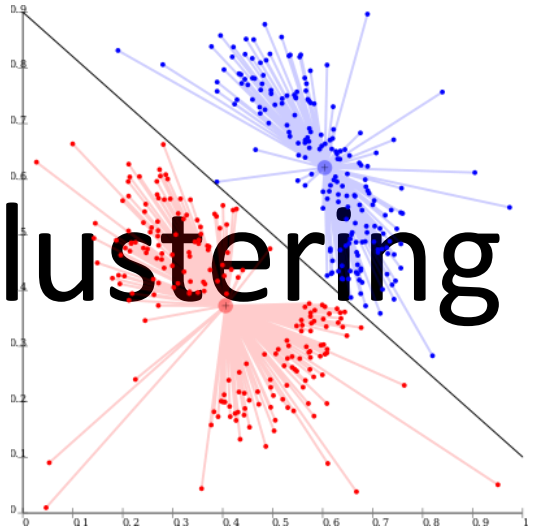
clustering



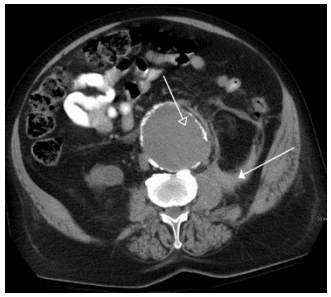
Algorithm Interest Group
presentation by Eli Chertkov

Image source: Wikipedia

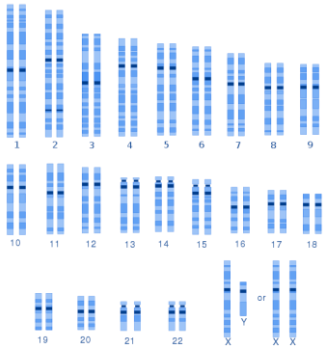
Clustering data

Ecology

Medical imaging



Genetics



Community detection in social networks

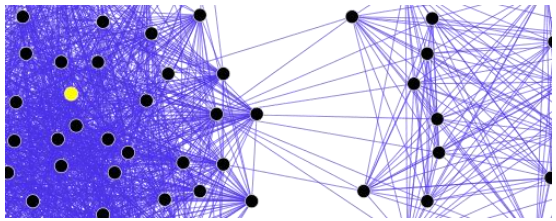
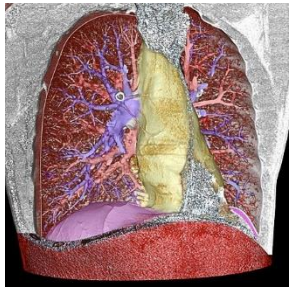


Image segmentation

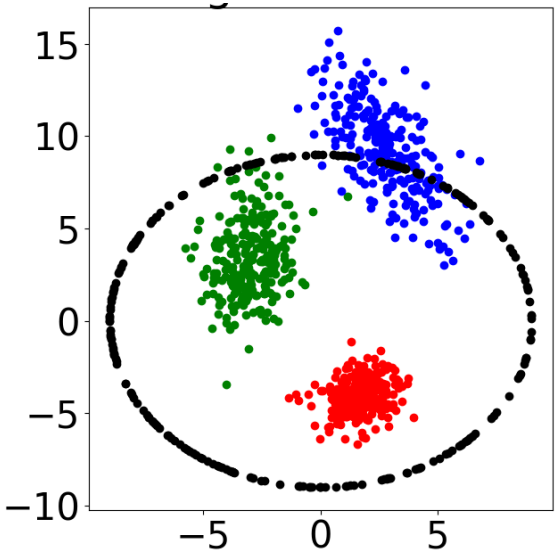


Source: Wikipedia

Clustering

Cluster analysis is a type of **unsupervised learning**, where given unlabeled data you attempt to interpret the correlations in the data and identity clusters of similar data points.

