FUNDAMENTALS OF MACHINE LEARNING

CLUSTERING

CSCI3320

Prof. John C.S. Lui, CSE Department, CUHK
Introduction to Machine Learning

Semi-parametric Density Estimation

- **Parametric:** Assume a single model for $p(\boldsymbol{x}|C_i)$ (Chapters 4 and 5), or assume samples come from a known distribution
- **Semi-parametric:** $p(\boldsymbol{x}|C_i)$ is a mixture of densities

  Multiple possible explanations/prototypes:

  Different handwriting styles, accents in speech
- **Nonparametric:** No model; data speaks for itself (Chapter 8)
- **Clustering methods:** allow learning the mixture parameters from data
- Will also discuss **vector quantization** and **hierarchical clustering**

Mixture Densities

$$p(\boldsymbol{x}) = \sum_{i=1}^{k} p(\boldsymbol{x}|\mathcal{G}_i)P(\mathcal{G}_i)$$

k, # of groups, is given

where $\mathcal{G}_i$ the components/groups/clusters,
$P(\mathcal{G}_i)$ mixture proportions (priors),
$p(\boldsymbol{x}|\mathcal{G}_i)$ component densities, *k* is specified

- Gaussian mixture where $p(\boldsymbol{x}|\mathcal{G}_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$
- Parameters we need to estimate:

$$\Phi = \{P(\mathcal{G}_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{k}$$

- Unlabeled sample $\mathcal{X} = \{\boldsymbol{x}^t\}_{t=1}^{N}$ (**unsupervised learning**)

Classes vs. Clusters

- **Supervised:** $\mathcal{X} = \{\boldsymbol{x}^t, \boldsymbol{r}^t\}_{t=1}^N$
- Classes $C_i \quad i = 1, \ldots, K$

Prior probability (via counting)

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

where $p(\boldsymbol{x}|C_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

$$\Phi = \{P(C_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$$

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N}$$

$$\boldsymbol{m}_i = \frac{\sum_t r_i^t \boldsymbol{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\boldsymbol{x}^t - \boldsymbol{m}_i)(\boldsymbol{x}^t - \boldsymbol{m}_i)^T}{\sum_t r_i^t}$$

- **Unsupervised:** $\mathcal{X} = \{\boldsymbol{x}^t\}_t$
- Clusters $\mathcal{G}_i, i = 1, \ldots, k$

$$p(\boldsymbol{x}) = \sum_{i=1}^{k} p(\boldsymbol{x}|\mathcal{G}_i)P(\mathcal{G}_i)$$

where $p(\boldsymbol{x}|\mathcal{G}_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

$$\boldsymbol{\Phi} = \{P(\mathcal{G}_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^k$$

Labels r^t ? *Prior probability (How????)*

To learn which sample belongs to which group, we have ***k*-mean** and **EM**

k-Means Clustering

- **Vector quantization:** mapping from a continuous space to discrete space. Example
- Given $\mathcal{X} = \{\boldsymbol{x}^t\}_{t=1}^N$, find *k* reference vectors (prototypes or codebook vectors or codewords) which best represent data
- Reference vectors, $\boldsymbol{m}_j,\ j = 1, \ldots, k$
- Use nearest (most similar) reference:

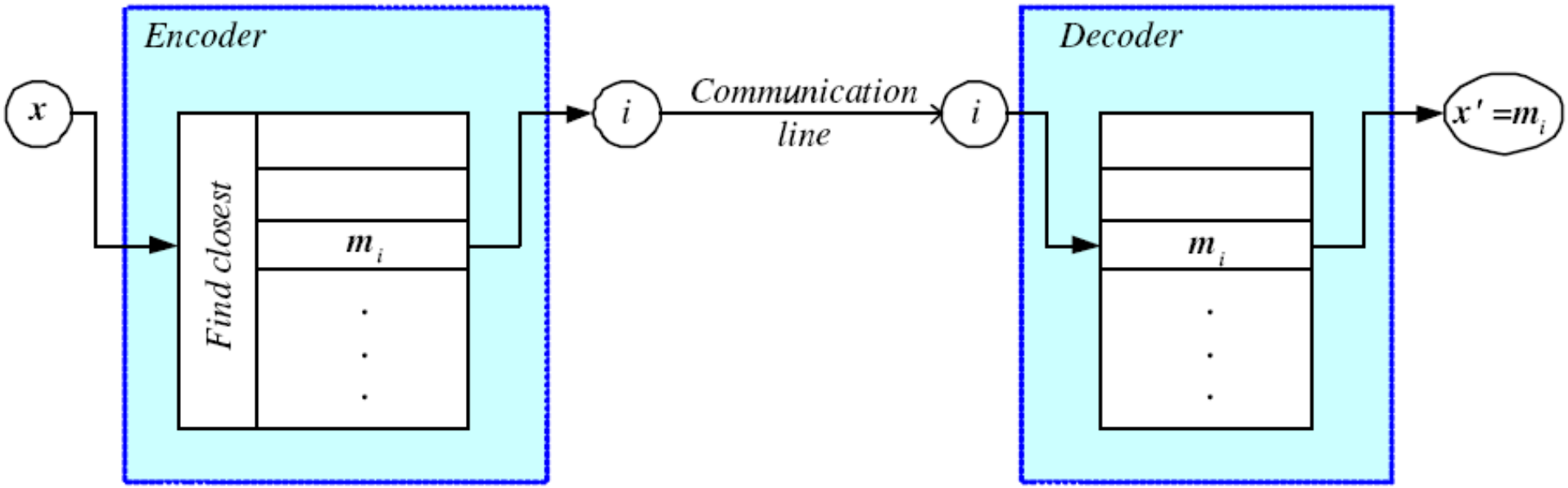
$$\|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\|$$

