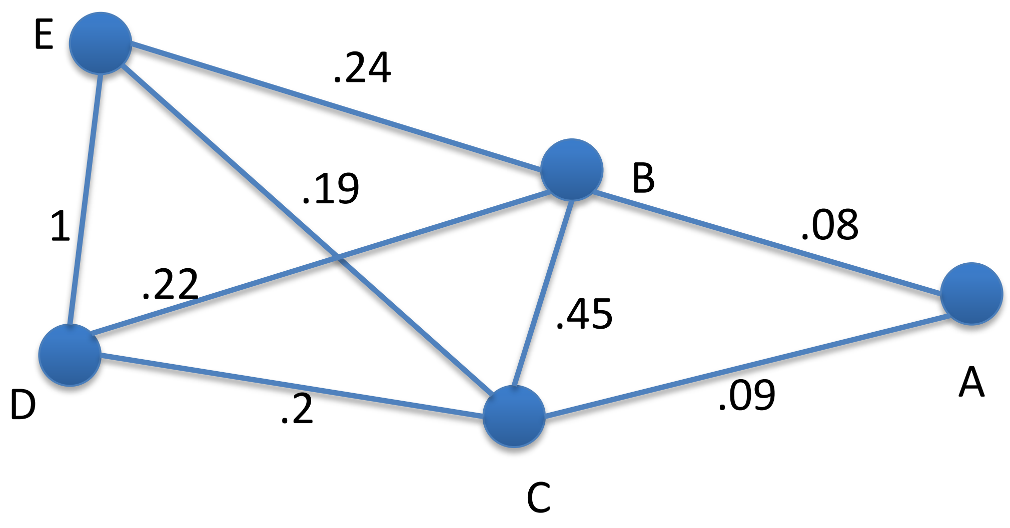
Graph Examples – Distance

Height	Weight
20	5
8	6
9	4
4	4
4	5



Graph Examples – Similarity

Height	Weight
20	5
8	6
9	4
4	4
4	5



Intuition

The minimum cut of a graph identifies an optimal partitioning of the data.

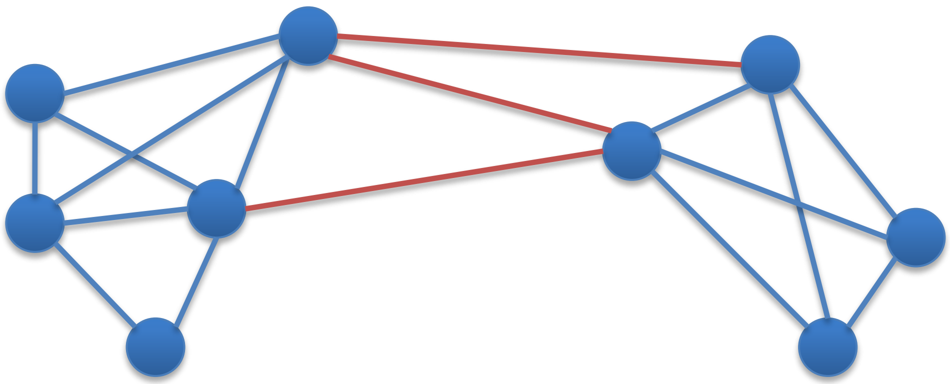
Spectral Clustering

- Recursively partition the data set
    - Identify the minimum cut
    - Remove edges
    - Repeat until k clusters are identified

Graph Cuts

Minimum (bipartitional) cut

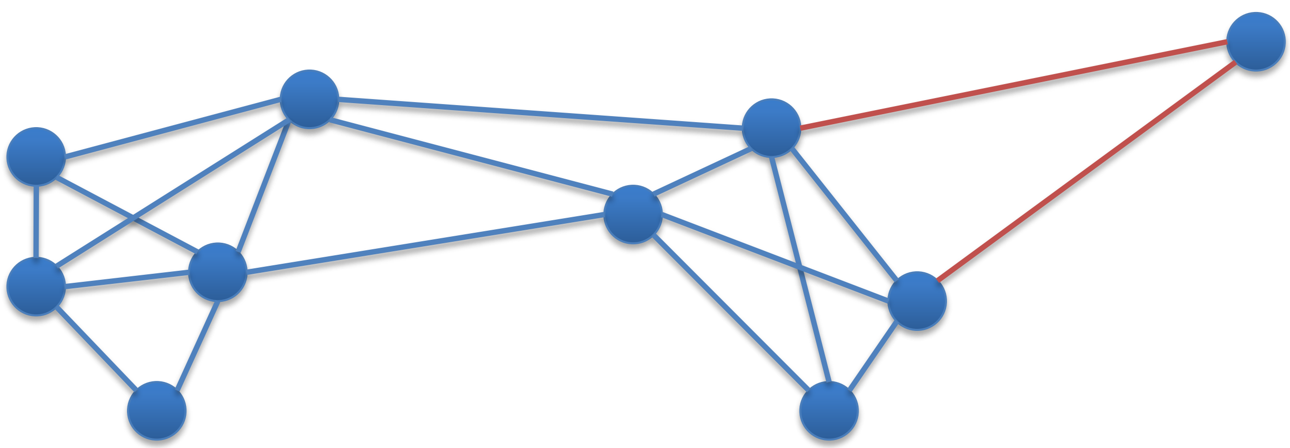
$$\min Cut(C_1, C_2) = \sum_{i \in C_1} \sum_{j \in C_2} w_{ij}$$



Graph Cuts

Minimum (bipartitional) cut

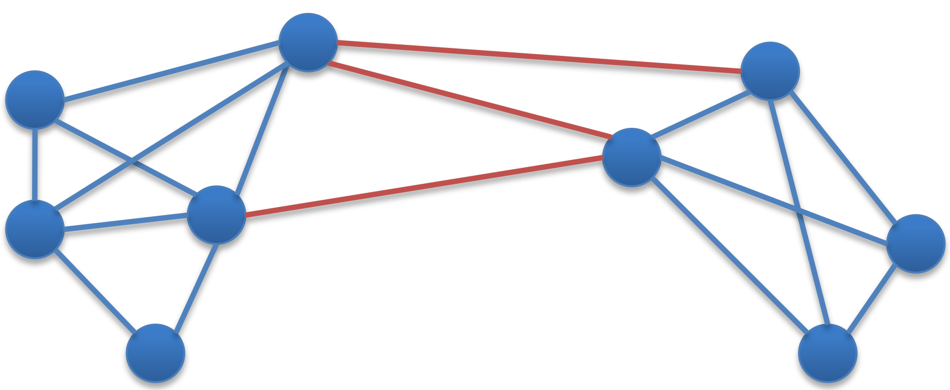
$$\min Cut(C_1, C_2) = \sum_{i \in C_1} \sum_{j \in C_2} w_{ij}$$