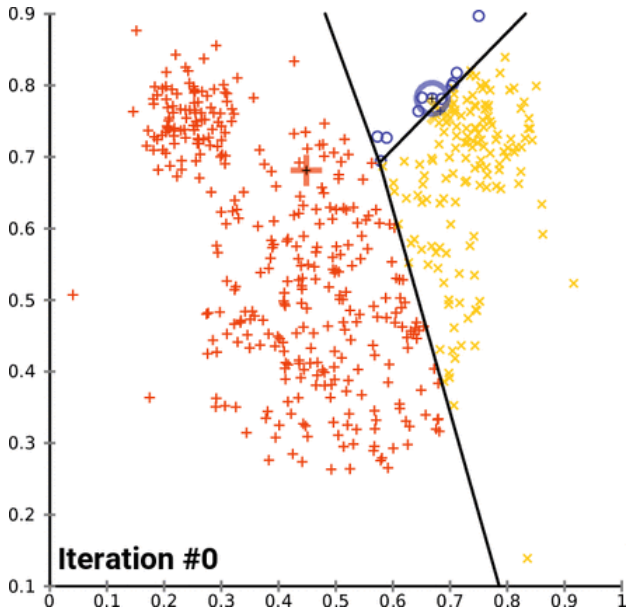
Toy example in 2D:
3 Gaussian clusters
1 ring cluster



k-means

One of the most common clustering algorithms. Simple, iterative, heuristic, greedy.

Idea: Group points into *k* clusters. Compute the centers of the clusters and assign points to the cluster with the closest center.



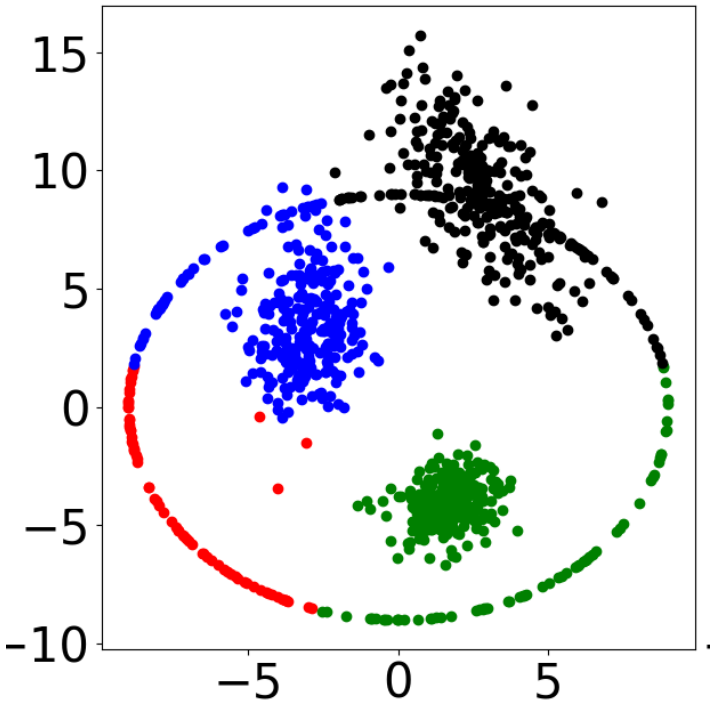
Algorithm:

1. Initialize cluster centers randomly.
2. Assign points to cluster with nearest center.
3. Recompute center of clusters.
4. Repeat 2 and 3 until converged.

Note: clustering is based on distances. Not always useful.

Source: Wikipedia

k-means



Distances do not capture all of the cluster information.

Spectral clustering

A clustering algorithm based on spectral embedding. Simple and based on linear algebra.

Idea: Use a ***kernel*** $K_{ij} = K(s_i, s_j)$ (similarity measure) between points $s_i \in R^d$ to embed the data into a new vector space. Perform *k*-means in the new space.

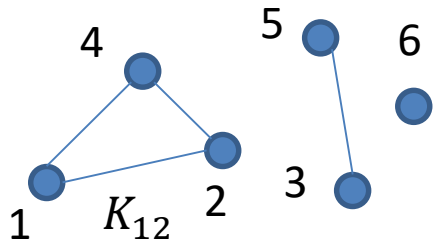
Algorithm:

Spectral embedding

1. Compute the ***graph Laplacian*** *L = D - K.*
2. Find the *k* lowest eigenvectors of *L.*
3. Embed the data into a *k*-dim space defined by the rows of these eigenvectors.
4. Perform *k*-means clustering on the embedded data.

