

# Clustering

CC 226 – Análise de padrões  
Carlos Henrique Q. Forster

# CLUSTERING

## ❖ Basic Concepts

In clustering or unsupervised learning no training data, with class labeling, are available. The goal becomes: **Group the data into a number of sensible clusters (groups)**. This unravels similarities and differences among the available data.

### ➤ Applications:

- Engineering
- Bioinformatics
- Social Sciences
- Medicine
- Data and Web Mining

- To perform clustering of a data set, a **clustering criterion** must first be adopted. Different clustering criteria lead, in general, to different clusters.

Blue shark,  
sheep, cat,  
dog

Lizard, sparrow,  
viper, seagull, gold  
fish, frog, red  
mullet

1. Two clusters
2. Clustering criterion:  
**How mammals bear  
their progeny**

Gold fish, red  
mullet, blue  
shark

Sheep, sparrow,  
dog, cat, seagull,  
lizard, frog, viper

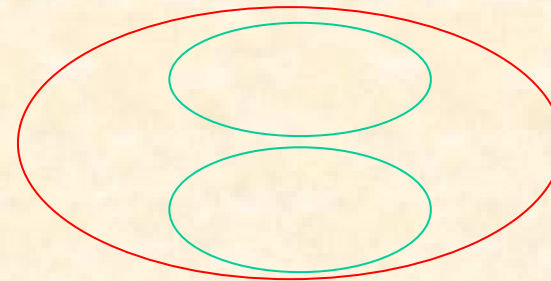
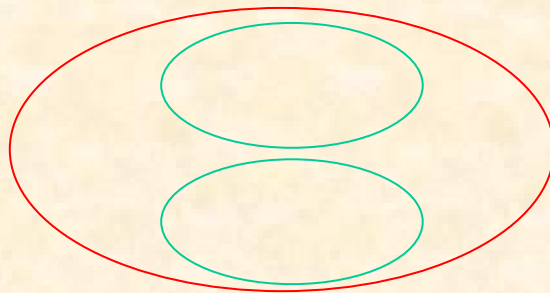
1. Two clusters
2. Clustering criterion:  
**Existence of lungs**

## ❖ Clustering task stages

- Feature Selection: Information rich features-**Parsimony**
- Proximity Measure: This quantifies the term **similar or dissimilar**.
- Clustering Criterion: This consists of a cost function or some type of rules.
- Clustering Algorithm: This consists of the set of **steps** followed to reveal the structure, based on the **similarity measure** and the adopted **criterion**.
- Validation of the results.
- Interpretation of the results.

- Depending on the similarity measure, the clustering criterion and the clustering algorithm different clusters may result. **Subjectivity** is a reality to live with from now on.

- A simple example: How many clusters??



**2 or 4 ??**

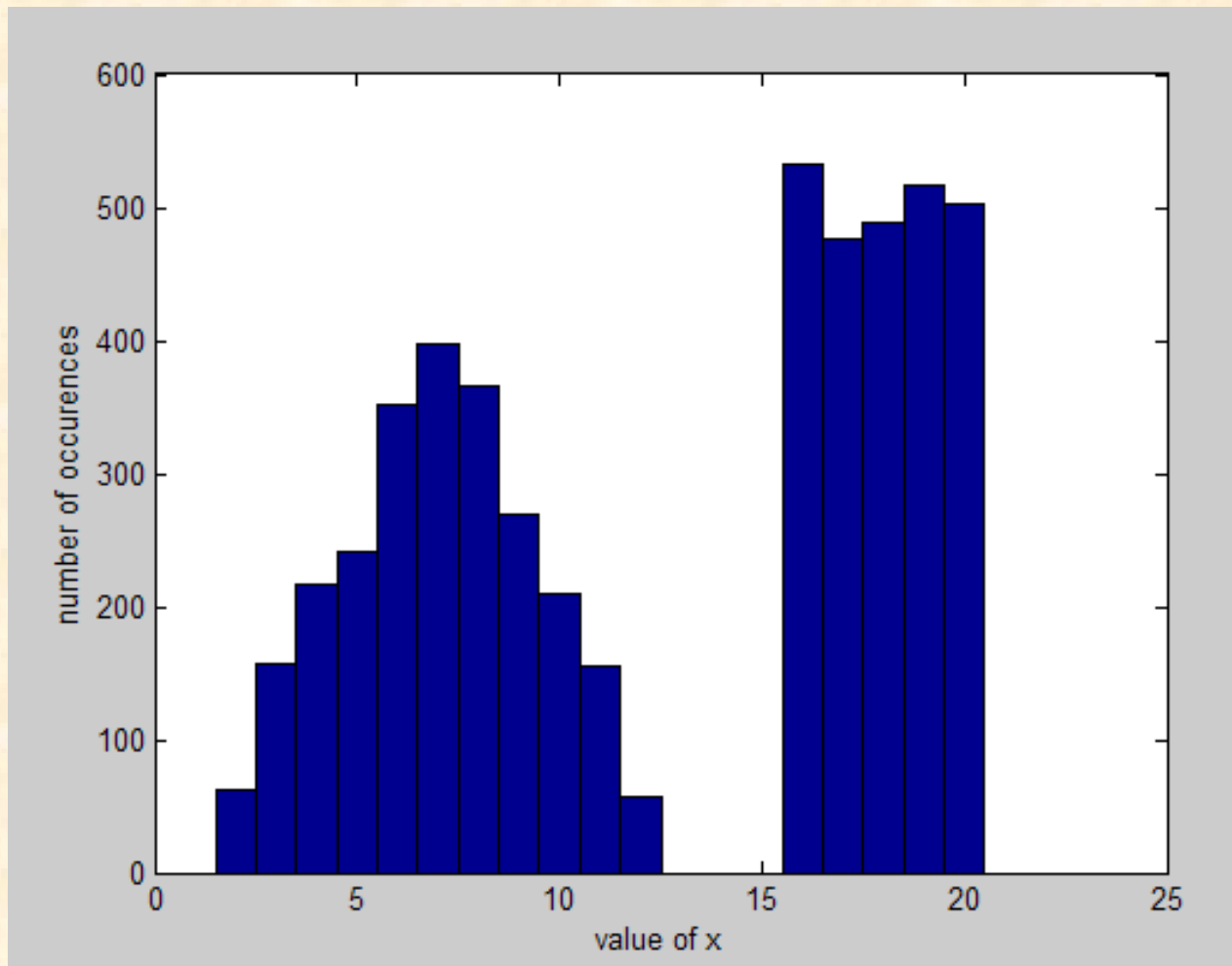
# Modelos de Misturas

- ❖ Distribuições que representam um fenômeno aleatório composto, cujo modelo corresponde a diferentes comportamentos com determinadas probabilidades.
- ❖ Cada distribuição que faz parte da mistura é uma distribuição componente.
- ❖ Cada distribuição componente tem uma probabilidade de ocorrência
  - Exemplo:
    - Jogo uma moeda,
    - Se der cara, puxo um carta de valor 16 a 20 de probabilidades idênticas,
    - Se der coroa, jogo dois dados de valor 1 a 6 igualmente prováveis e somo os resultados
  - Neste caso:

$$P = \alpha_1 P_1 + \alpha_2 P_2$$

A soma dos  $\alpha$  deve ser igual a 1!!!

## Resultados do "Jogo"



# Misturas

- ❖ Dada uma amostra e  $K$ , o número de componentes, o aprendizado corresponde a estimar as densidades de componentes e suas proporções (estimação semi-paramétrica).
- ❖ Por exemplo, se a distribuição de componentes for formada por normais multidimensionais, basta estimar os parâmetros  $\mu, \Sigma$  para cada componente, além das proporções da mistura.

$$p(x) = \sum_{i=1}^K p(x | G_i) \cdot P(G_i)$$

Componentes da mistura

Densidade de componentes

Proporções da mistura



# Aprendizado não-supervisionado

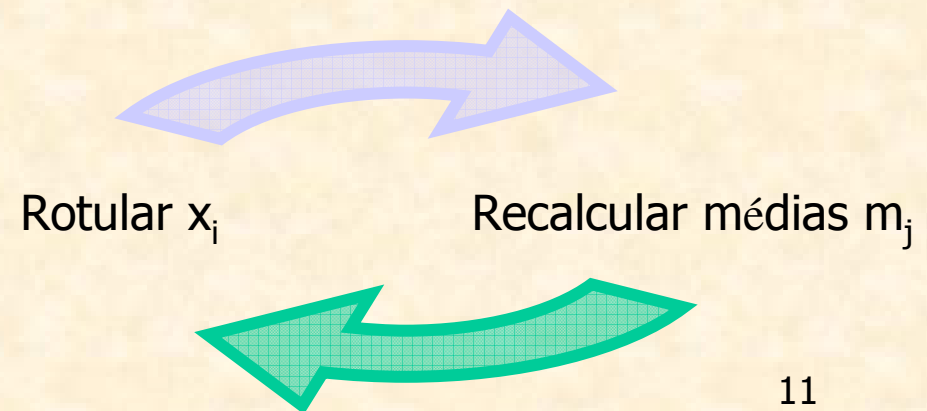
- ❖ No caso não-supervisionado, não sabemos a qual classe pertence cada observação da amostra.
- ❖ Queremos, na verdade, encontrar uma estrutura implícita na amostra.
- ❖ Para cada observação da amostra, precisamos, simultaneamente:
  - Rotular a qual classe pertence cada observação e
  - Estimar os parâmetros da distribuição de cada distribuição componente.

## Nossas hipóteses...

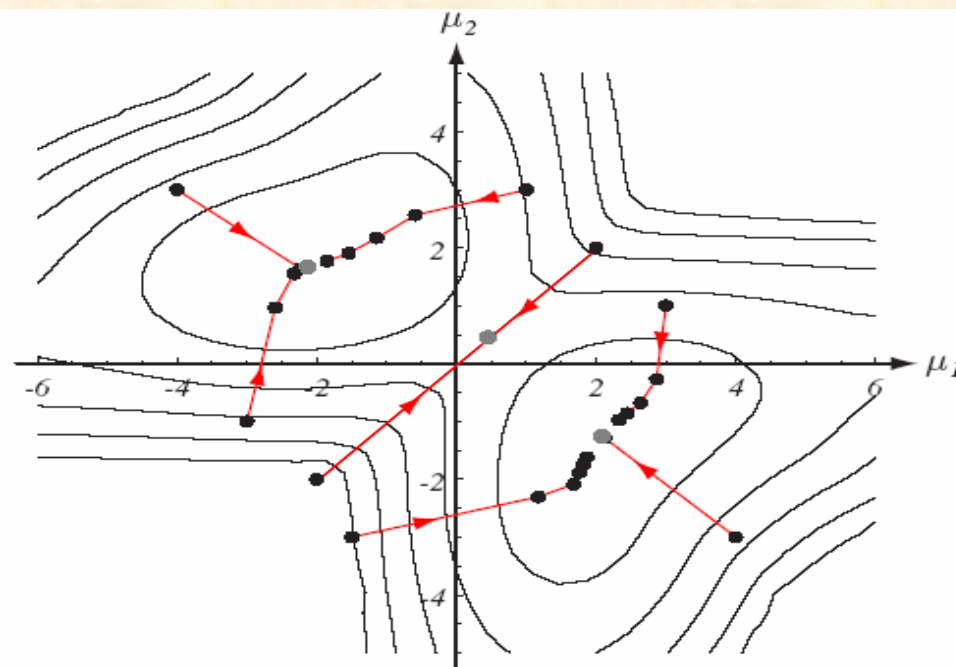
- ❖ Vamos supor, a princípio, que nosso espaço de atributos é tal que podemos calcular um vetor médio  $\mathbf{m}$
- ❖ Supomos também a existência de uma função de dissimilaridade  $d(\mathbf{x}, \mathbf{m})$ , que mede o quão diferentes são os vetores de atributos  $\mathbf{x}$  e  $\mathbf{m}$ .
- ❖ Dado um conjunto de amostras  $x_1 \dots x_N$  a serem classificadas em  $K$  classes  $m_1 \dots m_K$ , definimos o algoritmo K-Means.

