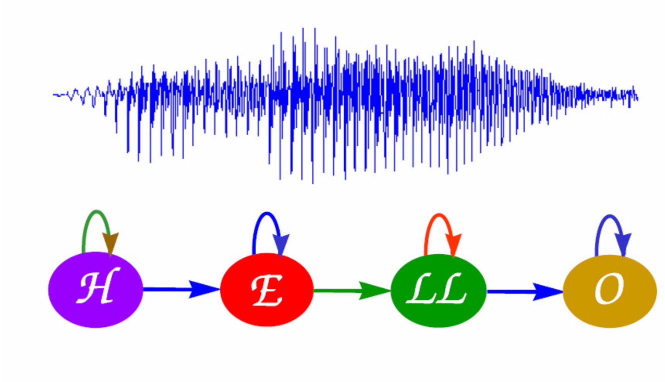


Hidden Markov Models for Speech Recognition

Berlin Chen 2004

References:

1. Rabiner and Juang. *Fundamentals of Speech Recognition*. Chapter 6
2. Huang et. al. *Spoken Language Processing*. Chapters 4, 8
3. Vaseghi. *Advanced Digital Signal Processing and Noise Reduction*. Chapter 5
4. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. Proceedings of the IEEE, vol. 77, No. 2, February 1989

Introduction

- Hidden Markov Model (HMM)

  History

  - Published in papers of Baum in late 1960s and early 1970s
  - Introduced to speech processing by Baker (CMU) and Jelinek (IBM) in the 1970s

  Assumption

  - Speech signal can be characterized as a parametric random process
  - Parameters can be estimated in a precise, well-defined manner

  Three fundamental problems

  - Evaluation of probability (likelihood) of a sequence of observations given a specific HMM
  - Determination of a best sequence of model states
  - Adjustment of model parameters so as to best account for observed signals

Observable Markov Model

- Observable Markov Model (Markov Chain)
  - **First-order** Markov chain of *N* states is a triple (***S***,***A***,π)
    - ***S*** is a set of *N* states
    - ***A*** is the $N \times N$ matrix of transition probabilities between states
      $P(s_t=j|s_{t-1}=i, s_{t-2}=k, \ldots\ldots)=P(s_t=j|s_{t-1}=i)\equiv A_{ij}$

First-order and time-invariant assumptions

    - π is the vector of initial state probability
      $\pi_j = P(s_1=j)$
  - The output of the process is the set of states at each instant of time, when each state corresponds to an observable event
  - The output in any given state is not random (***deterministic!***)
  - Too simple to describe the speech signal characteristics

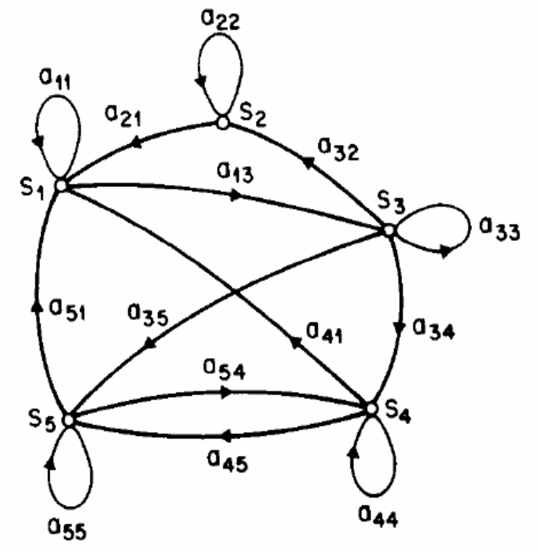


Fig. 1. A Markov chain with 5 states (labeled S_1 to S_5) with selected state transitions.

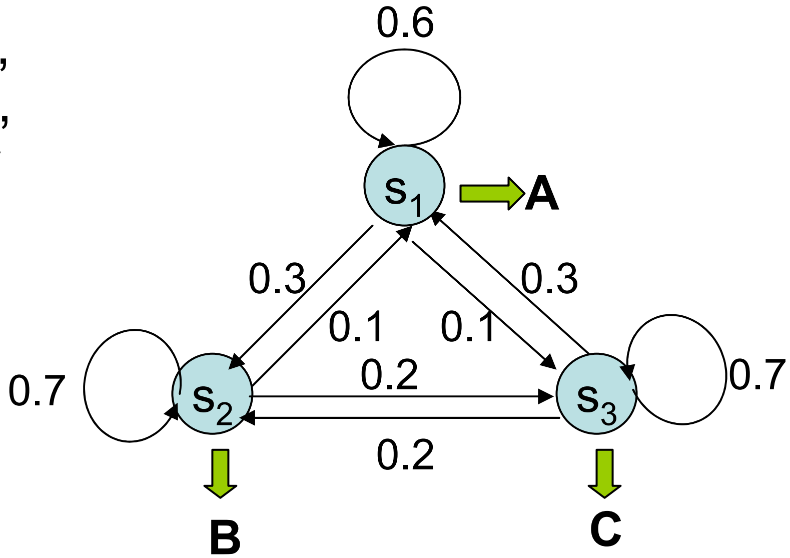
Observable Markov Model (cont.)

- Example 1: A 3-state Markov Chain λ

  State 1 generates symbol A **only**,
  State 2 generates symbol B **only**,
  and State 3 generates symbol C **only**

$$\mathbf{A} = \begin{bmatrix} 0.6 & 0.3 & 0.1 \\ 0.1 & 0.7 & 0.2 \\ 0.3 & 0.2 & 0.5 \end{bmatrix}$$

$$\pi = \begin{bmatrix} 0.4 & 0.5 & 0.1 \end{bmatrix}$$

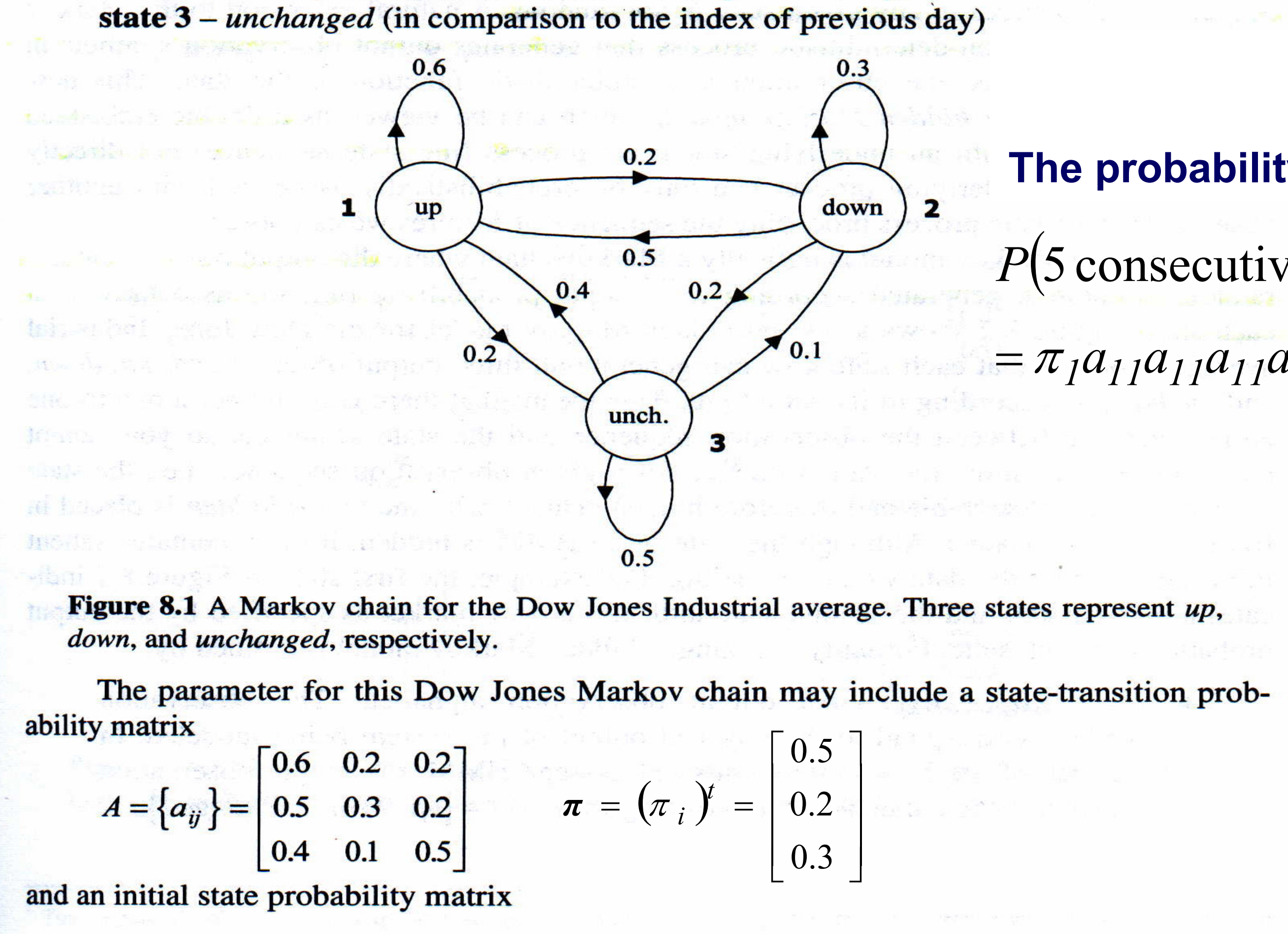


  - Given a sequence of observed symbols ***O***={CABBCABC}, the **only one** corresponding state sequence is $\{S_3S_1S_2S_2S_3S_1S_2S_3\}$, and the corresponding probability is

    $P(\boldsymbol{O}|\lambda)$
    $=P(S_3)P(S_1|S_3)P(S_2|S_1)P(S_2|S_2)P(S_3|S_2)P(S_1|S_3)P(S_2|S_1)P(S_3|S_2)$
    $=0.1\times0.3\times0.3\times0.7\times0.2\times0.3\times0.3\times0.2=0.00002268$

Observable Markov Model (cont.)

- Example 2: A three-state Markov chain for the *Dow Jones Industrial average*



state 1 – *up* (in comparison to the index of previous day)
state 2 – *down* (in comparison to the index of previous day)
state 3 – *unchanged* (in comparison to the index of previous day)

Figure 8.1 A Markov chain for the Dow Jones Industrial average. Three states represent *up*, *down*, and *unchanged*, respectively.

The parameter for this Dow Jones Markov chain may include a state-transition probability matrix

$$A = \{a_{ij}\} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.5 & 0.3 & 0.2 \\ 0.4 & 0.1 & 0.5 \end{bmatrix} \qquad \boldsymbol{\pi} = (\pi_i)^t = \begin{bmatrix} 0.5 \\ 0.2 \\ 0.3 \end{bmatrix}$$

and an initial state probability matrix

The probability of 5 consecutive *up* days

$$P(5\ \text{consecutive}\ up\ \text{days}) = P(1,1,1,1,1)$$

$$= \pi_1 a_{11} a_{11} a_{11} a_{11} = 0.5 \times (0.6)^4 = 0.0648$$

Observable Markov Model (cont.)

- Example 3: Given a Markov model, what is the mean occupancy duration of each state *i*

$$p_i(d) = prob.\ \text{density function of duration } d \text{ in state } i$$

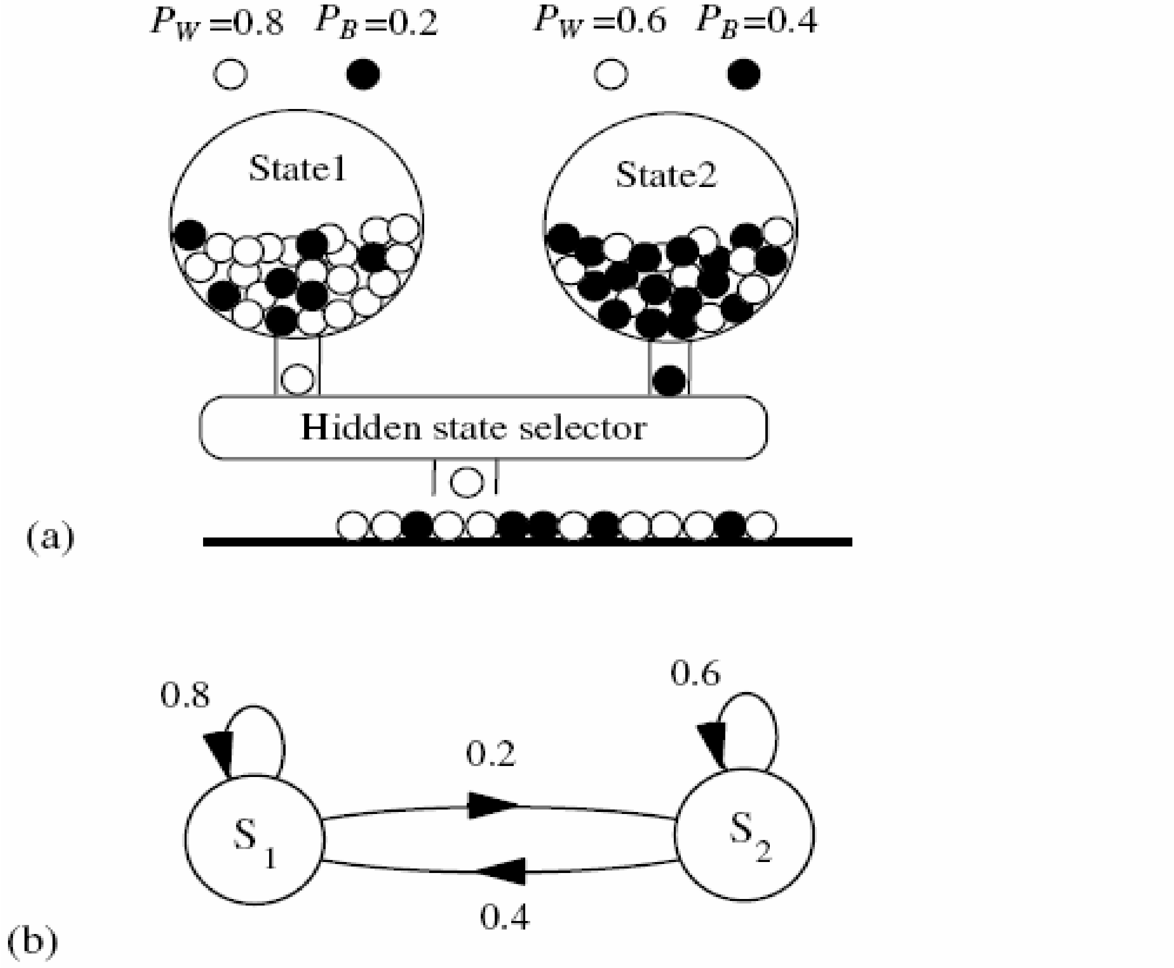
$$= (a_{ii})^{d-1}(1-a_{ii})$$

$$\text{Expected number of duration in a state}$$

$$\bar{d}_i = \sum_{d=1}^{\infty} d p_i(d) = \sum_{d=1}^{\infty} d (a_{ii})^{d-1}(1-a_{ii}) = (1-a_{ii}) \frac{\partial}{\partial a_{ii}} \sum_{d=1}^{\infty} (a_{ii})^d$$

$$= (1-a_{ii}) \frac{\partial}{\partial a_{ii}} \frac{1}{1-a_{ii}} = \frac{1}{1-a_{ii}}$$

Hidden Markov Model



(a) Illustration of a two-layered random process. (b) An HMM model of the process in (a).

Hidden Markov Model (cont.)

- HMM, an extended version of Observable Markov Model
  - The observation is turned to be a probabilistic function (discrete or continuous) of a state instead of an one-to-one correspondence of a state
  - The model is a doubly embedded stochastic process with an underlying stochastic process that is not directly observable (hidden)
    - What is hidden? ***The State Sequence!***
      According to the observation sequence, we are not sure which state sequence generates it!

- Elements of an HMM (the State-Output HMM) $\lambda=\{\boldsymbol{S},\boldsymbol{A},\boldsymbol{B},\pi\}$
  - $\boldsymbol{S}$ is a set of N states
  - $\boldsymbol{A}$ is the $N \times N$ matrix of transition probabilities between states
  - $\boldsymbol{B}$ is a set of N probability functions, each describing the observation probability with respect to a state
  - π is the vector of initial state probability

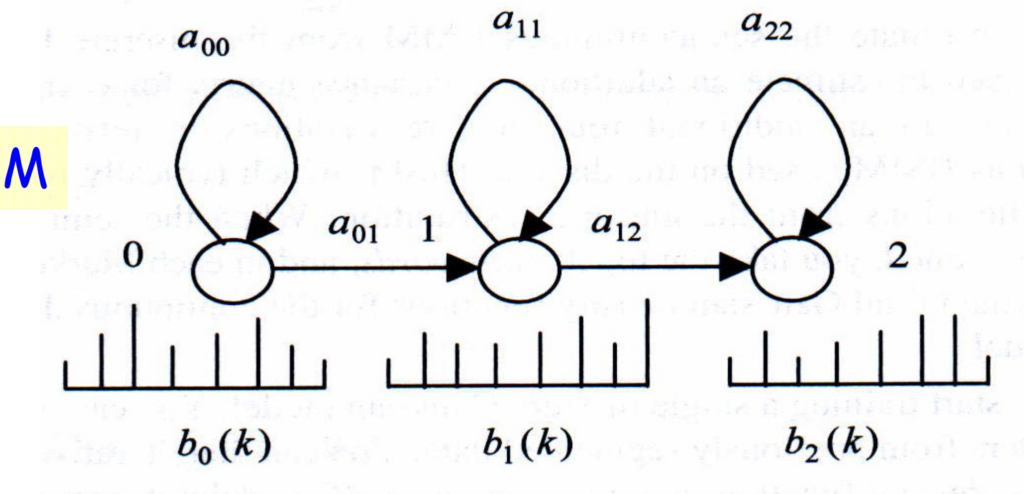
Hidden Markov Model (cont.)

- Two major assumptions
  - First order (Markov) assumption
    - The state transition depends only on the origin and destination
    - Time-invariant
  - Output-independent assumption
    - All observations are dependent on the state that generated them, not on neighboring observations

Hidden Markov Model (cont.)

- Two major types of HMMs according to the observations
  - **Discrete and finite observations:**
    - The observations that all distinct states generate are finite in number
      $\boldsymbol{V}=\{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \ldots\ldots, \boldsymbol{v}_M\}, \boldsymbol{v}_k \in \boldsymbol{R}^L$
    - In this case, the set of observation probability distributions $B=\{b_j(\mathbf{v}_k)\}$, is defined as $b_j(\mathbf{v}_k)=P(\mathbf{o}_t=\mathbf{v}_k|s_t=j)$, $1 \leq k \leq M$, $1 \leq j \leq N$
      $\mathbf{o}_t$: *observation at time t*, s_t : *state at time t*
      ⇨ *for state j*, $b_j(\mathbf{v}_k)$ consists of *only M probability values*

A left-to-right HMM



Hidden Markov Model (cont.)

- Two major types of HMMs according to the observations
  - **Continuous and infinite observations:**
    - The observations that all distinct states generate are infinite and continuous, that is, $\boldsymbol{V}=\{\mathbf{v} \mid \mathbf{v} \in \boldsymbol{R}^L\}$
    - In this case, the set of observation probability distributions $B=\{b_j(\mathbf{v})\}$, is defined as $b_j(\mathbf{v})=f_{\mathbf{O}|\mathbf{S}}(\mathbf{o}_t=\mathbf{v}|s_t=j)$, $1 \le j \le N$
      ⇨ $b_j(\mathbf{v})$ is a ***continuous probability density function (pdf)*** ***and is often a mixture of Multivariate Gaussian*** **(*Normal*)** ***Distributions***

$$b_j(\boldsymbol{v})=\sum_{k=1}^{M} w_{jk}\left(\frac{1}{(2\pi)^{L/2}\left|\boldsymbol{\Sigma}_{jk}\right|^{1/2}}\exp\left(-\frac{1}{2}\left(\boldsymbol{v}-\boldsymbol{\mu}_{jk}\right)^t \boldsymbol{\Sigma}_{jk}^{-1}\left(\boldsymbol{v}-\boldsymbol{\mu}_{jk}\right)\right)\right)$$

Mean Vector

Covariance Matrix

Observation Vector

Hidden Markov Model (cont.)

- Multivariate Gaussian Distributions
  - When ***X***=(***X***$_1$, ***X***$_2$,…, ***X***$_L$) is a *L*-dimensional random vector, the multivariate Gaussian pdf has the form:

$$f\left(\boldsymbol{X}=\boldsymbol{x}\middle|\boldsymbol{\mu},\boldsymbol{\Sigma}\right)=N\left(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}\right)=\frac{1}{(2\pi)^{L/2}\left|\boldsymbol{\Sigma}\right|^{1/2}}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{t}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

where $\boldsymbol{u}$ is the L - dimensional mean vector,

$\boldsymbol{\Sigma}$ is the coverance matrix, $\boldsymbol{\Sigma}=E\left[(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^{t}\right]=E\left[\boldsymbol{x}\boldsymbol{x}^{t}\right]-\boldsymbol{\mu}\boldsymbol{\mu}^{t}$

and $\left|\boldsymbol{\Sigma}\right|$ is the the dterminant of $\boldsymbol{\Sigma}$

The i-j^{th} elevment σ_{ij}^{2} of $\boldsymbol{\Sigma}$, $\sigma_{ij}^{2}=E\left[(x_i-\mu_i)(x_j-\mu_j)\right]=E\left[x_i x_j\right]-\mu_i\mu_j$

  - If ***X***$_1$, ***X***$_2$,…, ***X***$_L$ are independent, the covariance matrix is reduced to diagonal covariance
    - The distribution as *L* independent scalar Gaussian distributions
    - Model complexity is reduced

Hidden Markov Model (cont.)

- Multivariate Gaussian Distributions

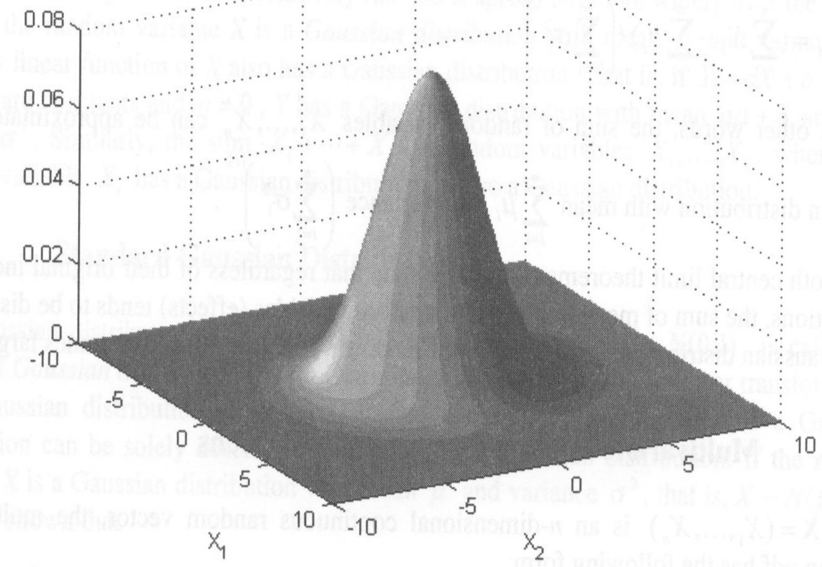


Figure 3.12 A two-dimensional multivariate Gaussian distribution with independent random variables x_1 and x_2 that have the same variance.