Graph Cuts

Minimal (bipartitional) normalized cut.

$$\min \frac{Cut(C_1, C_2)}{Vol(C_1)} + \frac{Cut(C_1, C_2)}{Vol(C_2)} = \min \left(\frac{1}{Vol(C_1)} + \frac{1}{Vol(C_2)} \right) Cut(C_1, C_2)$$



Unnormalized cuts are attracted to outliers.

Graph definitions

ε-neighborhood graph

- Identify a threshold value, ε, and include edges if the affinity between two points is greater than ε.

k-nearest neighbors

- Insert edges between a node and its k-nearest neighbors.
- Each node will be connected to (at least) k nodes.

Fully connected

- Insert an edge between every pair of nodes.

Intuition

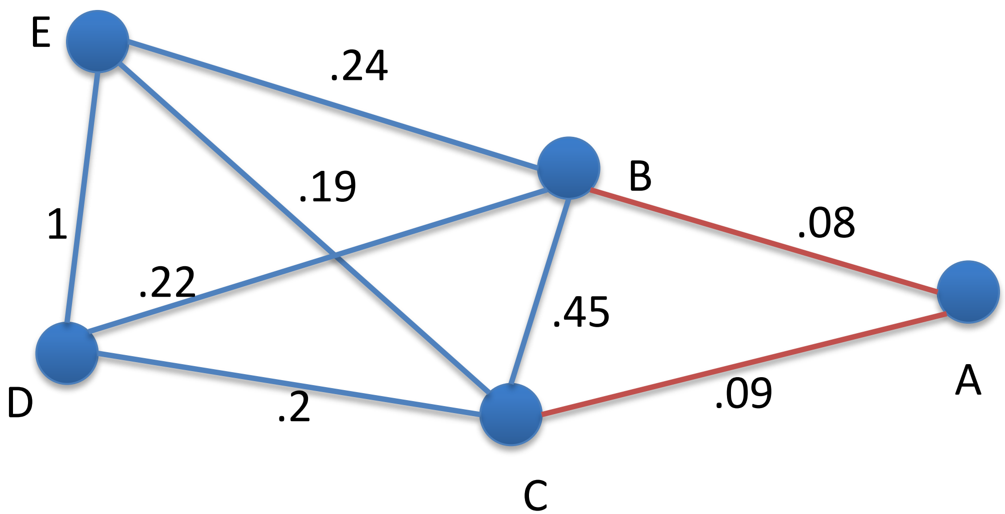
The minimum cut of a graph identifies an optimal partitioning of the data.

Spectral Clustering

- Recursively partition the data set
  - Identify the minimum cut
  - Remove edges
  - Repeat until k clusters are identified

Spectral Clustering Example

Minimum Cut



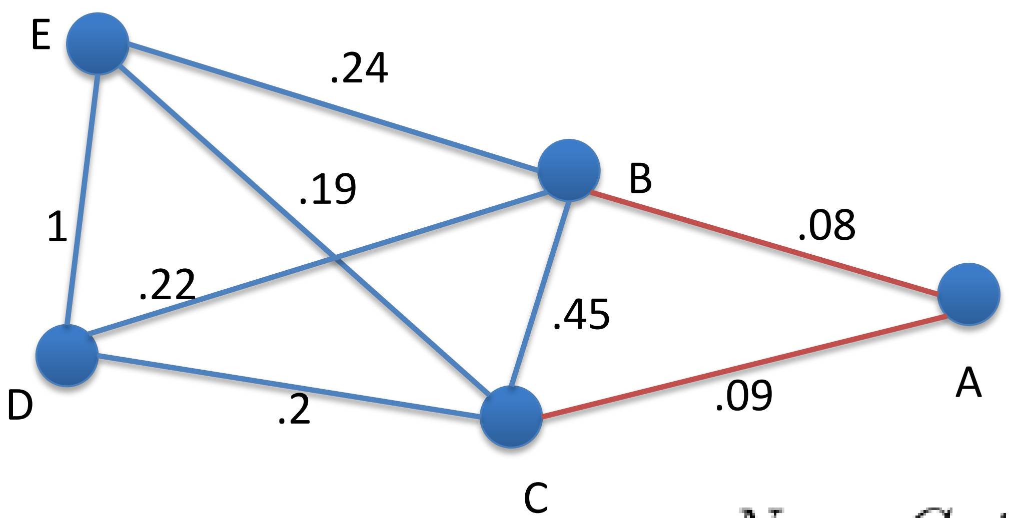
Height	Weight
20	5
8	6
9	4
4	4
4	5

$$Cut(BCDE, A) = 0.17$$

Spectral Clustering Example

Normalized Minimum Cut

$$NormCut(C_1, C_2) = \frac{Cut(C_1, C_2)}{Vol(C_1)} + \frac{Cut(C_1, C_2)}{Vol(C_2)}$$