# K-Means

- ❖ Inicializar aleatoriamente  $m_1 \dots m_K$ .
- ❖ À cada iteração:
  - Rotular  $x_i$ , associando ao  $m_j$  mais próximo
  - Ajustar  $m_j$  conforme as médias dos elementos  $x_i$  associados a  $m_j$
- ❖ Até que as rotulações não se alterem.

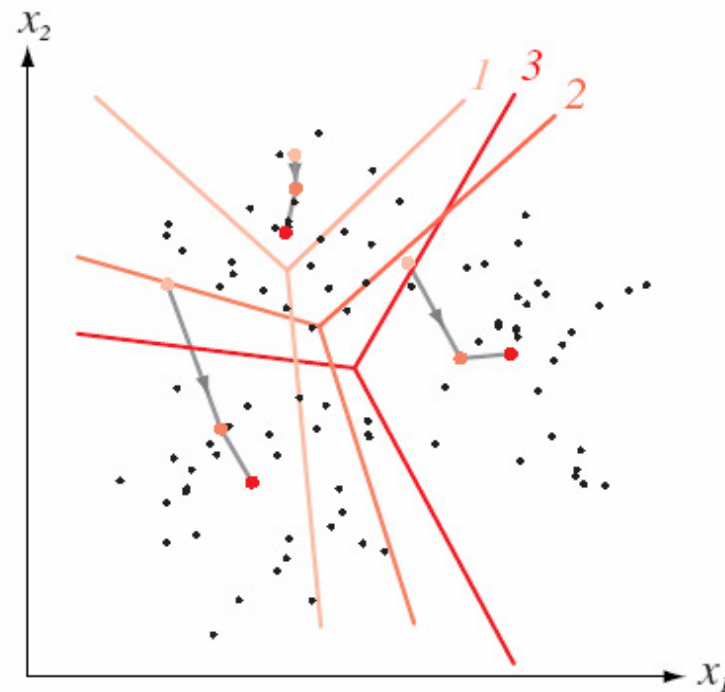


## K-means otimiza verossimilhança



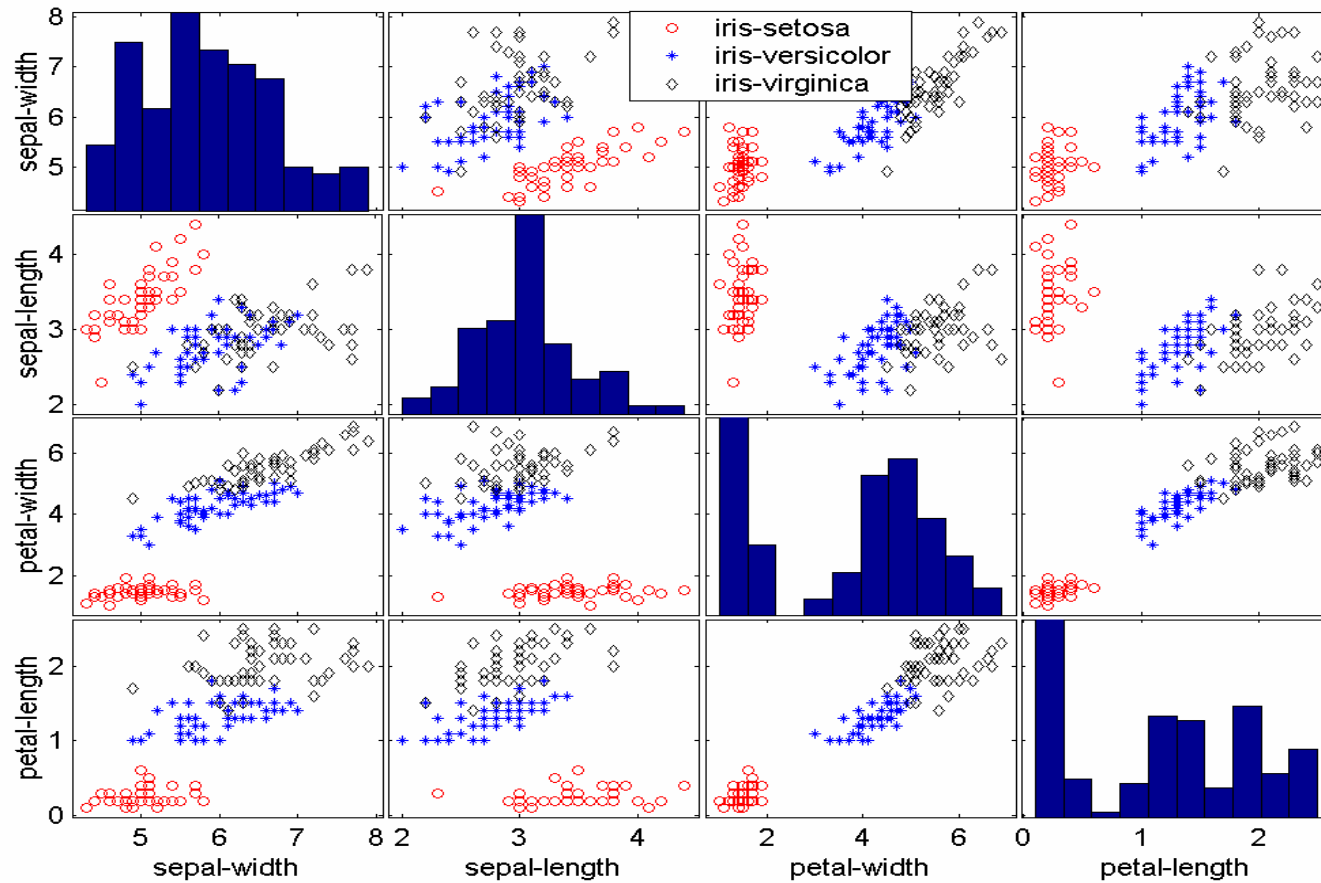
**FIGURE 10.2.** The  $k$ -means clustering procedure is a form of stochastic hill climbing in the log-likelihood function. The contours represent equal log-likelihood values for the one-dimensional data in Fig. 10.1. The dots indicate parameter values after different iterations of the  $k$ -means algorithm. Six of the starting points shown lead to local maxima, whereas two (i.e.,  $\mu_1(0) = \mu_2(0)$ ) lead to a saddle point near  $\boldsymbol{\mu} = \mathbf{0}$ . From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