- It is like "*coding*" to "*decoding*" (see next page)
- Reconstruction error

$$E(\{\boldsymbol{m}_i\}_{i=1}^k | \mathcal{X}) = \sum_t \sum_i b_i^t \|\boldsymbol{x}^t - \boldsymbol{m}_i\|^2$$

$$b_i^t = \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

Encoding/Decoding



k-means Clustering

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

For all $\boldsymbol{x}^t \in \mathcal{X}$

$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

cluster assignment phase

For all $\boldsymbol{m}_i, i = 1, \ldots, k$

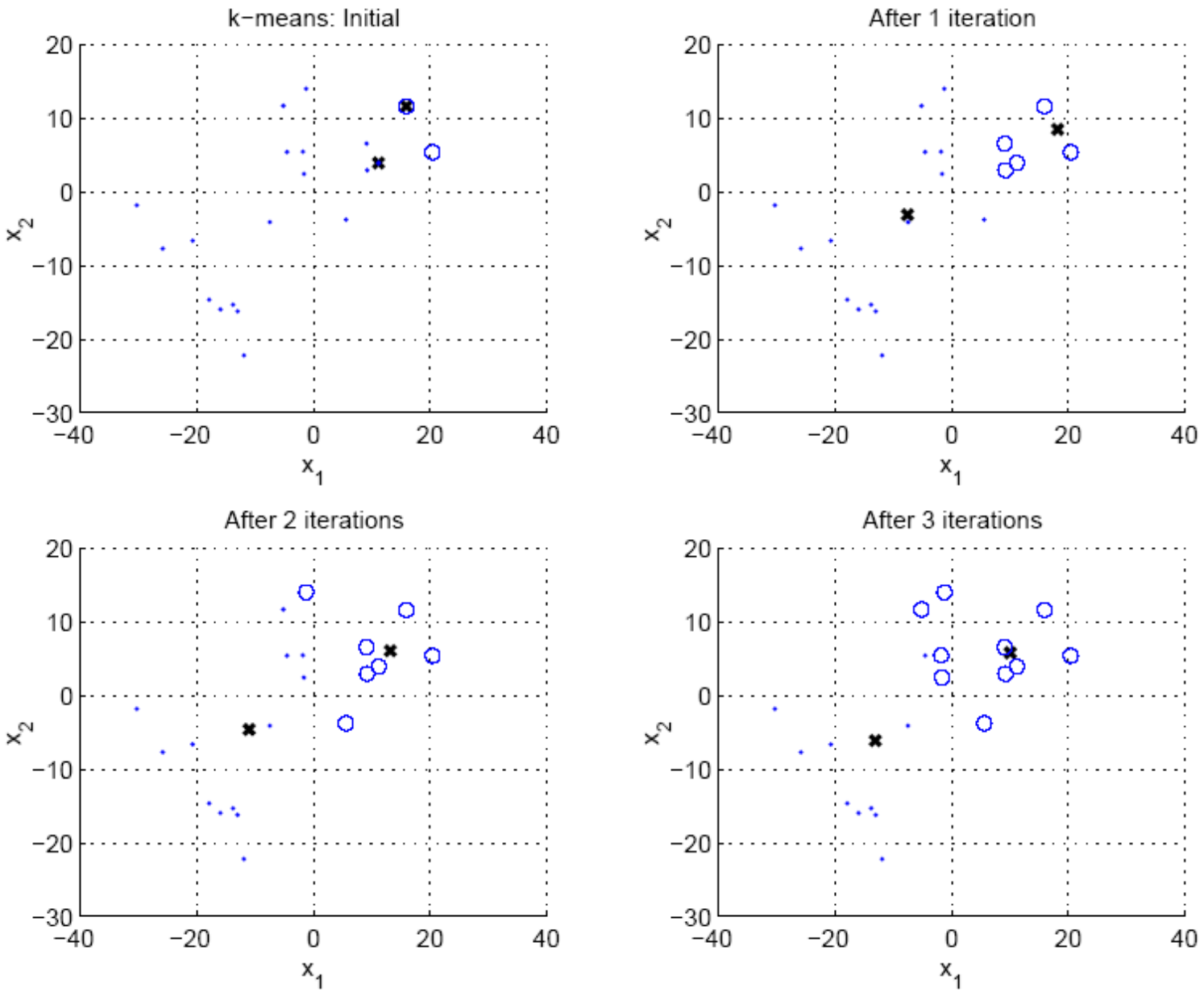
$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

centroid adjustment

Until $\boldsymbol{m}_i$ converge

What is the convergence criteria?

