AUTOMATIC SPEECH RECOGNITION

Lecture 9

Speech Recognition Using HMMs

Last Lecture

- Review of Probability
- Overview of HMMs

This Lecture (+Supplementary No.1 & No.2)

- HMM (Discrete HMM) parameters (Revisited)
- Isolated-word Speech Recognition
  - Forward/Backward Algorithm
- Speech Recognition based on subword units
  - Viterbi Algorithm
- Parameter Training
  - Baum-Welch Re-estimation Algorithm
- Continuous HMM
  - Multivariate Gaussian PDF
  - Gaussian Mixtures

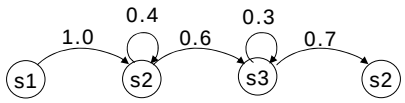
Discrete HMM Parameters

- Number of States and Connectivity (Topology)
- State Transition Probability (A)
- Initial State Probability (π)
- Output Probability Density Function (B)

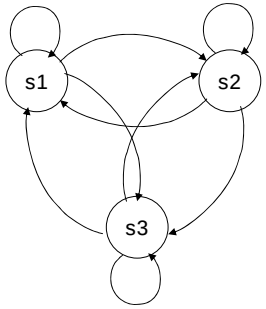
State Transition Probability

- State Transition Matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.4 & 0.6 & 0.0 \\ 0.0 & 0.0 & 0.3 & 0.7 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

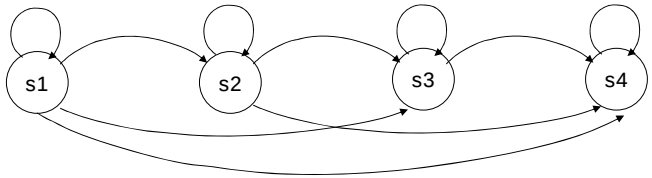


Ergodic Model



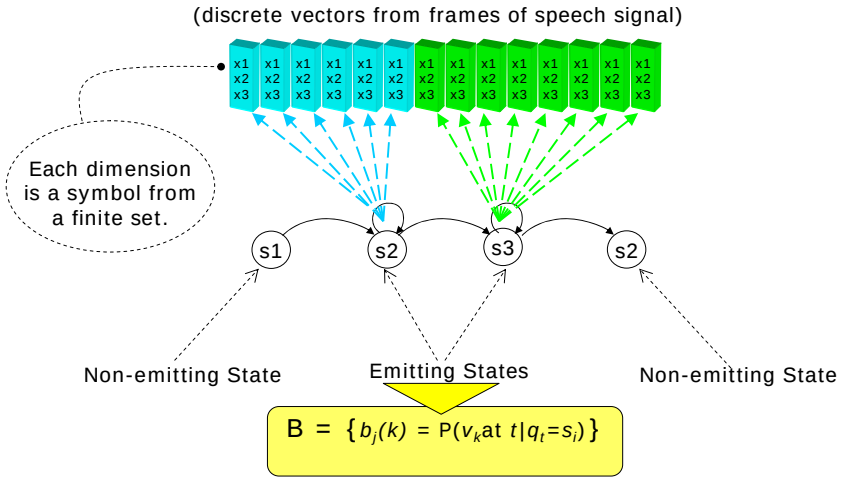
- An HMM is called ergodic or fully-connected model when every state of the model can be reached (in a single step) from every other state of the model.

