Clustering

LIN Juntong, XU Ruijia, YU Le

Nov 14, 2016

Agenda

Introduction

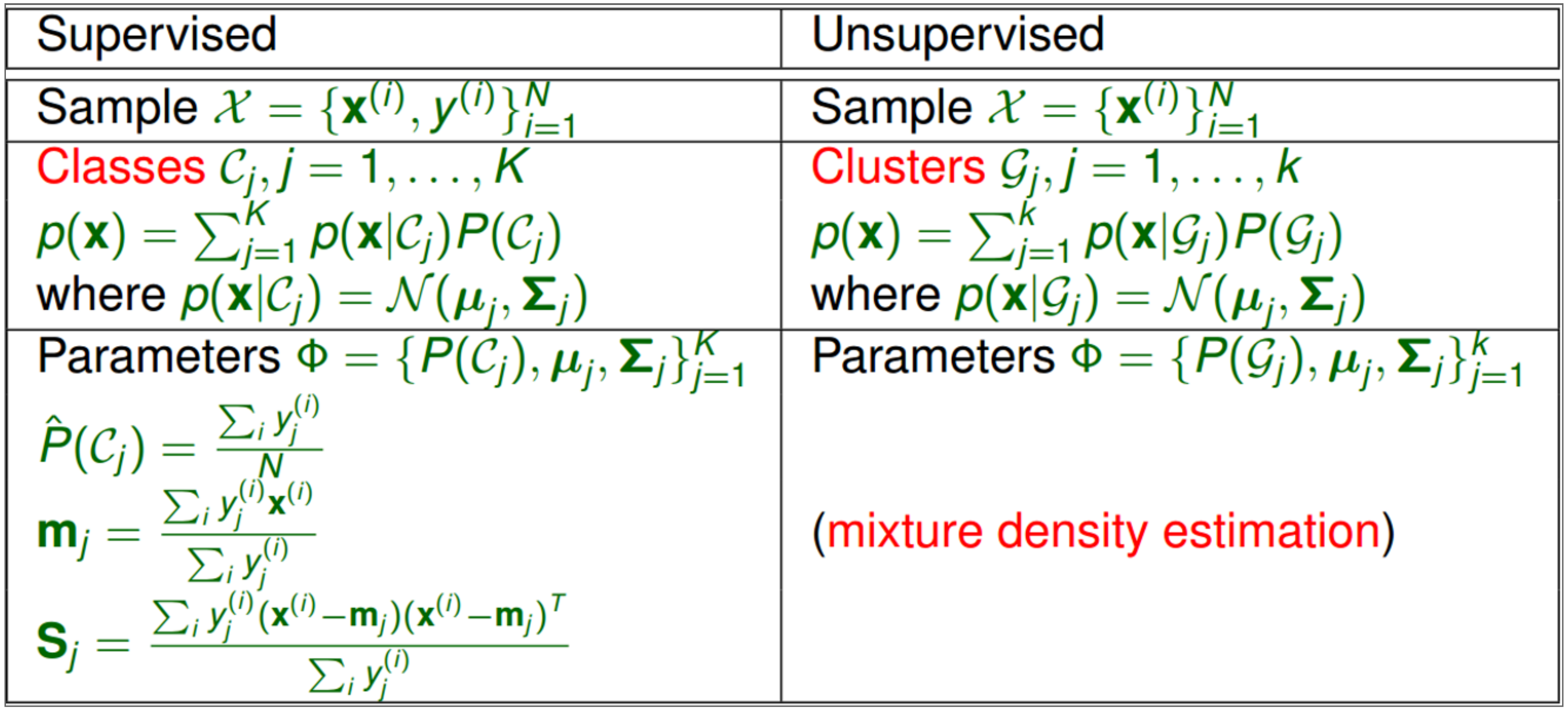
Different Approaches to Density Estimation

- What's density estimation
- Approaches to density estimation
  - Parametric: $p(\mathbf{x} \mid C_i)$ is represented by a single parametric model
  - Semiparametric: $p(\mathbf{x} \mid C_i)$ is represented by a mixture of densities
  - NonParametric: otherwise; the data speaks for itself (e.g. countless tiny models to fit all samples, analogious to LWR)
  - what's C_i ???, implicit label???
- The model flexibility increases (and hence the model bias decreases)
- Summary
  - How many different models used to represent $p(\mathbf{x} \mid C_i)$
  - Samples come from how many different distribution

Mixture Densities

- Mixture density
  $$p(\mathbf{x}) = \sum_{j=1}^{k} p(\mathbf{x}|\mathcal{G}_j)P(\mathcal{G}_j)$$
  where
  - $\mathcal{G}_j$: mixture components (or clusters or groups)
  - $p(\mathbf{x}|\mathcal{G}_j)$: component densities
  - $P(\mathcal{G}_j)$: mixing proportions (or priors)
- Gaussian mixture
  $p(\mathbf{x}|\mathcal{G}_j) = \mathcal{N}(\mu_j, \mathbf{\Sigma}_j)$
  - parameters: $\mathbf{\Phi} = \{P(\mathcal{G}_j), \mu_j, \mathbf{\Sigma}_j\}_{j=1}^{k}$

Classes vs. Clusters



Supervised	Unsupervised
Sample $\mathcal{X} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$	Sample $\mathcal{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$
Classes $\mathcal{C}_j, j = 1, \dots, K$ $p(\mathbf{x}) = \sum_{j=1}^{K} p(\mathbf{x} \mid \mathcal{C}_j) P(\mathcal{C}_j)$ where $p(\mathbf{x} \mid \mathcal{C}_j) = \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$	Clusters $\mathcal{G}_j, j = 1, \dots, k$ $p(\mathbf{x}) = \sum_{j=1}^{k} p(\mathbf{x} \mid \mathcal{G}_j) P(\mathcal{G}_j)$ where $p(\mathbf{x} \mid \mathcal{G}_j) = \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$
Parameters $\Phi = \{P(\mathcal{C}_j), \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^{K}$ $\hat{P}(\mathcal{C}_j) = \frac{\sum_i y_j^{(i)}}{N}$ $\mathbf{m}_j = \frac{\sum_i y_j^{(i)} \mathbf{x}^{(i)}}{\sum_i y_j^{(i)}}$ $\mathbf{S}_j = \frac{\sum_i y_j^{(i)} (\mathbf{x}^{(i)} - \mathbf{m}_j)(\mathbf{x}^{(i)} - \mathbf{m}_j)^T}{\sum_i y_j^{(i)}}$	Parameters $\Phi = \{P(\mathcal{G}_j), \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^{k}$ (mixture density estimation)