Graph Laplacian

For a graph with edge weights K_{ij}, the degree matrix D is $D_{ij} = \delta_{ij} \sum_j K_{ij}$.



$$K = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The graph Laplacian is $L = D - K$

$$L = \begin{pmatrix} 2 & -1 & 0 & -1 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ -1 & -1 & 0 & 2 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Graph Laplacian properties

1. For any vector v,

$$v^T L v = \sum_{ij} K_{ij} \left(v_i - v_j\right)^2 / 2$$

(Modified inner product weighted by the kernel.)

2. L is symmetric and positive-semi definite, so has non-negative eigenvalues.

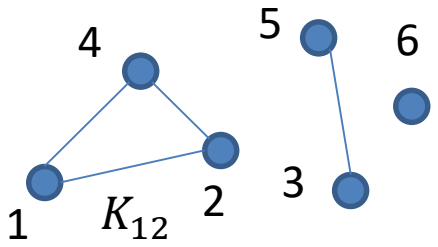
3. There is zero eigenvalue eigenvector of L that is the vector of all ones: $\mathbf{1} = (1,1,\ldots,1)$.

(If there are disjoint components $A_1, \ldots, A_n$ in the graph, then $\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_n}$ are all zero eigenvalue eigenvectors.)

Graph Laplacian (revisited)

The graph Laplacian is $L = D - K$

$$L = \begin{pmatrix} 2 & -1 & 0 & -1 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ -1 & -1 & 0 & 2 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



It has three zero eigenvalue eigenvectors:

$$v_1 \propto \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}^T$$

$$v_2 \propto \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}^T$$

$$v_3 \propto \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^T$$

Intuitively, these correspond to clusters connected by the *K* matrix. Roughly the same picture holds when we perturb *K* so that its entries are not all 1's and 0's.

Kernels

Rather than using adjacency matrices with 0s and 1s, we define our graph with edge weights given by the kernel K_{ij}.

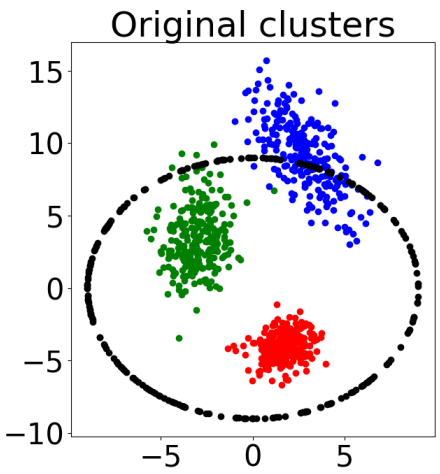
The kernel used in spectral clustering is picked empirically.

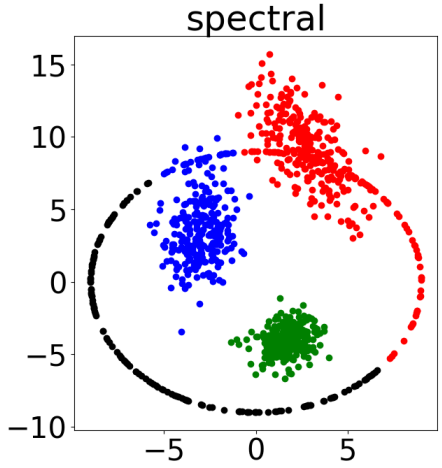
The most common kernel is the Gaussian (radial, or heat) kernel

$$K_{ij} = K(s_i, s_j) = e^{-|s_i - s_j|^2/(2\sigma^2)}$$

but it has an arbitrary parameter σ that needs to be tuned by hand.