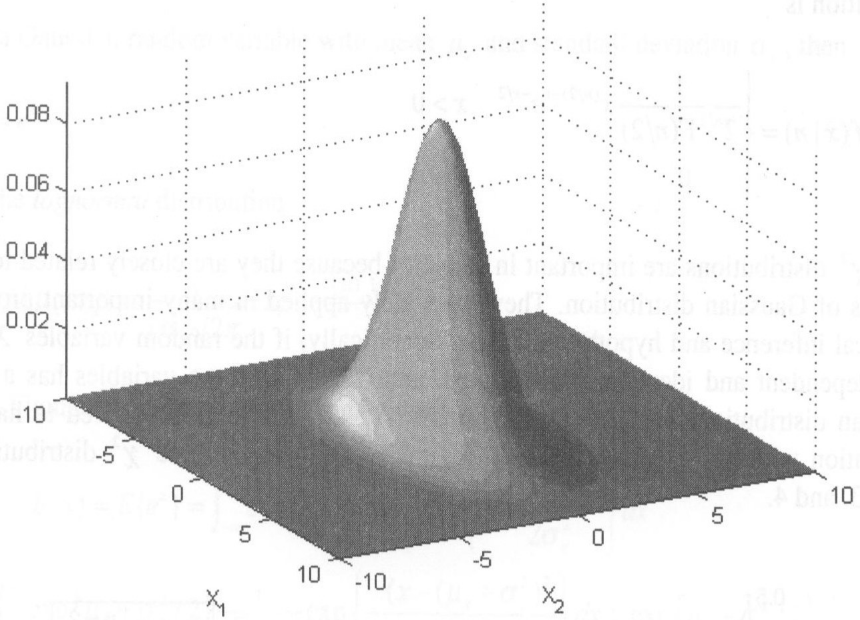
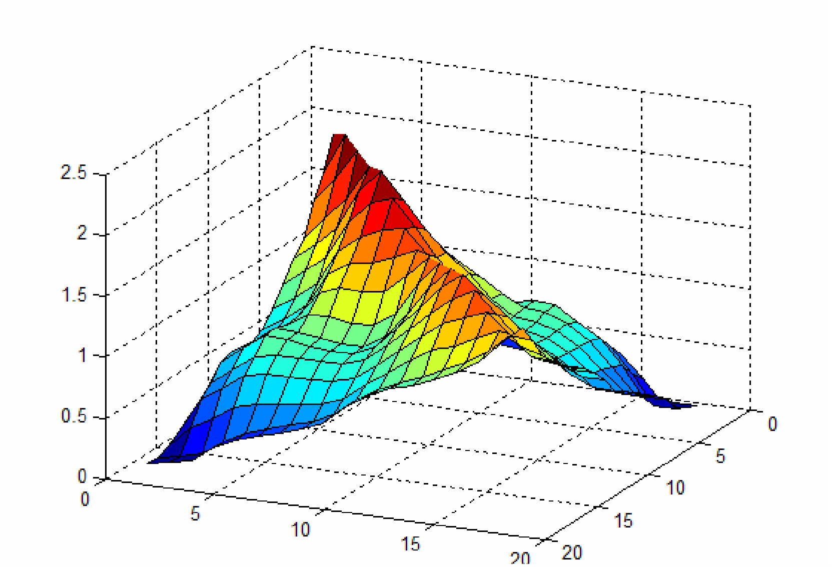


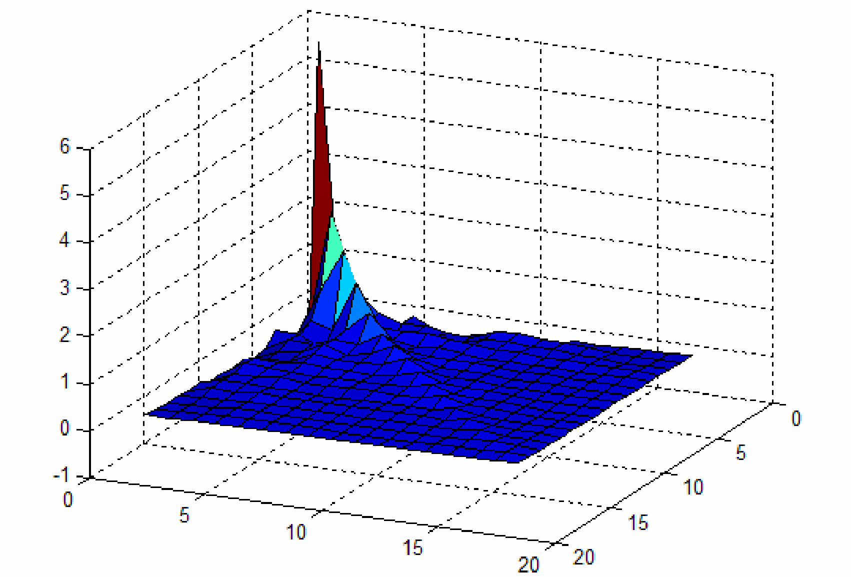
Figure 3.13 Another two-dimensional multivariate Gaussian distribution with independent random variable x_1 and x_2 which have different variances.

Hidden Markov Model (cont.)

- Covariance matrix of the correlated feature vectors (Mel Frequency filter bank outputs)



- Covariance matrix of the partially decorrelated feature vectors (MFCC cepstrum without C_0)

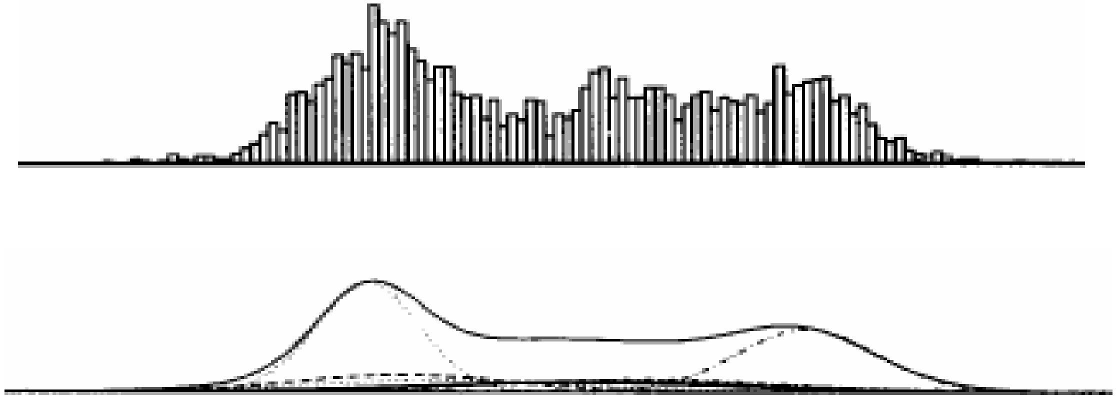


Hidden Markov Model (cont.)

- Multivariate ***Mixture*** Gaussian Distributions (cont.)
  - More complex distributions with multiple local maxima can be approximated by Gaussian (a unimodal distribution) mixture

$$f(\boldsymbol{x}) = \sum_{k=1}^{M} w_k N_k(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \qquad \sum_{k=1}^{M} w_k = 1$$

  - Gaussian mixtures with enough mixture components can approximate any distribution

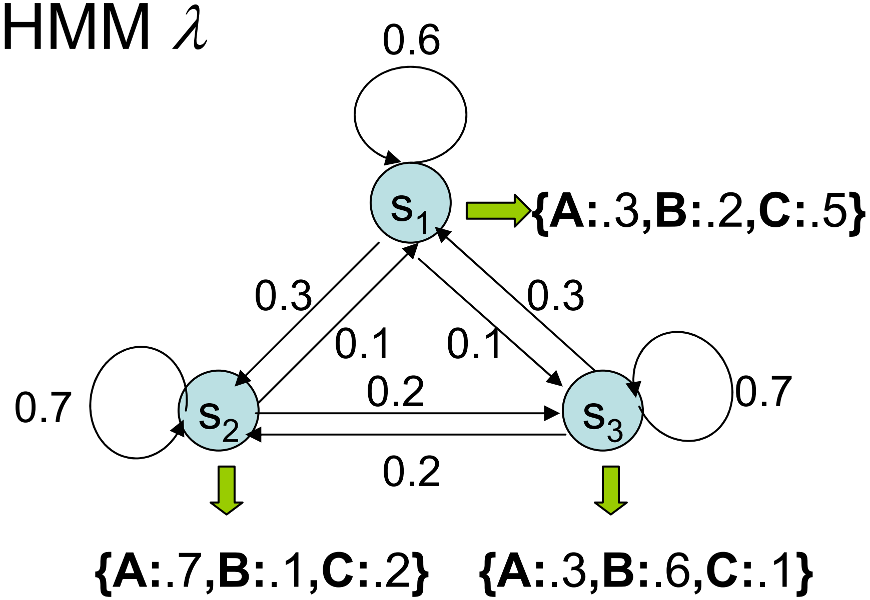


Hidden Markov Model (cont.)

- Example 4: a 3-state discrete HMM λ

$$\mathbf{A}=\begin{bmatrix}0.6 & 0.3 & 0.1\\ 0.1 & 0.7 & 0.2\\ 0.3 & 0.2 & 0.5\end{bmatrix}$$

$$b_1(\mathbf{A})=0.3, b_1(\mathbf{B})=0.2, b_1(\mathbf{C})=0.5$$
$$b_2(\mathbf{A})=0.7, b_2(\mathbf{B})=0.1, b_2(\mathbf{C})=0.2$$
$$b_3(\mathbf{A})=0.3, b_3(\mathbf{B})=0.6, b_3(\mathbf{C})=0.1$$
$$\pi=\begin{bmatrix}0.4 & 0.5 & 0.1\end{bmatrix}$$



  - Given a sequence of observations *O={ABC}*, there are **27 possible** corresponding state sequences, and therefore the corresponding probability is

$$P(\boldsymbol{O}|\lambda)=\sum_{i=1}^{27}P(\boldsymbol{O},\boldsymbol{S}_i|\lambda)=\sum_{i=1}^{27}P(\boldsymbol{O}|\boldsymbol{S}_i,\lambda)P(\boldsymbol{S}_i|\lambda), \qquad \boldsymbol{S}_i : \text{state sequence}$$

$$E.g.\text{ when } \boldsymbol{S}_i=\{s_2 s_2 s_3\}, P(\boldsymbol{O}|\boldsymbol{S}_i,\boldsymbol{\lambda})=P(\boldsymbol{A}|s_2)P(\boldsymbol{B}|s_2)P(\boldsymbol{C}|s_3)=0.7*0.1*0.1=0.007$$

$$P(\boldsymbol{S}_i|\lambda)=P(s_2)P(s_2|s_2)P(s_3|s_2)=0.5*0.7*0.2=0.07$$

Hidden Markov Model (cont.)

- Notation :
  - **$O=\{o_1o_2o_3......o_T\}$**: the observation (feature) sequence
  - **$S=\{s_1s_2s_3......s_T\}$** : the state sequence
  - λ : model, for HMM, $\lambda=\{\mathbf{A},\mathbf{B},\pi\}$
  - $P(\mathbf{O}|\lambda)$: 用model λ 計算 **O** 的機率值
  - $P(\mathbf{O}|\mathbf{S},\lambda)$:在**O**是state sequence **S** 所產生的前提下, 用model λ 計算 **O** 的機率值
  - $P(\mathbf{O},\mathbf{S}|\lambda)$:用model λ 計算[**O**，**S**]兩者同時成立的機率值
  - $P(\mathbf{S}|\mathbf{O},\lambda)$: 在已知**O**的前提下， 用model λ 計算 **S** 的機率值
- Useful formula
  - Bayesian Rule :

$$P(A|B)=\frac{P(A,B)}{P(B)}=\frac{P(B|A)P(A)}{P(B)} \quad \Rightarrow \quad P(A|B,\lambda)=\frac{P(A,B|\lambda)}{P(B|\lambda)}=\frac{P(B|A,\lambda)P(A|\lambda)}{P(B|\lambda)}$$

λ : model describing the probability

$$P(A,B)=P(B|A)P(A)=P(A|B)P(B)$$

chain rule

Hidden Markov Model (cont.)

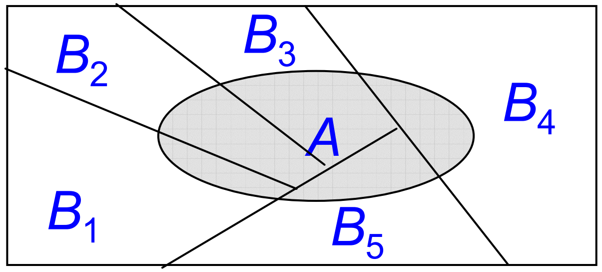
- Useful formula (Cont.):

marginal probability

$$P(A)=\begin{cases}\sum_{all\ B} P(A,B)=\sum_{all\ B} P(A|B)P(B), & \text{if } B \text{ is disrete and disjoint}\\ \int_B f(A,B)dB=\int_B f(A|B)f(B)dB, & \text{if } B \text{ is continuous}\end{cases}$$

if $x_1,x_2,\ldots\ldots,x_n$ are independent,

$$P(x_1,x_2,\ldots\ldots,x_n)=P(x_1)P(x_2)\ldots\ldots P(x_n)$$



$$E_z(q(z))=\begin{cases}\sum_k P(z=k)q(k), & z:\text{discrete}\\ \int_z f_z(z)q(z)dz, & z:\text{continuous}\end{cases}$$

Expectation

Three Basic Problems for HMM

- Given an observation sequence $O=(o_1,o_2,\ldots..,o_T)$, and an HMM $\lambda=(S,A,B,\pi)$
  - **Problem *1***:
    How to *efficiently* compute $P(\boldsymbol{O}|\lambda)$?
    ⇨ ***Evaluation problem***
  - **Problem *2*:**
    How to choose an optimal state sequence $\boldsymbol{S}=(s_1,s_2,\ldots\ldots, s_T)$?
    ⇨ ***Decoding Problem***
  - **Problem *3*:**
    How to adjust the model parameter $\lambda=(\boldsymbol{A},B,\pi)$ to maximize $P(\boldsymbol{O}|\lambda)$?
    ⇨ ***Learning / Training Problem***

Basic Problem 1 of HMM (cont.)

Given ***O*** and λ, find $P(\mathbf{O}|\lambda)$= Prob[observing *O* given λ]

- Direct Evaluation
  - Evaluating all possible state sequences of length *T* that generating observation sequence ***O***

$$P(\mathbf{O}|\lambda) = \sum_{all\ \mathbf{S}} P(\mathbf{O}, \mathbf{S}|\lambda) = \sum_{all\ S} P(\mathbf{O}|\mathbf{S}, \lambda) P(\mathbf{S}|\lambda)$$

  - $P(\mathbf{S}|\lambda)$: The probability of each path ***S***
    - By Markov assumption (First-order HMM)

$$P(\mathbf{S}|\lambda) = P(s_1|\lambda) \prod_{t=2}^{T} P\left(s_t \middle| s_1^{t-1}, \lambda\right)$$

By chain rule

$$\approx P(s_1|\lambda) \prod_{t=2}^{T} P(s_t|s_{t-1}, \lambda)$$

By Markov assumption

$$= \pi_{s_1} a_{s_1 s_2} a_{s_2 s_3} \dots a_{s_{T-1} s_T}$$

Basic Problem 1 of HMM (cont.)

- Direct Evaluation (cont.)
  - $P(\boldsymbol{O}|\boldsymbol{S},\lambda)$: The joint output probability along the path *S*
    - By output-independent assumption
      - The probability that a particular observation symbol/vector is emitted at time *t* depends only on the state s_t and is conditionally independent of the past observations

$$
\begin{aligned}
P(\boldsymbol{O}|\boldsymbol{S},\lambda) &= P\left(\boldsymbol{o}_1^T \middle| s_1^T,\lambda\right) \\
&= P\left(\boldsymbol{o}_1 \middle| s_1^T,\lambda\right)\prod_{t=2}^{T} P\left(\boldsymbol{o}_t \middle| \boldsymbol{o}_1^{t-1}, s_1^T,\lambda\right) \\
&\approx \prod_{t=1}^{T} P\left(\boldsymbol{o}_t \middle| s_t,\lambda\right) \qquad \text{By output-independent assumption} \\
&= \prod_{t=1}^{T} b_{s_t}(\boldsymbol{o}_t)
\end{aligned}
$$

Basic Problem 1 of HMM (cont.)

- Direct Evaluation (Cont.)

$$P(\boldsymbol{o}_t | s_t, \lambda) = b_{s_t}(\boldsymbol{o}_t)$$

$$P(\boldsymbol{O}|\lambda) = \sum_{all\ \boldsymbol{S}} P(\boldsymbol{S}|\lambda) P(\boldsymbol{O}|\boldsymbol{S}, \lambda)$$

$$= \sum_{all\ \boldsymbol{s}} \left(\left[\pi_{s_1} a_{s_1 s_2} a_{s_2 s_3} \ldots\ldots a_{s_{T-1} s_T} \right] \left[b_{s_1}(\boldsymbol{o}_1) b_{s_2}(\boldsymbol{o}_2) \ldots\ldots b_{s_T}(\boldsymbol{o}_T) \right] \right)$$

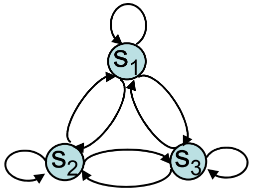
$$= \sum_{s_1, s_2, \ldots, s_T} \pi_{s_1} b_{s_1}(\boldsymbol{o}_1) a_{s_1 s_2} b_{s_2}(\boldsymbol{o}_2) \ldots\ldots a_{s_{T-1} s_T} b_{s_T}(\boldsymbol{o}_T)$$

  - Huge Computation Requirements: $O(N^T)$
    - Exponential computational complexity

$$\textbf{\textit{Complexity}} \quad : (2T\text{-}1) N^{T}\ MUL \quad \approx 2TN^{T},\ N^{T}\text{-}1\ \text{ADD}$$

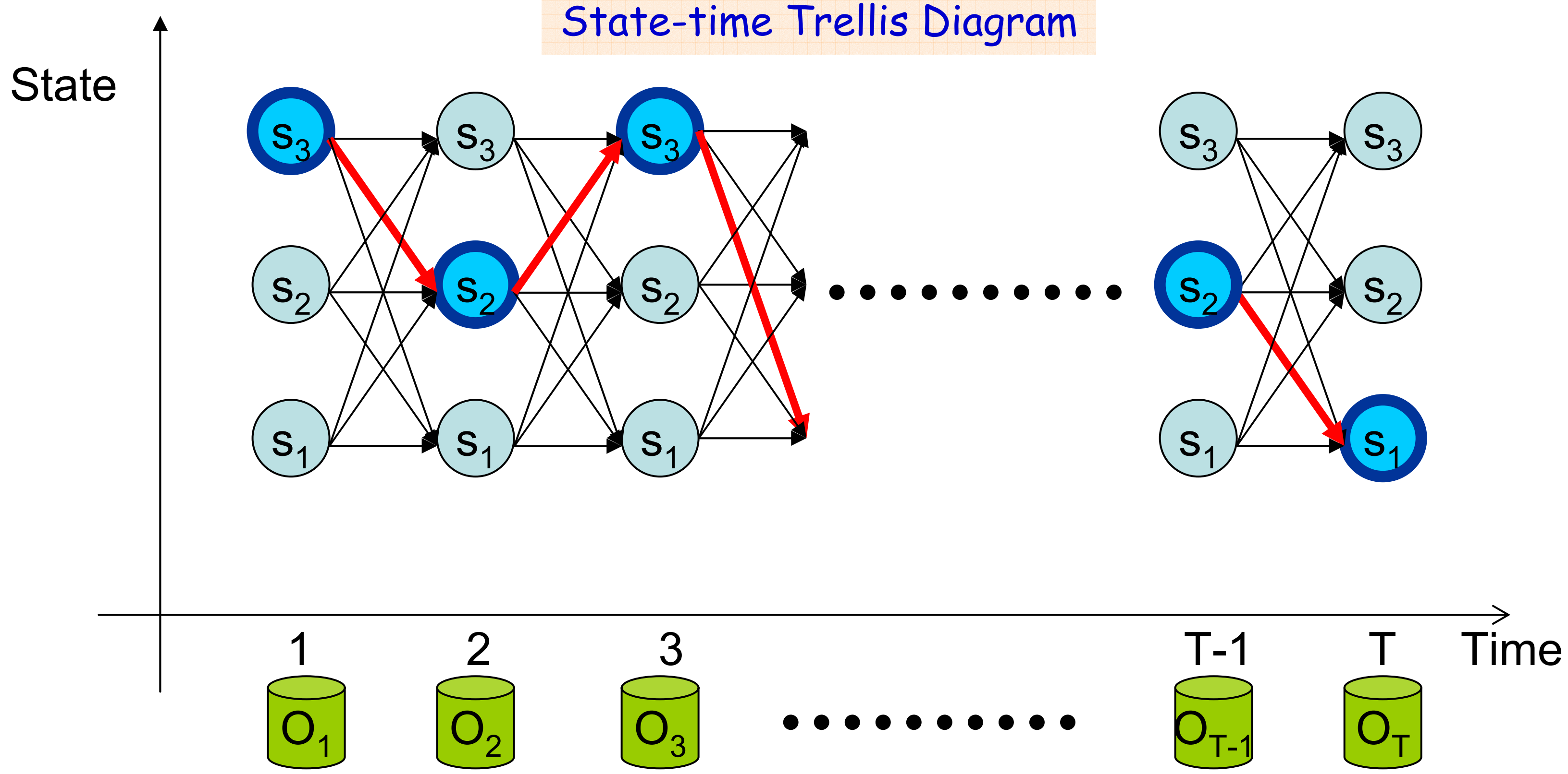
- A more efficient algorithms can be used to evaluate $P(\boldsymbol{O}|\lambda)$
  - *Forward/Backward Procedure/Algorithm*

Basic Problem 1 of HMM (cont.)



- Direct Evaluation (Cont.)

State-time Trellis Diagram



s_i **means $b_j(o_t)$ has been computed**

$\xrightarrow{a_{ij}}$ **means a_{ij} has been computed**

Basic Problem 1 of HMM

- The Forward Procedure

- Base on the HMM assumptions, the calculation of $P\left(s_t \middle| s_{t-1}, \lambda\right)$ and $P\left(\boldsymbol{o}_t \middle| s_t, \lambda\right)$ involves only s_{t-1}, s_t and $\boldsymbol{o}_t$, so it is possible to compute the likelihood with recursion on t

- Forward variable : $\alpha_t(i) = P\left(o_1 o_2 \ldots o_t, s_t = i \middle| \lambda\right)$
  - The probability that the HMM is in state *i* at time *t* having generating partial observation $\boldsymbol{o}_1\boldsymbol{o}_2\ldots\boldsymbol{o}_t$

Basic Problem 1 of HMM
- The Forward Procedure (cont.)

- Algorithm

1. Initialization $\alpha_1(i) = \pi_i b_i(\boldsymbol{o}_1),\ 1 \le i \le N$

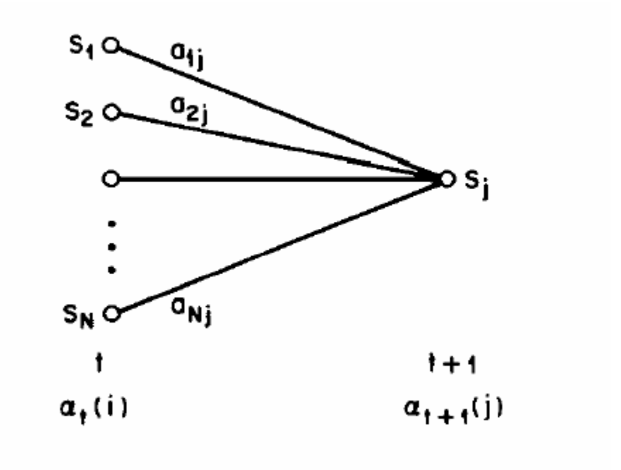
2. Induction $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(\boldsymbol{o}_{t+1}),\ \ 1 \le t \le T\text{-}1, 1 \le j \le N$

3. Termination $P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$

  - Complexity: $O(N^2T)$

$$\text{MUL} \quad : N(N+1)(T-1)+N \quad \approx \quad N^2T$$

$$\text{ADD} \quad : (N-1)N(T-1)+(N-1) \approx N^2T$$



- Based on the lattice (trellis) structure
  - Computed in a *time-synchronous* fashion from *left-to-right*, where each cell for time *t* is completely computed before proceeding to time *t+1*
- All state sequences, regardless how long previously, merge to *N* nodes (states) at each time instance *t*

Basic Problem 1 of HMM
- The Forward Procedure (cont.)

$$
\begin{aligned}
\alpha_t(j) &= P\left(o_1 o_2 \ldots o_t, s_t = j \middle| \lambda\right) \\
&= P\left(o_1 o_2 \ldots o_t \middle| s_t = j, \lambda\right) P\left(s_t = j \middle| \lambda\right) \\
&= P\left(o_1 o_2 \ldots o_{t-1} \middle| s_t = j, \lambda\right) P\left(o_t \middle| s_t = j, \lambda\right) P\left(s_t = j \middle| \lambda\right) \\
&= P\left(o_1 o_2 \ldots o_{t-1}, s_t = j \middle| \lambda\right) P\left(o_t \middle| s_t = j, \lambda\right) \\
&= P\left(o_1 o_2 \ldots o_{t-1}, s_t = j \middle| \lambda\right) b_j(o_t) \\
&= \left[\sum_{i=1}^{N} P\left(o_1 o_2 \ldots o_{t-1}, s_{t-1} = i, s_t = j \middle| \lambda\right)\right] b_j(o_t) \\
&= \left[\sum_{i=1}^{N} P\left(o_1 o_2 \ldots o_{t-1}, s_{t-1} = i \middle| \lambda\right) P\left(s_t = j \middle| o_1 o_2 \ldots o_{t-1}, s_{t-1} = i, \lambda\right)\right] b_j(o_t) \\
&= \left[\sum_{i=1}^{N} P\left(o_1 o_2 \ldots o_{t-1}, s_{t-1} = i \middle| \lambda\right) P\left(s_t = j \middle| s_{t-1} = i, \lambda\right)\right] b_j(o_t) \\
&= \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(o_t)
\end{aligned}
$$

$P(A,B) = P(B|A)P(A)$

output independent assumption

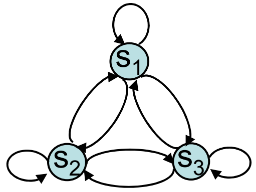
$P(B|A)P(A) = P(A,B)$

$P(o_t|s_t = j, \lambda) = b_j(o_t)$

$P(A) \sum_{all\ B} P(A,B)$

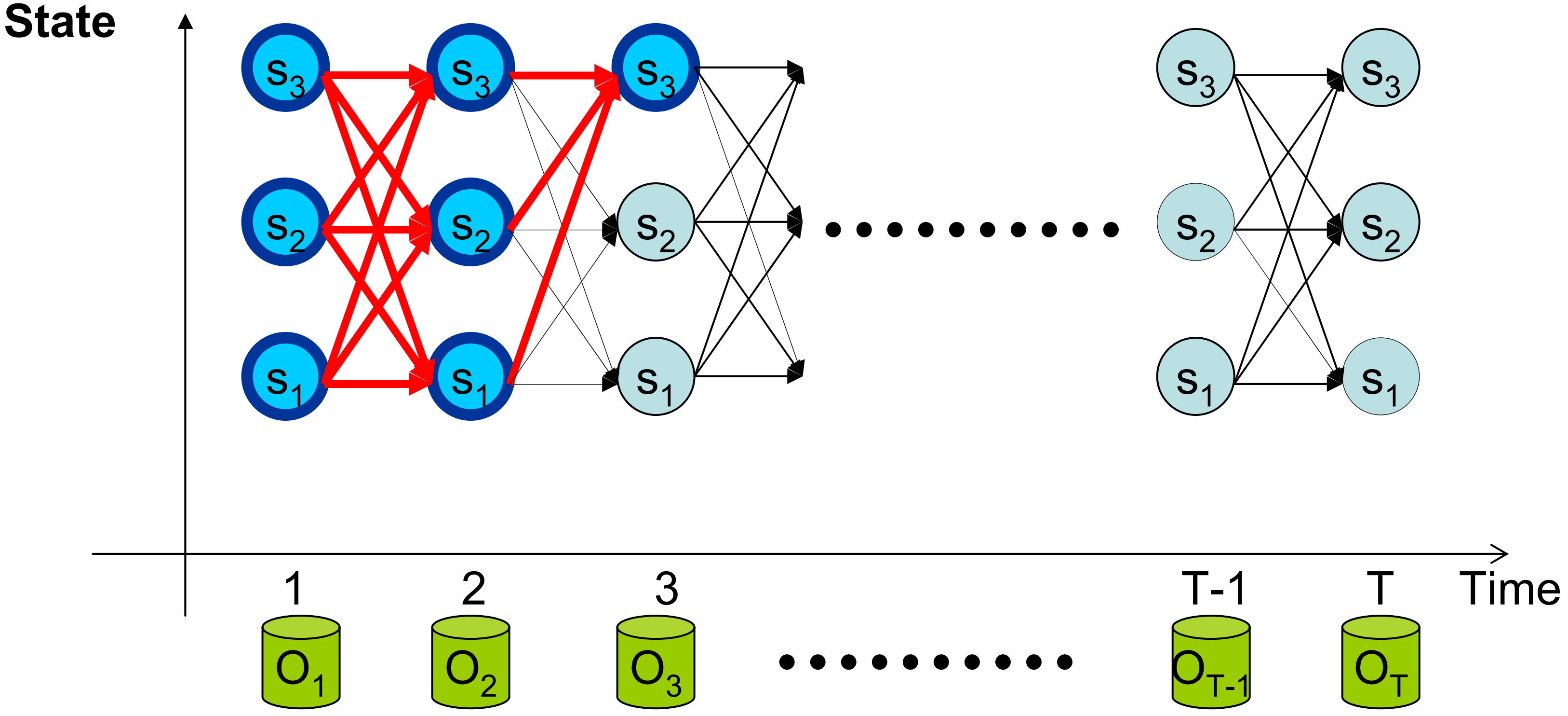
first-order Markov assumption

Basic Problem 1 of HMM
- The Forward Procedure (cont.)