Left-to-Right Model

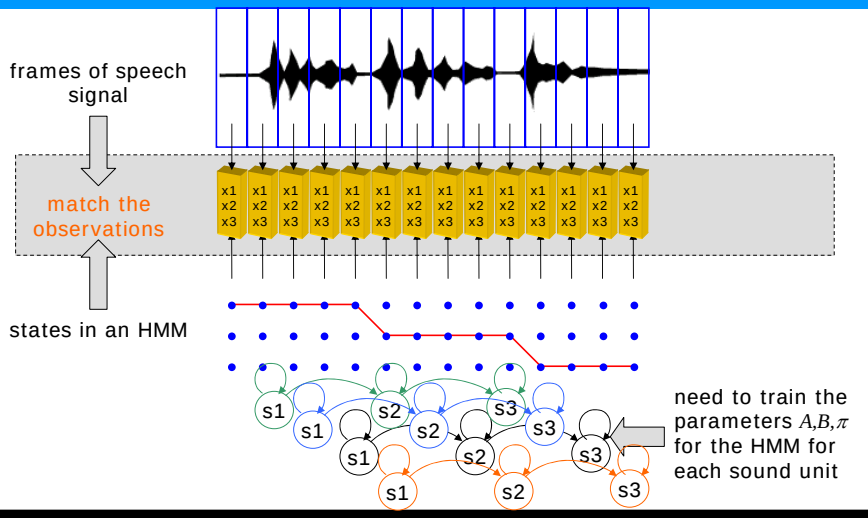


- The underlying state sequence associated with the model has the property that, as time increases, the state index increases.
- The left-to-right model exhibits the desirable property of being readily able to model speech whose properties change over time in a successive manner.

State Output

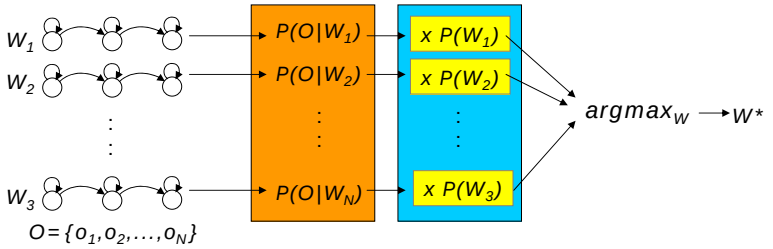


Acoustic Modeling Using HMMs

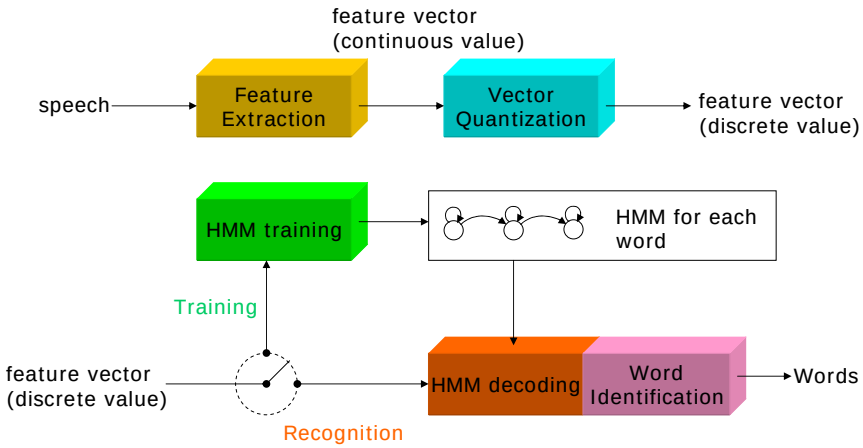


Modeling words with HMMs

- We seek to find $W^* = argmax_W(P(W|O))$
- $W^* = argmax_W(P(O|W)P(W)/P(O))$
- W is an element of set $\{W_1,W_2,W_3,..,W_N\}$
- So, we compare $P(O|W_i)$, $i=1,2,...,N$, and pick Wi that gives maximal $P(O|W_i)$.

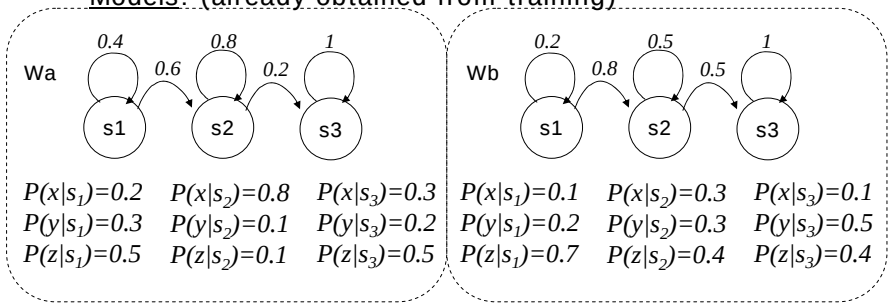


Speech Recognizer Based on discrete HMMs



Example

- 1D feature vector consisting of symbols {x,y,z}
- Vocab: {Wa,Wb}
- Observations: O = {x, x, z, y}
- Models: (already obtained from training)



- Decision rule: Pick Wi that maximize P(O|Wi)
- Constraints: 1st observation generated by s1. The last by s3.