Height	Weight
20	5
8	6
9	4
4	4
4	5

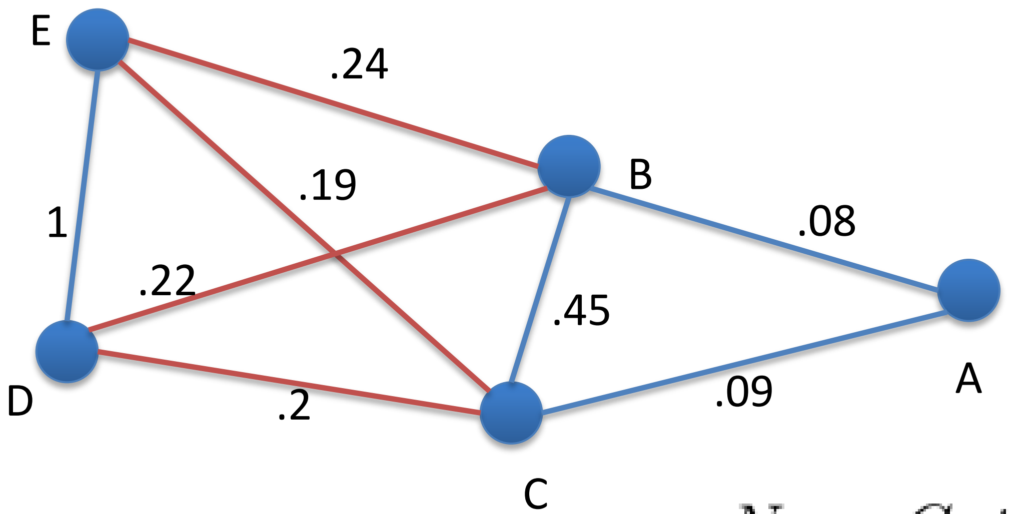
$$NormCut(BCDE, A) = 1.067$$

Spectral Clustering Example

Normalized Minimum Cut

$$NormCut(C_1, C_2) = \frac{Cut(C_1, C_2)}{Vol(C_1)} + \frac{Cut(C_1, C_2)}{Vol(C_2)}$$

Height	Weight
20	5
8	6
9	4
4	4
4	5



$$NormCut(BCDE, A) = 1.067$$

$$NormCut(ABC, DE) = 1.038$$

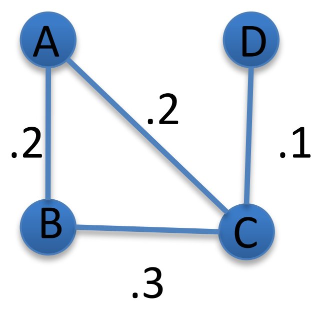
Problem and Solution

Identifying a minimum cut is NP-hard.

- there are efficient approximations using linear algebra.
- based on the Laplacian Matrix L, or **graph Laplacian**

L=D-A

A= affinity matrix



	A	B	C	D
A	.4	.2	.2	0
B	.2	.5	.3	0
C	.2	.3	.6	.1
D	0	0	.1	.1

D= diagonal matrix where $D_{ii} = \sum_j A_{ij}.$

- for each node i sum weights with all of its neighbors ij

Spectral Clustering

Identify eigenvectors of the Laplacian matrix $Lv = \lambda v$

- Eigenvalues of the laplacian are approximate solutions to mincut problem

Perform k-Means on this eigenvector transformation

k Eigenvectors

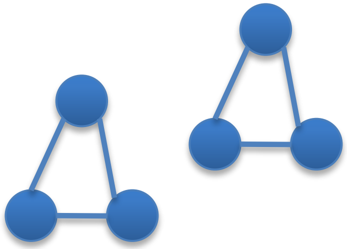
n data points

Each row represents a data point in the eigenvector space

Project back to the initial data representation.

SIMPLE EXAMPLE

Ideal Case



$$Lv = \lambda v$$

1	1	1	0	0	0
1	1	1	0	0	0
1	1	1	0	0	0
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

1	**0**
1	0
1	0
0	1
0	1
0	1

The Graph Laplacian

L = D-W

Positive semi-definite $x^T L x \geq 0$

The lowest eigenvalue is 0, eigenvector is $\vec{1}$

The second lowest contains the solution

- small values indicate good graph partitioning
- The corresponding eigenvector contains the cluster indicator for each data point

Each eigenvector partitions the data set into two clusters.

- The entry in the second eigenvector determines the first cut.
- Subsequent eigenvectors can be used to further partition into more sets.

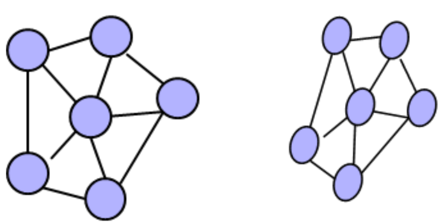
Eigenvectors of the Graph Laplacian

Cluster *E* into *k* clusters

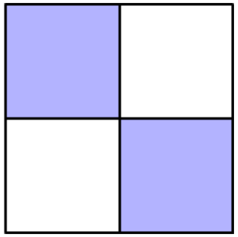
- assign a data point *i* to cluster *j* only if row *i* of *E* was assigned to cluster *j*
- this finalizes the spectral clustering
- in practice need to so some normalizations

Illustrative case:

- two isolated clusters



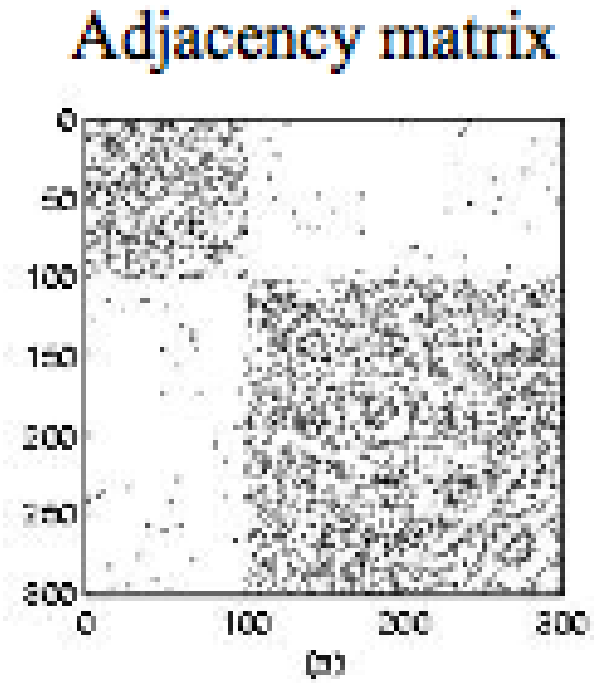
Kernel-linked graph of 2 isolated clusters