(optimization with respect to $\boldsymbol{m}_i$, or find the "center" in cluster i)



Animation: http://shabal.in/visuals/kmeans/1.html
http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html

Note:(1) *k*-mean is "iterative" + it's a heuristic (in *k* and **initial selection**),
(2) other methods to select initial point, (3) leader cluster algorithm

From K-mean to EM

- **K-mean in practice:**
  - Start with 100 (or some fraction of *N*) initial centroids
  - Run the K-mean algorithm for each initial centroids
  - Select the solution with the ***lowest error***
- Some important observations on K-mean
  - It is iterative between two phases: ***cluster assignment*** and ***centroid adjustment***
  - In the cluster assignment, we **ASSUME** the knowledge of centroids.
  - In the centroid adjustment, we **UPDATE** the centroid
- **Expectation Maximization (EM)** can be viewed as a generalization of K-mean

Intuitive Illustration of EM Algorithm

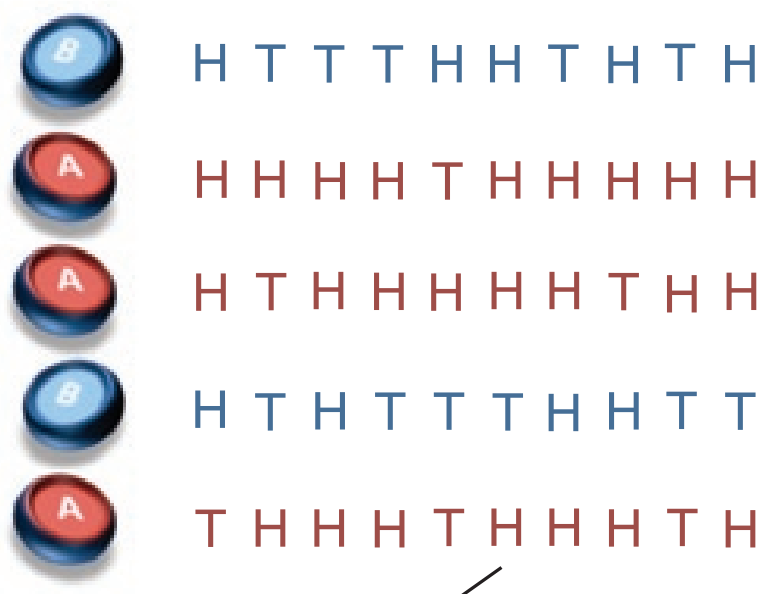
- **Coin flipping experiment:**
  - A pair of coins, *A* and *B*, of unknown biases, θ_A and θ_B (*or θ_A is the probability of seeing a head in coin A, similarly, θ_B is the probability of seeing a head in coin B*)
  - Our goal, **estimate** $\theta = (\theta_A, \theta_B)$ via the following procedure **five times:**
    - *Randomly pick one of the two coins*
    - *Perform ten independent coin tosses with the selected coin*
  - The entire procedure involves a total of 50 coin tosses
  - Keep track of two vectors $\boldsymbol{x}=(x_1,\ldots,x_5)$ and $\boldsymbol{z} = (z_1,..z_5)$
    - x_i in {0,1,…,10}, # of heads in the i^{th} experiment
    - z_i in {A,B}, coin we selected in the i^{th} experiment

Intuitive Illustration of EM Algorithm

$\log P(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta})$ is the log of the joint probability

find $\hat{\boldsymbol{\theta}} = (\hat{\theta}_A, \hat{\theta}_B)$ to maximize $\log P(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta})$

a Maximum likelihood



Coin A	Coin B
	5 H, 5 T
9 H, 1 T	
8 H, 2 T	
	4 H, 6 T
7 H, 3 T	
24 H, 6 T	9 H, 11 T

MLE

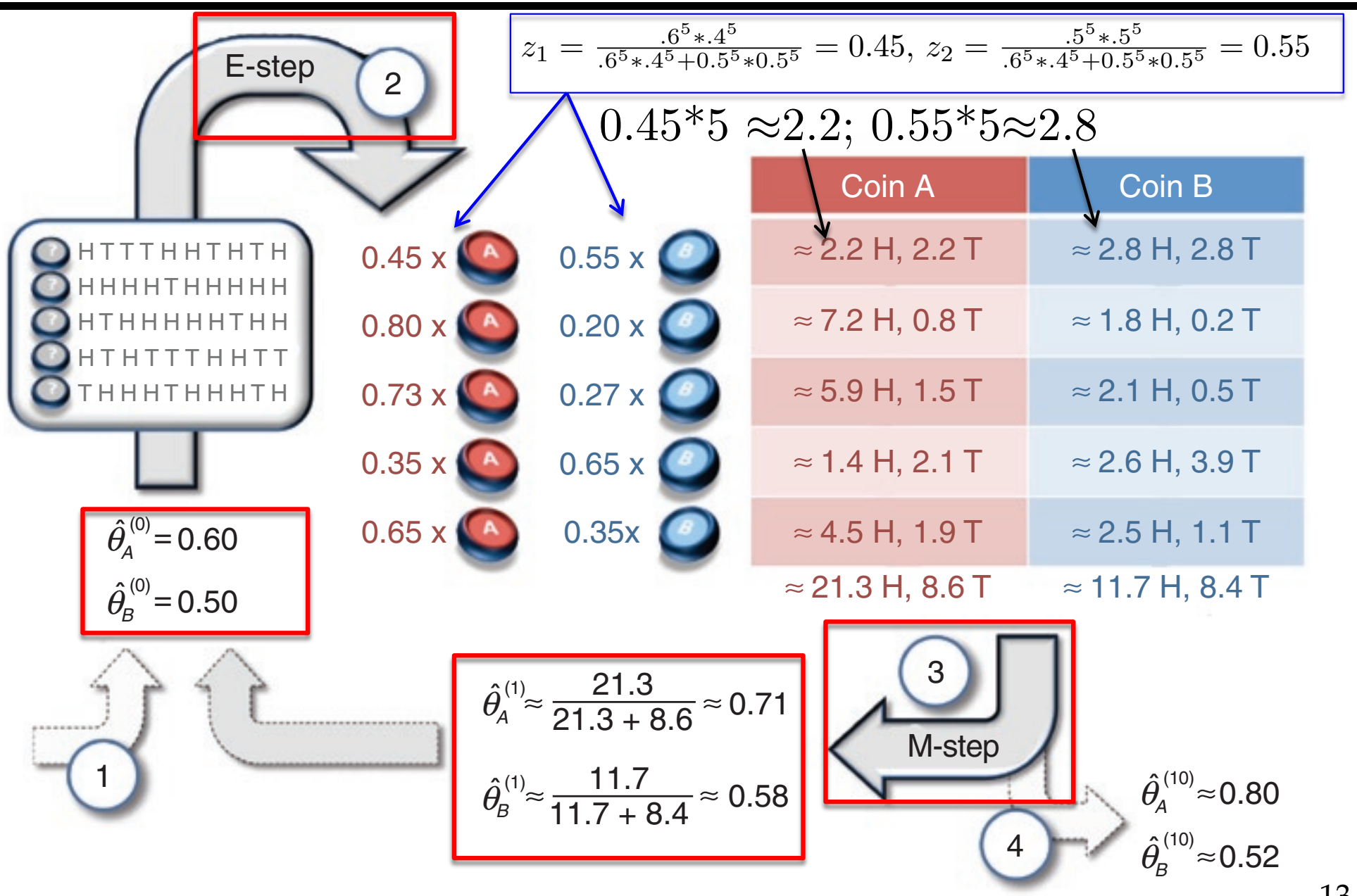
$$\hat{\theta}_A = \frac{24}{24+6} = 0.80$$