Calculating P(O|A,B,π)

- **Evaluation Problem**

Given the observation sequence $\boldsymbol{O} = \{O_1,O_2,O_3,\ldots,O_n\}$ and the model $\lambda = (\boldsymbol{A},\boldsymbol{B},\pi)$, how can the observation sequence probability $P(O|\lambda)$ be computed?

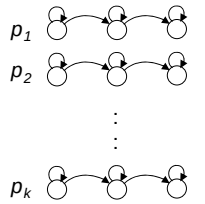
Forward / Backward Algorithm

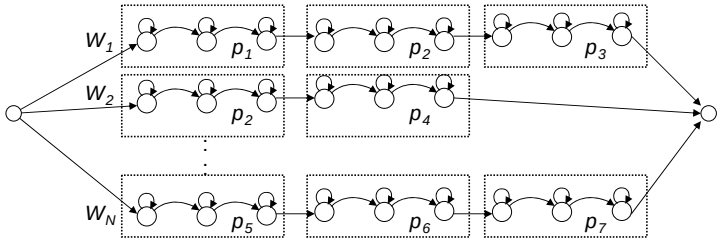
(See *supplementary No.1* for detailed algorithms and complexities)

Making use of subword units

- Suppose:

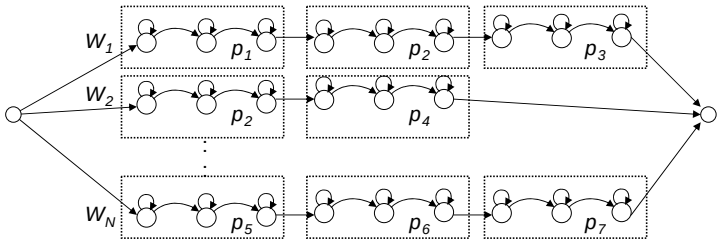
$W_1 = p_1p_2p_3$
$W_2 = p_2p_4$
....
$W_N = p_5p_6p_7$





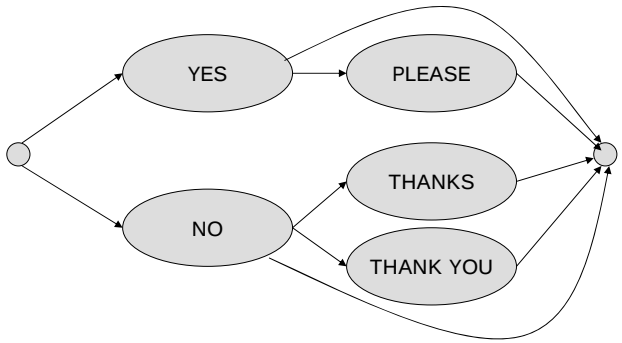
Modeling subword units with HMMS

- Find the best state sequence $Q = \{q_1,q_2,q_3,\ldots,q_n\}$



Speech Recognizer using Subword Units

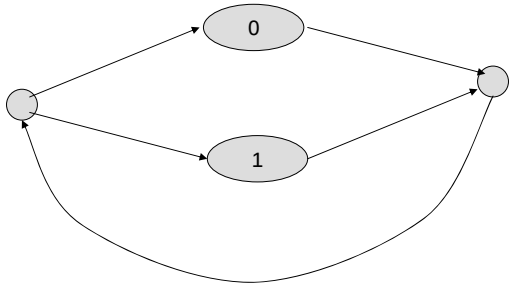
- Suppose a user can say:
  "Yes" / "Yes, please" / "No" / "No. Thanks" / "No. Thank you"
- Find the best state sequence $Q = \{q_1,q_2,q_3,\ldots,q_n\}$



