Chapter 5

Application of speech & audio signal processing

Basanta Joshi, PhD
basanta@ioe.edu.np

Lecture notes can be downloaded from
www.basantajoshi.com.np

Contents

- Algorithms:
  - Dynamic time warping
  - K-means clustering and Vector quantization
  - Gaussian mixture modeling
  - Hidden Markov modeling
- Automatic Speech Recognition
  - Feature Extraction for ASR
  - Deterministic sequence recognition
  - Statistical Sequence
  - Recognition
  - Language models
- Speaker identification and verification
- Voice response system
- Speech synthesis
  - Basics of articulatory
  - Source-filter
  - Concatenative synthesis

Dynamic Time Wrapping (DTW)

DYNAMIC TIME WRAPPING(DTW)

- an algorithm for measuring similarity between two sequences which may vary in time or speed
- Example – similarity between two speech sequences even if they are spoken at different speed, or even if there is acceleration and decelerations during the course of speech

- Application areas:
  - Automatic speech recognition
  - speaker recognition
  - handwriting matching
  - online signature recognition
  - gait recognition
  - Sign language recognition
  - Gesture recognition
  - Time series clustering
  - Protein sequence alignment
  - …

DTW (overview)

- Speech Recognition Process:
  - Compare incoming speech with stored template
  - Template with lowest distance measure from incoming pattern is considered recognized
  - Lowest distance measure is given by Dynamic programming called DTW
- DTW - Extremely efficient algorithm for time series similarity
  - it allows for *elastic* shifting in the time domain and matches sequences that are similar but out of phase.

DTW – How it works??

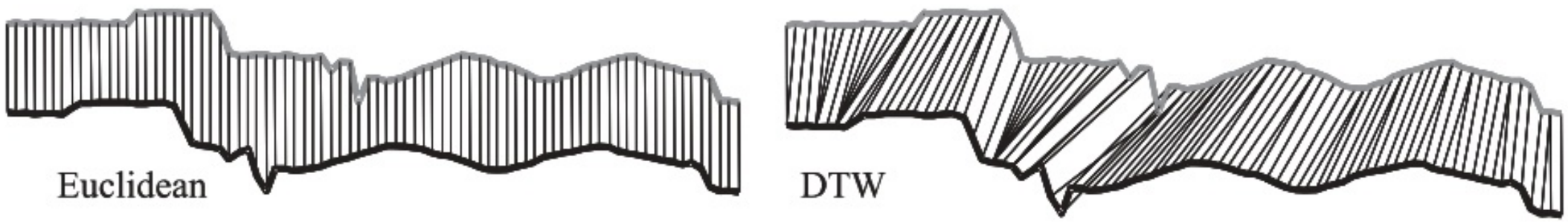


Fig. 1. Note that, while the two time series have an overall similar shape, they are not aligned in the time axis. Euclidean distance, which assumes the ith point in one sequence is aligned with the ith point in the other, will produce a pessimistic dissimilarity measure. The nonlinear dynamic time warped alignment allows a more intuitive distance measure to be calculated

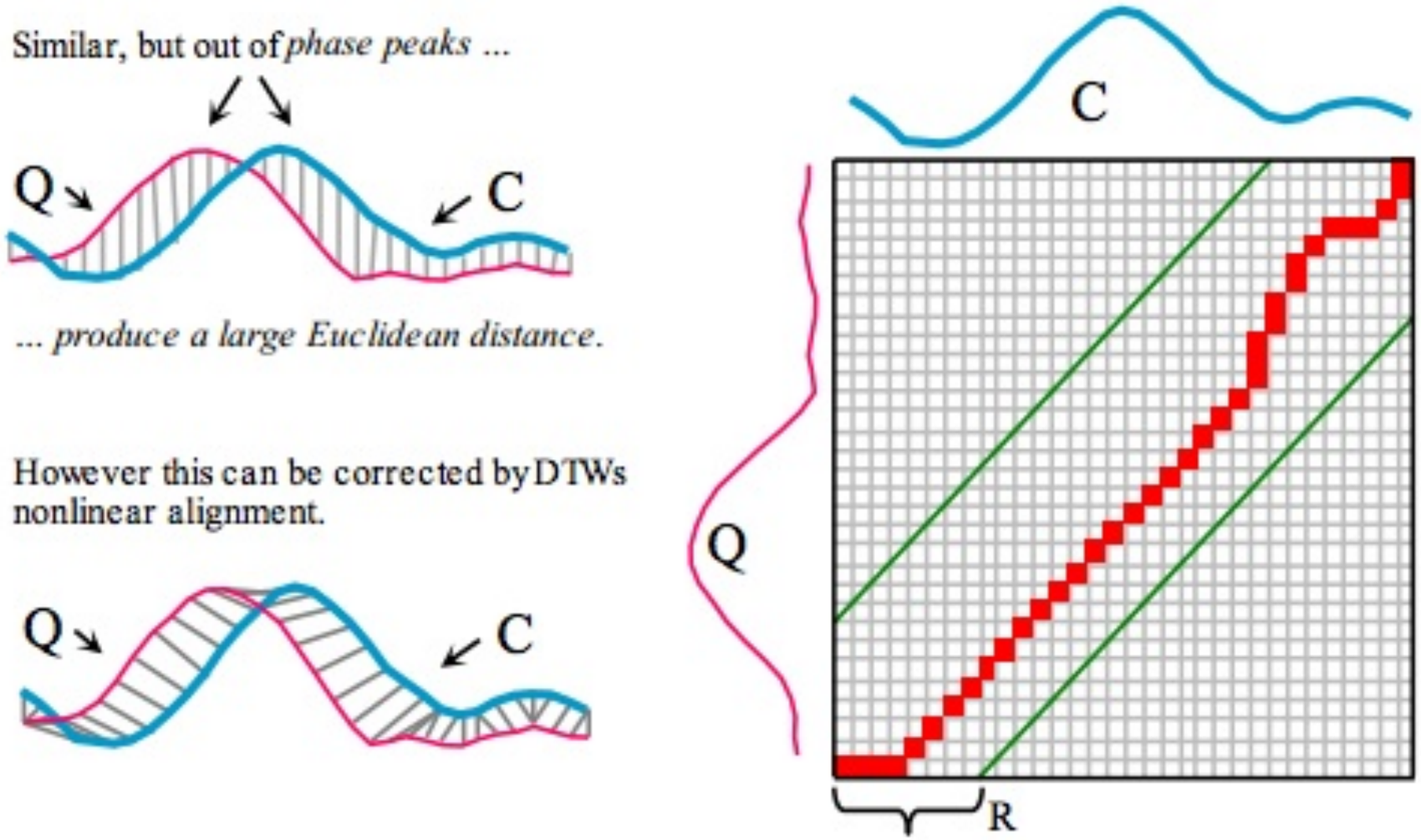


Figure 3: left) Two time series which are similar but out of phase. right) To align the sequences we construct a warping matrix, and search for the optimal warping path (red/solid squares). Note that Sakoe-Chiba Band with width R is used to constrain the warping path

The algorithm is used that searches the space of mappings from the time sequence of the input stream to that of the template stream, so that the total distance is minimized.

ALGORITHM

- There are an exponential number of paths (from one time series to the other) through the warping matrix.
- DTW is computed by identifying an *optimal* alignment path between two time series sequences. (The *optimal* path is found using dynamic programming subject to a few constraints)
- Let **A** and **B** be two given sequences with length M and N respectively.
- We can compare them by comparing their local distance for a specially chosen set of indices given by **m** for **A** and **n** for **B.** That is, the dynamic time warping distance will end up being the quantity:

$$C(A,B) = \sum_{k=0}^{M} d(A_{m_k}, B_{n_k})$$

- Of course, we should constrain the indices **m** and **n** such that the result is reasonable and computation is efficient.

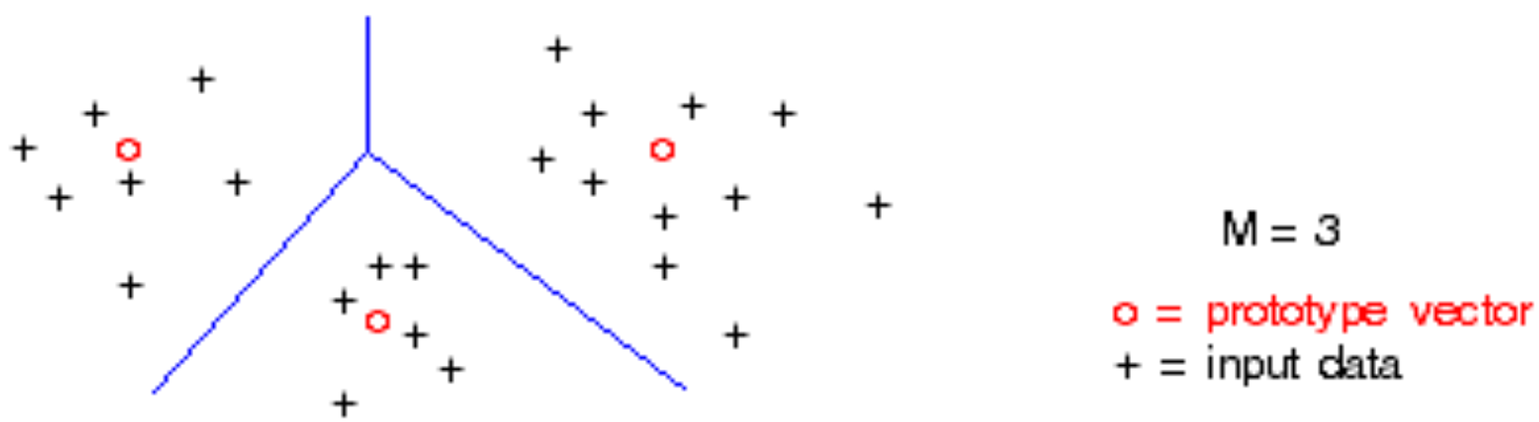
- **<u>Definition:</u>** A *warping path* for **A** and **B** is a pair of sequences (**m, n**) such that the following conditions:
  - $1 \leq m_k \leq M$ *and* $1 \leq n_k \leq N$ for all k
  - **Boundary Condition**:
    - The path starts at the bottom left and ends at the top right.
    - The sequences have endpoints ($\mathbf{m_1}$, $\mathbf{n_1}$) = (1, 1) and ($\mathbf{m_L}$, $\mathbf{n_L}$) = (M, N)
  - **Monotonic condition** *:*
    - The sequences ($\mathbf{m_k}$, $\mathbf{n_k}$) is monotonically increasing.
    - we cannot "step backward" in time as we compare local sequence entries
  - **Continuity Condition** :
    - One step at a time
    - ($\mathbf{m_k} - \mathbf{m_{k-1}}$, $\mathbf{n_k} - \mathbf{n_{k-1}}$) must be one of (1,0), (0,1) or (1, 1).
- There are many valid warping paths for any two sequences, but we need to pick one which has our desired feature. That is, it minimizes the sum of the local comparisons (equation in previous slide).
- Let us denote optimal value as DTW(**A, B**)

- Fourth Condition implies that to compute the optimal path requires a dynamic program. Specifically, the optimal path can be computed by solving the three sub-problems of finding the optimal warping path for:
  - $(A_{1\,..\,M-1}, B_{1..N})$
  - $(A_{1\,..\,M}, B_{1..N-1})$
  - $(A_{1\,..\,M-1}, B_{1..N-1})$
- Algorithm:
  - (w is window parameter to constraint the search indices)
  - DTW := array(M, N) of *Infinity*
  - DTW[0, 0] := 0
  - DTW[i, 0] := DTW[i-1, 0] + d(A[i], B[0]) for i = 1 to M
  - DTW[0, j] := DTW[0, j-1] + d(A[0], B[j]) for j = 1 to N
  - for i = 1 to M
    - for j = max(1, i - w) to min(1, i + w)
      - Cost := d(A[i], B[j])
      - DTW[i, j] := cost + min(DTW[i - 1, j], DTW[i, j - 1], DTW[i - 1, j - 1])
- Complexity = O(NM)

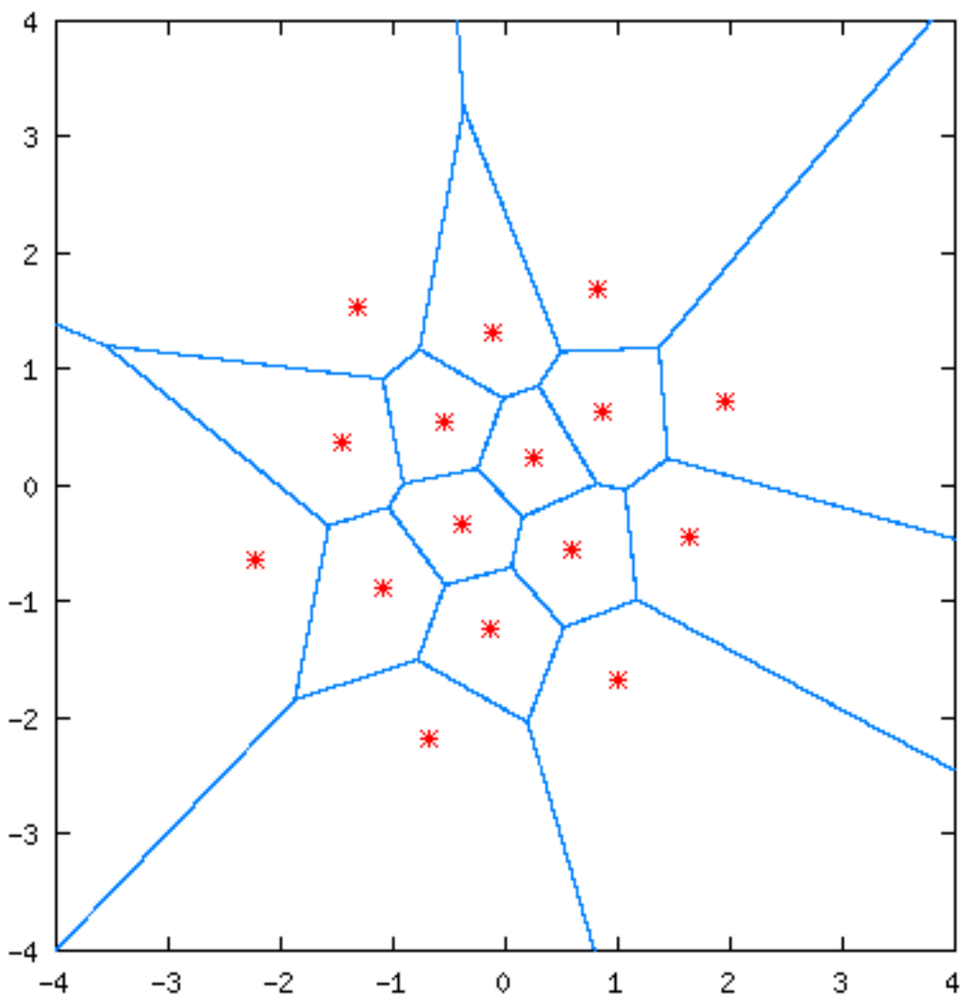
Vector Quantization and K Means Clustering