$$\hat{\theta}_B = \frac{9}{9+11} = 0.45$$

Intuitive Illustration of EM Algorithm

- A more challenging variant, we have x but not z (*latent factor or hidden variables*), or we have **incomplete data**
- We can use the **EM algorithm**, an ***iterative*** scheme:
  - Start with some initial parameters $\hat{\theta}^{(t)} = (\hat{\theta}_A^{(t)}, \hat{\theta}_B^{(t)})$
  - Determine for each five sets whether coin A or B *was more likely* to have generated the flips
  - Assume these coin assignments to be correct, apply the regular *maximum likelihood estimation procedure* to get $\hat{\theta}^{(t+1)}$
  - Repeat until convergence

Intuitive Illustration of EM Algorithm



Intuitive Illustration of EM Algorithm

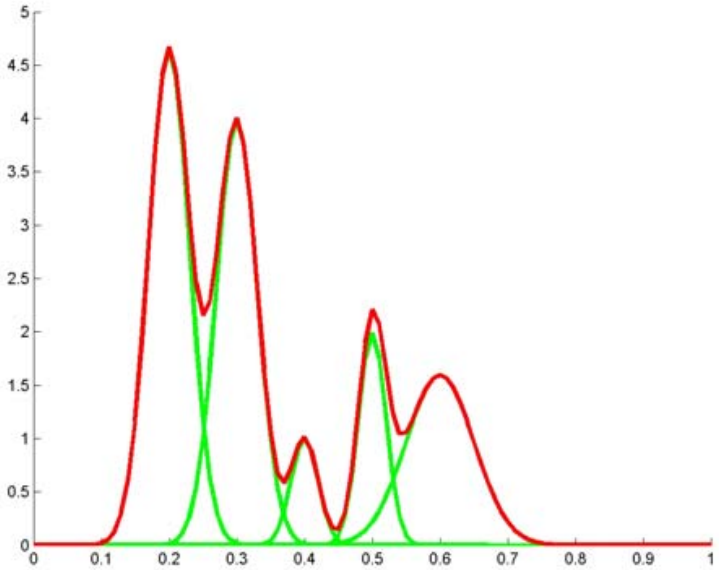
- The previous example illustrates:
  - Under the ***incomplete information environment***, how we can still *estimate parameters:* $\boldsymbol{\theta} = (\theta_A, \theta_B)$
  - The parameters are Bernoulli random variables
- We can extend the previous example to
  - Estimate parameters of **multinomial random variables**, e.g., outcome can take on *1* of the possible *k* > *1* possibilities
  - Estimate parameters for more than two groups, e.g.,

$$\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_n)$$

- Note that we only have the "***superposition***" of *k groups* (or incomplete information) and we want to find the parameters of all these *k groups*

Motivation of Expectation Maximization

- Another example is that our input is a ***mixture*** of ***different*** probability density functions (previous example was a mixture of Bernoulli RVs)
- **Example:** *mixture of Gaussian*



- EM helps us to ***estimate parameters*** of **each Gaussian**
- Can be a powerful to extend the parametric method

Expectation-Maximization (EM)

- EM looks for component density parameter to maximize the likelihood of the samples
- Log likelihood with a mixture model

$$\mathcal{L}(\Phi|\mathcal{X}) \quad = \quad \log \prod_t p(\boldsymbol{x}^t|\Phi) = \quad \sum_t \log \sum_{i=1}^{k} p(\boldsymbol{x}^t|\mathcal{G}_i) P(\mathcal{G}_i)$$

- Assume hidden variables Z, which when known, make optimization much simpler (*similar to centroids in K-mean*)
- Complete likelihood, $\mathcal{L}_c(\Phi|\mathcal{X}, \mathcal{Z})$, in terms of X and Z
- Incomplete likelihood, $\mathcal{L}(\Phi|\mathcal{X})$, in terms of X (work on this)
- In the case of mixtures, the hidden variables **Z**, are the "sources" (or component) of observations (or which observation belongs to which component)