- $\alpha_3(3)=P(\boldsymbol{o}_1,\boldsymbol{o}_2,\boldsymbol{o}_3,s_3=\mathbf{3}|\lambda)$

$$=[\alpha_2(1)^*a_{13}+\alpha_2(2)^*a_{23}+\alpha_2(3)^*a_{33}]b_3(\boldsymbol{o}_3)$$

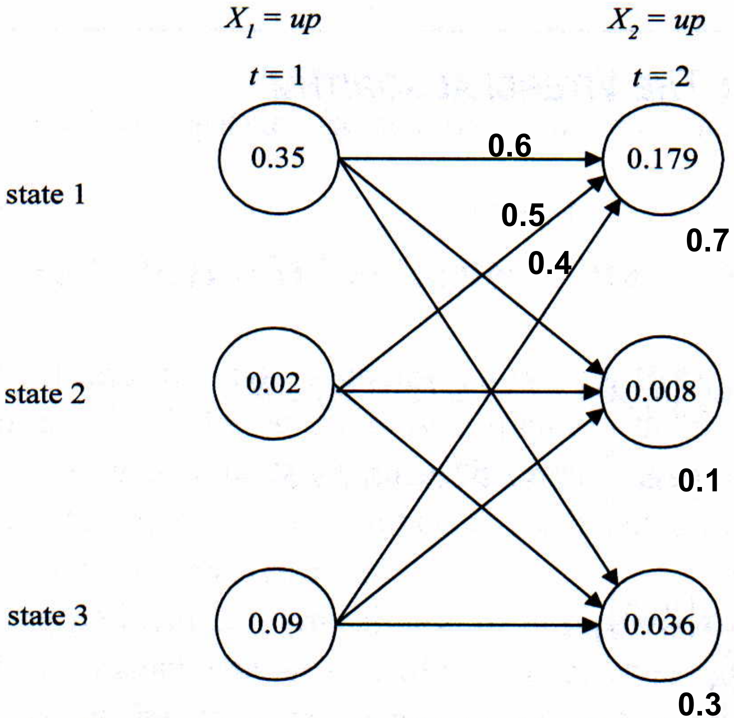


s_i **means $b_j(o_t)$ has been computed**

$\xrightarrow{a_{ij}}$ **means a_{ij} has been computed**

Basic Problem 1 of HMM
- The Forward Procedure (cont.)

- A three-state Hidden Markov Model for the *Dow Jones Industrial average*



(0.6*0.35+0.5*0.02+0.4*0.009)*0.7
=0.1792

Figure 8.4 The forward trellis computation for the HMM of the Dow Jones Industrial average.

Basic Problem 1 of HMM
- The Backward Procedure

- Backward variable : $\beta_t(i)=P(\boldsymbol{o}_{t+1},\boldsymbol{o}_{t+2},\ldots..,\boldsymbol{o}_T|s_t=i\ ,\ \lambda)$

$$1.\ \text{Initialization}: \beta_{\mathrm{T}}(i)=1,\ 1\le i\le N$$

$$2.\ \text{Induction}: \beta_t(i)=\sum_{j=1}^{N} a_{ij}b_j(\boldsymbol{o}_{t+1})\beta_{t+1}(j),\ 1\le t\le T\text{-}1, 1\le i\le N$$

$$3.\ \text{Termination}: P(\boldsymbol{O}|\lambda)=\sum_{j=1}^{N}\pi_j b_j(\boldsymbol{o}_1)\beta_1(j)$$

$$\text{Complexity MUL}: 2N^2(T\text{-}1)+2N \approx N^2T\ ;$$

$$\text{ADD}: (N\text{-}1)N(T\text{-}1)+N\approx N^2T$$

Basic Problem 1 of HMM
- Backward Procedure (cont.)

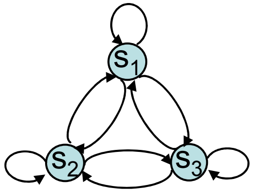
- Why $P(\boldsymbol{O}, s_t = i|\lambda) = \alpha_t(i)\,\beta_t(i)$?

$$\begin{aligned}
&\alpha_t(i)\,\beta_t(i) \\
&= P(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t, s_t = i|\lambda) \cdot P(\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, \ldots, \boldsymbol{o}_T | s_t = \boldsymbol{i}, \lambda) \\
&= P(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t | s_t = \boldsymbol{i}, \lambda) P(s_t = i|\lambda) P(\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, \ldots, \boldsymbol{o}_T | s_t = \boldsymbol{i}, \lambda) \\
&= P(\boldsymbol{o}_1, \ldots, \boldsymbol{o}_t, \ldots, \boldsymbol{o}_T | s_t = \boldsymbol{i}, \lambda) P(s_t = i|\lambda) \\
&= P(\boldsymbol{o}_1, \ldots, \boldsymbol{o}_t, \ldots, \boldsymbol{o}_T, s_t = i|\lambda) \\
&= P(\boldsymbol{O}, s_t = i|\lambda)
\end{aligned}$$

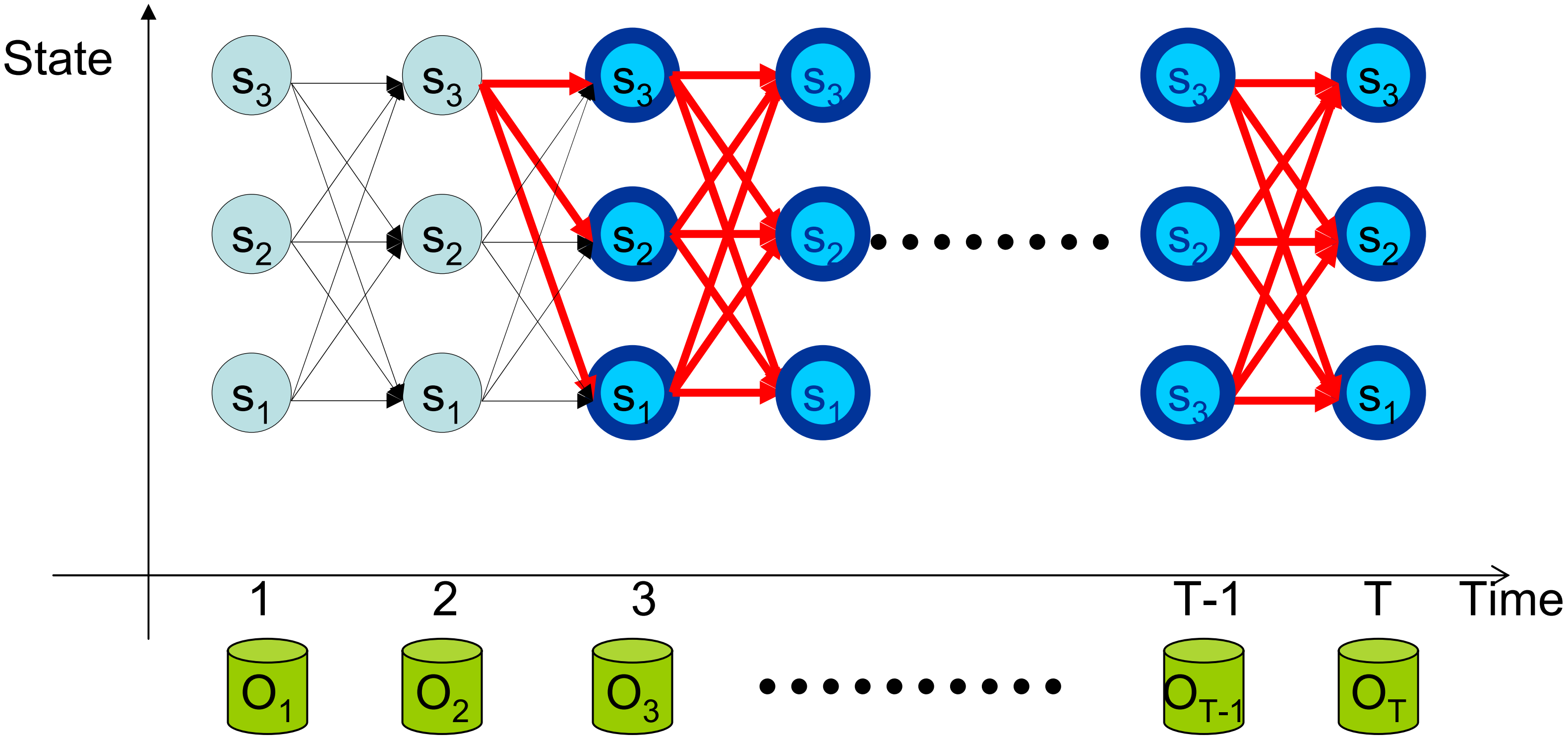
- $P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} P(\boldsymbol{O}, s_t = i|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\,\beta_t(i)$

Basic Problem 1 of HMM

- The Backward Procedure (cont.)



- $\beta_2(3)=P(o_3,o_4,\ldots,\ o_T|s_2=3,\lambda)$
  $=a_{31}{*}\ b_1(o_3){*}\beta_3(1)\ +a_{32}{*}\ b_2(o_3){*}\beta_3(2)+a_{33}{*}\ b_1(o_3){*}\beta_3(3)$



Basic Problem 2 of HMM

How to choose an optimal state sequence $S=(s_1, s_2, \ldots\ldots, s_T)$?

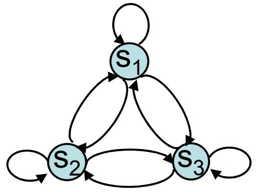
- The first optimal criterion: **Choose the states s_t are *individually* most likely at each time *t***

  Define a posteriori probability variable $\gamma_t(i) = P(s_t = i | \boldsymbol{O}, \boldsymbol{\lambda})$

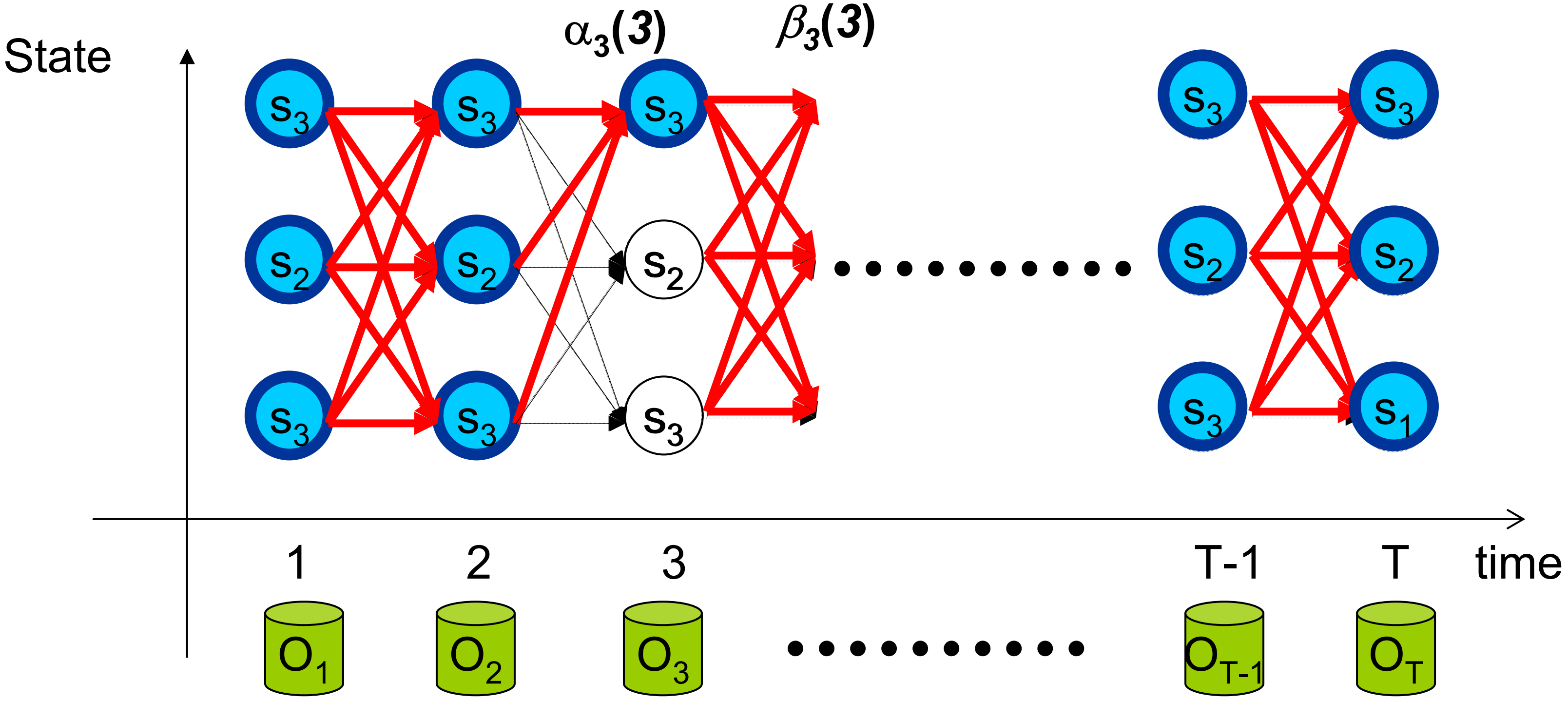
$$\gamma_t(i) = \frac{P(s_t = i, \boldsymbol{O} | \boldsymbol{\lambda})}{P(\boldsymbol{O} | \boldsymbol{\lambda})} = \frac{P(s_t = i, \boldsymbol{O} | \boldsymbol{\lambda})}{\sum_{m=1}^{N} P(s_t = m, \boldsymbol{O} | \boldsymbol{\lambda})} = \frac{\alpha_t(i)\,\beta_t(i)}{\sum_{m=1}^{N} \alpha_t(m)\,\beta_t(m)}$$

  - Solution : $s_t^* = \arg_i \max [\gamma_t(i)],\ 1 \le t \le T$
    - Problem: maximizing the probability at each time *t* individually $\boldsymbol{S}^* = s_1^* s_2^* \ldots s_T^*$ may not be a valid sequence (e.g. $a_{s_t^* s_{t+1}^*} = 0$)

Basic Problem 2 of HMM (cont.)



- $P(s_3 = 3 , \boldsymbol{O} \mid \lambda)=\alpha_3(3)*\beta_3(3)$



Basic Problem 2 of HMM
- The Viterbi Algorithm

- The second optimal criterion: The Viterbi algorithm can be regarded as the dynamic programming algorithm applied to the HMM or as a modified forward algorithm
  - Instead of summing up probabilities from different paths coming to the same destination state, the Viterbi algorithm picks and remembers the best path
    - Find a single optimal state sequence $\boldsymbol{S}=(s_1,s_2,\ldots\ldots, s_T)$
      - How to find the second, third, etc., optimal state sequences (difficult ?)
  - The Viterbi algorithm also can be illustrated in a trellis framework similar to the one for the forward algorithm
    - State-time trellis diagram

Basic Problem 2 of HMM

- The Viterbi Algorithm (cont.)

- Algorithm

Find a best state sequence $\boldsymbol{S}=(s_1, s_2, .., s_T)$ for a given observation $\boldsymbol{O}=(\boldsymbol{o}_1, \boldsymbol{o}_2, .., \boldsymbol{o}_T)$?

Define a new variable

$$\delta_t(i) = \max_{s_1, s_2, .., s_{t-1}} P\left[s_1, s_2, .., s_{t-1}, s_t = \boldsymbol{i}, \boldsymbol{o}_1, \boldsymbol{o}_2, .., \boldsymbol{o}_t \middle| \lambda\right]$$

= the best score along a single path at time t, which accounts for the first t observation and ends in state i

By induction $\therefore \delta_{t+1}(j) = \left[\max_{1 \le i \le N} \delta_t(i) a_{ij}\right] b_j(\boldsymbol{o}_{t+1})$

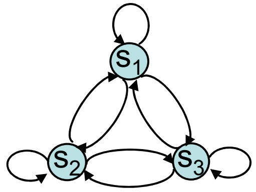
$$\psi_{t+1}(j) = \arg\max_{1 \le i \le N} \delta_t(i) a_{ij} \quad \text{.... For backtracing}$$

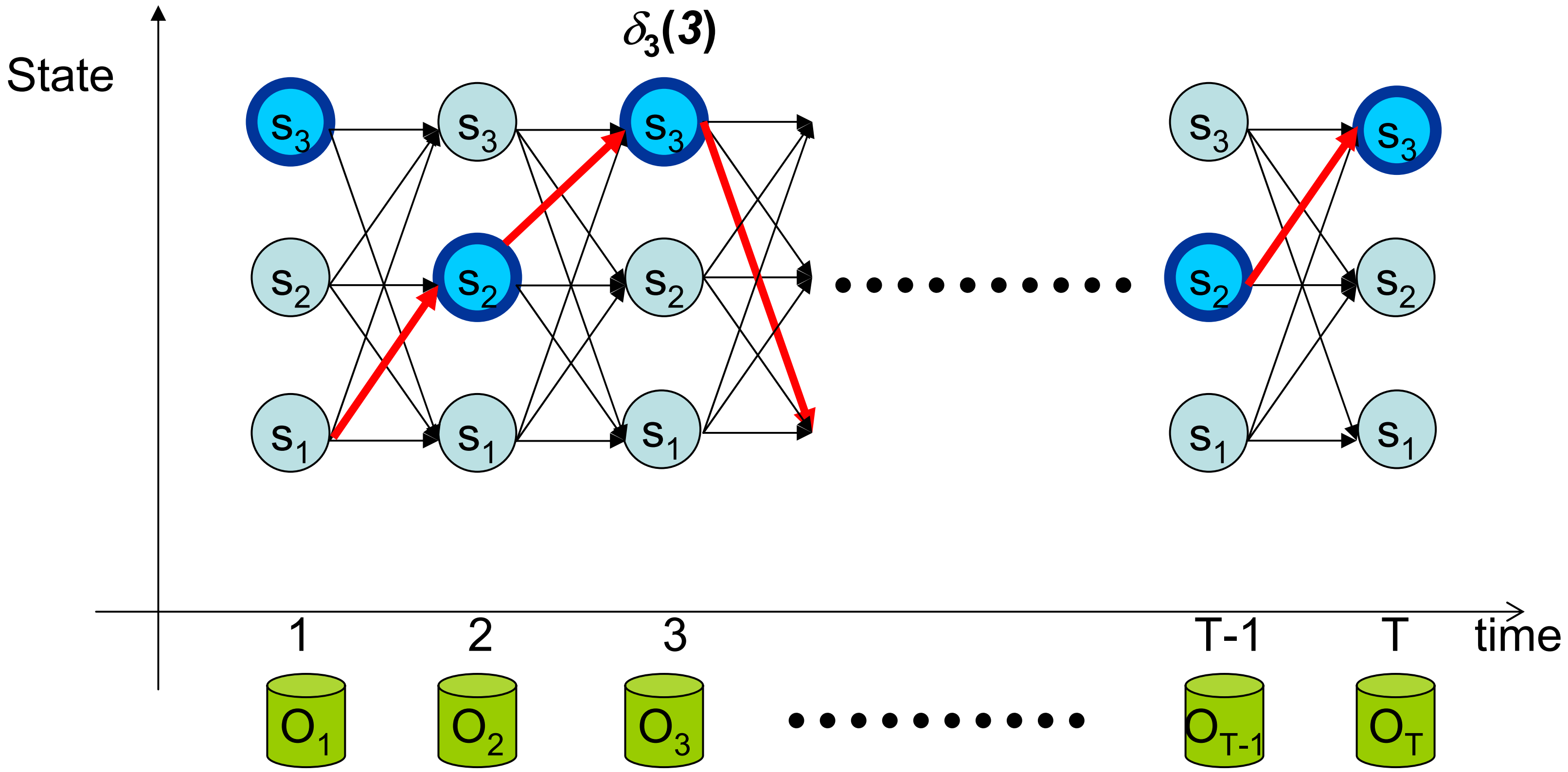
We can backtrace from $s_T^* = \arg\max_{1 \le i \le N} \delta_T(i)$

  - Complexity: $O(N^2T)$

Basic Problem 2 of HMM

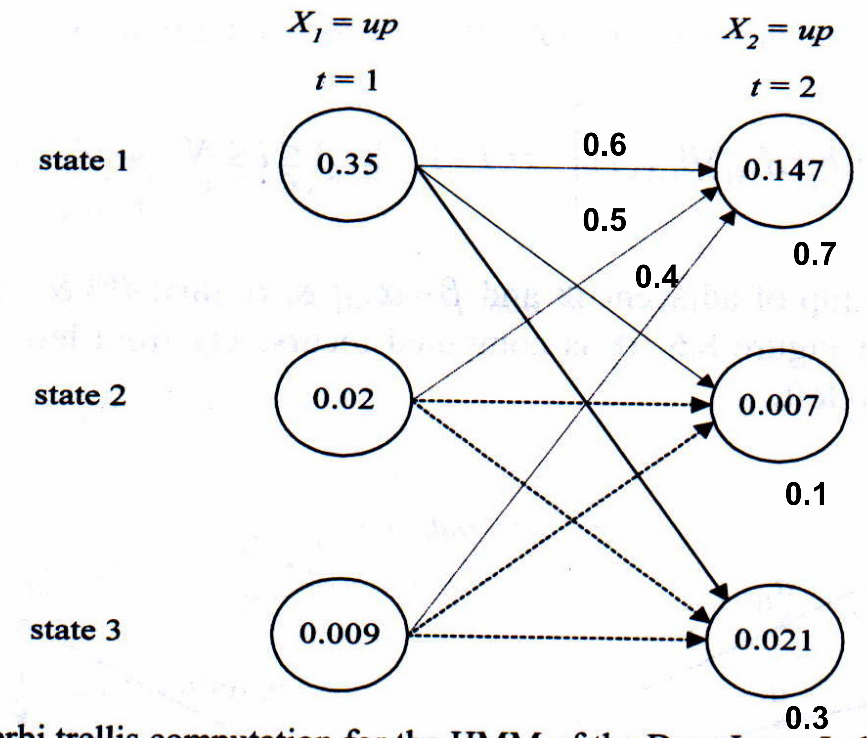
- The Viterbi Algorithm (cont.)





Basic Problem 2 of HMM
- The Viterbi Algorithm (cont.)

- A three-state Hidden Markov Model for the *Dow Jones Industrial average*



(0.6*0.35)*0.7
=0.147

Figure 8.5 The Viterbi trellis computation for the HMM of the Dow Jones Industrial average.

Basic Problem 2 of HMM
- The Viterbi Algorithm (cont.)

