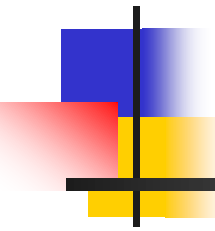
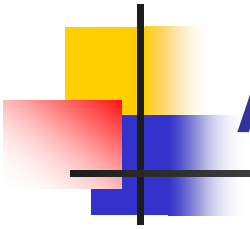


# SCC5895 – Análise de Agrupamento de Dados

## Algoritmos Particionais: Parte II

**Prof. Ricardo J. G. B. Campello**

PPG-CCMC / ICMC / USP

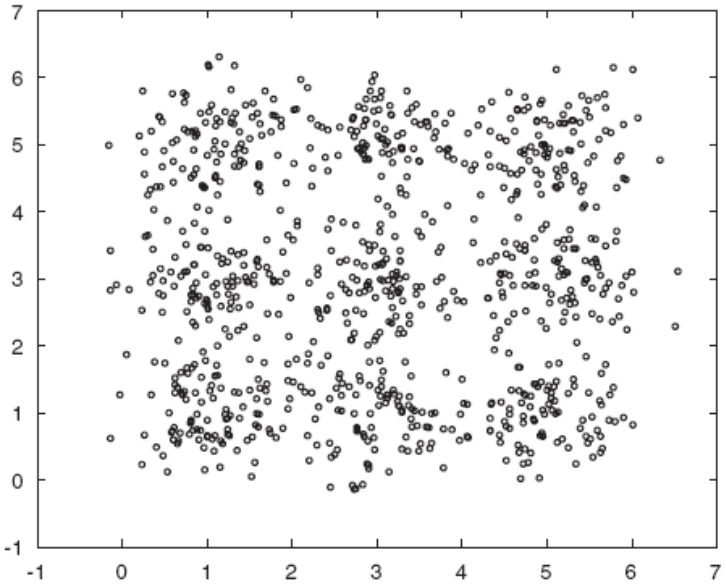


# Aula de Hoje

- Partições Com Sobreposição
  - Soft, Fuzzy e Probabilísticas
- Algoritmos Particionais Com Sobreposição
  - Fuzzy C-Means (FCM)
  - Expectation Maximization (EM)
- Algoritmos Baseados em Densidade
  - DBSCAN e Variantes

# Algoritmos de Partição Com Sobreposição

- Algoritmos particionais como o k-means, k-medóides e diversos outros produzem uma **partição rígida** da base de dados:
  - Cada objeto pertence a um único grupo, de forma integral
  - Usualmente refere-se a esse tipo de partição como **Hard** ou **Crisp**
- No entanto, muitos problemas envolvem grupos mal delineados, que não podem ser separados adequadamente dessa maneira
- Em outras palavras, existem situações nas quais os dados compreendem categorias que se sobrepõem umas às outras em diferentes níveis
- Por exemplo:



# Algoritmos de Partição Com Sobreposição

- Métodos de agrupamento com sobreposição, ou **overlapping clustering algorithms**, são concebidos para lidar com situações como esta
- Podem gerar partições de 3 tipos:
  - **Soft:** Objetos podem pertencer a mais de um grupo, porém necessariamente de forma integral
  - **Fuzzy:** Objetos pertencem a todos os grupos, com diferentes graus ou níveis de pertinência (possivelmente nulo)
  - **Probabilísticas:** Objetos possuem probabilidades de pertinência associadas a cada grupo
- Vamos discutir representantes clássicos dos últimos dois tipos, que são os mais comumente utilizados

# Definição de Partição Soft de Dados

- Consideremos um conjunto de $N$ objetos a serem agrupados: $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$

- **Partição Soft**: coleção de $k$ grupos, dada por $\mathbf{P} = \{\mathbf{C}_1, \mathbf{C}_2, ..., \mathbf{C}_k\}$, tal que:

$$\mathbf{C}_1 \cup \mathbf{C}_2 \cup ... \cup \mathbf{C}_k = \mathbf{X}$$

$$\mathbf{C}_i \neq \emptyset$$

  - Exemplo: $\mathbf{P} = \{ (\mathbf{x}_1, \mathbf{x}_4), (\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6), (\mathbf{x}_2, \mathbf{x}_5) \}$

# Matriz de Partição Soft

- **Matriz de Partição** (revisão): matriz com $k$ linhas (no. de grupos) e $N$ colunas (no. de objetos) na qual cada elemento $\mu_{ij}$ indica o *grau de pertinência* do j-ésimo objeto ($\mathbf{x}_j$) ao i-ésimo grupo ($\mathbf{C}_i$)

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

- **Matriz de Partição Soft**: matriz de partição **binária**, ou seja, $\mu_{ij} \in \{0,1\}$, sem qualquer outra restrição, exceto colunas não nulas

# Matriz de Partição Soft

- **Exemplo:**
  - $\mathbf{P} = \{ (\mathbf{x}_1, \mathbf{x}_4), (\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6), (\mathbf{x}_2, \mathbf{x}_5) \}$

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

# Partições Fuzzy e Probabilísticas

- **Matriz de Partição Fuzzy**: elementos assumem valores contínuos de pertinência, ao invés de binários, i.e., $\mu_{ij} \in [0,1]$

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

- **Matriz de Partição Probabilística**: matriz de partição fuzzy particular, tal que pertinências representam probabilidades
  - Logo, a seguinte restrição deve ser satisfeita: $\sum_i(\mu_{ij}) = 1 \ \forall j$

# Partições Fuzzy e Probabilísticas

- **Exemplo (Fuzzy):**

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

- **Exemplo (Fuzzy / Probabilística):**

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

# Algoritmo Fuzzy c-Means

- **Modelo de Otimização (Dunn 1973, Bezdek 1981):**

$$\min_{\mu_{ij},\,\mathbf{v}_i} \quad J = \sum_{j=1}^{N} \sum_{i=1}^{c} \mu_{ij}^{m} \left\| \mathbf{x}_j - \mathbf{v}_i \right\|^2$$

$$s.a \quad 0 \le \mu_{ij} \le 1$$

$$\sum_{i=1}^{c} \mu_{ij} = 1 \qquad \forall j \in \{1, 2, \ldots, N\}$$

$$0 < \sum_{j=1}^{N} \mu_{ij} < N \qquad \forall i \in \{1, 2, \ldots, c\}$$

onde $m > 1$ (parâmetro) e $\mathbf{v}_i \in \Re^n$ (**protótipos**)

# Algoritmo Fuzzy c-Means

**NOTAS:**

- Qual a interpretação do critério de custo $\boldsymbol{J}$ ... ?
- Qual a interpretação das restrições ... ?
- Qual o papel do “fator de fuzzificação” $\boldsymbol{m}$ ... ?

# Algoritmo Fuzzy c-Means

- **Algoritmo Básico** ($m = 2$, $\mathbf{v}_i \neq \mathbf{x}_j$)**:**

1 – Selecionar centros iniciais : $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c$

2 – Calcular $\mu_{ij}$ :

$$\mu_{ij} = \left[ \sum_{l=1}^{c} \frac{\left\| \mathbf{x}_j - \mathbf{v}_i \right\|^2}{\left\| \mathbf{x}_j - \mathbf{v}_l \right\|^2} \right]^{-1}$$

3 – Atualizar os centros :

$$\mathbf{v}_i = \frac{\sum_{j=1}^{N} \mu_{ij}^2 \, \mathbf{x}_j}{\sum_{j=1}^{N} \mu_{ij}^2}$$

4 – Parar em caso de convergênc ia ou voltar ao passo 2

# Algoritmo Fuzzy c-Means

- **Algoritmo Completo:**

1 – Selecionar os centros iniciais : $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c$

2 – Calcular $\mu_{ij}$ : Para cada $j \in \{1, \cdots, N\}$, se $\|\mathbf{x}_j - \mathbf{v}_i\| > 0$ para $i = 1, \cdots, c$ então

$$\mu_{ij} = \left[ \sum_{l=1}^{c} \left( \frac{\|\mathbf{x}_j - \mathbf{v}_i\|}{\|\mathbf{x}_j - \mathbf{v}_l\|} \right)^{\frac{2}{m-1}} \right]^{-1}$$

Se $\|\mathbf{x}_j - \mathbf{v}_i\| = 0$ para $i \in I \subseteq \{1, \cdots, c\}$, então definir $\mu_{ij}$ para $i \in I$ como $\forall$ número real não negativo que satisfaça $\sum_{i \in I} \mu_{ij} = 1$ e definir $\mu_{ij} = 0$ para $i \in \{1, \cdots, c\} - I$

3 – Atualizar os centros :

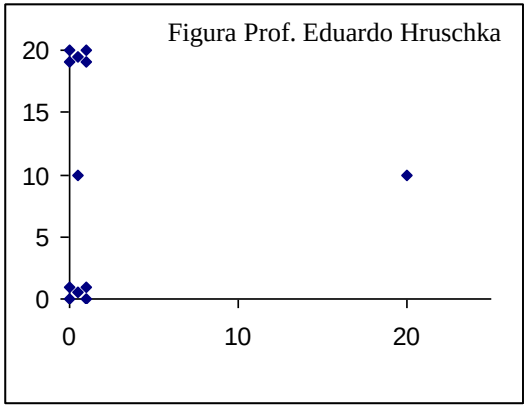
$$\mathbf{v}_i = \frac{\sum_{j=1}^{N} \mu_{ij}^m \, \mathbf{x}_j}{\sum_{j=1}^{N} \mu_{ij}^m}$$

4 – Parar em caso de convergênc ia ou voltar ao passo 2

# Algoritmo Fuzzy c-Means

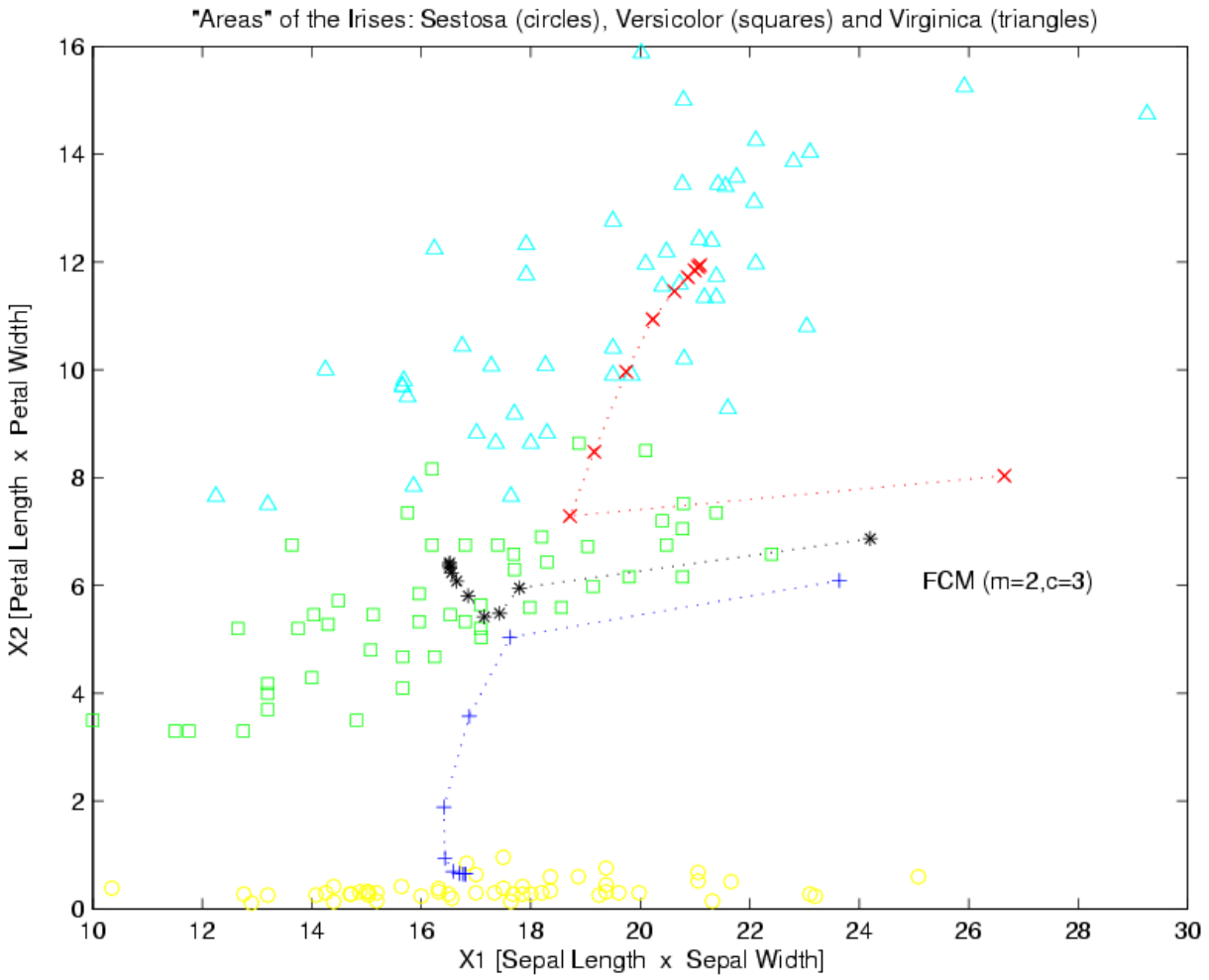
## Notas:

- Valor ótimo de $m$ desconhecido. Usualmente $m = 2$
- Trata-se de uma extensão de k-means para o domínio fuzzy
  - Como tal $\Rightarrow$ apenas garantia de convergência para soluções locais !
  - Ou seja, também é susceptível a mínimos locais da função objetivo $J$
    - depende da inicialização dos protótipos
    - esquemas de múltiplas execuções podem ser utilizados...
- Existem dezenas de variantes !
  - e.g. **versão possibilística**
    - considere a figura ao lado com c = 2
    - pertinência dos dois outliers... ?
  - e.g. versão “elipsoidal” (Gustafson-Kessel)
  - ...

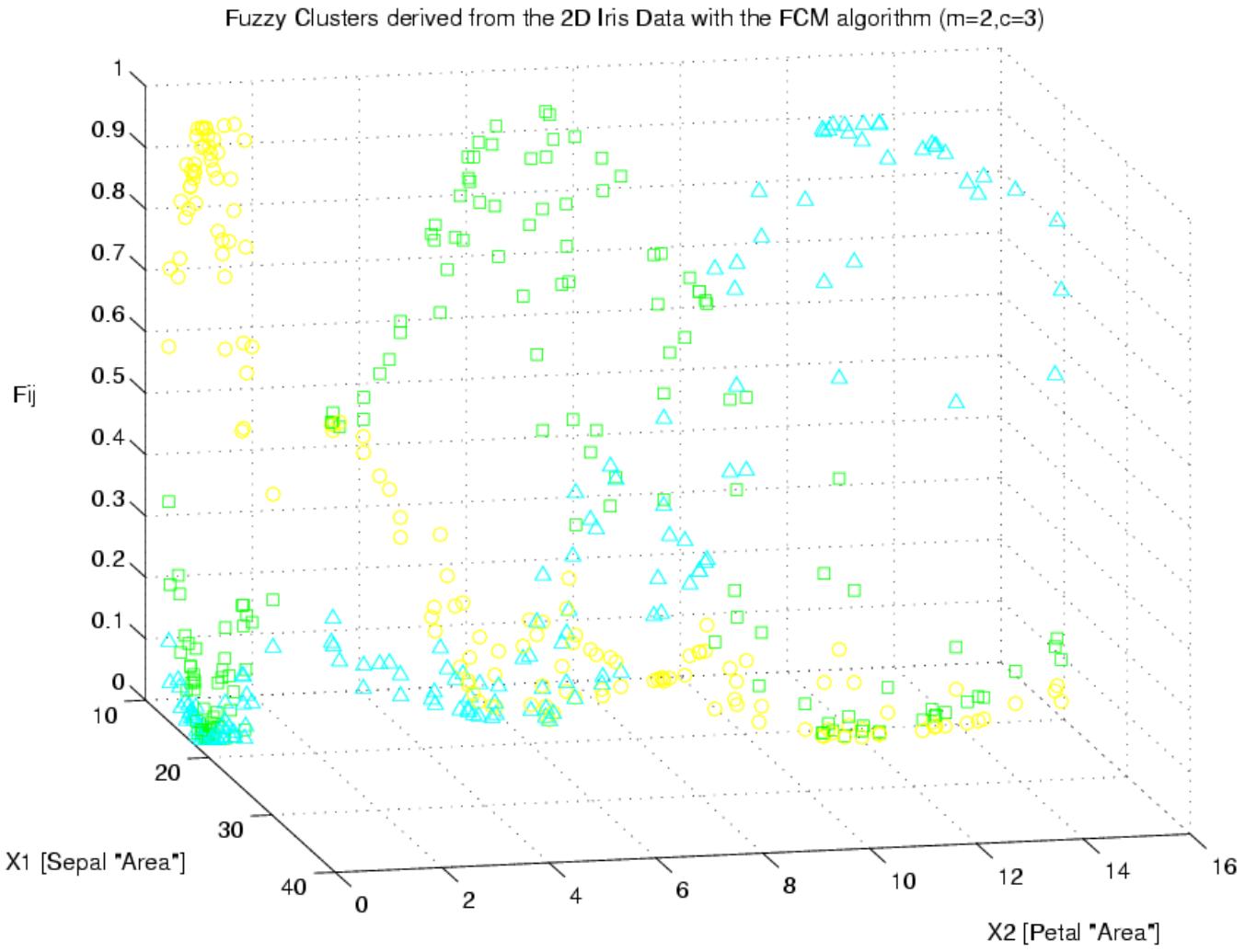


# FCM

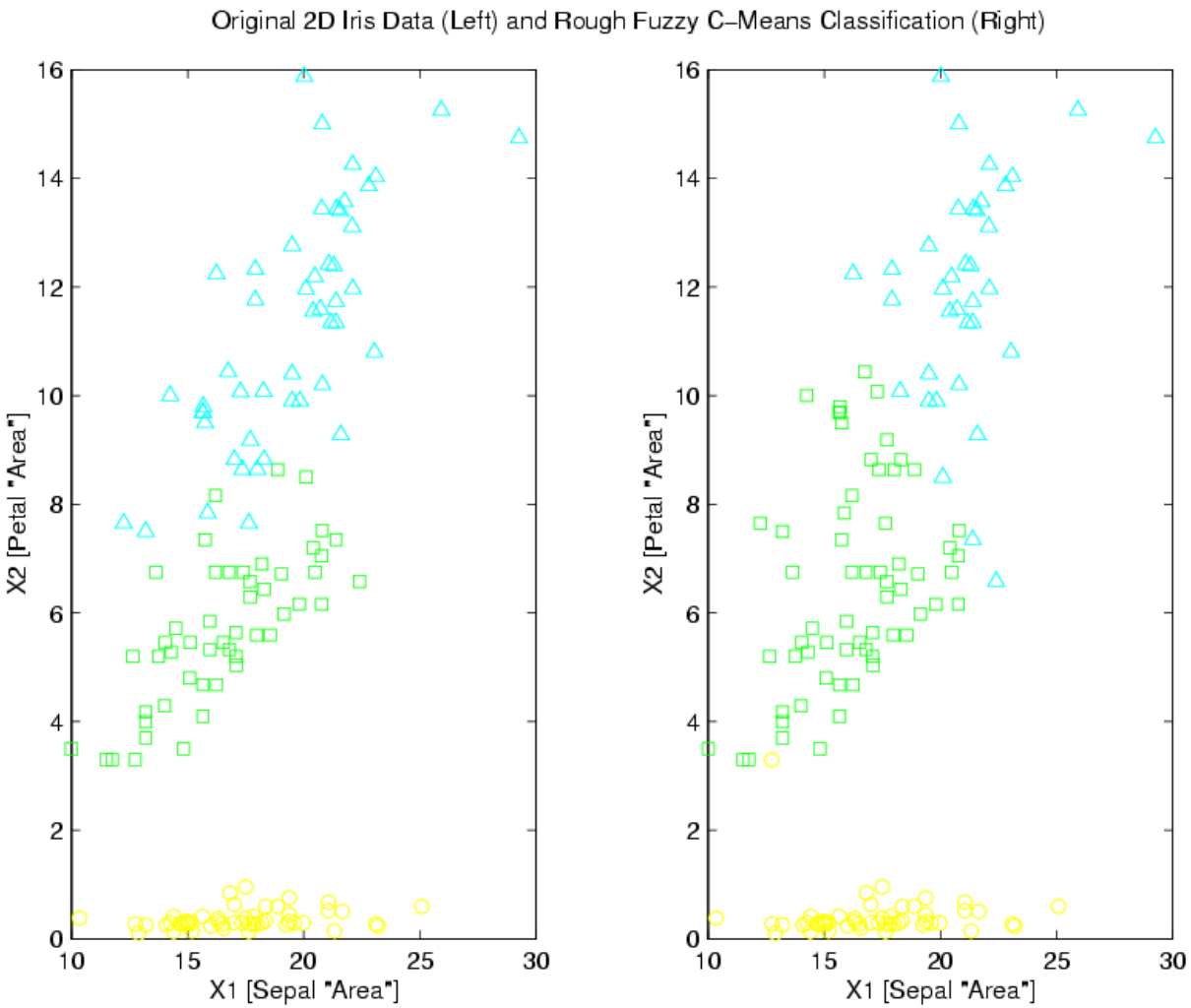
## Exemplo (“Iris” 2D):



# FCM


Fuzzy Clusters derived from the 2D Iris Data with the FCM algorithm (m=2,c=3)
Fij
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
10
20
30
40
X1 [Sepal "Area"]
0
2
4
6
8
10
12
14
16
X2 [Petal "Area"]

# FCM


Original 2D Iris Data (Left) and Rough Fuzzy C–Means Classification (Right)
X2 [Petal "Area"]
X1 [Sepal "Area"]
0
2
4
6
8
10
12
14
16
10
15
20
25
30

# Complexidade Computacional

- Complexidade de tempo do FCM é, em princípio, $\boldsymbol{O}(N\ n\ c^2)$
  - assumindo um no. constante de iterações fixado pelo usuário
- Porém...
  - é possível demonstrar que existe implementação linear !
  - complexidade $\boldsymbol{O}(N\ n\ c)$

