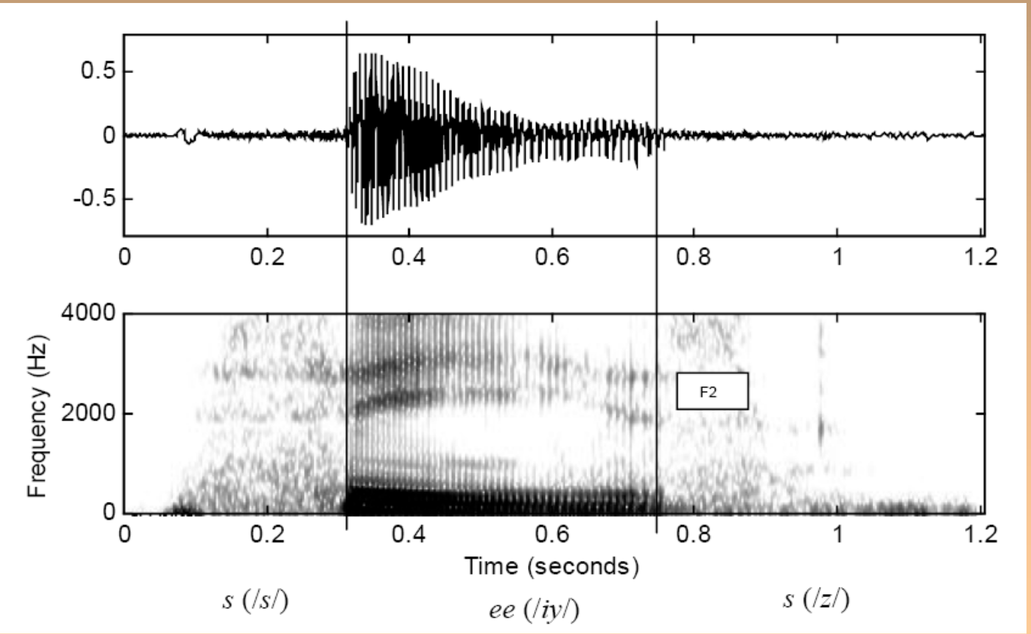
Gaussian Mixture Model and the EM algorithm in Speech Recognition