- Algorithm in the logarithmic form

Find a best state sequence $\boldsymbol{S}=(s_1,s_2,..,s_T)$ for a given observation $\boldsymbol{O}=(\boldsymbol{o}_1,\boldsymbol{o}_2,..,\boldsymbol{o}_T)$?

Define a new variable

$$\delta_t(i)=\max_{s_1,s_2,..,s_{t-1}} \log P[s_1,s_2,..,s_{t-1},s_t=i,\boldsymbol{o}_1,\boldsymbol{o}_2,..,\boldsymbol{o}_t|\lambda]$$

= the best score along a single path at time t, which accounts for the first t observation and ends in state i

By induction $\therefore \delta_{t+1}(j)=\left[\max_{1\le i\le N}\left(\delta_t(i)+\log a_{ij}\right)\right]+\log b_j(\boldsymbol{o}_{t+1})$

$$\psi_{t+1}(j)=\arg\max_{1\le i\le N}\left(\delta_t(i)+\log a_{ij}\right)$$ .... For backtracing

We can backtrace from $s_T^*=\arg\max_{1\le i\le N}\delta_T(i)$

Homework-1

- A three-state Hidden Markov Model for the *Dow Jones Industrial average*

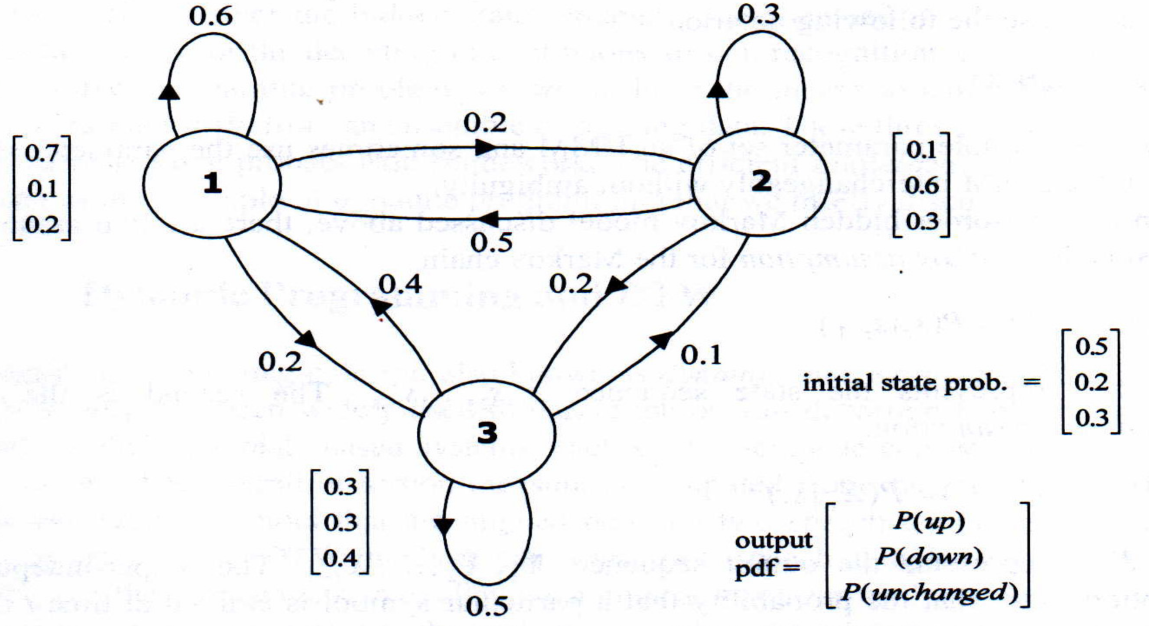


Figure 8.2 A hidden Markov model for the Dow Jones Industrial average. The three states no longer have deterministic meanings as in the Markov chain illustrated in Figure 8.1.

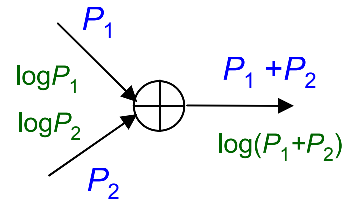
- *Find the probability:*

  $P(up, up, unchanged, down, unchanged, down, up|\lambda)$

- *Fnd the optimal state sequence of the model which generates the observation sequence:* (*up, up, unchanged, down, unchanged, down, up*)

Probability Addition in F-B Algorithm

- In Forward-backward algorithm, operations usually implemented in logarithmic domain



- Assume that we want to add P_1 and P_2

$$\text{if } P_1 \geq P_2$$

$$\log_b(P_1 + P_2) = \log P_1 + \log_b\left(1 + b^{\log_b P_2 - \log_b P_1}\right)$$

$$\text{else}$$

$$\log_b(P_1 + P_2) = \log P_2 + \log_b\left(1 + b^{\log_b P_1 - \log_b P_2}\right)$$

The values of $\log_b\left(1 + b^x\right)$ can be saved in in a table to speedup the operations

Probability Addition in F-B Algorithm (cont.)

- An example code

```
#define LZERO  (-1.0E10)   // ~log(0)
#define LSMALL (-0.5E10)   // log values < LSMALL are set to LZERO
#define minLogExp  -log(-LZERO)

double LogAdd(double x, double y)
{
 double temp,diff,z;
  if (x<y)
  {
     temp = x; x = y; y = temp;
  }
  diff = y-x; //notice that diff <= 0
  if (diff<minLogExp) // if y’ is far smaller than x’
     return  (x<LSMALL) ?  LZERO:x;
  else
   {
     z = exp(diff);
     return x+log(1.0+z);
  }
}
```

Basic Problem 3 of HMM

Intuitive View

- How to adjust (re-estimate) the model parameter $\lambda=(\boldsymbol{A},\boldsymbol{B},\pi)$ to maximize $P(\boldsymbol{O}|\lambda)$?
  - The most difficult of the three problems, because there is no known analytical method that maximizes the joint probability of the training data in a close form
  - The data is incomplete because of the hidden state sequences
  - Well-solved by the *Baum-Welch* (known as *forward-backward*) algorithm and *EM* (*Expectation-Maximization*) algorithm
    - Iterative update and improvement

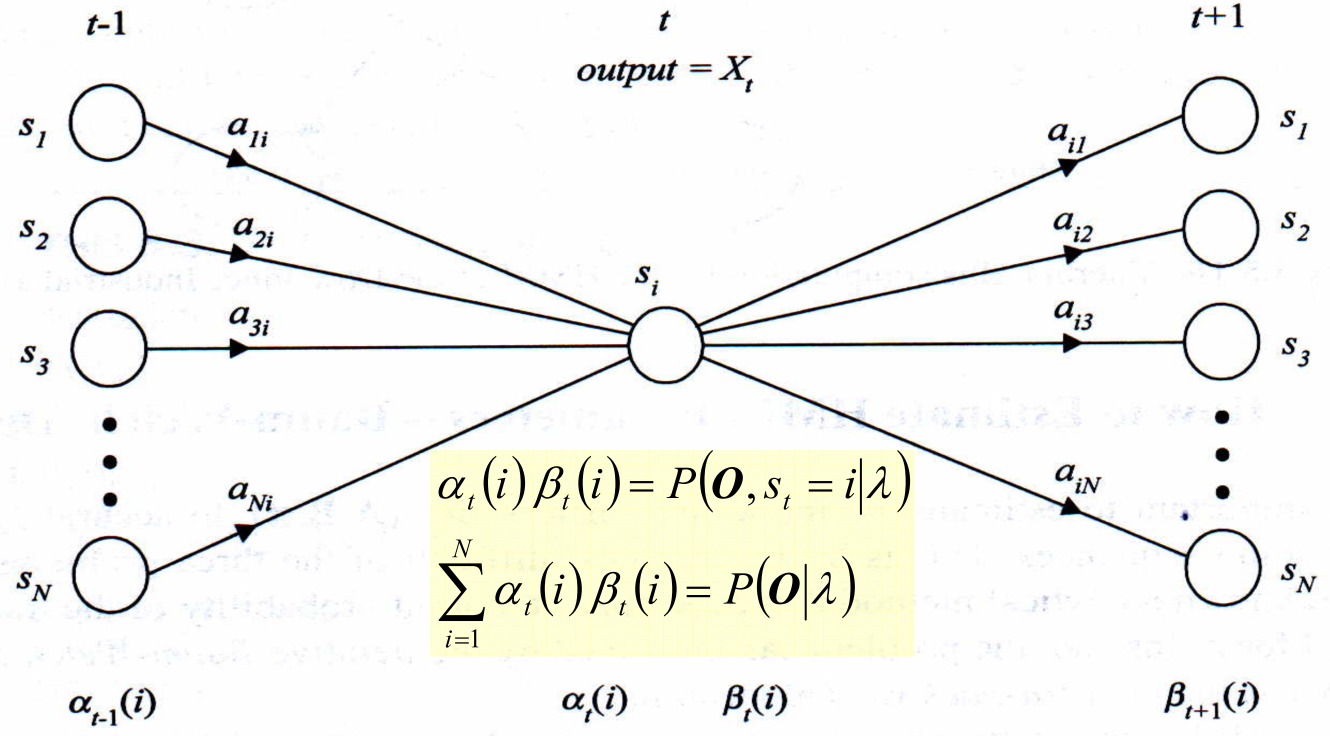
Basic Problem 3 of HMM

Intuitive View (cont.)

- Relation between the forward and backward variables

$$\alpha_t(i) = P(\boldsymbol{o}_1, \boldsymbol{o}_2, ..., \boldsymbol{o}_t, s_t = i|\lambda) = \left[\sum_{j=1}^{N} \alpha_{t-1}(j) a_{ji}\right] b_i(\boldsymbol{o}_t)$$

$$\beta_t(i) = P(\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, ..., \boldsymbol{o}_T | s_t = i, \lambda) = \sum_{j=1}^{N} \beta_{t+1}(j) b_j(\boldsymbol{o}_{t+1}) a_{ij}$$



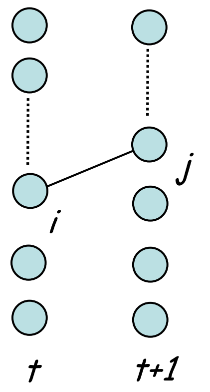
$$\alpha_t(i)\,\beta_t(i) = P(\boldsymbol{O}, s_t = i|\lambda)$$

$$\sum_{i=1}^{N} \alpha_t(i)\,\beta_t(i) = P(\boldsymbol{O}|\lambda)$$

Figure 8.6 The relationship of α_{t-1} and α_t and β_t and β_{t+1} in the forward-backward algorithm.

Basic Problem 3 of HMM

Intuitive View (cont.)



- Define a new variable:

$$\xi_t(i,j) = P\left(s_t = i, s_{t+1} = j \middle| \boldsymbol{O}, \boldsymbol{\lambda}\right)$$

  - Probability being at state *i* at time *t* and at state *j* at time *t+1*

$$p(A|B) = \frac{p(A,B)}{P(B)}$$

$$\xi_t(i,j) = \frac{P\left(s_t = i, s_{t+1} = j, \boldsymbol{O} \middle| \boldsymbol{\lambda}\right)}{P\left(\boldsymbol{O} \middle| \boldsymbol{\lambda}\right)}$$

$$= \frac{\alpha_t(i) a_{ij} b_j(\boldsymbol{o}_{t+1}) \beta_{t+1}(j)}{P\left(\boldsymbol{O} \middle| \boldsymbol{\lambda}\right)} = \frac{\alpha_t(i) a_{ij} b_j(\boldsymbol{o}_{t+1}) \beta_{t+1}(j)}{\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_t(m) a_{mn} b_n(\boldsymbol{o}_{t+1}) \beta_{t+1}(n)}$$

- Recall the posteriori probability variable:

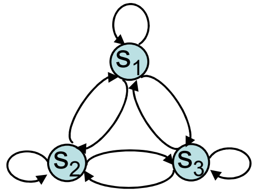
$$\gamma_t(i) = P\left(s_t = i \middle| \boldsymbol{O}, \lambda\right)$$

Note : $\gamma_t(i)$ also can be represente d as $\frac{\alpha_t(i)\beta_t(i)}{\sum_{m=1}^{N} \alpha_t(m)\beta_t(m)}$

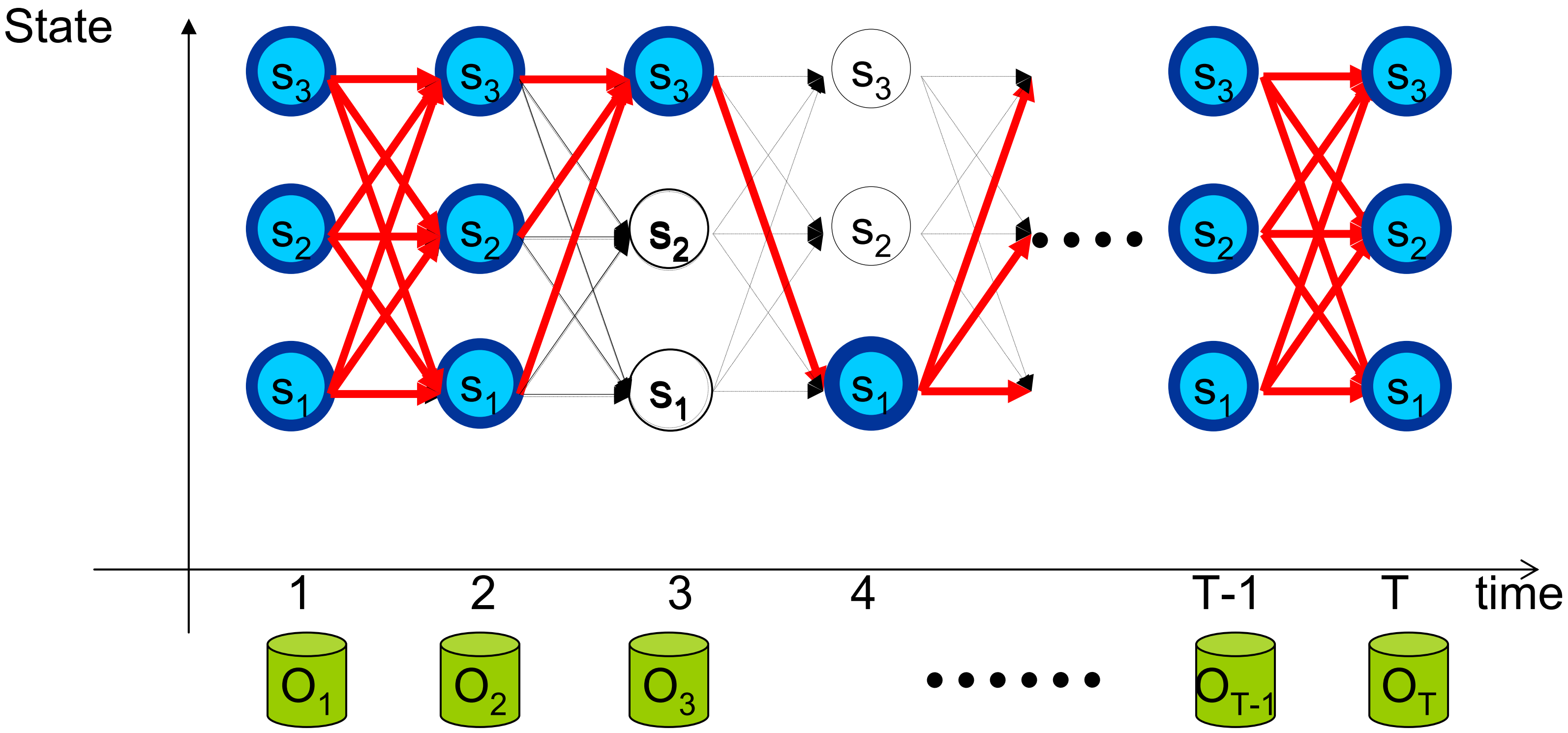
$$\gamma_t(i) = \sum_{j=1}^{N} P\left(s_t = i, s_{t+1} = j \middle| \boldsymbol{O}, \lambda\right) = \sum_{j=1}^{N} \xi_t(i,j) \quad (\text{for } t < T)$$

Basic Problem 3 of HMM

Intuitive View (cont.)



- $P(s_3 = 3, s_4 = 1, \mathbf{O} \mid \lambda) = \alpha_3(3) * a_{31} * b_1(o_4) * \beta_1(4)$



Basic Problem 3 of HMM
Intuitive View (cont.)

- $\xi_t(i,j) = P(s_t = i, s_{t+1} = j | \boldsymbol{O}, \boldsymbol{\lambda})$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{expected number of transitions from state } i \text{ to state } j \text{ in } \boldsymbol{O}$$

- $\gamma_t(i) = P(s_t = i | \boldsymbol{O}, \lambda)$

$$\sum_{t=1}^{T-1} \gamma_t(i) = \sum_{t=1}^{T-1} \sum_{j=1}^{N} \xi_t(i,j) = \text{expected number of transitions from state } i \text{ in } \boldsymbol{O}$$

- A set of reasonable re-estimation formula for {***A***, π} is

$$\begin{aligned} \overline{\pi}_i &= \text{expected freqency (number of times) in state } i \text{ at time } t = 1 \\ &= \gamma_1(i) \end{aligned}$$

$$\overline{a}_{ij} = \frac{\text{expected number of transition from state } i \text{ to state } j}{\text{expected number of transition from state } i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

Formulae for Single Training Utterance

Basic Problem 3 of HMM

Intuitive View (cont.)

- A set of reasonable re-estimation formula for $\{B\}$ is
  - For discrete and finite observation $b_j(\boldsymbol{v}_k)=P(\boldsymbol{o}_t=\boldsymbol{v}_k|\boldsymbol{s}_t=j)$

$$\bar{b}_j(\boldsymbol{v}_k)=\bar{P}(\boldsymbol{o}=\boldsymbol{v}_k|s=j)=\frac{\text{expected number of times in state } j \text{ and observing symbol } \boldsymbol{v}_k}{\text{expected number of times in state } j}=\frac{\sum_{\substack{t=1\\ \text{such that } \boldsymbol{o}=\boldsymbol{v}_k}}^{T}\gamma_t(j)}{\sum_{t=1}^{T}\gamma_t(j)}$$

  - For continuous and infinite observation $b_j(\boldsymbol{v})=f_{\boldsymbol{O}|\boldsymbol{S}}(\boldsymbol{o}_t=\boldsymbol{v}|\boldsymbol{s}_t=j)$,

$$\bar{b}_j(\boldsymbol{v})=\sum_{k=1}^{M}\bar{c}_{jk}N(\boldsymbol{v};\bar{\boldsymbol{\mu}}_{jk},\bar{\boldsymbol{\Sigma}}_{jk})=\sum_{k=1}^{M}\bar{c}_{jk}\left(\frac{1}{\left(\sqrt{2\pi}\right)^{L}\left|\bar{\boldsymbol{\Sigma}}_{jk}\right|^{1/2}}\exp\left(-\frac{1}{2}(\boldsymbol{v}-\bar{\boldsymbol{\mu}}_{jk})^{t}\bar{\Sigma}_{jk}^{-1}(\boldsymbol{v}-\bar{\boldsymbol{\mu}}_{jk})\right)\right)$$

Modeled as a mixture of multivariate Gaussian distributions

Basic Problem 3 of HMM
Intuitive View (cont.)

$$p(A|B)=\frac{p(A,B)}{P(B)}$$

- For continuous and infinite observation (Cont.)
  - Define a new variable $\gamma_t(j,k)$
    - $\gamma_t(j,k)$ is the probability of being in state *j* at time *t* with the *k*-th mixture component accounting for $\boldsymbol{o}_t$

$$\gamma_t(j,k)=P(s_t=j,m_t=k|\boldsymbol{O},\boldsymbol{\lambda})$$

$$=P(s_t=j|\boldsymbol{O},\boldsymbol{\lambda})P(m_t=k|s_t=j,\boldsymbol{O},\boldsymbol{\lambda})$$

$$=\gamma_t(j)P(m_t=k|s_t=j,\boldsymbol{O},\boldsymbol{\lambda})$$

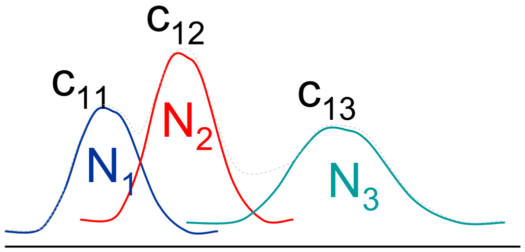
$$=\gamma_t(j)\frac{P(m_t=k,\boldsymbol{O}|s_t=j,\boldsymbol{\lambda})}{P(\boldsymbol{O}|s_t=j,\boldsymbol{\lambda})}$$

$$=\gamma_t(j)\frac{P(m_t=k|s_t=j,\boldsymbol{\lambda})P(\boldsymbol{O}|s_t=j,m_t=k,\boldsymbol{\lambda})}{P(\boldsymbol{O}|s_t=j,\boldsymbol{\lambda})}$$

...... (observati on - independen t assumption is applied)

$$=\gamma_t(j)\frac{P(m_t=k|s_t=j,\boldsymbol{\lambda})P(\boldsymbol{o}_t|s_t=j,m_t=k,\boldsymbol{\lambda})}{P(\boldsymbol{o}_t|s_t=j,\boldsymbol{\lambda})}$$

$$=\left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{s=1}^{N}\alpha_t(s)\beta_t(s)}\right]\left[\frac{c_{jk}N(\boldsymbol{o}_t;\boldsymbol{\mu}_{jk},\boldsymbol{\Sigma}_{jk})}{\sum_{m=1}^{M}c_{jm}N(\boldsymbol{o}_t;\boldsymbol{\mu}_{jm},\boldsymbol{\Sigma}_{jm})}\right]$$



Distribution for State 1

$$\text{Note}: \gamma_t(j)=\sum_{m=1}^{M}\gamma_t(j,m)$$

Basic Problem 3 of HMM

Intuitive View (cont.)

– For continuous and infinite observation (Cont.)

$$\overline{c}_{jk} = \frac{\text{expected number of times in state } j \text{ and mixture } k}{\text{expected number of times in state } j} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t(j,m)}$$

$$\overline{\boldsymbol{\mu}}_{jk} = \text{weighted average (mean) of observations at state } j \text{ and mixture } k = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \boldsymbol{o}_t}{\sum_{t=1}^{T} \gamma_t(j,k)}$$

$$\overline{\boldsymbol{\Sigma}}_{jk} = \text{weighted covariance of observations at state } j \text{ and mixture } k$$

$$= \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)^t}{\sum_{t=1}^{T} \gamma_t(j,k)}$$

Formulae for Single Training Utterance

Basic Problem 3 of HMM
Intuitive View (cont.)

- For continuous and infinite observation (Cont.)

$$\bar{\pi}_i = \text{expected freqency (number of times) in state } i \text{ at time } (t=1) = \frac{1}{L}\sum_{l=1}^{L}\gamma_1^l(i)$$

$$\bar{a}_{ij} = \frac{\text{expected number of transition from state } i \text{ to state } j}{\text{expected number of transition from state } i} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T\text{-}1}\xi_t^l(i,j)}{\sum_{l=1}^{L}\sum_{t=1}^{T\text{-}1}\gamma_t^l(i)}$$

$$\bar{c}_{jk} = \frac{\text{expected number of times in state } j \text{ and mixture } k}{\text{expected number of times in state } j} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T}\gamma_t^l(j,k)}{\sum_{l=1}^{L}\sum_{t=1}^{T}\sum_{m=1}^{M}\gamma_t^l(j,m)}$$

$$\bar{\boldsymbol{\mu}}_{jk} = \text{weighted average (mean) of observations at state } j \text{ and mixture } k = \frac{\sum_{l=1}^{L}\sum_{\text{t}=1}^{\text{T}}\gamma_t^l(j,k)\cdot\mathbf{o}_t}{\sum_{l=1}^{L}\sum_{\text{t}=1}^{\text{T}}\gamma_t^l(j,k)}$$

$$\bar{\boldsymbol{\Sigma}}_{jk} = \text{weighted covariance of observatio ns at state } j \text{ and mixture } k$$

$$= \frac{\sum_{l=1}^{L}\sum_{\text{t}=1}^{\text{T}}\gamma_t^l(j,k)\cdot\left(\boldsymbol{o}_t-\bar{\boldsymbol{\mu}}_{jk}\right)\left(\boldsymbol{o}_t-\bar{\boldsymbol{\mu}}_{jk}\right)^t}{\sum_{l=1}^{L}\sum_{\text{t}=1}^{\text{T}}\gamma_t^l(j,k)}$$

Formulae for Multiple (L) Training Utterances

Basic Problem 3 of HMM
Intuitive View (cont.)

