CSE 564
Visualization & Visual Analytics

Cluster Analysis & Dimension reduction

Klaus Mueller

Computer Science Department
Stony Brook University and SUNY Korea

Lecture	Topic	Projects
1	Intro and logistics	
2	Basic visualizations and tasks, data types, examples, ethical considerations	
3	Data preparation (cleaning, imputation, data set integration)	
4	AI-assisted coding for VIS applications (design, debugging, refactoring)	Project #1 out
5	Big data and data reduction (distance/sim metrics, intro to clustering)	
6	High-D data: concept, subspaces, dimension reduction, PCA	
7	Cluster analysis: hierarchical, density, model, embedding, temporal	
8	Perception and cognition (human visual system, color, contrast, bias)	Project #2(a) out
9	Visual design and aesthetics	
10	Visualization of multivariate and high-dimensional data: direct methods	
11	Visualization of multivariate and high-D data: projections & embeddings	
12	Visualization and AI: mutual support and capabilities (VIS4AI, AI4VIS)	Project #2(b) out
13	Principles of interaction: drive what is visualized, analyzed & how (HCI4VIS)	
14	Visual analytics (VA), human-centered AI, mixed-initiative system	
15	Midterm #1 (tentative date)	
16	VA system design and evaluation, collaborative VA, uncertainty, provenance	
17	Midterm #1 discussion (tentative date)	Final proj. proposal call out
18	Visualization of hierarchical data	
19	Visualization of maps and data with geo-reference	
20	Visualization of graphs, networks (incl. derivation of causal networks)	Final project proposal due
21	Vis. of time-varying, time-series, streaming data, progressive visualization	
22	Visualization of text, LLMs, and semantic data	
23	Ed Tufte revisited: principles, critiques and limits, responsible visualization	
24	Design of effective infographics	Final proj. prelim report due
25	Foundations scientific and medical visualization, intro to volume rendering	
26	Scientific visualization	Bonus project out (Vol Ren)
27	Story telling with data, data journalism	
28	Midterm #2 (tentative date)	
Final	Final project demo on zoom (public)	All final proj. materials due

Clustering = modeling similarity structure

Clustering Families

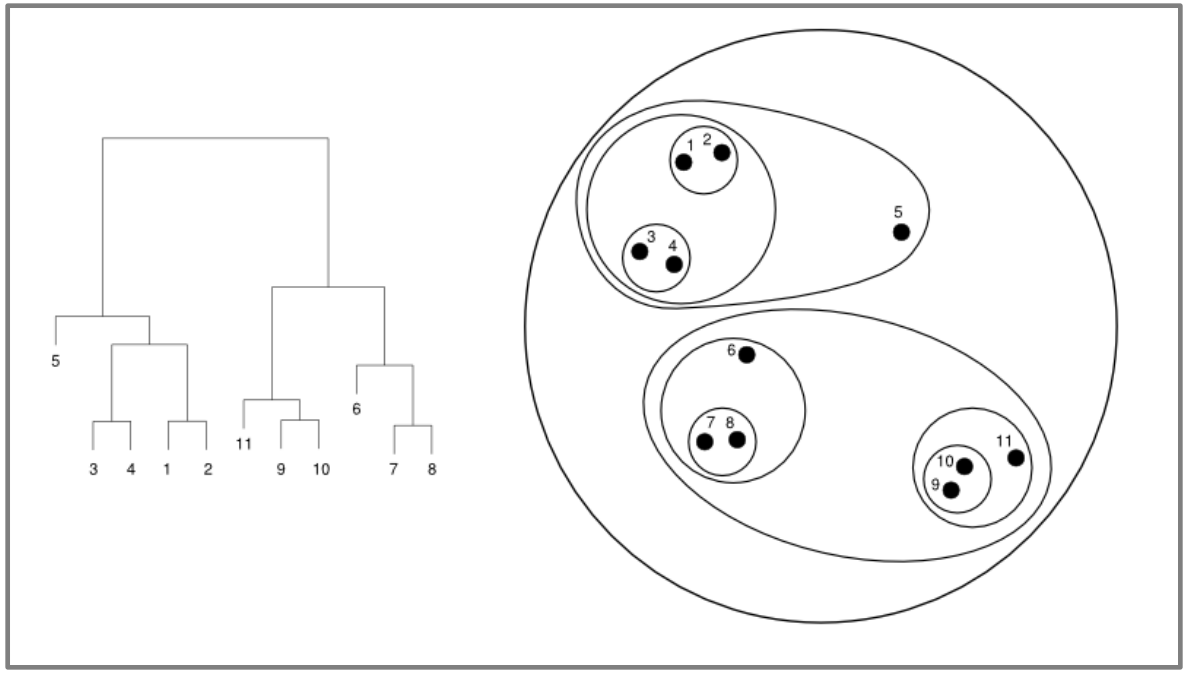
- Partition-based → K-means (see previous lecture)
- Hierarchical → Agglomerative
- Density-based → DBSCAN
- Model-based → Gaussian Mixture (EM)
- Embedding-based → Deep clustering
- Time-series → Tensors

Often a precursor to other analyses

- preprocessing step for classification and outlier detection
- sampling and data reduction

Hierarchical (Agglomerative) Clustering

Hierarchical Clustering

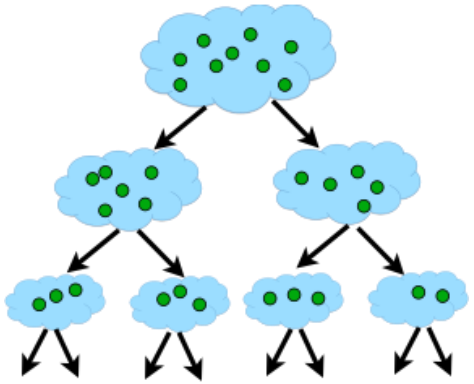


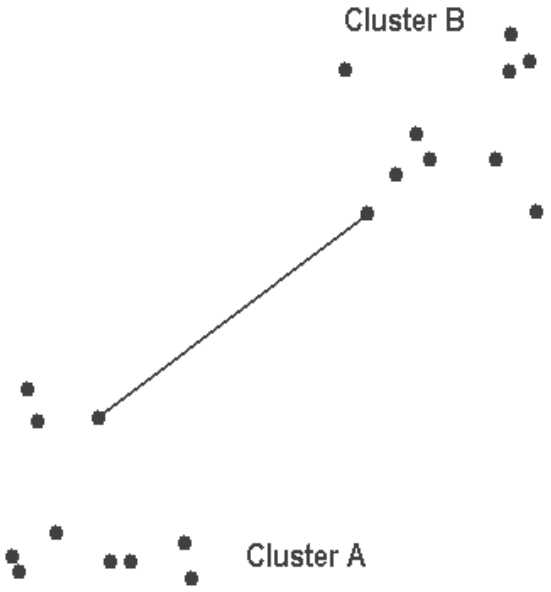
Two options for building the dendrogram on the left

- top down (divisive)
- bottom up (agglomerative)

Bottom-Up Agglomerative Methods

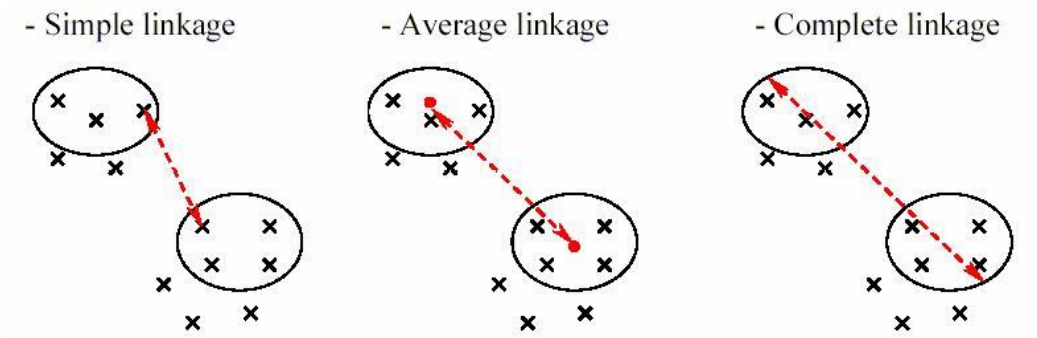
Algorithm *AgglomerativeMerge*(Data: $\mathcal{D}$)
begin
  Initialize $n \times n$ distance matrix M using $\mathcal{D}$;
  repeat
    Pick closest pair of clusters i and j using M;
    Merge clusters i and j;
    Delete rows/columns i and j from M and create
      a new row and column for newly merged cluster;
    Update the entries of new row and column of M;
  until termination criterion;
  return current merged cluster set;
end





How to merge?