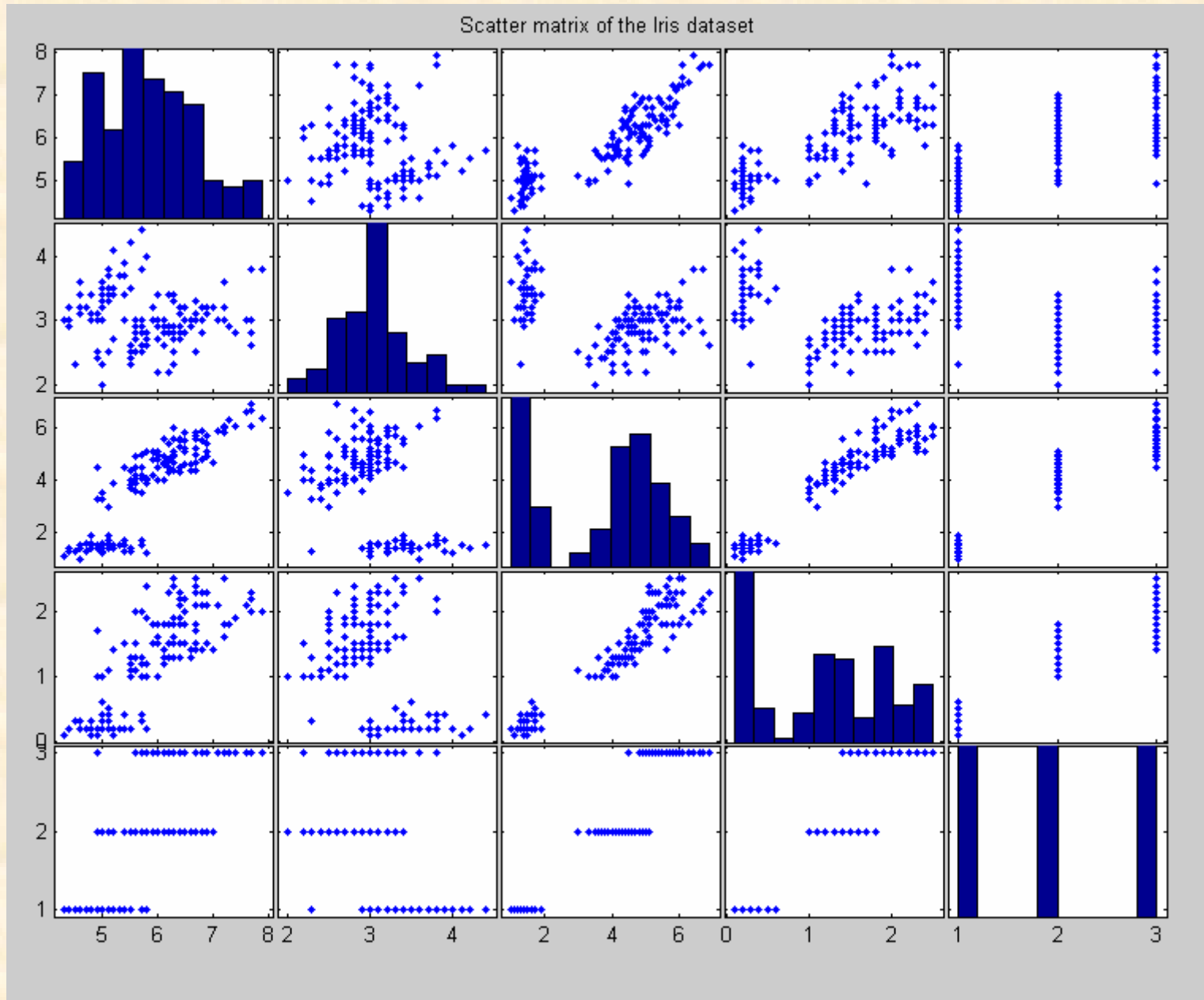
## Passos do K-means

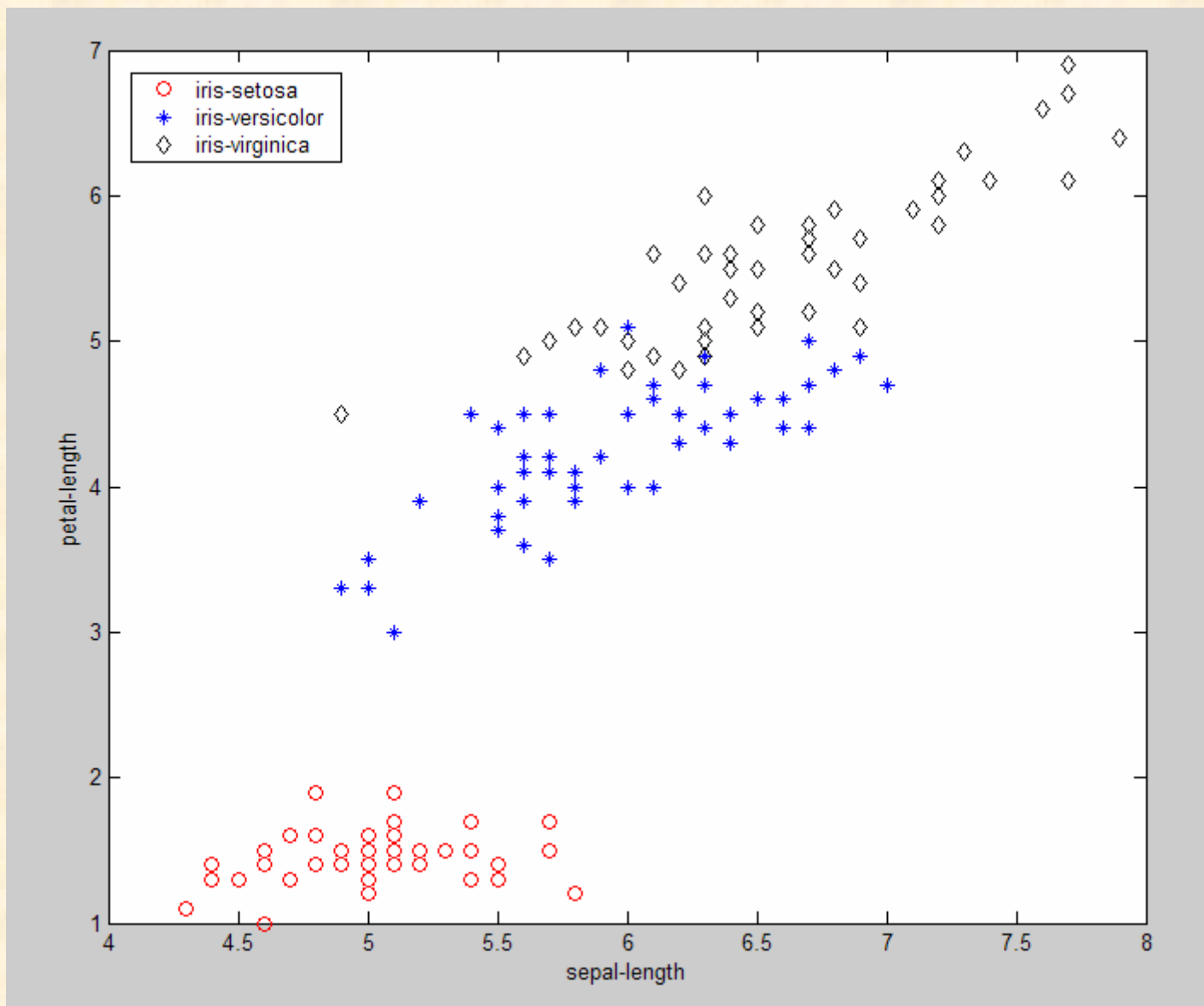


**FIGURE 10.3.** Trajectories for the means of the  $k$ -means clustering procedure applied to two-dimensional data. The final Voronoi tessellation (for classification) is also shown—the means correspond to the “centers” of the Voronoi cells. In this case, convergence is obtained in three iterations. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