- For discrete and finite observation (cont.)

$$\overline{\pi}_i = \text{expected freqency (number of times) in state } i \text{ at time } (t=1) = \frac{1}{L}\sum_{l=1}^{L}\gamma_1^l(i)$$

$$\overline{a}_{ij} = \frac{\text{expected number of transition from state } i \text{ to state } j}{\text{expected number of transition from state } i} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T\text{-}1}\xi_t^l(i,j)}{\sum_{l=1}^{L}\sum_{t=1}^{T\text{-}1}\gamma_t^l(i)}$$

$$\overline{b}_j(\boldsymbol{v}_k) = \overline{P}(\boldsymbol{o}=\boldsymbol{v}_k|s=j) = \frac{\text{expected number of times in state } j \text{ and observing symbol } \boldsymbol{v}_k}{\text{expected number of times in state } j} = \frac{\sum_{l=1}^{L}\ \sum_{\substack{\mathrm{t}=1 \\ \text{such that } \boldsymbol{o}=\boldsymbol{v}_k}}^{\mathrm{T}}\gamma_t^l(j)}{\sum_{l=1}^{L}\sum_{\mathrm{t}=1}^{\mathrm{T}}\gamma_t^l(j)}$$

Formulae for Multiple (L) Training Utterances

Semicontinuous HMMs

- The HMM state mixture density functions are tied together across all the models to form a set of shared kernels
  - The semicontinuous or tied-mixture HMM

$$b_j(\boldsymbol{o}) = \sum_{k=1}^{M} b_j(k) f(\boldsymbol{o}|v_k) = \sum_{k=1}^{M} \underbrace{b_j(k)}_{} \underbrace{N(\boldsymbol{o}, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}_{}$$

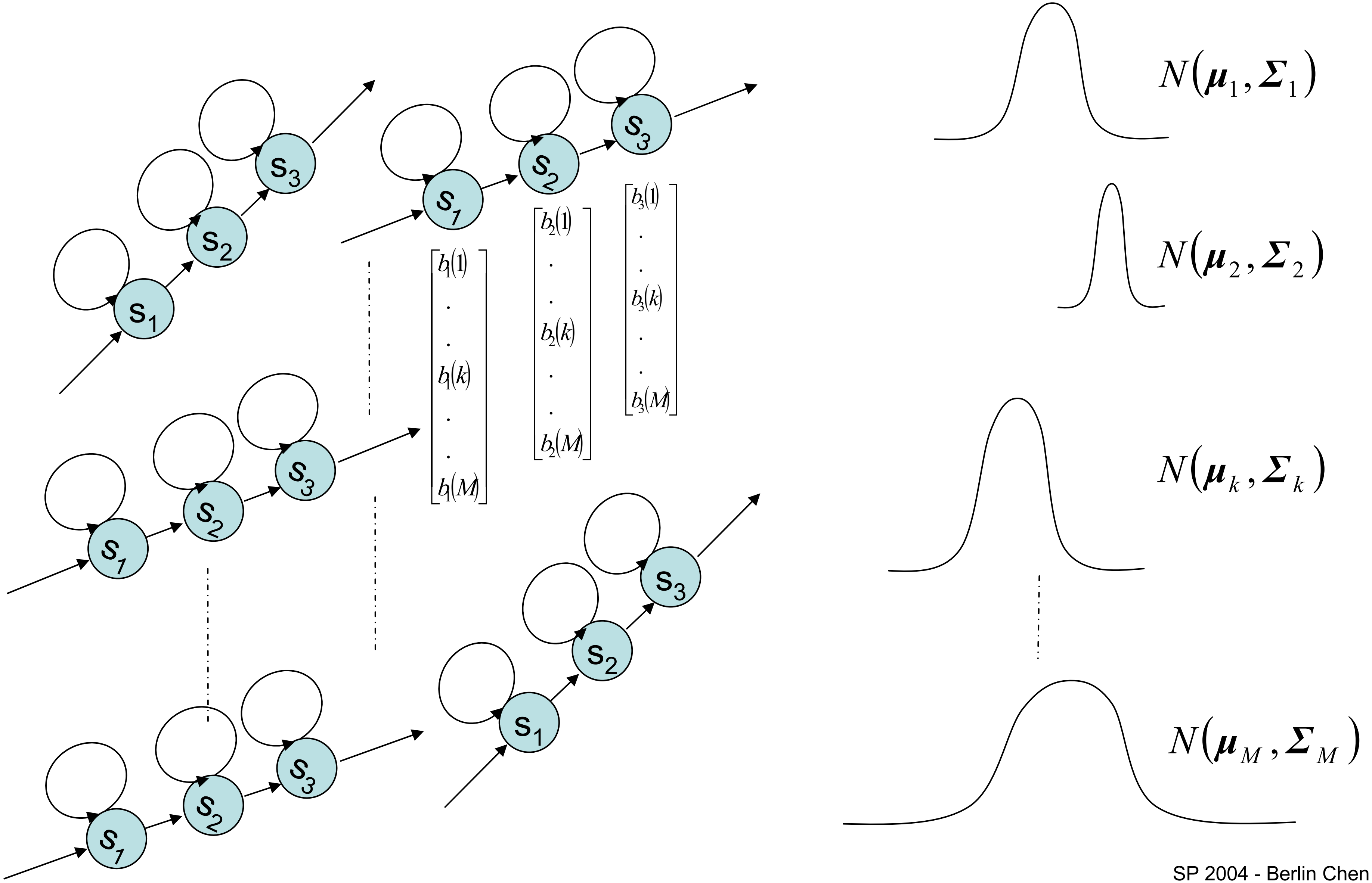
state output
Probability of state *j*

k-th mixture weight
t of state *j*
(discrete, model-dependent)

k-th mixture density function or *k*-th codeword
(shared across HMMs, *M* is very large)

  - A combination of the discrete HMM and the continuous HMM
    - A combination of *discrete* model-dependent weight coefficients and *continuous* model-independent codebook probability density functions
  - Because *M* is large, we can simply use the *L* most significant values $f(\boldsymbol{o}|v_k)$
    - Experience showed that *L* is *1~3%* of *M* is adequate
  - Partial tying of $f(\boldsymbol{o}|v_k)$ for different phonetic class

Semicontinuous HMMs (cont.)



Initialization of HMM

- A good initialization of HMM training : Segmental K-Means Segmentation into States
  - Assume that we have a training set of observations and an initial estimate of all model parameters
  - Step 1 : The set of training observation sequences is segmented into states, based on the initial model (finding the optimal state sequence by *Viterbi* Algorithm)
  - Step 2 :
    - For discrete density HMM (using M-codeword codebook)

$$\bar{b}_j(k) = \frac{\text{the number of vectors with codebook index } k \text{ in state } j}{\text{the number of vectors in state } j}$$

    - For continuous density HMM (M Gaussian mixtures per state)

$$\Rightarrow \text{cluster the observation vectors within each state } j \text{ into a set of } M \text{ clusters}$$

$$\bar{w}_{jm} = \text{number of vectors classified in cluster } m \text{ of state } j$$

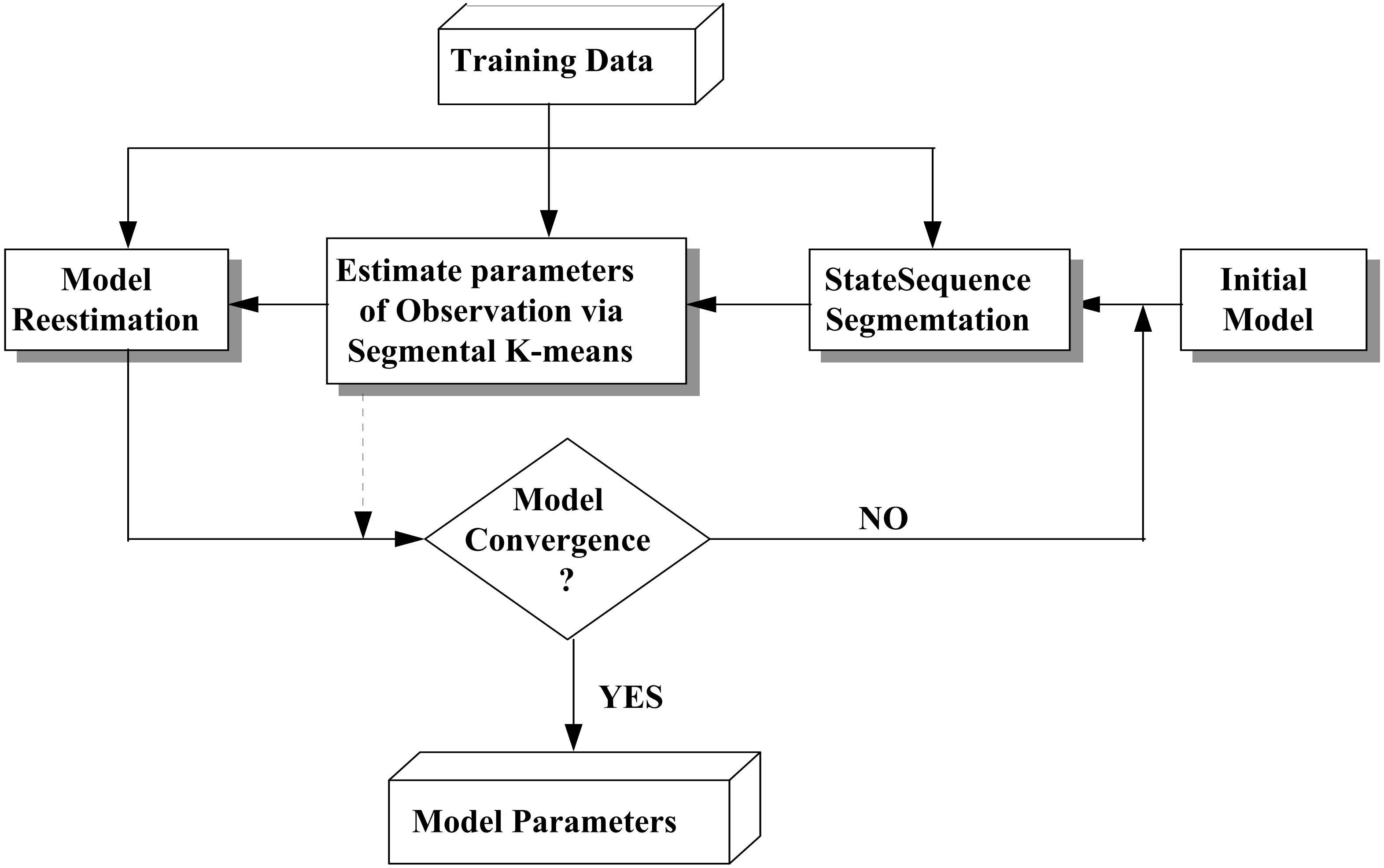
$$\text{divided by the number of vectors in state } j$$

$$\bar{\mu}_{jm} = \text{sample mean of the vectors classified in cluster } m \text{ of state } j$$

$$\bar{\Sigma}_{jm} = \text{sample covariance matrix of the vectors classified in cluster } m \text{ of state } j$$

  - Step 3: Evaluate the model score
    If the difference between the previous and current model scores is greater than a threshold, go back to Step 1, otherwise stop, the initial model is generated

Initialization of HMM (cont.)



Training Data

Model Reestimation

Estimate parameters of Observation via Segmental K-means

StateSequence Segmemtation

Initial Model

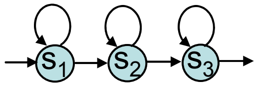
Model Convergence ?

NO

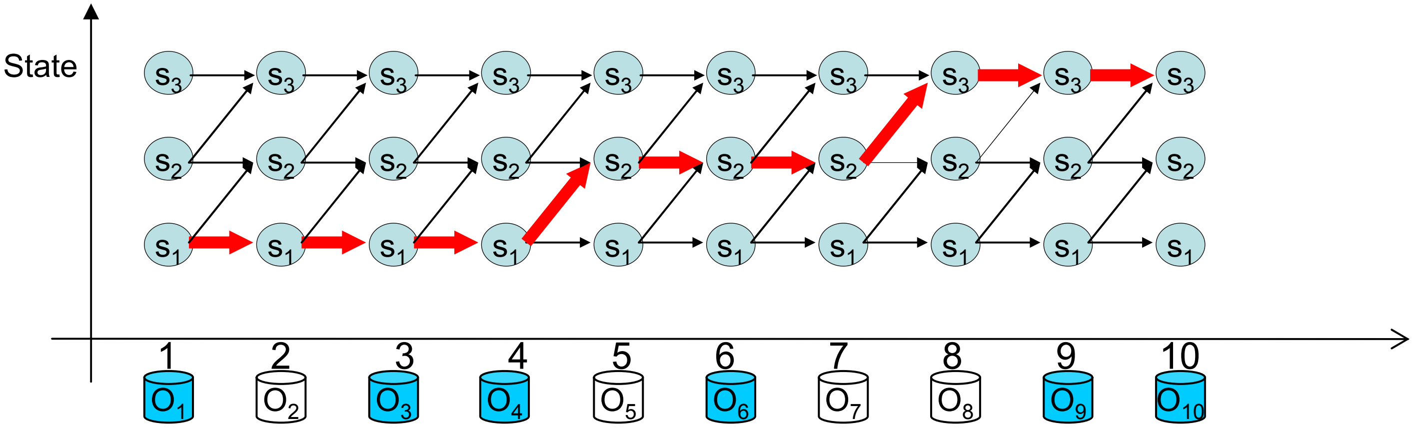
YES

Model Parameters

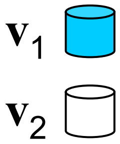
Initialization of HMM (cont.)



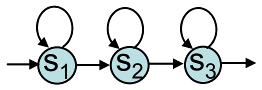
- An example for discrete HMM
  - 3 states and 2 codeword



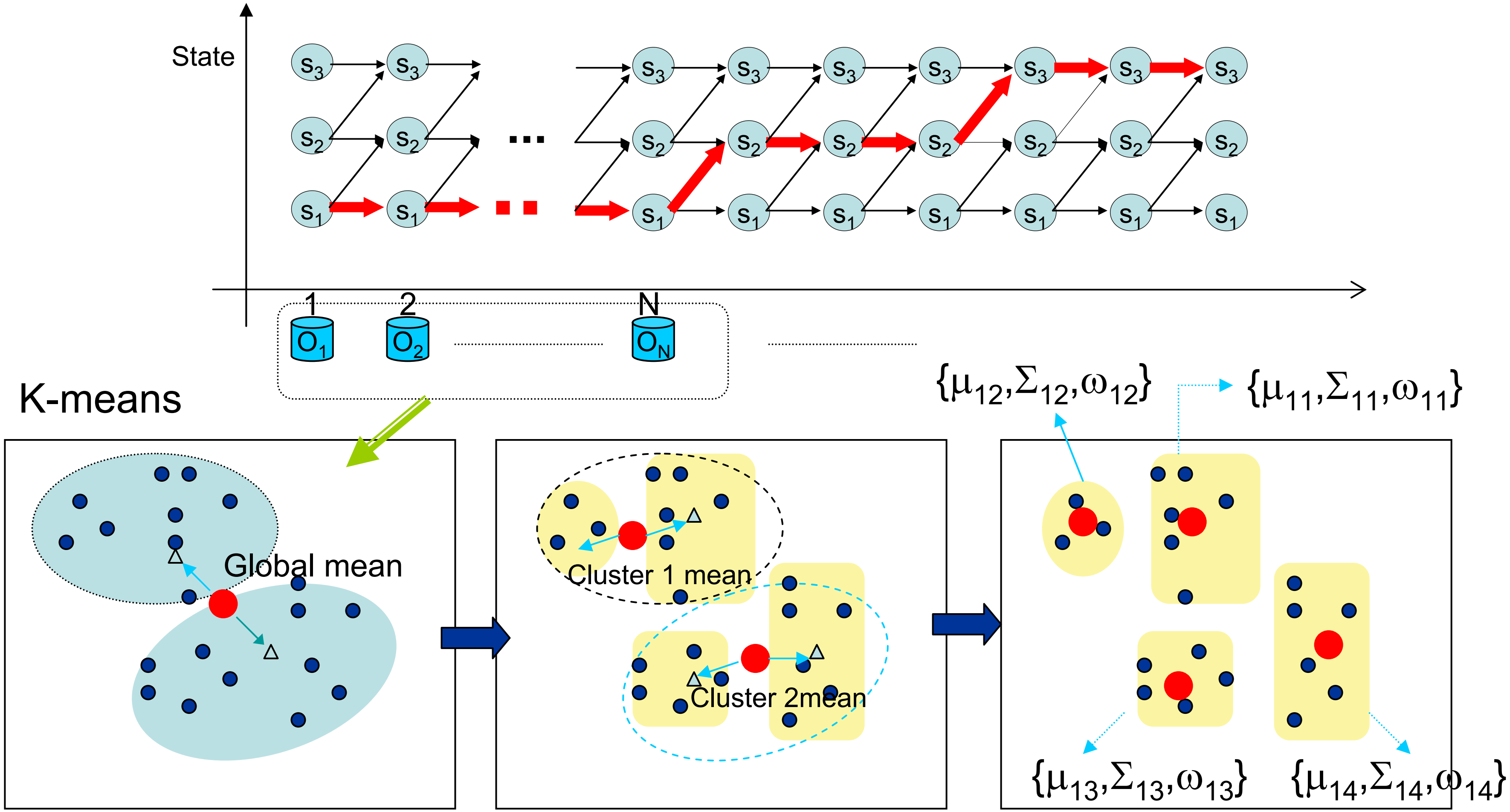
- $b_1(\mathbf{v}_1)=3/4$, $b_1(\mathbf{v}_2)=1/4$
- $b_2(\mathbf{v}_1)=1/3$, $b_2(\mathbf{v}_2)=2/3$
- $b_3(\mathbf{v}_1)=2/3$, $b_3(\mathbf{v}_2)=1/3$



Initialization of HMM (cont.)



- An example for Continuous HMM
  - 3 states and 4 Gaussian mixtures per state



HMM Topology

- Speech is time-evolving non-stationary signal
  - Each HMM state has the ability to capture some quai-stationary segment in the non-stationary speech signal
  - A *left-to-right* topology is a natural candidate to model the speech signal

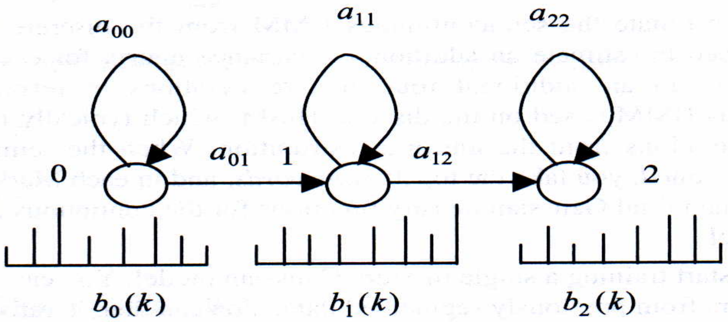


Figure 8.8 A typical hidden Markov model used to model phonemes. There are three states (0-2) and each state has an associated output probability distribution.

  - It is general to represent a phone using 3~5 states (English) and a syllable using 6~8 states (Mandarin Chinese)

Known Limitations of HMMs

- The assumptions of conventional HMMs in Speech Processing
  - The state duration follows an exponential distribution
    - Don’t provide adequate representation of the temporal structure of speech

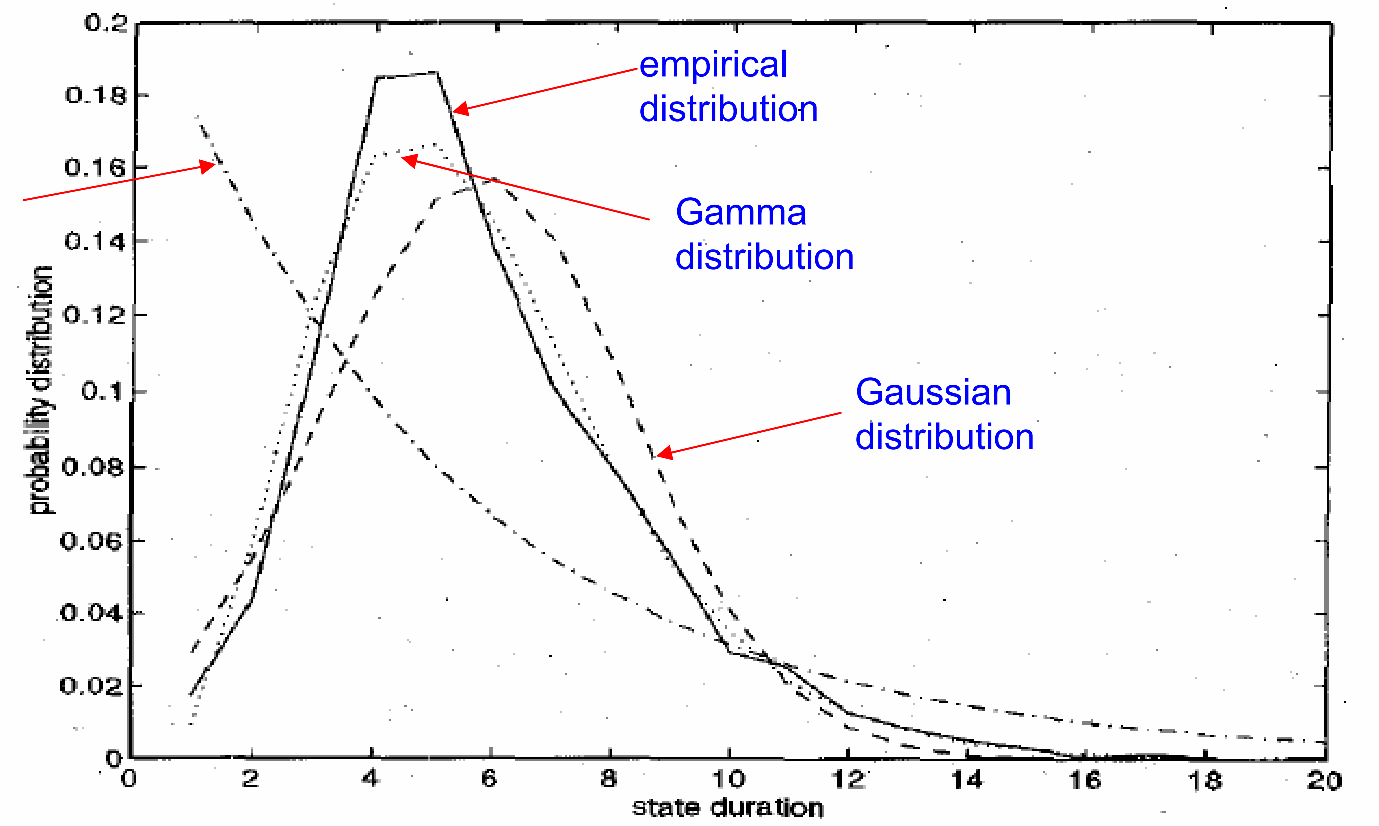
$$d_i(t) = a_{ii}^{t-1}(1 - a_{ii})$$

  - **First-order (Markov) assumption:** the state transition depends only on the origin and destination
  - **Output-independent assumption:** all observation frames are dependent on the state that generated them, not on neighboring observation frames

Researchers have proposed a number of techniques to address these limitations, albeit these solution have not significantly improved speech recognition accuracy for practical applications.

Known Limitations of HMMs (cont.)