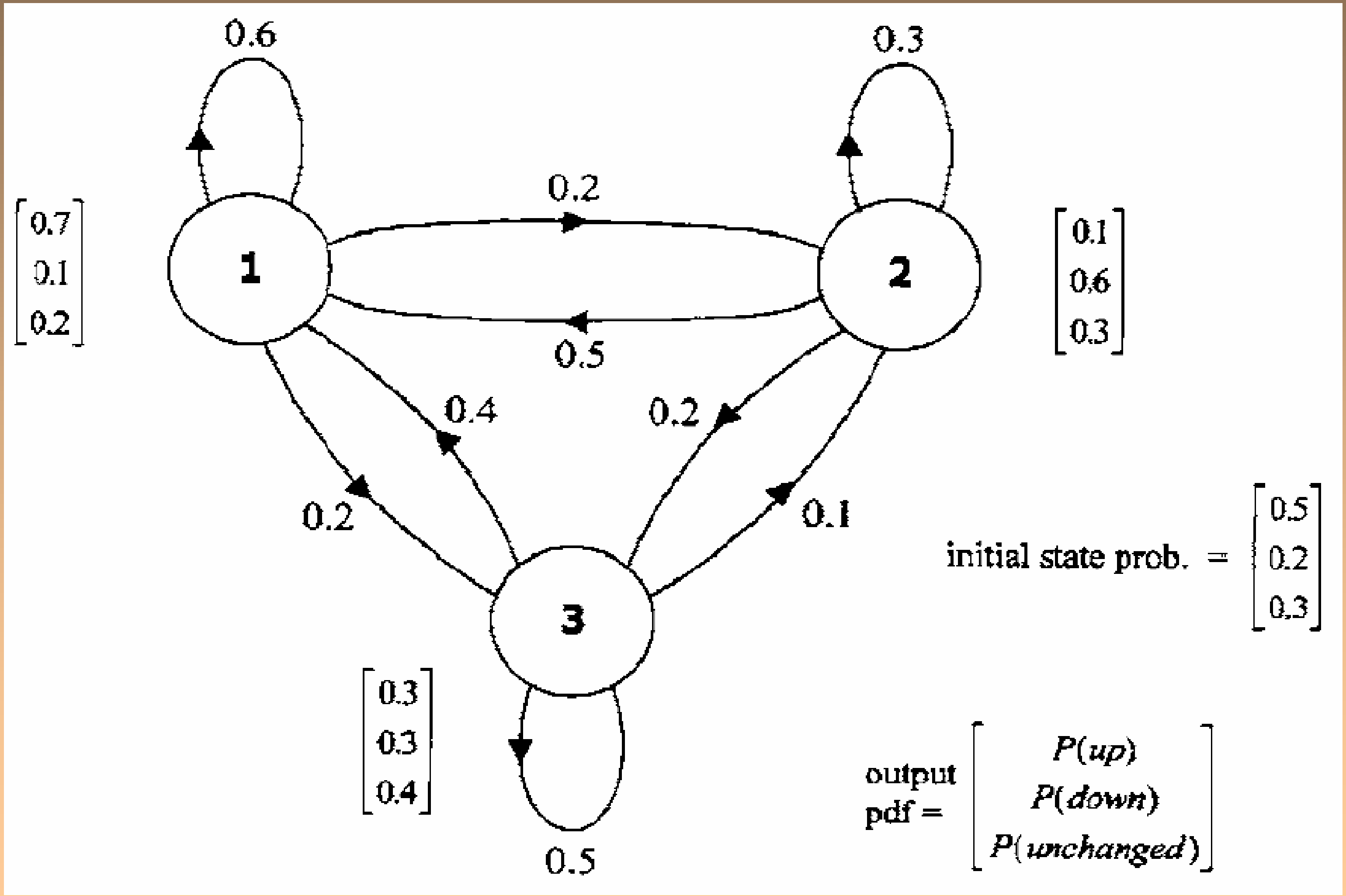
Puskás János-Pál

Speech Recognition

- Develop a method for computers to "understand" speech using mathematical methods



The Hidden Markov Model



First-order observable Markov Model

- a set of states
  - Q = q1, q2…qN; the state at time t is qt
- Current state only depends on previous state

$$P(q_i \mid q_1 \ldots q_{i-1}) = P(q_i \mid q_{i-1})$$

- Transition probability matrix A

$$a_{ij} = P(q_t = j \mid q_{t-1} = i) \quad 1 \le i, j \le N$$

- Special initial probability vector π

$$\pi_i = P(q_1 = i) \quad 1 \le i \le N$$

- Constraints:

$$\sum_{j=1}^{N} a_{ij} = 1; \quad 1 \le i \le N$$

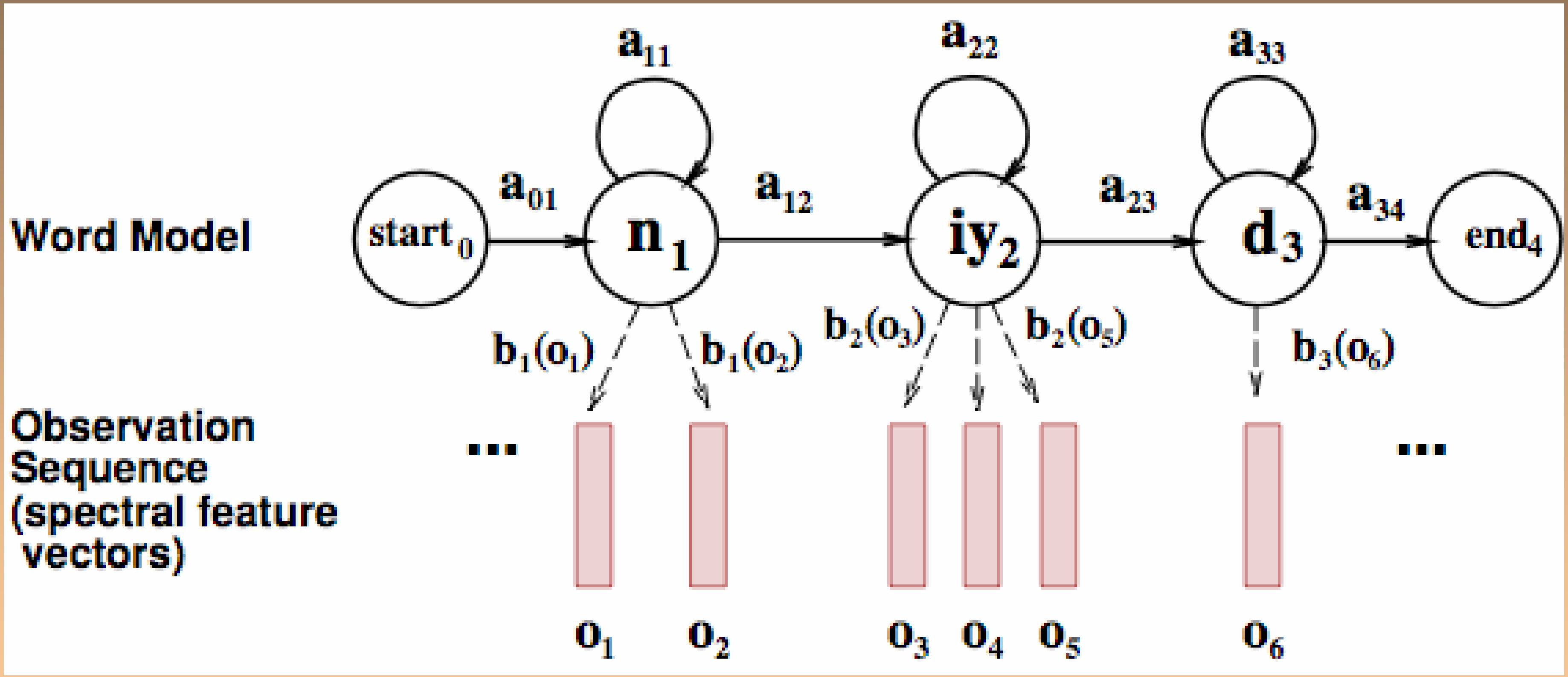
$$\sum_{j=1}^{N} \pi_j = 1$$

$$\sum_{k=1}^{M} b_i(k) = 1$$

Problem: how to apply HMM model to continuous observations?

- We have assumed that the output alphabet V has a finite number of symbols
- But spectral feature vectors are real-valued!
- How to deal with real-valued features?
  - Decoding: Given ot, how to compute P(ot|q)
  - Learning: How to modify EM to deal with real-valued features

HMM in Speech Recognition



Gaussian Distribution

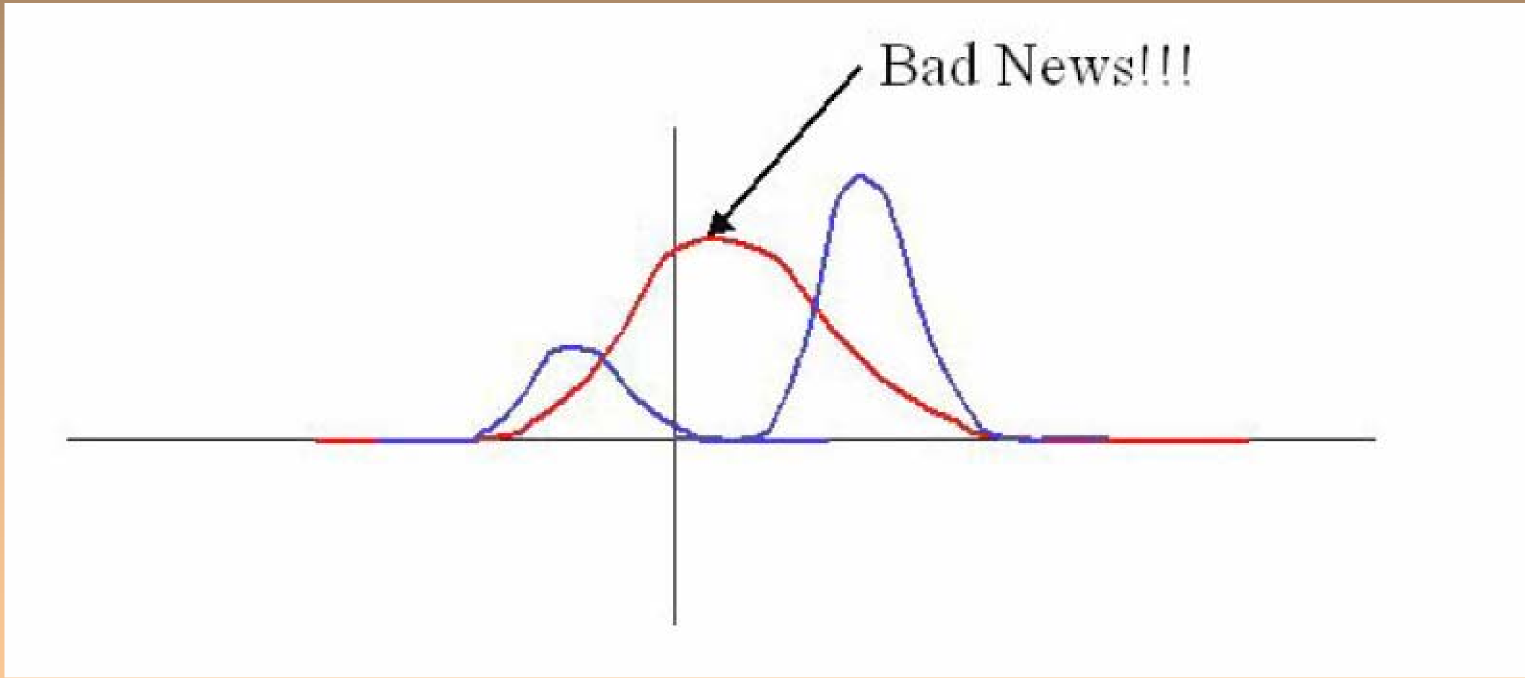
- For a D-dimensional input vector o, the Gaussian distribution with mean μ and positive definite covariance matrix Σ can be expressed as

$$N(o,\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma^{-1}|^{\frac{1}{2}}} e^{-\frac{1}{2}(o-\mu)^{\top}\Sigma^{-1}(o-\mu)}$$

- The distribution is completely described by the D parameters representing μ and the D(D+1)/2 parameters representing the symmetric covariance matrix Σ

Is it enough ?

