DM583

Data Mining

Overlapping Partitioning Clustering

Partition Matrix Revisited

- It is a matrix with k rows (no. of clusters) and N columns (no. of observations) in which entry μ_{ij} stands for the membership degree of the jth observation ($\mathbf{x}_j$) to the ith cluster ($\mathbf{C}_i$)

- In case of **hard** (**non-overlapping**) partitioning algorithms, each observation must belong to exactly one cluster, i.e:

$$\mu_{ij} \in \{0,1\}$$

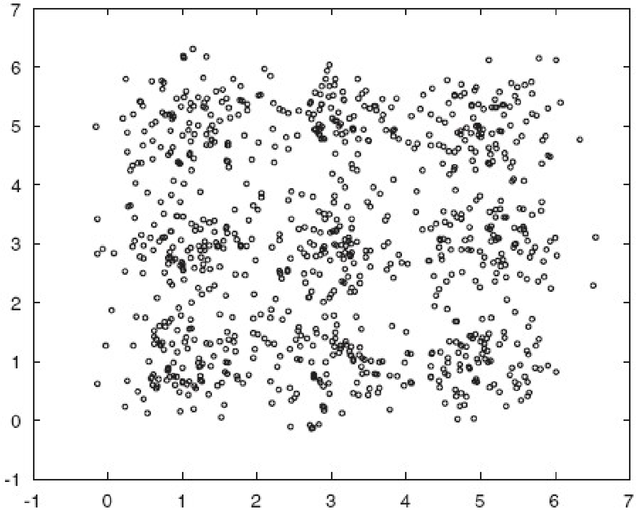
$$\textstyle\sum_i \mu_{ij} = 1 \quad \forall j$$

$$\mathbf{U}(\mathbf{X}) = \left.\overbrace{\begin{bmatrix} \mu_{11} & \mu_{12} & \cdots & \mu_{1N} \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2N} \\ \vdots & & \ddots & \vdots \\ \mu_{k1} & \mu_{k2} & \cdots & \mu_{kN} \end{bmatrix}}^{\text{observations}}\right\}\text{clusters}$$

Hard vs Overlapping Partitioning Clustering

- Partitioning algorithms such as k-means, k-medoids and others produce a **non-overlapping partition** of the data:
  - Each object fully belongs to exactly one cluster
  - Usually, we refer to this type of partition as **Hard** or **Crisp**
- However, many problems involve not clearly delineated clusters that cannot be properly represented in this way

- In other words, there are situations in which the data comprise categories that overlap each other at different levels
- For instance:



Overlapping Partitioning Clustering Algorithms

- **Overlapping clustering algorithms** exist that can cope with these situations
- They can be subdivided into three categories, depending on the type of partition that they produce:
    - **Soft:** Objects can (fully) belong to more than one cluster
    - **Fuzzy:** Objects can partially belong to multiple clusters
        - Cluster membership is a matter of degree (possibly null)
    - **Probabilistic:** There is uncertainty of a probabilistic nature on the association between objects and clusters
        - Cluster membership is interpreted as likelihood

Fuzzy and Probabilistic Partitions

- **Fuzzy Partition Matrix**: real-valued membership values, $\mu_{ij} \in [0,1]$

$$\mathbf{U}(\mathbf{X}) = \begin{bmatrix} \mu_{11} & \mu_{12} & \cdots & \mu_{1N} \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2N} \\ \vdots & & \ddots & \vdots \\ \mu_{k1} & \mu_{k2} & \cdots & \mu_{kN} \end{bmatrix}$$

- **Probabilistic Partition Matrix**: $\mu_{ij} \in [0,1]$ interpreted as probabilities
  - Therefore: $\sum_i(\mu_{ij}) = 1 \ \forall j$

Fuzzy and Probabilistic Partitions

- **Example (Fuzzy):**

$$\mathrm{U}(\mathbf{X}) = \begin{bmatrix} 1 & 0.1 & 0.5 & 0.1 \\ 0 & 0.1 & 0.5 & 0.9 \end{bmatrix}$$

- **Example (Fuzzy / Probabilistic):**

$$\mathrm{U}(\mathbf{X}) = \begin{bmatrix} 1 & 0.7 & 0.5 & 0.1 \\ 0 & 0.3 & 0.5 & 0.9 \end{bmatrix}$$

Model-Based Clustering

- In **Model-Based Clustering** we assume that a dataset is a sample from a population, then we make formal statistical assumptions about this population and use the sample to fit a *parametric model*
- The most well-known and commonly used model is the **Gaussian Mixture Model** (**GMM**), which assumes the data is drawn from a (multi-normal) *mixture distribution*
  - This is a *probabilistic model* *for* *overlapping partitioning clustering*

Model-Based Clustering

- GMM models can be fit using the *EM Algorithm*
- **EM** (**Expectation Maximization**) is a probabilistic modelling procedure based on the principle of **Maximum Likelihood Estimation** (**MLE**)

GMMs

- Gaussian Mixture Models are described by the following probability density function ***p***:

$$p(\mathbf{x}_j) = \sum_{i=1}^{k} \pi_i \mathcal{N}(\mathbf{x}_j | \mathbf{v}_i, \mathbf{\Sigma}_i)$$