Speech Recognizer using Subword Units

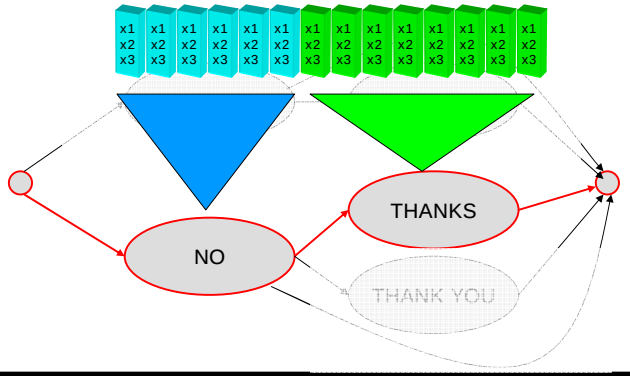
Example

- Suppose the system need to recognize a bit string without knowing its length.



Speech Recognizer using Subword Units

- In this case, the speech recognition task can be considered as finding the "best" path in the HMM network given the sequence of feature vectors.



Finding the Best State Sequence

- Hidden State Sequence Uncovering

  Given the observation sequence $\boldsymbol{O} = \{O_1,O_2,O_3,...,O_n\}$ and the model $\lambda = (\boldsymbol{A},\boldsymbol{B},\pi)$, how can a state sequence $Q=\{q_1,q_2,...,q_T\}$, which is optimal in some sense, be chosen?

Viterbi Algorithm

(See *supplementary No.1* for detailed algorithms and complexities)

Parameter Training

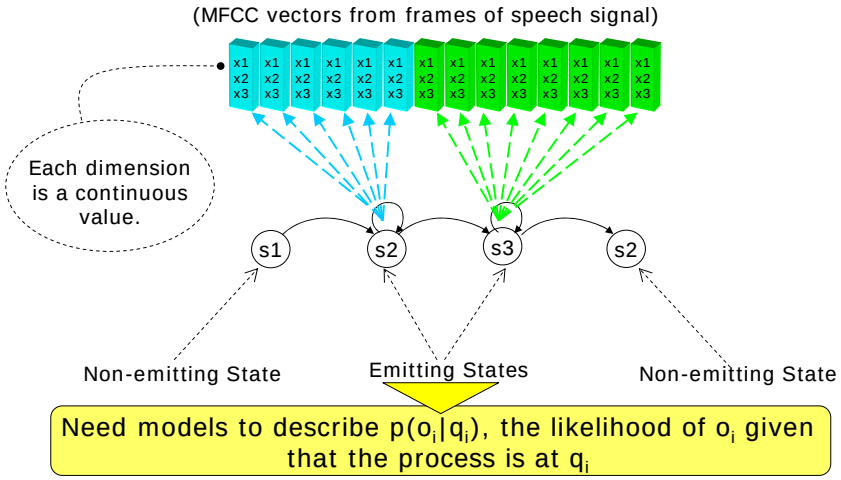
- Training Problem

  Given example observations $\boldsymbol{O}$, how can the model parameters $\lambda = (\boldsymbol{A},\boldsymbol{B},\pi)$ be adjusted to maximize the observation sequence probability $P(O|\lambda)$?

Baum-Welch Re-estimation Algorithm

(See *supplementary No.1* for detailed algorithms and complexities)

Continuous HMM



Output Probability / Likelihood

- Modeled with Multivariate Probability Density Functions (Joint PDF of many random variables)

Output Likelihood

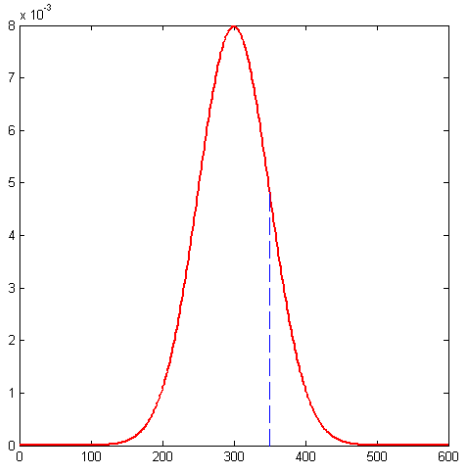
Example:

one-dimensional feature vector, x

If we know that
$p(X=x|q_i) \sim N(300,50)$

Then, $p(X=350|q_i) = 0.0034$

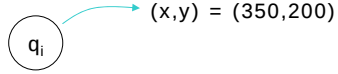
Use Gaussian PDF to calculate.