Merge Criteria



Single (best-case) linkage

- distance = minimum distance between all $m_i \cdot m_j$ pairs of objects
- joins the closest pair

Complete (worst-case) linkage

- distance = maximum distance between all $m_i \cdot m_j$ pairs of objects
- joins the pair furthest apart

Group-average linkage

- distance = average distance between all object pairs in the groups

Other methods:

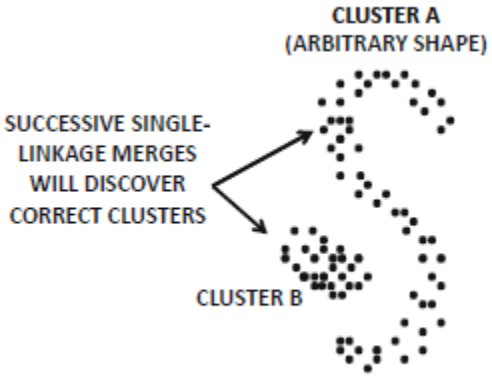
- closest centroid, variance-minimization, Ward's method

Comparison

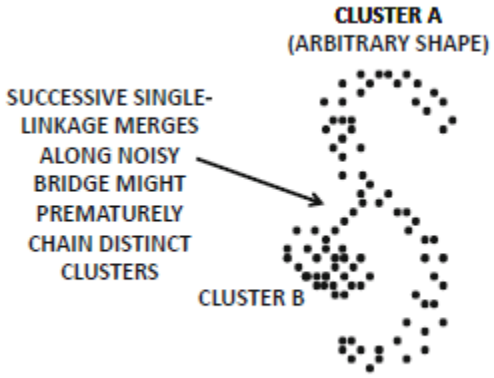
Centroid-based methods tend to merge large clusters

Single linkage method can merge chains of closely related points to discover clusters of arbitrary shape

- but can also (inappropriately) merge two unrelated clusters, when the chaining is caused by noisy points between two clusters



(a) Good case with no noise

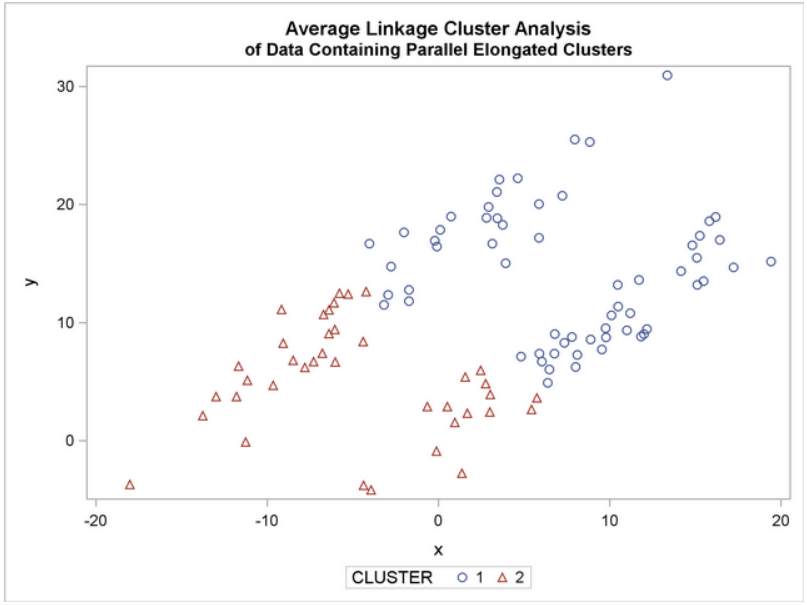


(b) Bad case with noise

Comparison

Complete (worst-case) linkage method tends to create spherical clusters with similar diameter

- will break up the larger odd-shaped clusters into smaller spheres
- also gives too much importance to data points at the noisy fringes of a cluster

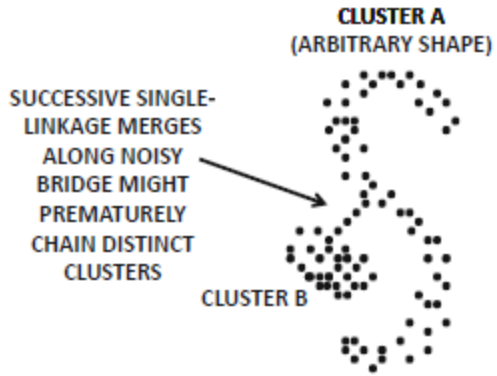


Comparison

The group average, variance, and Ward's methods are more robust to noise due to the use of multiple linkages in the distance computation

Hierarchical methods are sensitive to a small number of mistakes made during the merging process

- can be due to noise
- no way to undo these mistakes



(b) Bad case with noise

Density-Based Clustering

DBSCAN

Highly-cited density-based hierarchical clustering algorithm (Ester et al. 1996)

- clusters are defined as density-connected sets
- epsilon-distance neighbor criterion (Eps)

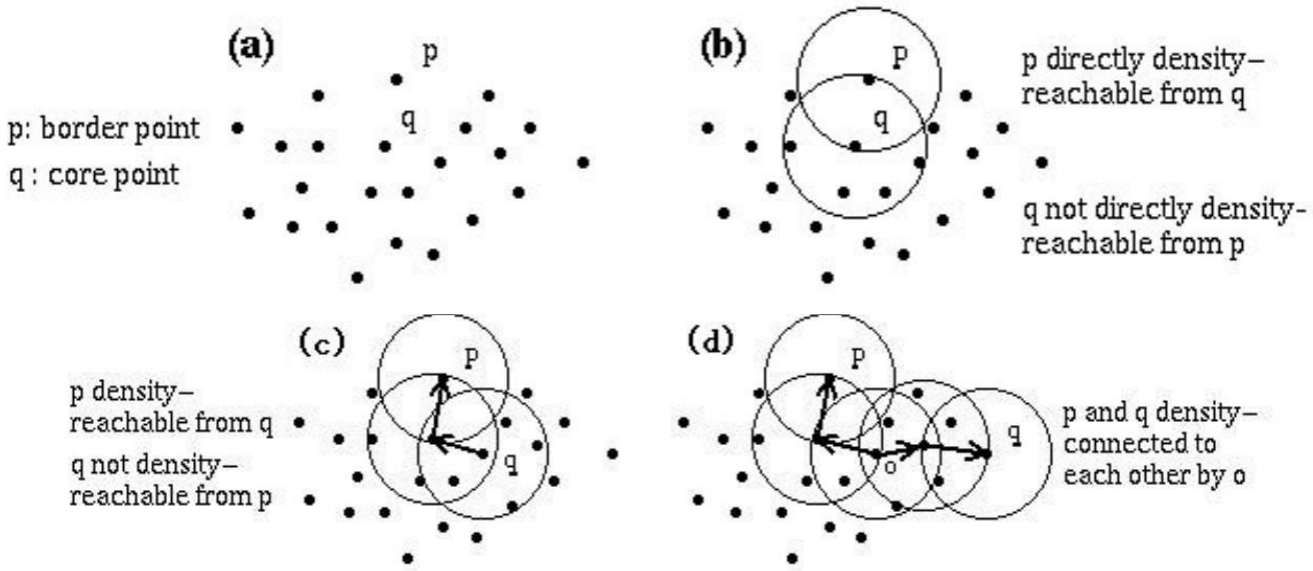
  $N_{Eps}(p) = \{q \in D \mid dist(p,q) \leq Eps\}$
- minimum point cluster membership and core point (MinPts)

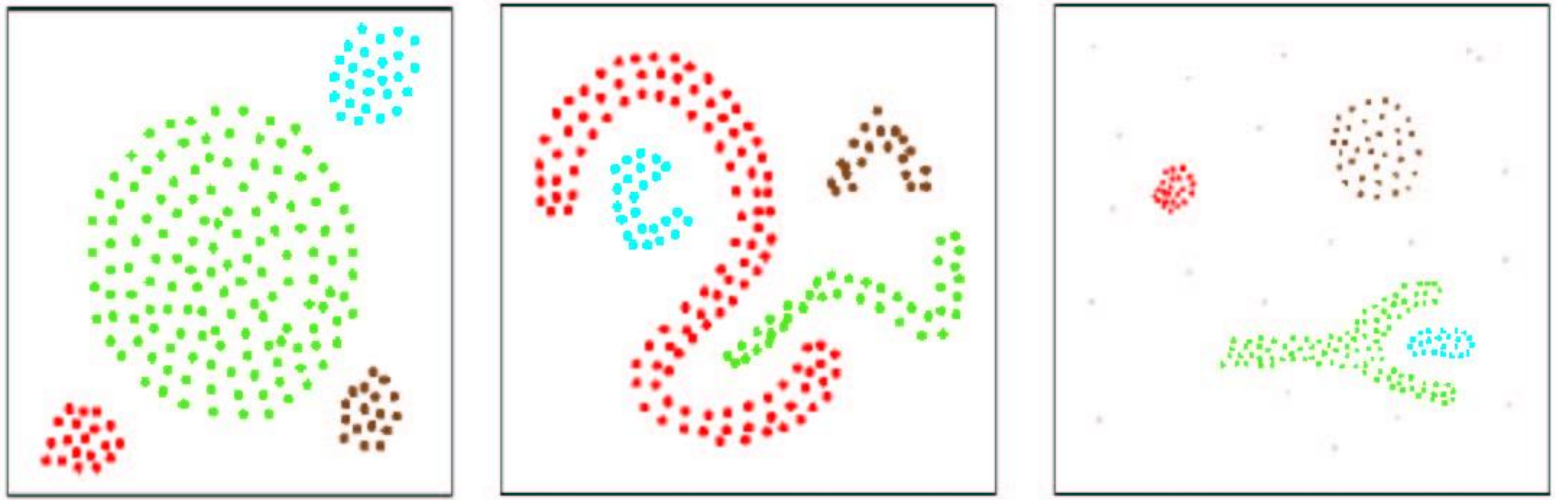
  $|N_{Eps}(q)| \geq MinPts$
- notions of density-connected & density-reachable (direct, indirect)
- a point p is directly density-reachable from a point q wrt. Eps, MinPts if

  $p \in N_{Eps}(q)$ and

  $|N_{Eps}(q)| \geq MinPts$ (core point condition)