Unobserved variables

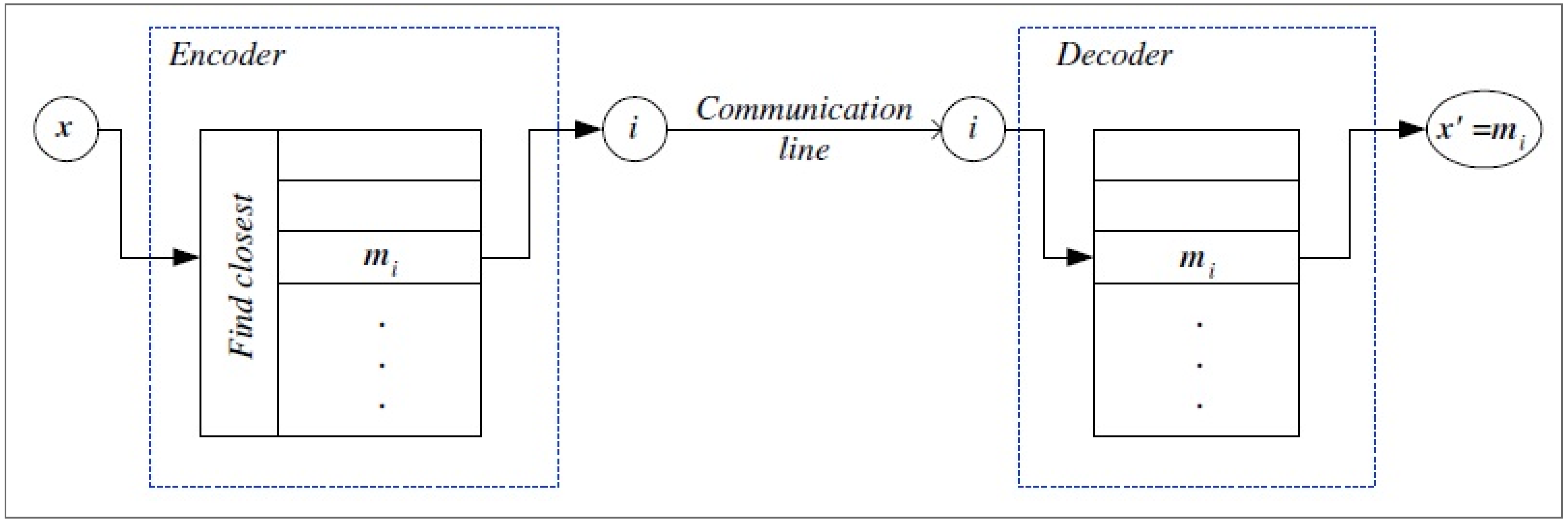
- Why a variable is unobserved (latent, unlabeled)
  - It is an **imaginary** quantity meant to provide some simplified and abstractive view of the data generation process. E.g., speech recognition models, mixture models
  - It is a real-world object and/or phenomena, but **difficult or impossible** to measure. E.g., the temperature of a star, causes of a disease, evolutionary ancestors
  - It is a real-world object and/or phenomena, but sometimes was **not measured**, because of faulty sensors; or was measure with a noisy channel, etc. E.g., traffic radio, aircraft signal on a radar screen
- **Discrete** latent variables can be used to **partition/cluster** data into sub-groups (mixture models, HMM in this lecture).
- **Continuous** latent variables can be used for **dimensionality reduction** (later lecture).

k-means

Clustering

- Problem formulation
  - Given a sample set $\mathcal{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$.
  - Find k reference vectors (or prototypes or codebook vectors or codewords) $\mathbf{m}_j$ $(j = 1, \ldots, k)$which best represent the data.
- Encoding/decoding view
  - Encoding: from a data point $\mathbf{x}^{(i)}$ to the index l of a reference vector.
  - Decoding: from an index l to the corresponding reference vector $\mathbf{m}_l$.

Illustration of Encoding and Decoding



Encoding/Decoding

- Each data points $\mathbf{x}^{(i)}$ is represented by the index l of the **nearest reference vector**:
  $l = \arg\min_j \|\mathbf{x}^{(i)} - \mathbf{m}_j\|$
- Encoding can lead to **data compression**: instead of storing (or transmitting) $\mathbf{x}^{(i)}$, we only need to store (or transmit) l.
- Since $\mathbf{x}^{(i)}$ is represented by $\mathbf{m}_l$ after encoding and then decoding, reconstruction error is incurred.
- **Total reconstruction error**:
  $$E(\{\mathbf{m}_l\}_{l=1}^{k} | \mathcal{X}) = \sum_i \sum_l b_l^{(i)} \|\mathbf{x}^{(i)} - \mathbf{m}_l\|^2$$
  where
  $$b_l^{(i)} = \begin{cases} 1 & \text{if } l = \arg\min_j \|x^{(i)} - \mathbf{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$
  which means whether $\mathbf{x}^{(i)}$ belongs to cluster l

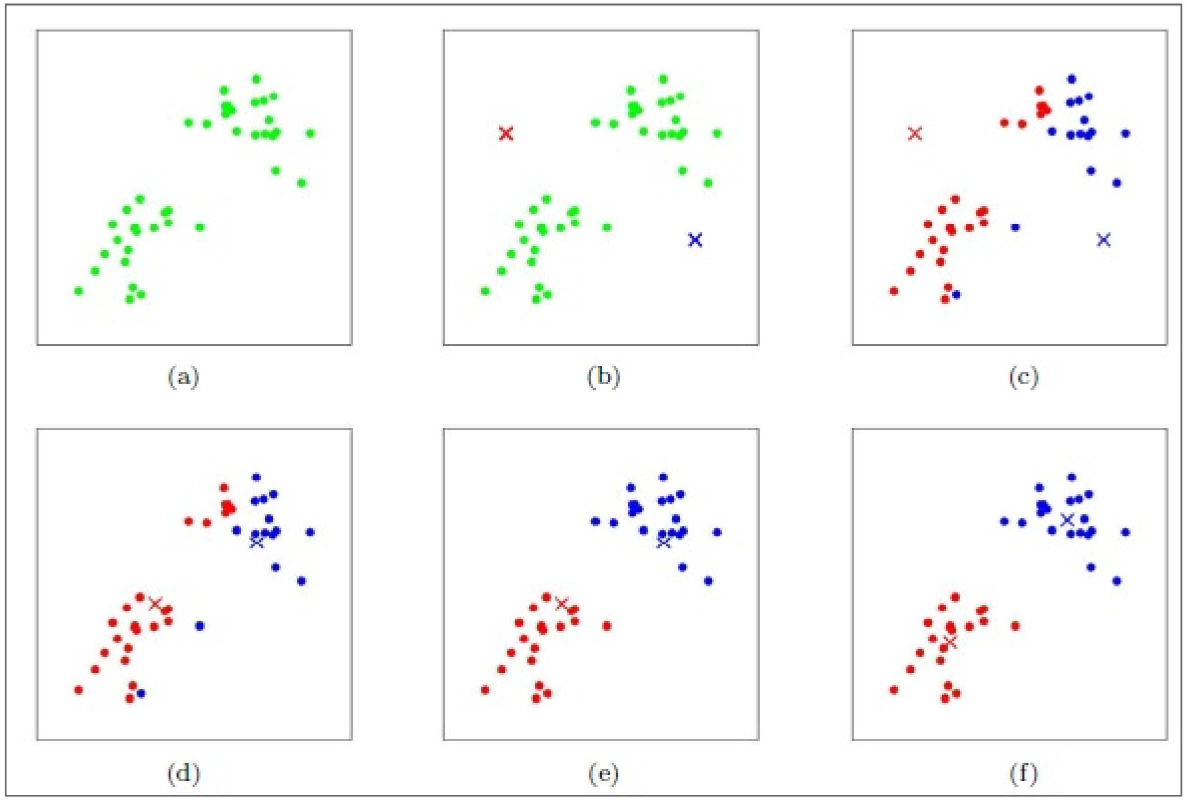
Optimization Problem

- The best reference vectors are those that minimize the total reconstruction error, so this corresponds to an optimization problem.
- However, since $b_l^{(i)}$ also depends on $\mathbf{m}_l$, the optimization problem cannot be solved analytically, but **iteratively**.
- The **k-means clustering algorithm** is an iterative algorithm for solving the optimization problem.