Iteration on E-steps and M-steps

Iterate the two steps, for iteration l

1. **E-step**: Estimate **Z** given **X** and current Φ^l
2. **M-step**: Find new Φ^{l+1} given **Z**, **X**, and old Φ^l

$$\text{E-step} \quad : \quad \mathcal{Q}(\Phi|\Phi^l) = E[\mathcal{L}_c(\Phi|\mathcal{X}, \mathcal{Z})|\mathcal{X}, \Phi^l]$$
$$\text{M-step} \quad : \quad \Phi^{l+1} = \arg\max_{\Phi} \mathcal{Q}(\Phi|\Phi^l)$$

An increase in $\mathcal{Q}$ implies an increases incomplete likelihood

$$\mathcal{L}(\Phi^{l+1}|\mathcal{X}) \geq \mathcal{L}(\Phi^l|\mathcal{X})$$

E-step: estimate "labels" given samples and component
M-step: update component given labels

EM in Gaussian Mixtures

- Define *indicator variables* $\boldsymbol{z}^t = \{z_1^t, \ldots, z_k^t\}$ where $z_i^t = 1$ if $\boldsymbol{x}^t$ belongs to cluster G_i, and 0 otherwise
- Define z_i^t is like labels r_i^t of supervised learning;
- **z** is a multinomial distribution from *k* categories with prior probability π_i, shorthand for $P(G_i)$
- We have: $$P(\boldsymbol{z}^t) = \prod_{i=1}^{k} \pi_i^{z_i^t}$$
- The likelihood of an observation $\boldsymbol{x}^t$ is

$$p(\boldsymbol{x}^t|\boldsymbol{z}^t) = \prod_{i=1}^{k} p(\boldsymbol{x}^t|G_i)^{z_i^t} \quad \text{and} \quad p(\boldsymbol{x}^t, \boldsymbol{z}^t) = P(\boldsymbol{z}^t)p(\boldsymbol{x}^t|\boldsymbol{z}^t)$$

EM in Gaussian Mixtures

- Complete data likelihood of the IID samples $\mathcal{X}$ is:

$$P(AB|C) = \frac{P(ABC)}{P(C)}$$

$$= \frac{P(A|BC)P(BC)}{P(C)}$$

$$= P(A|BC)P(B|C)$$

$$\mathcal{L}_c(\Phi|\mathcal{X},\mathcal{Z}) = \log \prod_t p(\boldsymbol{x}^t, \boldsymbol{z}^t|\Phi)$$

$$= \sum_t \log p(\boldsymbol{x}^t, \boldsymbol{z}^t|\Phi)$$

$$= \sum_t \log P(\boldsymbol{z}^t|\Phi) + \log p(\boldsymbol{x}^t|\boldsymbol{z}^t,\Phi)$$

$$= \sum_t \sum_i z_i^t [\log \pi_i + \log p_i(\boldsymbol{x}^t|\Phi)]$$

EM in Gaussian Mixtures

- **E-step:** Let's define

$$\begin{aligned}
\mathcal{Q}(\Phi|\Phi^l) &\equiv E\left[\log P(X,Z)|\mathcal{X},\Phi^l\right] \\
&= E\left[\mathcal{L}_c(\Phi|\mathcal{X},\mathcal{Z})|\mathcal{X},\Phi^l\right] \\
&= \sum_t \sum_i E[z_i^t|\mathcal{X},\Phi^l][\log \pi_i + \log p_i(\boldsymbol{x}^t|\Phi^l)]
\end{aligned}$$

where

$$\begin{aligned}
E[z_i^t|\mathcal{X},\Phi^l] &= E[z_i^t|\boldsymbol{x}^t,\Phi^l] \quad \boldsymbol{x}^t \text{ are iid} \\
&= P(z_i^t=1|\boldsymbol{x}^t,\Phi^l) \quad z_i^t \text{ is a } 0/1 \text{ random variable} \\
&= \frac{p(\boldsymbol{x}^t|z_i^t=1,\Phi^l)P(z_i^t=1|\Phi^l)}{p(\boldsymbol{x}^t|\Phi^l)} \quad \text{Bayes' rule} \\
&= \frac{p_i(\boldsymbol{x}^t|\Phi^l)\pi_i}{\sum_j p_j(\boldsymbol{x}^t|\Phi^l)\pi_j}
\end{aligned}$$

Supplementary note

Derivation of the Bayes' rule in last page:

$$
\begin{aligned}
P(A|BC) &= \frac{P(ABC)}{P(BC)} \\
&= \frac{P(B|AC)P(AC)}{P(B|C)P(C)} \\
&= \frac{P(B|AC)P(A|C)P(C)}{P(B|C)P(C)} \\
&= \frac{P(B|AC)P(A|C)}{P(B|C)}
\end{aligned}
$$

EM in Gaussian Mixtures

- **E-step:**

$$E[z_i^t|\mathcal{X},\Phi^l] = \frac{p_i(\boldsymbol{x}^t|\Phi^l)\pi_i}{\sum_j p_j(\boldsymbol{x}^t|\Phi^l)\pi_j} \equiv h_i^t$$

Note: h_i^t is like the average probability that $\boldsymbol{x}^t$ is from source $\mathcal{G}_i$