Output Likelihood

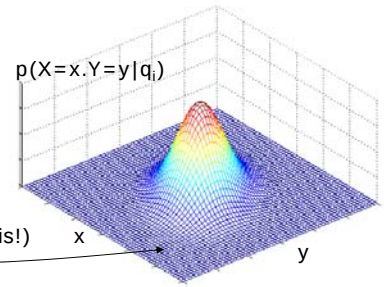
Example:

two-dimensional feature vector, (x,y)



If we know the joint PDF of X and Y, we can calculate $p(X=350, Y=200|q_i)$

$p(X=x.Y=y|q_i)$ is usually modelled with multivariated Gaussian PDF.



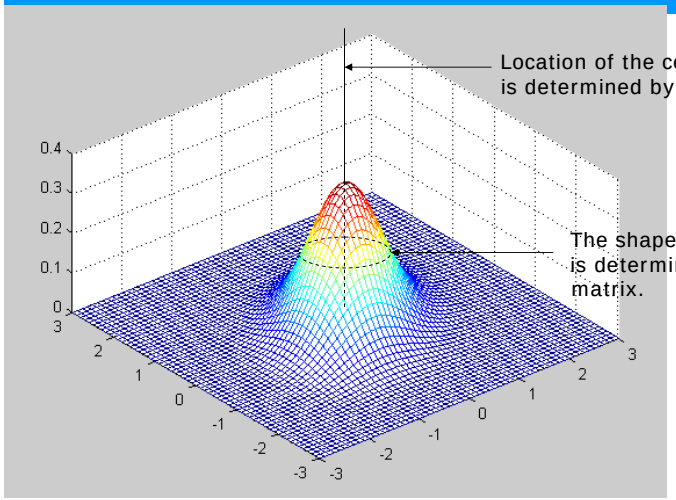
Multi-dimensional (Multivariate) Gaussian PDF

- A multi-dimensional Gaussian PDF can be expressed as:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

d is the number of dimensions
$\mathbf{x} = \{x_1,...,x_d\}$ is the variable vector
$\boldsymbol{\mu} = E[\mathbf{x}] = \{\mu_1,..., \mu_d\}$ is the mean vector
$\boldsymbol{\Sigma}$ is the covariance matrix with elements σ_{ij}

Multi-dimensional Gaussian PDF



Location of the center of the surface is determined by the mean vector

The shape of the surface is determined by the covariance matrix.

The Mean Vector

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \\ \vdots \\ x_{iN} \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \vdots \\ \mu_N \end{bmatrix} = \begin{bmatrix} E[x_1] \\ E[x_2] \\ E[x_3] \\ \vdots \\ E[x_N] \end{bmatrix}$$

Each dimension of the mean vector is the mean of the corresponding random variable in that dimension.