DBSCAN

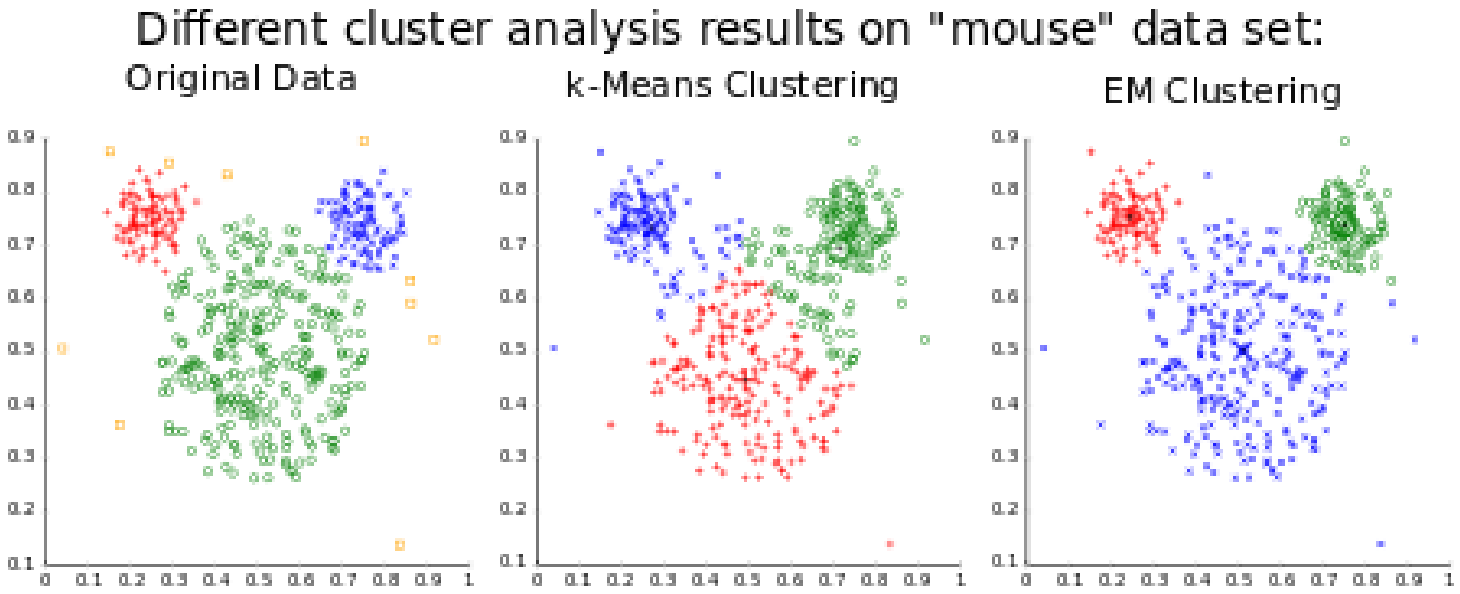

(a)
p
q
p: border point
q : core point
(b)
p
q
p directly density–reachable from q
q not directly density-reachable from p
(c)
p
q
p density–reachable from q
q not density–reachable from p
(d)
p
q
o
p and q density–connected to each other by o



Model-Based Clustering

Probabilistic Extension to K-Means

First a comparison:



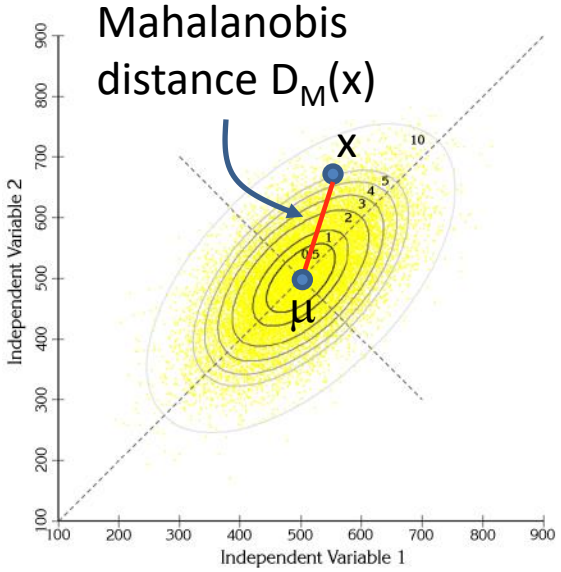
Mahalanobis Distance

The distance between a point X and a distribution D

- measures how many standard deviations X is away from the mean μ of D
- S is the covariance matrix of the distribution D
- the Mahanalobis distance D_M of a point x to a cluster center μ is

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1}(x-\mu)}.$$

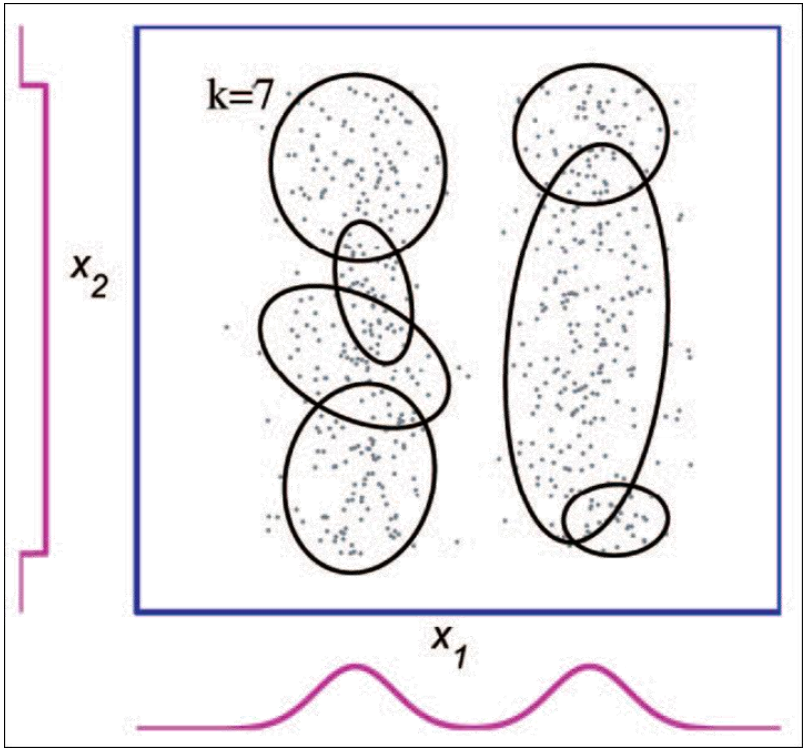
- x and μ are N-dimensional vectors
- S is the N×N covariance matrix
- the outcome $D_M(x)$ is a single-dimensional number



Probabilistic Clustering

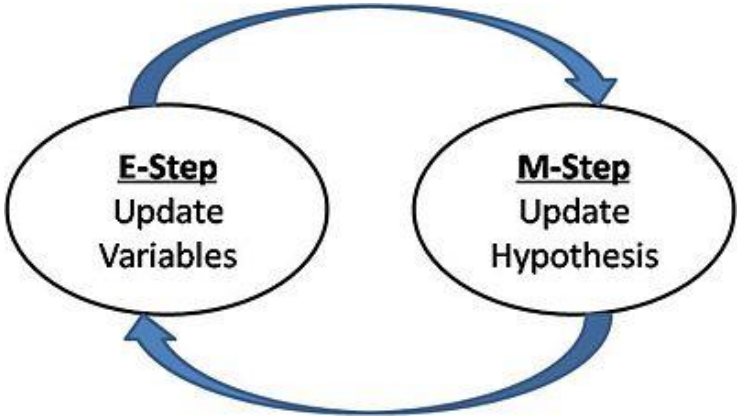
Is a better match for point distributions

- overlapping clusters are now possible
- better match with real world?
- Gaussian mixtures



Need a probabilistic algorithm

- Expectation-Maximization



EM Algorithm (Mixture Model)

probability that data point d_i is in class c_j
(= Mahanalobis distance of d_i to c_j)

- Initialize K cluster centers
- Iterate between two steps
  - **E**xpectation step: assign points to *m* clusters/classes

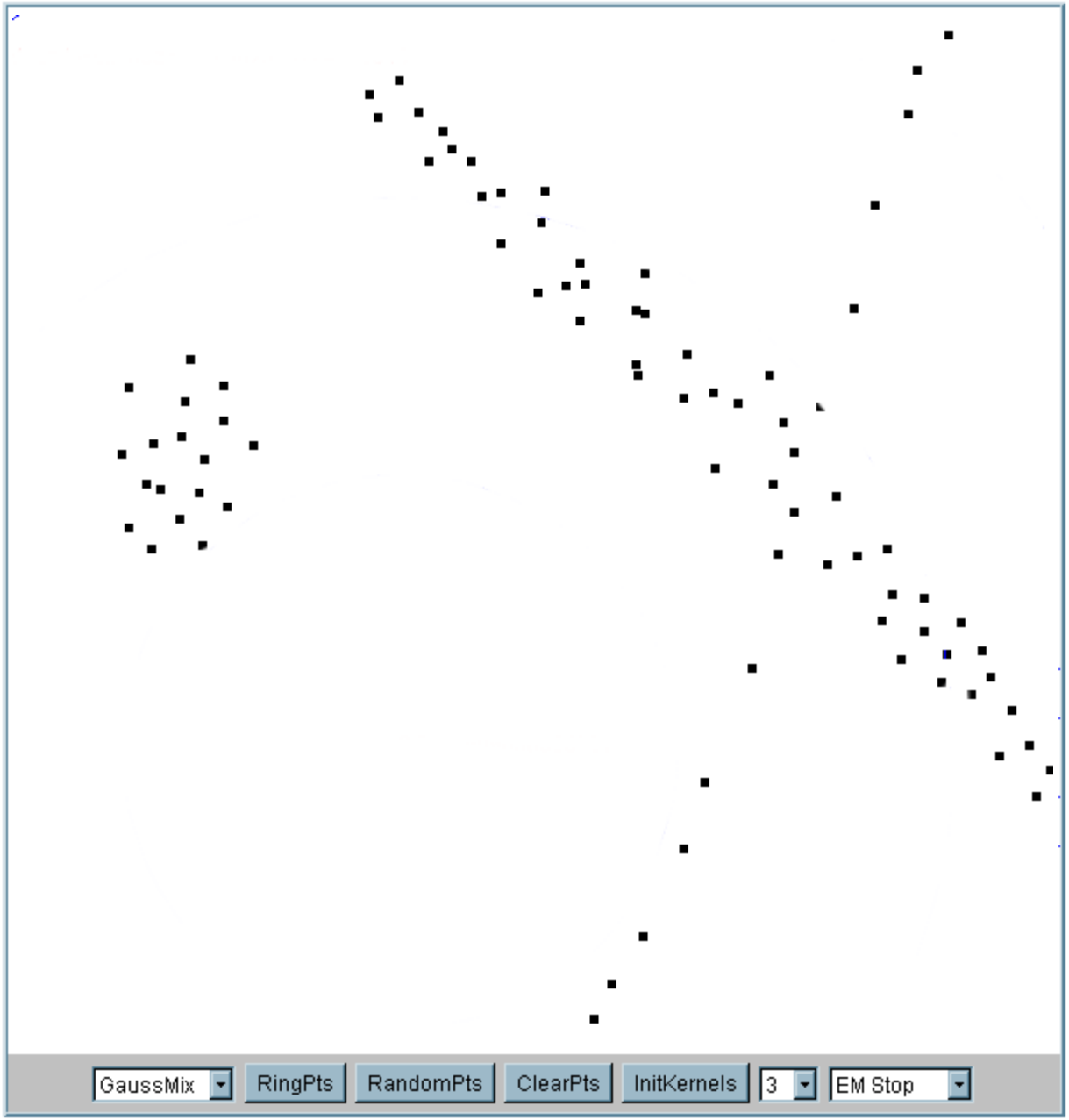
$$P(d_i \in c_k) = w_k \Pr(d_i \mid c_k) \Big/ \sum_j w_j \Pr(d_i \mid c_j)$$