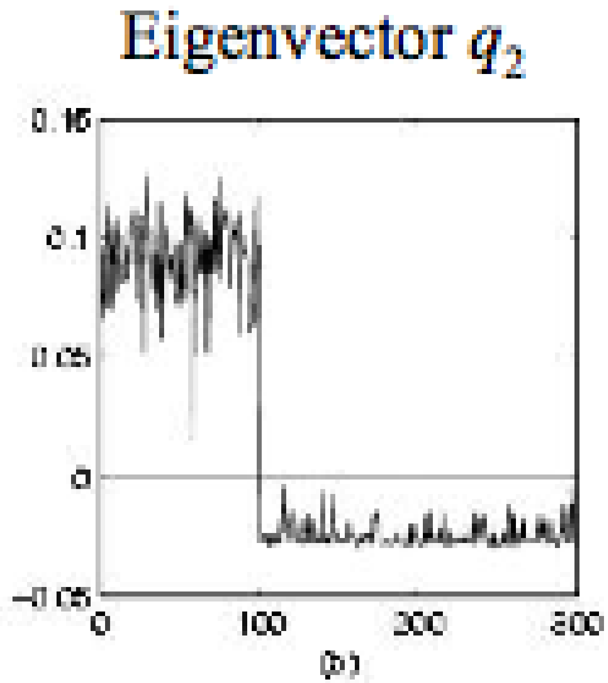


Its adjacency matrix after clustering (stylized)

EXAMPLE

Dense clusters with some sparse connections

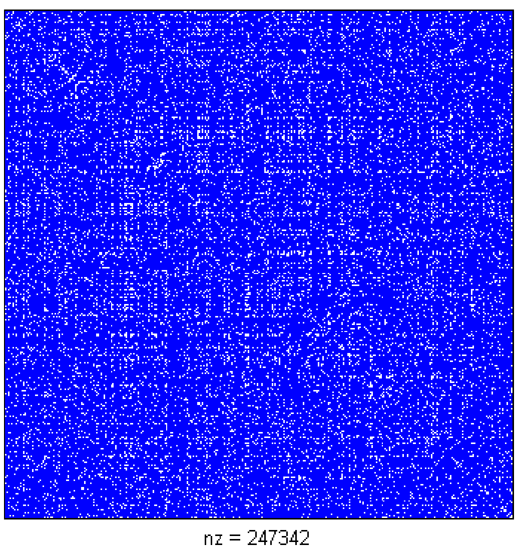




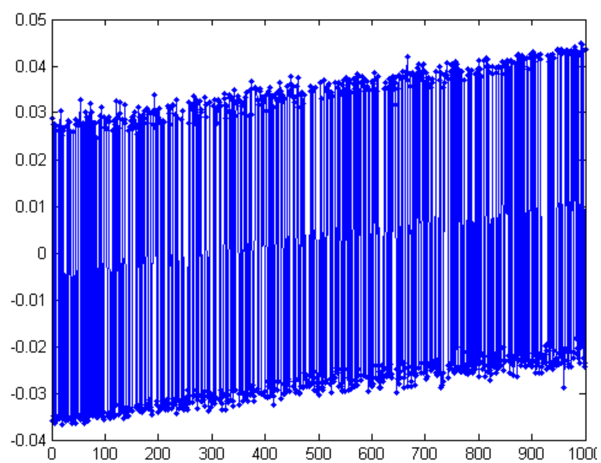
Example 1

Two groups or points

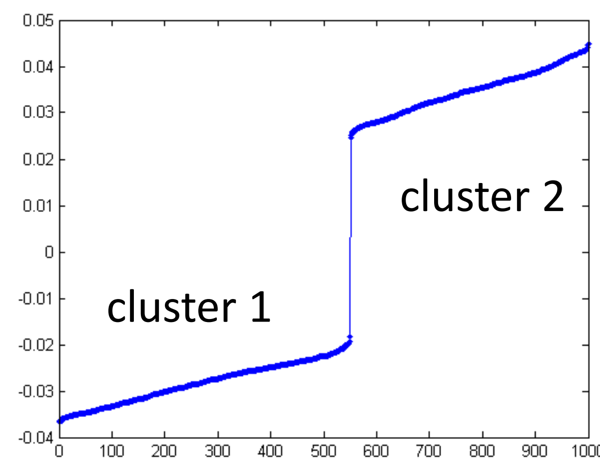
- one larger than the other but rather close
- second eigenvalue relatively large (46.7158) – indicates that a good cut is not to be expected
- sorted second eigenvector has large gap – indicates two clusters



A



unsorted



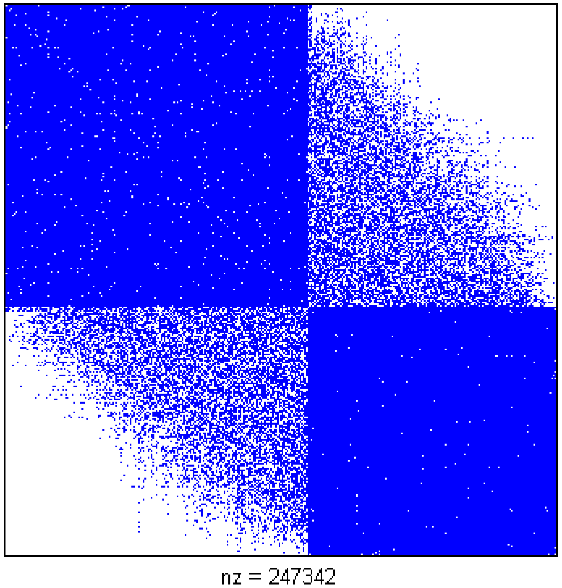
sorted

second Eigenvector

Example 1

Perform the same sorting on A and obtain the following

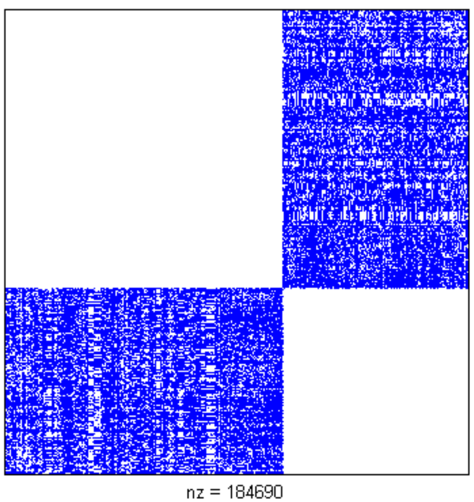
- two clusters
- but no clear cut as predicted by second Eigenvalue