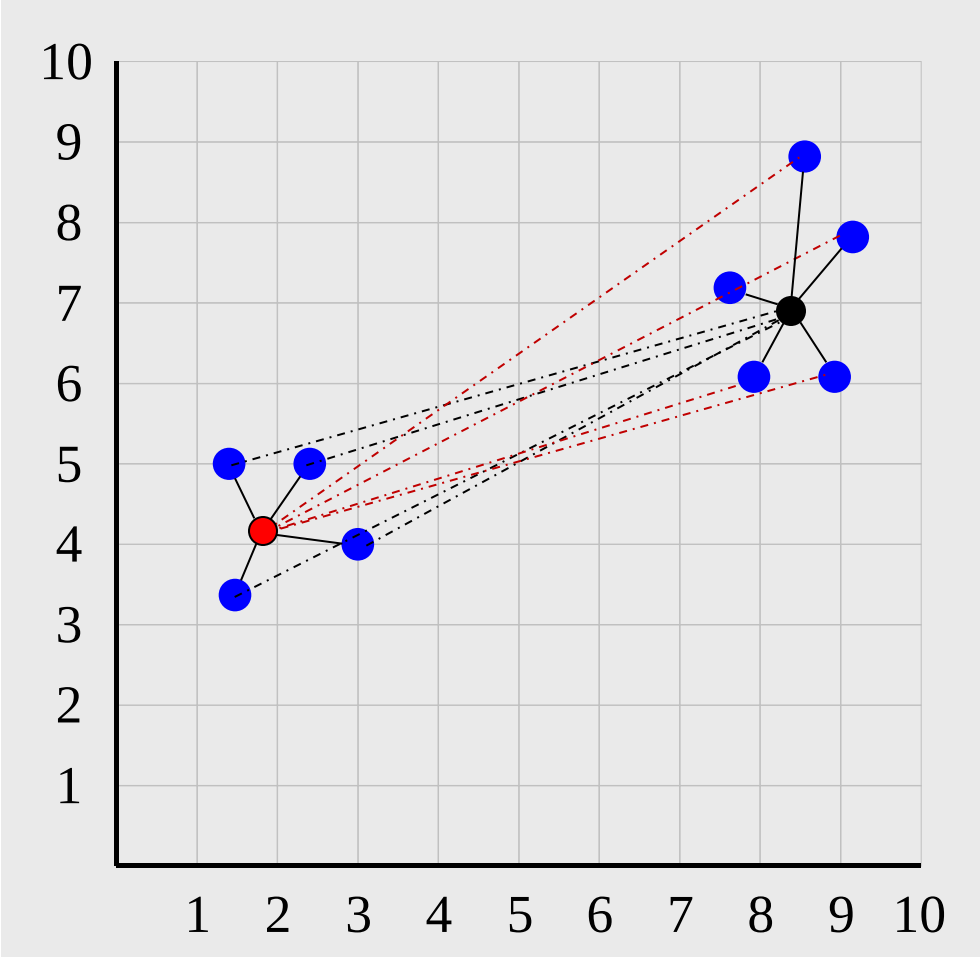
# Determinação do No. de Grupos...

- Múltiplas execuções ordenadas de FCM, com uso da função objetivo ***J***

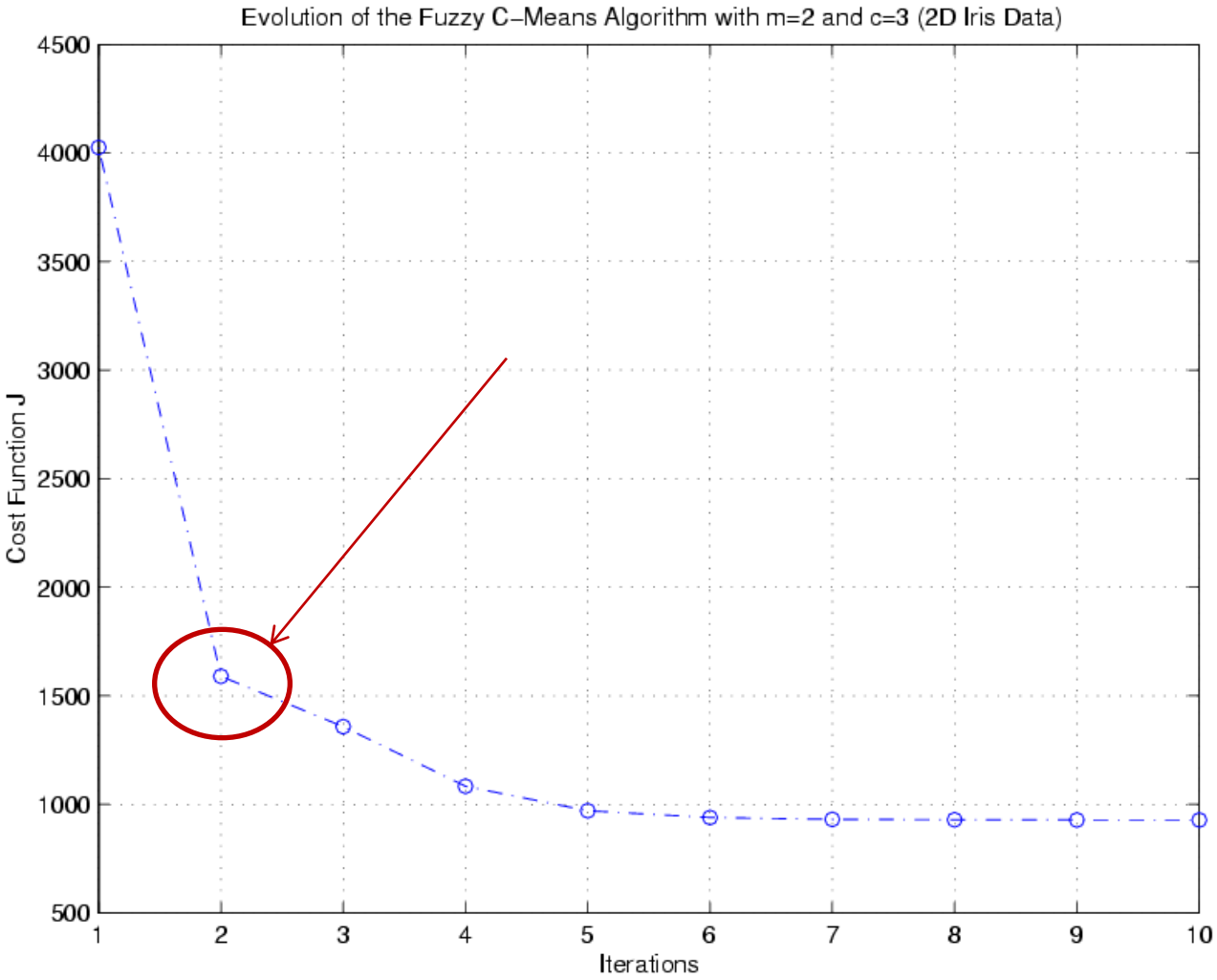
Erro Quadrático Ponderado:

$$J = \sum_{j=1}^{N} \sum_{i=1}^{c} \mu_{ij}^{m} d\left(\mathbf{x}_j, \mathbf{v}_i\right)^2$$

Função Objetivo

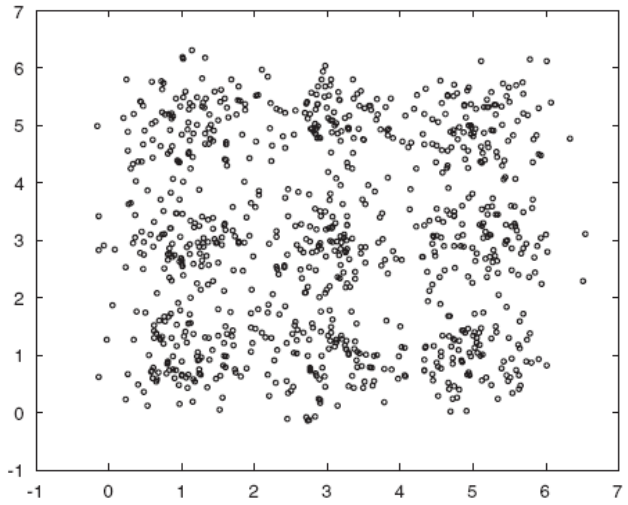


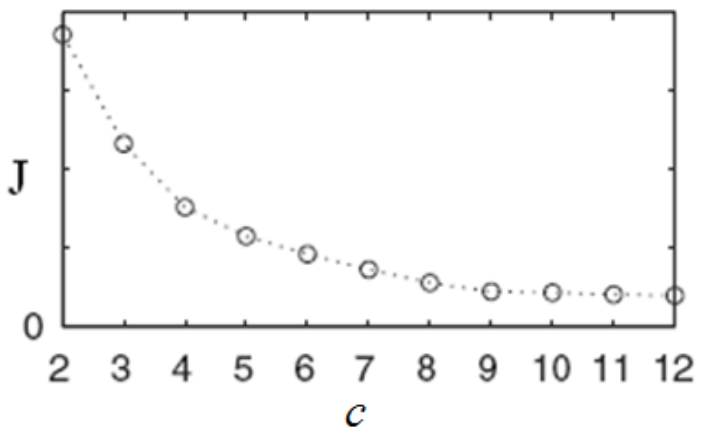
# Exemplo (IRIS 2D)



# Limitações

- Mesmas já vistas para o k-means:
  - 1. Casos Ambíguos:



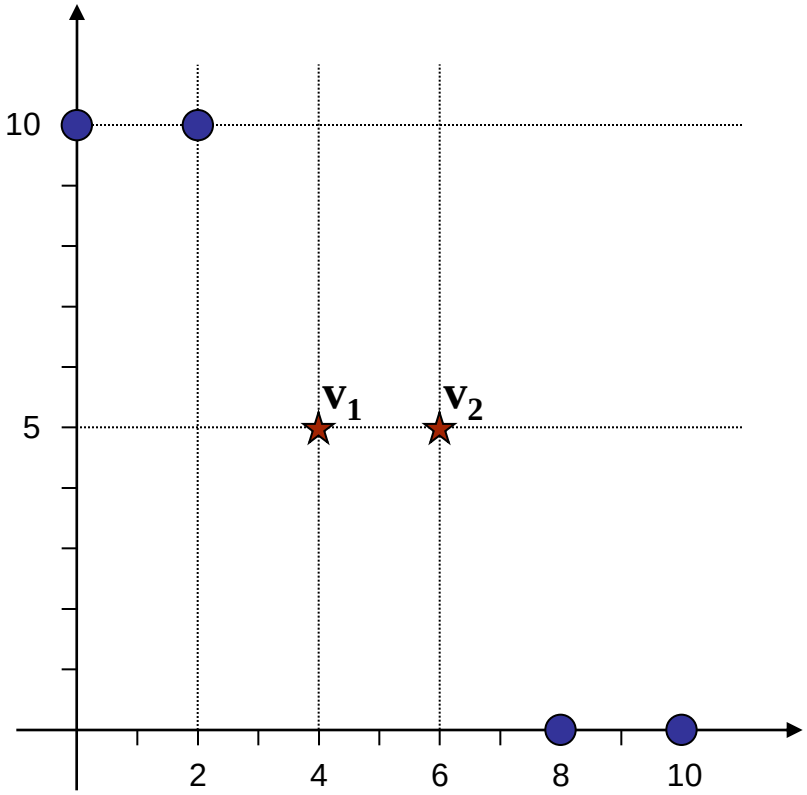




  - 2. Busca Guiada:
    - FCM evolutivo, ... ?

- Solução: vide aulas de **validação de agrupamento** !

# Exercício

- Agrupar os dados em azul na figura abaixo através do método FCM com 2 clusters, *m*=2 e centros iniciais assinalados em vermelho. Apresentar os centros dos grupos e a matriz de valores de pertinência para cada iteração.




Computacional ⇒ 5 iterações.

Manual ⇒ 1ª iteração.

# FCM Paralelo / Distribuído

- Discussão no quadro...