- *This completes the* **Expectation Step (E-step)**

$$\text{Note that } h_1^t + h_2^t + \cdots + h_k^t = \sum_{i=1}^k h_i^t = 1$$

EM in Gaussian Mixtures: M-step

- **M-step:** maximize $\mathcal{Q}$ to get the next set of parameter values Φ^{l+1}:

$$\Phi^{l+1} = \arg\max_{\Phi} \mathcal{Q}(\Phi|\Phi^l)$$

- which is: $\mathcal{Q}(\Phi|\Phi^l) = \sum_t \sum_i h_i^t [\log \pi_i + \log p_i(\boldsymbol{x}^t|\Phi^l)]$

$$= \sum_t \sum_i h_i^t \log \pi_i + \sum_t \sum_i h_i^t \log p_i(\boldsymbol{x}^t|\Phi^l)$$

- We optimize with π_i. Note that the 2nd term is **not** function of π_i and sum of all π_i is 1, we have

$$\nabla_{\pi_i} \sum_t \sum_i h_i^t \log \pi_i - \lambda \left(\sum_i \pi_i - 1 \right) = 0$$

$$\text{get} \quad \pi_i = \frac{\sum_t h_i^t}{N}$$

EM in Gaussian Mixtures: M-step

- The first part of $\mathcal{Q}(\Phi|\Phi^l)$ is independent of Φ^l, we have

$$\nabla_\Phi \sum_t \sum_i h_i^t \log p_i(\boldsymbol{x}^t|\Phi) = 0$$

- **Assuming Gaussian component,** $\hat{p}_i(\boldsymbol{x}^t|\Phi) \sim \mathcal{N}(\boldsymbol{m}_i, \mathbf{S}_i)$

$$\boldsymbol{m}_i^{l+1} = \frac{\sum_t h_i^t \boldsymbol{x}^t}{\sum_t h_i^t}$$

$$\mathbf{S}_i^{l+1} = \frac{\sum_t h_i^t (\boldsymbol{x}^t - \boldsymbol{m}_i^{l+1})(\boldsymbol{x}^t - \boldsymbol{m}_i^{l+1})^T}{\sum_t h_i^t}$$

This is similar to parameter estimation in Chapter 5

EM in Gaussian Mixtures: M-step

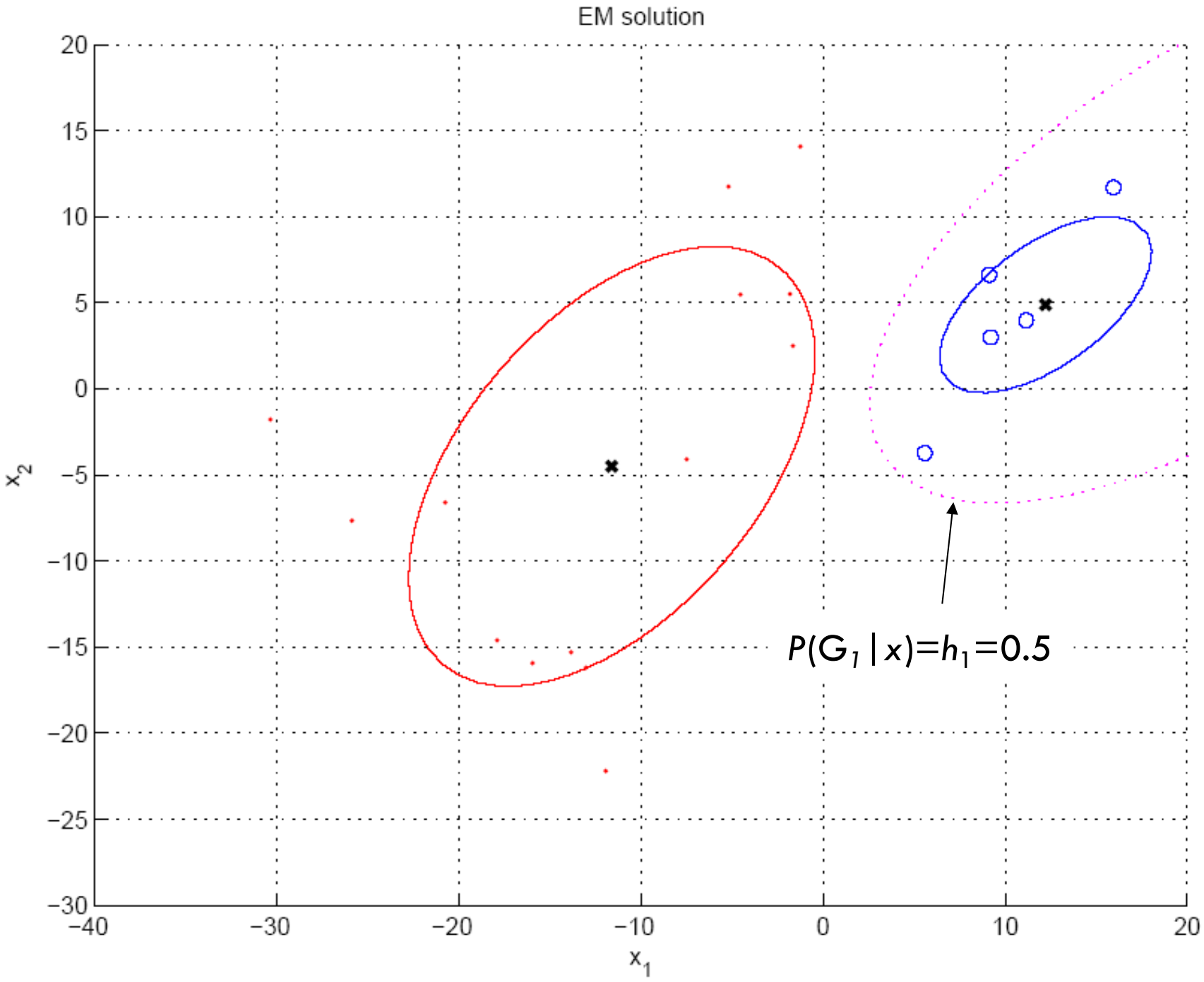
- Note that h_i^t is like a probability, and for Gaussian components, we have