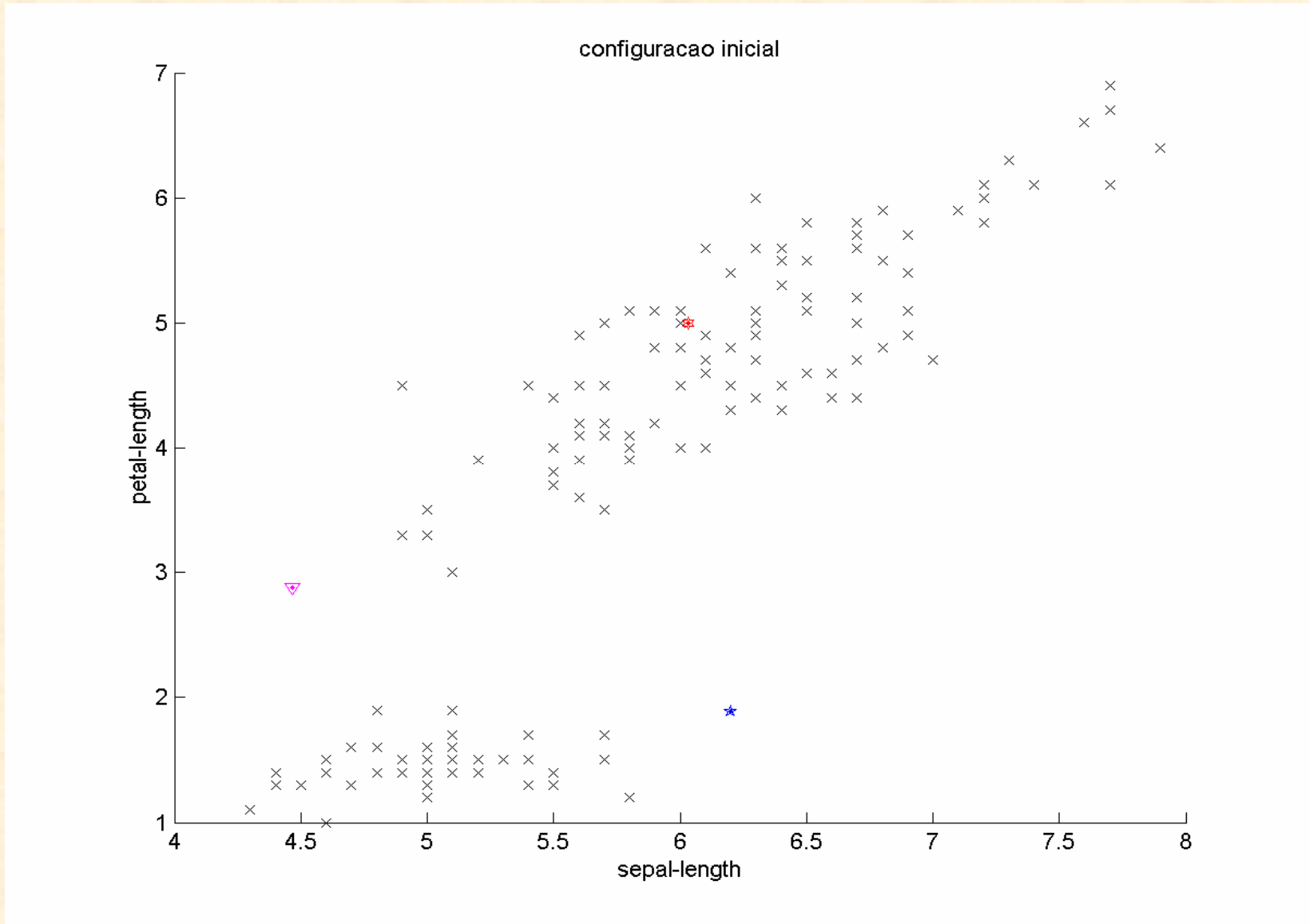
# Exemplo: Iris



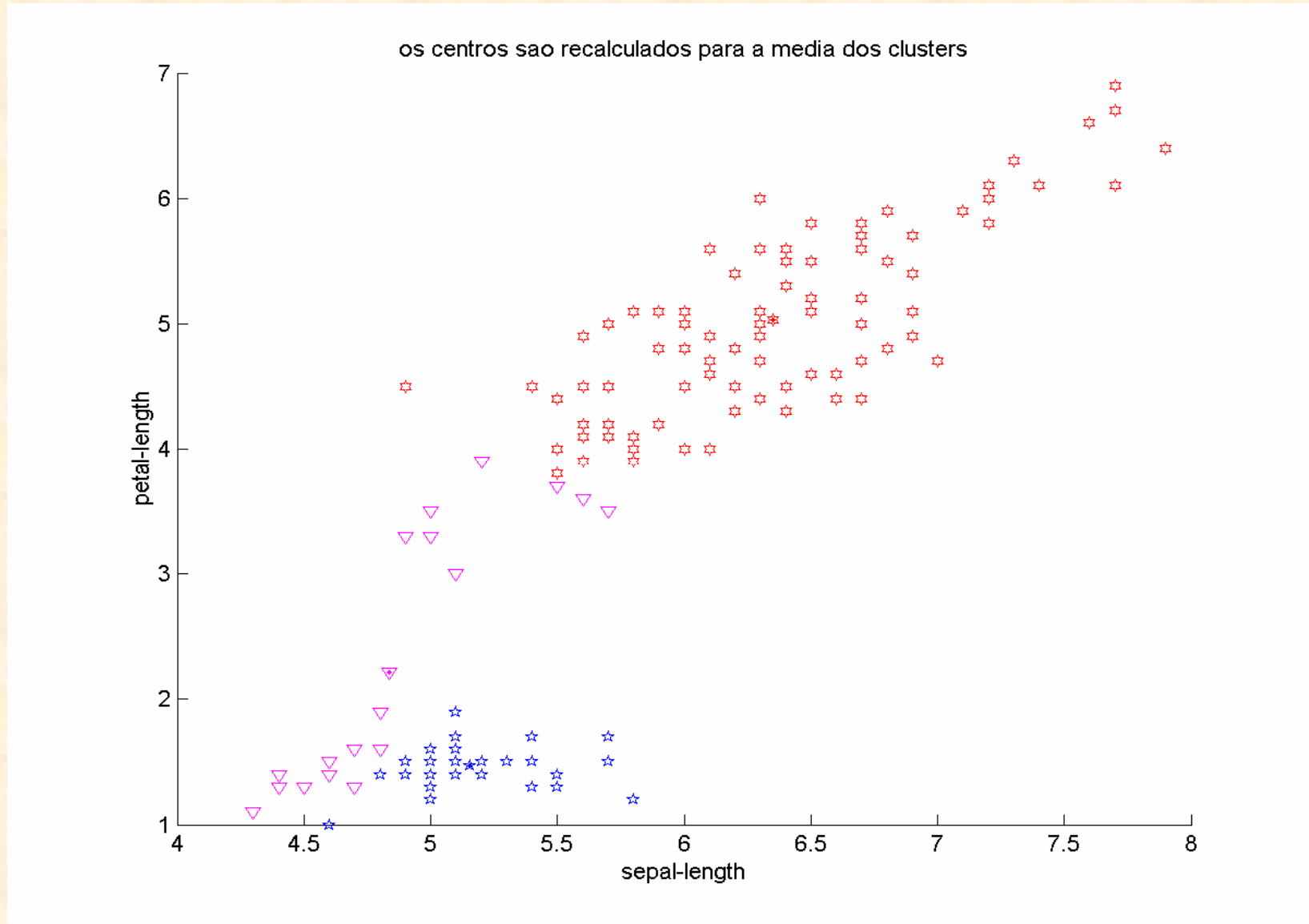


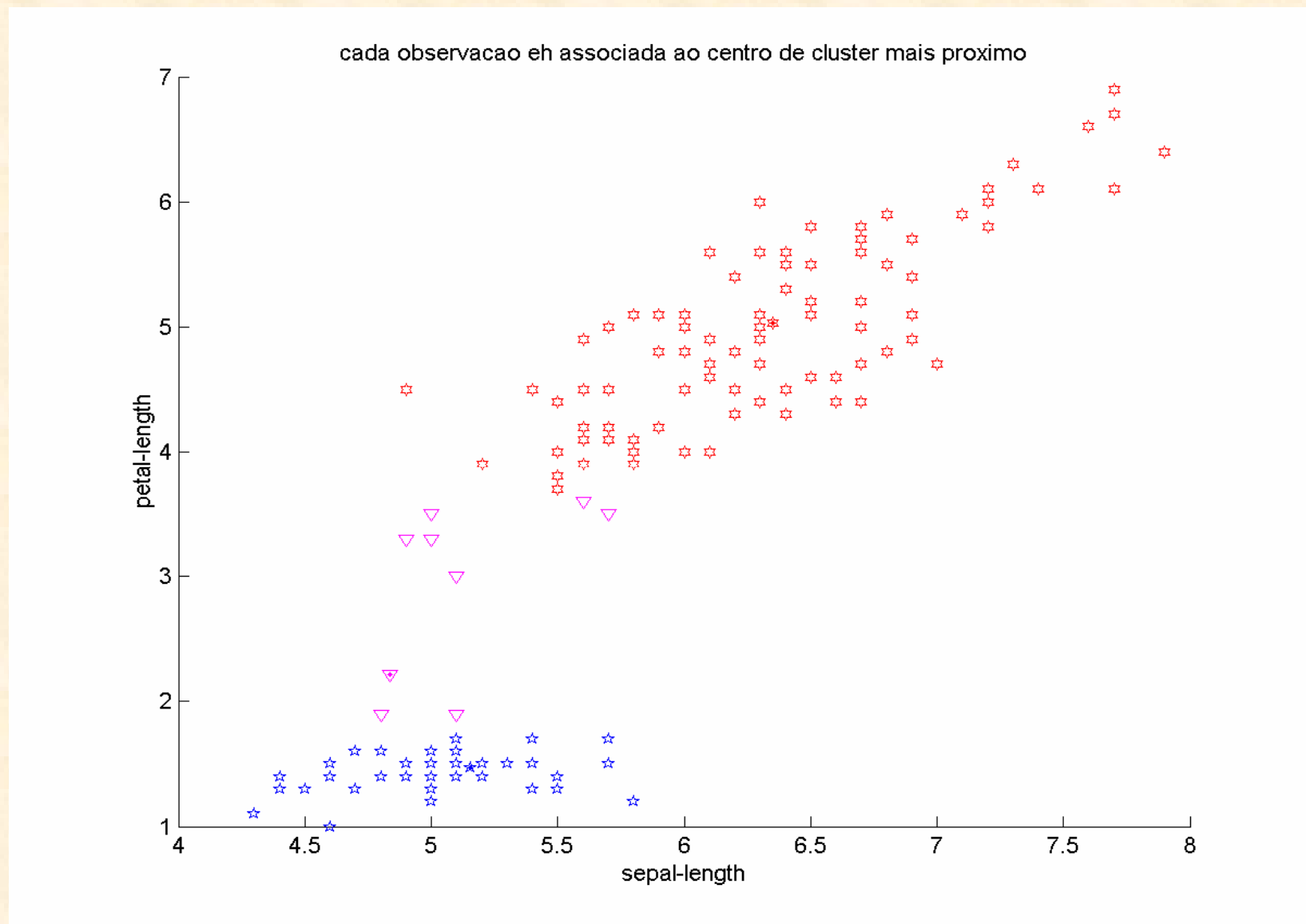


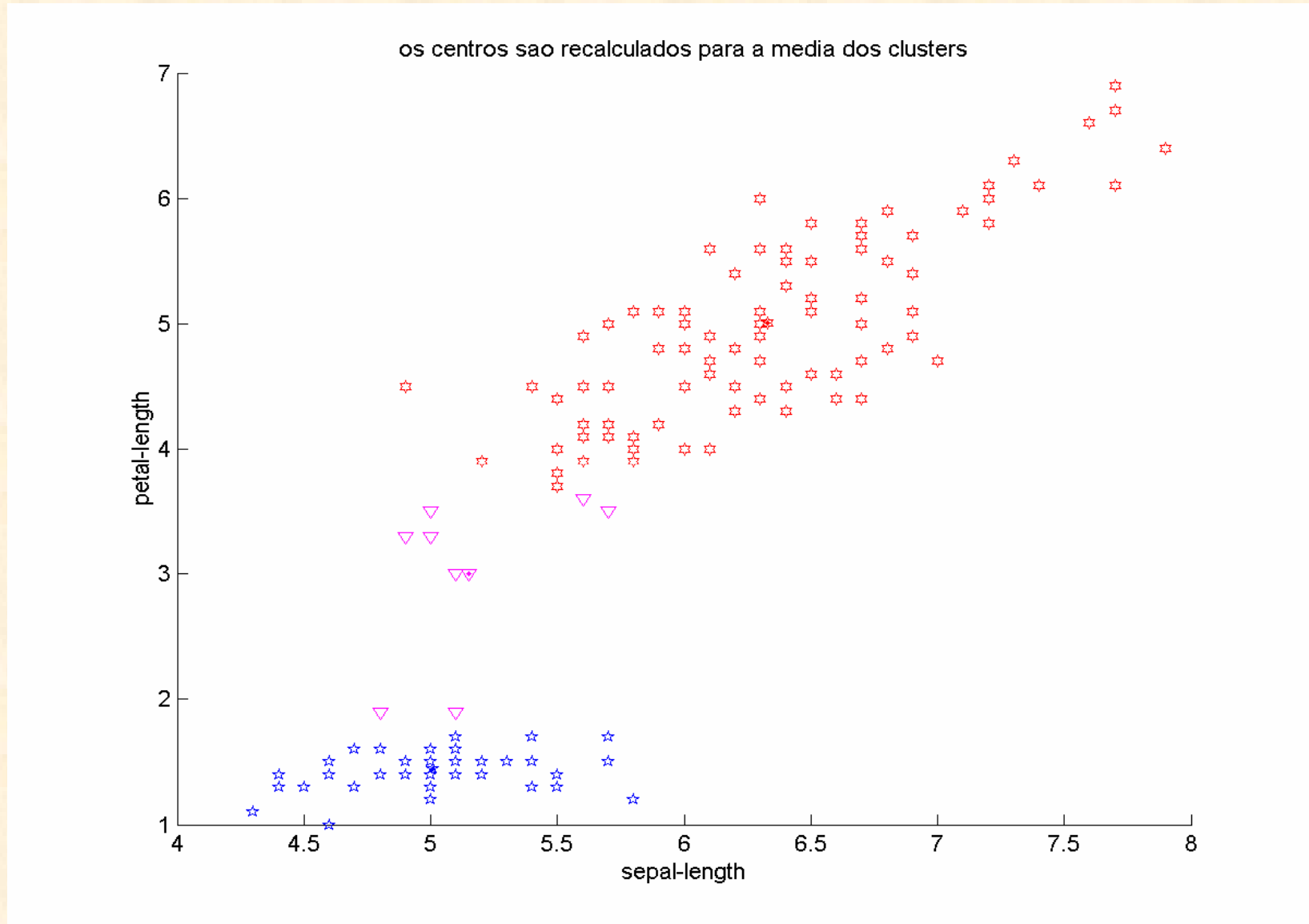


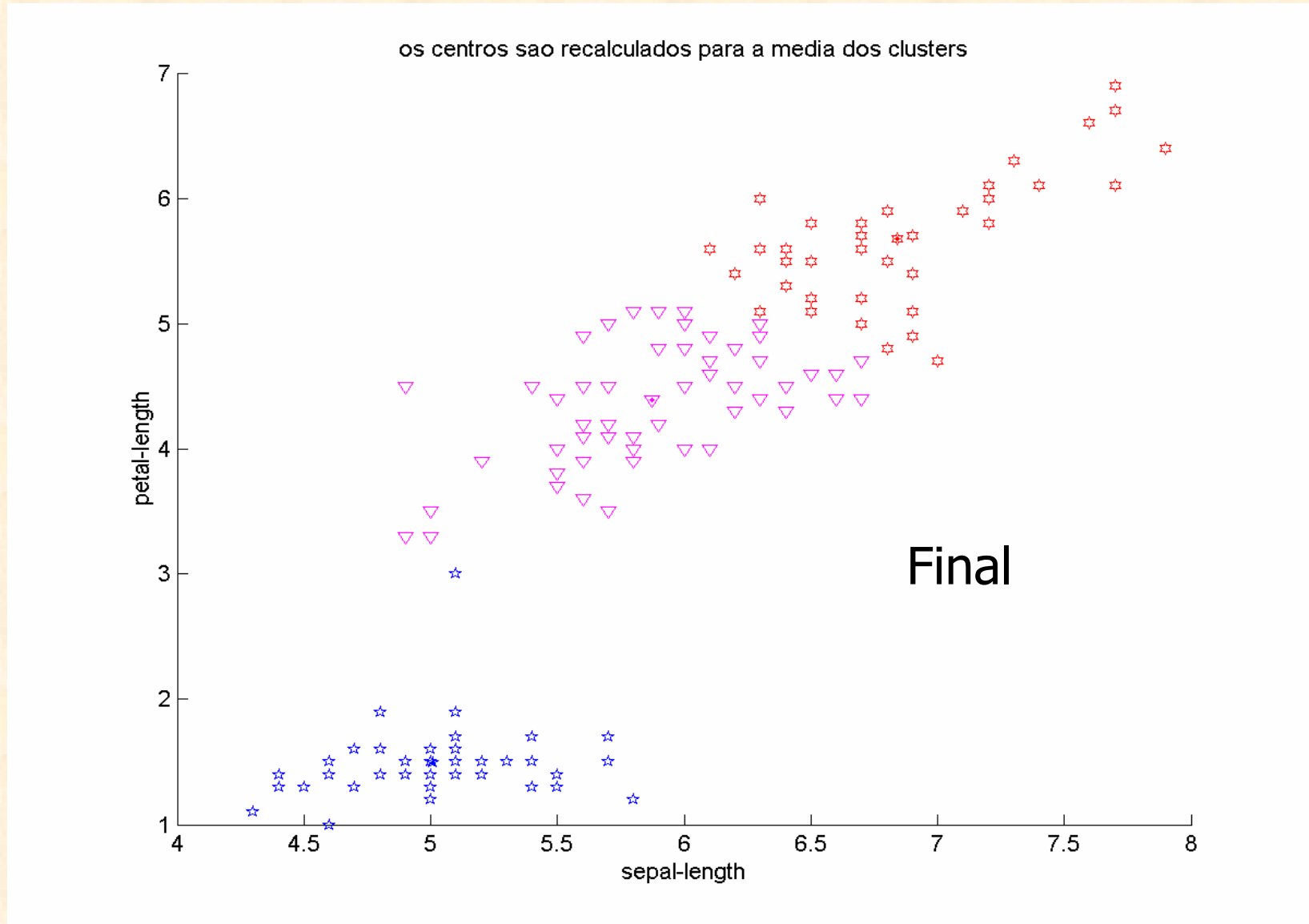












# Kmeans.m (parte 1)

- ❖ loadiris;
- ❖ % vamos pegar sepal-length e petal-length
- ❖ data=[iris(:,1),iris(:,3)];
- ❖ k=3; % tres clusters
- ❖ %clu[j,l] onde l eh o atributo e j eh o cluster
- ❖ %sorteia centro dos clusters "dentro do grafico"
- ❖ for(i=1:k)
- ❖     clu(i,1)=min(data(:,1))+rand\*(max(data(:,1))-min(data(:,1)));
- ❖     clu(i,2)=min(data(:,2))+rand\*(max(data(:,2))-min(data(:,2)));
- ❖ end
- ❖ % plotar dados e centros dos clusters
- ❖ figure; hold on;
- ❖ plot(data(:,1),data(:,2),'kx');
- ❖ plot(clu(1,1),clu(1,2),'rh');plot(clu(1,1),clu(1,2),'r.');
- ❖ plot(clu(2,1),clu(2,2),'bp');plot(clu(2,1),clu(2,2),'b.');
- ❖ plot(clu(3,1),clu(3,2),'mv');plot(clu(3,1),clu(3,2),'m.');
- ❖ xlabel('sepal-length');
- ❖ ylabel('petal-length');
- ❖ title('configuracao inicial');