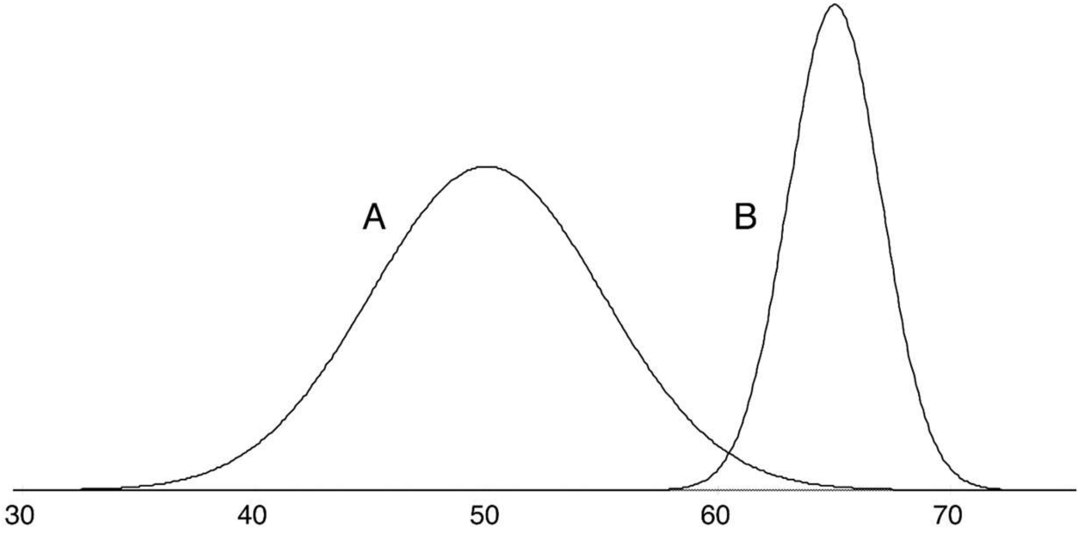
  - $\mathbf{x}_j$ is a vector of n real-valued random variables (a data object in $\Re^n$)
  - $\mathcal{N}$ is a multivariate Gaussian (same dimension as the objects)
    - $\mathbf{v}_i$ is the centre of the ith Gaussian (vector of same dimension as $\mathbf{x}_j$)
    - Σ_i is the covariance matrix of the ith Gaussian
  - π_i is the prior probability associated with the ith Gaussian
  - k is the number of Gaussians

GMMs: 1-dimensional Example

Objects:

A	51	B	62	B	64	A	48	A	39	A	51
A	43	A	47	A	51	B	64	B	62	A	48
B	62	A	52	A	52	A	51	B	64	B	64
B	64	B	64	B	62	B	63	A	52	A	42
A	45	A	51	A	49	A	43	B	63	A	48
A	42	B	65	A	48	B	65	B	64	A	41
A	46	A	48	B	62	B	66	A	48		
A	45	A	49	A	43	B	65	B	64		
A	45	A	46	A	40	A	46	A	48		

Model:



$\nu_A=50$, $\sigma_A=5$, $\pi_A=0.6$ $\nu_B=65$, $\sigma_B=2$, $\pi_B=0.4$

Witten & Frank, Data Mining: Practical Machine Learning Tools and Techniques (Chapter 6)

EM for GMMs

- The following quantity plays a fundamental role when fitting GMMs with EM:

$$\boxed{\mu_{ij} = \frac{\pi_i \mathcal{N}\left(\mathbf{x}_j \middle| \mathbf{v}_i, \boldsymbol{\Sigma}_i\right)}{\sum_{l=1}^{k} \pi_l \mathcal{N}\left(\mathbf{x}_j \middle| \mathbf{v}_l, \boldsymbol{\Sigma}_l\right)}}$$

- μ_{ij} is the **posterior probability** that an observed value $\mathbf{x}_j$ was generated by the ith cluster (i.e., by the ith multivariate Gaussian component)
  - The probability associated with the ith component *conditioned* to $\mathbf{x}_j$
  - It trivially follows from the Bayes theorem

EM for GMMs

- Another fundamental quantity (**likelihood**):
  - Given a sample (dataset) $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ with N i.i.d. observations $\mathbf{x}_i \in \Re^n$, their joint distribution is:
    - $p(\mathbf{X}) = p(\mathbf{x}_1 \,\&\, \mathbf{x}_2 \,\&\ldots\&\, \mathbf{x}_N) = \prod_{j=1}^{N} p(\mathbf{x}_j) = \prod_{j=1}^{N} \sum_{l=1}^{k} \pi_l \mathcal{N}(\mathbf{x}_j | \mathbf{v}_l, \mathbf{\Sigma}_l)$
  - This is the likelihood that a sample **X** will be observed/drawn from the Gaussian mixture distribution in question, with parameters:
    - $\Sigma = \{\Sigma_1, \ldots, \Sigma_k\}$, $\mathbf{v} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ and $\pi = \{\pi_1, \ldots, \pi_k\}$
    - For this reason, we also use the notation p(**X** | π, Σ, **v**)
  - For mathematical convenience, a log transformation is applied to replace products with sums, resulting in the **log-likelihood** function:
    - $\ln(p(\mathbf{X} | \boldsymbol{\pi}, \mathbf{\Sigma}, \mathbf{v})) = \sum_{j=1}^{N} \ln\left(\sum_{l=1}^{k} \pi_l \mathcal{N}(\mathbf{x}_j | \mathbf{v}_l, \mathbf{\Sigma}_l) \right)$