# Expectation Maximization (EM)

- O Algoritmo **EM** (**Expectation Maximization**) é um procedimento genérico para a modelagem **probabilística** de um conjunto de dados
- Basicamente, o algoritmo otimiza os parâmetros de uma função de distribuição de probabilidades de forma que esta represente os dados da forma mais verossímil possível
  - Maximização da Verossimilhança
- Modelo mais utilizado é aquele cuja função de distribuição de probabilidades é dada por uma **Mistura de Gaussianas**

# EM – Mistura de Gaussianas

- O modelo de mistura de Gaussianas é dado pela seguinte função de densidade de probabilidade $\boldsymbol{p}$:

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

  - $\mathbf{x}_j$ é um padrão (objeto)
  - $\mathcal{N}$ é uma Gaussiana (da mesma dimensão dos padrões)
    - $\mathbf{v}_i$ é o centro da i-ésima Gaussiana (vetor da mesma dimensão de $\mathbf{x}_j$)
    - $\boldsymbol{\Sigma}_i$ é a matriz de covariância da i-ésima Gaussiana
  - $\pi_i$ é uma probabilidade associada à i-ésima Gaussiana
  - k é o número de Gaussianas que compõem a mistura

# EM – Mistura de Gaussianas

- Para compreender a distribuição de probabilidade $p(\mathbf{x}_j)$, seja uma var. aleatória binária k-dimensional $\mathbf{z}_j$, tal que:
  - $\mathbf{z}_j = [z_{1j} \ldots z_{kj}]^T$ assume apenas valores em representação 1-de-k:
    - $z_{ij} = 1$ para um dado $i \in \{1, \ldots, k\}$; todos os demais são nulos
    - em outras palavras, $z_{lj} \in \{0, 1\}$ e $\sum_l z_{lj} = 1$
  - Define-se $\pi_i = p(z_{ij} = 1)$ como a **probabilidade a priori** que um padrão qualquer seja gerado pela i-ésima Gaussiana, $\mathcal{N}(\mathbf{x}_j \mid \mathbf{v}_i, \boldsymbol{\Sigma}_i)$
    - logo, $0 \leq \pi_i \leq 1$ e $\sum_i \pi_i = 1$
  - A distribuição de probabilidade $p(\mathbf{z}_j)$ é tal que:
    - $p(\mathbf{z}_j) = \pi_i$ para a realização de $\mathbf{z}_j$ tal que $z_{ij} = 1$

# EM – Mistura de Gaussianas

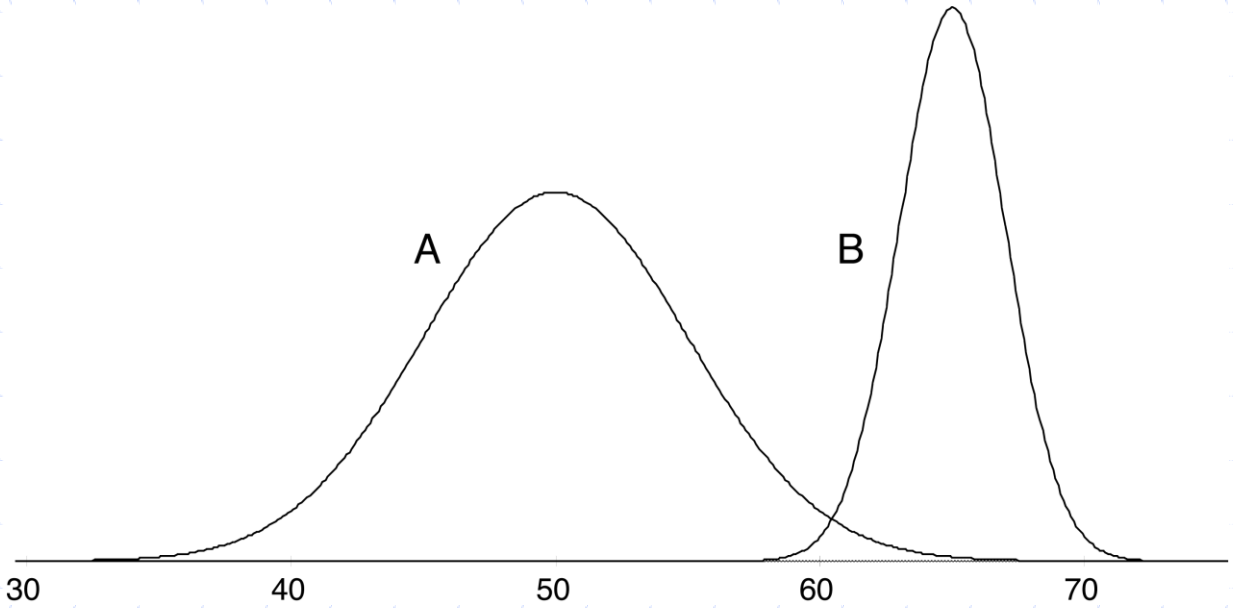
- Note que a i-ésima Gaussiana corresponde à distribuição condicional de $\mathbf{x}_j$ dado um valor particular de $\mathbf{z}_j$, i.e.:
  - $p(\mathbf{x}_j \mid z_{ij} = 1) = \mathcal{N}(\mathbf{x}_j \mid \mathbf{v}_i, \Sigma_i)$, ou (equivalentemente)
  - $p(\mathbf{x}_j \mid \mathbf{z}_j) = \mathcal{N}(\mathbf{x}_j \mid \mathbf{v}_i, \Sigma_i)$ para a realização de $\mathbf{z}_j$ tal que $z_{ij} = 1$
- Das distribuições $p(\mathbf{z}_j)$ e $p(\mathbf{x}_j \mid \mathbf{z}_j)$ tem-se a distribuição conjunta $p(\mathbf{x}_j \,\&\, \mathbf{z}_j)$, como segue:
  - $p(\mathbf{x}_j \,\&\, \mathbf{z}_j) = p(\mathbf{z}_j)\, p(\mathbf{x}_j \mid \mathbf{z}_j)$
- A distribuição $p(\mathbf{x}_j)$ é obtida então como:
  - $p(\mathbf{x}_j) = \sum_{\mathbf{z}_j} p(\mathbf{x}_j \,\&\, \mathbf{z}_j) = \sum_{i=1}^{k} \pi_i \mathcal{N}(\mathbf{x}_j \mid \mathbf{v}_i, \Sigma_i)$

# Modelo para 1 Atributo e 2 Grupos:

Objetos:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | | |

Modelo:




$\mu_A$=50, $\sigma_A$ =5, $\pi_A$=0.6 $\mu_B$=65, $\sigma_B$ =2, $\pi_B$=0.4

# EM – Mistura de Gaussianas

- Uma grandeza fundamental para definição do algoritmo:

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

  - $p(\mathbf{z}_j \mid \mathbf{x}_j) = p(\mathbf{x}_j \,\&\, \mathbf{z}_j) \,/\, p(\mathbf{x}_j) \quad \rightarrow \quad p(\mathbf{z}_j \mid \mathbf{x}_j) = p(\mathbf{z}_j)\, p(\mathbf{x}_j \mid \mathbf{z}_j) \,/\, p(\mathbf{x}_j) \quad \rightarrow$

    $\rightarrow \quad p(z_{ij} = 1 \mid \mathbf{x}_j) = p(z_{ij} = 1)\, p(\mathbf{x}_j \mid z_{ij} = 1) \,/\, p(\mathbf{x}_j)$

  - É a probabilidade **a posteriori** de $z_{ij} = 1$ dado que se observou $\mathbf{x}_j$
    - Em contraste com $\pi_i$, que é a probabilidade a priori
  - Em outras palavras, é a probabilidade a posteriori que a observação $\mathbf{x}_j$ tenha sido produzida pela i-ésima Gaussiana

# EM – Mistura de Gaussianas

- Outra grandeza fundamental para definição do algoritmo:
  - Dado um conjunto $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ de N observações independentes, tem-se a distribuição conjunta dada por:
    - $$p(\mathbf{X}) = p(\mathbf{x}_1 \,\&\, \mathbf{x}_2 \,\&\, \ldots \,\&\, \mathbf{x}_N) = \prod_{j=1}^{N} p(\mathbf{x}_j) = \prod_{j=1}^{N} \sum_{l=1}^{k} \pi_l \mathcal{N}(\mathbf{x}_j | \mathbf{v}_l, \boldsymbol{\Sigma}_l)$$
  - Essa distribuição governa o conjunto de observações **X** dado que este seja descrito pela mistura de Gaussianas em questão:
    - $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_k\}$, $\mathbf{v} = \{\mathbf{v}_1, ..., \mathbf{v}_k\}$ e $\boldsymbol{\pi} = \{\pi_1, ..., \pi_k\}$
    - Por esta razão, em geral refere-se a esta distribuição por $p(\mathbf{X} \mid \boldsymbol{\pi}, \boldsymbol{\Sigma}, \mathbf{v})$
  - Por conveniência matemática, aplica-se o logaritmo para eliminar o produtório, obtendo assim a **função de log-verossimilhança**:
    - $$\ln\left(p(\mathbf{X} \mid \boldsymbol{\pi}, \boldsymbol{\Sigma}, \mathbf{v})\right) = \sum_{j=1}^{N} \ln\left(\sum_{l=1}^{k} \pi_l \mathcal{N}(\mathbf{x}_j | \mathbf{v}_l, \boldsymbol{\Sigma}_l)\right)$$

# EM – Mistura de Gaussianas

- Maximizar a verossimilhança pode ser visto como maximizar a compatibilidade entre as N observações e o modelo
- EM (Dempster et al., 1977) é um algoritmo de otimização que visa maximizar a f. de (log) verossimilhança em 2 passos:
  - **Passo E** (Expectation)
    - Avalia as probabilidades a posteriori $\mu_{ij}$ (i = 1, ..., k; j = 1, ..., N)
      - a partir das N observações $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ e do modelo corrente, dado pelos parâmetros $\Sigma = \{\Sigma_1, ..., \Sigma_k\}$, $\mathbf{v} = \{\mathbf{v}_1, ..., \mathbf{v}_k\}$ e $\pi = \{\pi_1, ..., \pi_k\}$
  - **Passo M** (Maximization)
    - Ajusta o modelo visando maximizar a função de log-verossimilhança

# EM – Mistura de Gaussianas

- **Passo E** (Expectation):
  - Avalia as probabilidades a posteriori $\mu_{ij}$ (i = 1, ..., k; j = 1, ..., N)

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

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

# EM – Mistura de Gaussianas

- **Passo M** (Maximization):
  - Ajusta o modelo visando maximizar a verossimilhança

$$\begin{cases} \mathbf{v}_i = \dfrac{1}{N_i} \displaystyle\sum_{j=1}^{N} \mu_{ij} \mathbf{x}_j \\ \boldsymbol{\Sigma}_i = \dfrac{1}{N_i} \displaystyle\sum_{j=1}^{N} \mu_{ij} (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T \\ \pi_i = \dfrac{N_i}{N} \end{cases}$$

$\mathbf{v}_i = \dfrac{1}{N_i} \displaystyle\sum_{j=1}^{N} \mu_{ij} \mathbf{x}_j$ ➡ centróide ponderado

$\boldsymbol{\Sigma}_i = \dfrac{1}{N_i} \displaystyle\sum_{j=1}^{N} \mu_{ij} (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T$ ➡ covariância ponderada

$\pi_i = \dfrac{N_i}{N}$ ➡ responsabilidade relativa do i-ésimo grupo (Gaussiana)

$N_i = \displaystyle\sum_{j=1}^{N} \mu_{ij}$ ➡ responsabilidade absoluta do i-ésimo grupo (Gaussiana)