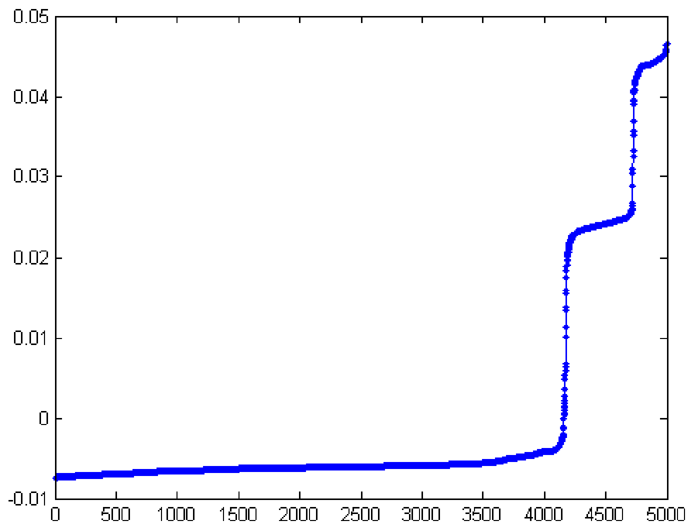
Example 2

Three clusters but better separated

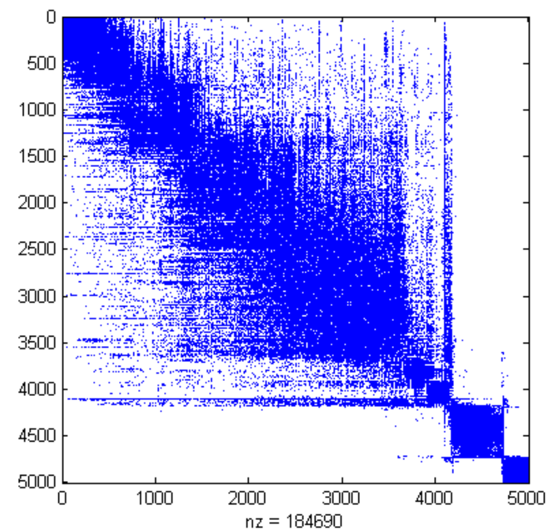
- second Eigenvalue is smaller (0.6031)
- we expect to find some fairly small cuts or rather tight clusters
- sorted second eigenvector has two large gaps – expect 3 clusters



A



sorted second Eigenvector

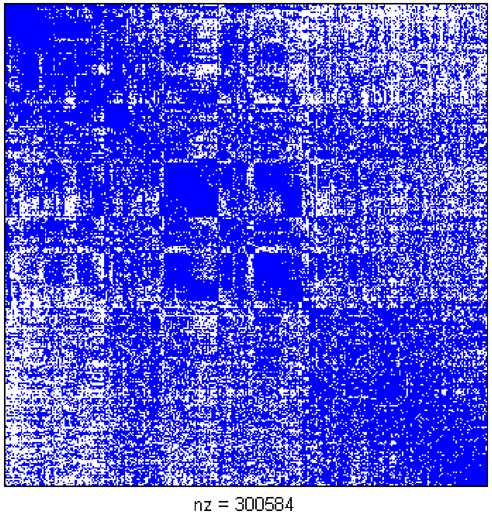


sorted A and three clusters emerge

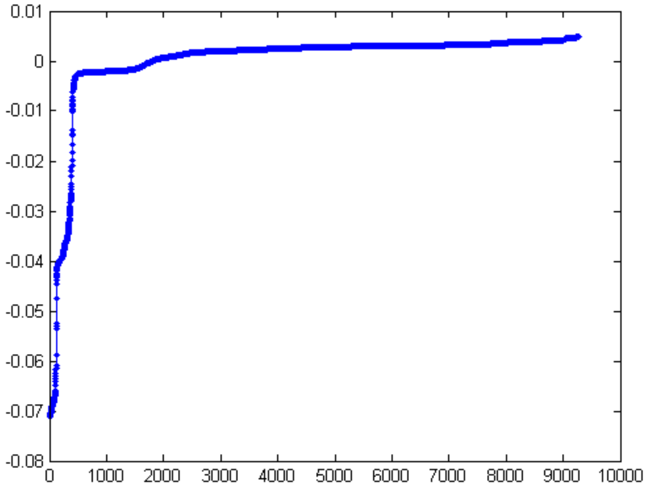
Example 3

Many small clusters

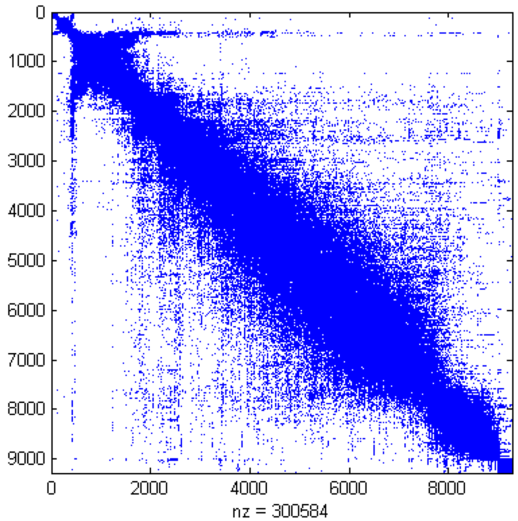
- Eigenvalue analysis shows that second eigenvalue is small (0.0738)
- sorted second Eigenvector shows one revealing gap – the small cluster on the top left of the sorted A