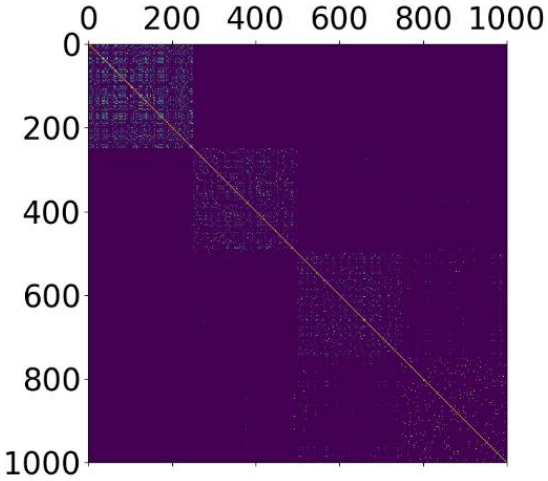
Toy example

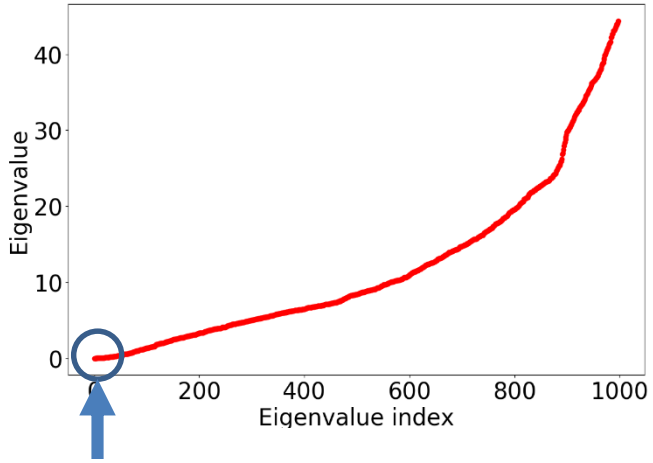




$\sigma = 0.4$

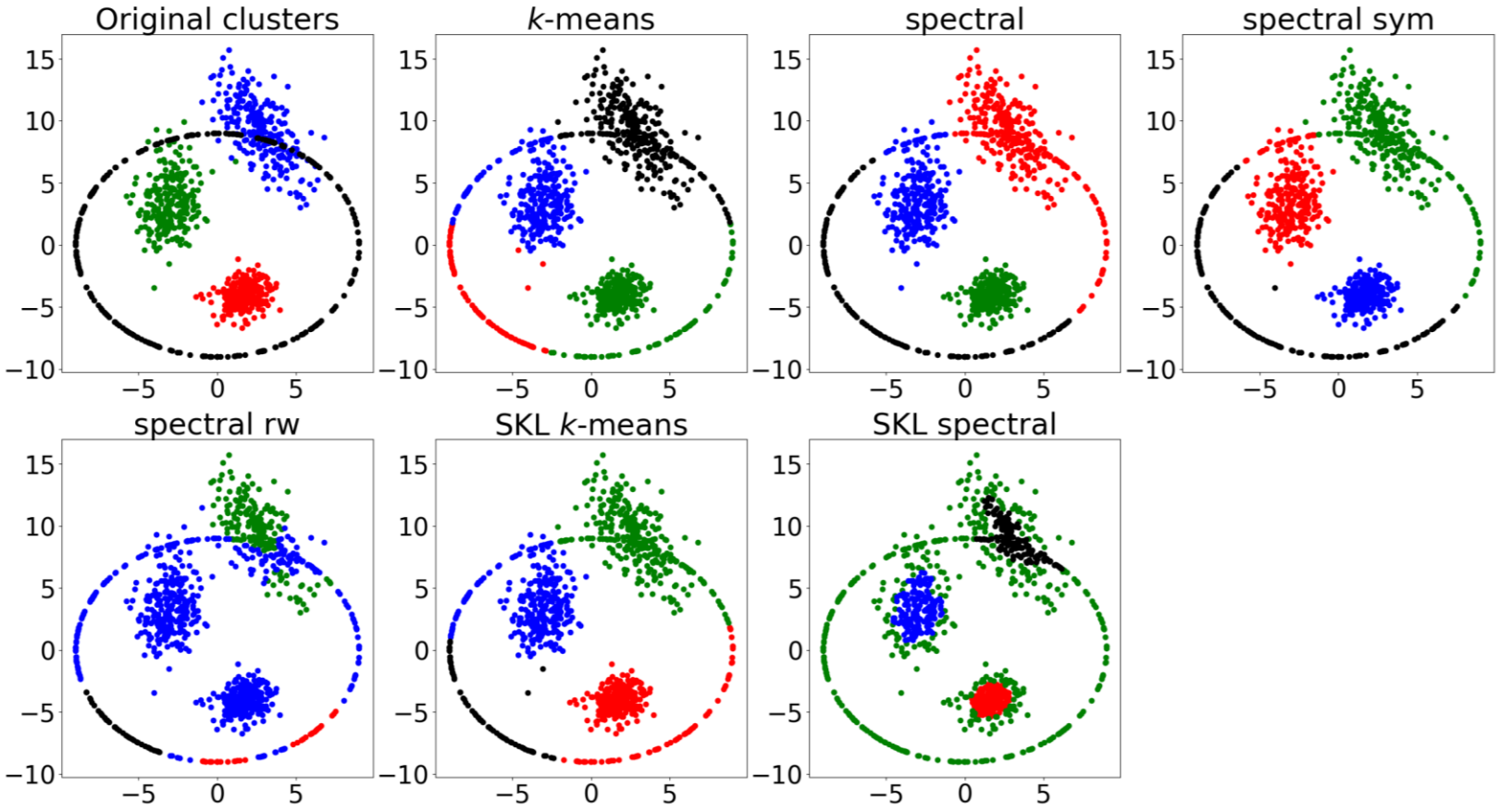
$K(s_i, s_j)$





Look at *k* smallest eigenvalues of *L*

Toy example (results)

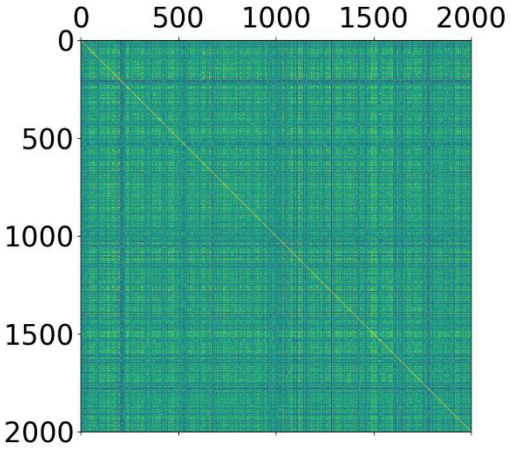


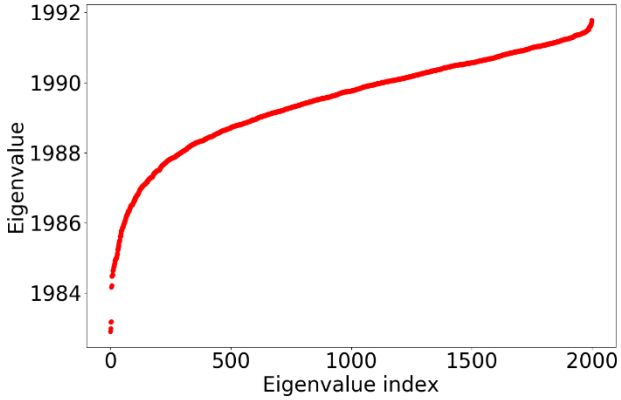
MNIST example

MNIST: Dataset of images of handwritten digits used as a benchmark in many ML problems.

$\sigma = 50$

$K(s_i, s_j)$





MNIST example (results)

Different clusters

4,3,5,8's cluster

6's cluster well

Digits assigned to each cluster

References

U. Luxburg. *A Tutorial on Spectral Clustering*. TR-149. 2007.

Scikit-Learn Documentation and Tutorials.

Jun Song's Phys 598 SDA course notes.

Spectral embedding

Spectral embedding is a way to embed data $s_i \in R^d$ into a space spanned by *k* vectors $\tau_1(s_i), \ldots, \tau_k(s_i) \in R^d$. These vectors are chosen to minimize the distance between highly similar data points according to the kernel $K(s_i, s_j)$:

$$E(\tau) = \sum_{i,j=1}^{n} K(s_i, s_j)||\tau(s_i) - \tau(s_j)||^2$$

$$= 2\sum_{\alpha=1}^{k} \tau_\alpha^T L \tau_\alpha$$

To minimize this objective function, we choose $\tau_\alpha(s_i) = v_{\alpha i}$ where v_α are the *k* lowest eigenvectors of *L*.