Covariance Matrix

$$\Sigma = E[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T]$$

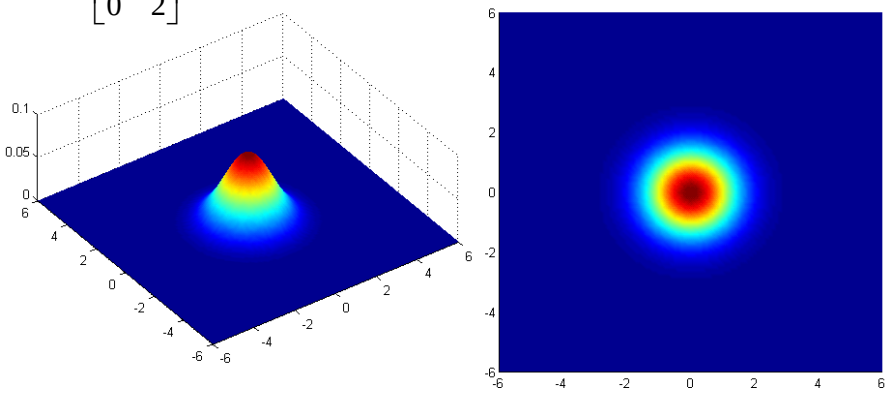
$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \cdots & \sigma_{1,d} \\ \sigma_{21} & \sigma_2^2 & & & \vdots \\ \vdots & & \ddots & & \vdots \\ & & & \sigma_{d-1}^2 & \sigma_{d-1,d} \\ \sigma_{d,1} & \cdots & \cdots & \sigma_{d,d-1} & \sigma_d^2 \end{bmatrix}$$

Diagonal component σ_{ii}^2 is the variance of the random variable in the i^{th} dimension.

$\sigma_{ij} = \sigma_{ji} = E[(x_i-\mu_i)(x_j-\mu_j)] = E(x_i x_j) - \mu_i \mu_j$

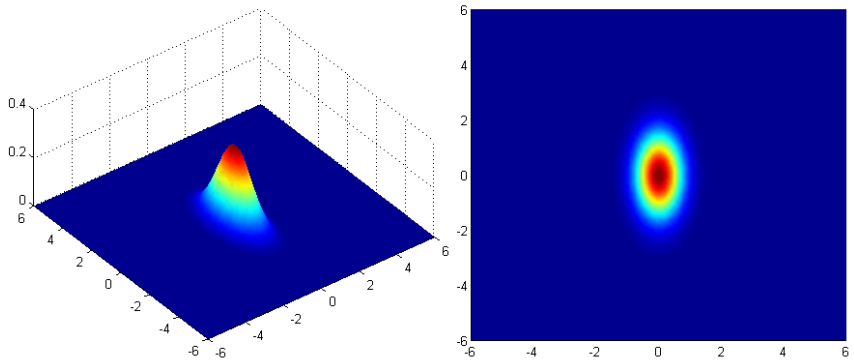
Diagonal Covariance Matrix

$$\Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$



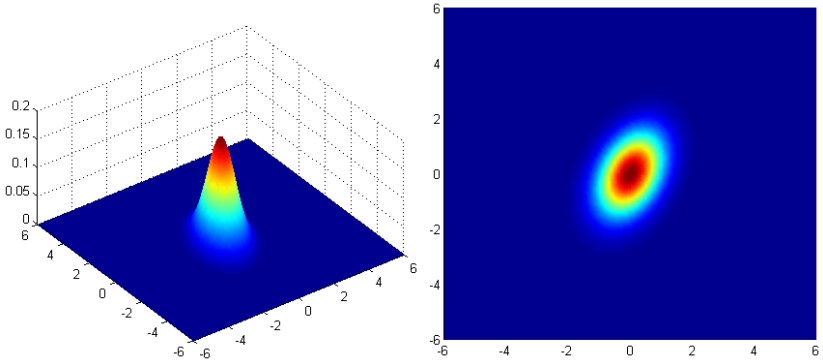
Diagonal Covariance Matrix

$$\Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}$$



General Covariance Matrix

$$\Sigma = \begin{bmatrix} 2 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

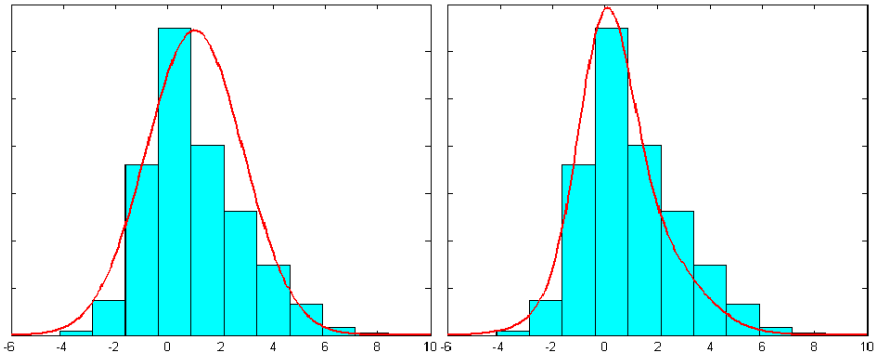


Parameter Training

- For each PDF, its parameters including the mean vector and the covariance matrix has to be estimated from training data.
- Maximum Likelihood estimation of Gaussian parameters is described in details in *Supplementary No.2*

It does not look normal??