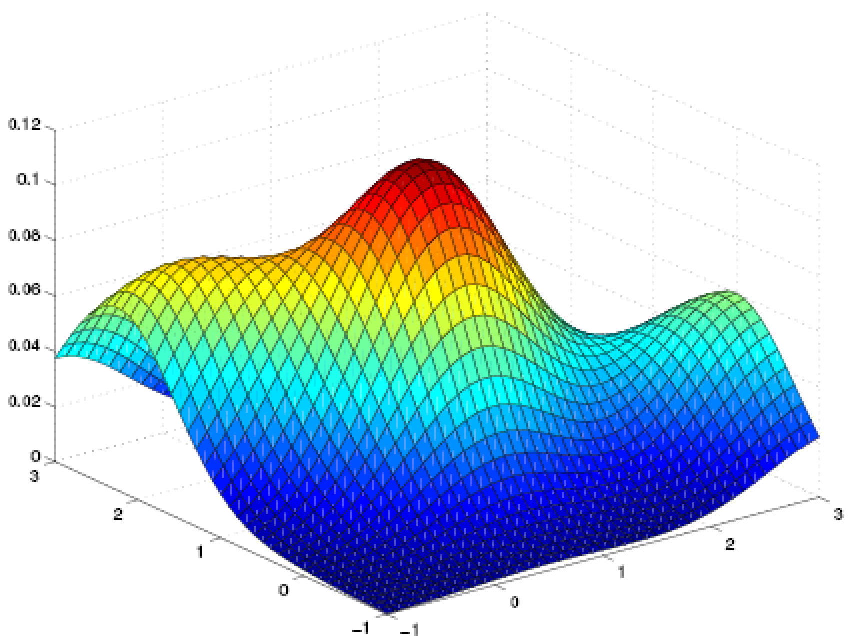
- Single Gaussian may do a bad job of modeling distribution in any dimension:



- Solution: Mixtures of Gaussians

Figure from Chen, Picheney et al slides

Gaussian Mixture Models (GMM)



- Weighted sum of N Gaussians:

$$p(x) = \sum_{i=1}^{N} w_i \mathcal{N}(x, \mu_i, \Sigma_i)$$

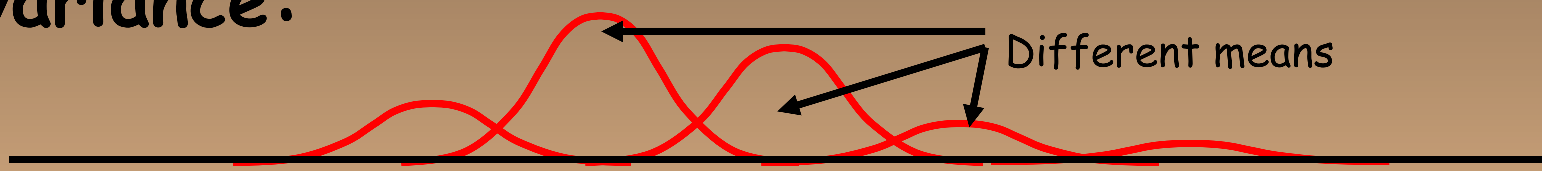
- Can model arbitrary densities.
- Complexity increases *linearly* with N