Vector Quantization (VQ)

- Unsupervised – Competitive learning method
- network "discovers" structure in the data by finding how the data is clustered
- Applications :
  - Data encoding
  - Data compression
- In vector quantization, we assume there is a codebook which is defined by a set of M prototype vectors. (M is chosen by the user and the initial prototype vectors are chosen arbitrarily).
- input belongs to cluster i if i is the index of the closest prototype



Example OF 2D VQ



- every pair of numbers falling in a particular region are approximated by a red star associated with that region
- Note that there are 16 regions and 16 red stars -- each of which can be uniquely represented by 4 bits
- the red stars are called ***codevectors*** *or* ***prototype vectors***
- the regions defined by the blue borders are called ***encoding regions***
- The set of all codevectors is called the ***codebook***
- the set of all encoding regions is called the ***partition of the space***

VQ and Data Compression

- Vector quantization can be used for (lossy) data compression. If we are sending information over a phone line, we
  - initially send the codebook vectors
  - for each input, we send the index of the class that the input belongs
- For a large amount of data, this can be a significant reduction. If M=64, then it takes only 6 bits to encode the index. If the data itself consists of floating point numbers (4 bytes) there is an 80% reduction
  - (100*(1 - 6/32)) = 80% reduction

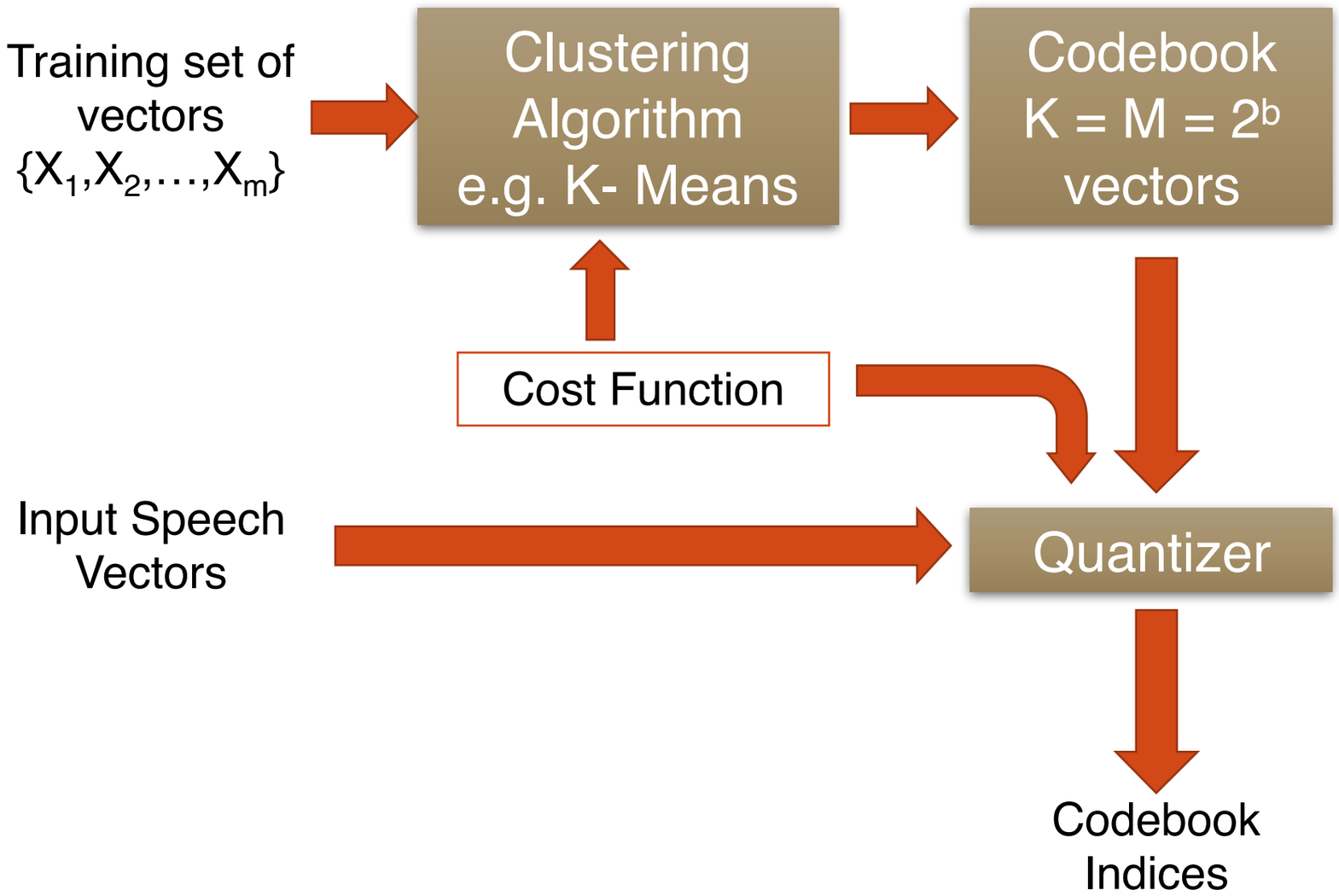
VQ for Speech Signals

Advantages

+ Reduced storage
+ Reduced computation for determining spectral similarity of speech signals
+ Discrete representation of speech sounds – associate a phonetic class to each prototype vector

Disadvantages

- Quantization error or Distortion (assigning a prototype vector is equivalent to quantization), it can be reduced by choosing large value for M
- The larger we choose M, the more storage is required for codebook entries – tradeoff is needed among Quantization error, processing needed to choose codewords and storage of codebook



VQ Implementation

- The training set is used to construct an optimal set of codebook vectors for representing spectral variability in training set
- We construct :
- $b = \text{ceil}\ (\log_2 M\ ))$ bit codebook

K Means Clustering

- Clustering:
  - Form of unsupervised learning whereby a set of observations is partitioned into natural groupings or clusters of patterns in such a way that the measure of similarity between any pair of observations assigned to each cluster minimizes a specified cost function. – Simon Hykin
- K-Means : simple to implement yet effective in performance
  - partition **m** observations into **k** clusters in which each observation belongs to the cluster with the nearest mean
  - Every learning problem is an optimization problem – We need to search for values of parameters in parameter space that minimizes some defined cost function
  - For K-means, its Euclidean distance

ALGORITHM

- Input:
  - K : # of Clusters
  - Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $x^{(i)} \in \mathbb{R}^n$
- Goal:
  - Find the encoder ***C*** that assigns observations in training set to the ***K*** clusters in such a way that, within each cluster, the average measure of dissimilarity of the assigned observations from the cluster centroid/mean is minimized.
  - *i.e.*, the following cost function is minimized:

$$J(C) = \sum_{k=1}^{K} \sum_{C(i)=k} \|x^{(i)} - \mu_k\|^2$$

- Minimization can be carried out by an iterative descent algorithm with 2 step optimization
  - 1st :Minimize the inner summation wrt to encoder C for a given man vector μ_k. This is called as cluster assignment step.
  - 2nd: Use nearest neighbor rule to minimize $J(C)$ wrt to mean vector μ_k for a given encoder C. This is called move centroid step.

Algorithm contd…

- Procedure:
  - Randomly initialize ***K*** cluster centroids, $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$
    - i.e. Randomly pick ***K*** training examples, and set centroids to these examples.
  - Repeat till convergence:
    - cluster assignment step:
      - for a given set of cluster means $\{\mu_k\}$ *for k = 1 to K,* optimize Encoder ***C*** as:

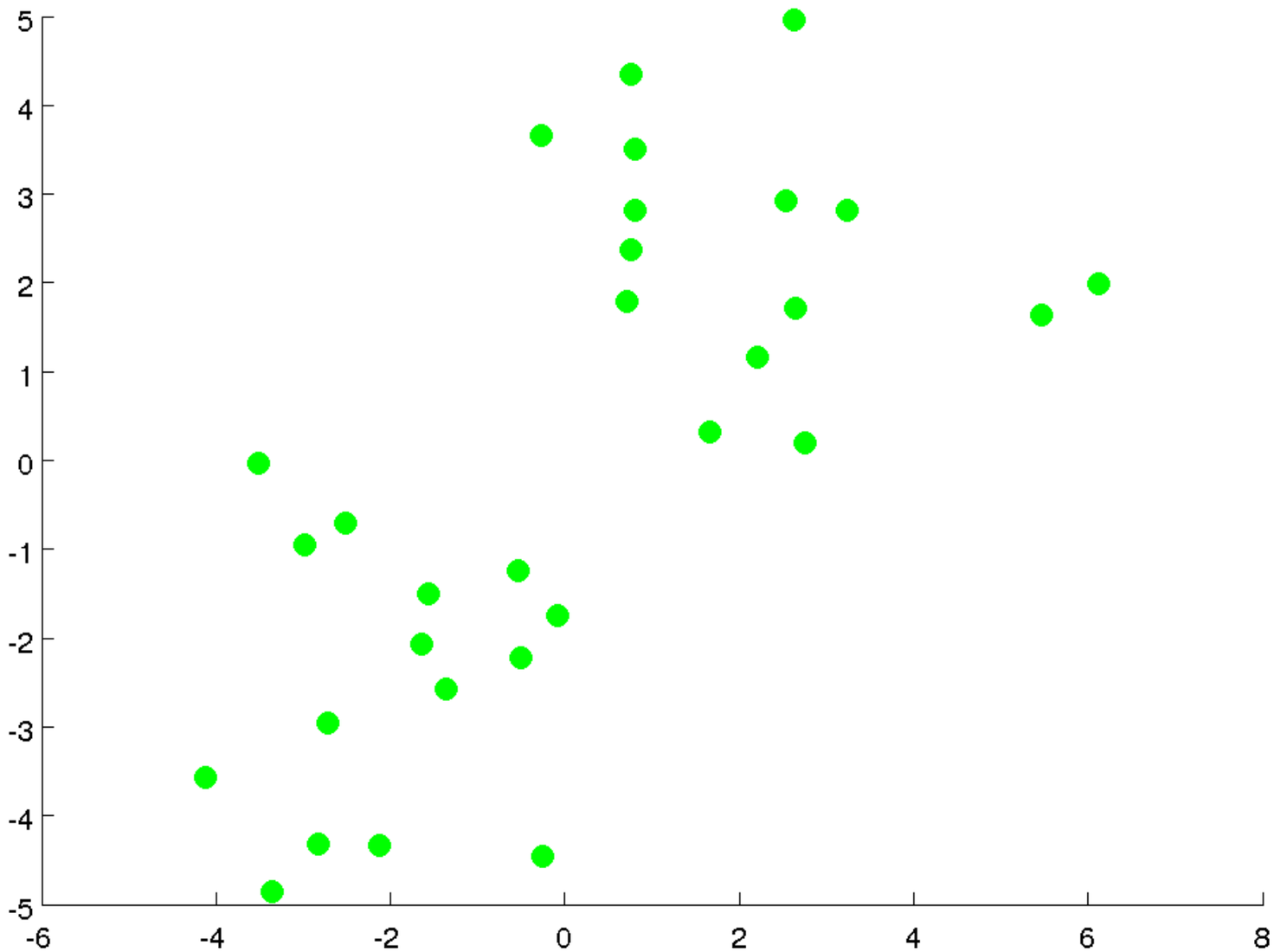
$$C(i) = k : \min_k \|x^{(i)} - \mu_k\|^2$$

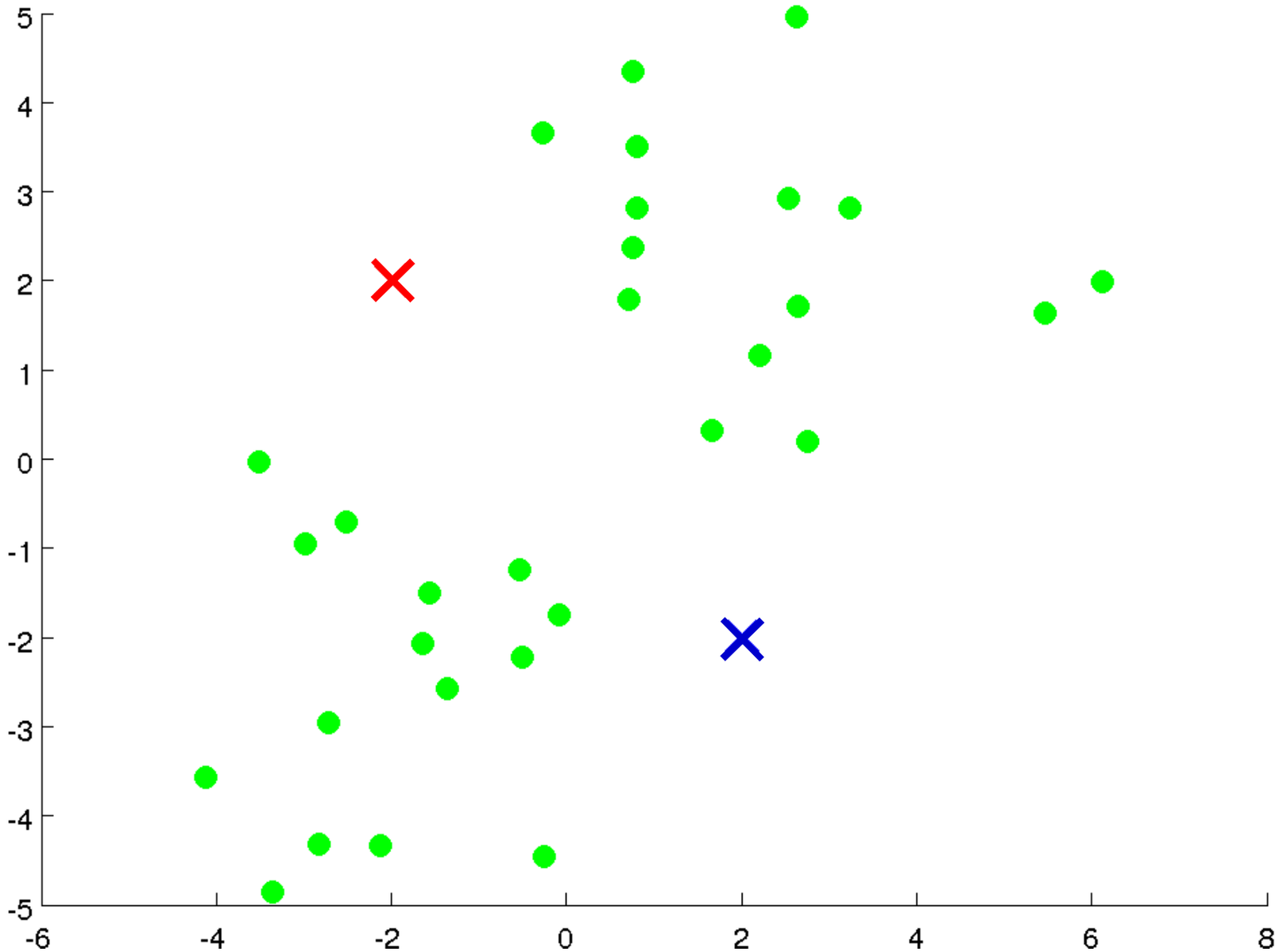
    - move centroid step:
      - For a given Encoder ***C*** , the total cluster variance is minimized w.r.t set of cluster centroids $\{\mu_k\}$ *for k = 1 to K* as:

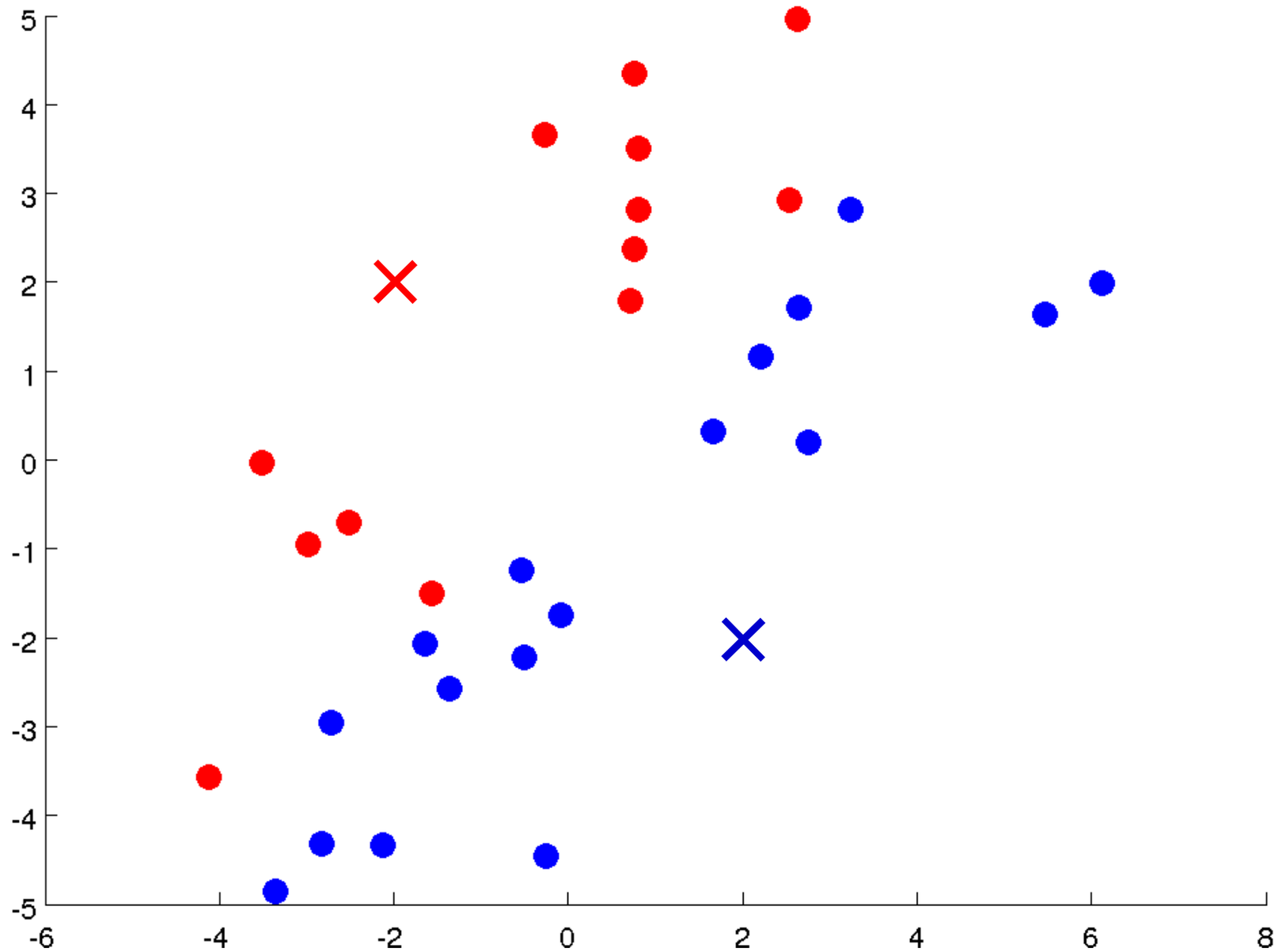
$$\min_{\mu_k} J(C) \text{ , for } k = 1 \text{ to } K$$

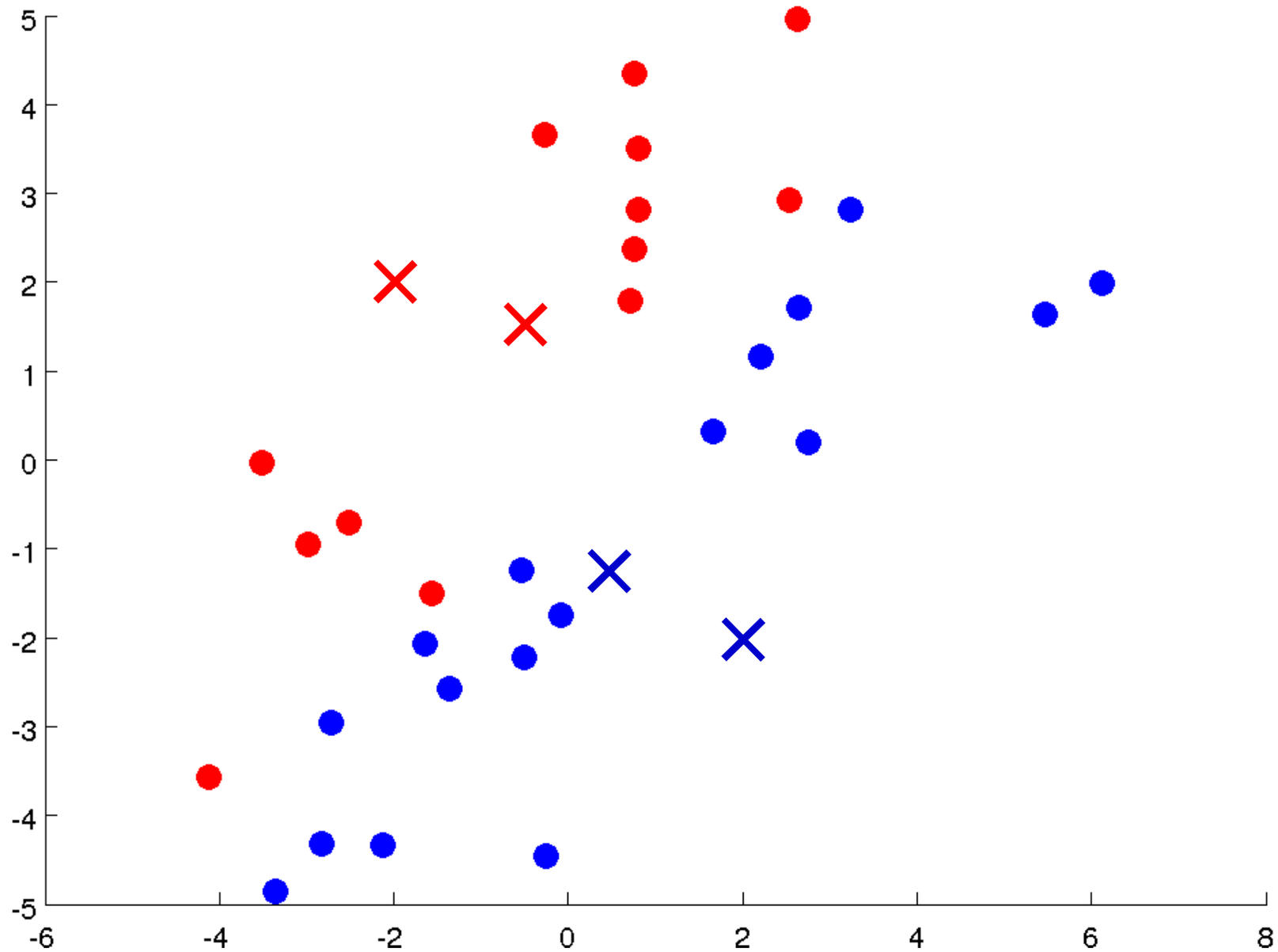
i.e. new centroids are computed for each cluster by taking average of datapoints assigned to that cluster in previous step

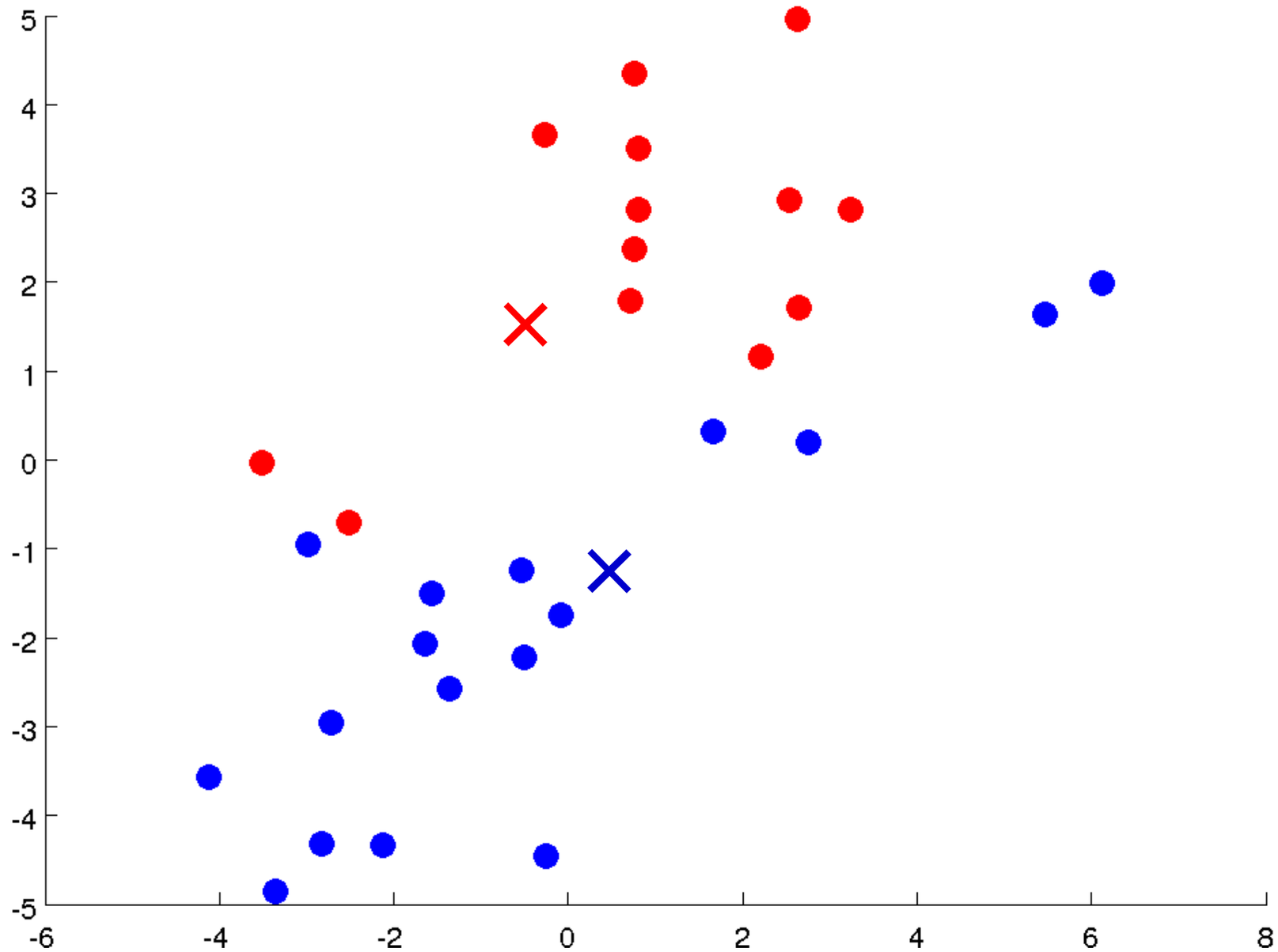
How it works?? (From Andrew Ng's lectures)