## Kmeans.m (parte 2)

```
❖ oldb=zeros( size(data,1), k);
❖ b=ones( size(oldb) );

❖ while any( any( oldb~=b ) )
❖     % associar os elementos aos clusters mais proximos
❖     % b(i,j) onde i eh o elemento e j eh o cluster
❖
❖     oldb=b;
❖     for i=1:size(data,1)
❖         dists = (data(i,1)-clu(:,1)).^2 + (data(i,2)-clu(:,2)).^2;
❖         b(i,:)=zeros(1,k);
❖         loc=find(dists==min(dists));
❖         b(i,loc(1))=1;
❖     end
❖
❖     % plotar dados ligados a cada cluster
❖     pause; cla reset; hold on;
❖     plot(data(find(b(:,1)),1),data(find(b(:,1)),2),'rh');
❖     plot(data(find(b(:,2)),1),data(find(b(:,2)),2),'bp');
❖     plot(data(find(b(:,3)),1),data(find(b(:,3)),2),'mv');
❖     plot(clu(1,1),clu(1,2),'rh');plot(clu(1,1),clu(1,2),'r. ');
❖     plot(clu(2,1),clu(2,2),'bp');plot(clu(2,1),clu(2,2),'b. ');
❖     plot(clu(3,1),clu(3,2),'mv');plot(clu(3,1),clu(3,2),'m. ');
❖     xlabel('sepal-length');
❖     ylabel('petal-length');
❖     title('cada observacao eh associada ao centro de cluster mais proximo');
```



## Kmeans.m (parte 3)

```
❖ % recalcular a posicao dos clusters
❖ % pela media dos elementos associados
❖
❖ clu=(b'*data) ./ (b'*ones(size(data)));
❖
❖ % plotar os novos clusters
❖ pause; cla reset; hold on;
❖ plot(data(find(b(:,1)),1),data(find(b(:,1)),2),'rh');
❖ plot(data(find(b(:,2)),1),data(find(b(:,2)),2),'bp');
❖ plot(data(find(b(:,3)),1),data(find(b(:,3)),2),'mv');
❖ plot(clu(1,1),clu(1,2),'rh');plot(clu(1,1),clu(1,2),'r. ');
❖ plot(clu(2,1),clu(2,2),'bp');plot(clu(2,1),clu(2,2),'b. ');
❖ plot(clu(3,1),clu(3,2),'mv');plot(clu(3,1),clu(3,2),'m. ');
❖ xlabel('sepal-length');
❖ ylabel('petal-length');
❖ title('os centros sao recalculados para a media dos clusters');
❖
❖ end
❖ disp('\n\nTerminou!!!!');
```

## Aplicações do K-means

### ❖ Quantização cromática

- ❖ Representar uma imagem colorida de 24-bits (até 16 milhões de cores por pixel) em uma imagem com 8-bits por pixel.
- ❖ Utilizar uma tabela de cores em que há 256 cores disponíveis dentre as 16 milhões.
- ❖ Mapear cada pixel na cor mais apropriada, de forma a ficar mais semelhantes à imagem original.

## Aplicações do K-means

### ❖ Quantização de vetores

- ❖ Caso geral do problema anterior, utilizado para compressão de sinais com perda.
- ❖ Amostra  $x_1 \dots x_i \dots x_N$  e  $K$  vetores de referência  $m_1 \dots m_j \dots m_K$ .
- ❖ Os vetores  $m_j$  formam um dicionário ou codebook.
- ❖ Para codificar, escolhemos  $m_i$  de menor distância para a observação  $x_i$ .
- ❖ Para criar o código, utilizamos o k-means para escolher o conjunto dos vetores  $m_j$ .

## Fuzzy K-Means

- ❖ Em oposição ao método K-Means que rotula cada instância como pertencente a uma classe, o FCM (Fuzzy c-means) ou FKM atribui um grau de pertinência ao par instância-classe.
- ❖ O grau de pertinência está associado à distância da instância ao centro de cada cluster.
- ❖ Os centros dos clusters são, então, re-estimados com base numa média ponderada das observações.
- ❖ Observa-se pouca vantagem em relação ao K-means.
- ❖ Entretanto, se ao invés de médias utilizamos “protótipos” das formas dos clusters e ao invés de “distâncias” de ponto a ponto utilizamos as pertinências de pontos aos “protótipos”, esse método é recomendado.

## E.M.

- ❖ Expectation-maximization ou Maximização do valor esperado é um método de estimação de parâmetros.
- ❖ É utilizado quando não conseguimos resolver a estimação de parâmetros por métodos diretos. (por exemplo, no caso em que há valores não-tabelados ou no caso das misturas de distribuições).
- ❖ No caso das misturas, os valores faltantes são os rótulos dizendo a que classe pertence cada observação.
- ❖ É uma técnica iterativa com convergência estocástica.
- ❖ É um método guloso: vai encontrar mínimos locais.

## E.M.

### ❖ Divide-se em dois passos:

- Passo E: (valor esperado) Utilizo as distribuições de probabilidades estimadas para "completar" os dados faltantes com seus valores esperados. (corresponde à rotulação do K-Means)
- Passo M: (maximização) utilizamos as tabelas "completadas" com os dados faltantes estimados para estimar novos parâmetros através da maximização da verossimilhança. (corresponde à estimação das médias no K-means)

## E.M. para Clustering

- ❖ É uma “extensão” do K-means
- ❖ Como, em geral, o processo de EM é pesado e precisa de uma estimativa inicial próxima do resultado esperado, o K-means é utilizado para iniciar o algoritmo de EM.
- ❖ São feitos alguns passos do K-means.
- ❖ A verossimilhança a ser maximizada é

$$L(\Phi | D) = \log \prod_i p(x_i | \Phi) = \sum_i \log \sum_{j=1}^K p(x_i | G_j) P(G_j)$$

- ❖  $P(G_j)$  corresponde à proporção das misturas e chamaremos de  $\pi_j$ .
- ❖  $D$  é o conjunto de observações  $x_i$ .
- ❖  $\Phi$  consiste em  $\pi_j$ , mais estatísticas suficientes para  $p(x_i | G_j)$

## E.M. para clustering (passo E)

- ❖ Consideramos, agora uma versão da função de verossimilhança em que os dados escondidos (faltantes) são representados num conjunto  $Z$  de variáveis.

$$L_C(\Phi | D, Z)$$

- ❖ No caso do clustering,  $Z$  corresponde aos rótulos associando cada observação a uma classe.
- ❖ Definimos  $Q$  como o valor esperado para a verossimilhança completa, com base nos dados conhecidos e nos parâmetros estimados até então.

$$Q(\Phi | \Phi_l) = E[L_C(\Phi | D, Z) | D, \Phi_l]$$

- ❖ Como a verossimilhança completa depende de  $Z$  diretamente, estamos estimando  $Z$  no passo E.



## E.M. para clustering (passo M)

- ❖ Uma vez estimados os valores de  $Z$ , vamos estimar novos valores para os atributos  $\Phi$  de forma a maximizar a verossimilhança  $Q$

$$\Phi_{l+1} = \arg \max_{\Phi} Q(\Phi, \Phi_l)$$

- ❖ Neste passo, obtemos as médias para cada cluster e também as matrizes de covariância.
- ❖ Definindo a verossimilhança da instância  $x_i$  pertencer à classe  $G_j$  como um conjunto de variáveis escondidas  $h_{i,j}$ :

## E.M. algoritmo

❖ Passo E:

$$h_{i,j} = \frac{\pi_j |S_j|^{-1/2} \exp\left[-(1/2)(x_i - m_j)^T S_j^{-1} (x_i - m_j)\right]}{\sum_g \pi_g |S_g|^{-1/2} \exp\left[-(1/2)(x_i - m_g)^T S_g^{-1} (x_i - m_g)\right]}$$