- Duration modeling



geometric/
exponential
distribution

empirical
distribution

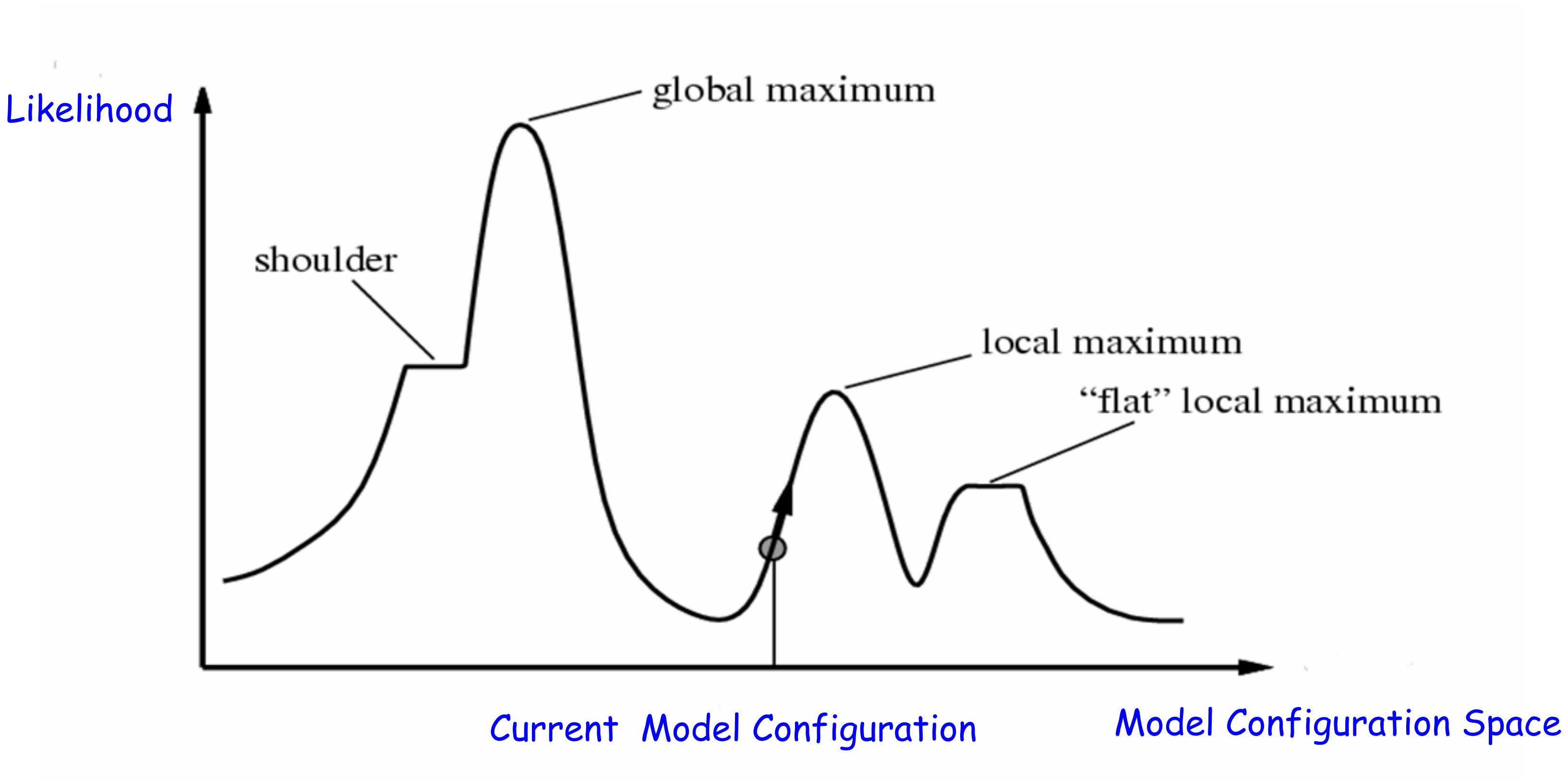
Gamma
distribution

Gaussian
distribution

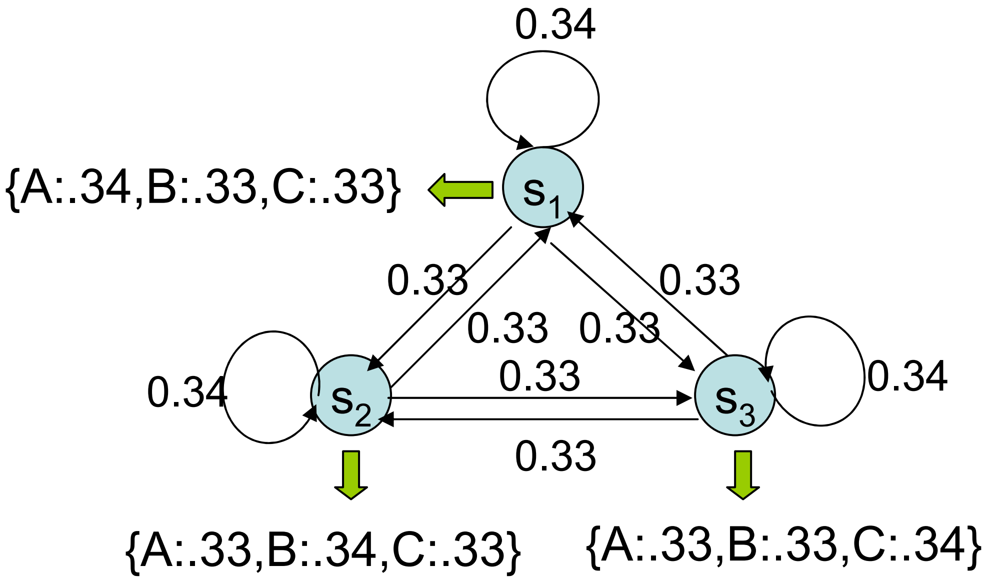
Duration distributions for the seventh state of the word "seven:" empirical distribution (solid line); Gauss fit (dashed line); gamma fit (dotted line); and (d) geometric fit (dash-dot line).

HMM Limitations (cont.)

- The HMM parameters trained by the *Baum-Welch* algorithm and *EM* algorithm were only locally optimized



Homework-2A



TrainSet 1:

1. ABBCABCAABC
2. ABCABC
3. ABCA ABC
4. BBABCAB
5. BCAABCCAB
6. CACCABCA
7. CABCABCA
8. CABCA
9. CABCA

TrainSet 2:

1. BBBCCBC
2. CCBABB
3. AACCBBB
4. BBABBAC
5. CCA ABBAB
6. BBBCCBAA
7. ABBBBABA
8. CCCCC
9. BBAAA

Homework-2A (cont.)

P1. Please specify the model parameters after the first and 50th iterations of Baum-Welch training

P2. Please show the recognition results by using the above training sequences as the testing data (The so-called inside testing).

*You have to perform the recognition task with the HMMs trained from the first and 50th iterations of Baum-Welch training, respectively

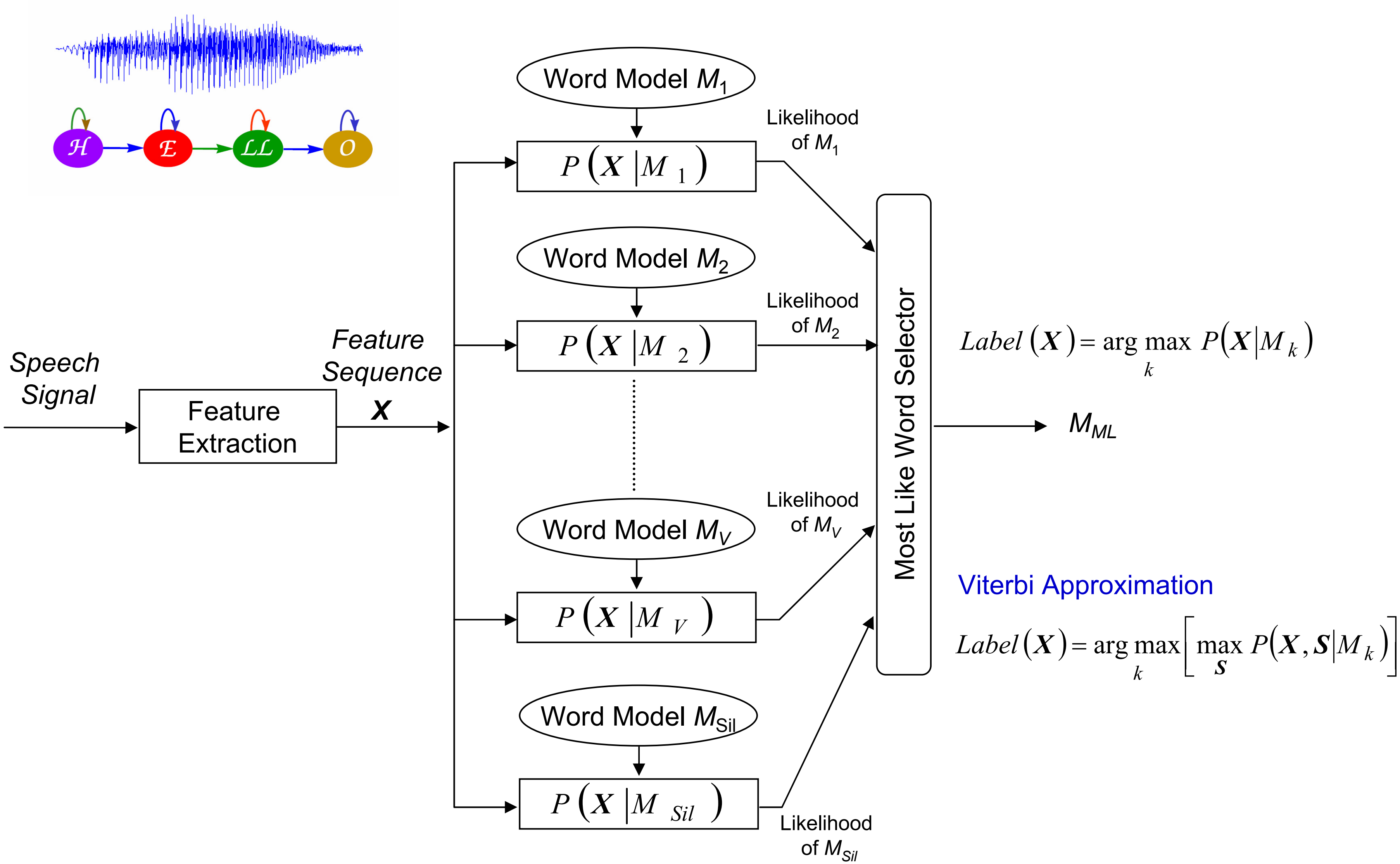
P3. Which class do the following testing sequences belong to?

ABCABCCAB

AABABCCCCBBB

P4. What are the results if Observable Markov Models were instead used in P1, P2 and P3?

Isolated Word Recognition



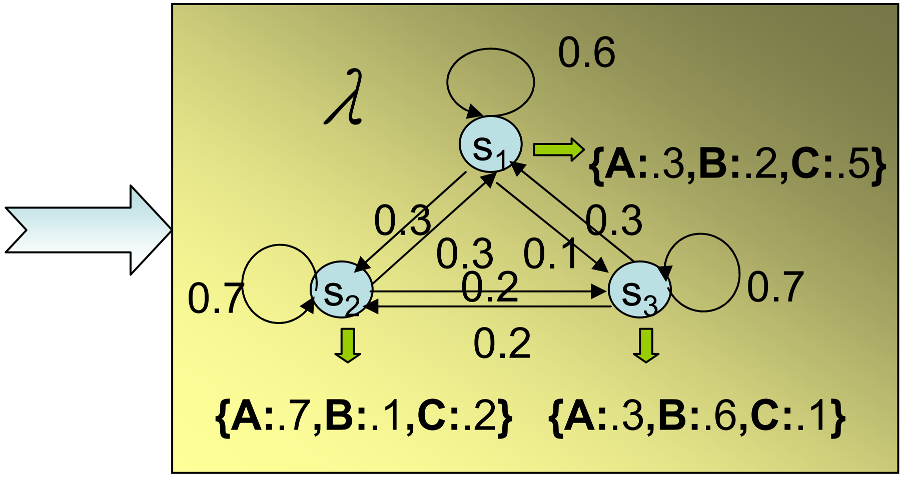
$$Label(\boldsymbol{X}) = \arg\max_k P(\boldsymbol{X}|M_k)$$

Viterbi Approximation

$$Label(\boldsymbol{X}) = \arg\max_k \left[\max_{\boldsymbol{S}} P(\boldsymbol{X},\boldsymbol{S}|M_k)\right]$$

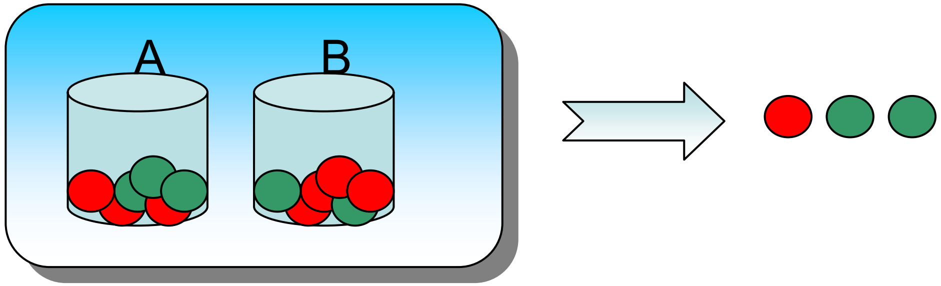
The EM Algorithm

$o_1, o_2, \ldots\ldots, o_T$



$p(\mathbf{O}|\lambda)$

$p(\mathbf{O}|\overline{\lambda}) > p(\mathbf{O}|\lambda)$



Observed data : ***O*** : “ball sequence”
Latent data : ***S*** : “bottle sequence”

Parameters to be estimated to maximize log$P(\mathbf{O}|\lambda)$
$\lambda=\{P(A),P(B),P(B|A),P(A|B),P(R|A),P(G|A),P(R|B),P(G|B)\}$

The EM Algorithm (cont.)

- Introduction of EM (Expectation Maximization):
  - Why EM?
    - Simple optimization algorithms for likelihood function relies on the intermediate variables, called latent (隱藏的)data
      In our case here, ***the state sequence** is the latent data*
    - Direct access to the data necessary to estimate the parameters is impossible or difficult
      In our case here, it is almost impossible to estimate {***A***,***B***, π} without consideration of the ***state sequence***
  - Two Major Steps :
    - ***E*** : expectation with respect to the latent data using the current estimate of the parameters and conditioned on the observations $E[\bullet]_{\mathbf{S}|\lambda,\mathbf{O}}$
    - ***M***: provides a new estimation of the parameters according to Maximum likelihood (ML) or Maximum A Posterior (MAP) Criteria

The EM Algorithm (cont.)

ML and MAP

- Estimation principle based on observations:

$$\boldsymbol{x} = \left(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\right) \quad \Longleftrightarrow \quad \boldsymbol{X} = \left\{X_1, X_2, \ldots, X_n\right\}$$

  - **The Maximum Likelihood (ML) Principle**
    find the model parameter $\boldsymbol{\Phi}$ so that the likelihood $p\left(\boldsymbol{x} \middle| \boldsymbol{\Phi}\right)$ is maximum
    for example, if $\boldsymbol{\Phi} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ *is the parameters of a multivariate normal distribution, and* ***X*** *is i.i.d.* (independent, identically distributed), *then the ML estimate of* $\boldsymbol{\Phi} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ *is*

$$\boldsymbol{\mu}_{ML} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i \quad , \quad \boldsymbol{\Sigma}_{ML} = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_{ML}\right)\left(\boldsymbol{x}_i - \boldsymbol{\mu}_{ML}\right)^t$$

  - **The Maximum A Posteriori (MAP) Principle**
    find the model parameter $\boldsymbol{\Phi}$ so that the likelihood $p\left(\boldsymbol{\Phi} \middle| \boldsymbol{x}\right)$ is maximum

The EM Algorithm (cont.)

- The EM Algorithm is important to HMMs and other learning techniques
  - Discover new model parameters to maximize the log-likelihood of incomplete data $\log P(\boldsymbol{O}|\lambda)$ by iteratively maximizing the expectation of log-likelihood from complete data $\log P(\boldsymbol{O},\boldsymbol{S}|\lambda)$

- Using scalar random variables to introduce the EM algorithm
  - The observable training data $\boldsymbol{O}$
    - We want to maximize $P(\boldsymbol{O}|\lambda)$, $\boldsymbol{\lambda}$ is a parameter vector
  - The hidden (unobservable) data $\boldsymbol{S}$
    - E.g. the component densities of observable data $\boldsymbol{O}$, or the underlying state sequence in HMMs

The EM Algorithm (cont.)

- Assume we have λ and estimate the probability that each $\boldsymbol{S}$ occurred in the generation of $\boldsymbol{O}$
- Pretend we had in fact observed a complete data pair $(\boldsymbol{O},\boldsymbol{S})$ with frequency proportional to the probability $P(\boldsymbol{O},\boldsymbol{S}|\lambda)$, to computed a new $\bar{\lambda}$, the maximum likelihood estimate of λ
- Does the process converge?
- **Algorithm** unknown model setting

$$P(\boldsymbol{O},\boldsymbol{S}|\bar{\lambda}) = P(\boldsymbol{S}|\boldsymbol{O},\bar{\lambda})P(\boldsymbol{O}|\bar{\lambda})$$

Bayes' rule

complete data likelihood

incomplete data likelihood

  - **Log-likelihood expression** and expectation taken over ***S***

$$\log P(\boldsymbol{O}|\bar{\lambda}) = \log P(\boldsymbol{O},\boldsymbol{S}|\bar{\lambda}) - \log P(\boldsymbol{S}|\boldsymbol{O},\bar{\lambda})$$

take expectation over ***S***

$$\log P(\boldsymbol{O}|\bar{\lambda}) = \sum_{s}\left[P(\boldsymbol{S}|\boldsymbol{O},\lambda)\log P(\boldsymbol{O}|\bar{\lambda})\right]$$

$$= \sum_{\boldsymbol{S}}\left[P(\boldsymbol{S}|\boldsymbol{O},\lambda)\log P(\boldsymbol{O},\boldsymbol{S}|\bar{\lambda})\right] - \sum_{\boldsymbol{S}}\left[P(\boldsymbol{S}|\boldsymbol{O},\lambda)\log P(\boldsymbol{S}|\boldsymbol{O},\bar{\lambda})\right]$$

The EM Algorithm (cont.)

- Algorithm (Cont.)
  - We can thus express $\log P(\mathbf{O}|\bar{\lambda})$ as follows

$$
\begin{aligned}
&\log P(\mathbf{O}|\bar{\lambda}) \\
&= \sum_{\mathbf{S}} \left[P(\mathbf{S}|\mathbf{O},\lambda) \log P(\mathbf{O},\mathbf{S}|\bar{\lambda}) \right] - \sum_{\mathbf{S}} \left[P(\mathbf{S}|\mathbf{O},\lambda) \log P(\mathbf{S}|\mathbf{O},\bar{\lambda}) \right] \\
&= Q(\lambda,\bar{\lambda}) - H(\lambda,\bar{\lambda})
\end{aligned}
$$

where

$$Q(\lambda,\bar{\lambda}) = \sum_{\mathbf{S}} \left[P(\mathbf{S}|\mathbf{O},\lambda) \log P(\mathbf{O},\mathbf{S}|\bar{\lambda}) \right]$$

$$H(\lambda,\bar{\lambda}) = \sum_{\mathbf{S}} \left[P(\mathbf{S}|\mathbf{O},\lambda) \log P(\mathbf{S}|\mathbf{O},\bar{\lambda}) \right]$$

  - We want $\log P(\mathbf{O}|\bar{\lambda}) \geq \log P(\mathbf{O}|\lambda)$

$$
\begin{aligned}
&\log P(\mathbf{O}|\bar{\lambda}) - \log P(\mathbf{O}|\lambda) \\
&= \left[Q(\lambda,\bar{\lambda}) - H(\lambda,\bar{\lambda}) \right] - \left[Q(\lambda,\lambda) - H(\lambda,\lambda) \right] \\
&= Q(\lambda,\bar{\lambda}) - Q(\lambda,\lambda) \cancel{- H(\lambda,\bar{\lambda}) + H(\lambda,\lambda)}
\end{aligned}
$$

The EM Algorithm (cont.)

- $-H(\lambda,\bar{\lambda})+H(\lambda,\lambda)$ has the following property

$$
\begin{aligned}
& -H(\lambda,\bar{\lambda})+H(\lambda,\lambda) \\
& = -\sum_{\mathbf{S}}\left[P(\mathbf{S}|\mathbf{O},\lambda)\log\frac{P(\mathbf{S}|\mathbf{O},\bar{\lambda})}{P(\mathbf{S}|\mathbf{O},\lambda)}\right] \\
& \geq \sum_{\mathbf{S}}\left[P(\mathbf{S}|\mathbf{O},\lambda)\left(1-\frac{P(\mathbf{S}|\mathbf{O},\bar{\lambda})}{P(\mathbf{S}|\mathbf{O},\lambda)}\right)\right] \qquad (\because \log x \leq x-1) \\
& = \sum_{\mathbf{S}}\left[P(\mathbf{S}|\mathbf{O},\lambda)-P(\mathbf{S}|\mathbf{O},\bar{\lambda})\right] \\
& = \ 0 \\
& \therefore\ -H(\lambda,\bar{\lambda})+H(\lambda,\lambda)\geq 0
\end{aligned}
$$

Kullbuack-Leibler (KL) distance

Jensen's inequality

– Therefore, for maximizing $\log P(\mathbf{O}|\bar{\lambda})$, we only need to maximize the *Q*-function (auxiliary function)

$$
Q(\lambda,\bar{\lambda})=\sum_{\mathbf{S}}\left[P(\mathbf{S}|\mathbf{O},\lambda)\log P(\mathbf{O},\mathbf{S}|\bar{\lambda})\right]
$$

Expectation of the complete data log likelihood with respect to the latent state sequences

EM Applied to Discrete HMM Training

- Apply EM algorithm to iteratively refine the HMM parameter vector $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$
  - By maximizing the auxiliary function

$$Q(\lambda, \bar{\lambda}) = \sum_{\boldsymbol{S}} \left[P(\boldsymbol{S} | \boldsymbol{O}, \lambda) log\ P(\boldsymbol{O}, \boldsymbol{S} | \bar{\lambda}) \right]$$

$$= \sum_{\boldsymbol{S}} \left[\frac{P(\boldsymbol{O}, \boldsymbol{S} | \lambda)}{P(\boldsymbol{O} | \lambda)} log\ P(\boldsymbol{O}, \boldsymbol{S} | \bar{\lambda}) \right]$$

  - Where $P(\boldsymbol{O}, \boldsymbol{S} | \lambda)$ and $P(\boldsymbol{O}, \boldsymbol{S} | \bar{\lambda})$ can be expressed as

$$P(\boldsymbol{O}, \boldsymbol{S} | \lambda) = \pi_{s_1} \left[\prod_{t=1}^{T-1} a_{s_t s_{t+1}} \right] \left[\prod_{t=1}^{T} b_{s_t}(\boldsymbol{o}_t) \right]$$

$$\log\ P(\boldsymbol{O}, \boldsymbol{S} | \lambda) = \log\ \pi_{s_1} + \sum_{t=1}^{T-1} \log\ a_{s_t s_{t+1}} + \sum_{t=1}^{T} \log\ b_{s_t}(\boldsymbol{o}_t)$$

$$\log\ P(\boldsymbol{O}, \boldsymbol{S} | \bar{\lambda}) = \log\ \bar{\pi}_{s_1} + \sum_{t=1}^{T-1} \log\ \bar{a}_{s_t s_{t+1}} + \sum_{t=1}^{T} \log\ \bar{b}_{s_t}(\boldsymbol{o}_t)$$

EM Applied to Discrete HMM Training (cont.)

- Rewrite the auxiliary function as

$$Q(\lambda,\bar{\lambda}) = Q_{\pi}(\lambda,\bar{\pi}) + Q_{a}(\lambda,\bar{a}) + Q_{b}(\lambda,\bar{b})$$

$$Q_{\pi}(\lambda,\bar{\pi}) = \sum_{\text{all } S}\left[\frac{P(O,S|\lambda)}{P(O|\lambda)}\log \bar{\pi}_{s_1}\right] \overset{?}{=} \sum_{i=1}^{N}\left[\overset{w_i}{\boxed{\frac{P(O,s_1=i|\lambda)}{P(O|\lambda)}}}\log \overset{y_i}{\boxed{\bar{\pi}_i}}\right]$$

$$Q_{a}(\lambda,\bar{a}) = \sum_{\text{all } S}\left[\frac{P(O,S|\lambda)}{P(O|\lambda)}\sum_{t=1}^{T-1}\log \bar{a}_{s_t s_{t+1}}\right] \overset{?}{=} \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{t=1}^{T-1}\left[\frac{P(O,s_t=i,s_{t+1}=j|\lambda)}{P(O|\lambda)}\log \bar{a}_{ij}\right]$$

$$Q_{b}(\lambda,\bar{b}) = \sum_{\text{all } S}\left[\frac{P(O,S|\lambda)}{P(O|\lambda)}\sum_{t=1}^{T}\log \bar{b}_{s_t}(k)\right] \overset{?}{=} \sum_{j=1}^{N}\sum_{k}\sum_{t\in o_t=v_k}\left[\frac{P(O,s_t=j|\lambda)}{P(O|\lambda)}\log \bar{b}_j(k)\right]$$

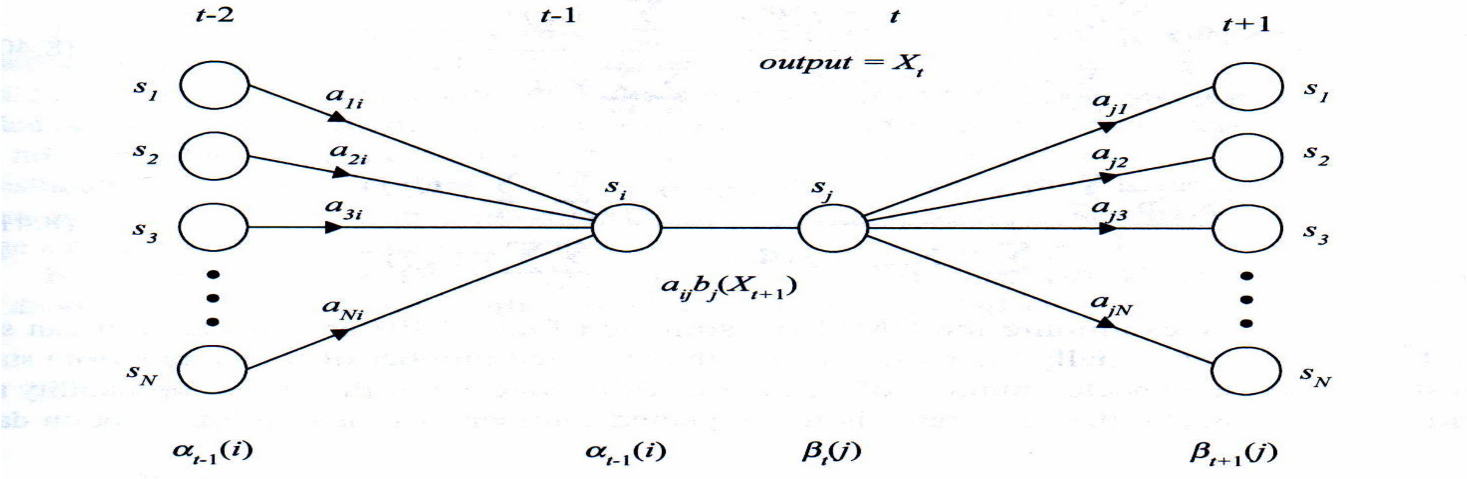


Figure 8.7 Illustration of the operations required for the computation of $\gamma_t(i,j)$, which is the probability of taking the transition from state i to state j at time t.

EM Applied to Discrete HMM Training (cont.)