Ricardo Campello

EM for GMMs

- Maximising the likelihood (or, equivalently, the log-likelihood) can be seen as maximising the agreement/match between the sample and the model
- EM (Dempster et al., 1977) is an optimisation algorithm that aims to maximise the (log) likelihood function in 2 steps:
  - **Step E** (Expectation)
    - Evaluate the posterior probabilities μ_{ij} (i = 1, ..., k; j = 1, ..., N)
      - from the N observations $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ and the current model, given by parameters $\Sigma = \{\Sigma_1, \ldots, \Sigma_k\}$, $\mathbf{v} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ and $\pi = \{\pi_1, \ldots, \pi_k\}$
  - **Step M** (Maximisation)
    - Adjust the model parameters to locally maximise the log-likelihood function

EM for GMMs

- **Step E** (Expectation):
  - Evaluate the posterior probabilities μ_{ij} (i = 1, ..., k; j = 1, ..., N)

$$\mu_{ij} = \frac{\pi_i \mathcal{N}\left(\mathbf{x}_j \middle| \mathbf{v}_i, \boldsymbol{\Sigma}_i\right)}{\sum_{l=1}^{k} \pi_l \mathcal{N}\left(\mathbf{x}_j \middle| \mathbf{v}_l, \boldsymbol{\Sigma}_l\right)}$$

$$\mathcal{N}\left(\mathbf{x}_j \mid \mathbf{v}_i, \boldsymbol{\Sigma}_i\right) = \frac{1}{(2\pi)^{n/2} \det\left(\boldsymbol{\Sigma}_i\right)^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_j - \mathbf{v}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \mathbf{v}_i)\right\}$$

EM for GMMs

- **Step E** (Expectation):
  - Evaluate the posterior probabilities μ_{ij} (i = 1, …, k; j = 1, …, N)

$$\mu_{ij} = \frac{\pi_i \mathcal{N}\left(\mathbf{x}_j \middle| \mathbf{v}_i, \boldsymbol{\Sigma}_i\right)}{\sum_{l=1}^{k} \pi_l \mathcal{N}\left(\mathbf{x}_j \middle| \mathbf{v}_l, \boldsymbol{\Sigma}_l\right)}$$

$$\mathcal{N}\left(\mathbf{x}_j \mid \mathbf{v}_i, \boldsymbol{\Sigma}_i\right) = \frac{1}{(2\pi)^{n/2} \det\left(\boldsymbol{\Sigma}_i\right)^{1/2}} \exp\left\{ -\frac{1}{2} \boxed{(\mathbf{x}_j - \mathbf{v}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \mathbf{v}_i)} \right\}$$

Mahalanobis distance (squared)
from observation to cluster center

EM for GMMs

- **Step M** (Maximisation):
  - Adjust the model parameters

$$\left\{\begin{aligned} \mathbf{v}_i &= \frac{1}{N_i}\sum_{j=1}^{N}\mu_{ij}\mathbf{x}_j && \Rightarrow \text{ weighted centroid} \\ \boldsymbol{\Sigma}_i &= \frac{1}{N_i}\sum_{j=1}^{N}\mu_{ij}\left(\mathbf{x}_j-\mathbf{v}_i\right)\left(\mathbf{x}_j-\mathbf{v}_i\right)^T && \Rightarrow \text{ weighted covariance} \\ \pi_i &= \frac{N_i}{N} && \Rightarrow \text{ relative “responsibility” of the ith cluster (Gaussiana)} \end{aligned}\right.$$

$$N_i = \sum_{j=1}^{N}\mu_{ij} \quad \Rightarrow \text{ absolute “responsibility” of the ith cluster (Gaussiana)}$$

EM for GMMs

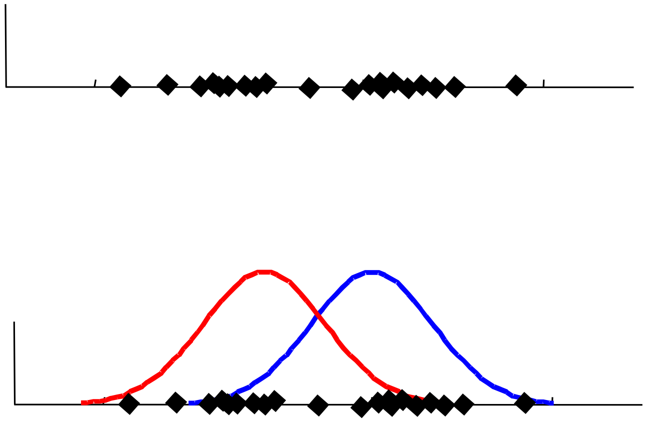
- **Algorithm**:

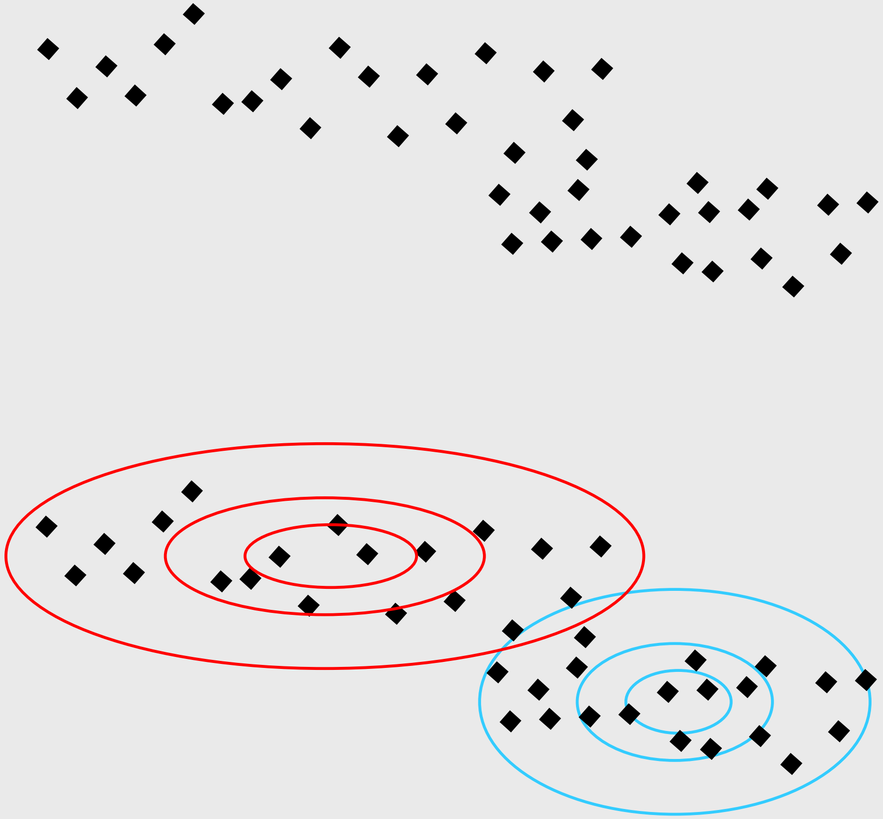
1. **Initialisation** (e.g. via k-means)
   - prototypes $\mathbf{v}_i$ = resulting k-means centroids
   - covariances Σ_i = sample covariance matrices of the resulting clusters
   - probabilities μ_{ij} (for N_i and π_i) = resulting (hard) partition matrix
2. **Step E**
3. **Step M**
4. **Evaluate Stopping Criterion**
   - e.g. Log-likelihood function, no. iterations, ...
5. **Stop or Return to Step 2**

EM (GMMs) × k-Means

- EM provides much more information about the data
  - PDF model describing the distributions of clusters
  - Posterior probabilities for each observation
  - Outliers...
- It can model elongated, ellipsoidal clusters with arbitrary covariances
- However, this comes with a price:
  - GMMs have a much larger number of parameters to be fit (more data needed, etc.)
  - Computing the inverse of the covariance matrices $\sum_i$ requires $O(n^3)$ time
    - There are variants and simplified versions that are more robust and/or faster (e.g., see **MCLUST**)
- k-means can be shown to be a particular limit of EM-GMM
  - Both are subject to local minima

EM-GMMs (Examples)