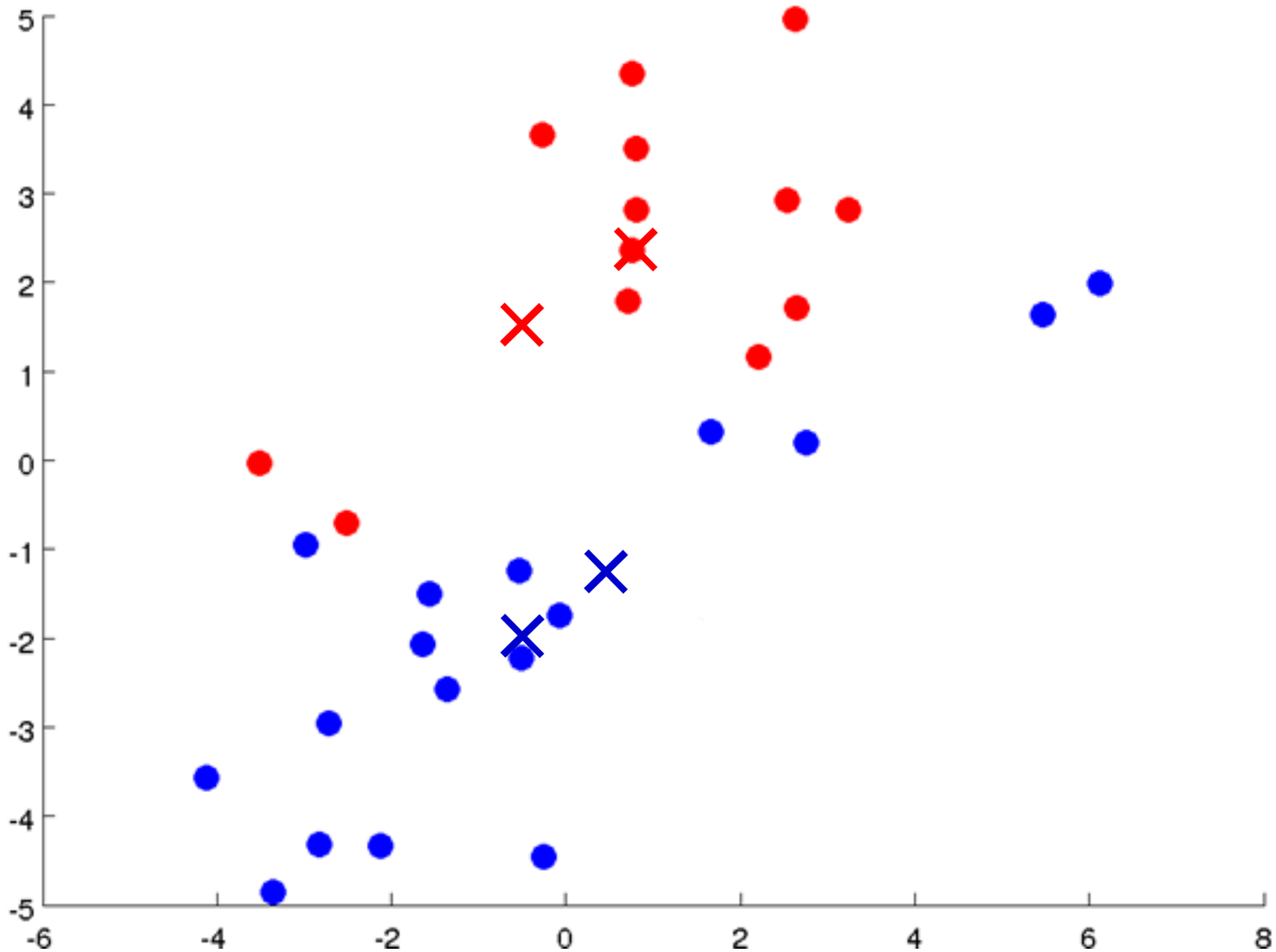


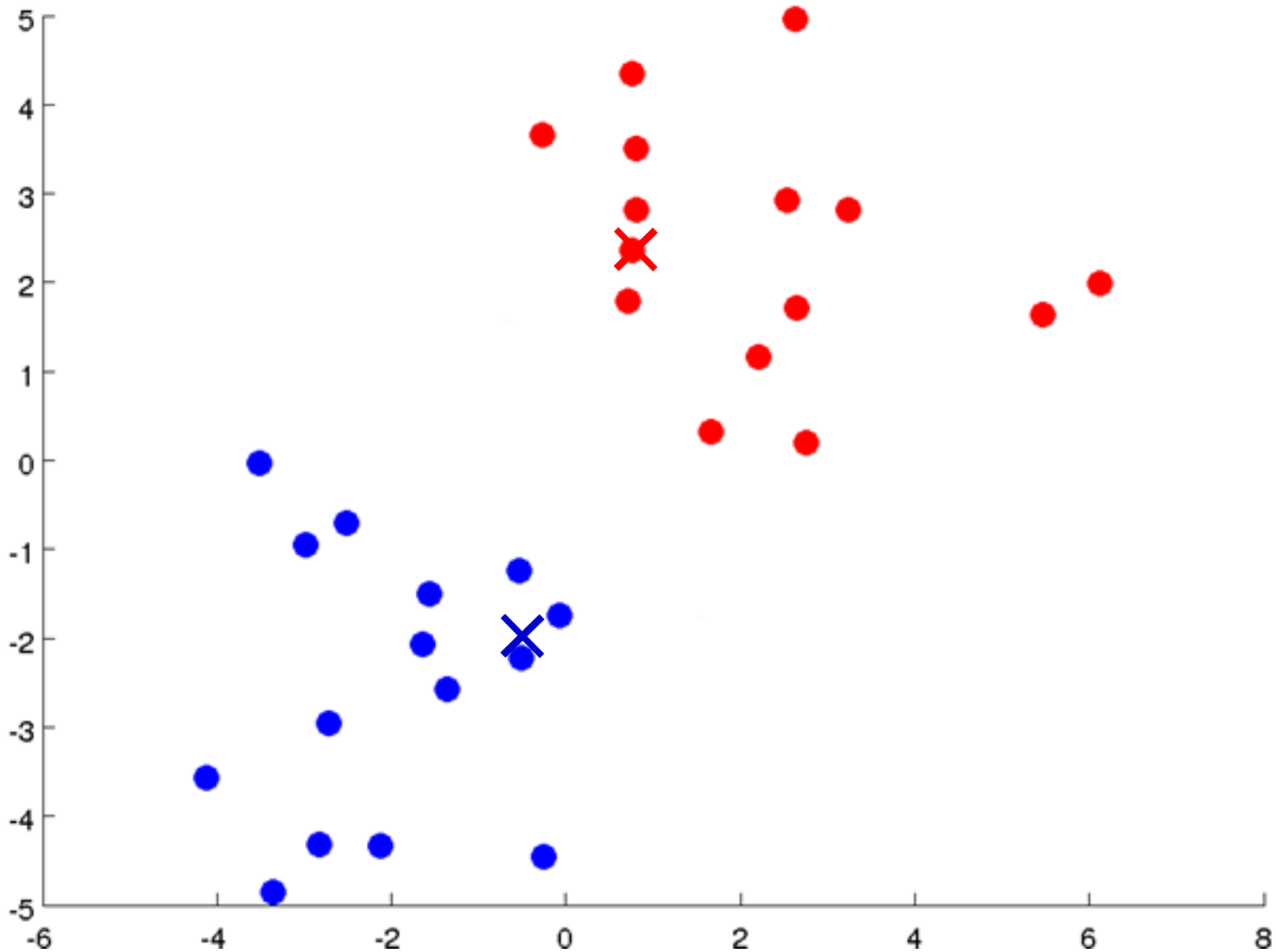

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

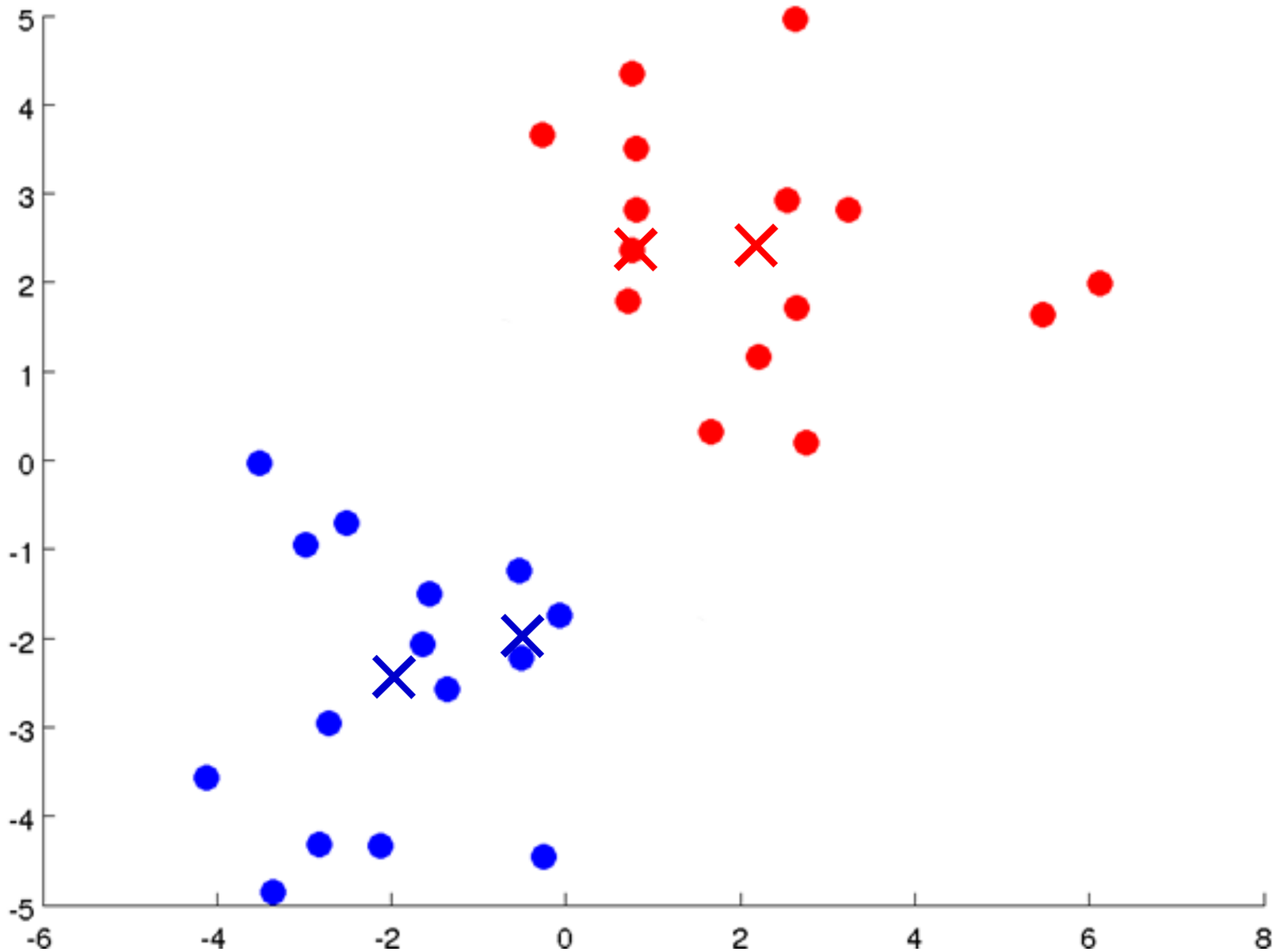

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

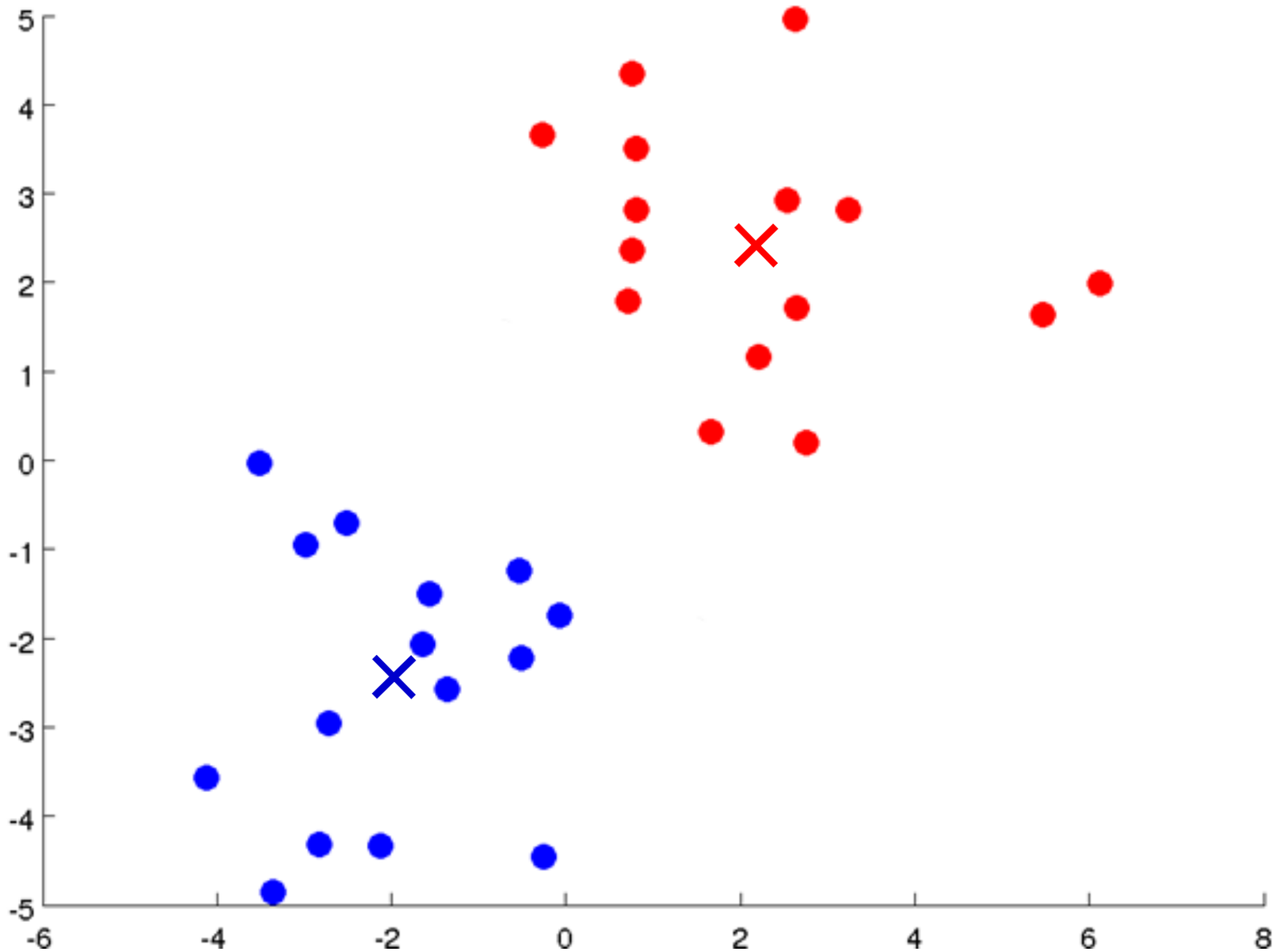

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

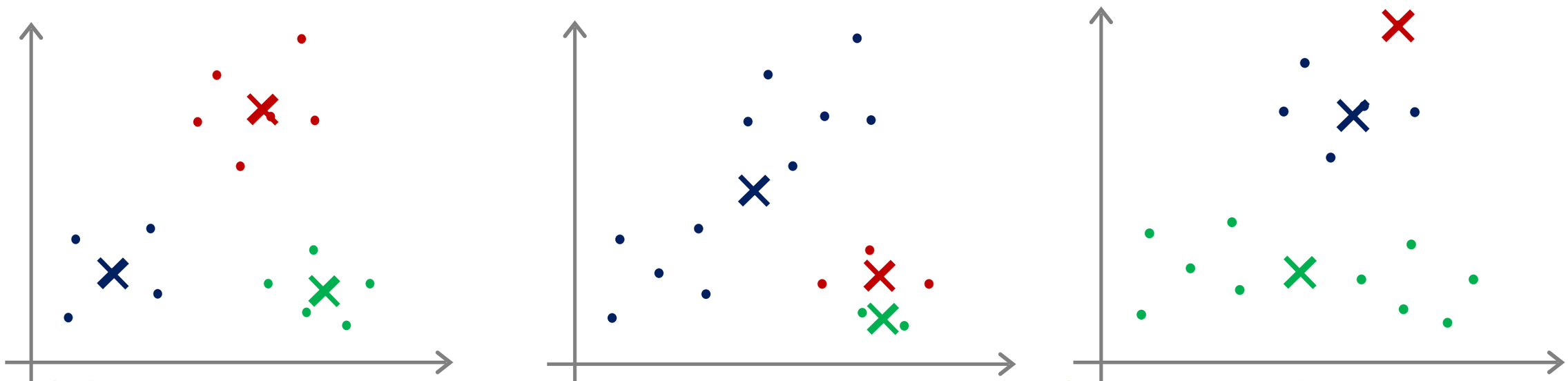

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

Suboptimal Solution

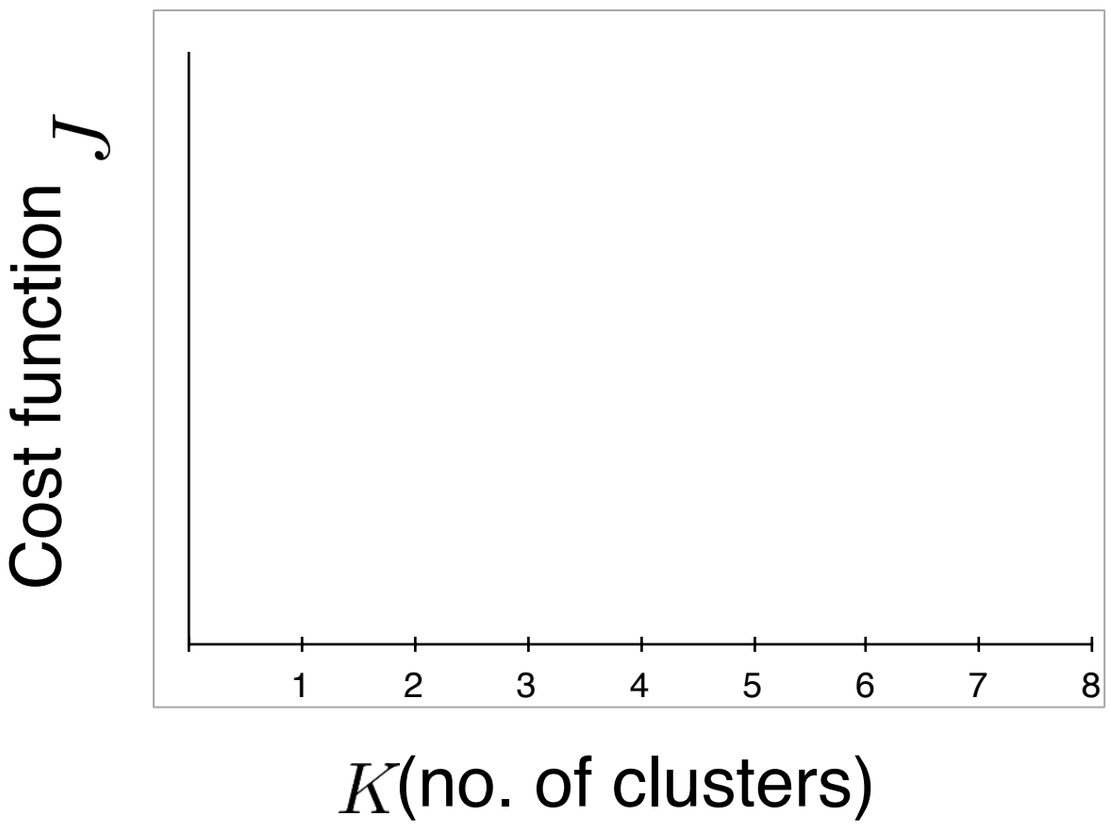
- Starting from the initial centroids, the algorithm goes back and forth between the two steps until there is no further change in cluster assignments
- Each of the two steps is designed to reduce J(C), hence convergence is assured.