$$w_k = \frac{\sum_i \Pr(d_i \in c_k)}{N} = \text{probability of class } c_k$$

  - **M**aximation step: estimate model parameters

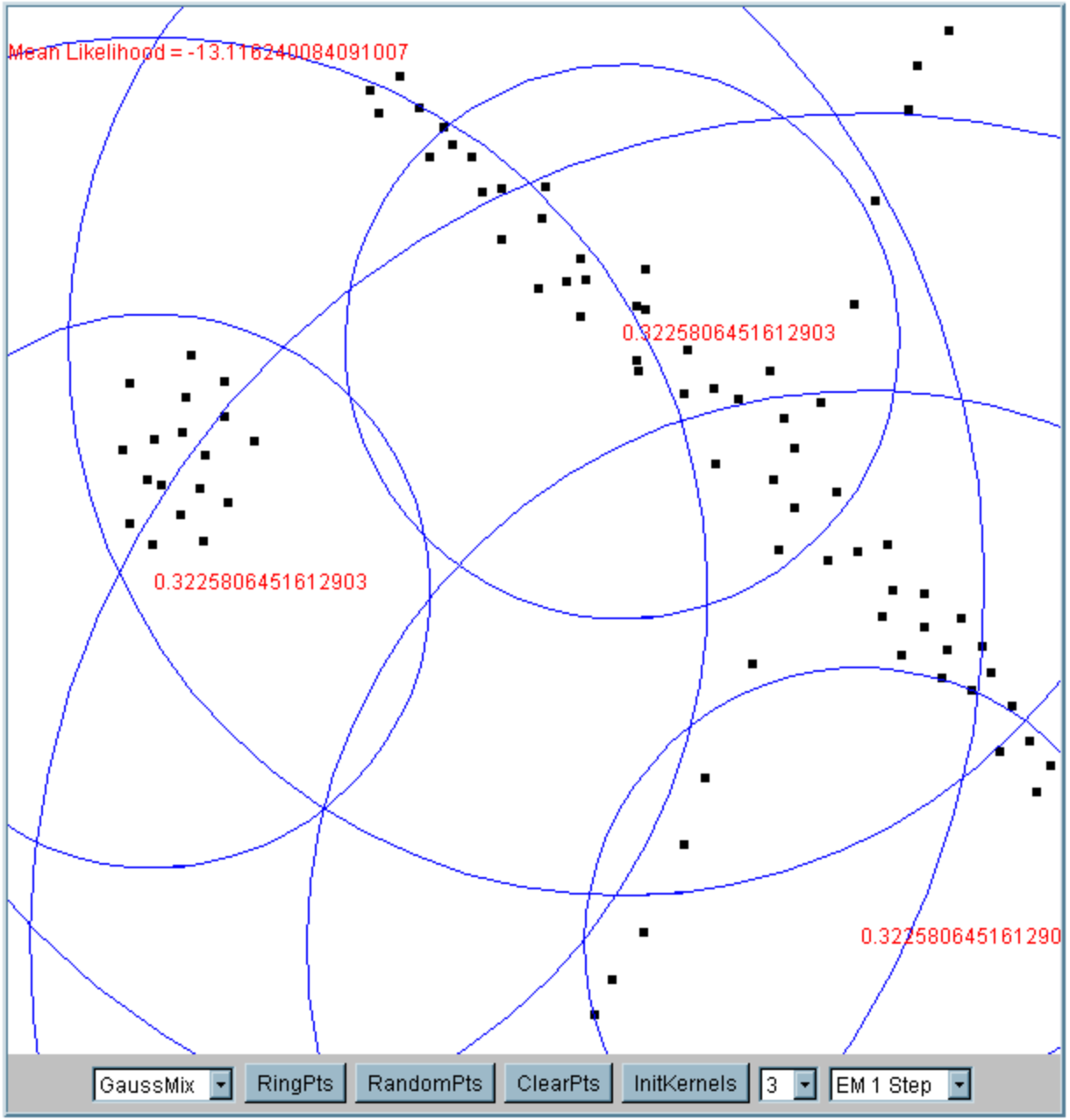
do similar also for covariance matrix S

$$\mu_k = \frac{1}{m} \sum_{i=1}^{m} \frac{d_i P(d_i \in c_k)}{\sum_k P(d_i \in c_j)}$$