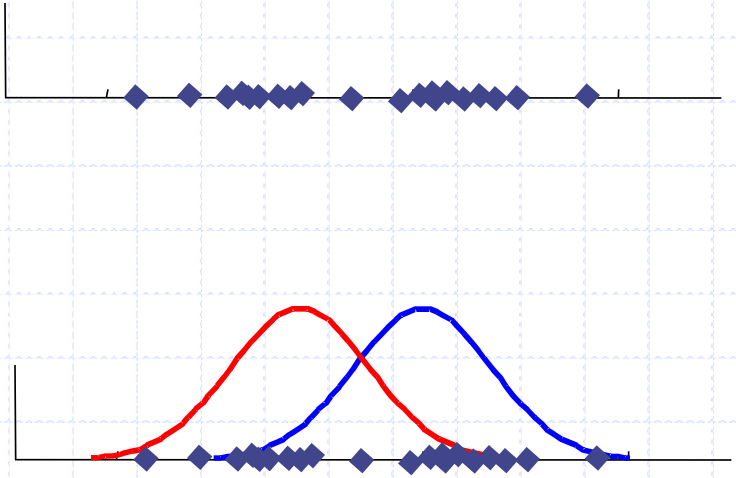
# EM – Mistura de Gaussianas

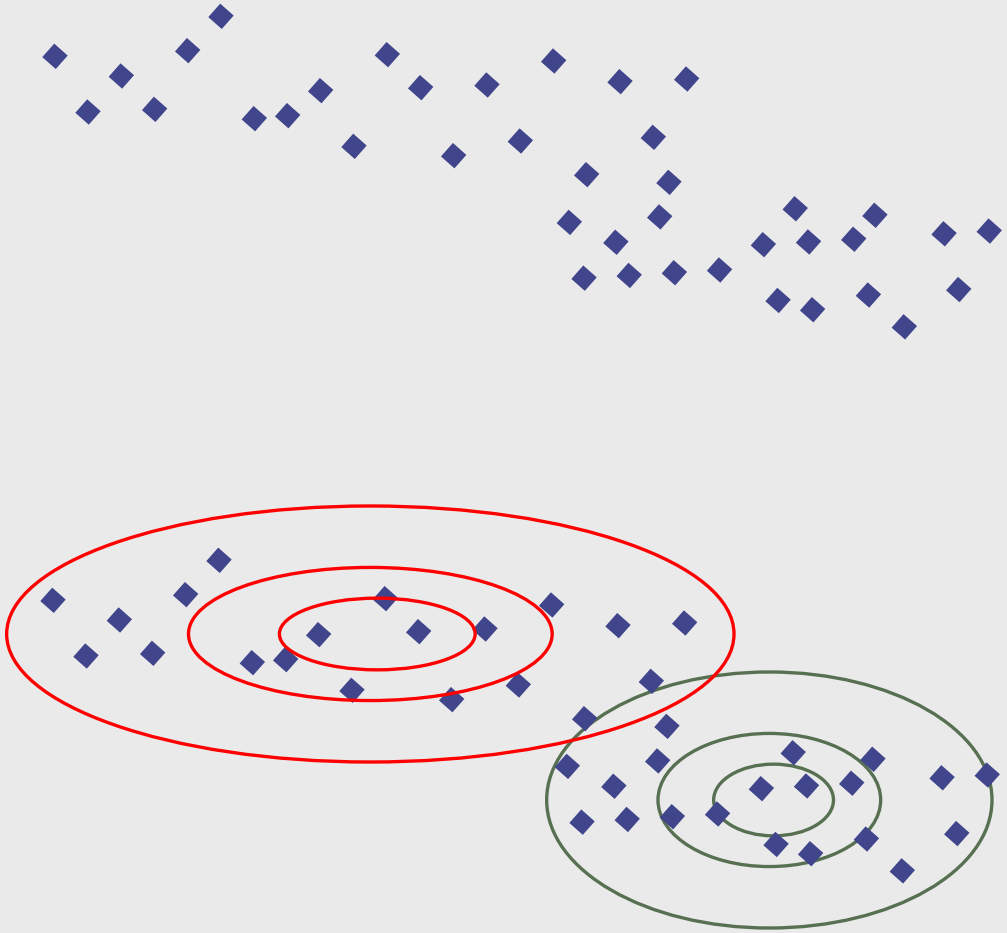
- **Algoritmo**:
  1. **Inicialização** (via k-means)
     - protótipos $\mathbf{v}_i$ = centróides finais do k-means
     - covariâncias $\sum_i$ = matrizes de covariância finais (dos grupos)
     - probabilidades $\mu_{ij}$ (para $N_i$ e $\pi_i$) = matriz de partição rígida final
  2. **Passo E**
  3. **Passo M**
  4. **Avaliação do Critério de Parada**
     - e.g. função de log-verossimilhança
  5. **Interrupção ou Retorno ao Passo 2**

# EM × k-Means

- EM produz informação muito mais rica sobre os dados
  - Probabilidades associadas a cada padrão / cluster
  - Outliers...
- É capaz de representar clusters alongados, elipsoidais, com atributos correlacionados
- No entanto, as vantagens acima vêm com um elevado custo computacional associado...
  - Cálculo das Normais Multi-Dimensionais $\mathcal{N}$ demanda as inversas das matrizes de covariância $\sum_i$, que é $O(n^3)$
  - Existem variantes e simplificações mais robustas e/ou computacionalmente mais eficientes (e.g., vide MCLUST)
- k-means é um caso particular de EM. Ambos estão sujeitos a mínimos locais

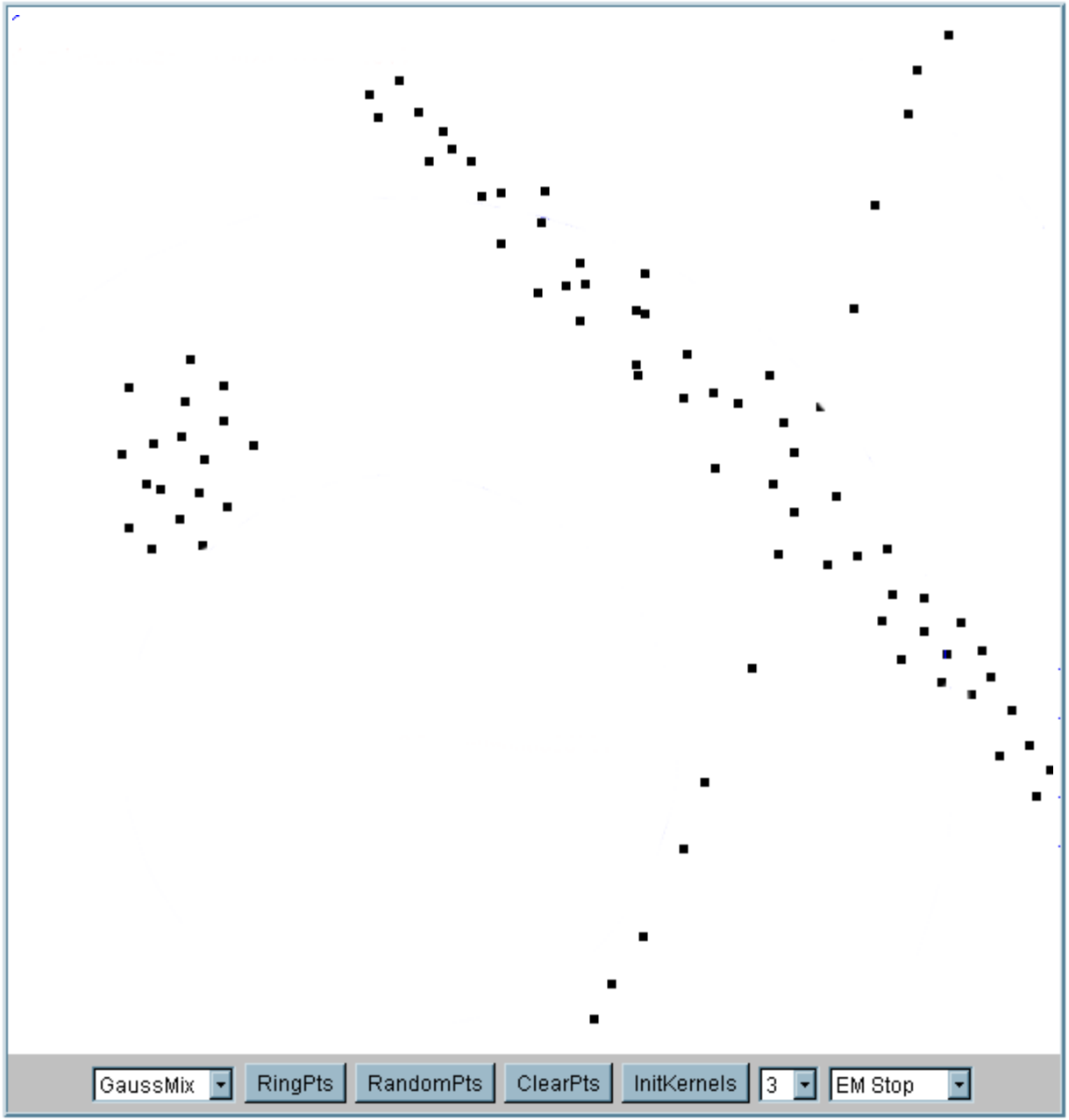
# **EM** (Mistura de Gaussianas – Exemplos)



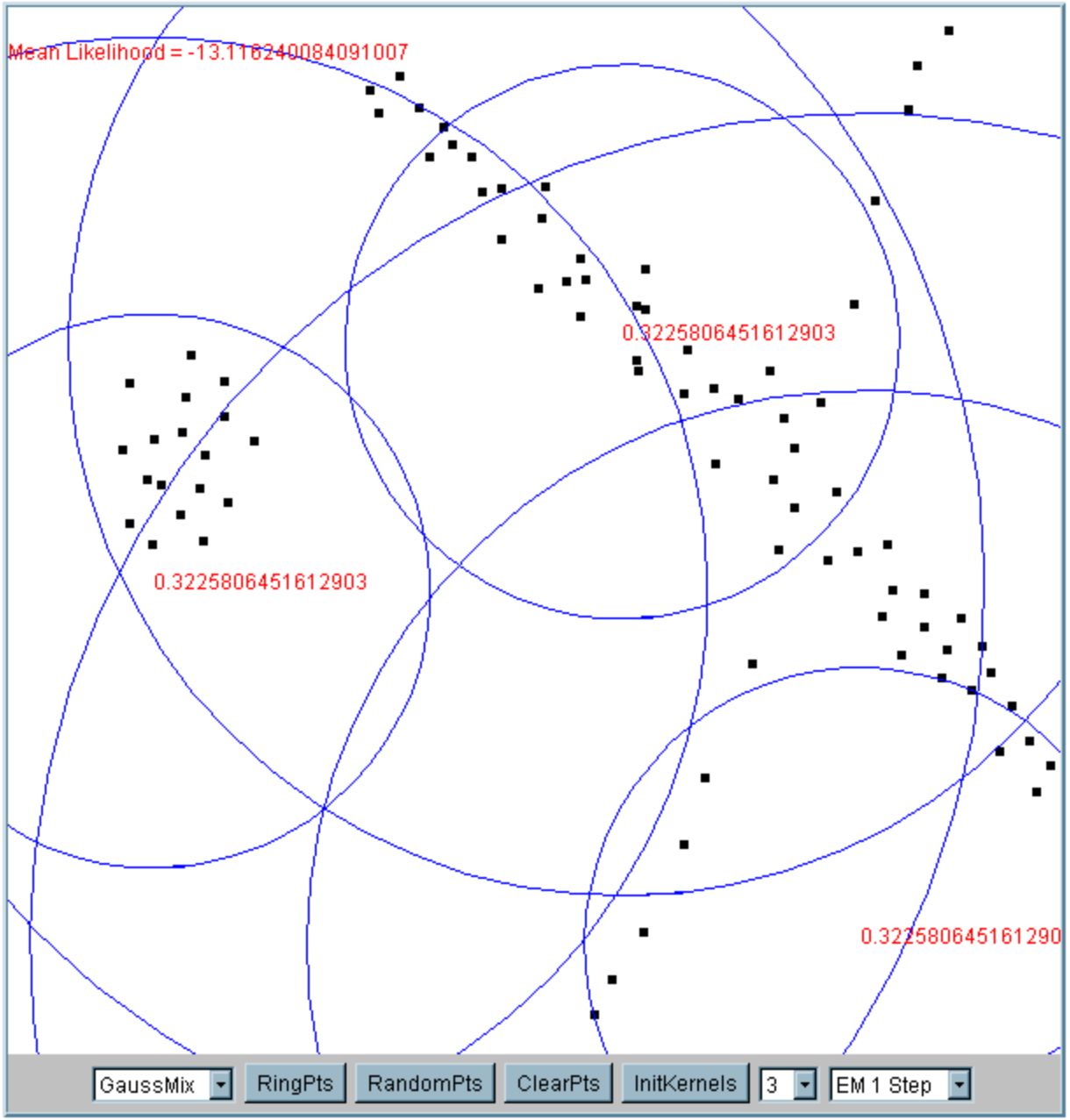


Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Manaus.