- k-Means Algorithm

```
**Initialize** $\mathbf{m}_l, l = 1, \ldots, k$ (e.g., $k$ randomly selected $\mathbf{x}^{(i)}$)
**Repeat**
    For all $\mathbf{x}^{(i)} \in \mathcal{X}$
        $b_l^{(i)} = \begin{cases} 1 & \text{if } l = \arg\min_j \|\mathbf{x}^{(i)} - \mathbf{m}_j\| \\ 0 & \text{otherwise} \end{cases}$
    For all $\mathbf{m}_l, l = 1, \ldots, k$
        $\mathbf{m}_l = \frac{\sum_i b_l^{(i)} \mathbf{x}^{(i)}}{\sum_i b_l^{(i)}}$
**Until** $\mathbf{m}_l$ converge.
```

Evolution of k-Means



Convergence of k-Means

- k-means is exactly **coordinate descent** on the reconstruction error E.
- E monotonically decreases, and the value of E converges, so do the clustering results.
- It is possible for k-means to oscillate between a few different clusterings, but this almost never happens in practice.
- E is **non-convex**, so coordinate descent on E **cannot guaranteed** to converge to **global minimum**. One common thing to do is running k-means many times and pick the best one.

Gaussian Mixture Models and EM Algorithm

Gaussian Mixture Models

- Consider a mixture of k Gaussian components:
  - The **latent class indicator vector** $\mathbf{z}^{(i)} = (z_1^{(i)}, \ldots, z_k^{(i)})^T$:
    $$z_l^{(i)} \begin{cases} 1 & \text{if } \mathbf{x}^{(i)} \text{ belones to cluster } \mathcal{G}_l \\ 0 & \text{otherwise} \end{cases}$$
  - The **likelihood** of a sample $\mathbf{x}$:
    $$\begin{aligned} p(\mathbf{x}|\mathbf{\Phi}) &= \sum_{l=1}^{k} p(z_l = 1|\mathbf{\Phi}) p(\mathbf{x}|z_l = 1, \mathbf{\Phi}) \\ &= \sum_{l} \pi_l \mathcal{N}(\mathbf{x}|\mu_l, \mathbf{\Sigma}_l) \end{aligned}$$
    - **Gaussian component densities**: $p(\mathbf{x}|\mathcal{G}_l) = \mathcal{N}(\mu_l, \mathbf{\Sigma}_l)$parameter $\Theta_l = \{\mu_l, \mathbf{\Sigma}_l\}$
    - The **prior** probability: $P(\mathcal{G}_l) = \pi_l$

Different between supervised and unsupervised

- In **fully observed** IID settings, the log likelihood decomposes into a **sum of local terms**.

$$\mathcal{L}(\mathbf{\Phi}|\mathcal{X},\mathcal{Z}) = \sum_i \log p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{\Phi}) = \sum_i (\log p(\mathbf{z}^{(i)}|\mathbf{\Phi}_\mathbf{z}) + \log p(\mathbf{x}^{(i)}|\mathbf{z}^{(i)},\mathbf{\Phi}_\mathbf{x}$$

- With **latent variables**, all the parameters become **coupled** together via **marginalization**.
$$\mathcal{L}(\mathbf{\Phi}|\mathcal{X}) = \sum_i \log \sum_{\mathbf{z}^{(i)}} p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{\Phi})$$

Expectation-Maximization Algorithm

- EM finds the component density parameters that maximize the likelihood with respect to a mixture model.
- **Log likelihood**:

  $$\mathcal{L}(\mathbf{\Phi}|\mathcal{X}) = \log \prod_i p(\mathbf{x}^{(i)}|\mathbf{\Phi}) = \sum_i \log \sum_{l=1}^{k} p(\mathbf{x}^{(i)}|\mathcal{G}_l) P(\mathcal{G}_l)$$

  where the parameters $\mathbf{\Phi}$ include the priors $P(\mathcal{G}_l)$ and the sufficient statistics of the component densities $p(\mathbf{x}^{(i)}|\mathcal{G}_l)$.
- **Optimization** of $\mathcal{L}(\mathbf{\Phi}|\mathcal{X})$w.r.t. $\mathbf{\Phi}$ cannot be solved analytically.

Iterative Optimization

- EM is an **iterative** algorithm for solving the maximum likelihood estimation (MLE) problem.
- EM is suitable for problems that involves two sets of random variables: **observation variables $\mathbf{x}$** and **hidden variables $\mathbf{z}$**.
- Goal: maximize the likelihood $\mathcal{L}(\mathbf{\Phi}|\mathcal{X})$given $\mathcal{X}$.
- EM is suitable if it is difficult to maximizing the incomplete-data likelihood $\mathcal{L}(\mathbf{\Phi}|\mathcal{X})$directly but it is easier to work with the complete-data likelihood $\mathcal{L}_C(\mathbf{\Phi}|\mathcal{X},\mathcal{Z})$
- Because the $\mathbf{z}$ value are not observed, we cannot work directly with $\mathcal{L}_C(\mathbf{\Phi}|\mathcal{X},\mathcal{Z})$ Instead, we work with an **auxiliary function** which is the expectation of the complete-data likelihood given $\mathcal{X}$ and the current (iteration t) parameter values $\mathbf{\Phi}^t$:
  $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t) = E[\mathcal{L}_C(\mathbf{\Phi}|\mathcal{X},\mathcal{Z})|\mathcal{X},\mathbf{\Phi}^t]$

E-Step and M-Step

- EM iterates between two steps:
  - **E-step**: evaluation of **expectation** $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t) = E[\mathcal{L}_C(\mathbf{\Phi}|\mathcal{X}, \mathcal{Z})|\mathcal{X}, \mathbf{\Phi}^t]$
  - **M-step**: **maximization** of expectation $\mathbf{\Phi}^{t+1} = \arg\max_{\mathbf{\Phi}} \mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$
- k-means can be used to **initialize** EM. ???
- An increase in $\mathcal{Q}$ implies an increase in the incomplete-data likelihood: $\mathcal{L}(\mathbf{\Phi}^{t+1}|\mathcal{X}) \geq \mathcal{L}(\mathbf{\Phi}^t|\mathcal{X})$
- EM finds a **local maximum** of the likelihood.

Jensen's Inequality

- Jensen's Inequality:
  Let f be a **convex** function and X a random variable. Then,
  $E[f(X)] \geq f(E[X])$.
  Moreover, if f is **strictly convex**, then $E[f(X)] = f(E[X])$holds true if and only if $X = E[X]$with probability 1 (i.e., X is a constant).
  - Jensen's inequality also holds for **concave** functions f, but with reversed inequalities, i.e., $E[f(X)] \leq f(E[X])$.
  - f is (strictly) concave if and only if $-f$ is (strictly) convex.