A



sorted second Eigenvector

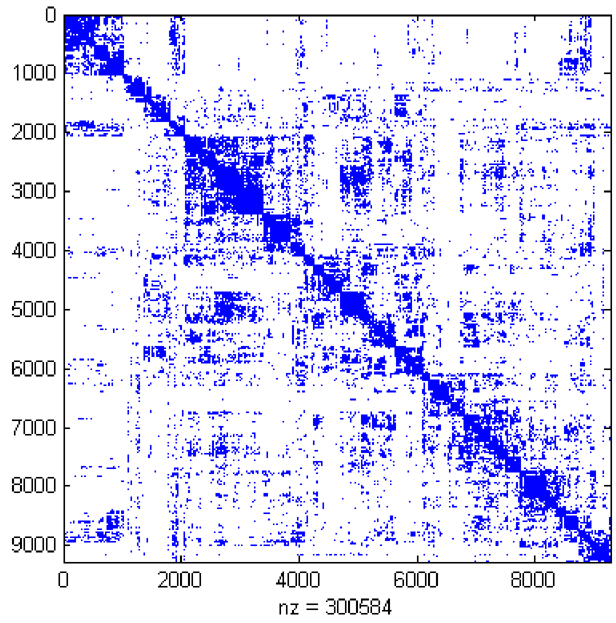


sorted A and 1 cluster on top left emerges

Example 3

To expose the remainder of the structure

- apply the second smallest eigenvector recursively
- use the second smallest eigenvector of the full graph to determine a good way to split the graph into two pieces
- then repeat the process on each subgraph



More Information

See [this webpage]()