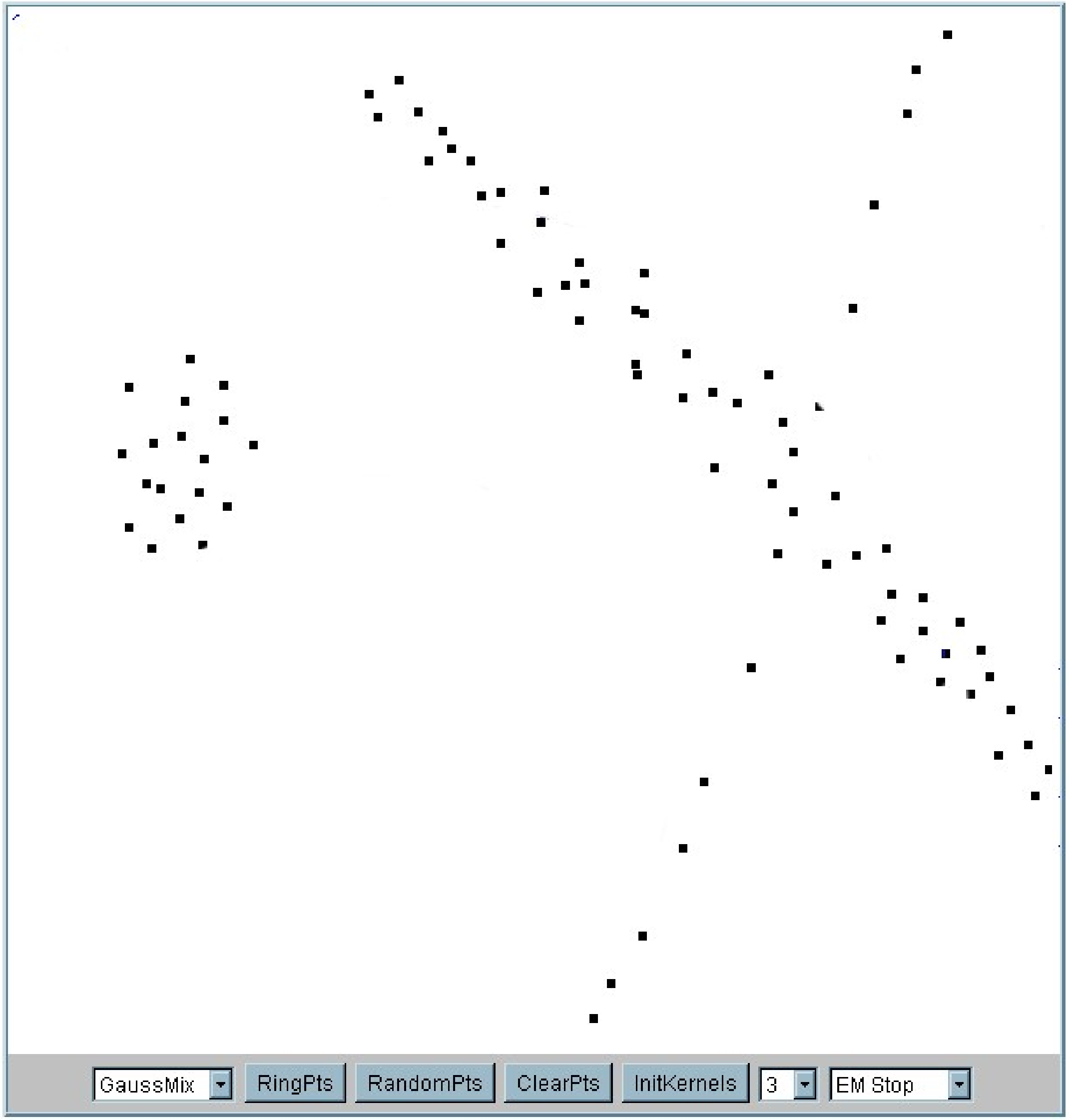




Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Brazil.

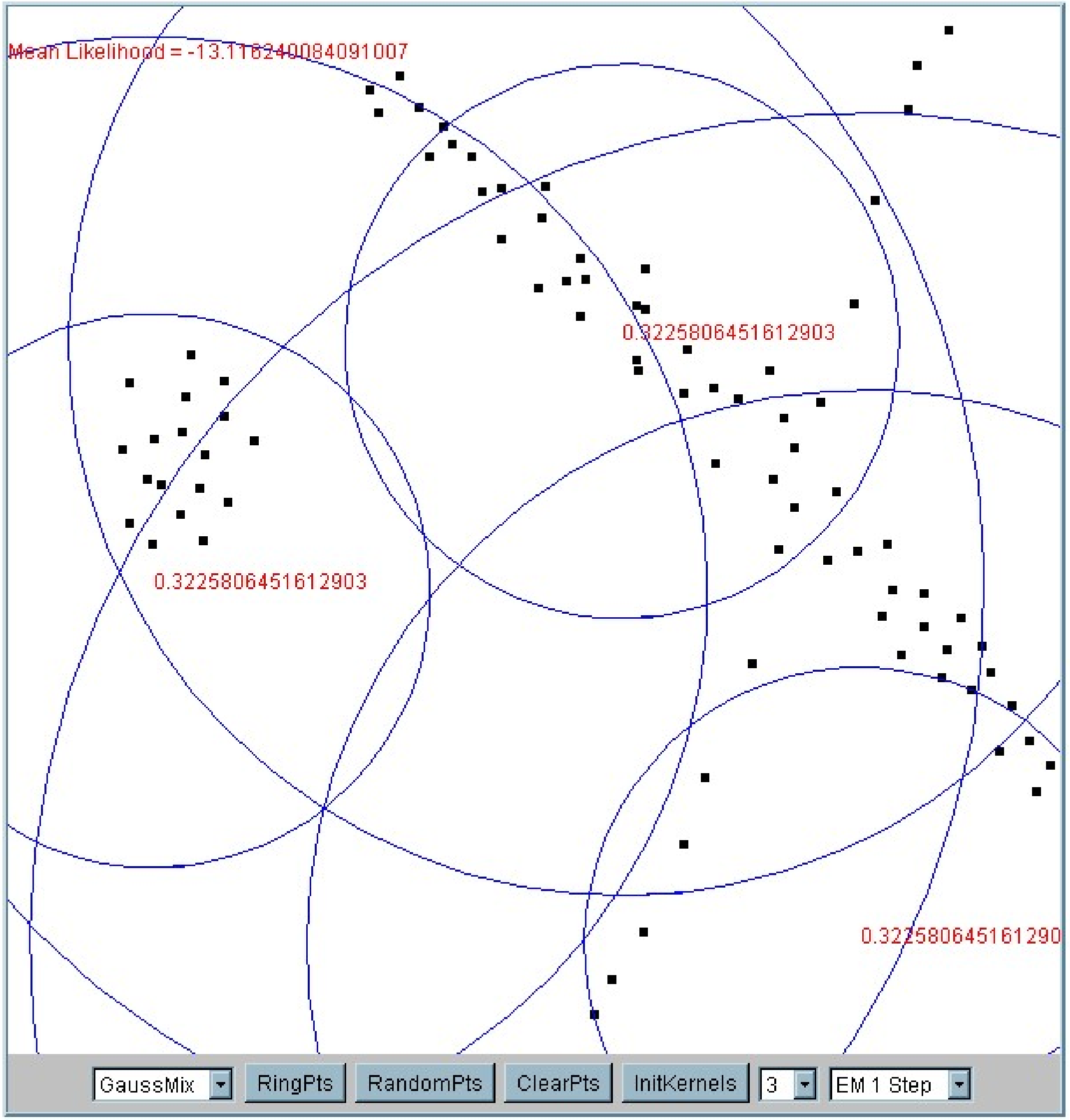
Example

(step-by-step)



Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Brazil.