GMM Estimation

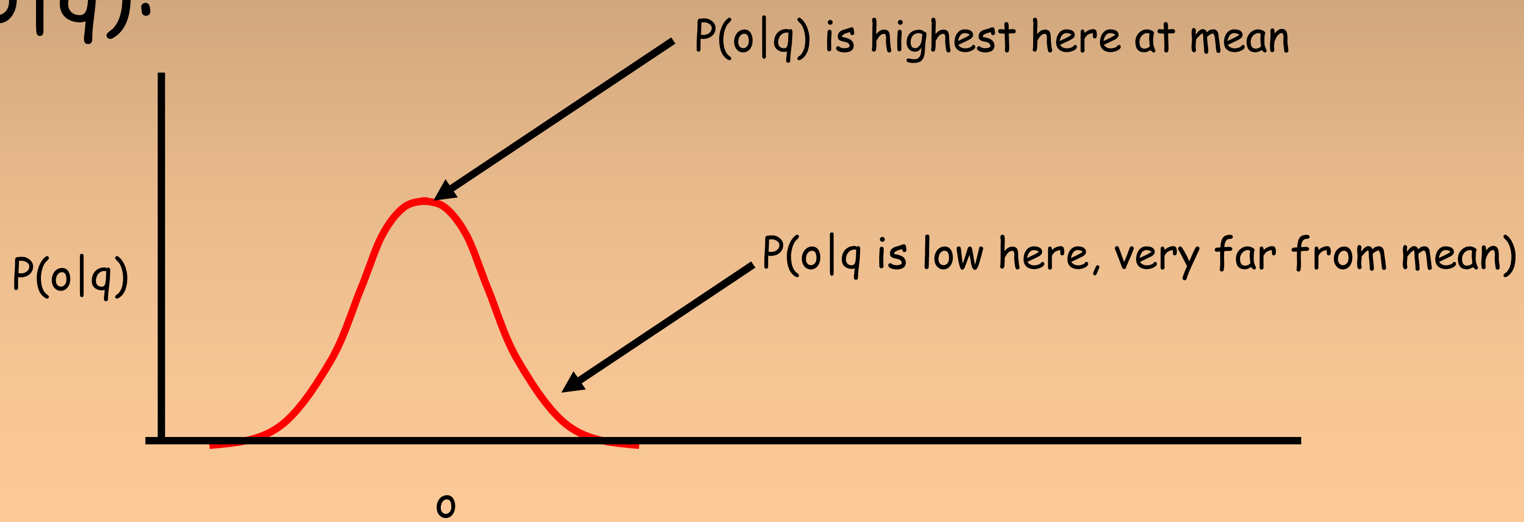
- We will assume that the data as being generated by a set of N distinct sources, but that we only observe the input observation ot without knowing from which source it comes.
- Summary: each state has a likelihood function parameterized by:
  - M Mixture weights
  - M Mean Vectors of dimensionality D
  - Either
    - M Covariance Matrices of DxD
  - Or more likely
    - M Diagonal Covariance Matrices of DxD
    - which is equivalent to
    - M Variance Vectors of dimensionality D

Gaussians for Acoustic Modeling

A Gaussian is parameterized by a mean and a variance:



- P(o|q):



The EM Algorithm

- The EM algorithm is an iterative algorithm that has two steps.
  - In the Expectation step, it tries to "guess" the values of the zt's.
  - In the Maximization step, it updates the parameters of our models based on our guesses.

- The random variables zt indicates which of the N Gaussians each ot had come from.
- Note that the zt's are latent random variable, meaning they are hidden/unobserved. This is what make our estimation problem difficult.

The EM Algorithm in Speech Recognition

The Posteriori Probability (zt) :
("fuzzy membership" of ot to ith gaussian)

$$p_i(o_t) = \frac{m_i N_i(o_t)}{\sum_{k=1}^{M} m_k N_k(o_t)}$$

Mixture weight update:

$$m_i = \frac{1}{T}\sum_{t=1}^{T} p_i(o_t)$$

Mean vector update:

$$\mu_i = \frac{\sum_{t=1}^{T} p_i(o_t) o_t}{\sum_{t=1}^{T} p_i(o_t)}$$

Covariance matrix update:

$$\Sigma_i = \frac{\sum_{t=1}^{T} p_i(x_t) o_i^2}{\sum_{t=1}^{T} p_t(o_t)} - \mu_i^2$$

Baum-Welch for Mixture Models

- Let's define the probability of being in state *j* at time *t* with the *k*th mixture component accounting for ot:

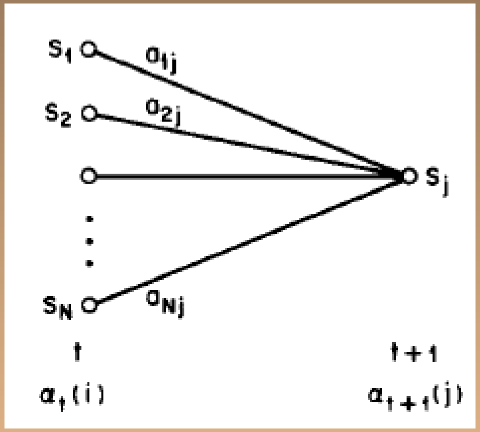
$$\xi_{tm}(j) = \frac{\sum_{i=1}^{N} \alpha_{t-1}(j) a_{ij} c_{jm} b_{jm}(o_t) \beta_j(t)}{\alpha_F(T)}$$