$$h_i^t = \frac{\pi_i |\mathbf{S}_i|^{-1/2} \exp[-(1/2)(\boldsymbol{x}^t - \boldsymbol{m}_i)^T \mathbf{S}_i^{-1} (\boldsymbol{x}^t - \boldsymbol{m}_i)]}{\sum_j \pi_j |\mathbf{S}_j|^{-1/2} \exp[-(1/2)(\boldsymbol{x}^t - \boldsymbol{m}_j)^T \mathbf{S}_j^{-1} (\boldsymbol{x}^t - \boldsymbol{m}_j)]}$$

- How to initialize EM algorithm? We can use ***k-mean***.
- After few iterations of k-mean, we have estimate of

$$\boldsymbol{m}_i \quad \mathbf{S}_i \quad \pi_i$$

- Then we run EM from that point on
- Read book on shared covariance matrix (for different classes) and shared diagonal matrix (all *d* dimension have same variance)


EM solution
20
15
10
5
0
−5
−10
−15
−20
−25
−30
−40
−30
−20
−10
0
10
20
x2
x1
P(G1 | x)=h1=0.5

Mixtures of Latent Variable Models

- Previously, we used ***full covariance matrices***, but if the input dimensionality is high and the sample is small, we may over-fitting problem
- To reduce # of parameters, assuming common covariance matrix may be risky since clusters may have different *shape*.
- To reduce # of parameters, assuming diagonal matrices is risky because we are removing correlations
- One solution is to do dimensionality reduction in the clusters (to reduce # of parameters and still capture correlations)
- Use PCA/FA to decrease dimensionality: Mixtures of PCA/FA

$$p(\boldsymbol{x}^t|\mathcal{G}_i) \sim \mathcal{N}(\boldsymbol{m}_i, \mathbf{V}_i\mathbf{V}_i^T + \boldsymbol{\Psi}_i)$$

Can use EM to learn $\mathbf{V}_i$ (Ghahramani and Hinton, 1997; Tipping and Bishop, 1999)

Supervised Learning After Clustering

- **Dimensionality reduction** (Chapter 6) methods find correlations between features and group features
- **Clustering methods** (Chapter 7) find similarities between instances and group instances
- Allows knowledge extraction through
  - number of clusters,
  - prior probabilities,
  - cluster parameters, i.e., center, range of features

  Example:
  - Customer Relationship Management (CRM)
  - Customer Segmentation

Clustering as Preprocessing

- Estimated group labels h_i (soft) or b_i (hard) may be seen as the dimensions of a new k dimensional space, where we can then learn our discriminant or regressor
- Local representation (only one b_i is 1, all others are 0; only few h_i are nonzero) vs

  Distributed representation (After PCA; all z_i are nonzero)

Mixture of Mixtures

- In classification, the input comes from a mixture of classes (supervised)
- If each class is also a mixture, we have a *mixture of mixtures:*

$$p(\boldsymbol{x}|C_i) = \sum_{j=1}^{k_i} p(\boldsymbol{x}|\mathcal{G}_{ij})P(\mathcal{G}_{ij})$$

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

- Where k_i is the number of components making up $p(\boldsymbol{x}|\ C_i)$ and $\mathcal{G}_{ij}$ is the component j of class i

Spectral Clustering

- Cluster using predefined pairwise similarities B_{rs} instead of using Euclidean or Mahalanobis distance
- Can be used even if instances not vectorially represented
- Steps:
  - Use **Laplacian Eigenmaps** (chapter 6) to map to a new **z** space using B_{rs}
  - Use *k*-means in this new **z** space for clustering

Hierarchical Clustering

- Cluster based on similarities/distances
- Aim is to find groups s.t. instances within a group are more *similar* to each other than instances of other groups
- Distance measure between instances $\boldsymbol{x}^r$ and $\boldsymbol{x}^s$

Minkowski distance (L_p) (Euclidean for $p = 2$)

$$d_m(\boldsymbol{x}^r, \boldsymbol{x}^s) = \left[\sum_{j=1}^{d} (x_j^r - x_j^s)^p \right]^{1/p}$$

City-block distance

$$d_{cb}(\boldsymbol{x}^r, \boldsymbol{x}^s) = \sum_{j=1}^{d} |x_j^r - x_j^s|$$

Agglomerative / Divisive Clustering

- **Agglomerative clustering (AC):** Start with *N* groups each with one instance and merge two closest groups at each iteration
- **Divisive clustering (DC):** Start with one group and divide to many groups
- In AC, each iteration we *merge* two *closest groups*
- Distance between two groups $\mathcal{G}_i$ and $\mathcal{G}_j$:
  - **Single-link:**

$$d(\mathcal{G}_i, \mathcal{G}_j) = \min_{\boldsymbol{x}^r \in \mathcal{G}_i, \boldsymbol{x}^s \in \mathcal{G}_j} d(\boldsymbol{x}^r, \boldsymbol{x}^s)$$