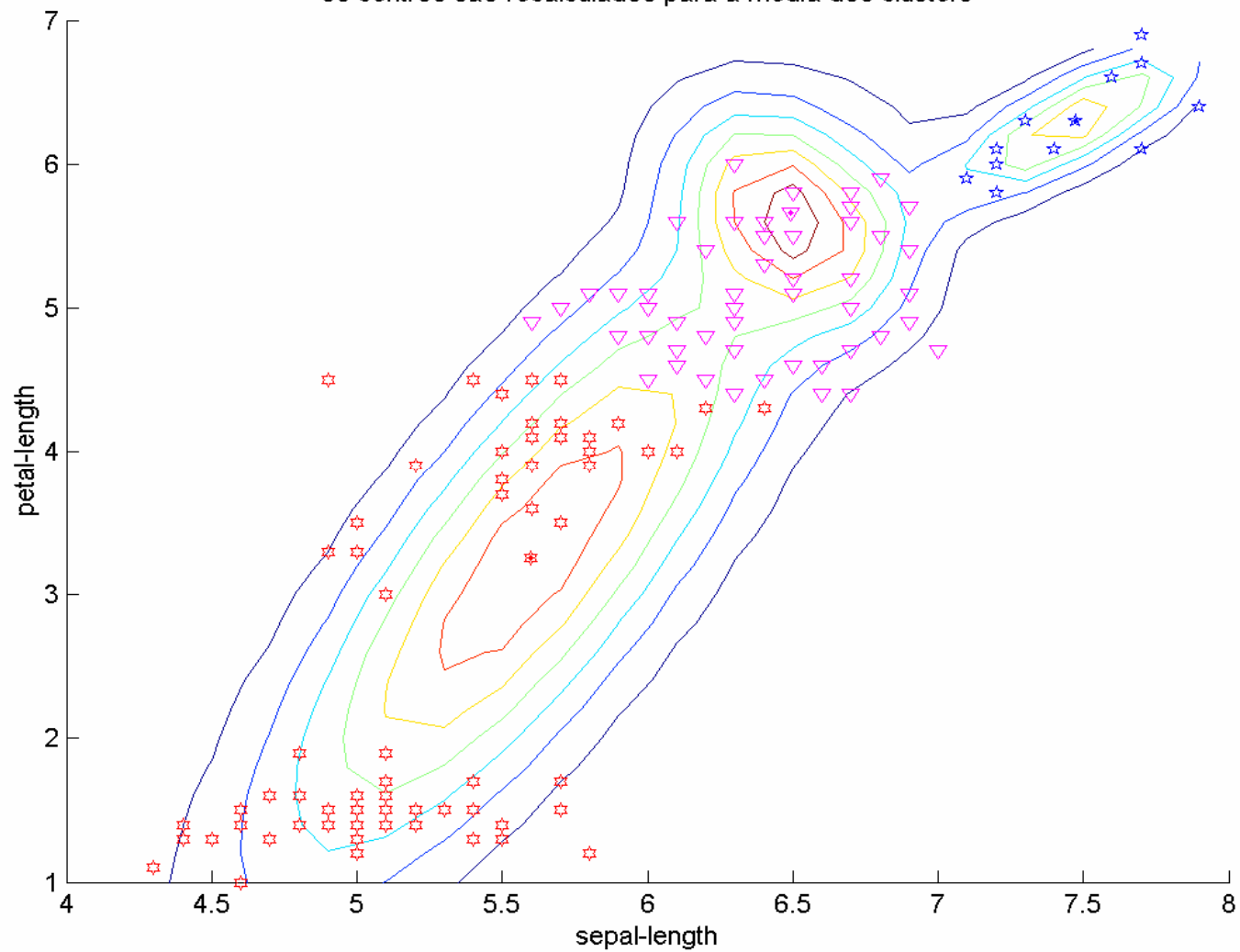
$$\pi_j = \frac{\sum_i h_{i,j}}{N}$$

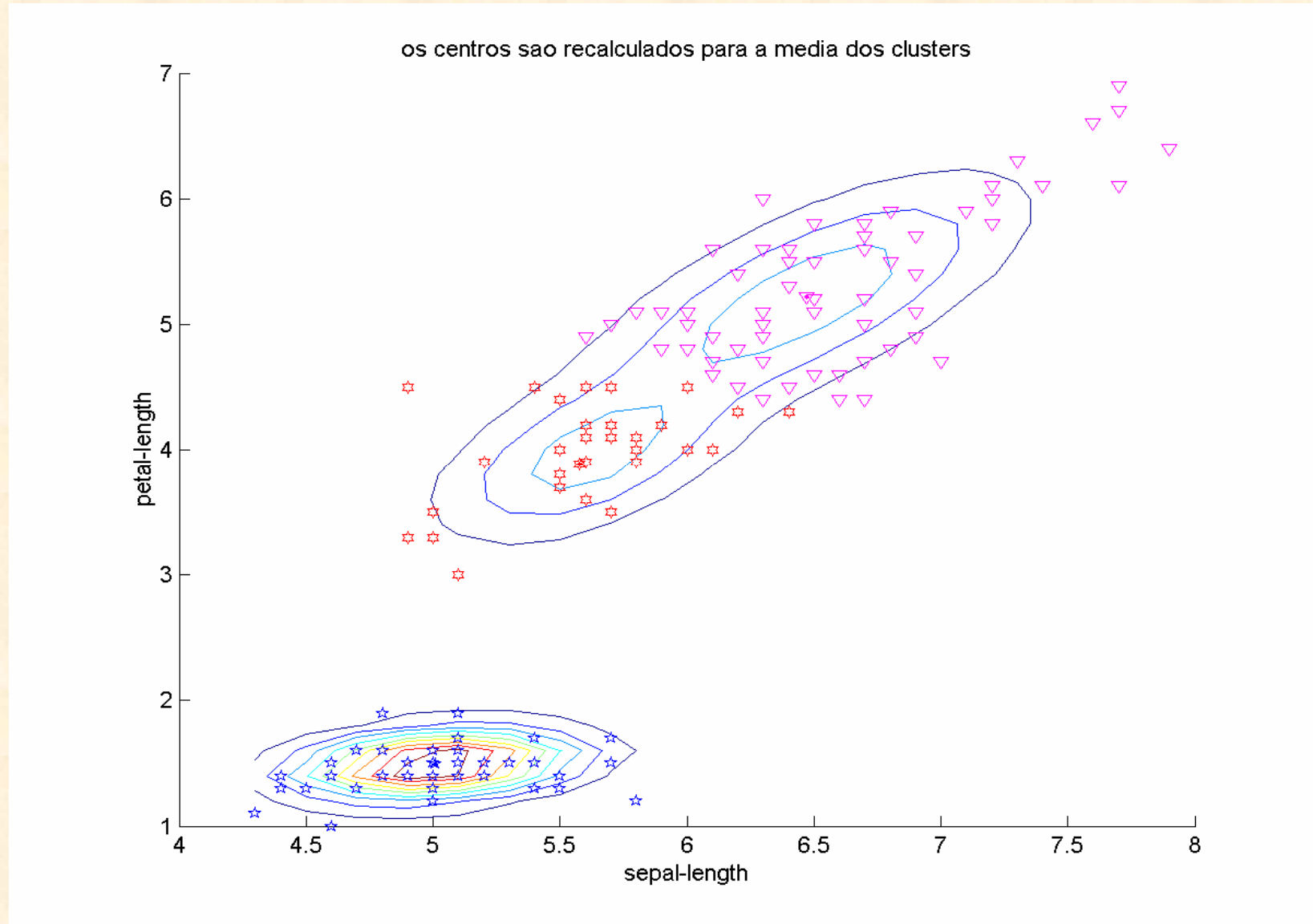
❖ Passo M:

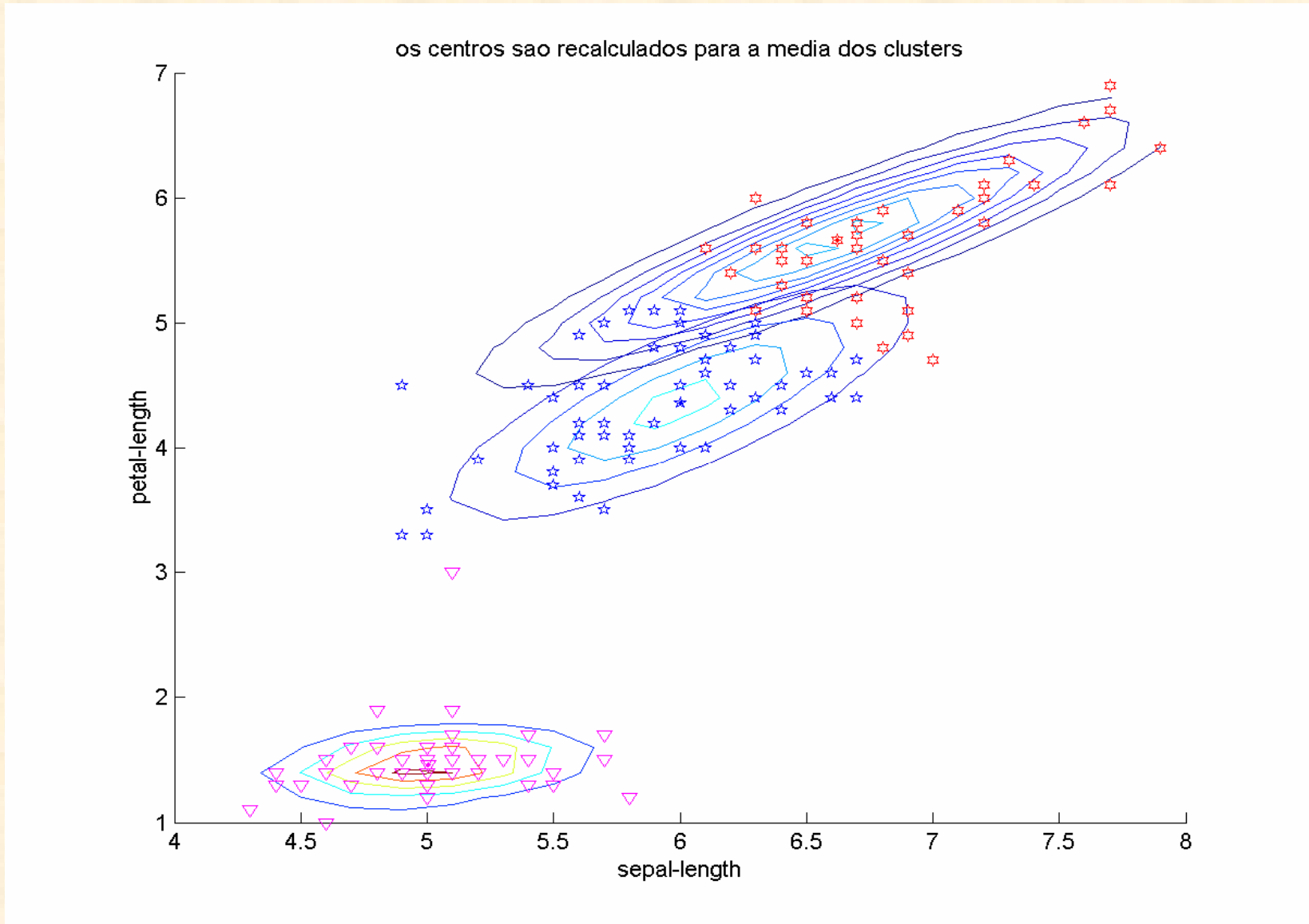
$$m_j = \frac{\sum_j h_{i,j} x_i}{\sum_j h_{i,j}}$$

$$S_j = \frac{\sum_j h_{i,j} (x_i - m_j)(x_i - m_j)^T}{\sum_j h_{i,j}}$$

os centros são recalculados para a media dos clusters







## K-Means online

- ❖ Algoritmos online fazem acesso a cada elemento do conjunto de treinamento uma única vez.
- ❖ Definimos o erro de reconstrução como

$$E(\{m_j\}_{j=1}^K | D) = \frac{1}{2} \sum_i \sum_j b_{i,j} \|x_i - m_j\|^2$$

- ❖ Onde b é 1 se a distância de  $x_i$  ao  $m_j$  é mínima ou b=0 caso contrário
- ❖ Utilizando a descida na direção do gradiente para minimizar E. O método do gradiente estocástico realiza a descida online, enquanto recebe novas observações.

$$\Delta m_{k,j} = -\eta \frac{\partial E}{\partial m_{k,j}} = \eta b_{i,k} (x_{i,j} - m_{k,j})$$

## K-Means online

- ❖ Em outras palavras: Move-se o centro do cluster que é mais próximo da última observação na direção desta por um fator  $\eta$ .
- ❖ Inicializar  $m_j$  com  $j=1 \dots K$
- ❖ Repetir para cada  $x_i$  vindo em ordem aleatória:

$$q = \arg \min_j \|x_i - m_j\|$$

$$m_q = m_q + \eta(x_i - m_q)$$

- ❖ Até a convergência de  $m$ .

# Algoritmos de clustering

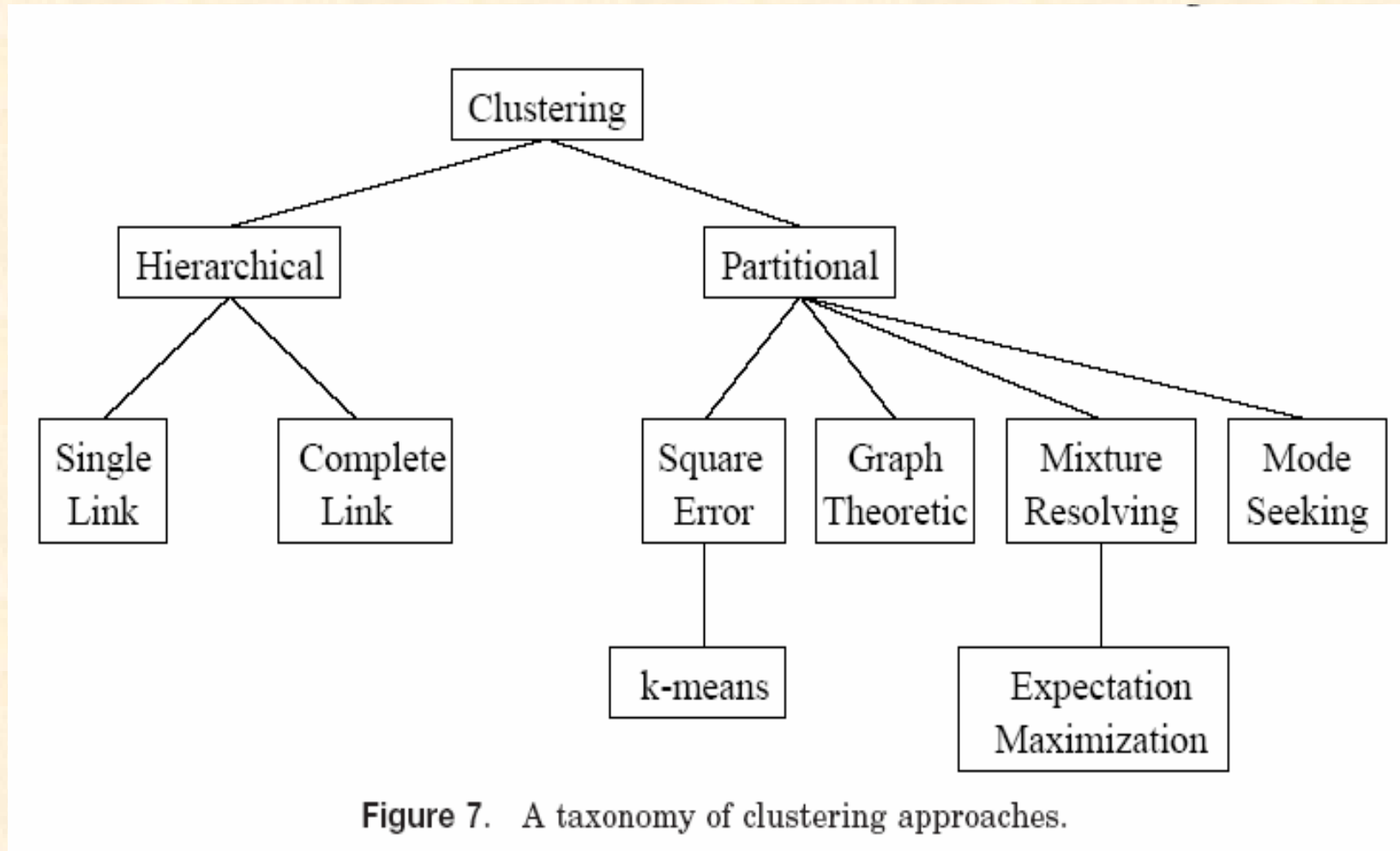


Figure 7. A taxonomy of clustering approaches.