- However, due to lack of global optimality criterion, result may be suboptimal solution(convergence to local optimum)

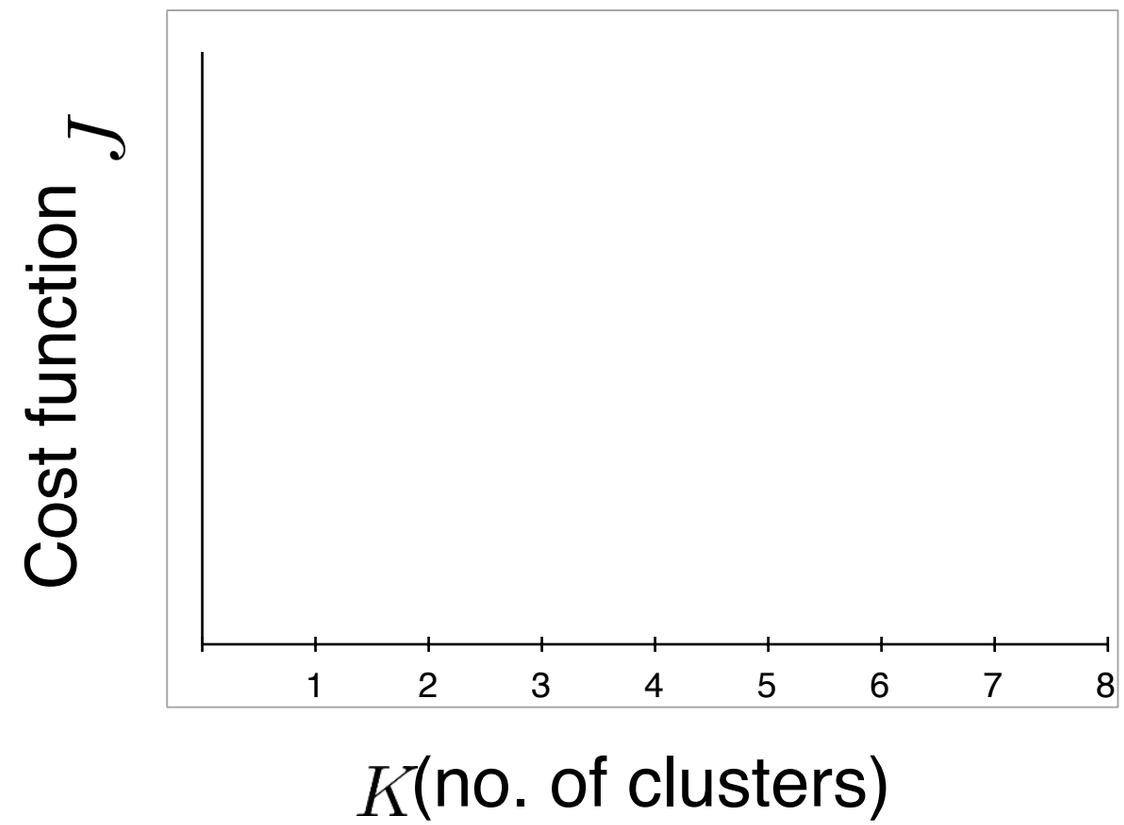


- Solution: Run k-means n times (say n = 100), keep track of *J(C)* at the end of each run, and our solution is the one which has lowest *J(C)*

Choosing the value of K (from Andrew Ng's)

Elbow method:

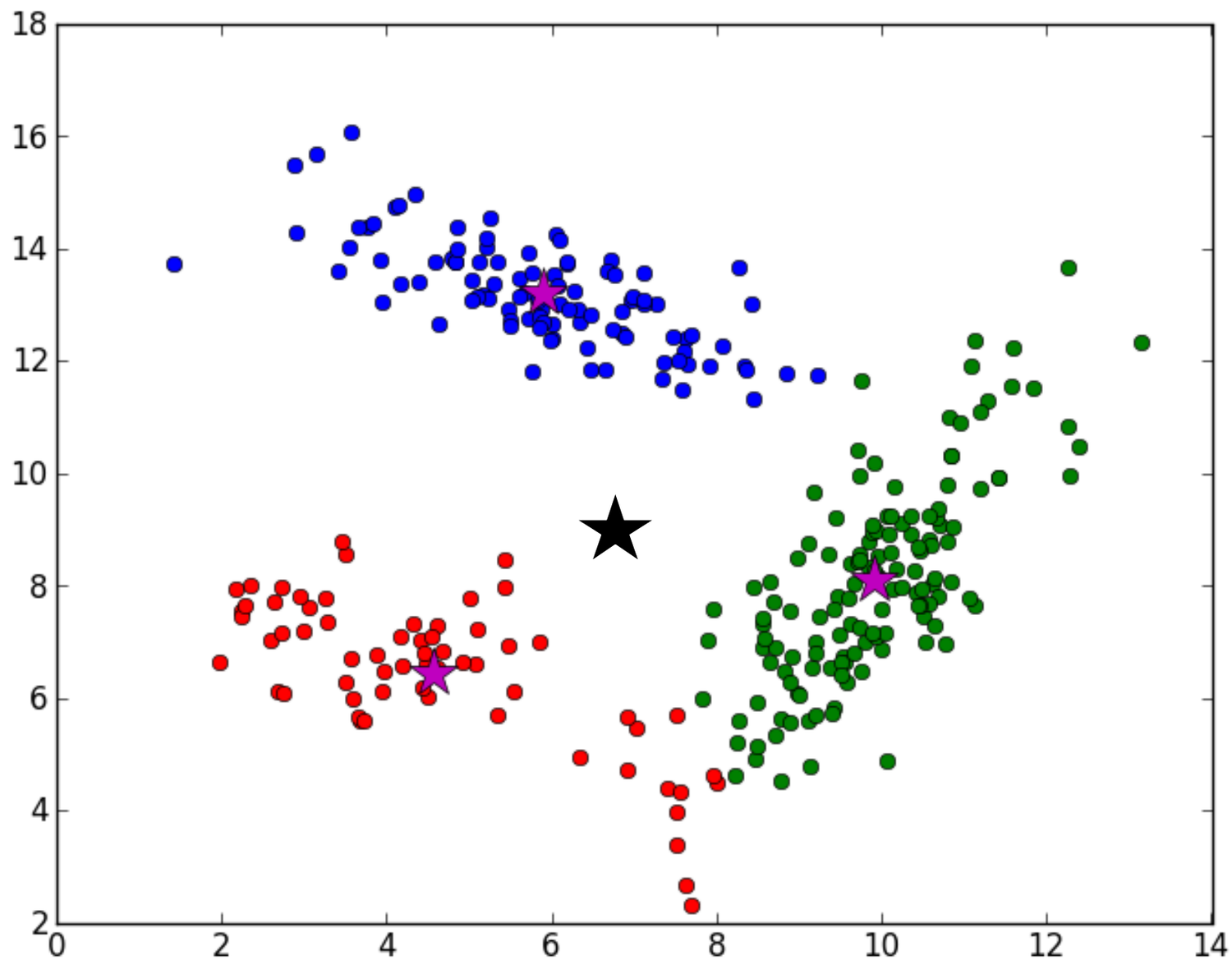


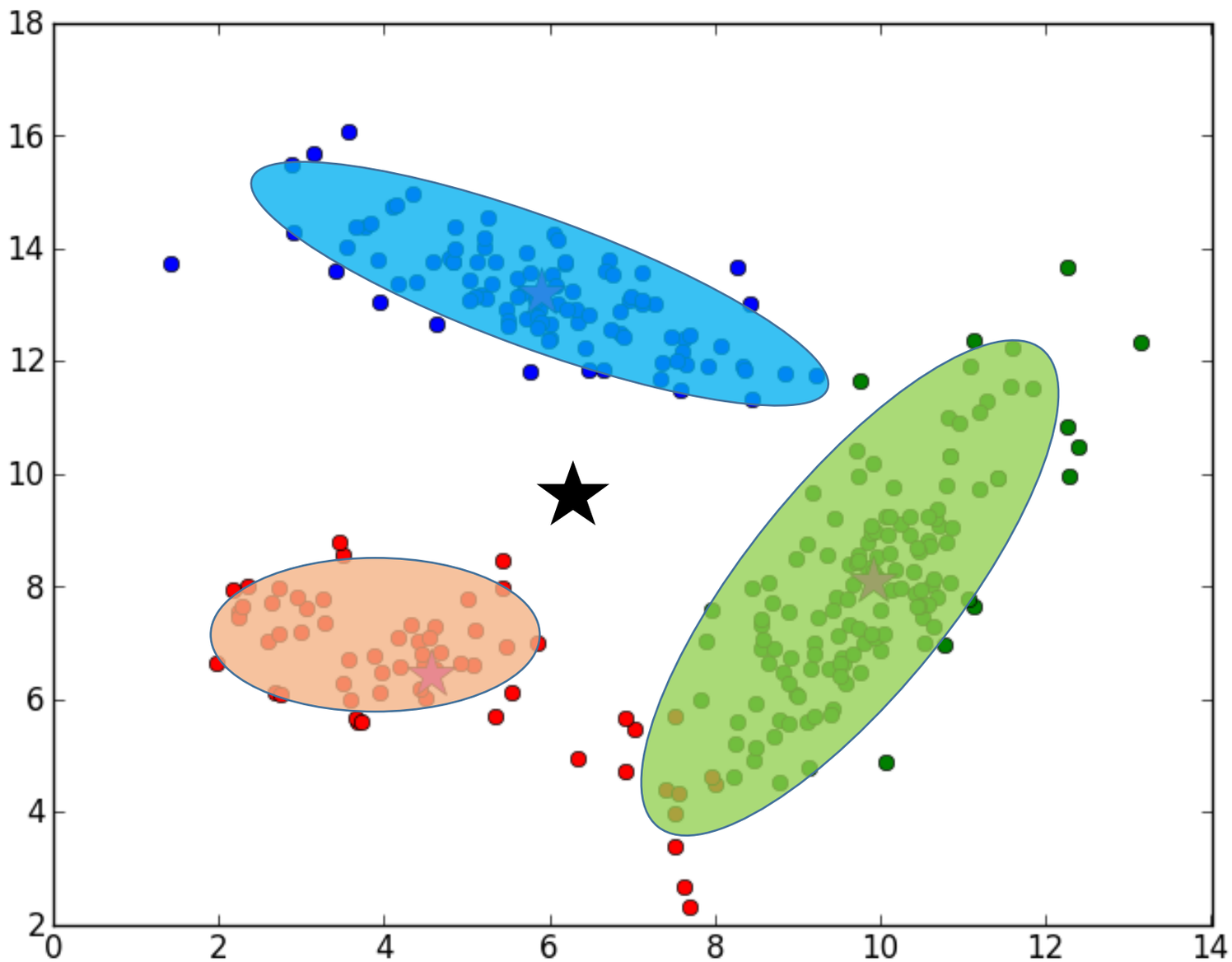


Gaussian Mixture Modeling (GMM)

K Means Problem

- If clusters are nearer and skewed/ skinny ovals
- What if test point if far from all clusters??- kMeans – Only one assignment





GMM

- Note 3 clusters that can be modeled as *Gaussian* distribution with some (μ, Σ)
- For a test point, stochastic membership to a class makes more sense.