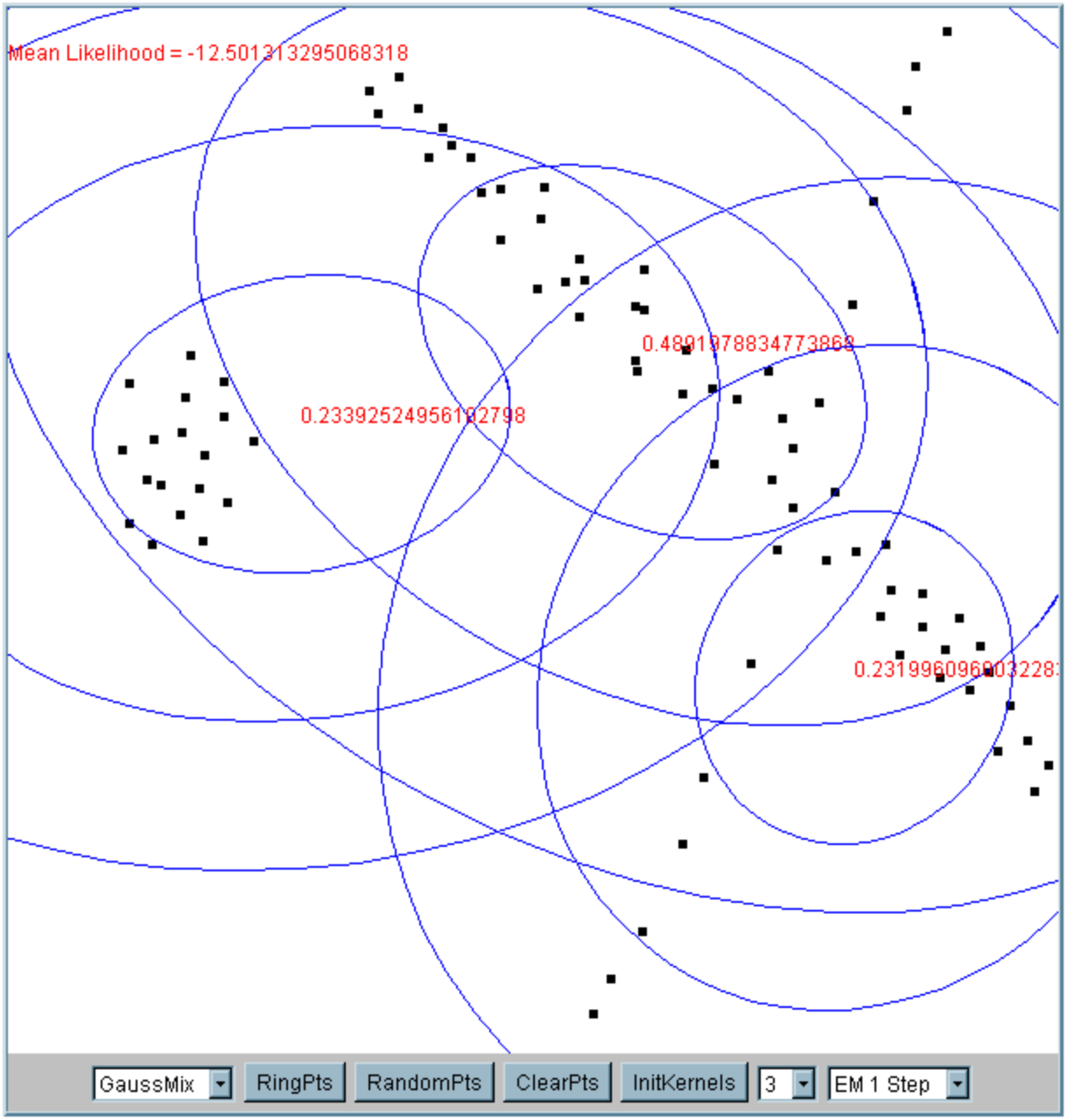

GaussMix
RingPts
RandomPts
ClearPts
InitKernels
3
EM Stop

Iteration 1

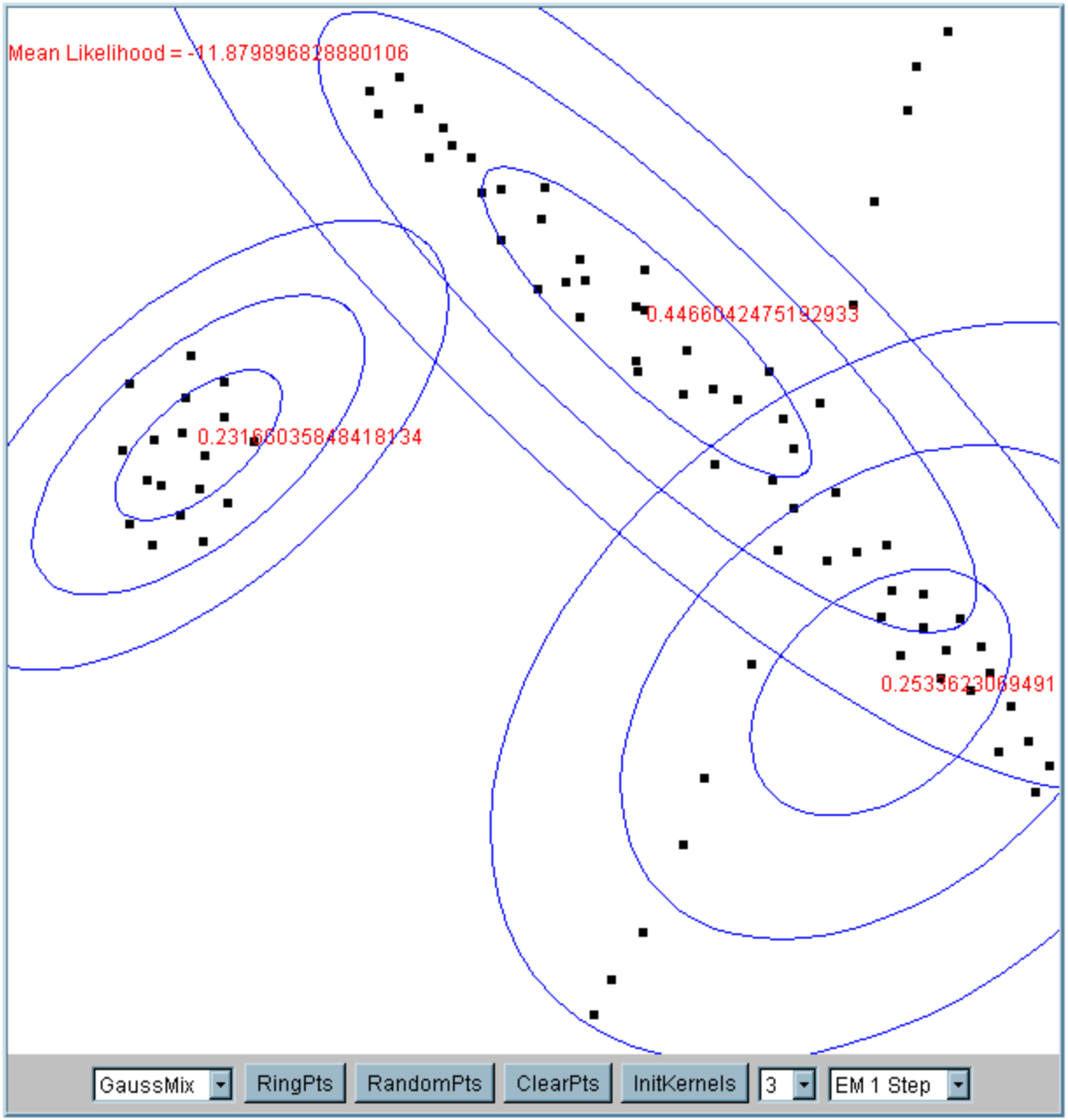
The cluster means are randomly assigned



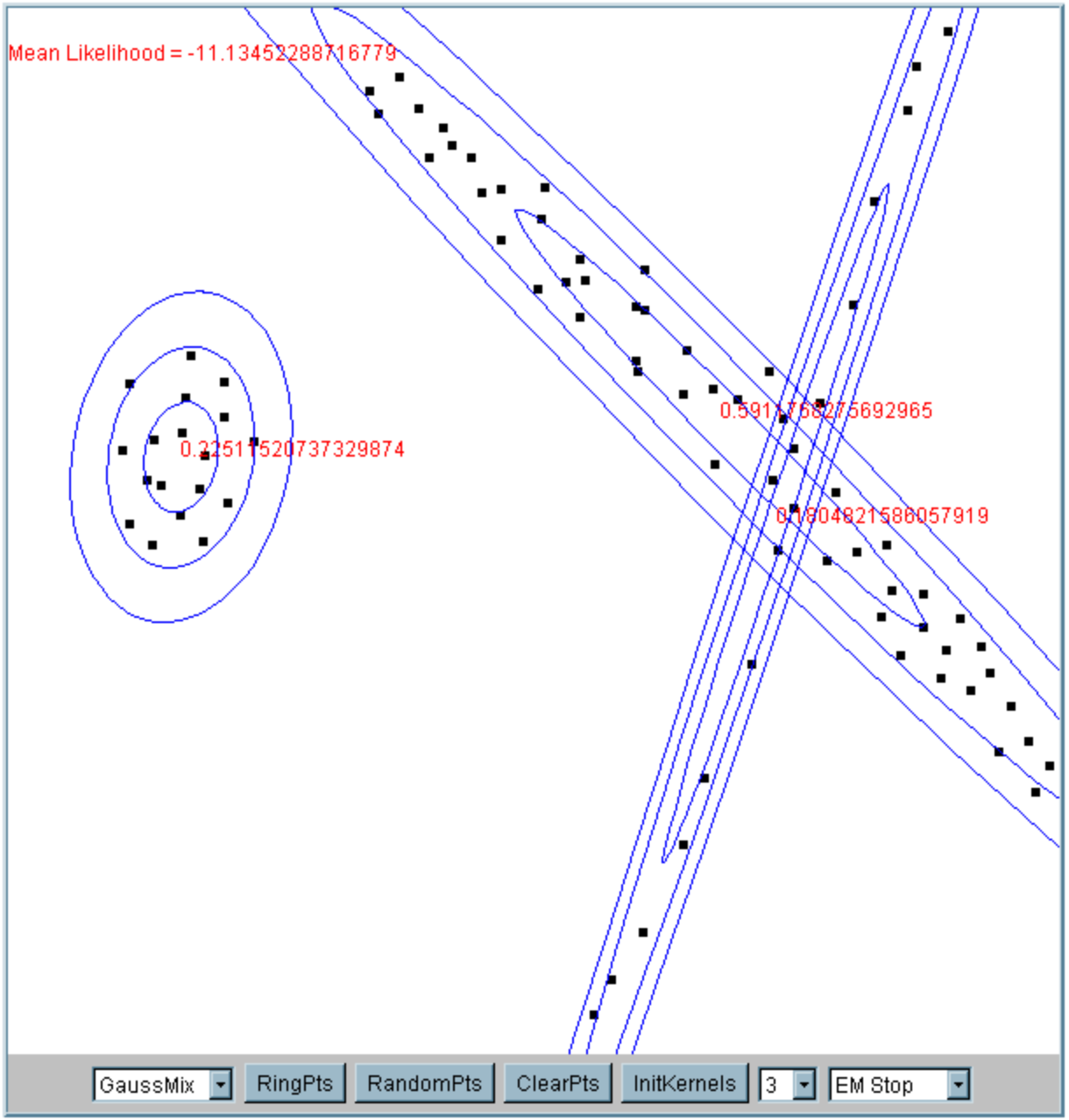
Iteration 2


Mean Likelihood = -12.501313295068318
0.48918788834773868
0.233925249561102798
0.2319960960032281
GaussMix
RingPts
RandomPts
ClearPts
InitKernels
3
EM 1 Step

Iteration 5


Mean Likelihood = -11.879896828880106
0.4466042475192933
0.2316803584841813 4
0.2535623069491
GaussMix
RingPts
RandomPts
ClearPts
InitKernels
3
EM 1 Step

Iteration 25

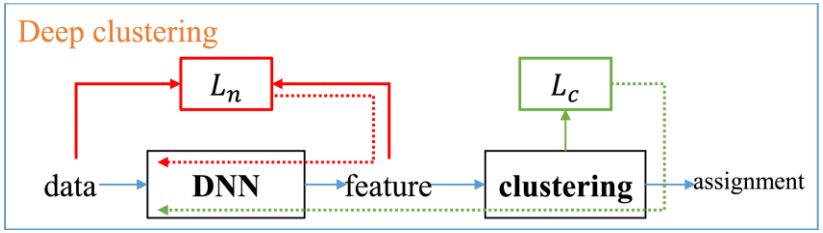


Deep Clustering

Strategy

Embed data into a latent space

- Reduce dimensionality via variational auto-encoder
- Perform clustering in latent space