- The auxiliary function contains three independent terms, π_i , a_{ij} and $b_j(k)$
  - Can be maximized individually
  - All of the same form

$$F(\boldsymbol{y}) = g(y_1, y_2, ..., y_N) = \sum_{j=1}^{N} w_j \, log \, y_j, \ \text{where} \ \sum_{j=1}^{N} y_j = 1, \ \text{and} \ y_j \geq 0$$

$$F(\boldsymbol{y}) \ \text{has maximum value when}: \ y_j = \frac{w_j}{\sum_{j=1}^{N} w_j}$$

EM Applied to Discrete HMM Training (cont.)

- **Proof**: Apply Lagrange Multiplier

By applying Lagrange Multiplier ℓ

Suppose that $F = \sum_{j=1}^{N} w_j \log y_j = \sum_{j=1}^{N} w_j \log y_j + \ell\left(\sum_{j=1}^{N} y_j - 1\right)$

Constraint

$$\frac{\partial F}{\partial y_j} = \frac{w_j}{y_j} + \ell = 0 \Rightarrow \ell = -\frac{w_j}{y_j} \ \forall\ j$$

$$\ell \sum_{j=1}^{N} y_j = -\sum_{j=1}^{N} w_j \Rightarrow \ell = -\sum_{j=1}^{N} w_j$$

$$\therefore y_j = \frac{w_j}{\sum_{j=1}^{N} w_j}$$

EM Applied to Discrete HMM Training (cont.)

- The new model parameter set $\bar{\lambda} = \left(\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{A}}, \bar{\boldsymbol{B}}\right)$ can be expressed as:

$$\overline{\pi_i} = \frac{P\left(\boldsymbol{O}, s_1 = i \middle| \lambda\right)}{P\left(\boldsymbol{O} \middle| \lambda\right)} = \gamma_1(i)$$

$$\overline{a_{ij}} = \frac{\sum_{t=1}^{T-1} P\left(\boldsymbol{O}, s_t = i, s_{t+1} = j \middle| \lambda\right)}{\sum_{t=1}^{T-1} P\left(\boldsymbol{O}, s_t = i \middle| \lambda\right)} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\overline{b_i}(k) = \frac{\sum_{\substack{t=1 \\ \text{s.t. } o_t = v_k}}^{T} P\left(\boldsymbol{O}, s_t = i \middle| \lambda\right)}{\sum_{t=1}^{T} P\left(\boldsymbol{O}, s_t = i \middle| \lambda\right)} = \frac{\sum_{\substack{t=1 \\ \text{s.t. } o_t = v_k}}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

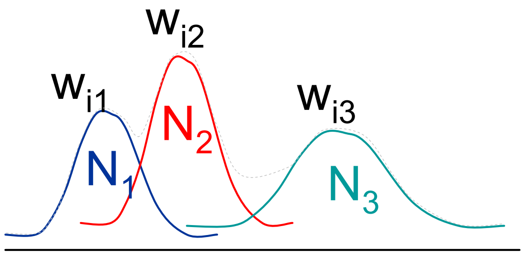
EM Applied to Continuous HMM Training (cont.)

- Continuous HMM: the state observation does not come from a finite set, but from a continuous space
  - The difference between the discrete and continuous HMM lies in a different form of state output probability
  - Discrete HMM requires the quantization procedure to map observation vectors from the continuous space to the discrete space
- Continuous Mixture HMM
  - The state observation distribution of HMM is modeled by multivariate Gaussian mixture density functions (*M* mixtures)

$$b_j(\boldsymbol{o})=\sum_{k=1}^{M} c_{jk} b_{jk}(\boldsymbol{o})$$

$$=\sum_{k=1}^{M} c_{jk} N(\boldsymbol{o};\boldsymbol{\mu}_{jk},\boldsymbol{\Sigma}_{jk})=\sum_{k=1}^{M} c_{jk}\left(\frac{1}{\left(\sqrt{2\pi}\right)^{L}\left|\boldsymbol{\Sigma}_{jk}\right|^{1/2}}\exp\left(-\frac{1}{2}(\boldsymbol{o}-\boldsymbol{\mu}_{jk})^{t}\boldsymbol{\Sigma}_{jk}^{-1}(\boldsymbol{o}-\boldsymbol{\mu}_{jk})\right)\right)$$

$$\sum_{k=1}^{M} c_{jk} = 1$$



Distribution for State *i*

EM Applied to Continuous HMM Training (cont.)

- Express $b_j(\boldsymbol{o})$ with respect to each single mixture component $b_{jk}(\boldsymbol{o})$

Note:

$$\prod_{t=1}^{T}\left(\sum_{k_t=1}^{M} a_{tk_t}\right)$$
$$=(a_{11}+a_{12}+...+a_{1M})(a_{21}+a_{22}+...+a_{2M})...(a_{T1}+a_{T2}+...+a_{TM})$$
$$=\sum_{k_1=1}^{M}\sum_{k_2=1}^{M}...\sum_{k_T=1}^{M}\prod_{t=1}^{T}a_{tk_t}$$

$$P(\boldsymbol{O},\boldsymbol{S}|\lambda)=\pi_{s_1}\left\{\prod_{t=1}^{T-1}a_{s_t s_{t+1}}\right\}\left\{\prod_{t=1}^{T}b_{s_t}(\boldsymbol{o}_t)\right\}$$

$$=\pi_{s_1}\left\{\prod_{t=1}^{T-1}a_{s_t s_{t+1}}\right\}\left\{\sum_{k_1=1}^{M}\sum_{k_2=1}^{M}\cdots\sum_{k_T=1}^{M}\boxed{\prod_{t=1}^{T}\left[c_{s_t k_t}b_{s_t k_t}(\boldsymbol{o}_t)\right]}\right\}$$

$$P(\boldsymbol{O},\boldsymbol{S},\boldsymbol{K}|\lambda)=\pi_{s_1}\left\{\prod_{t=1}^{T-1}a_{s_t s_{t+1}}\right\}\left\{\prod_{t=1}^{T}\left[c_{s_t k_t}b_{s_t k_t}(\boldsymbol{o}_t)\right]\right\}$$

$\boldsymbol{K}$: one of the possible mixture component sequence along with the state sequence $\boldsymbol{S}$

$$P(\boldsymbol{O}|\lambda)=\sum_{\boldsymbol{S}}\sum_{\boldsymbol{K}}P(\boldsymbol{O},\boldsymbol{S},\boldsymbol{K}|\lambda)$$

EM Applied to Continuous HMM Training (cont.)

- Therefore, an auxiliary function for the EM algorithm can be written as:

$$Q\left(\lambda,\bar{\lambda}\right)=\sum_{\boldsymbol{S}}\sum_{\boldsymbol{K}}\left[P\left(\boldsymbol{S},\boldsymbol{K}\middle|\boldsymbol{O},\lambda\right)\log P\left(\boldsymbol{O},\boldsymbol{S},\boldsymbol{K}\middle|\bar{\lambda}\right)\right]$$

$$=\sum_{\boldsymbol{S}}\sum_{\boldsymbol{K}}\left[\frac{P\left(\boldsymbol{O},\boldsymbol{S},\boldsymbol{K}\middle|\lambda\right)}{P\left(\boldsymbol{O}\middle|\lambda\right)}\log P\left(\boldsymbol{O},\boldsymbol{S},\boldsymbol{K}\middle|\bar{\lambda}\right)\right]$$

$$\log P\left(\boldsymbol{O},\boldsymbol{S},\boldsymbol{K}\middle|\bar{\lambda}\right)=\log\bar{\pi}_{s_1}+\sum_{t=1}^{T-1}\log\bar{a}_{s_t s_{t+1}}+\sum_{t=1}^{T}\log\bar{b}_{s_t k_t}\left(\boldsymbol{o}_t\right)+\sum_{t=1}^{T}\log\bar{c}_{s_t k_t}$$

$$\Downarrow$$

$$Q\left(\lambda,\bar{\lambda}\right)=Q_{\boldsymbol{\pi}}\left(\lambda,\bar{\pi}\right)+Q_{\boldsymbol{a}}\left(\lambda,\bar{\boldsymbol{a}}\right)+Q_{\boldsymbol{b}}\left(\lambda,\bar{\boldsymbol{b}}\right)+Q_{c}\left(\lambda,\bar{c}\right)$$

initial probabilities

state transition probabilities

Gaussian density functions

mixture components

EM Applied to Continuous HMM Training (cont.)

- The only difference we have when compared with Discrete HMM training

$$Q_{\boldsymbol{b}}\left(\boldsymbol{\lambda},\overline{\boldsymbol{b}}\right)=\sum_{t=1}^{T}\left\{\left[\sum_{j=1}^{N}\sum_{k=1}^{M}\overset{\gamma_t(j,k)}{\boxed{P\left(s_t=j,k_t=k\middle|\boldsymbol{O},\boldsymbol{\lambda}\right)}}\right]\log\overline{b}_{jk}\left(\boldsymbol{o}_t\right)\right\}$$

$$Q_{c}\left(\boldsymbol{\lambda},\overline{\boldsymbol{c}}\right)=\sum_{t=1}^{T}\left\{\left[\sum_{j=1}^{N}\sum_{k=1}^{M}P\left(s_t=j,k_t=k\middle|\boldsymbol{O},\boldsymbol{\lambda}\right)\right]\log\overline{c}_{jk}\left(\boldsymbol{o}_t\right)\right\}$$

EM Applied to Continuous HMM Training (cont.)

Let $\gamma_t(j,k) = \sum_{k=1}^{M} P\left(s_t = j, k_t = k \middle| \boldsymbol{O}, \boldsymbol{\lambda}\right)$

$$\overline{b}_{jk}(\boldsymbol{o}_t) = N\left(\boldsymbol{o}_t; \overline{\boldsymbol{\mu}}_{jk}, \overline{\boldsymbol{\Sigma}}_{jk}\right) = \frac{1}{(2\pi)^{L/2}\left|\overline{\boldsymbol{\Sigma}}_{jk}\right|^{1/2}} \exp\left(-\frac{1}{2}\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)^t \overline{\boldsymbol{\Sigma}}_{jk}^{-1}\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)\right)$$

$$\log \overline{b}_{jk}(\boldsymbol{o}_t) = -L/2 \cdot \log(2\pi) + 1/2 \cdot \log\left|\overline{\boldsymbol{\Sigma}}_{jk}^{-1}\right| - \left(\frac{1}{2}\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)^t \overline{\boldsymbol{\Sigma}}_{jk}^{-1}\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)\right)$$

$$\frac{\partial \log \overline{b}_{jk}(\boldsymbol{o}_t)}{\partial \overline{\boldsymbol{\mu}}_{jk}} = \overline{\boldsymbol{\Sigma}}_{jk}^{-1}\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)$$

$$\frac{d\left(\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{C}\boldsymbol{x}\right)}{d\boldsymbol{x}} = \left(\boldsymbol{C} + \boldsymbol{C}^{\boldsymbol{T}}\right)\boldsymbol{x}$$

and Σ_{jk}^{-1} is symmetric here

$$\frac{\partial Q_{\boldsymbol{b}}\left(\boldsymbol{\lambda}, \overline{\boldsymbol{b}}\right)}{\partial \overline{\boldsymbol{\mu}}_{jk}} = \frac{\partial \sum_{t=1}^{T}\left\{\left[\sum_{j=1}^{N}\sum_{k=1}^{M}\gamma_t(j,k)\log \overline{b}_{jk}(\boldsymbol{o}_t)\right]\right\}}{\partial \overline{\boldsymbol{\mu}}_{jk}}$$

$$\Rightarrow \sum_{t=1}^{T}\left\{\gamma_t(j,k)\overline{\boldsymbol{\Sigma}}_{jk}^{-1}\left(\boldsymbol{o}_t - \overline{\boldsymbol{\mu}}_{jk}\right)\right\} = 0$$

$$\Rightarrow \overline{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T}\left[\gamma_t(j,k) \cdot \boldsymbol{o}_t\right]}{\sum_{t=1}^{T}\gamma_t(j,k)}$$

EM Applied to Continuous HMM Training (cont.)

$$\log \bar{b}_{jk}(\boldsymbol{o}_t) = -L/2 \cdot \log(2\pi) - 1/2 \cdot \log\left|\bar{\boldsymbol{\Sigma}}_{jk}\right| - \left(\frac{1}{2}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t \bar{\boldsymbol{\Sigma}}_{jk}^{-1}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})\right)$$

$$\frac{\partial \log \bar{b}_{jk}(\boldsymbol{o}_t)}{\partial(\bar{\boldsymbol{\Sigma}}_{jk})} = -\left[\frac{1}{2}\cdot \cancel{\left|\bar{\boldsymbol{\Sigma}}_{jk}\right|^{-1} \cdot \left|\bar{\boldsymbol{\Sigma}}_{jk}\right|} \cdot \bar{\boldsymbol{\Sigma}}_{jk}^{-1} - \left(\bar{\boldsymbol{\Sigma}}_{jk}^{-1}\frac{1}{2}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t \bar{\boldsymbol{\Sigma}}_{jk}^{-1}\right)\right]$$

$$= -\frac{1}{2}\cdot\left[\bar{\boldsymbol{\Sigma}}_{jk}^{-1} - \bar{\boldsymbol{\Sigma}}_{jk}^{-1}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t \bar{\boldsymbol{\Sigma}}_{jk}^{-1}\right]$$

$$\frac{d\,(\boldsymbol{a}^{\boldsymbol{T}}\boldsymbol{X}^{-1}\boldsymbol{b})}{d\boldsymbol{X}} = -\boldsymbol{X}^T\boldsymbol{a}\boldsymbol{b}^{\boldsymbol{T}}\boldsymbol{X}^T$$

$$\frac{d[\det(\boldsymbol{X})]}{d\boldsymbol{X}} = \det(\boldsymbol{X})\cdot\boldsymbol{X}^{-\boldsymbol{T}}$$

and Σ_{jk} is symmetric here

$$\frac{\partial Q_{\boldsymbol{b}}(\lambda, \bar{\boldsymbol{b}})}{\partial(\bar{\boldsymbol{\Sigma}}_{jk})} = \frac{\partial \sum_{t=1}^{T}\left\{\left[\sum_{j=1}^{N}\sum_{k=1}^{M}\gamma_t(j,k)\log \bar{b}_{jk}(\boldsymbol{o}_t)\right]\right\}}{\partial(\bar{\boldsymbol{\Sigma}}_{jk}^{-1})}$$

$$\Rightarrow \sum_{t=1}^{T}\left\{\gamma_t(j,k)\left(-\frac{1}{2}\right)\cdot\left[\bar{\boldsymbol{\Sigma}}_{jk}^{-1} - \bar{\boldsymbol{\Sigma}}_{jk}^{-1}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t \bar{\boldsymbol{\Sigma}}_{jk}^{-1}\right]\right\} = 0$$

$$\Rightarrow \sum_{t=1}^{T}\gamma_t(j,k)\bar{\boldsymbol{\Sigma}}_{jk}^{-1} = \sum_{t=1}^{T}\gamma_t(j,k)\bar{\boldsymbol{\Sigma}}_{jk}^{-1}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t \bar{\boldsymbol{\Sigma}}_{jk}^{-1}$$

$$\Rightarrow \sum_{t=1}^{T}\gamma_t(j,k)\bar{\boldsymbol{\Sigma}}_{jk}\bar{\boldsymbol{\Sigma}}_{jk}^{-1}\bar{\boldsymbol{\Sigma}}_{jk} = \sum_{t=1}^{T}\gamma_t(j,k)\bar{\boldsymbol{\Sigma}}_{jk}\bar{\boldsymbol{\Sigma}}_{jk}^{-1}(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t \bar{\boldsymbol{\Sigma}}_{jk}^{-1}\bar{\boldsymbol{\Sigma}}_{jk}$$

$$\Rightarrow \bar{\boldsymbol{\Sigma}}_{jk} = \frac{\sum_{t=1}^{T}\left[\gamma_t(j,k)\cdot(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_{jk})^t\right]}{\sum_{t=1}^{T}\gamma_t(j,k)}$$

EM Applied to Continuous HMM Training (cont.)

- The new model parameter set for each mixture component and mixture weight can be expressed as:

$$\overline{\boldsymbol{\mu}}_{jk}=\frac{\sum_{t=1}^{T}\left[\frac{P(\boldsymbol{O},s_t=j,k_t=k|\boldsymbol{\lambda})}{P(\boldsymbol{O}|\boldsymbol{\lambda})}\boldsymbol{o}_t\right]}{\sum_{t=1}^{T}\frac{P(\boldsymbol{O},s_t=j,k_t=k|\boldsymbol{\lambda})}{P(\boldsymbol{O}|\boldsymbol{\lambda})}}=\frac{\sum_{t=1}^{T}\left[\gamma_t(j,k)\boldsymbol{o}_t\right]}{\sum_{t=1}^{T}\gamma_t(j,k)}$$

$$\overline{\boldsymbol{\Sigma}}_{jk}=\frac{\sum_{t=1}^{T}\left[\frac{P(\boldsymbol{O},s_t=j,k_t=k|\boldsymbol{\lambda})}{P(\boldsymbol{O}|\boldsymbol{\lambda})}(\boldsymbol{o}_t-\overline{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t-\overline{\boldsymbol{\mu}}_{jk})^t\right]}{\sum_{t=1}^{T}\frac{P(\boldsymbol{O},s_t=j,k_t=k|\boldsymbol{\lambda})}{P(\boldsymbol{O}|\boldsymbol{\lambda})}}=\frac{\sum_{t=1}^{T}\left[\gamma_t(j,k)(\boldsymbol{o}_t-\overline{\boldsymbol{\mu}}_{jk})(\boldsymbol{o}_t-\overline{\boldsymbol{\mu}}_{jk})^t\right]}{\sum_{t=1}^{T}\gamma_t(j,k)}$$

$$\overline{c}_{jk}=\frac{\sum_{t=1}^{T}\gamma_t(j,k)}{\sum_{t=1}^{T}\sum_{k=1}^{M}\gamma_t(j,k)}$$

Measures of ASR Performance

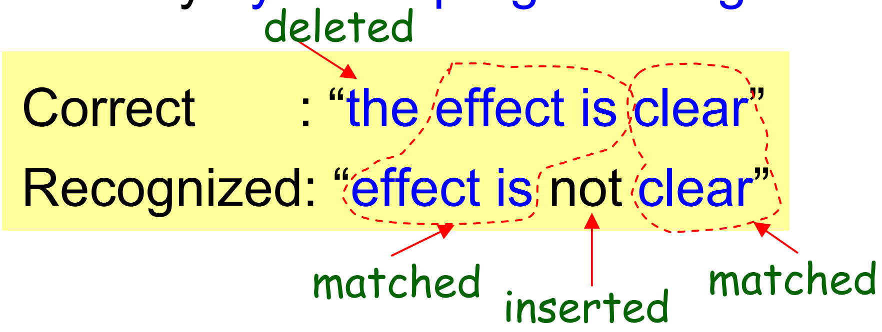
- Evaluating the performance of automatic speech recognition (ASR) systems is critical, and the Word Recognition Error Rate (WER) is one of the most important measures

- There are typically three types of word recognition errors
  - Substitution
    - An incorrect word was substituted for the correct word
  - Deletion
    - A correct word was omitted in the recognized sentence
  - Insertion
    - An extra word was added in the recognized sentence

- How to determine the minimum error rate?

Measures of ASR Performance (cont.)

- Calculate the WER by aligning the correct word string against the recognized word string
  - A maximum substring matching problem
  - Can be handled by dynamic programming
- Example:



Correct : “the effect is clear”

Recognized: “effect is not clear”

  - Error analysis: one deletion and one insertion
  - Measures: word error rate (WER), word correction rate (WCR), word accuracy rate (WAR)

Might be higher than 100%

$$\text{Word Error Rate} = 100\% \frac{\text{Sub.} + \text{Del.} + \text{Ins. words}}{\text{No. of words in the correct sentence}} = \frac{2}{4} = 50\%$$

WER+ WAR =100%

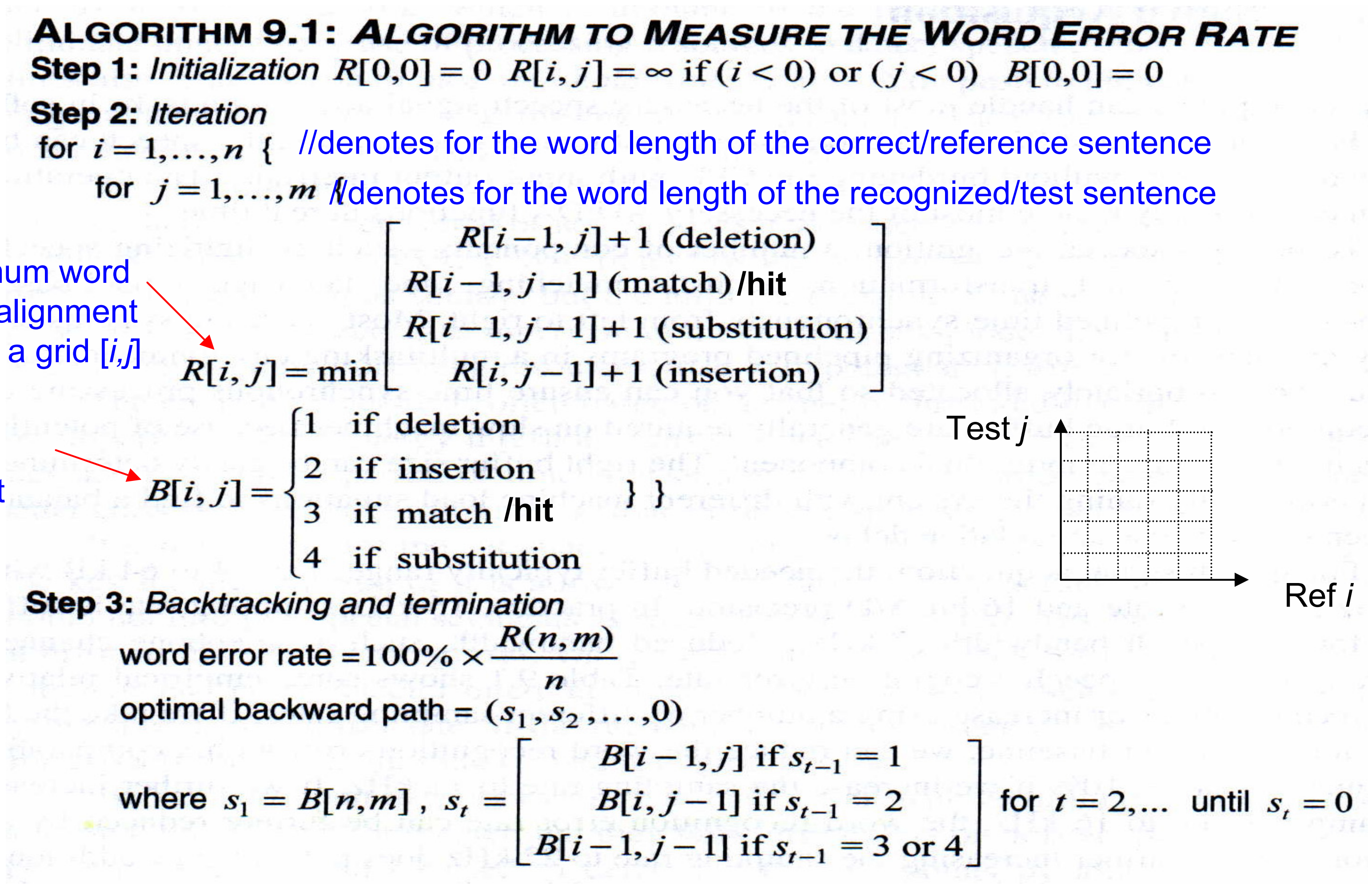
$$\text{Word Correction Rate} = 100\% \frac{\text{Matched words}}{\text{No. of words in the correct sentence}} = \frac{3}{4} = 75\%$$

$$\text{Word Accuracy Rate} = 100\% \frac{\text{Matched - Ins. words}}{\text{No. of words in the correct sentence}} = \frac{3-1}{4} = 50\%$$

Might be negative

Measures of ASR Performance (cont.)

- A Dynamic Programming Algorithm (Textbook)



ALGORITHM 9.1: *ALGORITHM TO MEASURE THE WORD ERROR RATE*

Step 1: *Initialization* $R[0,0]=0$ $R[i,j]=\infty$ if $(i<0)$ or $(j<0)$ $B[0,0]=0$

Step 2: *Iteration*

minimum word error alignment at the a grid [*i*,*j*]

kinds of alignment

```
for $i = 1,\ldots,n$ {   //denotes for the word length of the correct/reference sentence
    for $j = 1,\ldots,m$ {   //denotes for the word length of the recognized/test sentence
        $R[i,j] = \min \begin{bmatrix} R[i-1,j]+1 \text{ (deletion)} \\ R[i-1,j-1] \text{ (match) /hit} \\ R[i-1,j-1]+1 \text{ (substitution)} \\ R[i,j-1]+1 \text{ (insertion)} \end{bmatrix}$
        $B[i,j] = \begin{cases} 1 & \text{if deletion} \\ 2 & \text{if insertion} \\ 3 & \text{if match /hit} \\ 4 & \text{if substitution} \end{cases}$
} }
```

Step 3: *Backtracking and termination*

word error rate $= 100\% \times \dfrac{R(n,m)}{n}$

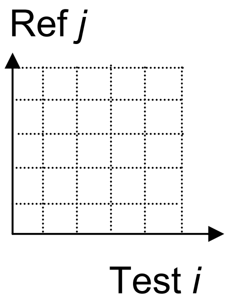
optimal backward path = $(s_1, s_2, \ldots, 0)$

where $s_1 = B[n,m]$, $s_t = \begin{bmatrix} B[i-1,j] \text{ if } s_{t-1} = 1 \\ B[i,j-1] \text{ if } s_{t-1} = 2 \\ B[i-1,j-1] \text{ if } s_{t-1} = 3 \text{ or } 4 \end{bmatrix}$ for $t = 2, \ldots$ until $s_t = 0$

Measures of ASR Performance (cont.)