Jensen's Inequality in Likelihood Function

- Let $\mathcal{Q}$ be some distribution over $\mathbf{z}$'s ($\sum_{\mathbf{z}} \mathcal{Q}(\mathbf{z}) = 1$and $\mathcal{Q}(\mathbf{z}) \geq 0$), we have:

$$\begin{aligned} \mathcal{L}(\mathbf{\Phi}|\mathcal{X}) =& \sum_i \log p(x^{(i)}|\mathbf{\Phi}) = \sum_i \log \sum_{\mathbf{z}^{(i)}} p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi}) \\ =& \sum_i \log \sum_{\mathbf{z}^{(i)}} \mathcal{Q}(\mathbf{z}^{(i)}) \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi})}{\mathcal{Q}(\mathbf{z}^{(i)})} \\ \geq& \sum_i \sum_{\mathbf{z}^{(i)}} \mathcal{Q}(\mathbf{z}^{(i)}) \log \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi})}{\mathcal{Q}(\mathbf{z}^{(i)})} \end{aligned}$$

  - $f(x) = \log x$is a concave function
  - $E_{\mathbf{z}}[\frac{p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{\Phi})}{\mathcal{Q}(\mathbf{z}^{(i)})}] = \sum_{\mathbf{z}^{(i)}} \mathcal{Q}(\mathbf{z}^{(i)}) \left(\frac{p(\mathbf{x}^{(i)},\mathbf{z}^{(i)}|\mathbf{\Phi})}{\mathcal{Q}(\mathbf{z}^{(i)})}\right)$
- The above formula gives a **lower-bound on the likelihood** $\mathcal{L}(\mathbf{\Phi}|\mathcal{X})$.

The Choice of Auxiliary Function

- Choose $\mathcal{Q}$ to make the lower-bound tight at some value of $\mathbf{\Phi}$, i.e., make the inequality above hold with equality at current particular value of $\mathbf{\Phi}$.
- To make the Jensen's inequality hold with equality, it is sufficient that the expectation is taken over a constant variable, i.e.,
$$\frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} | \mathbf{\Phi})}{\mathcal{Q}(\mathbf{z}^{(i)})} = C$$
- Since $\sum_{\mathbf{z}^{(i)}} \mathcal{Q}(\mathbf{z}^{(i)}) = 1$, we can choose:
$$\begin{aligned}\mathcal{Q}(\mathbf{z}^{(i)}) &= \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} | \mathbf{\Phi})}{\sum_{\mathbf{z}^{(i)}} p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} | \mathbf{\Phi})} \\ &= \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} | \mathbf{\Phi})}{p(\mathbf{x}^{(i)} | \mathbf{\Phi})} \\ &= p(\mathbf{z}^{(i)} | \mathbf{x}^{(i)}, \mathbf{\Phi})\end{aligned}$$

The Convergence of EM (1)

- Let $\mathbf{\Phi}^t$ and $\mathbf{\Phi}^{t+1}$ be the parameters from two successive iterations of EM.
- We have chosen $\mathcal{Q}^t(\mathbf{z}^{(i)}) = p(\mathbf{z}^{(i)}|\mathbf{x}^{(i)}, \mathbf{\Phi}^t)$ to ensure the Jensen's equality:

$$\mathcal{L}(\mathbf{\Phi}^t|\mathcal{X}) = \sum_i \sum_{\mathbf{z}^{(i)}} \mathcal{Q}^t(\mathbf{z}^{(i)}) \log \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi}^t)}{\mathcal{Q}^t(\mathbf{z}^{(i)})}$$

The Convergence of EM (2)

- $\mathbf{\Phi}^{t+1}$ is then obtained by maximizing the right hand side of the Equation of the last page. Thus,

$$\begin{aligned}\mathcal{L}(\mathbf{\Phi}^{t+1}|\mathcal{X}) &\geq \sum_i \sum_{\mathbf{z}^{(i)}} \mathcal{Q}^t(\mathbf{z}^{(i)}) \log \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi}^{t+1})}{\mathcal{Q}^t(\mathbf{z}^{(i)})} \\ &\geq \sum_i \sum_{\mathbf{z}^{(i)}} \mathcal{Q}^t(\mathbf{z}^{(i)}) \log \frac{p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi}^{t})}{\mathcal{Q}^t(\mathbf{z}^{(i)})} \\ &= \mathcal{L}(\mathbf{\Phi}^t|\mathcal{X})\end{aligned}$$

- First Equation: Jensen's inequality hold for all $\mathcal{Q}$ and $\mathbf{\Phi}$
- Second Equation: $\mathbf{\Phi}^{t+1}$ is chosen to maximize r.h.s. of Equation on the last page

- **monotonically converge**

EM for Gaussian Mixtures (1)

- Indicator variables $\mathbf{z}^{(i)} = (z_1^{(i)}, \ldots, z_k^{(i)})^T$:
  $$z_l^{(i)} = \begin{cases} 1 & \text{if } \mathbf{x}^{(i)} \text{ belones to cluster } \mathcal{G}_l \\ 0 & \text{otherwise} \end{cases}$$
- Gaussian component densities:
  $p(\mathbf{x}|\mathcal{G}_l) = \mathcal{N}(\mu_l, \mathbf{\Sigma}_l)$
- Let the prior probabilities $P(\mathcal{G}_l) = \pi_l$, so
  $$P(\mathbf{z}^{(i)}) = \prod_{l=1}^{k} \pi_l^{z_l^{(i)}}$$

EM for Gaussian Mixtures (2)

- Probability of observed variable given hidden variable:
  $$p(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) = \prod_{l=1}^{k}[p_l(\mathbf{x}^{(i)})]^{z_l^{(i)}}$$
  where $p_l(\mathbf{x}^{(i)})$ is the shorthand for $p(\mathbf{x}^{(i)}|G_l)$.
- Joint density: $p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}) = P(\mathbf{z}^{(i)})p(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})$
- **Complete-data log likelihood**:
  $$\begin{aligned}\mathcal{L}_C(\mathbf{\Phi}|\mathcal{X},\mathcal{Z}) &= \log\prod_i p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi}) = \sum_i \log p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}|\mathbf{\Phi}) \\ &= \sum_i [\log p(\mathbf{z}^{(i)}|\mathbf{\Phi}) + \log p(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}, \mathbf{\Phi})] \\ &= \sum_i \sum_l z_l^{(i)} [\log \pi_l + \log p_l(\mathbf{x}^{(i)}|\mathbf{\Phi})]\end{aligned}$$

E-Step for Gaussian Mixtures (1)