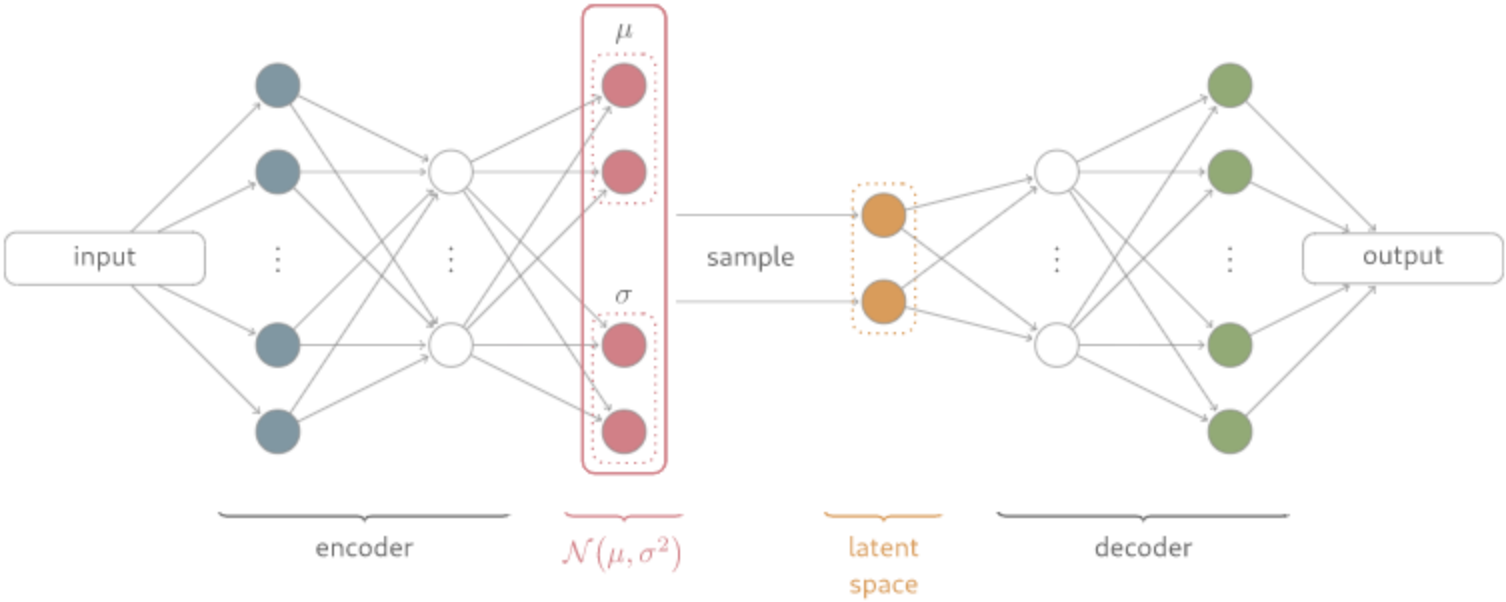
Key advantages

- Traditional clustering (e.g., k-means, DBSCAN) assumes data lies in a flat Euclidean space
- Deep clustering
  - Learns a non-linear latent manifold
  - "Unfolds" complex structures in high-D data
  - Makes clusters more separable than in raw feature space

Reduction via Neural Network

Train a Variational Autoencoder (VAE)

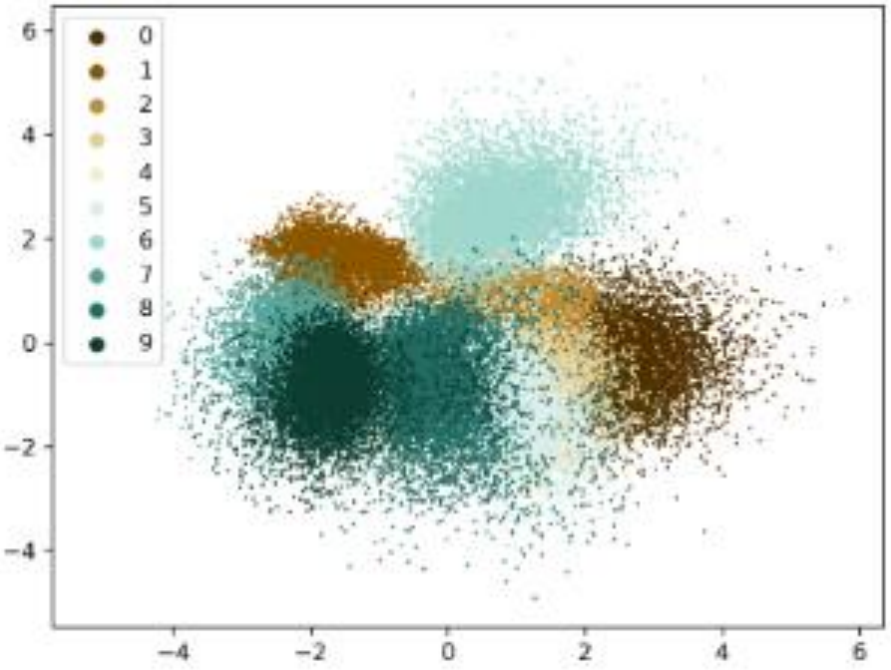
- optimize the output reconstruction loss of the input
- also optimize the latent distribution to be standard normal



Reduction via NN: Results

Dataset: 60,000 images of handwritten digits (MINST)

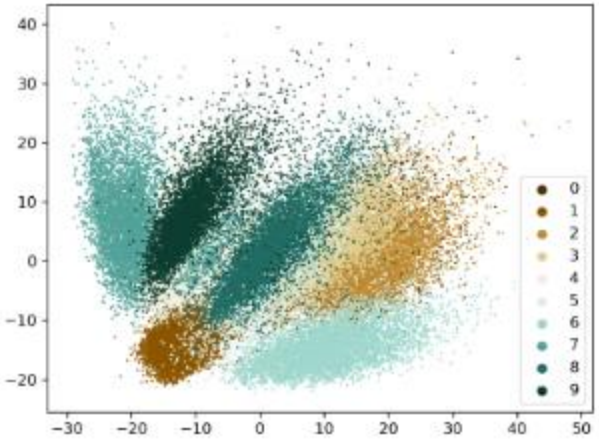
- each image is 28 × 28 → 784 D space



PCA projection of its 4D latent space

Reduction via NN: Results

Result when not assuring a standard normal distribution in the latent space

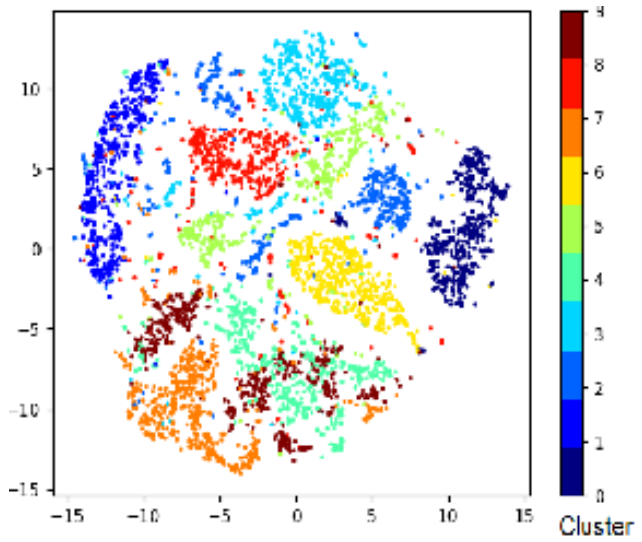


$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$

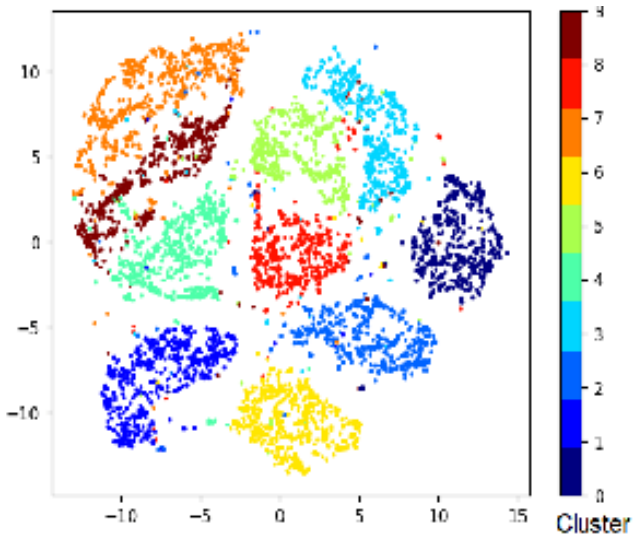
Reconstruction loss

Kullback-Leibler divergence

Example



(a) k-Means



(b) Autoencoder + k-Means

t-SNE visualizations for clustering on MNIST dataset

By: Aljalbout et al. ‘Clustering with Deep Learning: Taxonomy and New Methods’

Allows Dimensionality Reduction with Semantics

High-D feature spaces suffer from:

- Distance concentration
- Noise dominance
- Irrelevant dimensions

VAE latent spaces:

- Compress data into a small, information-dense representation
- Preserve semantic similarity, not just numeric similarity
- Suppress noise and redundant dimensions
- This improves cluster stability and interpretability

Smooth, Continuous Latent Space

Because VAEs enforce a structured latent distribution (typically Gaussian):

- Nearby latent points correspond to similar data instances
- Clusters become compact and well-behaved
- Boundary artifacts caused by raw feature sparsity are reduced

This is especially useful for:

- Image data
- Text embeddings
- Sensor data

Better Cluster Shape Assumptions

Classic clustering often assumes:

- Spherical clusters (k-means)
- Uniform density (DBSCAN)

Deep clustering:

- Allows arbitrary cluster shapes in original space
- Converts them into more regular shapes in latent space
- Enables simple algorithms (e.g., k-means) to work better

Key idea:

- Complexity is absorbed by the encoder, not the clustering algorithm.

Limitations

Limitation of deep clustering

- Less interpretable latent dimensions
- Requires more data
- Harder to tune and debug
- Training instability
- More computationally expensive

Extra: Interpolation in Latent Space

What's the advantage of it?