# Exemplo
(passo-a-passo)

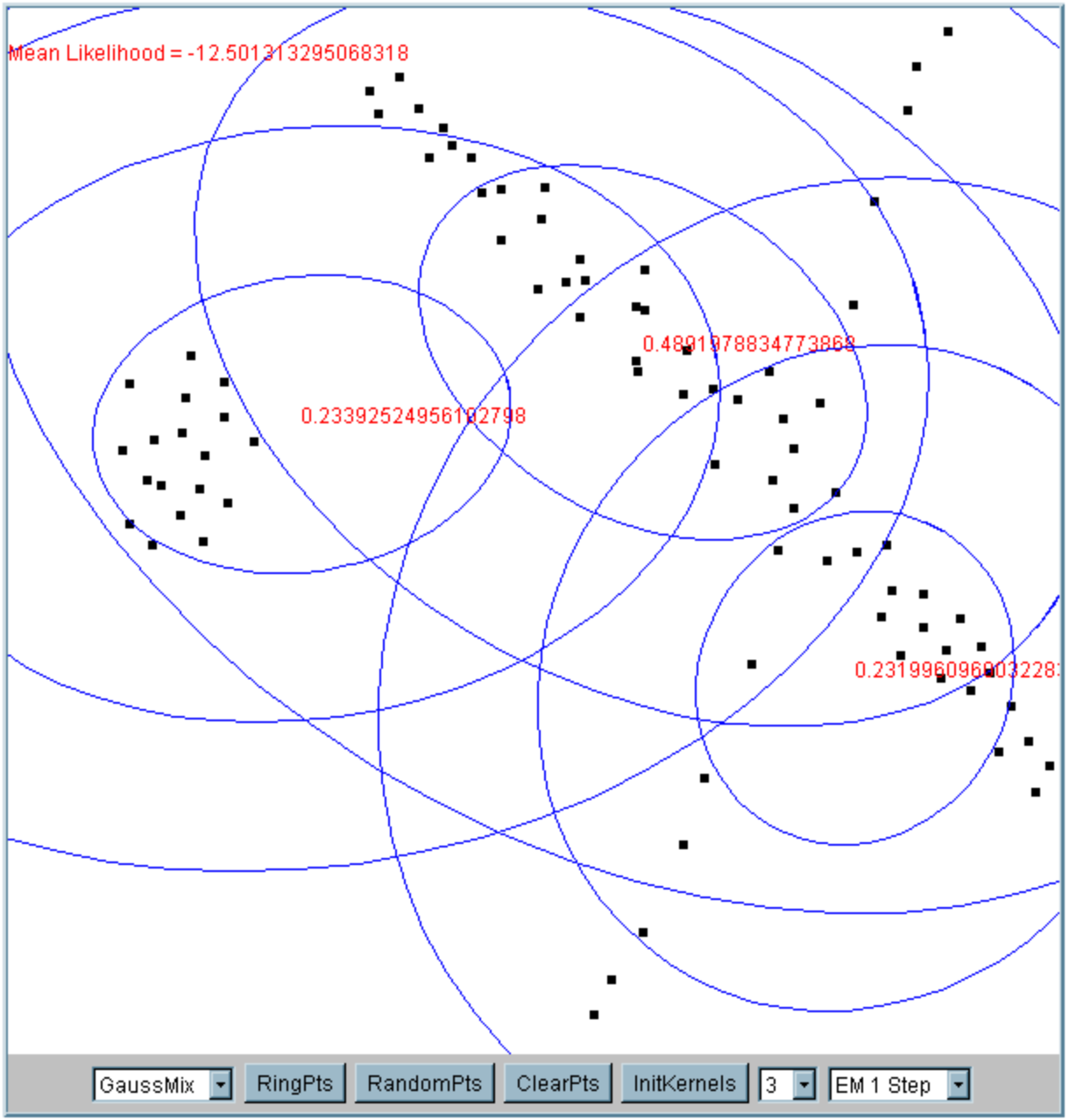


# Iteração 1

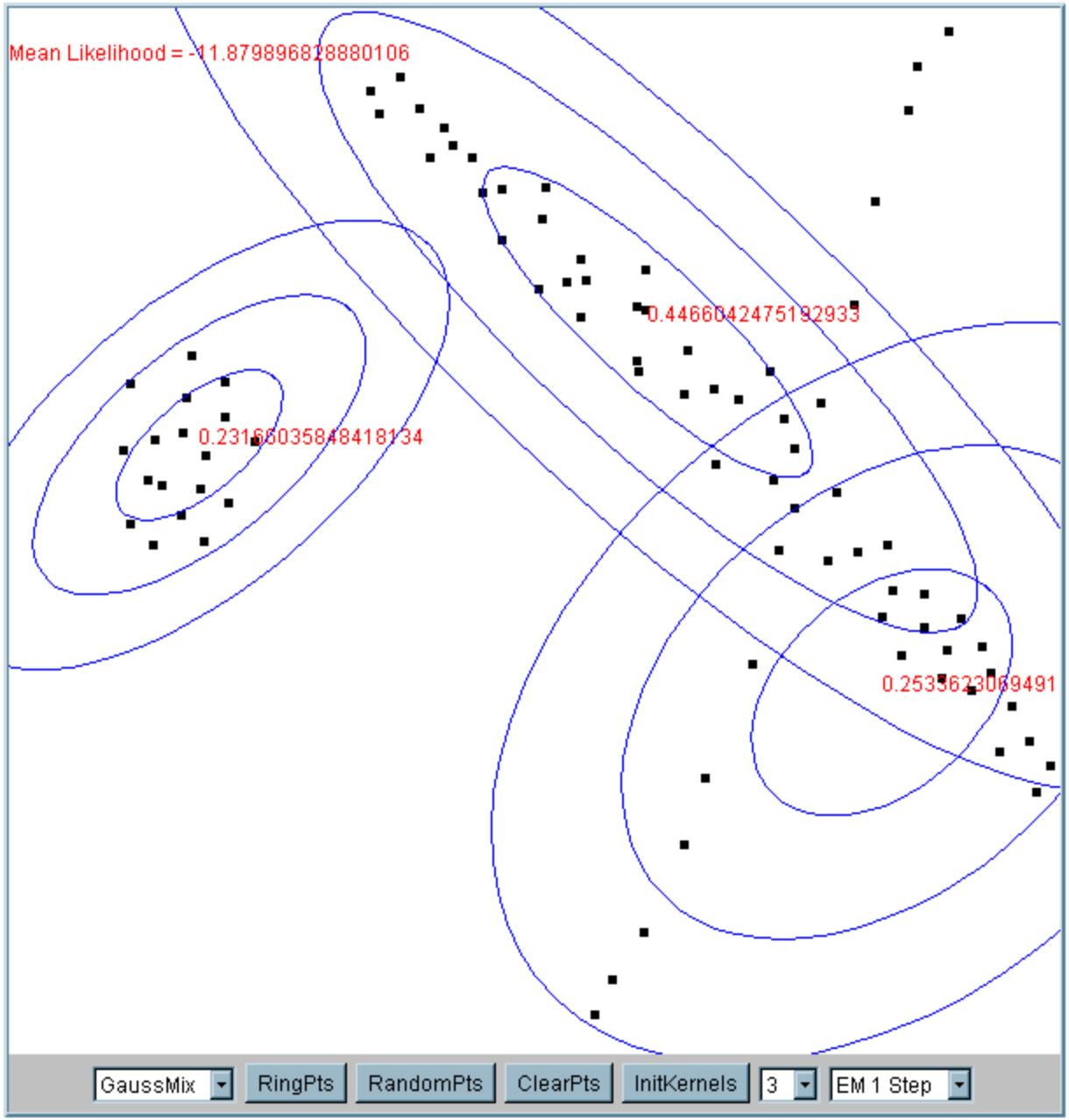



Keogh, E. A Gentle Introduction to Machine Learning and Data Mining for the Database Community, SBBD 2003, Manaus.

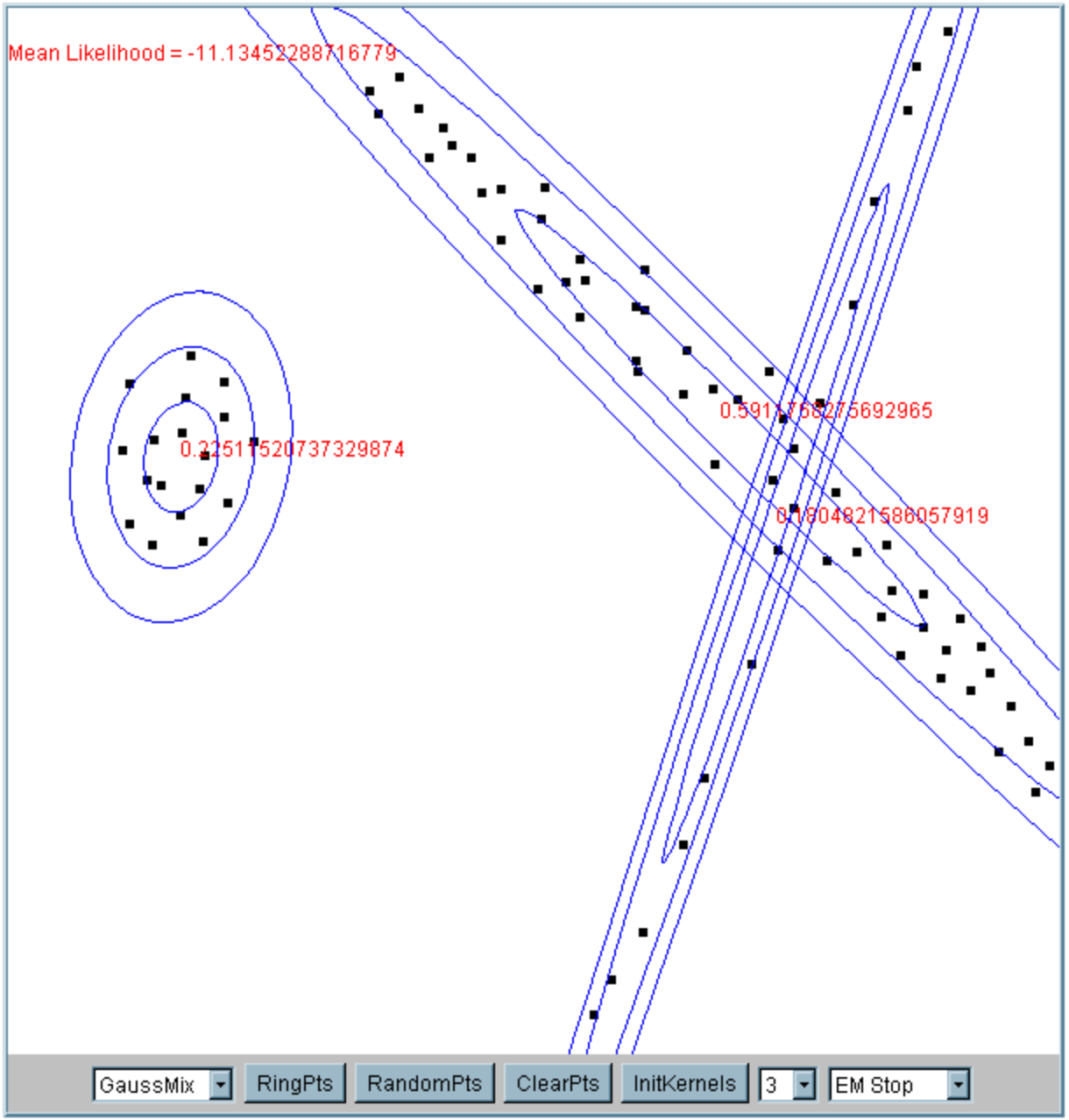
# Iteração 2


Mean Likelihood = -12.501313295068318
0.48918788347738
0.233925249561027
0.231996096803228
GaussMix
RingPts
RandomPts
ClearPts
InitKernels
3
EM 1 Step