- Now,

$$\bar{\mu}_{jm} = \frac{\sum_{t=1}^{T} \xi_{tm}(j) o_t}{\sum_{t=1}^{T} \sum_{k=1}^{M} \xi_{tk}(j)} \qquad \bar{c}_{jm} = \frac{\sum_{t=1}^{T} \xi_{tm}(j)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \xi_{tk}(j)} \qquad \bar{\Sigma}_{jm} = \frac{\sum_{t=1}^{T} \xi_{tm}(j)(o_t - \mu_j)(o_t - \mu_j)^T}{\sum_{t=1}^{T} \sum_{k=1}^{M} \xi_{tm}(j)}$$

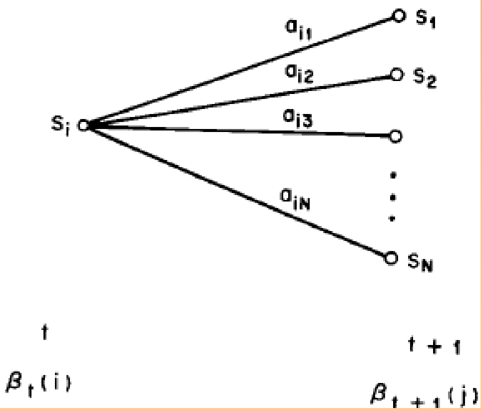
The Forward and Backward algorithms

- Forward (α) algorithm



$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1})$$

- Backward (β) algorithm



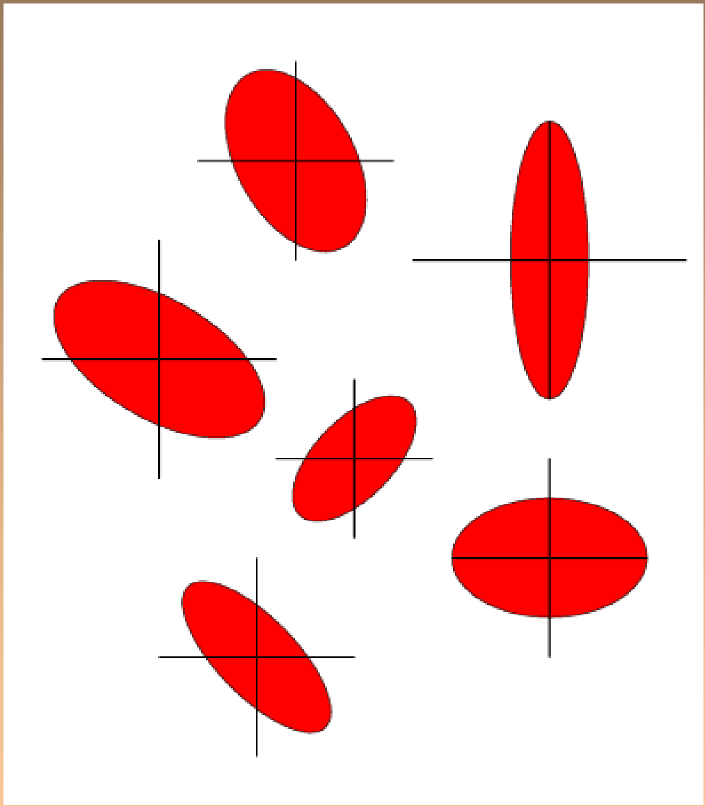
$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$

How to train mixtures?

- Choose M (often 16; or can tune M optimally)
- Then can do various splitting or clustering algorithms
- One simple method for "splitting":
  - Compute global mean μ and global variance
  - Split into two Gaussians, with means $\mu \pm \varepsilon$ (sometimes ε is 0.2σ
  - Run Forward-Backward to retrain
  - Go to 2 until we have 16 mixtures
- Or choose starting clusters with the K-means algorithm

The Covariance Matrix

- Represents correlations in a Gaussian.
- Symmetric matrix.
- Positive definite.
- D(D+1)/2 parameters when x has D dimensions.