- latent space allows easy interpolation
- move between samples in latent space and reconstruct novel instances by the decoder
- not easily possible using other non-linear layouts like MDS, T-SNE

See an example here

Clustering of Time-Series Data

Time Series Data

Rectangular data set with a temporal component

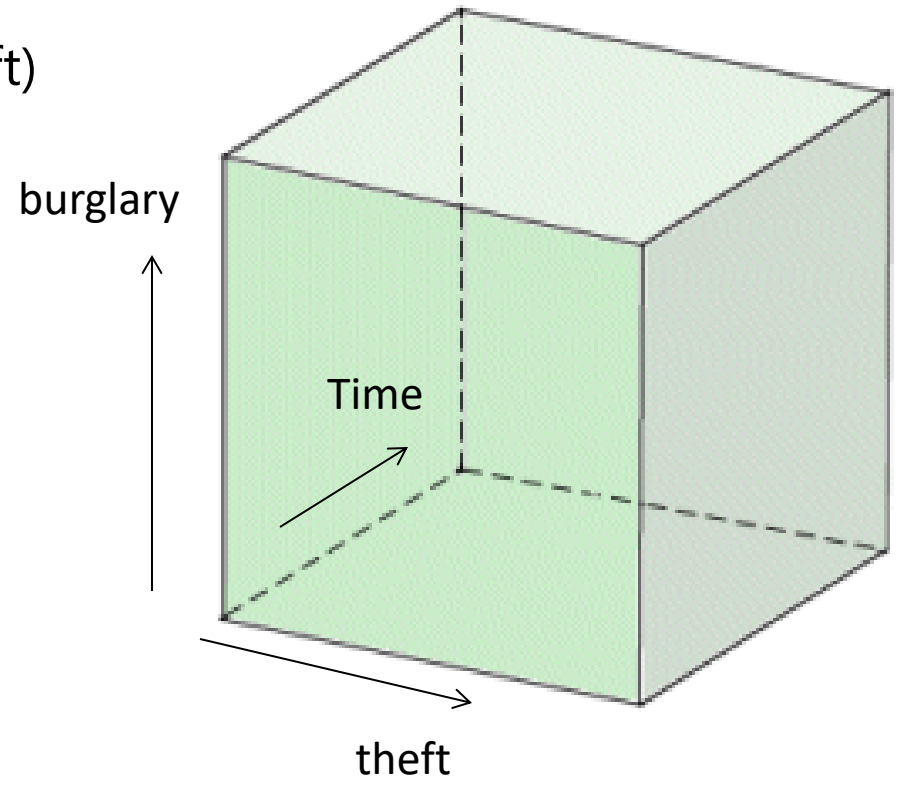
	A	K	L	M	N
1	State	Burglary	Larceny-theft	Motor Vehicle Theft	Arson2
2	CALIFORNIA	2,616	6,298	3,344	71
3	MICHIGAN	1,049	979	154	72
4	MICHIGAN	5,638	8,451	5,828	296
5	TENNESSEE	5,604	12,141	1,373	177
6	MISSOURI	1,960	6,432	1,542	79
7	MARYLAND	3,372	8,761	1,936	137
8	ALABAMA	1,942	3,964	451	
9	OHIO	3,759	5,118	2,008	148
10	ILLINOIS	875	2,242	196	18
11	ARKANSAS	1,852	5,012	589	51
12	CALIFORNIA	2,212	4,450	1,100	43
13	WISCONSIN	2,819	7,622	1,719	120
14	GEORGIA	3,007	8,209	2,328	37

- assume you have these data for each year
- how to handle that, you might ask?

Time Cube

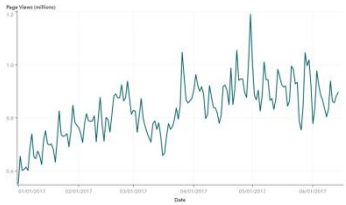
Assume for now we have

- two attributes (burglary, theft)
- both observed over time



Can visualize

- but each point is a time series!



Similarity Measures

Needed it for clustering

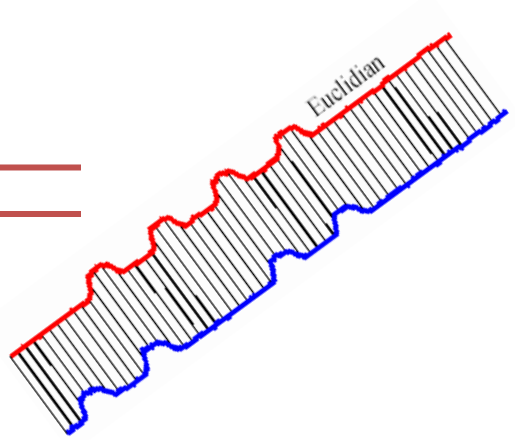
- recall Euclidean, correlation, cosine distances

similarity of two states

similarity of two crimes

	A	K	L	M	N	O
1	State	Burglary	Larceny-theft	Motor Vehicle Theft	Arson2	Violen
2	CALIFORNIA	2,616	6,298	3,344	71	4
3	MICHIGAN	1,049	979	154	72	1
4	MICHIGAN	5,638	8,451	5,828	296	6
5	TENNESSEE	5,604	12,141	1,373	177	5
6	MISSOURI	1,960	6,432	1,542	79	2
7	MARYLAND	3,372	8,761	1,936	137	4
8	ALABAMA	1,942	3,964	451		1
9	OHIO	3,759	5,118	2,008	148	2
10	ILLINOIS	875	2,242	196	18	
11	ARKANSAS	1,852	5,012	589	51	1
12	CALIFORNIA	2,212	4,450	1,100	43	1
13	WISCONSIN	2,819	7,622	1,719	120	3
14	GEORGIA	2,007	8,209	2,328	37	2

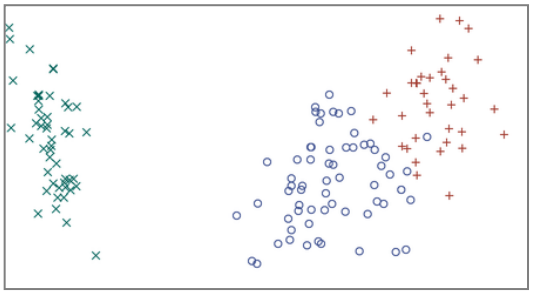
two time series



similarity of two crimes in a given state over time

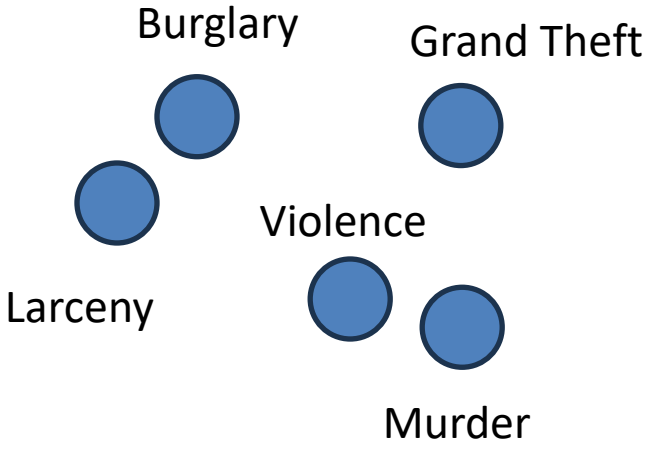
similarity of two states for a given crime over time

Clustering Rows or Columns

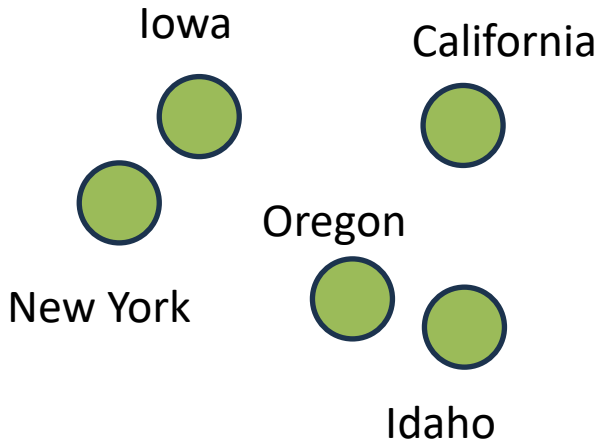


What can be clustered with these measures?

- crimes (averaged over time)
- states (averaged over time)
- crimes in a given state (taking time series into account)
- states for a given crime (taking time series into account)



(Crimes for New York)

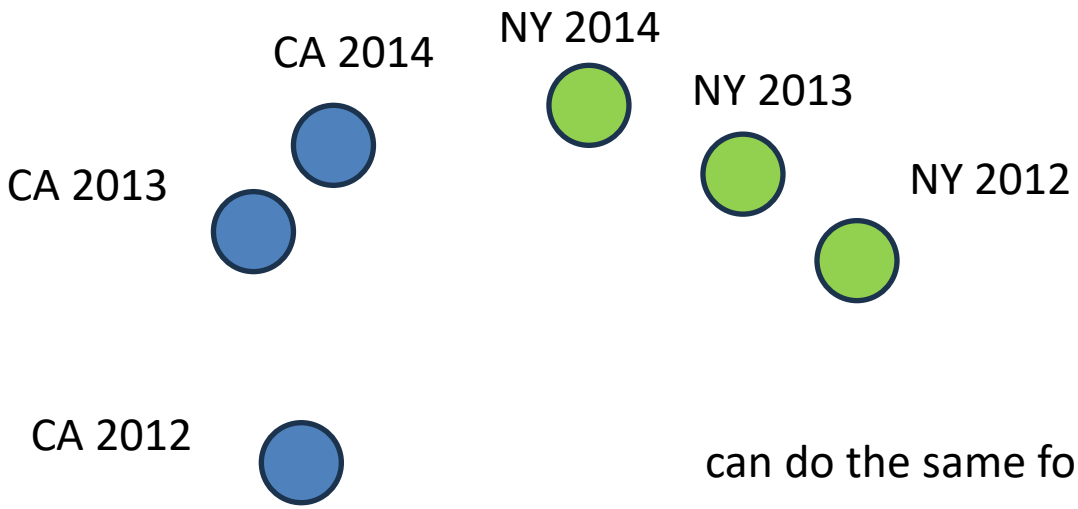


(States for Murder)

Separate Time

You may want to just keep time instances as separate entities

- that will work too
- then you might discover clusters that are sensitive to time
- or you can see how the years relate to another along a trajectory



can do the same for crimes as well

Time Series Alignment

The time series might not be aligned

- one crime might cause another
- can apply dynamic time warping (see next)