- **Evaluation** of $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$:

$$\begin{aligned}\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t) &= E[\mathcal{L}_C(\mathbf{\Phi}|\mathcal{X},\mathcal{Z})|\mathcal{X},\mathbf{\Phi}^t] \\ &= \sum_i \sum_l E[z_l^{(i)}|\mathcal{X},\mathbf{\Phi}^t][\log \pi_l + \log p_l(\mathbf{x}^{(i)}|\mathbf{\Phi})]\end{aligned}$$

where

$$\begin{aligned}E[z_l^{(i)}|\mathcal{X},\mathbf{\Phi}^t] &= E[z_l^{(i)}|\mathbf{x}^{(i)},\mathbf{\Phi}^t] = P(z_l^{(i)}=1|\mathbf{x}^{(i)},\mathbf{\Phi}^t) \\ &= \frac{p(\mathbf{x}^{(i)}|z_l^{(i)}=1,\mathbf{\Phi}^t)P(z_l^{(i)}=1|\mathbf{\Phi}^t)}{p(\mathbf{x}^{(i)}|\mathbf{\Phi}^t)} \\ &= \frac{p_l(\mathbf{x}^{(i)}|\mathbf{\Phi}^t)\pi_l}{\sum_j p_j(\mathbf{x}^{(i)}|\mathbf{\Phi}^t)\pi_j} \\ &= \frac{p(\mathbf{x}^{(i)}|\mathcal{G}_l,\mathbf{\Phi}^t)P(\mathcal{G}_l)}{\sum_j p(\mathbf{x}^{(i)}|\mathcal{G}_j,\mathbf{\Phi}^t)P(\mathcal{G}_j)} = P(\mathcal{G}_l|\mathbf{x}^{(i)},\mathbf{\Phi}^t) \equiv h_l^{(i)}\end{aligned}$$

E-Step for Gaussian Mixtures (2)

- continued
- The **expected value of the hidden variable**, $h_l^{(i)}$, is the **posterior probability** that $\mathbf{x}^{(i)}$ is generated by component $\mathcal{G}_l$, i.e., $P(\mathcal{G}_l|\mathbf{x}^{(i)}, \mathbf{\Phi}^t)$
- Since $P(\mathcal{G}_l|\mathbf{x}^{(i)}, \mathbf{\Phi}^t) \in [0, 1]$ it can be seen as a **soft label**, as opposed to the hard label ($\in \{0, 1\}$) for k-means.
- Strictly speaking, the E-step only computes $h_l^{(i)}$ but not $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$.
- For Gaussian components, $\hat{p}_l(\mathbf{x}^{(i)}|\Theta_l) = \mathcal{N}(\mathbf{m}_l, \S_l)$ and so

$$h_l^{(i)} = \frac{\pi_l |\S_l|^{-1/2} \exp[-\frac{1}{2}(\mathbf{x}^{(i)} - \mathbf{m}_l)^T \S_l^{-1} (\mathbf{x}^{(i)} - \mathbf{m}_l)]}{\sum_j \pi_j |\S_j|^{-1/2} \exp[-\frac{1}{2}(\mathbf{x}^{(i)} - \mathbf{m}_j)^T \S_j^{-1} (\mathbf{x}^{(i)} - \mathbf{m}_j)]}$$

M-Step for Gaussian Mixtures (1)

- **Maximization** of $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$:
  $$\begin{aligned}\mathbf{\Phi}^{t+1} &= \arg max_{\mathbf{\Phi}} \mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t) \\ &= \arg max_{\mathbf{\Phi}} [\sum_i \sum_l h_l^{(i)} \log \pi_l + \sum_i \sum_l h_l^{(i)} \log p_l(\mathbf{x}^{(i)}|\mathbf{\Phi})]\end{aligned}$$
- The second term of $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$ does not depend on π_l. Using the constraint $\sum_l \pi_l = 1$to define the Lagrangian, we solve for
  $$\nabla_{\pi_l} [\sum_i \sum_l h_l^{(i)} \log \pi_l - \lambda(\sum_l \pi_l - 1)] = 0$$
  to get
  $$\pi_l = \frac{\sum_i h_l^{(i)}}{N}$$

M-Step for Gaussian Mixtures (2)

- The first term of $\mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$ does not depend on Θ_l. We solve for $\nabla_{\Theta_l} \sum_i \sum_l h_l^{(i)} \log p_l(\mathbf{x}^{(i)}|\mathbf{\Phi}) = 0$
- For Gaussian components, $\hat{p}_l(\mathbf{x}^{(i)}|\Theta_l) = \mathcal{N}(\mathbf{m}_l, \S_l)$ and so we get

$$\mathbf{m}_l^{t+1} = \frac{\sum_i h_l^{(i)} \mathbf{x}^{(i)}}{\sum_i h_l^{(i)}}$$

$$\S_l^{t+1} = \frac{\sum_i h_l^{(i)} (\mathbf{x}^{(i)} - \mathbf{m}_l^{t+1})(\mathbf{x}^{(i)} - \mathbf{m}_l^{t+1})^T}{\sum_i h_l^{(i)}}$$

Introducing Model Bias

- When the sample is small, **overfitting** may occur.
- Possible solutions via introducing model bias: \begin{itemize}
  - **Constraining the covariance matrices** of the Gaussian components, e.g., use shared or diagonal covariance matrices.
  - Applying Principal Component Analysis (PCA) or Factor Analysis (FA) to perform **dimensionality reduction** in the components, e.g., mixtures of latent variable models: $p(\mathbf{x}^{(i)}|\mathcal{G}_l) = \mathcal{N}(\mathbf{m}_l, \mathbf{V}_l\mathbf{V}_l^T + \mathbf{\Psi}_l)$

Summary: EM Algorithm

- A way of maximizing likelihood function for **latent variable models**. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:
  - **Estimate some missing or unobserved data** from observed data and current parameters.
  - Using this complete data, find the **maximum likelihood parameter estimates**.
- Alternate between **filling in the latent variables** using the best guess (posterior) and **updating the parameters** based on this guess.
  - E-step: ${h_l}^{t+1} = E[z_l|\mathcal{X}, \mathbf{\Phi}^t]$
  - M-step: $\mathbf{\Phi}^{t+1} = \arg max_{\mathbf{\Phi}} \mathcal{Q}(\mathbf{\Phi}|\mathbf{\Phi}^t)$
- In the M-step we optimize a lower bound on the likelihood. In the E-step we close the gap, making bound = likelihood.