## Exemplo: Algoritmo monotético

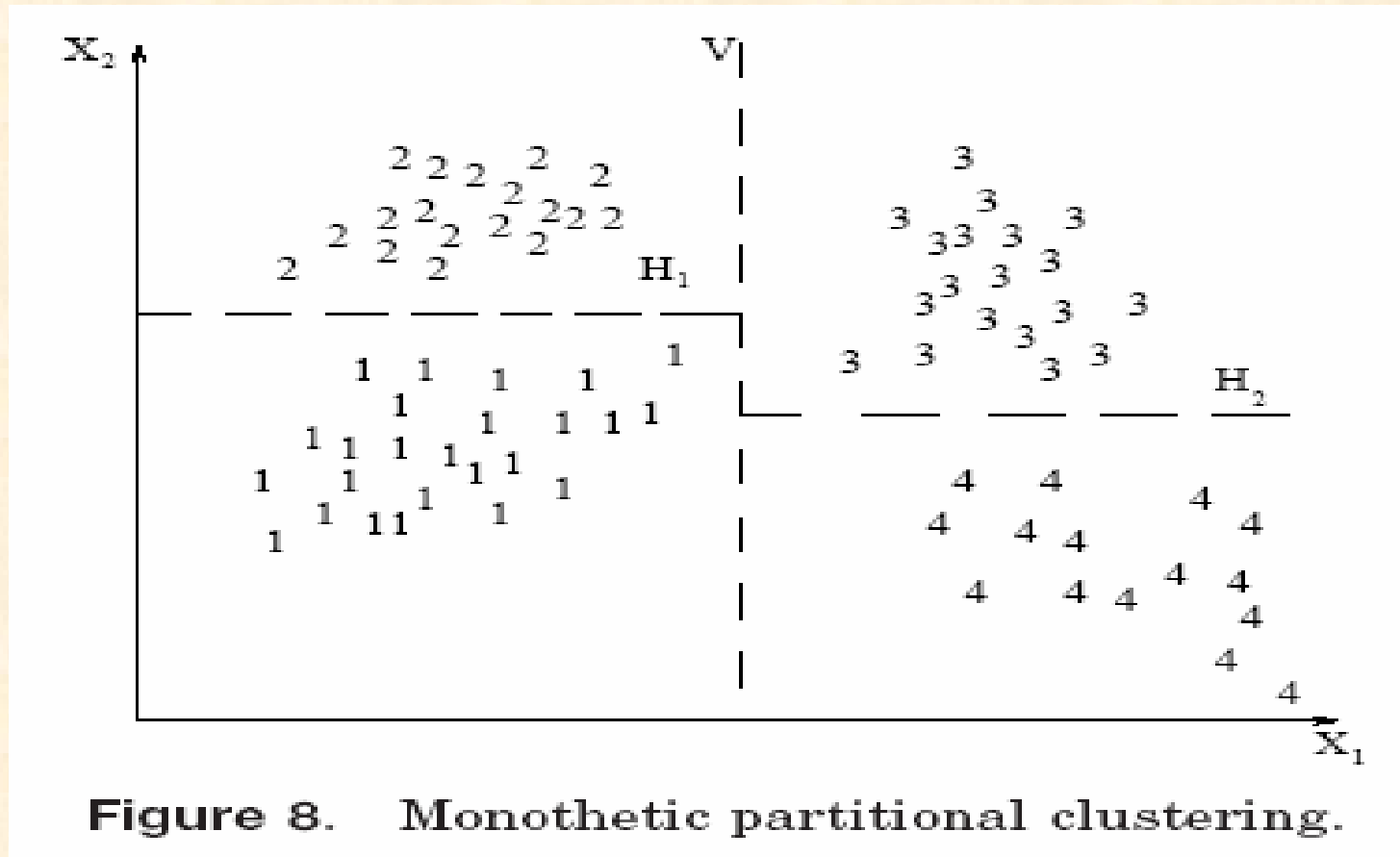


Figure 8. Monothetic partitional clustering.

Jain et al., ACM Computing Surveys, Vol. 31, No. 3, September 1999

# PROXIMITY MEASURES

## ❖ *Between vectors*

➤ **Dissimilarity measure** (between vectors of  $X$ ) is a function

$$d : X \times X \longrightarrow \mathfrak{R}$$

with the following properties

- $\exists d_0 \in \mathfrak{R} : -\infty < d_0 \leq d(\underline{x}, \underline{y}) < +\infty, \forall \underline{x}, \underline{y} \in X$
- $d(\underline{x}, \underline{x}) = d_0, \forall \underline{x} \in X$
- $d(\underline{x}, \underline{y}) = d(\underline{y}, \underline{x}), \forall \underline{x}, \underline{y} \in X$

If in addition

- $d(\underline{x}, \underline{y}) = d_0$  if and only if  $\underline{x} = \underline{y}$
- $d(\underline{x}, \underline{z}) \leq d(\underline{x}, \underline{y}) + d(\underline{y}, \underline{z}), \quad \forall \underline{x}, \underline{y}, \underline{z} \in X$

(triangular inequality)

$d$  is called a **metric dissimilarity measure**.

➤ **Similarity measure** (between vectors of  $X$ ) is a function

$$s : X \times X \longrightarrow \mathfrak{R}$$

with the following properties

- $\exists s_0 \in \mathfrak{R} : -\infty < s(\underline{x}, \underline{y}) \leq s_0 < +\infty, \forall \underline{x}, \underline{y} \in X$
- $s(\underline{x}, \underline{x}) = s_0, \forall \underline{x} \in X$
- $s(\underline{x}, \underline{y}) = s(\underline{y}, \underline{x}), \forall \underline{x}, \underline{y} \in X$

If in addition

- $s(\underline{x}, \underline{y}) = s_0$  if and only if  $\underline{x} = \underline{y}$
  - $s(\underline{x}, \underline{y})s(\underline{y}, \underline{z}) \leq [s(\underline{x}, \underline{y}) + s(\underline{y}, \underline{z})]s(\underline{x}, \underline{z}), \quad \forall \underline{x}, \underline{y}, \underline{z} \in X$
- $s$  is called a **metric** similarity measure.

❖ Between sets

Let  $D_i \subset X, i=1, \dots, k$  and  $U = \{D_1, \dots, D_k\}$

A **proximity measure**  $\wp$  on  $U$  is a function

$$\wp : U \times U \longrightarrow \mathfrak{R}$$

A **dissimilarity measure** has to satisfy the relations of dissimilarity measure between vectors, where  $D_i$ 's are used in place of  $\underline{x}, \underline{y}$  (similarly for **similarity measures**).

# PROXIMITY FUNCTIONS BETWEEN A VECTOR AND A SET

❖ Let  $X = \{\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N\}$  and  $C \subset X$ ,  $\underline{x} \in X$

❖ All points of  $C$  contribute to the definition of  $\wp(\underline{x}, C)$

➤ Max proximity function

$$\wp_{\max}^{ps}(\underline{x}, C) = \max_{\underline{y} \in C} \wp(\underline{x}, \underline{y})$$

➤ Min proximity function

$$\wp_{\min}^{ps}(\underline{x}, C) = \min_{\underline{y} \in C} \wp(\underline{x}, \underline{y})$$

➤ Average proximity function

$$\wp_{\text{avg}}^{ps}(\underline{x}, C) = \frac{1}{n_C} \sum_{\underline{y} \in C} \wp(\underline{x}, \underline{y}) \quad (n_C \text{ is the cardinality of } C)$$

❖ A representative(s) of  $C, r_C$ , contributes to the definition of  $\rho(\underline{x}, C)$

In this case:  $\rho(\underline{x}, C) = \rho(\underline{x}, r_C)$

Typical representatives are:

➤ The mean vector:

$$\underline{m}_p = \left( \frac{1}{n_C} \right) \sum_{y \in C} \underline{y}$$

where  $n_C$  is the cardinality of  $C$

➤ The mean center:

$$\underline{m}_C \in C : \sum_{\underline{y} \in C} d(\underline{m}_C, \underline{y}) \leq \sum_{\underline{y} \in C} d(\underline{z}, \underline{y}), \quad \forall \underline{z} \in C$$

➤ The median center:

$$\underline{m}_{med} \in C : med(d(\underline{m}_{med}, \underline{y}) \mid \underline{y} \in C) \leq med(d(\underline{z}, \underline{y}) \mid \underline{y} \in C), \quad \forall \underline{z} \in C$$

$d$ : a dissimilarity measure

**NOTE:** Other representatives (e.g., hyperplanes, hyperspheres) are useful in certain applications (e.g., object identification using clustering techniques).

# PROXIMITY FUNCTIONS BETWEEN SETS

❖ Let  $X = \{\underline{x}_1, \dots, \underline{x}_N\}$ ,  $D_i, D_j \subset X$  and  $n_i = |D_i|$ ,  $n_j = |D_j|$

❖ All points of each set contribute to  $\wp(D_i, D_j)$

➤ **Max** proximity function (measure but **not** metric, only if  $\wp$  is a similarity measure)

$$\wp_{\max}^{ss}(D_i, D_j) = \max_{\underline{x} \in D_i, \underline{y} \in D_j} \wp(\underline{x}, \underline{y})$$

➤ **Min** proximity function (measure but **not** metric, only if  $\wp$  is a dissimilarity measure)

$$\wp_{\min}^{ss}(D_i, D_j) = \min_{\underline{x} \in D_i, \underline{y} \in D_j} \wp(\underline{x}, \underline{y})$$

➤ **Average** proximity function (**not** a measure, even if  $\wp$  is a measure)

$$\wp_{\text{avg}}^{ss}(D_i, D_j) = \left( \frac{1}{n_i n_j} \right) \sum_{\underline{x} \in D_i} \sum_{\underline{y} \in D_j} \wp(\underline{x}, \underline{y})$$



- ❖ Each set  $D_i$  is represented by its representative vector  $\underline{m}_i$ 
  - Mean proximity function (it is a measure provided that  $\wp$  is a measure):

$$\wp_{mean}^{ss}(D_i, D_j) = \wp(\underline{m}_i, \underline{m}_j)$$

- $$\wp_e^{ss}(D_i, D_j) = \sqrt{\frac{n_i n_j}{n_i + n_j}} \wp(\underline{m}_i, \underline{m}_j)$$

**NOTE:** Proximity functions between a vector  $\underline{x}$  and a set  $C$  may be derived from the above functions if we set  $D_i = \{\underline{x}\}$ .