Can also cluster the E and use all the Eigenvectors directly

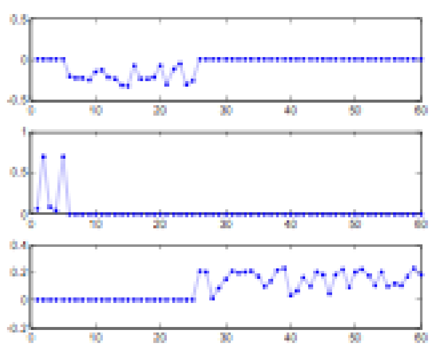
- three class partition

Affinity matrix

$W; A$

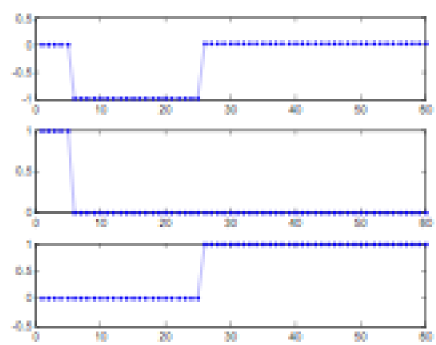
eigenvectors

$V = [v_1, v_2, v_3]$

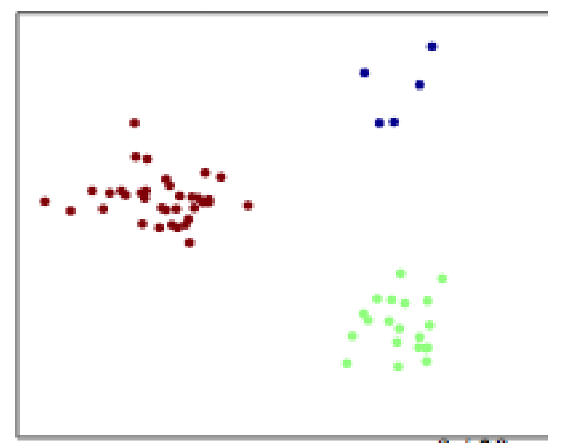


row normalization

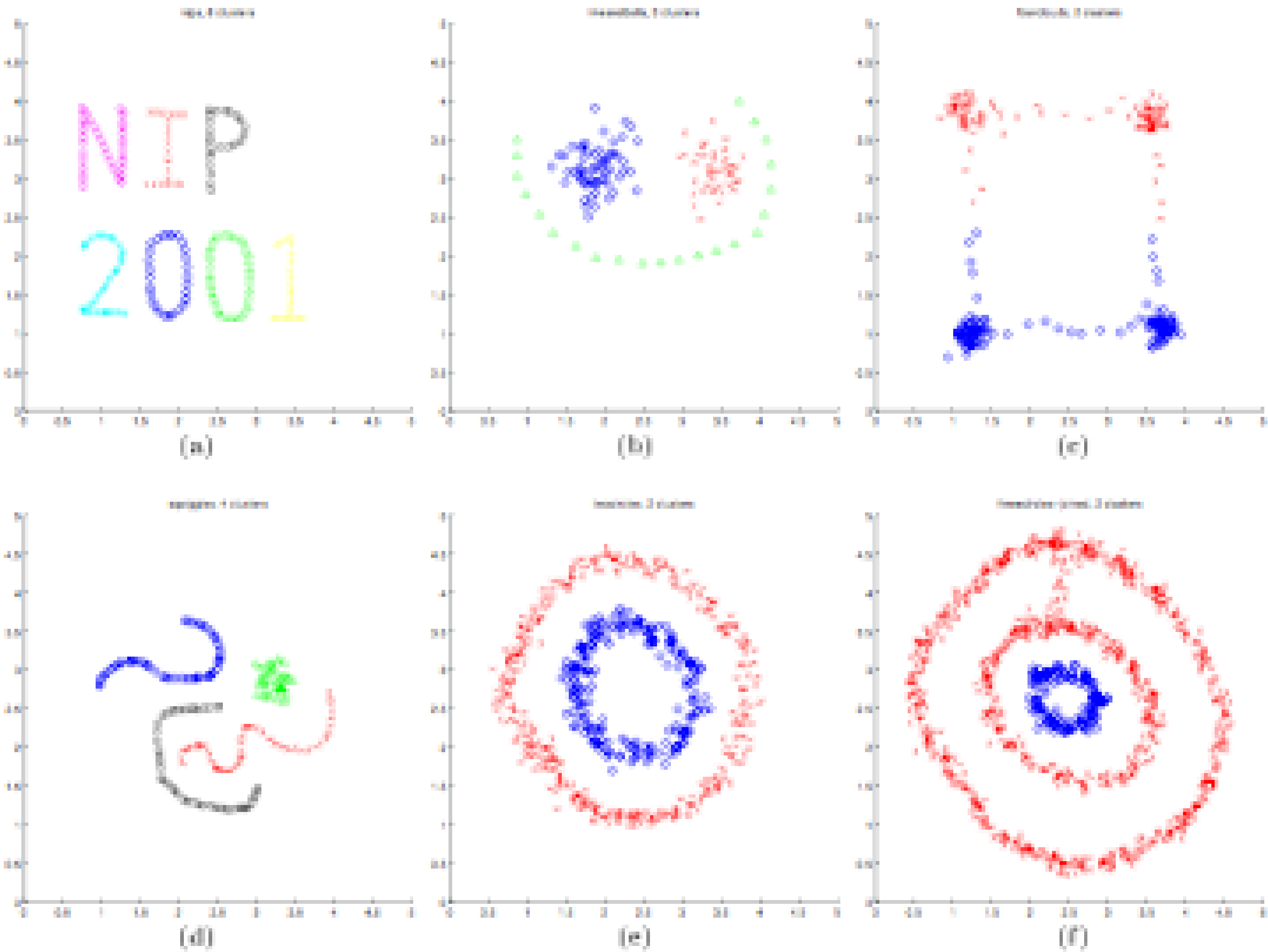
$U = [u_1, u_2, u_3]$



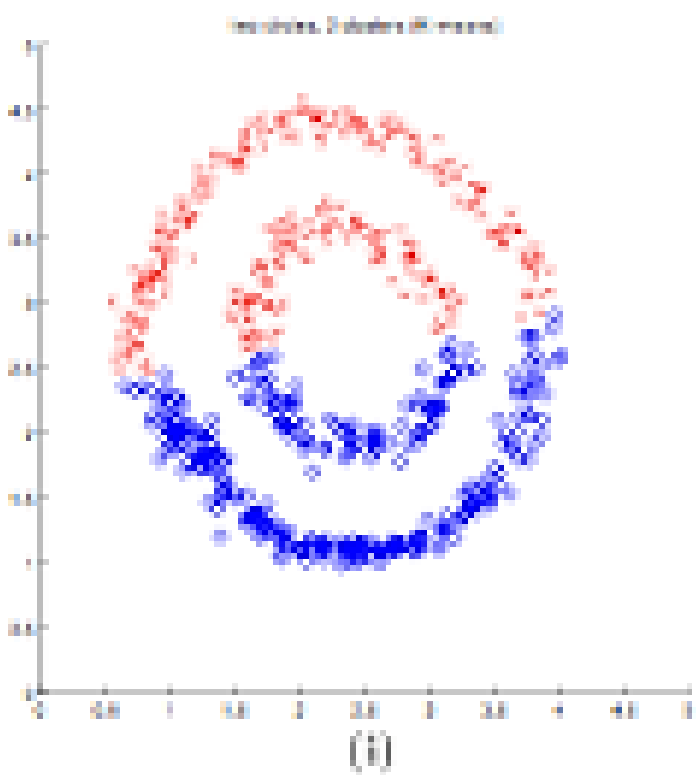
output



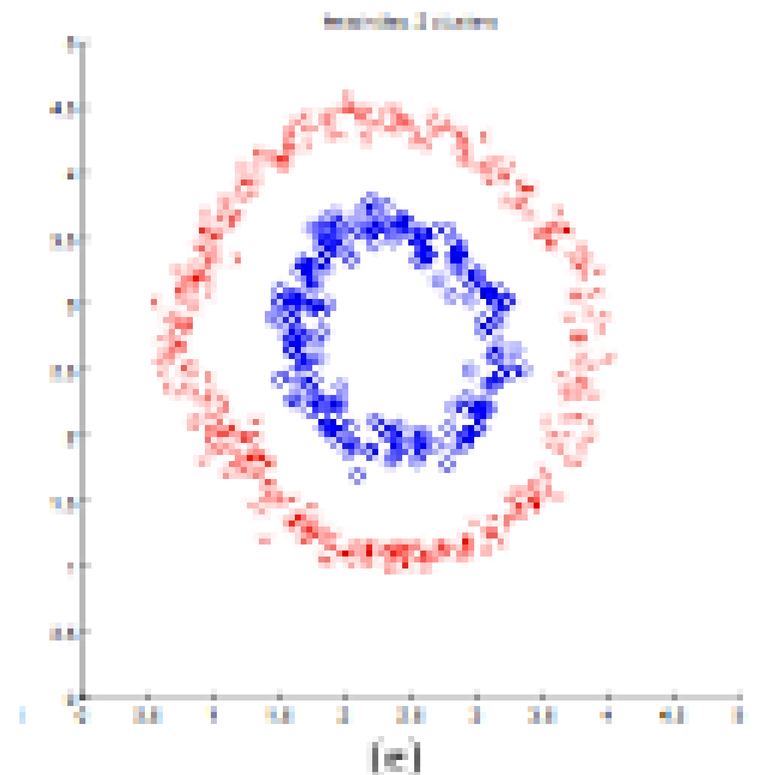
Example [Ng et al. 2001]