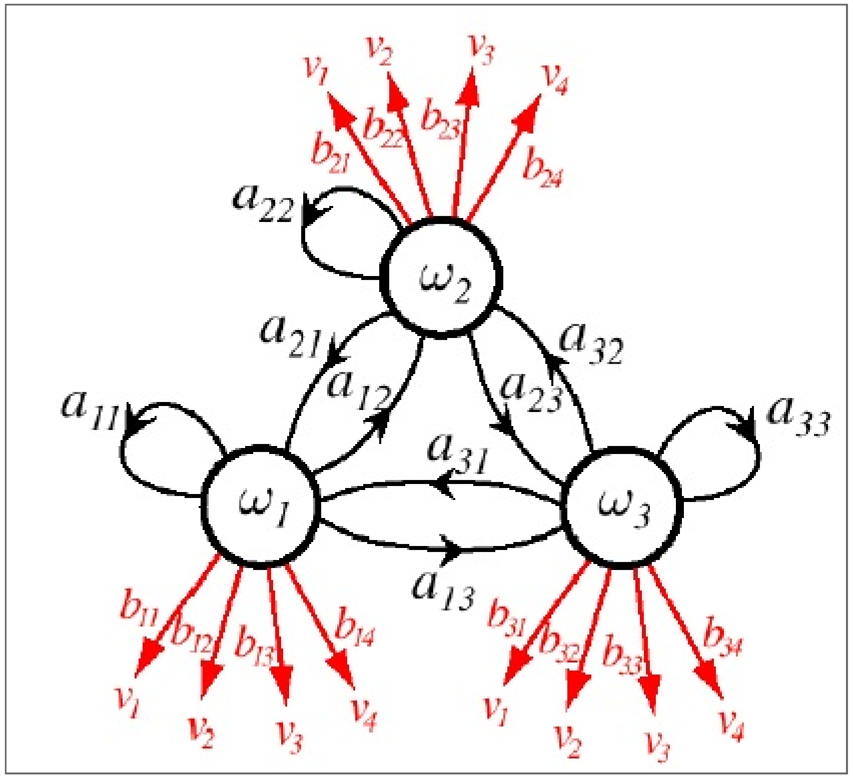
Hidden Markov Models

Markov Chains

- Discrete-time state sequence:
  $\omega(1), \ldots, \omega(t), \ldots$
  where $\omega(t)$ denotes the state at time t
- **Fist-order Markov process**:
  $P(\omega(t+1)|\omega(1), \ldots, \omega(t)) = P(\omega(t+1)|\omega(t))$
- **Transition probability matrix**:
  $$\begin{bmatrix} a_{11} & a_{12} & a_{11} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$
  where $a_{ij} = P(\omega(t+1) = \omega_j|\omega(t) = \omega_i) \sum_j a_{ij} = 1$
- The state sequence may be observable or unobservable.

Hidden Markov Models

- The **state sequence** of a **hidden Markov model (HMM)** is **unobservable**.
- Each hidden or unobservable state is associated with a separate probability distribution of the observable output events.
- HMM diagram:



Model Formulation for Discrete HMM

- Set of all **states**: $\Omega = \{\omega_1, \ldots, \omega_c\}$
- Set of all **observation** symbols: $V = \{v_1, \ldots, v_m\}$
- State sequence: $\omega = \omega(1)\omega(2)\ldots\omega(T)$
- Observation sequence: $\mathbf{X} = x(1)x(2)\ldots x(T)$
- State transition probability distribution: $\mathbf{A} = \{a_{ij} | a_{ij} = P(\omega(t+1) = \omega_j | \omega(t) = \omega_i)\}$
- Observation symbol probability distribution: $\mathbf{B} = \{b_{jk} | b_{jk} = P(x(t) = v_k | \omega(t) = \omega_j)\}$
- Initial state probability distribution: $\Pi = \{\pi_i | \pi_i = P(\omega(1) = \omega_i)\}$
- **Model parameter vector**: $\theta = (\mathbf{A}, \mathbf{B}, \Pi)$

Three Fundamental Problems

- **Likelihood evaluation problem**:
  - Given
    - Observation sequence $\mathbf{X}$
    - Model parameter vector θ
  - Find
    - Likelihood $P(\mathbf{X}|\theta)$
- **State sequence decoding problem**:
  - Given
    - Observation sequence $\mathbf{X}$
    - Model parameter vector θ
  - Find
    - State sequence ω that is optimal w.r.t. some optimality criterion
- **Parameter estimation (or learning) problem**:
  - Given
    - Observation sequence $\mathbf{X}$
  - Find
    - ML estimate of θ such that: $\theta_{ML} = argmax_\theta P(\mathbf{X}|\theta)$

Likelihood Evaluation Problem

- Brute-force computation:

  $$P(\mathbf{X}|\theta) = \sum_{\omega} P(\mathbf{X}|\omega,\theta)P(\omega|\theta)$$

  $$P(\mathbf{X}|\omega,\theta) = b_{\omega(1)x(1)} b_{\omega(2)x(2)} \dots b_{\omega(T)x(T)}$$

  $$P(\omega|\theta) = \pi_{\omega(1)} a_{\omega(1)\omega(2)} a_{\omega(2)\omega(3)} \dots a_{\omega(T-1)\omega(T)}$$

  Hence,

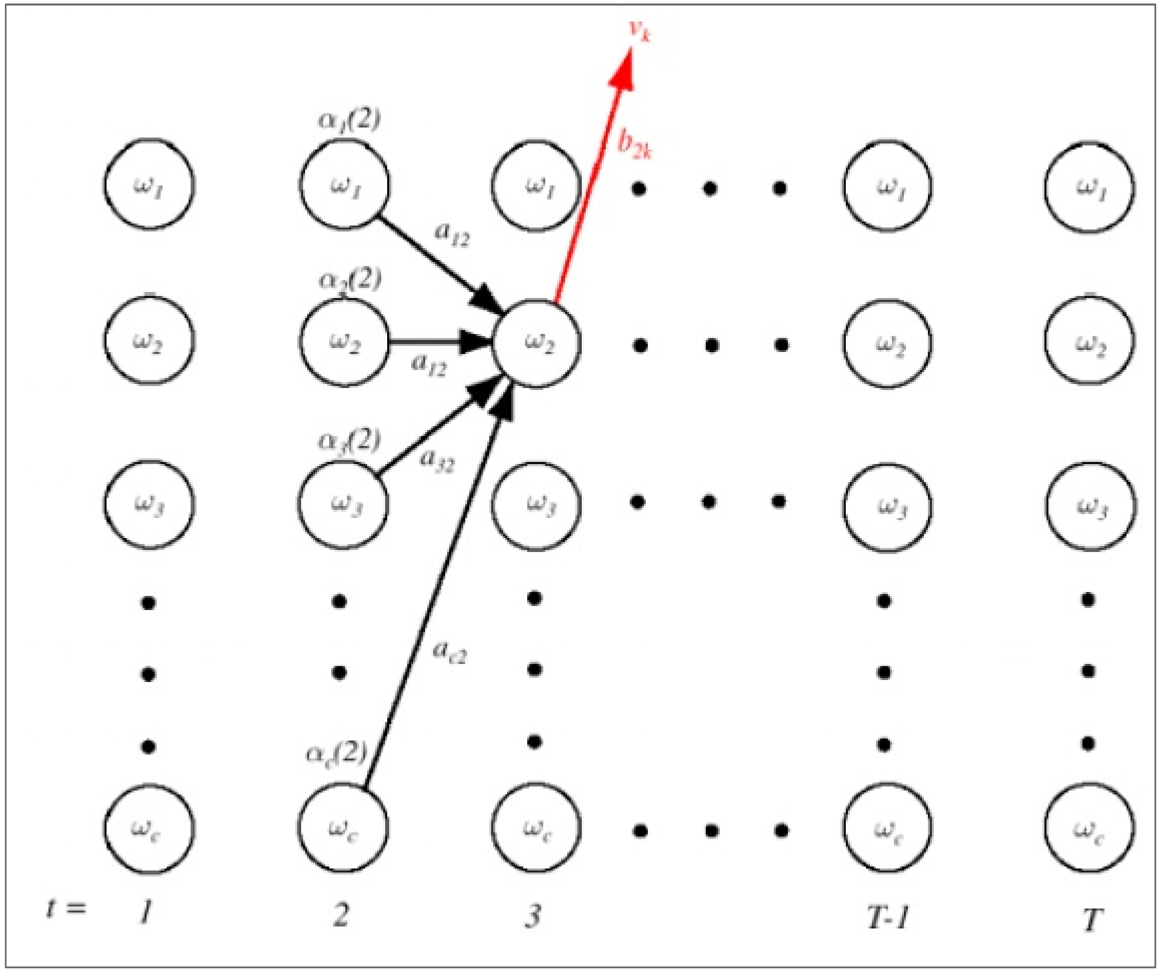
  $$P(\mathbf{X}|\theta) = \sum_{\omega} \prod_{t=1}^{T} a_{\omega(t-1)\omega(t)} b_{\omega(t)x(t)}$$