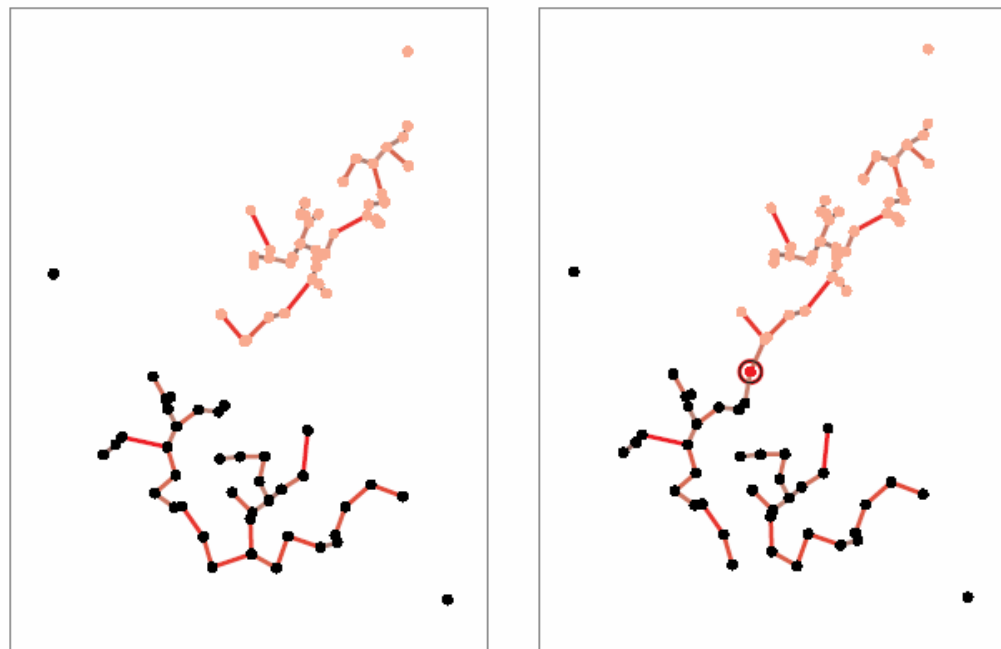
➤ Remarks:

- Different choices of proximity functions between sets may lead to **totally different** clustering results.
- Different proximity measures between vectors in the same proximity function between sets may lead to **totally different** clustering results.
- The only way to achieve a proper clustering is
  - by trial and error and,
  - taking into account the opinion of an expert in the field of application.

# Métodos hierárquicos

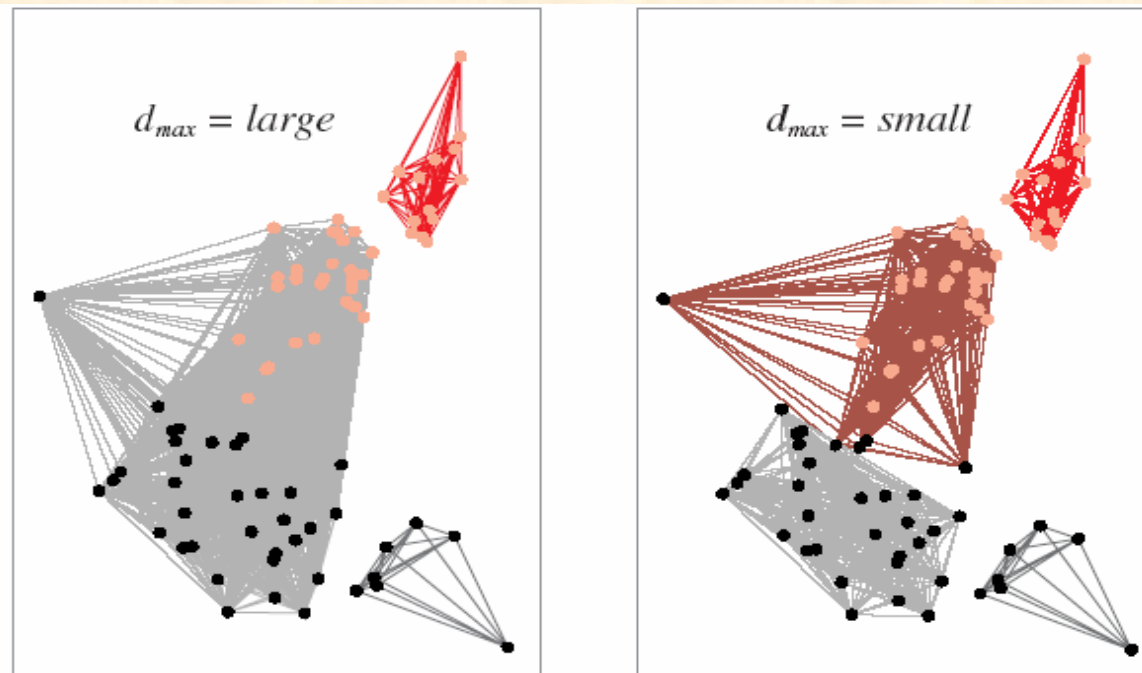
- ❖ Método hierárquico pode ser
  - Aglomerativo: partindo de vários clusters de um único elemento e unindo clusters até que todos elementos pertençam a um único cluster.
  - Divisivo: parte de um único cluster e vai particionando cada cluster até obter um elemento em cada cluster.
  
- ❖ Em função da medida da distância entre dois clusters pode ser:
  - De ligação simples: a distância entre dois clusters é a menor distância possível entre dois pontos, um em cada cluster.
  - De ligação completa: a distância entre dois clusters é a maior distância entre dois pontos, um em cada cluster.

## Single-link



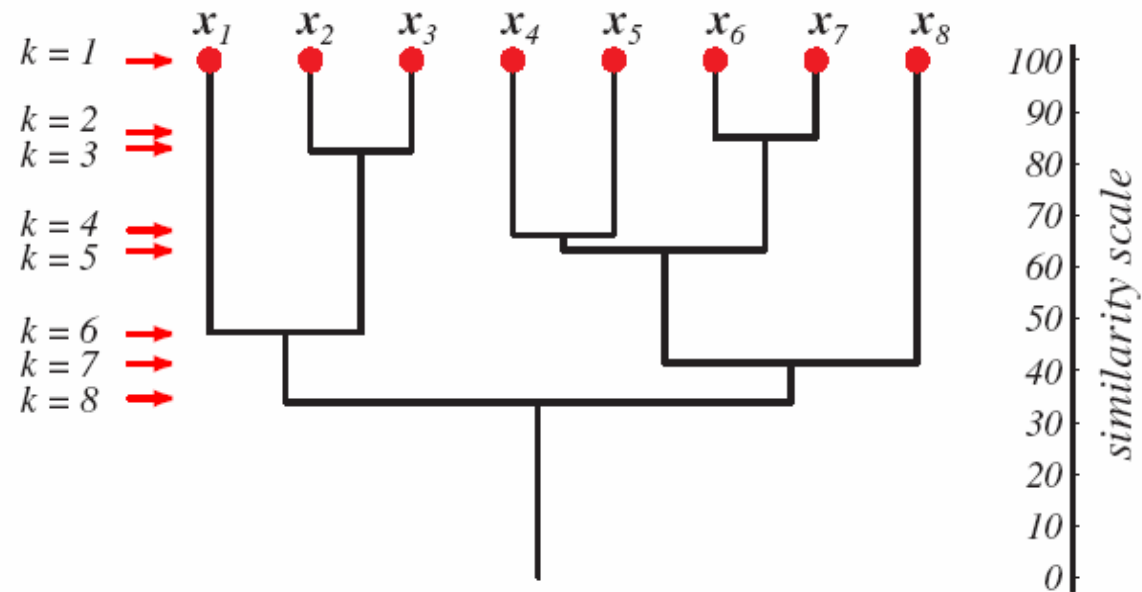
**FIGURE 10.13.** Two Gaussians were used to generate two-dimensional samples, shown in pink and black. The nearest-neighbor clustering algorithm gives two clusters that well approximate the generating Gaussians (left). If, however, another particular sample is generated (circled red point at the right) and the procedure is restarted, the clusters do not well approximate the Gaussians. This illustrates how the algorithm is sensitive to the details of the samples. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

## Complete-link



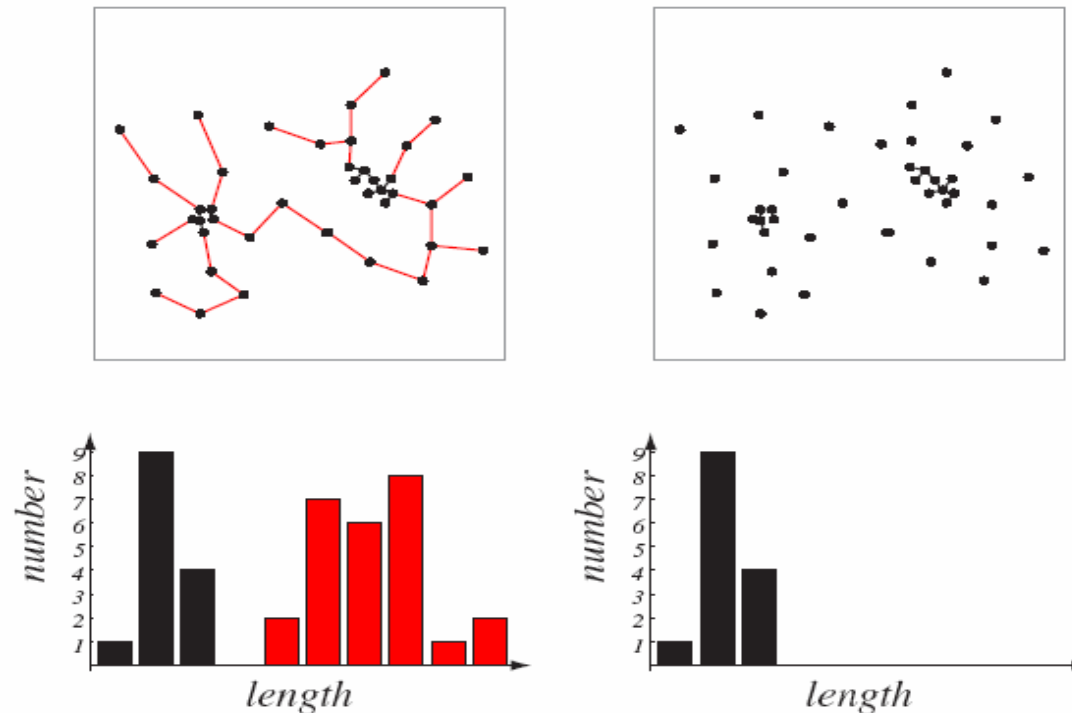
**FIGURE 10.14.** The farthest-neighbor clustering algorithm uses the separation between the most distant points as a criterion for cluster membership. If this distance is set very large, then all points lie in the same cluster. In the case shown at the left, a fairly large  $d_{max}$  leads to three clusters; a smaller  $d_{max}$  gives four clusters, as shown at the right. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Dendrograma



**FIGURE 10.11.** A dendrogram can represent the results of hierarchical clustering algorithms. The vertical axis shows a generalized measure of similarity among clusters. Here, at level 1 all eight points lie in singleton clusters; each point in a cluster is highly similar to itself, of course. Points  $x_6$  and  $x_7$  happen to be the most similar, and are merged at level 2, and so forth. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Minimal Spanning Tree



**FIGURE 10.21.** A minimal spanning tree is shown at the left; its bimodal edge length distribution is evident in the histogram below. If all links of intermediate or high length are removed (red), the two natural clusters are revealed (right). From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Referências

- ❖ Duda, Hart, Stork (livro)
- ❖ Teodoridis, Koutrombas (transparências e livro)
- ❖ Andrew Moore (transparências)