K-MEANS VS. SPECTRAL CLUSTERING



K-means



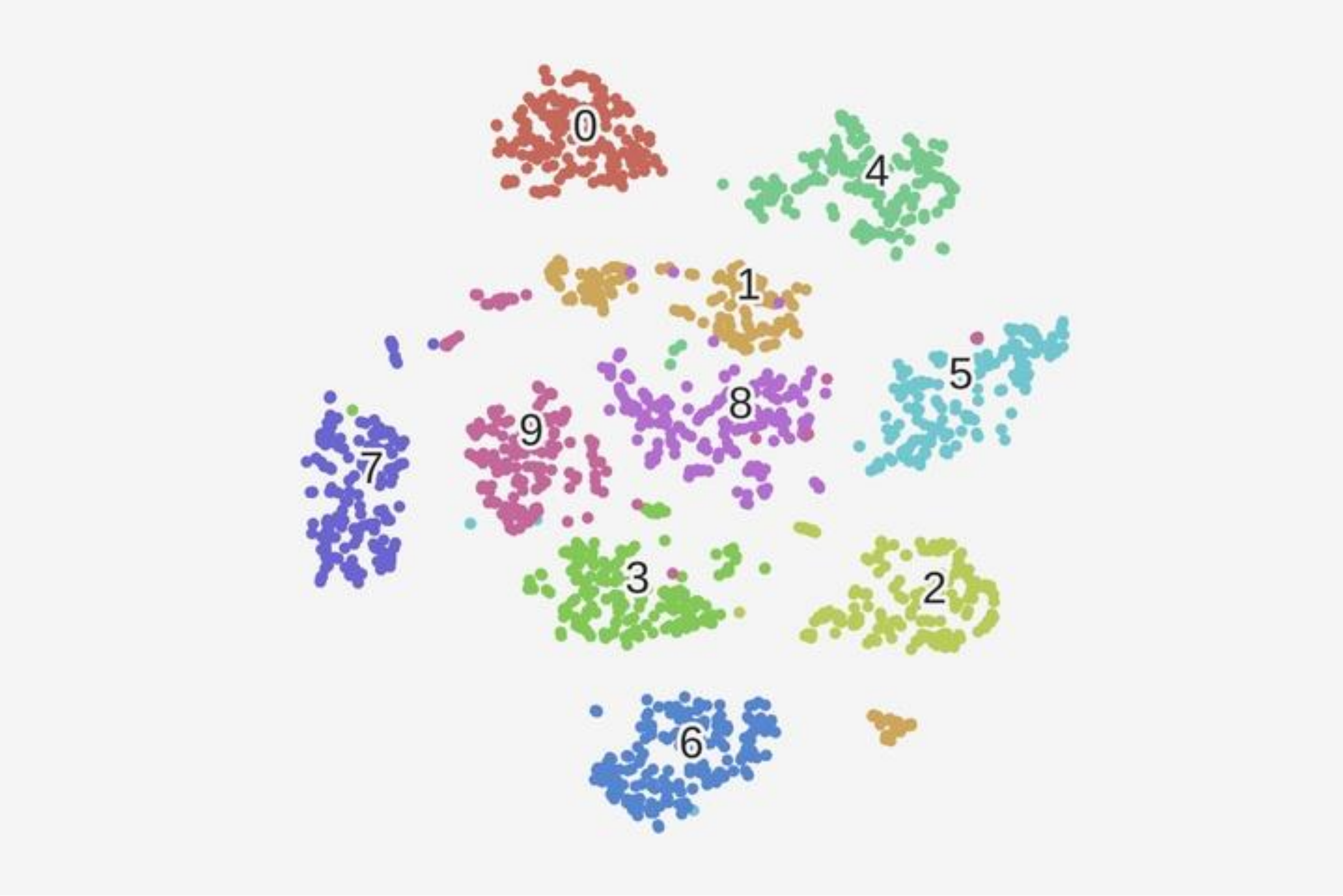
Spectral Clustering

DBSCAN

See slides by M.Ester, H.P.Kriegel, J.Sander and Xu

T-SNE

t-distributed stochastic neighbor embedding



T-SNE Distance Metric

Uses the following density-based (probabilistic) distance metric

$$p_{j|i} = \frac{\exp\left(-|x_i - x_j|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-|x_i - x_k|^2 / 2\sigma_i^2\right)}$$

Measures how close x_j is from x_i, considering a Gaussian distribution around x_i with a given variance σ^2_i.

- this variance is different for every point
- t is chosen such that points in dense areas are given a smaller variance than points in sparse areas

T-SNE IMPLEMENTATION

Use a symmetrized version of the conditional similarity:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

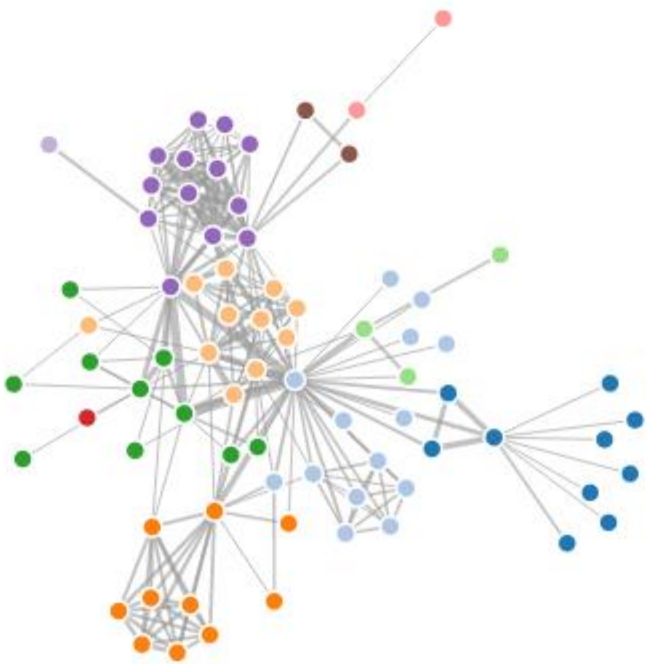
Similarity (distance) metric for map points:

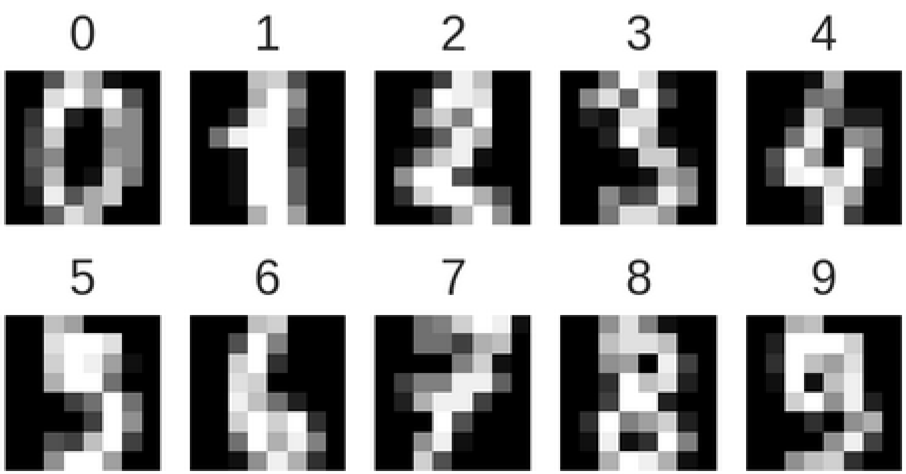
$$q_{ij} = \frac{f(|x_i - x_j|)}{\sum_{k \neq i} f(|x_i - x_k|)} \quad \text{with} \quad f(z) = \frac{1}{1+z^2}$$

This uses the t-student distribution with one degree of freedom, or Cauchy distribution, instead of a Gaussian distribution

Layout

Can use mass-spring system enforcing minimum of $|p_{ij} - q_{ij}|$





The classic *handwritten digits* datasets. It contains 1,797 images with 8∗8=64 pixels each.

Animated layout

9

More information

See [this webpage]