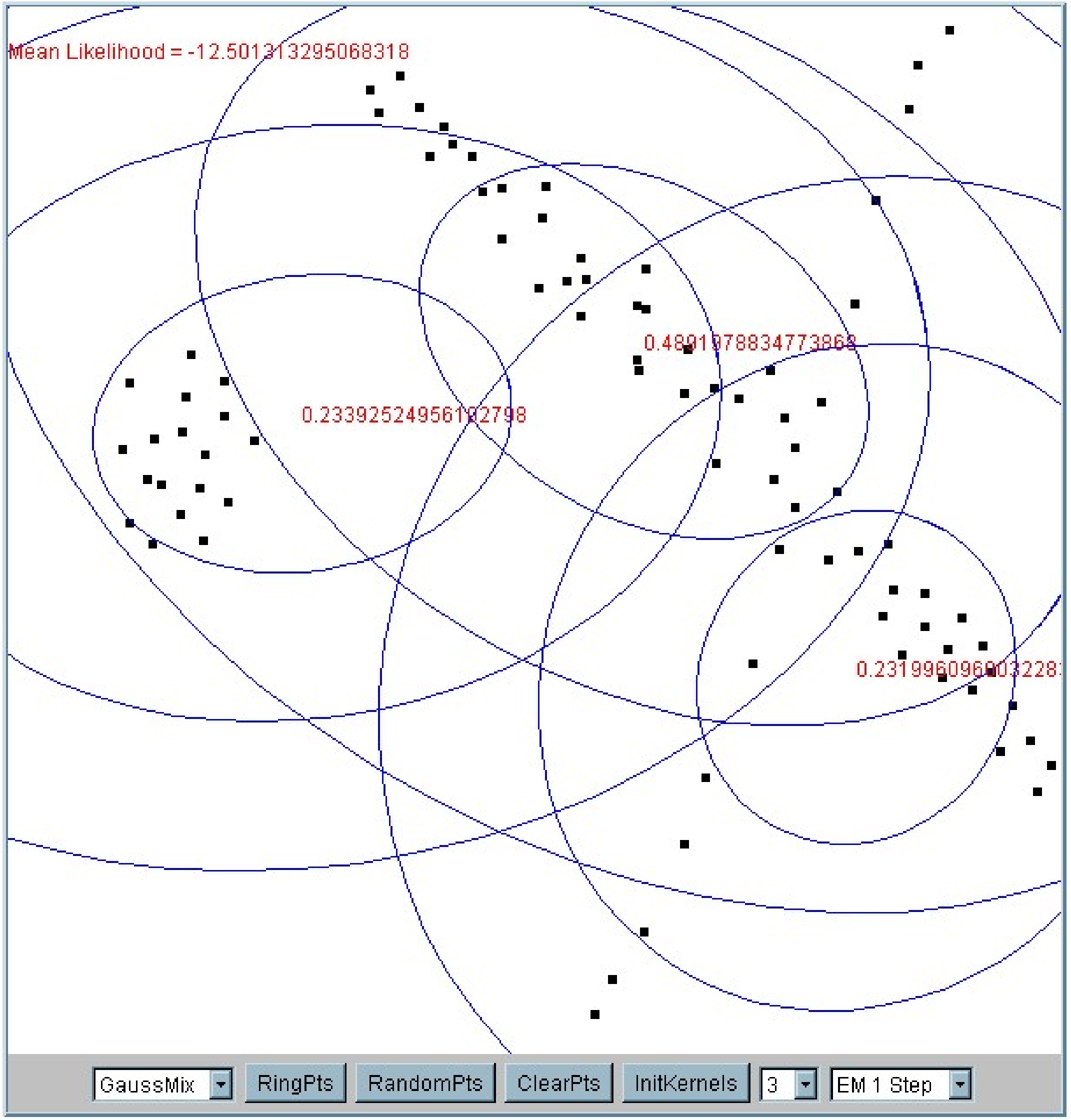
Iteration 1



Mean Likelihood = -13.116240084091007

Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Brazil.