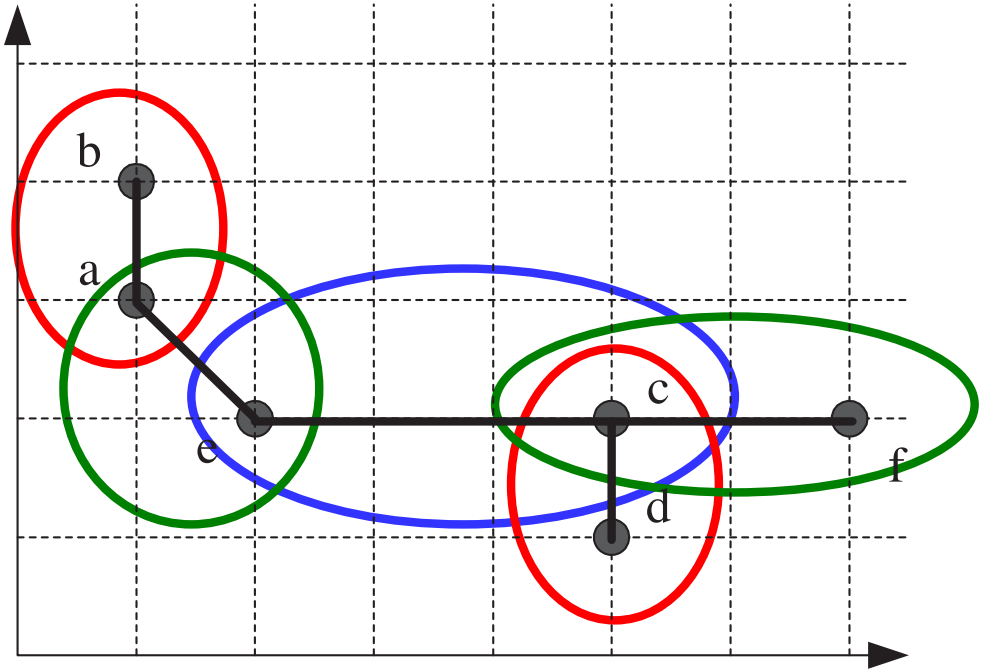
  - **Complete-link:**

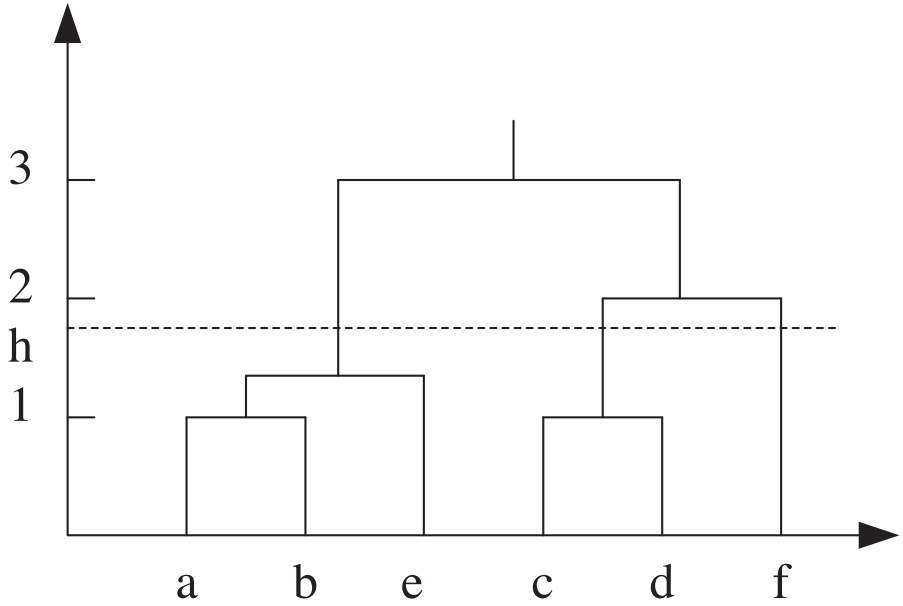
$$d(\mathcal{G}_i, \mathcal{G}_j) = \max_{\boldsymbol{x}^r \in \mathcal{G}_i, \boldsymbol{x}^s \in \mathcal{G}_j} d(\boldsymbol{x}^r, \boldsymbol{x}^s)$$

  - **Average-link, centroid** $d\left(\mathcal{G}_i, \mathcal{G}_j\right) = \underset{\mathbf{x}^r \in \mathcal{G}_i, \mathbf{x}^s \in \mathcal{G}_j}{\text{ave}} d\left(\mathbf{x}^r, \mathbf{x}^s\right)$

Example: Single-Link Clustering

$$d(\mathcal{G}_i, \mathcal{G}_j) = \min_{\boldsymbol{x}^r \in \mathcal{G}_i, \boldsymbol{x}^s \in \mathcal{G}_j} d(\boldsymbol{x}^r, \boldsymbol{x}^s)$$





Dendrogram

Choosing *k*

- Defined by the application, e.g., image quantization
- Plot data (after PCA) and check for clusters
- **Incremental (leader-cluster) algorithm**: Add one at a time until "elbow" (reconstruction error/log likelihood/intergroup distances)
- Manually check for meaning