# Iteração 5


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

# Iteração 25


Mean Likelihood = -11.13452288716779
0.22517520737329874
0.5911768275692965
0.18048215880057919
GaussMix
RingPts
RandomPts
ClearPts
InitKernels
3
EM Stop

# Exercício

| Objeto | $x_1$ |
|---|---|
| 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 |

- Execute manualmente iterações do EM na base de dados ao lado (n = 1, N = 10), com k = 2. Tome protótipos iniciais arbitrários e os demais parâmetros inicializados a partir destes, de maneira análoga à inicialização via k-means

- Ilustre o resultado obtido de forma gráfica

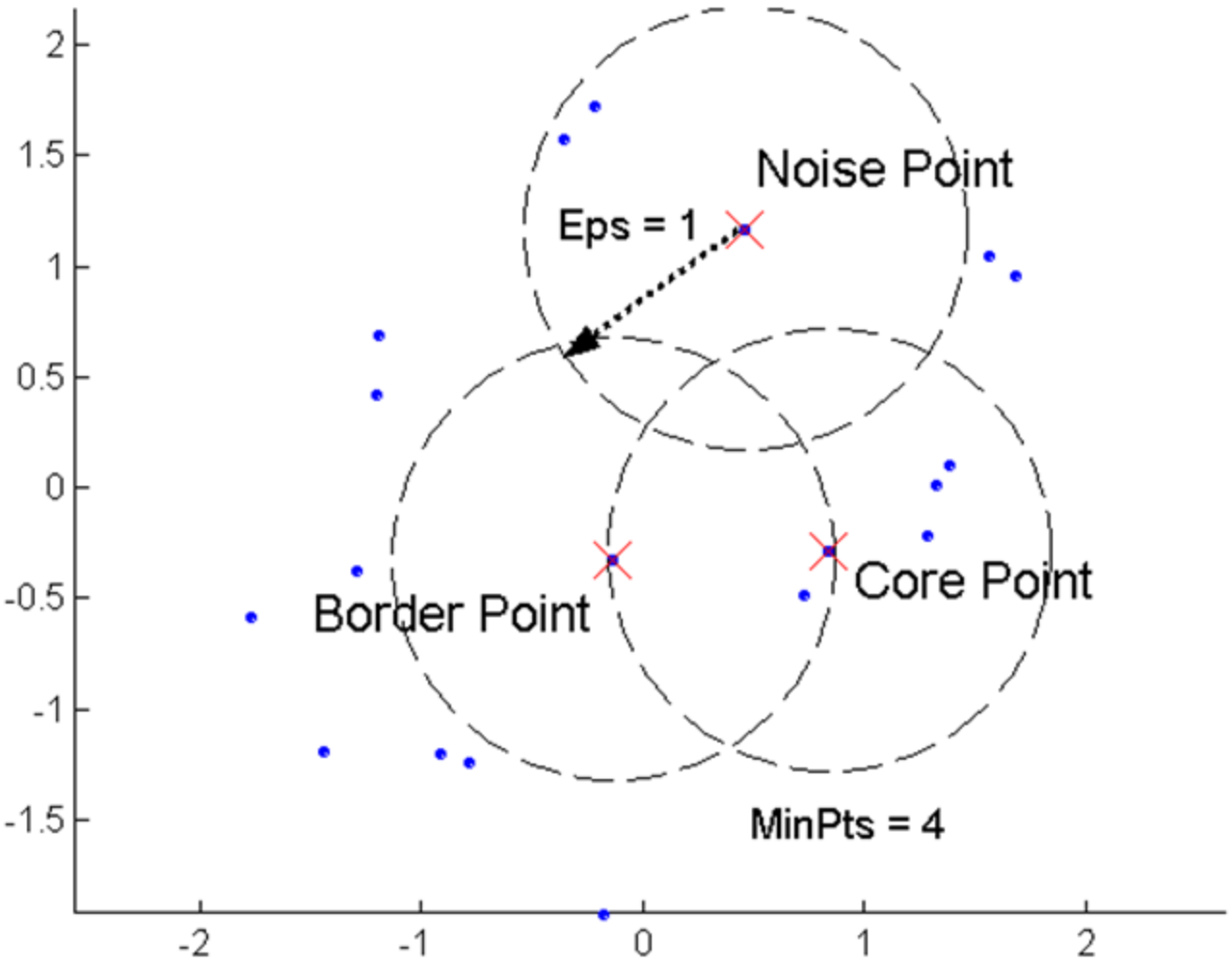
# Algoritmos Baseados em Densidade

- **Paradigma de Agrupamento por Densidade**
  - Clusters como regiões de alta concentração de objetos
    - Regiões essas separadas por regiões de baixa concentração de objetos
  - Paradigma alternativo àquele baseado em protótipos
    - K-means e variantes, FCM, EM, etc
- Existem vários algoritmos
- Veremos a seguir um dos mais conhecidos, o **DBSCAN**

# DBSCAN

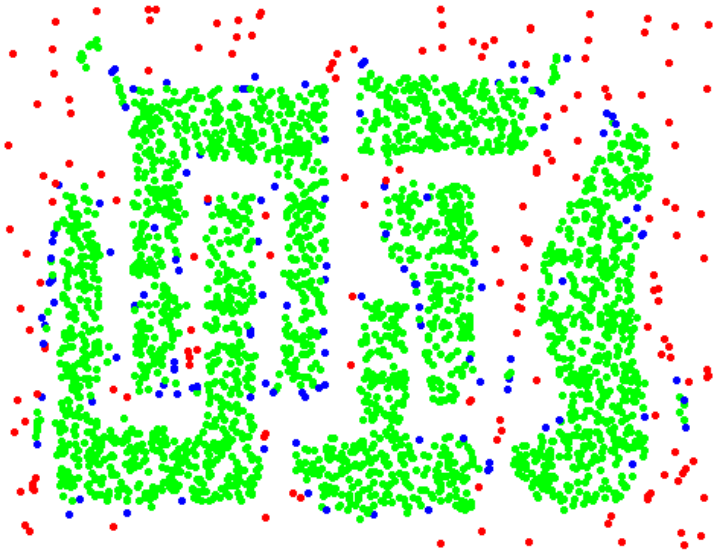
- DBSCAN is a density-based algorithm
  - It uses the concept of **Center-Based Density**
    - number of points within a specified radius (**Eps**)
- Definitions:
  - A point is a **core point** if it has at least a specified number of points (**MinPts**) within the radius Eps (including the point itself)
    - these are points that are in the interior of a cluster
  - A **border point** has fewer than MinPts within Eps, but is in the neighborhood (within the radius) of at least 1 core point
  - A **noise point** is neither a core point nor a border point

# DBSCAN: Core, Border, and Noise Points



# DBSCAN: Core, Border and Noise Points

Original Points



Point types: core, border and noise

Eps = 10, MinPts = 4

# DBSCAN

- **Algoritmo Conceitual:**

1. Percorra a BD e rotule os objetos como core, border ou noise
2. Elimine aqueles objetos rotulados como **noise**
3. Insira uma aresta entre cada par de objetos **core** vizinhos
   - 2 objetos são vizinhos se um estiver dentro do raio Eps do outro
4. Faça cada componente conexo resultante ser um cluster
5. Atribua cada **border** ao cluster de um de seus core associados
   - Resolva empates se houver objetos core associados de diferentes clusters

# DBSCAN

- **Exercício:**
  - Aplique o algoritmo DBSCAN na BD abaixo, com Eps = 5 e MinPts = 3 (que inclui o objeto em questão), indicando os rótulos de cada objeto (core, border ou noise) e os grupos

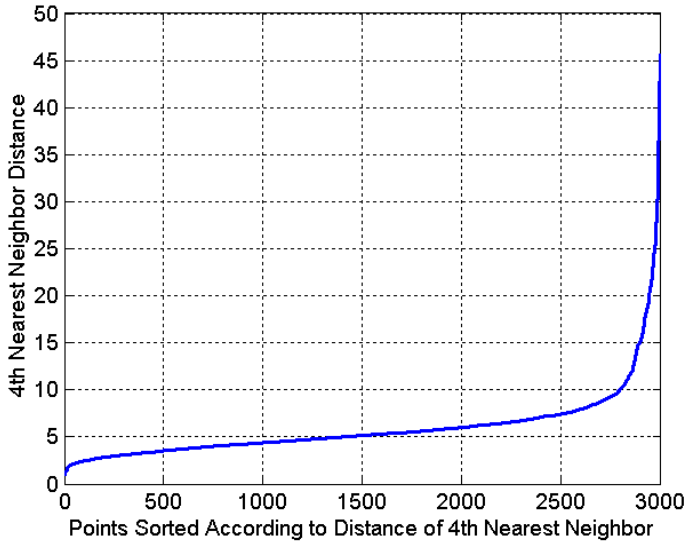
$$\mathbf{D} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{bmatrix} 0 & & & & & & \\ 1 & 0 & & & & & \\ 1 & 2 & 0 & & & & \\ 10 & 9 & 11 & 0 & & & \\ 11 & 10 & 12 & 2 & 0 & & \\ 21 & 20 & 22 & 18 & 19 & 0 & \\ 14 & 13 & 15 & 6 & 4 & 25 & 0 \end{bmatrix}$$

# DBSCAN

- **Complexidade Computacional:**
  - Na prática, a rotulação dos clusters pode ser realizada simultaneamente à rotulação dos objetos (passagem 1)
  - Logo, o tempo é O(N $\times$ tempo p/ encontrar os objetos vizinhos)
    - Em geral, isso significa $O(N^2)$
    - Para bases de dados com poucas dimensões é possível obter tempo O(N log N) usando estruturas de dados apropriadas
      - Árvores kd $\rightarrow$ **seminários**
  - Em termos de memória, só é preciso armazenar as rotulações de cada objeto. Logo, tem-se O(N)

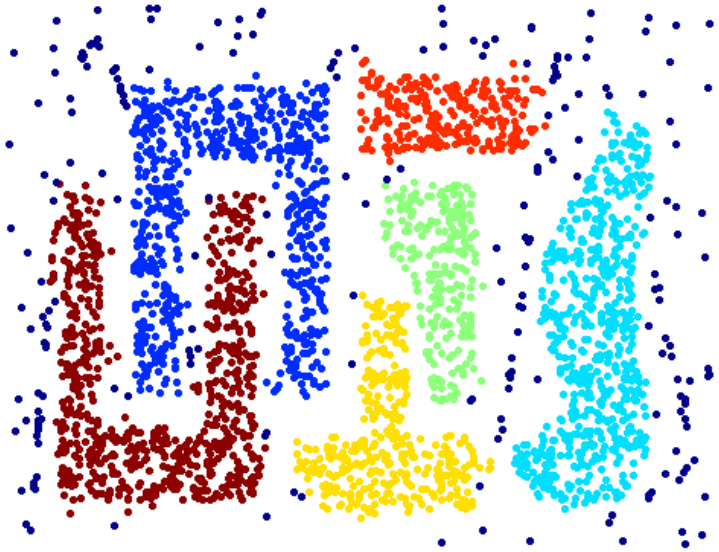
# (Attempt at) Determining EPS and MinPts