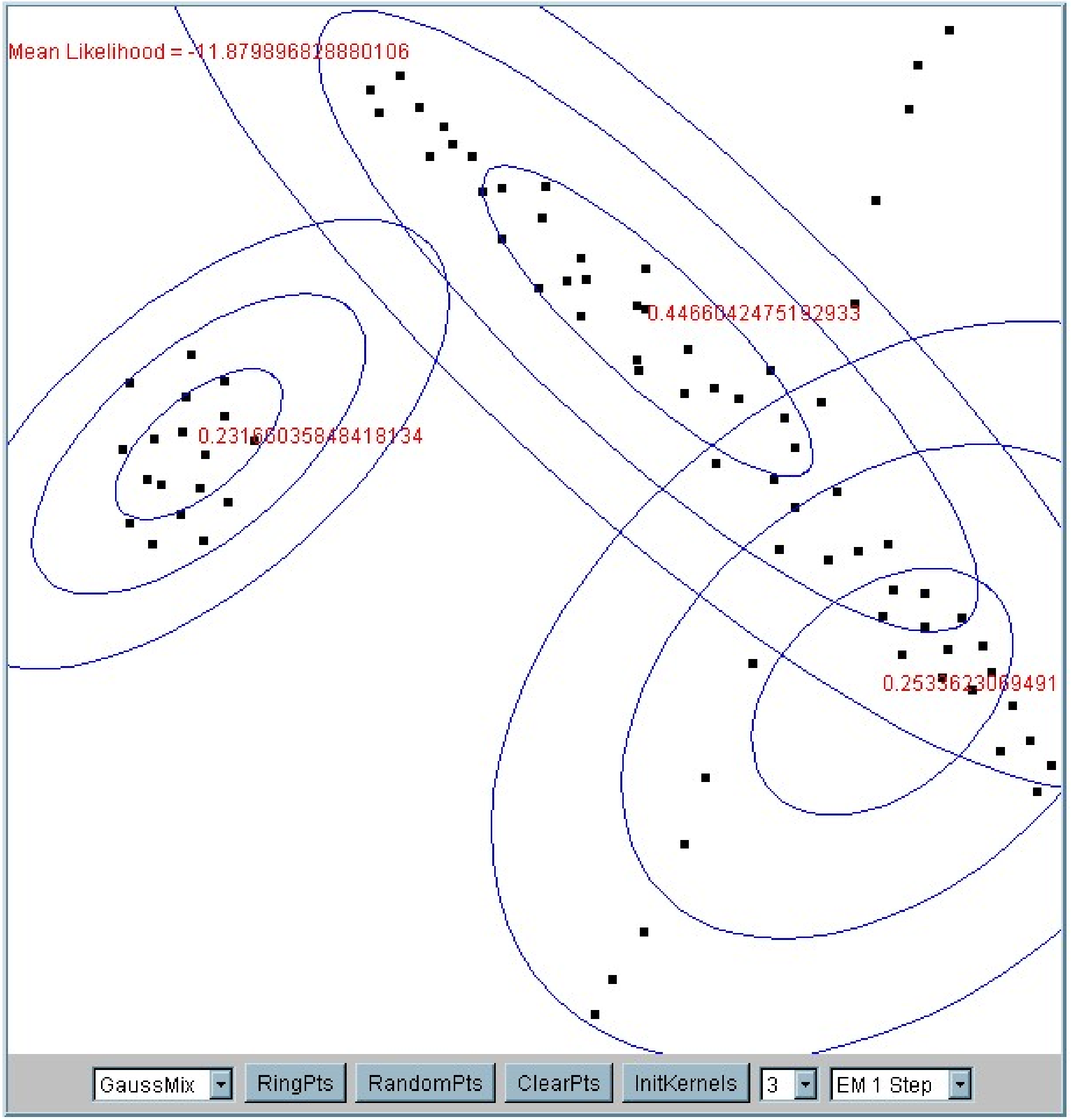
Iteration 2



Mean Likelihood = -12.501313295068318

Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Brazil.