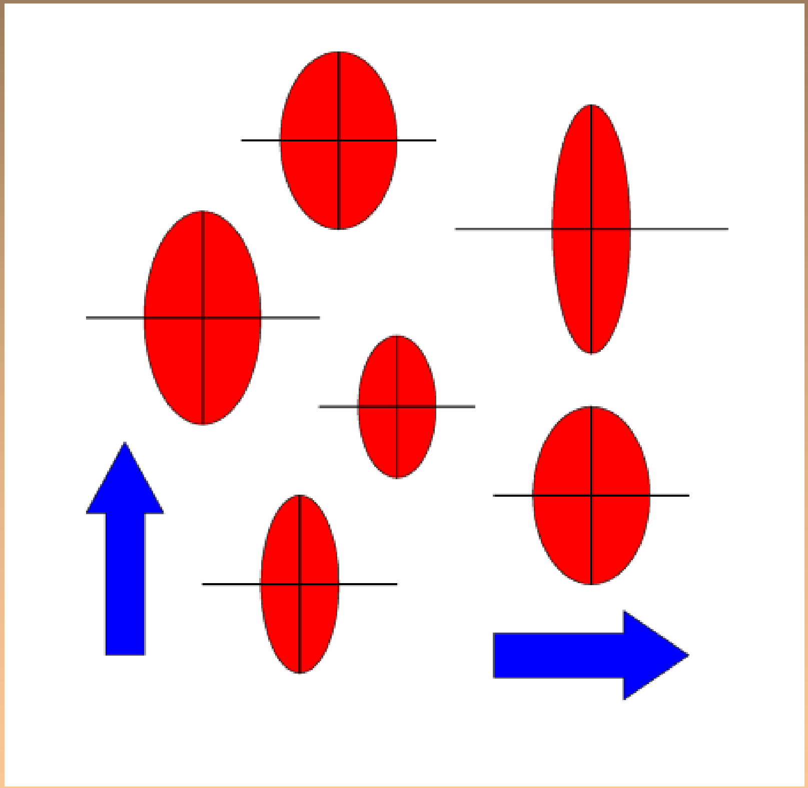


But: assume diagonal covariance

- I.e., assume that the features in the feature vector are uncorrelated
- This isn't true for FFT features, but is true for MFCC features.
- Computation and storage much cheaper if diagonal covariance.
- I.e. only diagonal entries are non-zero
- Diagonal contains the variance of each dimension σ_{ii}^2
- So this means we consider the variance of each acoustic feature (dimension) separately

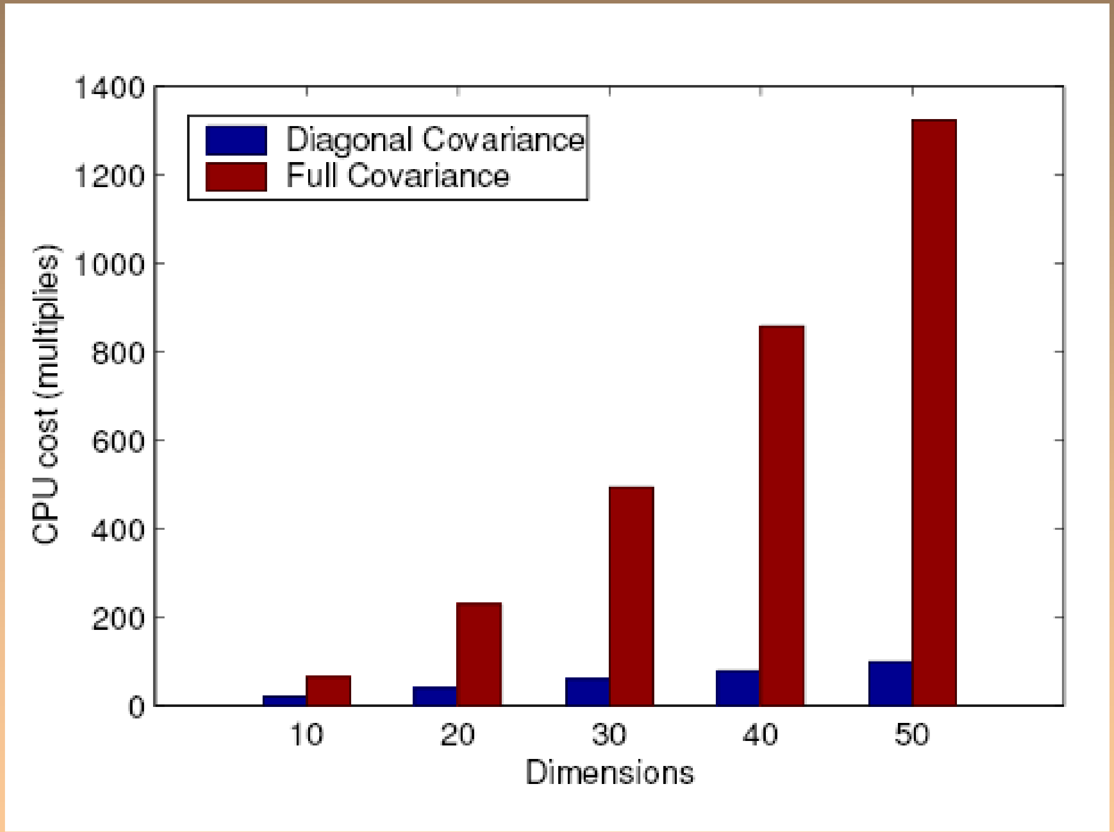
Diagonal Covariance Matrix

- Simplified model:
- Assumes orthogonal principal axes.
- D parameters.
- Assumes independence between components of x.