Mixture Model and Density Estimation

Mixture Model : Probabilistic model for representing a presence of sub-populations within the population.

Mixture Distribution : weighted sum of probabilistic distributions to form overall population

The density estimation problem : Given a set of *m* points in *n* dimensions, $x^{(1)}, \ldots, x^{(m)}$ 安 R^n, and a family *F* of probability density functions p(x) on R^n, find the probability density *p(x)* 安 *F* that is most likely to have generated the given points.

- **GMM** : family *F* is mixture of *Gaussians* functions.

Unsupervised Learning: Clustering approach used to estimate distribution parameters (e.g. means and covariances in GMM + weighting parameter)

- A *n* multivariate Gaussian Distribution is:

$$p(x \mid \boldsymbol{\mu}, \sum) = \frac{1}{\sqrt{2\pi^n |\sum|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \sum^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$

  - Where μ is mean vector, $\sum$ is covariance matrix.
- The mixture of Gaussian then is:

$$p(x) = \sum_{k=1}^{K} p(x \mid \mu_k, \sum_k).p(k)$$

  - Where p(k) is the weight for cluster k : probability that a randomly picket data point belongs to cluster k.

EM Algorithm

- an expectation–maximization (EM) algorithm is an iterative method for finding maximum likelihood estimates of parameters in statistical models.
- Iteratively estimates parameters in 2 step:
  - E Step : Use old estimates to assign responsibility(membership of each data point to a cluster)
  - M Step: Use responsibilities to compute parameters
- In K means, each data-point is assigned to only one cluster. But in GMM, each data point is assigned to multiple clusters with some probability of membership. This is called responsibility and is represented by Responsibility matrix.
- **<u>Problem</u>**: Given a set of training datapoints: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $x^{(i)} \in \mathbb{R}^n$

  Find a set of K multivariate (n –variables), whose sum represent these samples

  note, K is provided.

Expectation Step

•• Let us assume data points are independent

$$L = \prod_i p(x^{(i)}), \text{ likelihood}$$

$$where\ p\left(x^{(i)}\right) = \sum_{k=1}^{K} p\left(x^{(i)} \mid \mu_k, \boldsymbol{\Sigma}_k\right).p(k)$$

- **<u>Likelihood</u>** : measure of how good data fits a given model
- Log Likelihood for our GMM:

$$\log p(X \mid \theta) = \sum_{i=1}^{m} log \sum_{k=1}^{K} p\left(\boldsymbol{x}^{(i)} \mid \mu_k, \boldsymbol{\Sigma}_k\right)$$

  - Where θ is the parameters we are searching for. It consists of mean, covariance and weights (or probabilities) for each cluster k.

- Responsibility Matrix : is a m * K dim matrix, whose element R_{ik} denotes the probability of membership of ith training sample(datapoint) in cluster k
  - $R_{ik} = p(k \mid x^{(i)}) = \frac{p(x^{(i)} \mid k)p(k)}{p(x^{(i)})} = p(x^{(i)} \mid k)p(k) = p(x^{(i)} \mid \mu_k, \boldsymbol{\Sigma}_k)p(k)$
  - Then normalize R_{ik} so that

$$\sum_k R_{ik} = 1, \text{ for every i}$$

Maximization Step

- • Estimate parameters from prob. of each point $x^{(i)}$belonging to a cluster k.

$$\mu_k = \frac{1}{m * p(k)} \sum_{i=1}^{m} R_{ik}\, x^{(i)}$$

$$\sum_k = \frac{1}{m*p(k)} \sum_{i=1}^{m} R_{ik}\ (\boldsymbol{x}^{(i)} - \boldsymbol{\mu_k})\ (\boldsymbol{x}^{(i)} - \boldsymbol{\mu_k})^T$$

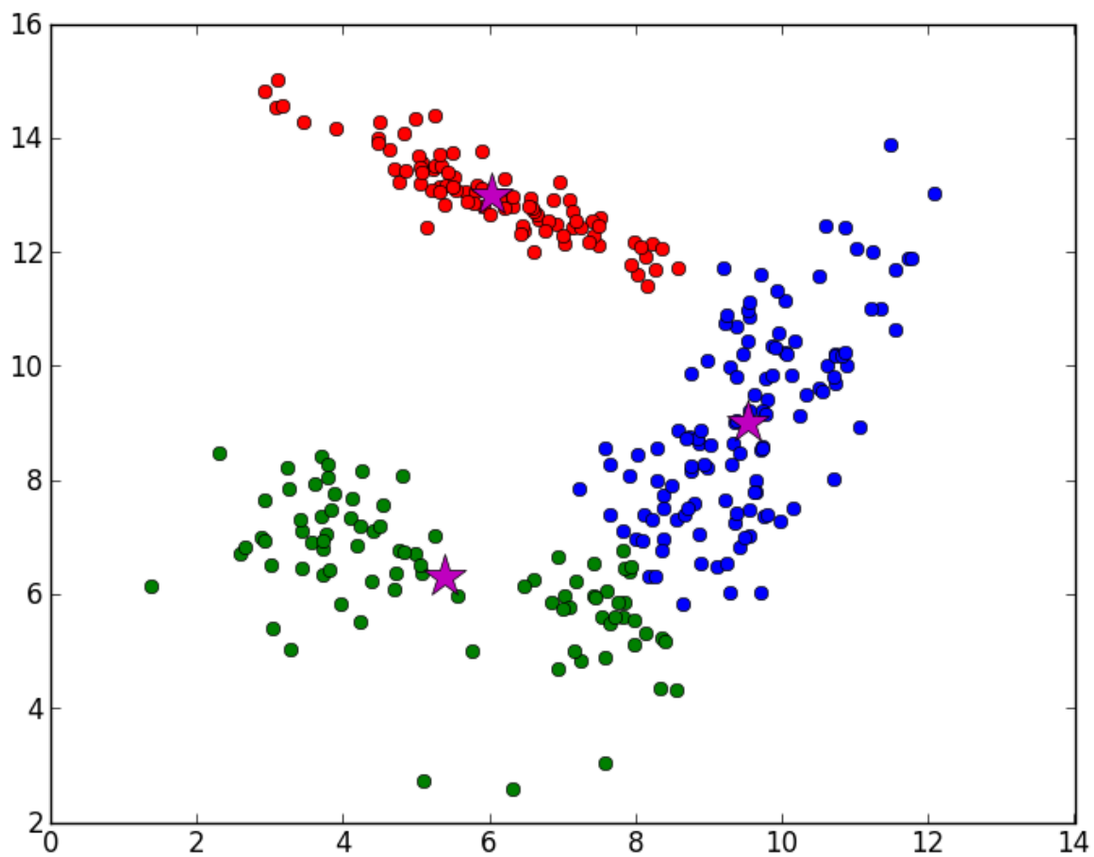
$$p(k) = \frac{1}{m} \sum_{i=1}^{m} R_{ik}$$

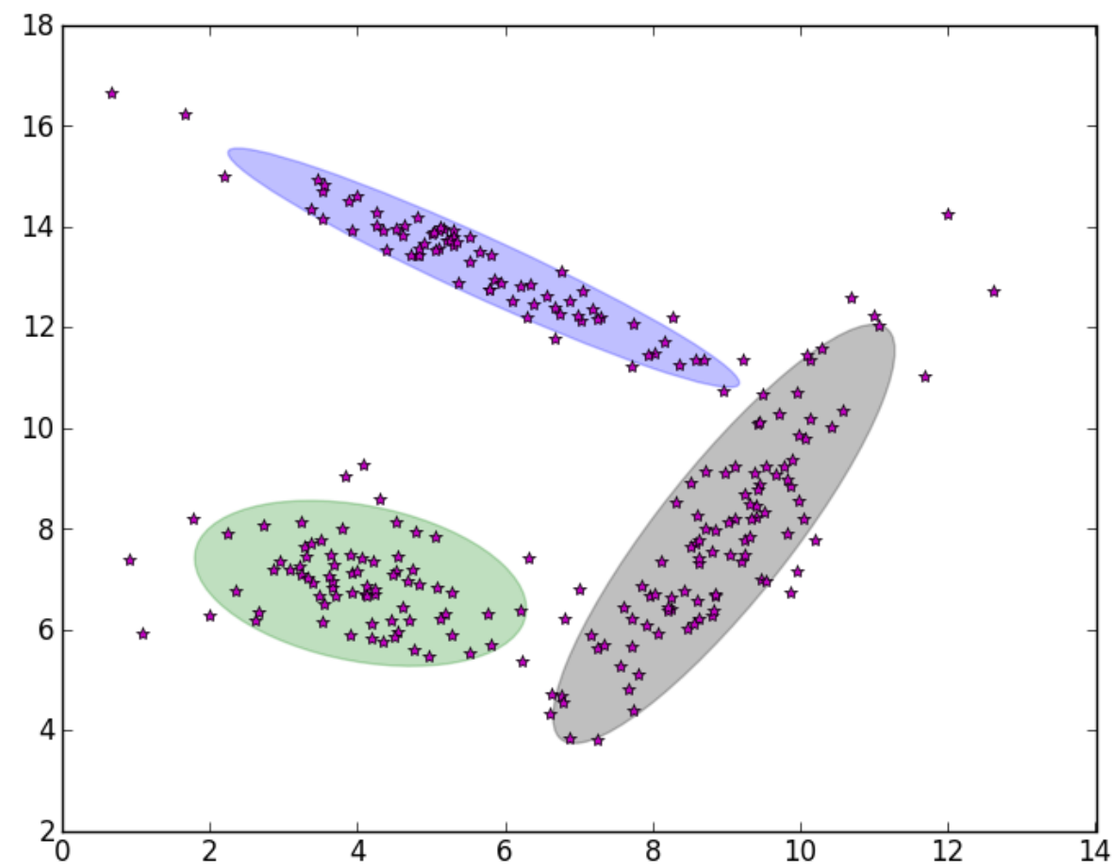
- i.e. compute parameters that maximize our objective function (log likelihood).

Algorithm

-
  - Initialize {$\boldsymbol{\mu_k}$, $\sum_k$ and *p(k) for k = 1 to K, p(k)>0* and $\sum_k p(k) = 1$ }
  - Until convergence do:
    - <u>E step</u>:
      - For each i in 1..m
        - For each k in 1..K
          - Compute R_{ik}
        - normalizeR_{ik}so that $\sum_k R_{ik} = 1$
    - <u>M step</u>:
      - For every k in 1..K compute
        - μ_k , $\sum_k$ and p(k)
    - Compute Log-Likelihood L, stop if L - L_{old} < ε
  - Get implementation codes of section 5.1 https://gist.github.com/bistaumanga

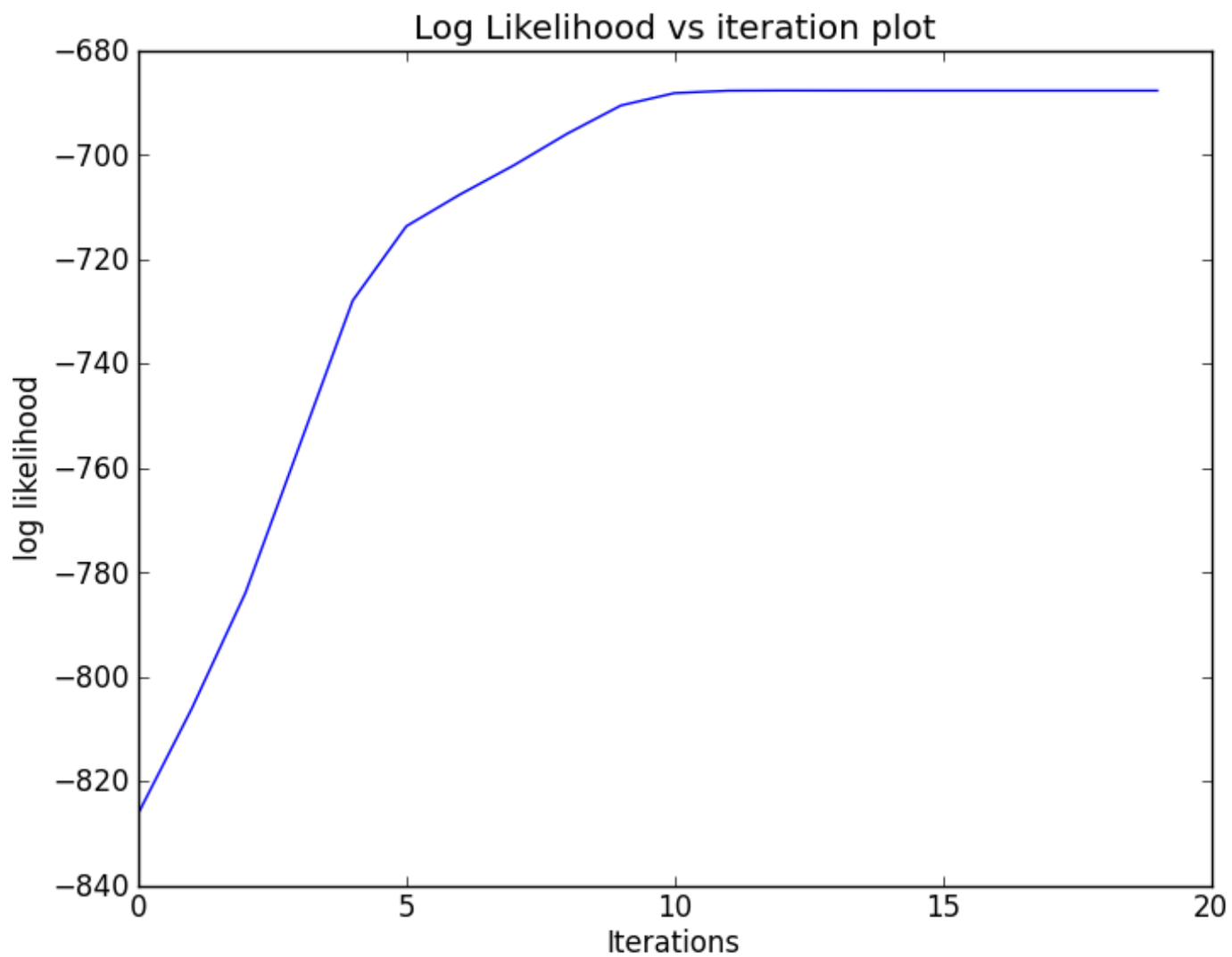
K Means VS GMM





Log Likelihood Plot

- Log Likelihood is maximized in M-step



Hidden Markov models

- HIDDEN MARKOV MODEL (HMM) HMM is the extension of Markov Process
- Markov Process consist of observable states
- HMM has hidden states and observable symbols per states
- HMM is the stochastic model