  where $a_{\omega(0)\omega(1)}$ denotes $\pi_{\omega(1)}$
  - Computationally expensive!
- Computationally attractive alternatives:
  - Either the **forward algorithm** or the **backward algorithm** can be used to compute $P(\mathbf{X}|\theta)$.

Likelihood Evaluation Problem - Forward Algorithm

- **Forward variables**: $\alpha_i(t) \equiv P(x(1) \ldots x(t), \omega(t) = \omega_i | \theta)$
- **Basis step**: For every state i:
  $\alpha_i(1) = \pi_i b_{ix(1)}$
- **General steps**: For every time step from $t = 2$ to T, for every state j:
  $$\alpha_j(t) = [\sum_{i=1}^{c} \alpha_i(t-1) a_{ij}] b_{jx(t)}$$
- **Final step**:
  $$P(\mathbf{X}|\theta) = \sum_{i=1}^{c} \alpha_i(T)$$
- Time complexity: $O(c^2 T)$

Illustration of Forward Algorithm



Likelihood Evaluation Problem - Backward Algorithm

- **Backward variables**: $\beta_i(t) \equiv P(x(t+1)x(t+2)\dots x(T)|\omega(t)=\omega_i,\theta)$
- **Basis step**: For every state i:
  $$\beta_i(T) = \frac{1}{c}$$
- **General steps**: For every time step from $t = T - 1$to 1, for every state i:
  $$\beta_i(t) = \sum_{j=1}^{c} a_{ij} b_{jx(t+1)} \beta_j(t+1)$$
- **Final step**:
  $$P(\mathbf{X}|\theta) = \sum_{i=1}^{c} \pi_i b_{ix(1)} \beta_i(1)$$
- Time complexity: $O(c^2T)$

State Sequence Decoding Problem (1)

- Find the **best state sequence** ω^* such that

$$\begin{aligned}\omega^* &= \arg\max_{\omega} P(\mathbf{X}, \omega|\theta) \\ &= \arg\max_{\omega} P(\omega|\mathbf{X}, \theta) P(\mathbf{X}|\theta) \\ &= \arg\max_{\omega} P(\omega|\mathbf{X}, \theta)\end{aligned}$$

  Thus, ω^* **maximizes the posterior probability** of ω.

State Sequence Decoding Problem (2)

- Equivalent form of optimality criterion:

$$\begin{aligned} P(\mathbf{X}, \omega|\theta) &= P(\omega|\theta)P(\mathbf{X}|\omega, \theta) \\ &= P(\omega(1), \omega(2), \ldots, \omega(T)|\theta) \cdot \\ &\quad P(x(1), x(2), \ldots, x(T)|\omega(1), \omega(2), \ldots, \omega(T), \theta) \\ &= \pi_{\omega(1)} \prod_{t=1}^{T-1} P(\omega(t+1)|\omega(t), \theta) \prod_{t=1}^{T} P(x(t)|\omega(t), \theta) \\ &= \prod_{t=1}^{T} a_{\omega(t-1)\omega(t)} b_{\omega(t)x(t)} \end{aligned}$$

  where $a_{\omega(0)\omega(1)}$ denotes $\pi_{\omega(1)}$
- **Dynamic programming** can be used to solve this optimization problem efficiently.

State Sequence Decoding Problem - Viterbi Algorithm

- **Best cumulative scores**:
  $\delta_i(t) = \max_{\omega(1),\ldots,\omega(t-1)} P(\omega(1),\ldots,\omega(t-1),\omega(t)=\omega_i, x(1),\ldots,x(t)|\theta)$
- **Basis step**: For every state i,
  $\delta_i(1) = \pi_i b_{ix(1)}, \;\; \psi_i(1) = 0$
- **General steps**: For every time step from $t = 2$ to T, for every state j,
  $\delta_j(t) = [\max_{1\le i\le c} \delta_i(t-1)a_{ij}]b_{jx(t)}, \;\; \psi_j(t) = \arg\max_{1\le i\le c} \delta_i(t-1)a_{ij}$
- **Final step**:
  $P^* = \max_{1\le i\le c} \delta_i(T), \;\; \omega^*(T) = \arg\max_{1\le i\le c} \delta_i(T)$

State Sequence Decoding Problem - Viterbi Algorithm (2)

- **State sequence backtracking**: For every time step from $t = T - 1$to 1, $\omega^*(t) = \psi_{\omega^*(t+1)}(t+1)$
- Time complexity: $O(c^2 T)$
- A modified version of Viterbi algorithm works in the logarithm domain to replace the multiplication operations by additions. Another advantage of operating in the logarithm domain is that computational underflow can be avoided.

Parameter Estimation Problem

- **Baum-Welch re-estimation algorithm** (or called **forward-backward algorithm**):
  - An EM algorithm (similar to that for mixture density estimation)
- Posterior probability of a state sequence being in state i at time t and in state j at time $t+1$, given $\mathbf{X}$ and θ:

$$\begin{aligned}\xi_{ij}(t) &\equiv P(\omega(t)=\omega_i, \omega(t+1)=\omega_j|\mathbf{X},\theta) \\ &= \frac{P(\omega(t)=\omega_i, \omega(t+1)=\omega_j, \mathbf{X}|\theta)}{P(\mathbf{X}|\theta)} \\ &= \frac{\alpha_i(t)a_{ij}b_{jx(t+1)}\beta_j(t+1)}{\sum_{i=1}^{c}\alpha_i(T)}\end{aligned}$$

Parameter Estimation Problem (2)

- Posterior probability of state sequence being in state i at time t, given $\mathbf{X}$ and θ:

$$\begin{aligned}\gamma_i(t) &\equiv P(\omega(t)=\omega_i|\mathbf{X},\theta) \\ &= \frac{P(\omega(t)=\omega_i,\mathbf{X}|\theta)}{P(\mathbf{X}|\theta)} \\ &= \frac{\alpha_i(t)\beta_i(t)}{\sum_{i=1}^{c}\alpha_i(T)}\end{aligned}$$