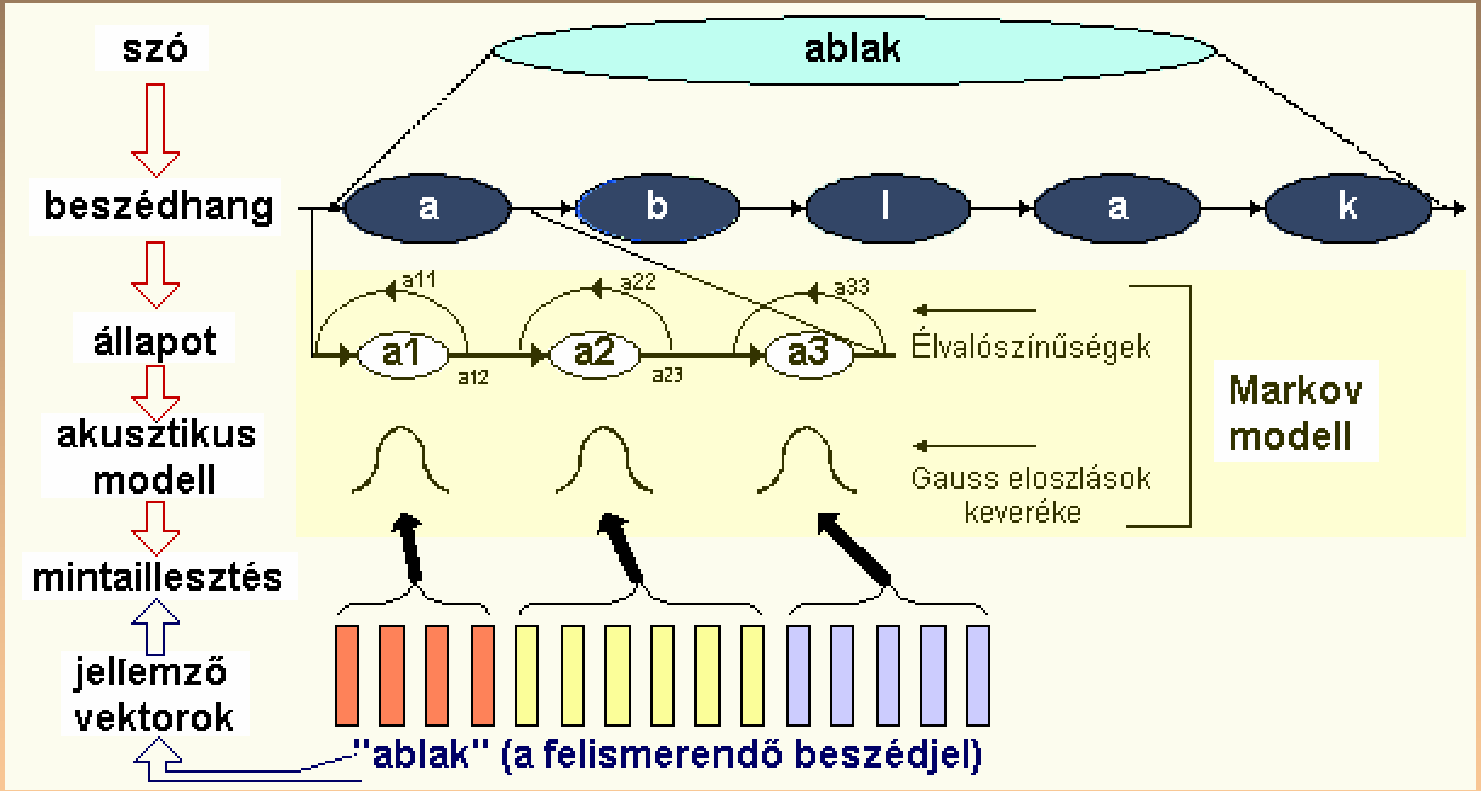


$$f(x \mid \mu, \sigma) = \frac{1}{2\pi^{D/2} \prod_{d=1}^{D} \sigma_d^{\,2}} \exp\left(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d - \mu_d)^2}{\sigma_d^{\,2}}\right)$$

Cost of Gaussians in High Dimensions



How does the system work



References

- Lawrence R. Rabiner - A Tutorial on Hidden Markov Models and Selected Aplivations in Speech Recognition
- Magyar nyelvi beszédtechnológiai alapismeretek Multimédiás szoftver CD - Nikol Kkt. 2002.
- Dan Jurafsky – “CS Speech Recognition and Synthesis” Lecture 8 -10 Stanford University, 2005