- Idea is that for points in a cluster, their $k^{th}$ nearest neighbors (NN) are at roughly the same distance
- Noise points have the $k^{th}$ NN at farther distance
- **Heuristic**:
  - plot sorted distance of every point to its $k^{th}$ nearest neighbor



# When DBSCAN Works Well

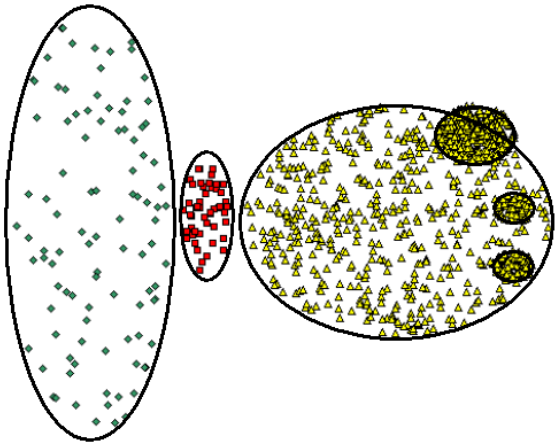
Original Points



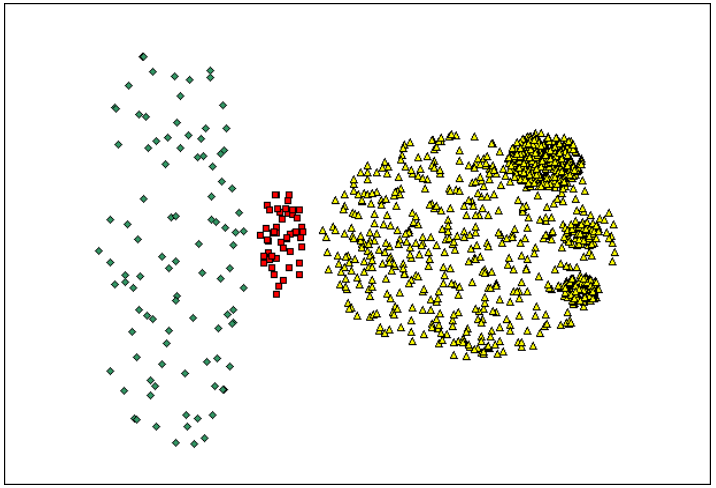
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

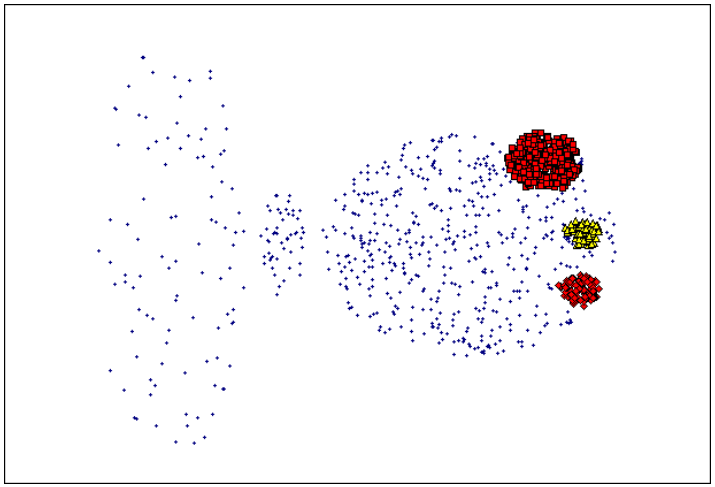
# When DBSCAN Does NOT Work Well



**Original Points**



(MinPts=4, Eps=9.75).

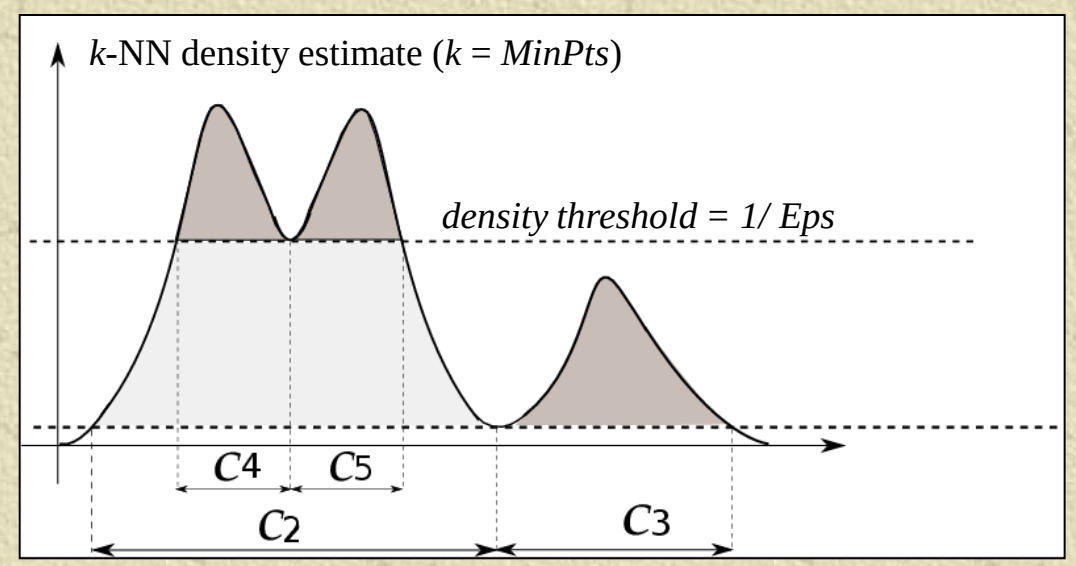


(MinPts=4, Eps=9.92)

- **Varying densities**
- **High-dimensional data**

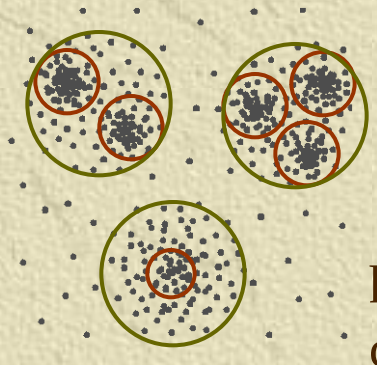
# Note (DBSCAN from a Density Estimate Perspective)

- **DBSCAN *without border* objects (DBSCAN*)**
  - Connected components of *dense* (*core*) objects
  - Density threshold *Eps* is critical and difficult to set
  - A single, global threshold that separates all clusters may not exist



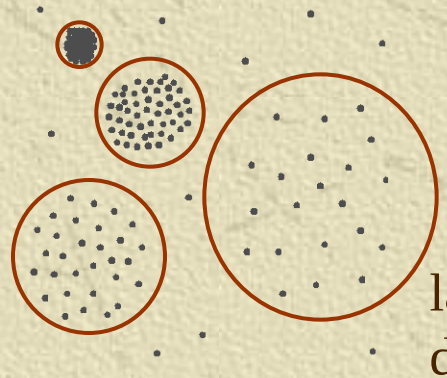
# Note (From Flat Solutions to Hierarchies)

- Situations in which (a global) threshold *Eps* is particularly critical:



hierarchical cluster structure

and/or

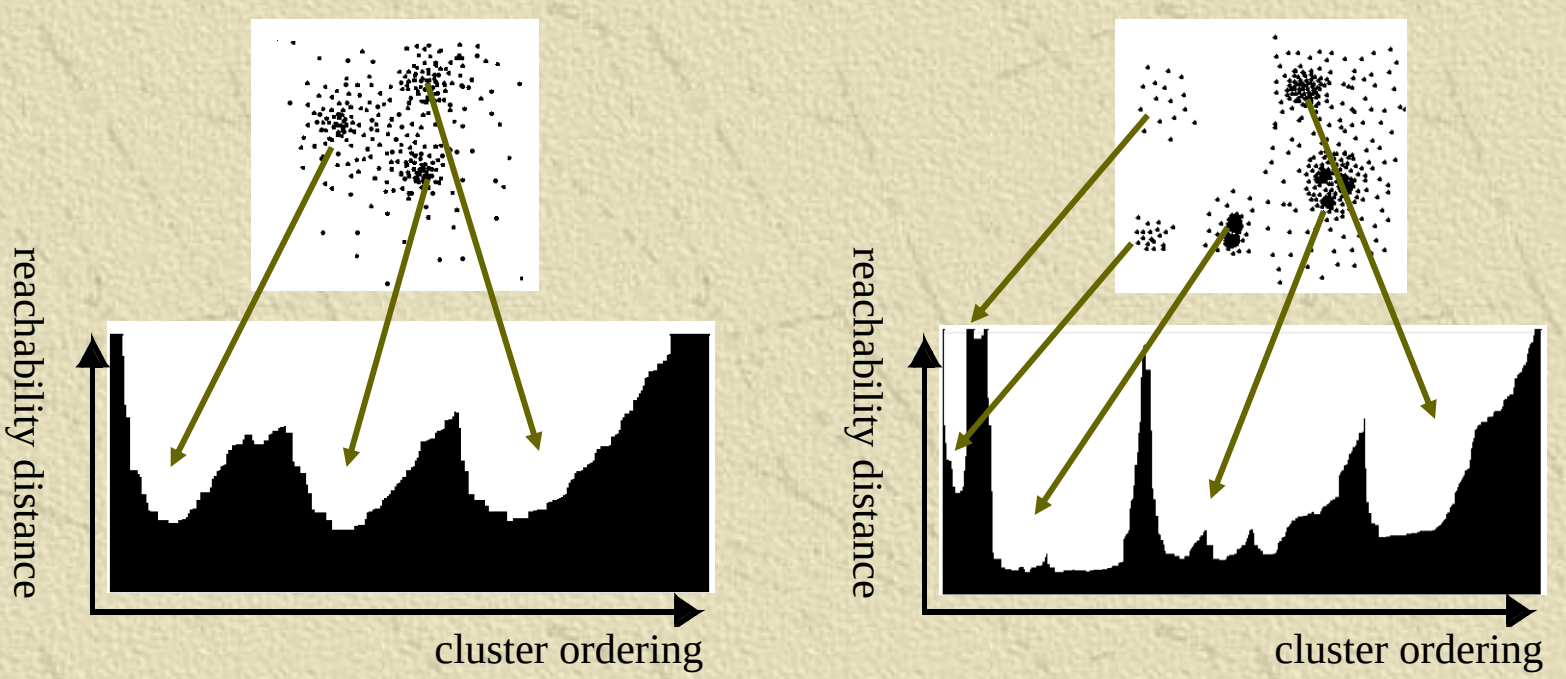


largely differing densities and sizes

- Need a hierarchical clustering algorithm in these situations

# Note (Hierarchical Density-Based Clustering)

- **OPTICS (1999)**
  - Ordering of points → *reachability plot*
  - *Encodes* DBSCAN clusterings for *all* density thresholds *Eps* ∈ [0,∞)
  - Single, non-critical parameter *MinPts* (smoothing factor)



# Note (Hierarchical DBSCAN* – HDBSCAN*)

- *R. J. G. B. Campello, D. Moulavi, J. Sander, "Density-Based Clustering Based on Hierarchical Density Estimates", PAKDD, 160-172, 2013*
- **Improvement over OPTICS**
- With a *single*, *non-critical parameter MinPts*, it provides:
  - **A complete density-based clustering hierarchy**
    - all unique DBSCAN* clusterings for $Eps \in [0,\infty)$
    - from which a reachability plot can be easily extracted (if desired)
  - **A simplified cluster tree of significant clusters**
  - **A flat clustering from <u>local</u> cuts through the hierarchy (optional)**
  - **Other visualizations and degrees of *outlierness* (ongoing work)**

# Referências


- 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