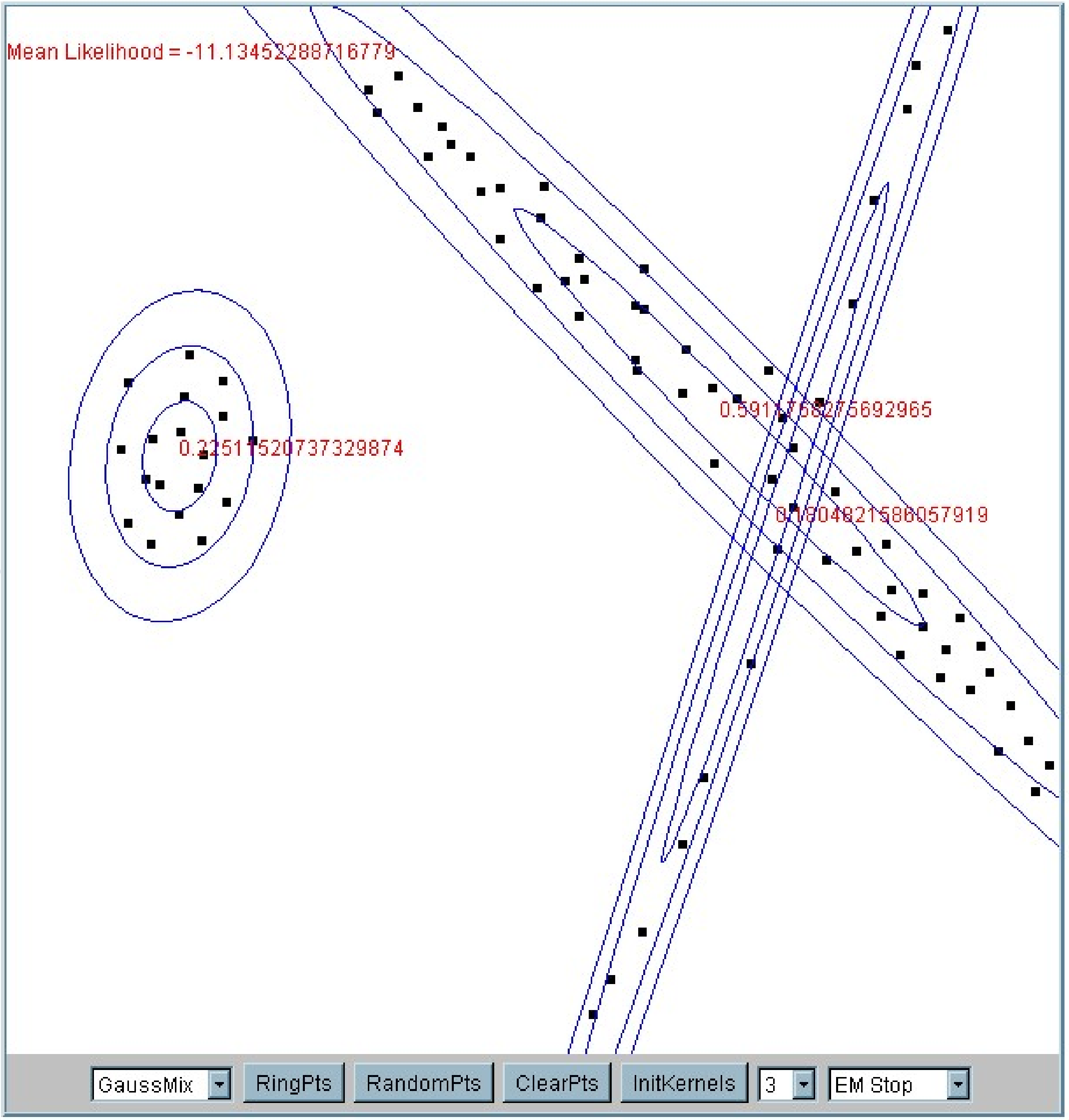
Iteration 5



Mean Likelihood = -11.879896828880106

Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Brazil.

Iteration 25



```
Mean Likelihood = -11.13452288716779
0.22511520737329874
0.5911768275692965
0.1804821586057919
```

Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Brazil.

Exercise

Object ID	x_1
1	-1.31
2	-0.43
3	0.34
4	3.57
5	2.76
6	0.30
7	9.06
8	4.45
9	2.87
10	4.42

- Manually perform EM in this dataset (n = 1, N = 10), with k = 2. Start from arbitrary/random clusters

- Illustrate the results graphically

MCLUST Framework [OPTIONAL]

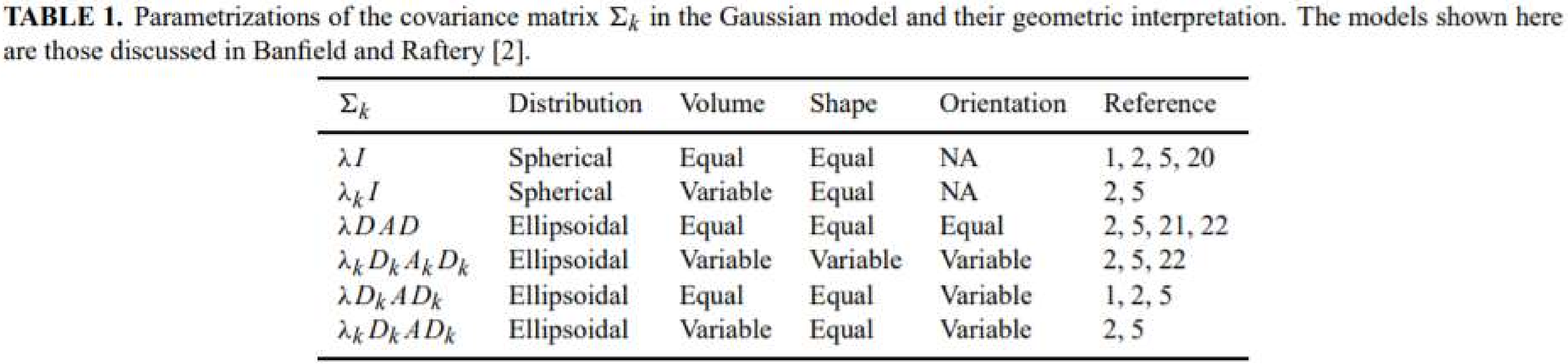


TABLE 1. Parametrizations of the covariance matrix Σ_k in the Gaussian model and their geometric interpretation. The models shown here are those discussed in Banfield and Raftery [2].

Σ_k	Distribution	Volume	Shape	Orientation	Reference
λI	Spherical	Equal	Equal	NA	1, 2, 5, 20
$\lambda_k I$	Spherical	Variable	Equal	NA	2, 5
λDAD	Ellipsoidal	Equal	Equal	Equal	2, 5, 21, 22
$\lambda_k D_k A_k D_k$	Ellipsoidal	Variable	Variable	Variable	2, 5, 22
$\lambda D_k A D_k$	Ellipsoidal	Equal	Equal	Variable	1, 2, 5
$\lambda_k D_k A D_k$	Ellipsoidal	Variable	Equal	Variable	2, 5

- C. Fraley and Adrian E. Raftery "How many clusters? Which clustering method? Answers via model-based cluster analysis" The computer journal 41.8 (1998): 578-588
- C. Fraley and Adrian E. Raftery "MCLUST: Software for model-based cluster analysis" *Journal of classification* 16.2 (1999): 297-306.

References

- Höppner, F., Klawonn, F., Kruse, R., Runkler, T., *Fuzzy Cluster Analysis*, 1999
- Bezdek, J. C., *Pattern Recognition with Fuzzy Objective Function Algorithm*, Plenum Press, 1981
- Bishop, C. M., *Pattern Recognition and Machine Learning*, Springer, 2006
- I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Edition, Morgan Kaufmann, 2005
- Tan, P.-N., Steinbach, M., and Kumar, V., *Introduction to Data Mining*, Addison-Wesley, 2006