Baum-Welch Re-estimation Formula

- Re-estimation of model parameters (changing θ to $\hat{\theta}$):

$$\begin{aligned}
\hat{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)} \\
&= \frac{\text{expected number of transitions from state } i \text{ to } j}{\text{expected number of transitions from state } i} \\
\hat{b}_{jk} &= \frac{\sum_{\{t|x(t)=v_k\}} \gamma_j(t)}{\sum_{t=1}^{T} \gamma_j(t)} \\
&= \frac{\text{expected number of times in state } j \text{ when } v_k \text{ is observed}}{\text{expected number of times in state } j} \\
\hat{\pi}_i &= \frac{\gamma_i(1)}{\sum_{i=1}^{c} \gamma_i(1)} = \gamma_i(1) \\
&= \text{expected frequency in state } i \text{ at } t = 1
\end{aligned}$$

Baum-Welch Re-estimation Algorithm

- Using the EM ideas, it can be proved that $P(\mathbf{X}|\hat{\theta}) \geq P(\mathbf{X}|\theta)$
- Algorithm:

```
Initialize $\theta$
Repeat
    use Baum-Welch re-estimation formula to compute $\hat{\theta}$ from $\theta$ and $\mathbf{X}$
    $\hat{\theta} \leftarrow \theta$
Until convergence
Return $\theta$
```

Summary of HMMs

- **Elements** of an HMM $\theta = (\mathbf{A}, \mathbf{B}, \Pi)$
  - state transition probability matrix $\mathbf{A} = \{a_{ij}\}$:
    $a_{ij} = P(\omega(t+1) = \omega_j | \omega(t) = \omega_i)$
  - observation probability distribution $\mathbf{B} = \{b_{jk}\}$:
    $b_{jk} = P(x(t) = v_k | \omega(t) = \omega_j)$
  - initial state distribution $\Pi = \{\pi_i\}$:
    $\pi_i = P(\omega(1) = \omega_i)$
- **Basic problems**
  - evaluation problem (forward and backward algorithms)
  - decoding problem (Viterbi algorithm)
  - learning/estimation problem (Baum-Welch algrithm)
- Variants: continuous-density HMMs, semi-continuous HMMs, etc.

Application of HMMs

- HMM is probabilistic model for **sequential data**. It can be used to solve the learning problems such as:
  - K HMMs for K class **classification** Mixture of HMMs for **clustering**
- Applications of HMMs
  - Speech recognition
  - Speech synthesis
  - Part-of-speech tagging
  - Machine translation
  - Gene prediction
  - Alignment of bio-sequences
  - Activity recognition
  - $\ldots$

Hierarchical Clustering

Hierarchical Clustering

- **Hierarchical clustering** generally refers to methods that cluster instances into a hierarchical structure (called dendrogram) based on some similarity or distance measure between them.
- Distance measures:
  - **Minkowski distance**:

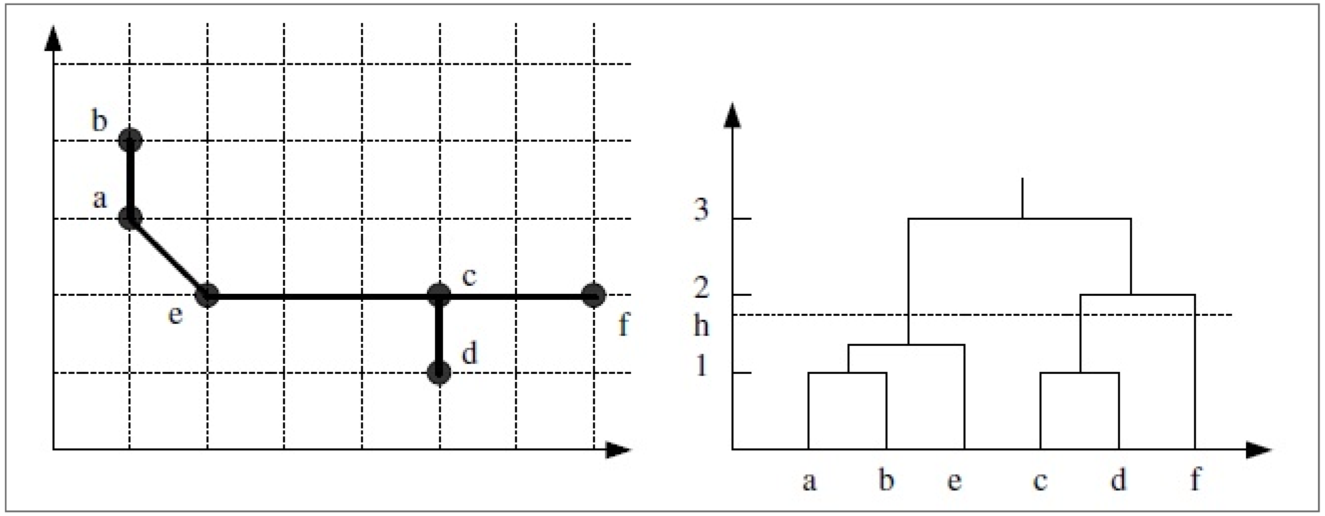
$$d_m(\mathbf{x}^{(r)}, \mathbf{x}^{(s)}) = [\sum_{j=1}^{D} |x_j^{(r)} - x_j^{(s)}|^p]^{1/p}$$

  - **Euclidean distance**: special case of Monkowski distance with $p = 2$
  - **City-block distance**: special case of Monkowski distance with $p = 1$

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

Agglomerative vs. Divisive Clustering

- An **agglomerative clustering** algorithm starts with *N* groups each with one instance.
- At each iteration, the two most similar groups are merged, reducing the number of groups by one.
- The process stops when there is only one group left.
- Dendrogram (can be intersected at any level to get the clusters):



- A **divisive clustering** algorithm starts with one group and goes in the opposite direction.

Distance between Groups $\mathcal{G}_i$ and $\mathcal{G}_j$

- Single-link clustering:
  $$d_{min}(\mathcal{G}_i, \mathcal{G}_j) = \min_{\mathbf{x}^{(r)} \in \mathcal{G}_i, \mathbf{x}^{(s)} \in \mathcal{G}_j} d(\mathbf{x}^{(r)}, \mathbf{x}^{(s)})$$
- Complete-link clustering:
  $$d_{max}(\mathcal{G}_i, \mathcal{G}_j) = \max_{\mathbf{x}^{(r)} \in \mathcal{G}_i, \mathbf{x}^{(s)} \in \mathcal{G}_j} d(\mathbf{x}^{(r)}, \mathbf{x}^{(s)})$$
- Average-link clustering:
  $$d_{avg}(\mathcal{G}_i, \mathcal{G}_j) = \frac{1}{|\mathcal{G}_i| \cdot |\mathcal{G}_j|} \sum_{\mathbf{x}^{(r)} \in \mathcal{G}_i, \mathbf{x}^{(s)} \in \mathcal{G}_j} d(\mathbf{x}^{(r)}, \mathbf{x}^{(s)})$$
- Other possibilities, e.g., distance between group means (centroids).

Choosing k

- This is a **model selection** issue for clustering.
- Some common possibilities:
  - Defined by the **application**, e.g., image
  - **Visualize** the data (e.g., in 2-D using PCA) and check for clusters.
  - **Incremental approach**: add one at a time and monitor the reconstruction error, log likelihood, or intergroup distances.
  - **Domain experts** check the clustering result.