Hidden Markov models

- Example(Reference: Jivan)
- A person is only staying at his room.
- He receives food daily from another X person.
- He has to predict weather (Sunny, rainy, winter e.t.c) which are hidden states.
- For him the states or observation based upon which he can predict can be
  - If X is wearing shorts
  - If X carries umbrella
  - If X is wearing warm clothes

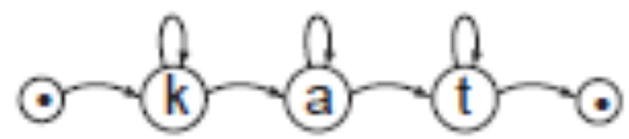
Hidden Markov models (2)

- **HMM is specified by:**

- states q^i

- transition probabilities a_{ij}

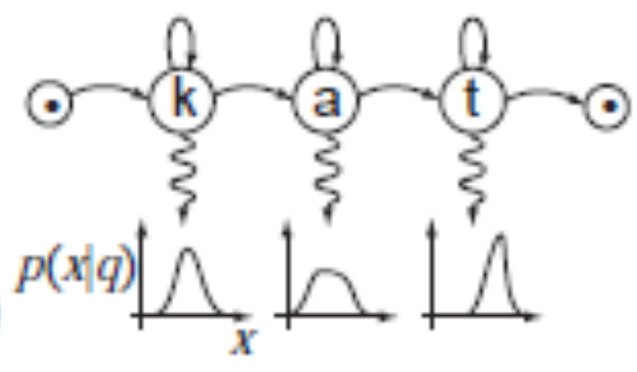
$$p(q_n^j | q_{n-1}^i) \equiv a_{ij}$$



	k	a	t	•
•	1.0	0.0	0.0	0.0
k	0.9	0.1	0.0	0.0
a	0.0	0.9	0.1	0.0
t	0.0	0.0	0.9	0.1

- emission distributions $b_i(x)$

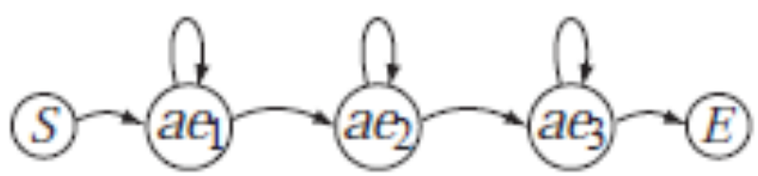
$$p(x | q^i) \equiv b_i(x)$$

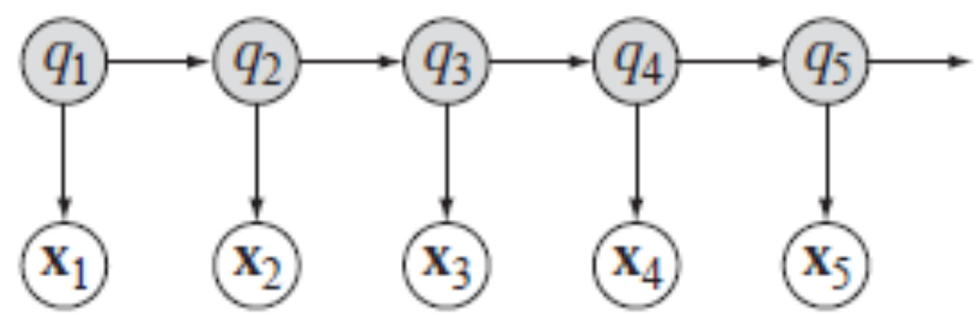


+ (initial state probabilities $p(q_1^i) \equiv \pi_i$)

Markov models for speech

- **Speech models** M_j
  - usually left-to-right HMMs (sequence constraint)
  - observation & evolution
    are conditionally independent of rest
    given (hidden) state q_n

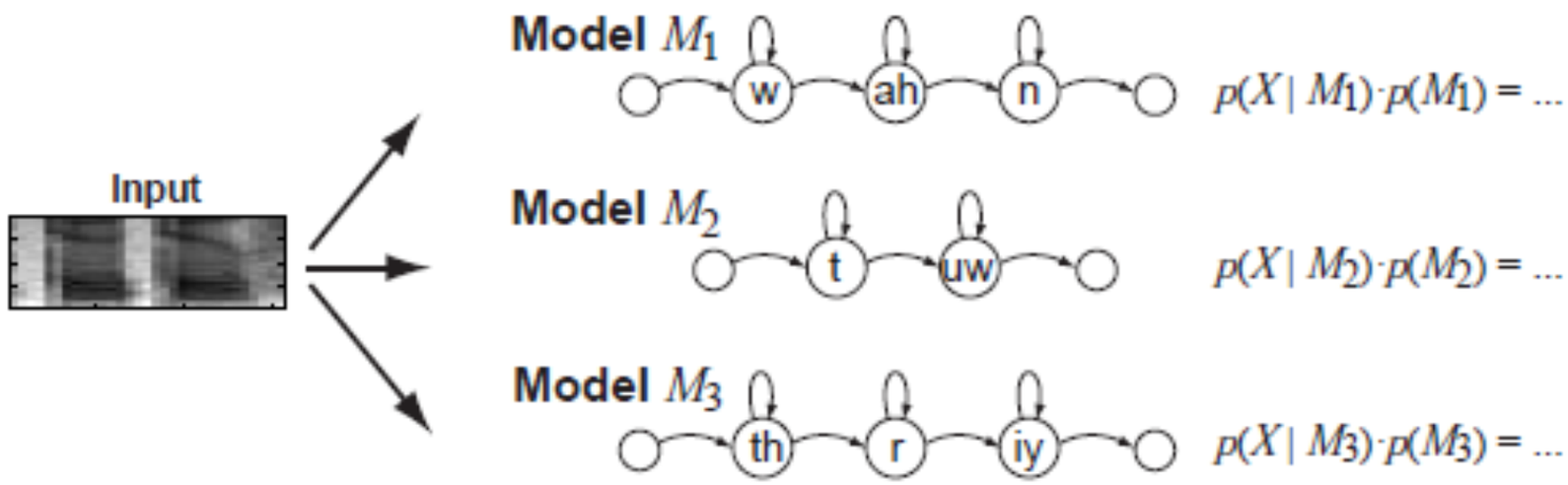




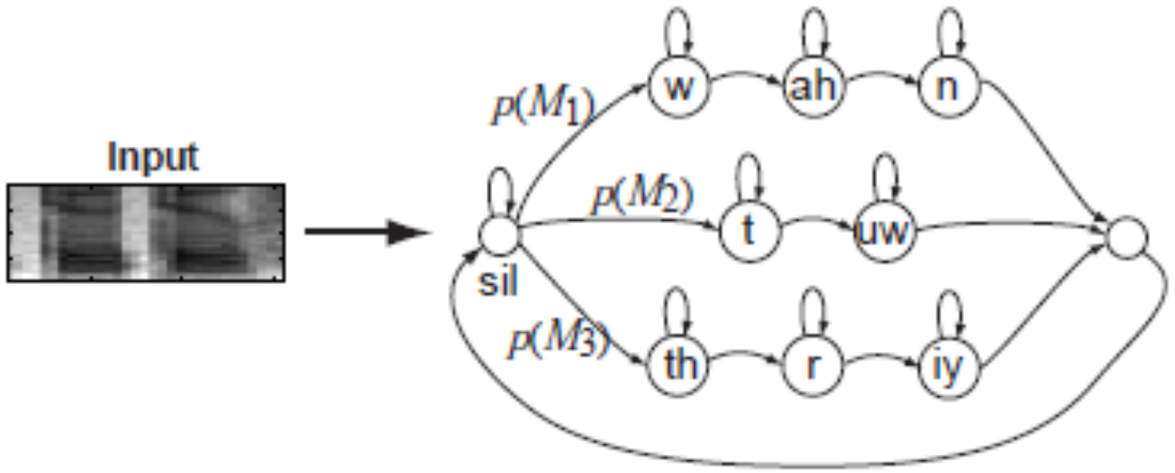
  - self-loops for time dilation

Recognition with HMMs

- Isolated word
  - choose best $p(M|X) \propto p(X|M)p(M)$



- Continuous speech
  - Viterbi decoding of one large HMM gives words



Speech Processing

5.2 Automatic Speech Recognition
5.3 Speaker identification and verification

Deepesh Lekhak
(066/BEX/414)

Automatic Speech Recognition

- ASR is the translation of spoken words into text.
- Goal is to generate the optimal word sequence subject to linguistic constraints.
- Identify the words regardless of the speaker

(Speaker independent but text dependent recognition)

5.2.1 Feature Extraction for ASR

- Feature extraction is the process of retaining useful information of the signal while discarding redundant and unwanted information.
- The first step in almost every practical pattern recognition (PR) application.
- objectives:
  - High discrimination between sub-word classes.
  - Low Speaker variability.
  - Invariance to degradations in the speech signal due to channel and noise.

Feature Extraction for ASR (2)

- Broadly the feature extraction techniques are classified as :
  - temporal analysis:
    - speech waveform itself is used for analysis
  - spectral analysis
    - spectral representation of speech signal is used for analysis

Temporal Analysis

- processing of the waveform of speech signal directly

1. Power Estimation

It is computed on frame by frame basis as:

$$P(n) = (1/N_s) \sum_{m=0}^{N_s-1} \left(w(m)\, s(n - N_s/2 + m) \right)$$

Where Ns is the number of samples used to compute the power, *s(n) denotes the signal, w(m) denotes the* window function, *n denotes the sample index of center of the window.*

Hamming window is exclusively used in speech recognition

$$w[n] = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad \text{, if n=0,1,2,...,N-1}$$

$$=0 \quad \text{, otherwise}$$

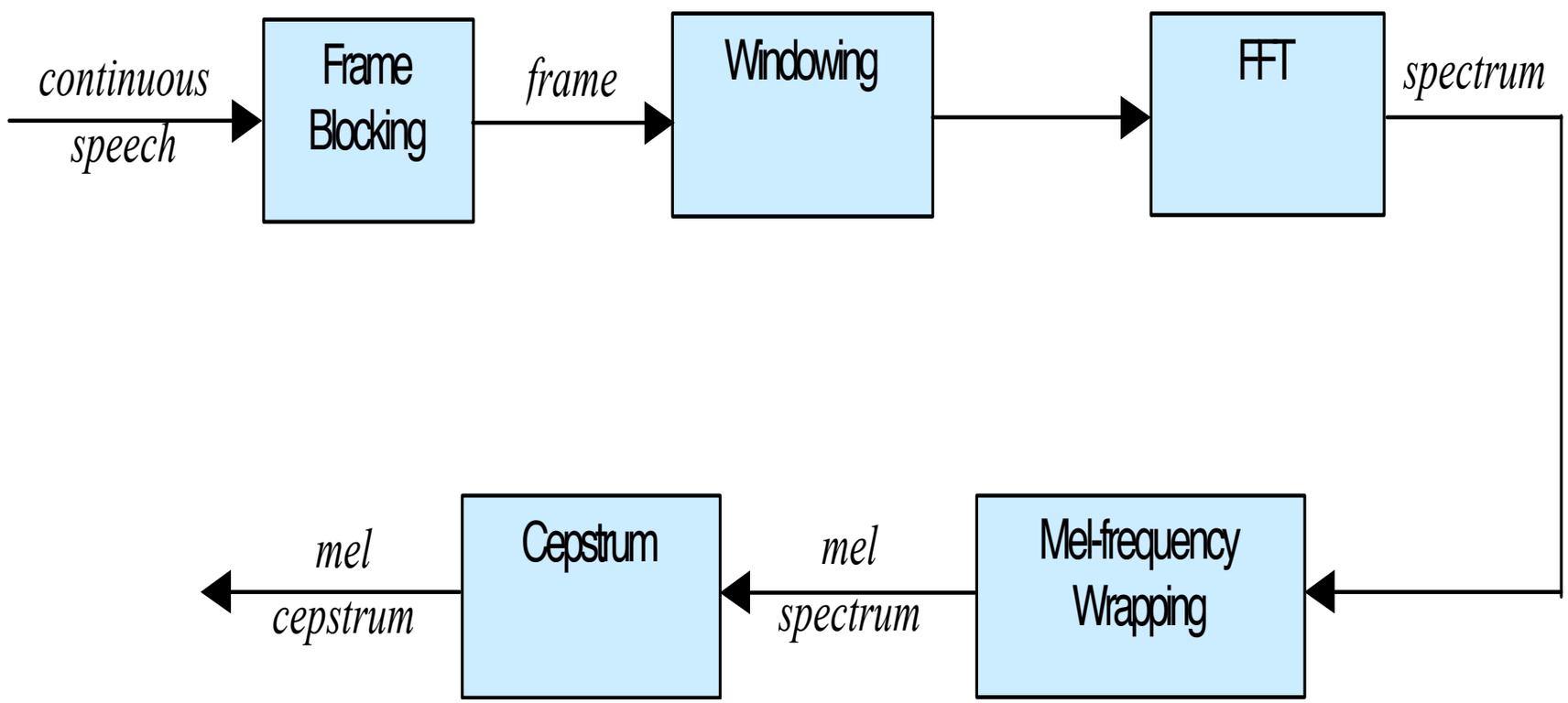
2. Fundamental Frequency Estimation

- Fundamental Frequency (*f0) or pitch is defined as the frequency at which the vocal cords vibrate during a voiced* sound.
- It is useful in speech recognition of tonal languages (e.g. Chinese) and languages that have some tonal components (e.g. Japanese).
- Fundamental frequency is often processed on logarithmic scale, rather than a linear scale to match the resolution of human auditory system.