- Algorithm (by Berlin Chen)

```
Step 1: Initialization :
        G[0][0] = 0;
        for i = 1,..., n  { //test
            G[i][0] = G[i-1][0] + 1;
            B[i][0] = 1; //Insertion
        }   (Horizontal Direction)
        for j = 1,..., m  { //reference
            G[0][j] = G[0][j-1] + 1;
            B[0][j] = 2; // Deletion
        }   (Vertical Direction)
```



```
Step 2: Iteration :
    for i = 1,..., n  { //test
      for j = 1,..., m  { //reference
                         G[i-1][j] + 1  (Insertion)
                         G[i][j-1] + 1   (Delection)
        G[i][j] = min    G[i-1][j-1] + 1 (if LR[j] != LT[i], Substitution)
                         G[i-1][j-1]   (if LR[j] = LT[i], Match)

                   1; //Insertion,  (Horizontal Direction)
        B[i][j] =  2; //Deletion ,  (Vertical Direction)
                   3; //Substitution  (Diagonal Direction)
                   4; //match (Diagonal Direction)

      } //for j, reference
    } //for i, test
```

```
Step 3: Measure and Backtrace :
    Word Error Rate = 100% × $\frac{G[n][m]}{m}$
    Word Accuracy Rate = 100% – Word Error Rate
    Optimal backtrace path = (B[n][m] → ..... → B[0][0])
    if        B[i][j] = 1   print "          LT[i]" ; //Insertion,  then go left
    else if   B[i][j] = 2   print "LR[j]          "; //Deletion ,  then go down
    else                    print "LR[j]    LR[i] "; //Hit/Match or Substitution,  then go down diagonally
```

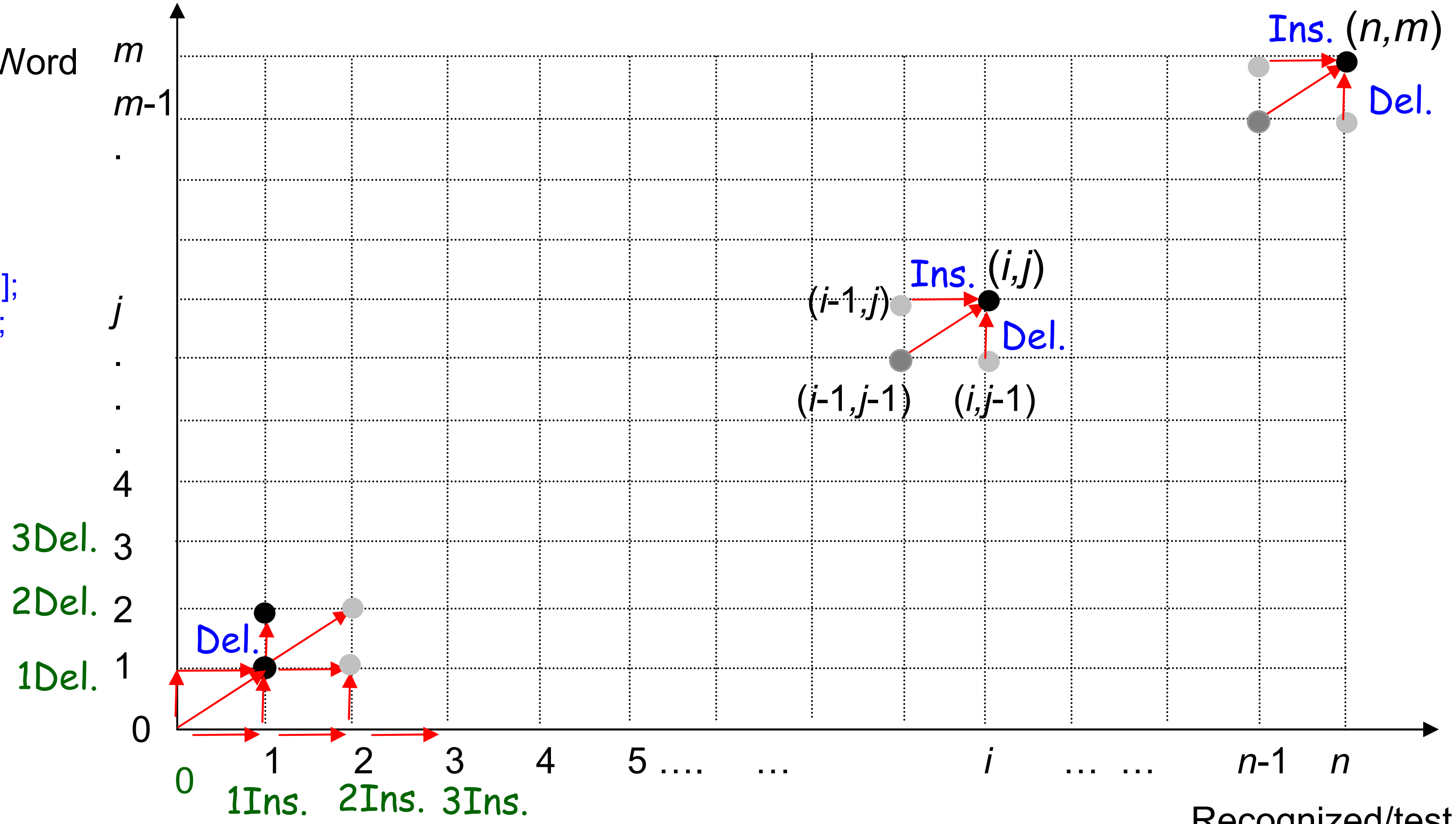
Note: the penalties for substitution, deletion and insertion errors are all set to be 1 here

Measures of ASR Performance (cont.)

- A Dynamic Programming Algorithm
  - Initialization

Correct/Reference Word Sequence

```
for (j=1;j<=m;j++)
 { //reference
  grid[0][j] = grid[0][j-1];
  grid[0][j].dir = VERT;
  grid[0][j].score
       += DelPen;
  grid[0][j].del ++;

 }
```



HTK

```
grid[0][0].score = grid[0][0].ins
= grid[0][0].del = 0;
grid[0][0].sub = grid[0][0].hit = 0;
grid[0][0].dir = NIL;
```

```
for (i=1;i<=n;i++) { // test
      grid[i][0] = grid[i-1][0];
      grid[i][0].dir = HOR;
      grid[i][0].score +=InsPen;
      grid[i][0].ins ++;
}
```

Recognized/test Word Sequence

Measures of ASR Performance (cont.)

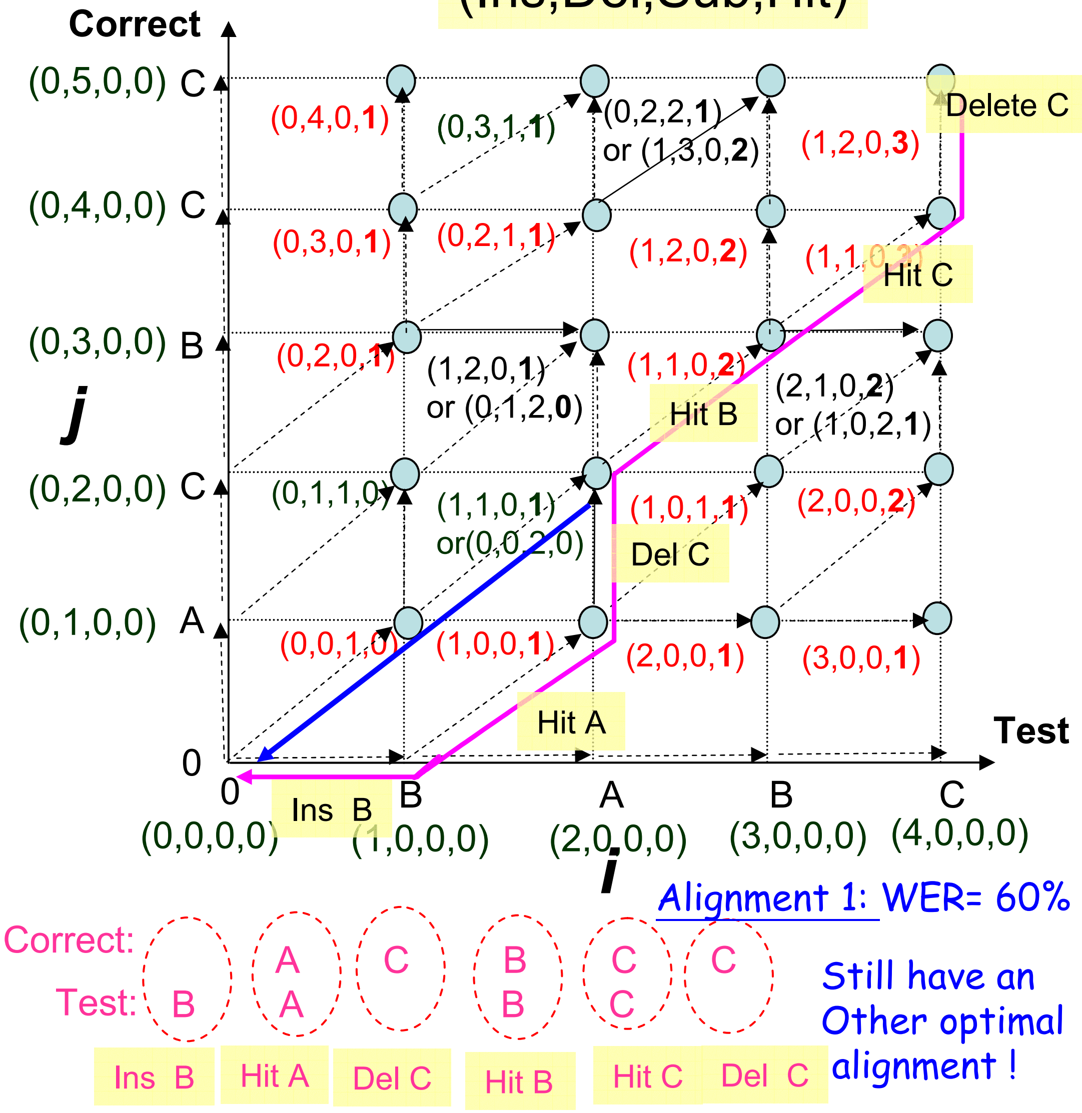
- Program

HTK

```
for (i=1;i<=n;i++) //test
 {   gridi = grid[i]; gridi1 = grid[i-1];
     for (j=1;j<=m;j++) //reference
     {       h = gridi1[j].score +insPen;
             d = gridi1[j-1].score;
             if (lRef[j] != lTest[i])
                d += subPen;
             v = gridi[j-1].score + delPen;
             if (d<=h && d<=v) { /* DIAG = hit or sub */
                gridi[j] = gridi1[j-1]; //structure assignment
                gridi[j].score = d;
                gridi[j].dir = DIAG;
                if (lRef[j] == lTest[i])   ++gridi[j].hit;
                else  ++gridi[j].sub;
             }
              else if (h<v) {         /* HOR = ins */
                gridi[j] = gridi1[j];     //structure assignment
                gridi[j].score = h;
                gridi[j].dir = HOR;
                ++ gridi[j].ins;
             }
             else {                   /* VERT = del */
                gridi[j] = gridi[j-1];    //structure assignment
                gridi[j].score = v;
                gridi[j].dir = VERT;
                ++gridi[j].del;   }
     }              /* for j */
 } /* for i */
```

- Example 1

(Ins,Del,Sub,Hit)



Alignment 1: WER= 60%

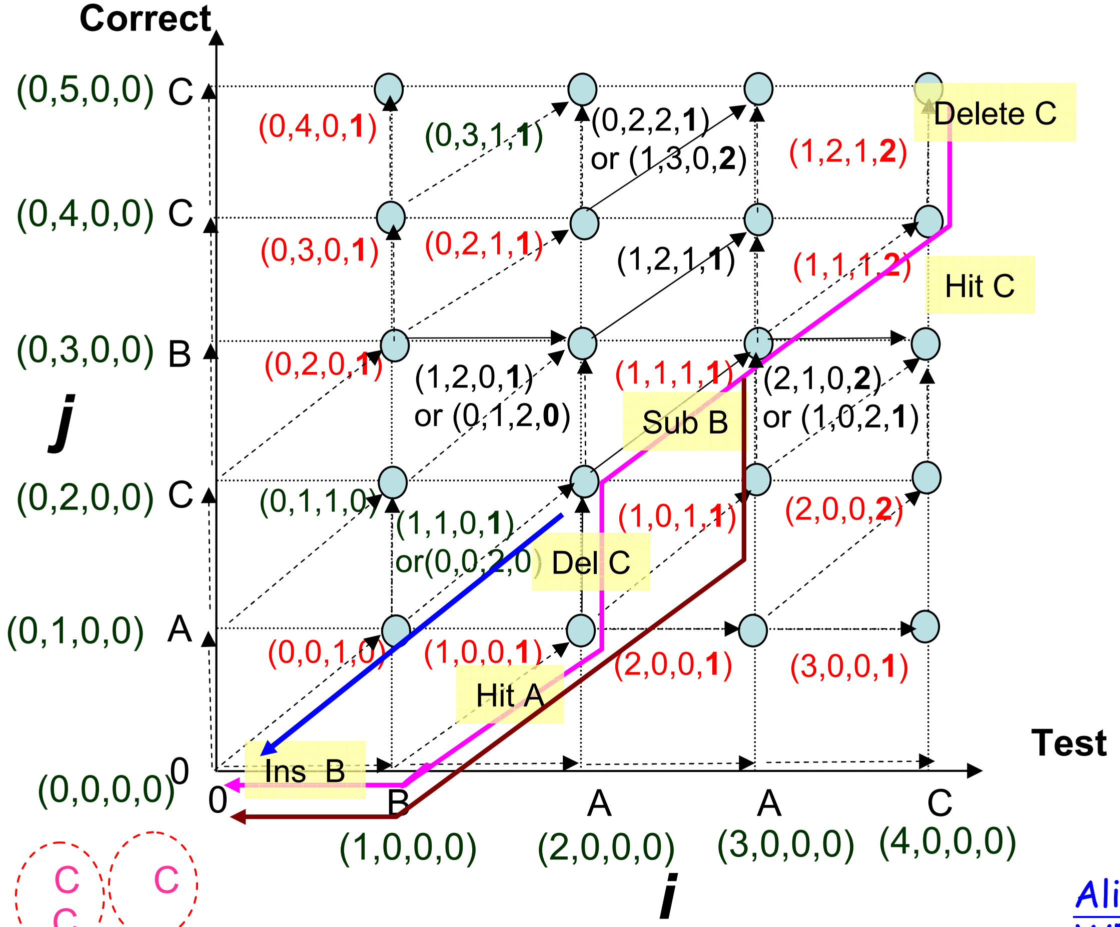
Still have an Other optimal alignment !

Measures of ASR Performance (cont.)

- Example 2

Note: the penalties for substitution, deletion and insertion errors are all set to be 1 here

(Ins,Del,Sub,Hit)



Alignment 1: WER= 80%

Correct:		A	C	B	C	C
Test:	B	A		A	C	
	Ins B	Hit A	Del C	Sub B	Hit C	Del C

Correct:	A	C	B	C	C
Test:	B	A	A	C	
	Sub A	Sub C	Sub B	Hit C	Del C

Alignment 2: WER=80%

Alignment 3: WER=80%

Correct:		A	C	B	C	C
Test:	B	A	A		C	
	Ins B	Hit A	Sub C	Del B	Hit C	Del C

Measures of ASR Performance (cont.)

- Two common settings of different penalties for substitution, deletion, and insertion errors

```
/* HTK error penalties */
 subPen = 10;
 delPen = 7;
 insPen = 7;

/* NIST error penalties*/
 subPenNIST = 4;
 delPenNIST = 3;
 insPenNIST = 3;
```

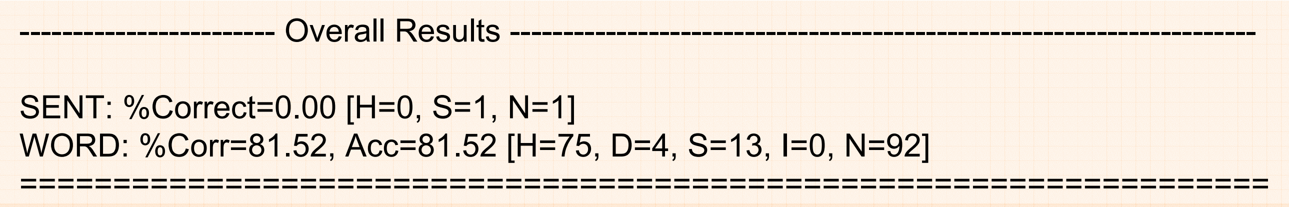
Homework-2B

- Measures of ASR Performance

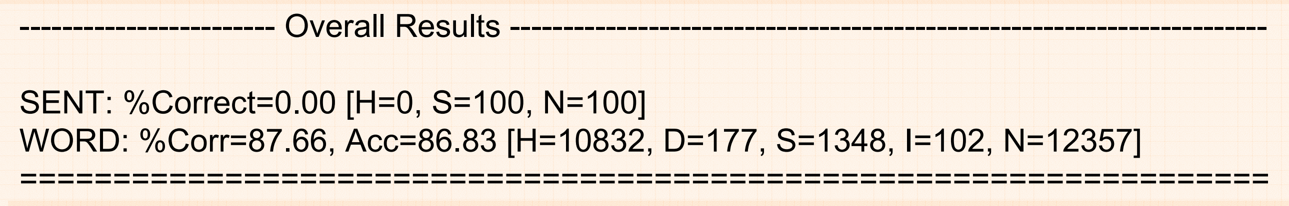
Reference	**ASR Output**
100000 100000 桃	100000 100000 桃
100000 100000 芝	100000 100000 芝
100000 100000 颱	100000 100000 颱
100000 100000 風	100000 100000 風
100000 100000 重	100000 100000 重
100000 100000 創	100000 100000 創
100000 100000 花	100000 100000 花
100000 100000 蓮	100000 100000 蓮
100000 100000 光	100000 100000 光
100000 100000 復	100000 100000 復
100000 100000 鄉	100000 100000 鄉
100000 100000 大	**100000 100000 打**
100000 100000 興	**100000 100000 新**
100000 100000 村	100000 100000 村
100000 100000 死	**100000 100000 次**
100000 100000 傷	100000 100000 傷
100000 100000 慘	**100000 100000 殘**
100000 100000 重	**100000 100000 周**
100000 100000 感	100000 100000 感
100000 100000 觸	100000 100000 觸
100000 100000 最	100000 100000 最
100000 100000 多	100000 100000 多
......	....

Homework-2B (count.)

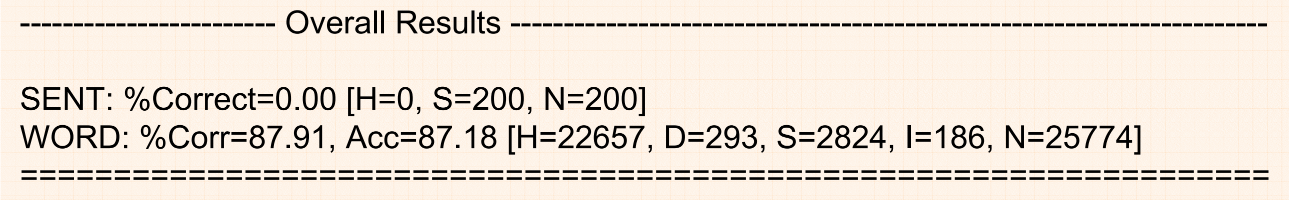
- 506 BN stories of ASR outputs
  - Report the CER (character error rate) of the first one, 100, 200, and 506 stories
  - The result should show the number of substitution, deletion and insertion errors



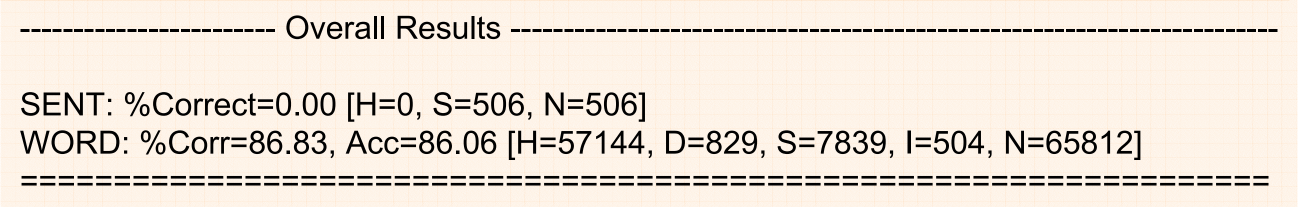
```
------------------------ Overall Results -------------------------------------------------------------------

SENT: %Correct=0.00 [H=0, S=1, N=1]
WORD: %Corr=81.52, Acc=81.52 [H=75, D=4, S=13, I=0, N=92]
======================================================================
```



```
------------------------ Overall Results -------------------------------------------------------------------

SENT: %Correct=0.00 [H=0, S=100, N=100]
WORD: %Corr=87.66, Acc=86.83 [H=10832, D=177, S=1348, I=102, N=12357]
======================================================================
```

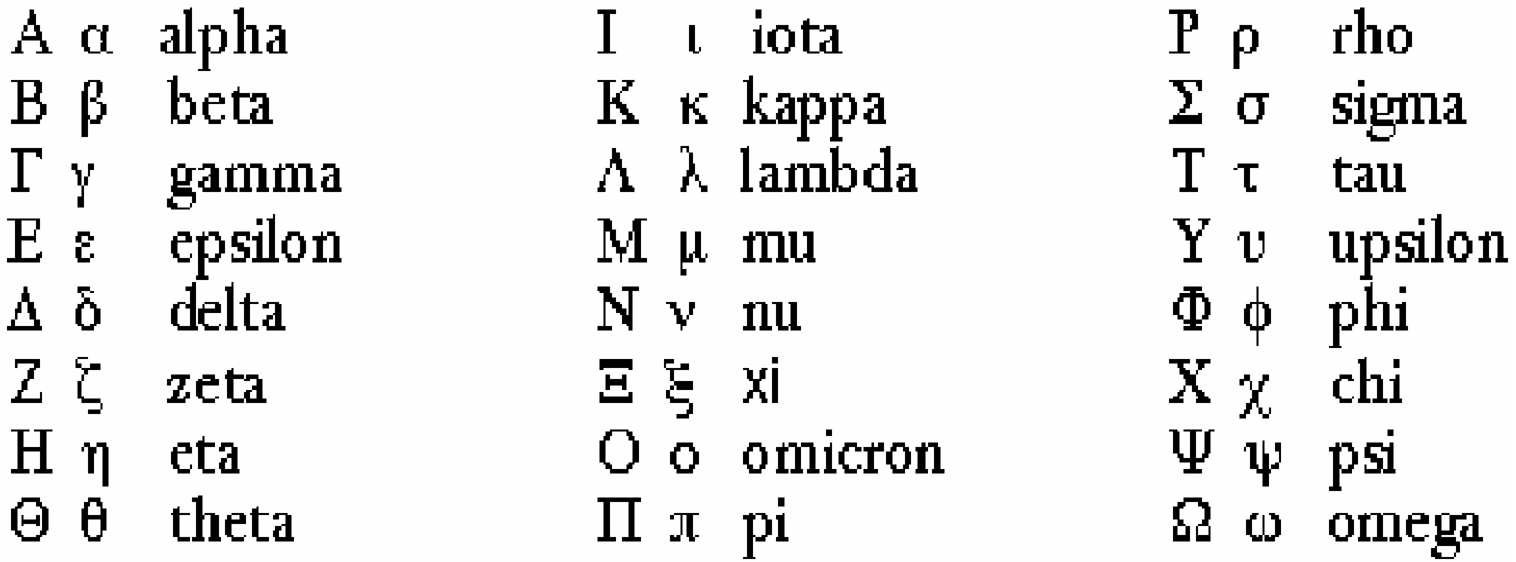


```
------------------------ Overall Results -------------------------------------------------------------------

SENT: %Correct=0.00 [H=0, S=200, N=200]
WORD: %Corr=87.91, Acc=87.18 [H=22657, D=293, S=2824, I=186, N=25774]
======================================================================
```



```
------------------------ Overall Results -------------------------------------------------------------------

SENT: %Correct=0.00 [H=0, S=506, N=506]
WORD: %Corr=86.83, Acc=86.06 [H=57144, D=829, S=7839, I=504, N=65812]
======================================================================
```

Symbols for Mathematical Operations



Α	α	alpha	Ι	ι	iota	Ρ	ρ	rho
Β	β	beta	Κ	κ	kappa	Σ	σ	sigma
Γ	γ	gamma	Λ	λ	lambda	Τ	τ	tau
Ε	ε	epsilon	Μ	μ	mu	Υ	υ	upsilon
Δ	δ	delta	Ν	ν	nu	Φ	φ	phi
Ζ	ζ	zeta	Ξ	ξ	xi	Χ	χ	chi
Η	η	eta	Ο	ο	omicron	Ψ	ψ	psi
Θ	θ	theta	Π	π	pi	Ω	ω	omega