- Sometimes, the distribution of the feature vector of interest does not appear Gaussian



Mixture Densities

- PDF is composed of a mixture of m component densities $\{\lambda_1,..., \lambda_m\}$

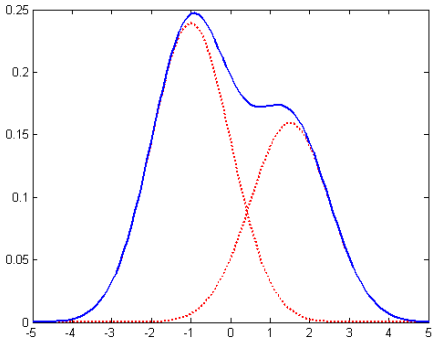
$$p(\mathbf{x}) = \sum_{j=1}^{m} p(\mathbf{x} \mid \lambda_j) P(\lambda_j)$$

- The mixture weights, $P(\lambda_j)$, as well as the parameters for each component $P(\mathbf{x} \mid \lambda_j)$, are typically unknown and needs parameter estimation in the form of "unsupervised learning"

Gaussian Mixtures

- Gaussian Mixtures assume Normal components:

$$p(\mathbf{x} \mid \lambda_j) \sim N(\boldsymbol{\mu}_j, \Sigma_j)$$

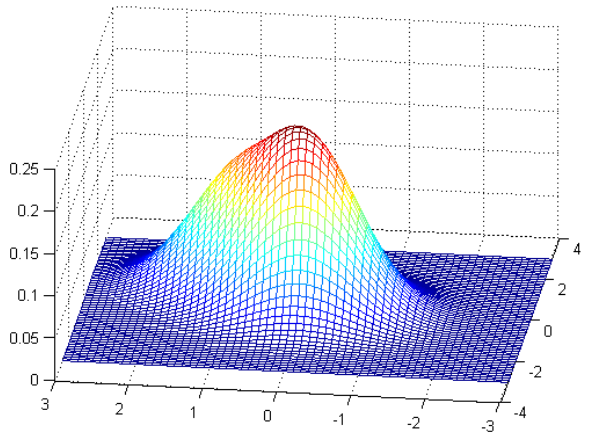


$$p(\mathbf{x}) = \sum_{j=1}^{2} p(\mathbf{x} \mid \lambda_j) P(\lambda_j)$$

$$p(\mathbf{x}) = 0.6\, p(\mathbf{x} \mid \lambda_1) + 0.4\, p(\mathbf{x} \mid \lambda_2)$$

$$p(\mathbf{x} \mid \lambda_1) \sim N(-1,1),$$
$$p(\mathbf{x} \mid \lambda_2) \sim N(1.5,1)$$

2D Mixtures



Parameter Estimations

$$p(\mathbf{x}) = \sum_{j=1}^{m} p(\mathbf{x} \mid \lambda_j) P(\lambda_j)$$

- For Gaussian Mixtures, parameters to be estimated include:
  - The mean vector of every component
  - The covariance matrix of every component
  - The weights associated with every component
- Maximum Likelihood estimation of these parameters must be performed “iteratively”. The detailed algorithm is described in *Supplementary No.2*

Implementation Variations

- Diagonal Gaussians are often used instead of full-covariance ones
  - can reduce number of parameters
  - can model the underlying PDF just as well if enough components are used
- Richter Gaussians share the same mean in order to better model the PDF tails
- Tied-Mixtures share the same parameters across all classes. Only the mixture weights are class specific.

Next Lecture

- Building a speech recognizer using Hidden Markov Models ToolKits (HTK)