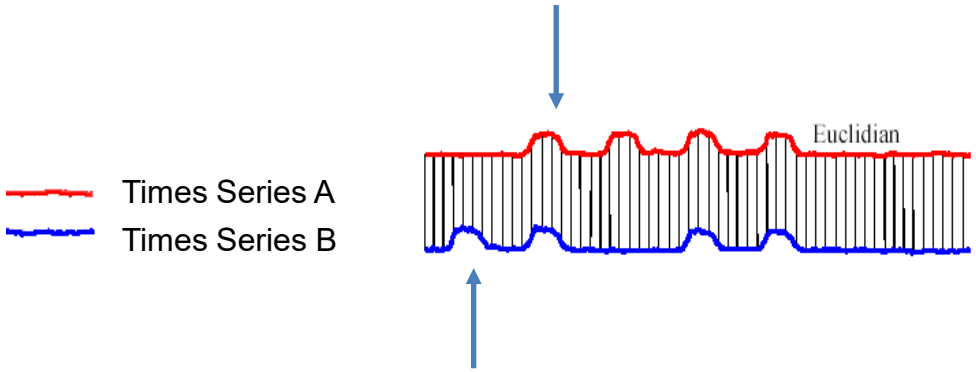
L_P Norm and its Shortcomings

Standard pairwise distance

$$Dist(\overline{X}, \overline{Y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

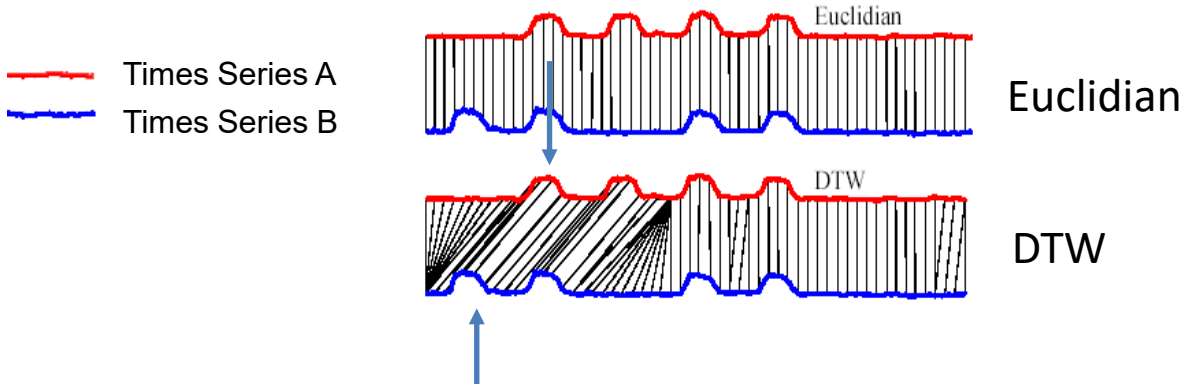
Shortcomings:

- designed for time series of equal length
- cannot address distortions on the temporal (contextual) attributes



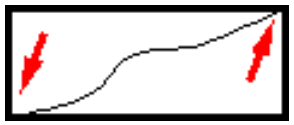
Dynamic Time Warping Distance

Can better accommodate local mismatches



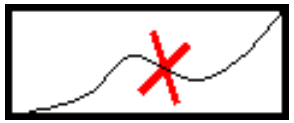
Three constraints

- no skipping of beginning or ends of either sequence

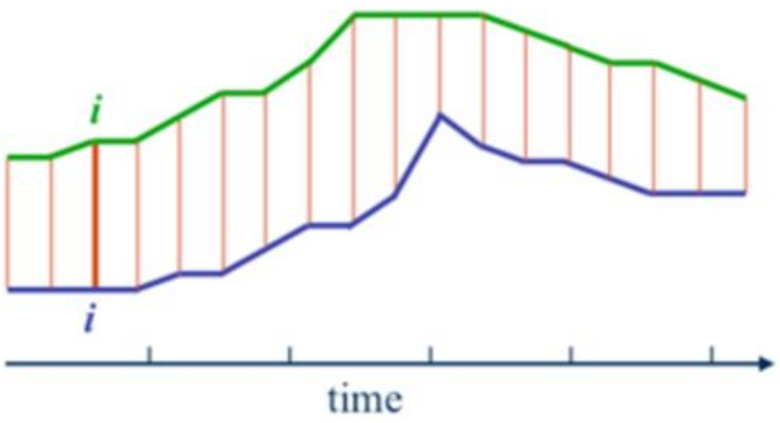


- continuity – no jumps

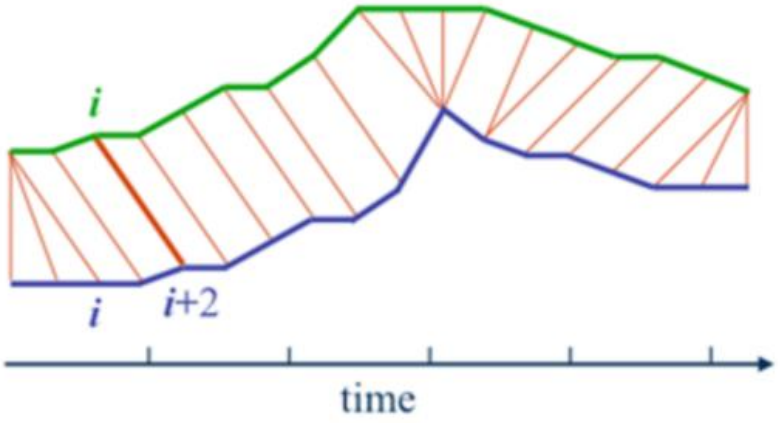
- monotonicity – can't go back in time



DTW – Find The Minimum Cost Path

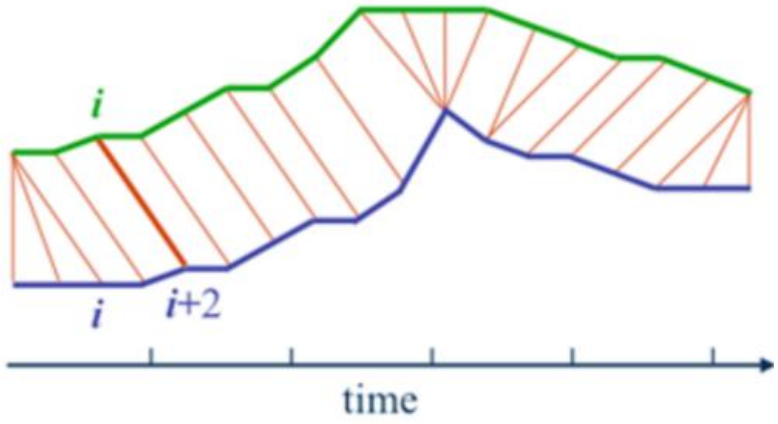


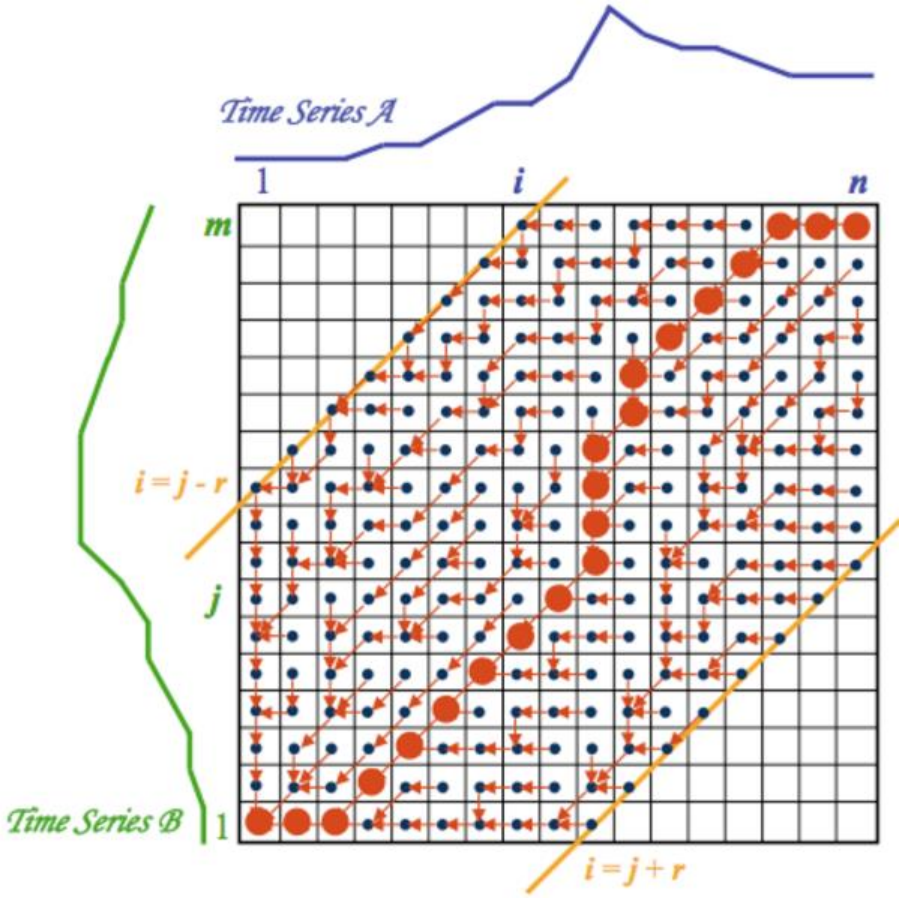
Euclidian



DTW

DTW – Find The Minimum Cost Path





DTW

Compute using dynamic programming

Available in [python]

Summary

Structure ≠ visualization

- Visual layouts may *suggest* clusters
- Clustering must still be defined algorithmically
- Representation learning (e.g., deep clustering) can make structure clearer — but it does not replace critical judgment
- Subsequent visualization is key, possibly accompanies by user interaction -> subject of a future lecture
- Clusters are hypotheses about structure —their usefulness depends on assumptions, representation, and context and can be optimized -> subject of a future lecture

Summary Visualized