Spectral Analysis techniques

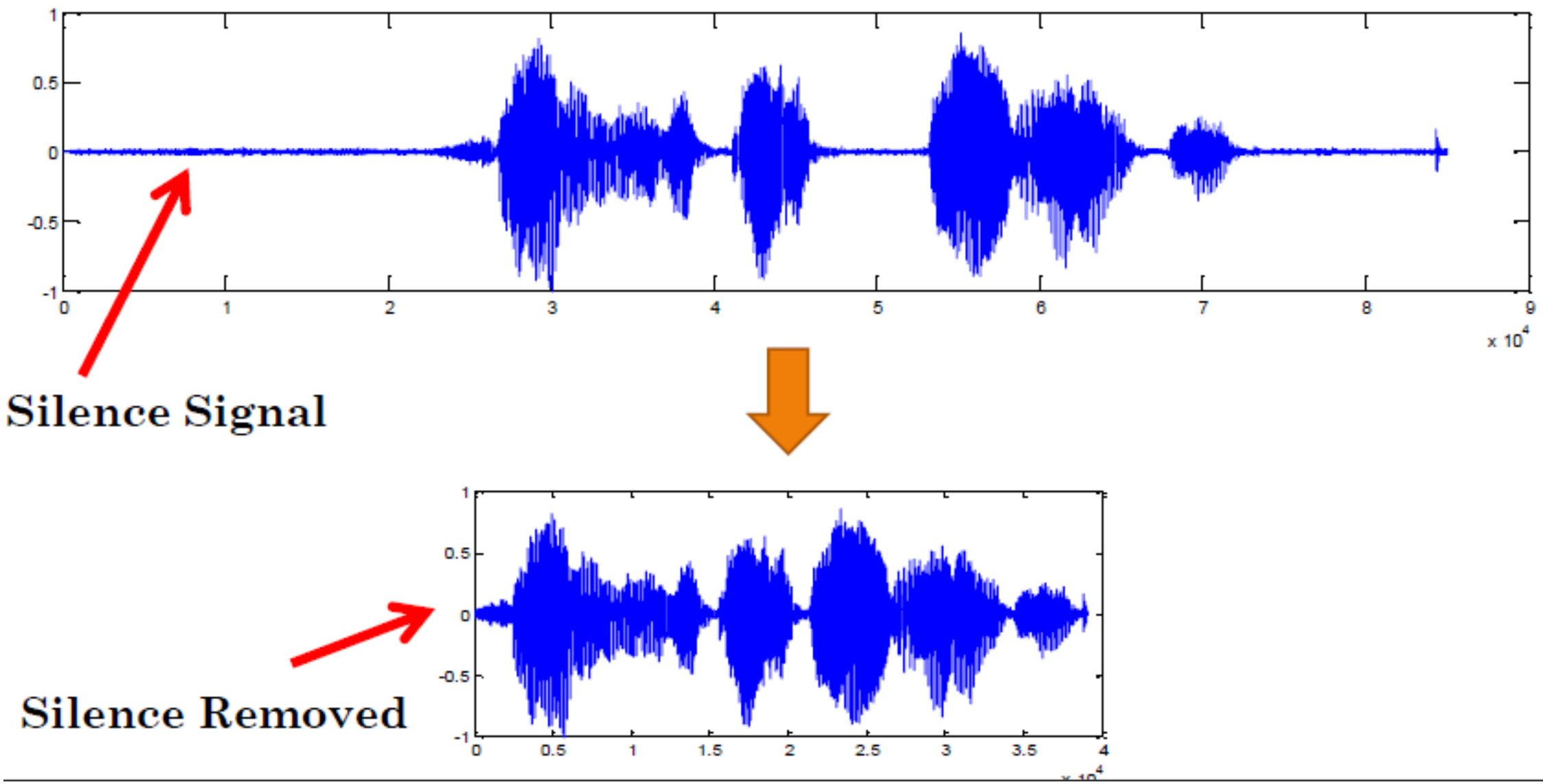
1. Mel-frequency cepstrum coefficients(MFCC)

   - This analysis technique uses cepstrum with a nonlinear frequency axis following *mel scale.*



Pre-processing Steps

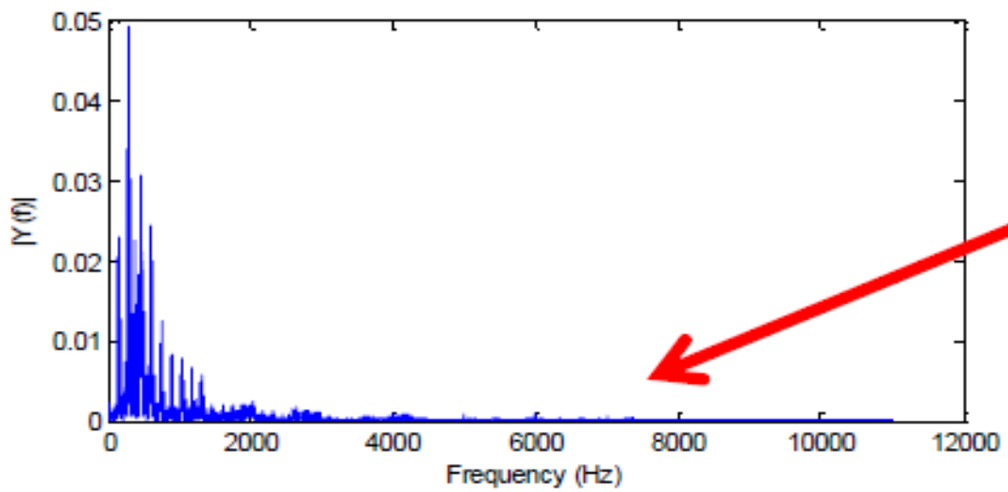
1) Silence removal



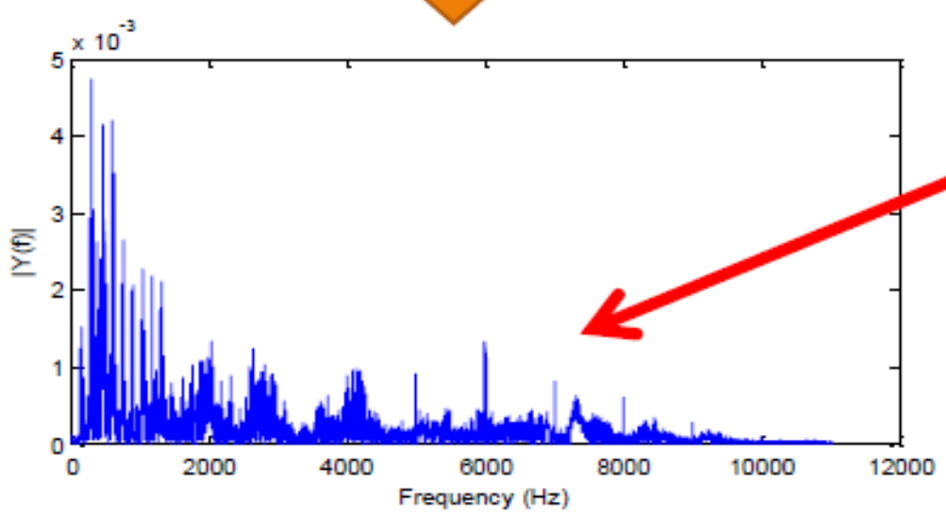
Pre-processing Steps (2)

1)Silence removal 2) Pre-emphasis

$s'[n]=s[n]-a\,s[n-1]$ …… [1]



Suppressed high Frequencies



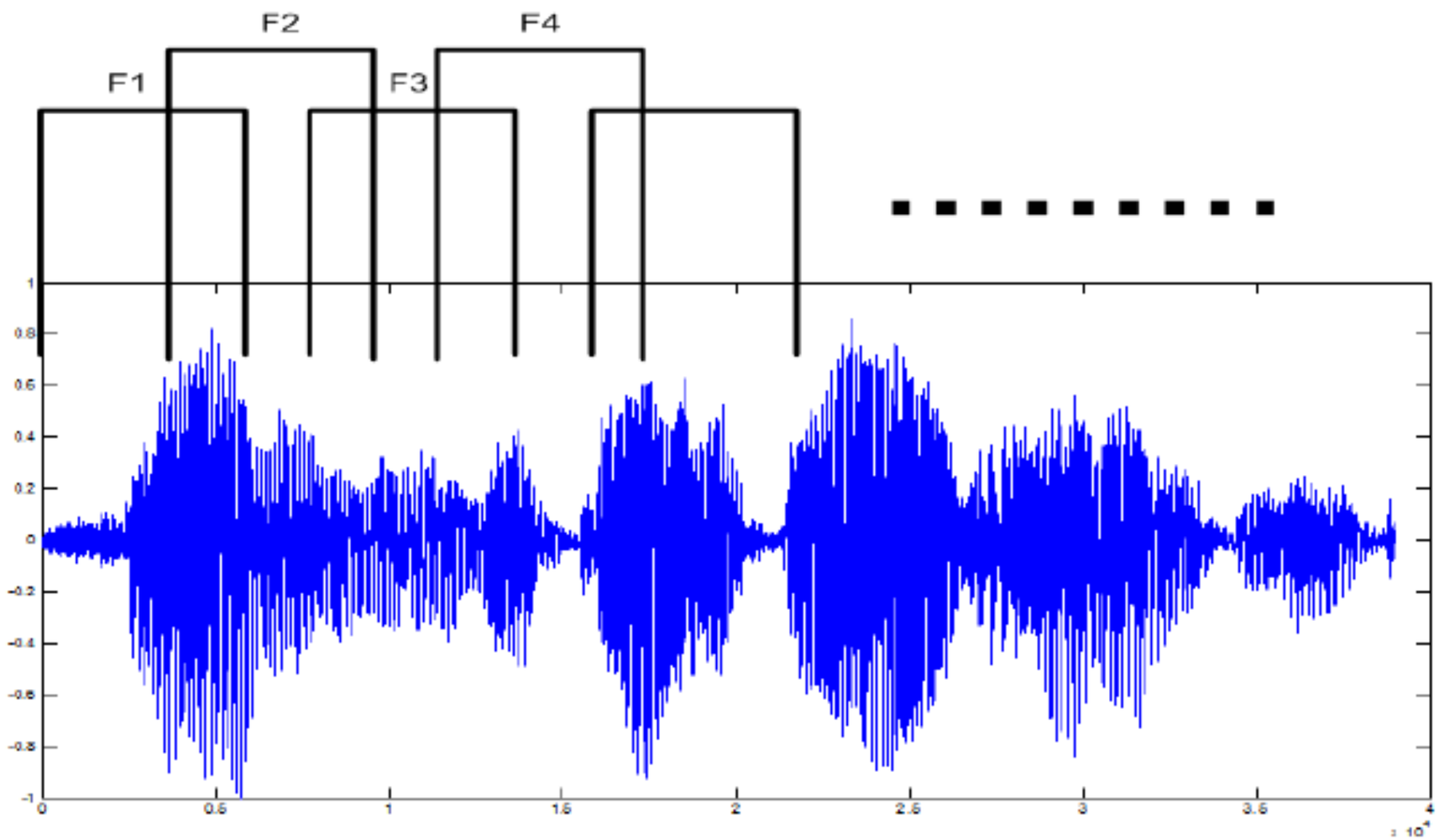
Boosted high Frequencies

Pre-processing Steps (3)

1)Silence removal 2) Pre-emphasis 3)Framing

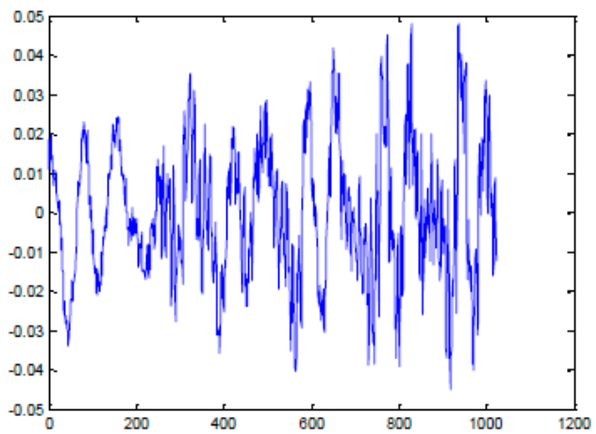
- Overlapping frames

- frame block of 23.22ms with 50% overlapping i.e., 512 samples per frame



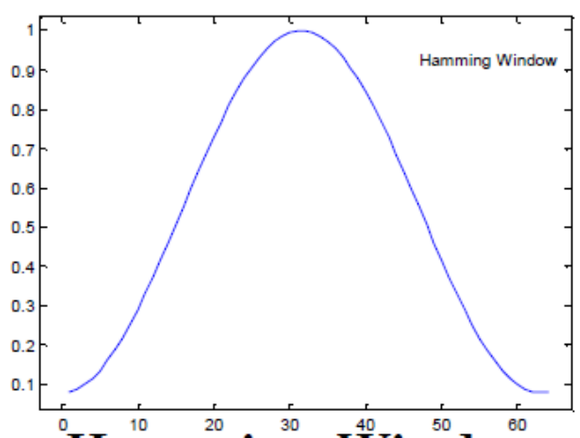
Pre-processing Steps (4)

1)Silence removal 2) Pre-emphasis 3)Framing 4)Windowing

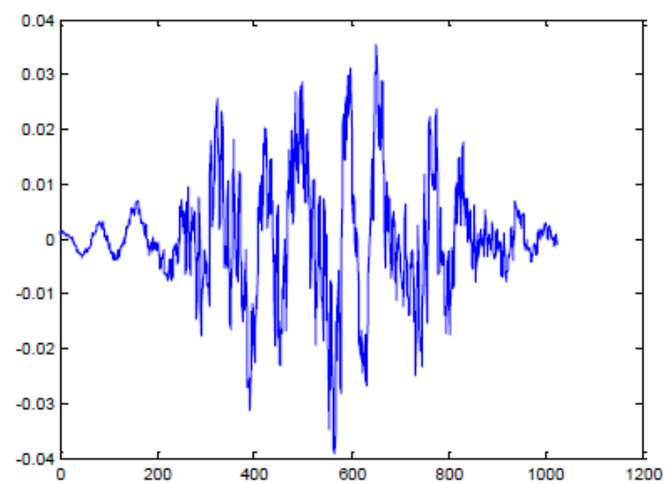


$$w[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right)$$ if n=0,1,2,…,N-1 [2]

x[n] = s'[n] . w[n-m] if n=m,m+1,…..m+N-1



Hamming Window



Windowed Signal

Feature extraction using MFCC

Steps for MFCC

1. Power spectrum of signal is computed i.e. $X[k]|^2$
2. The linear frequency is converted to Mel Scale using

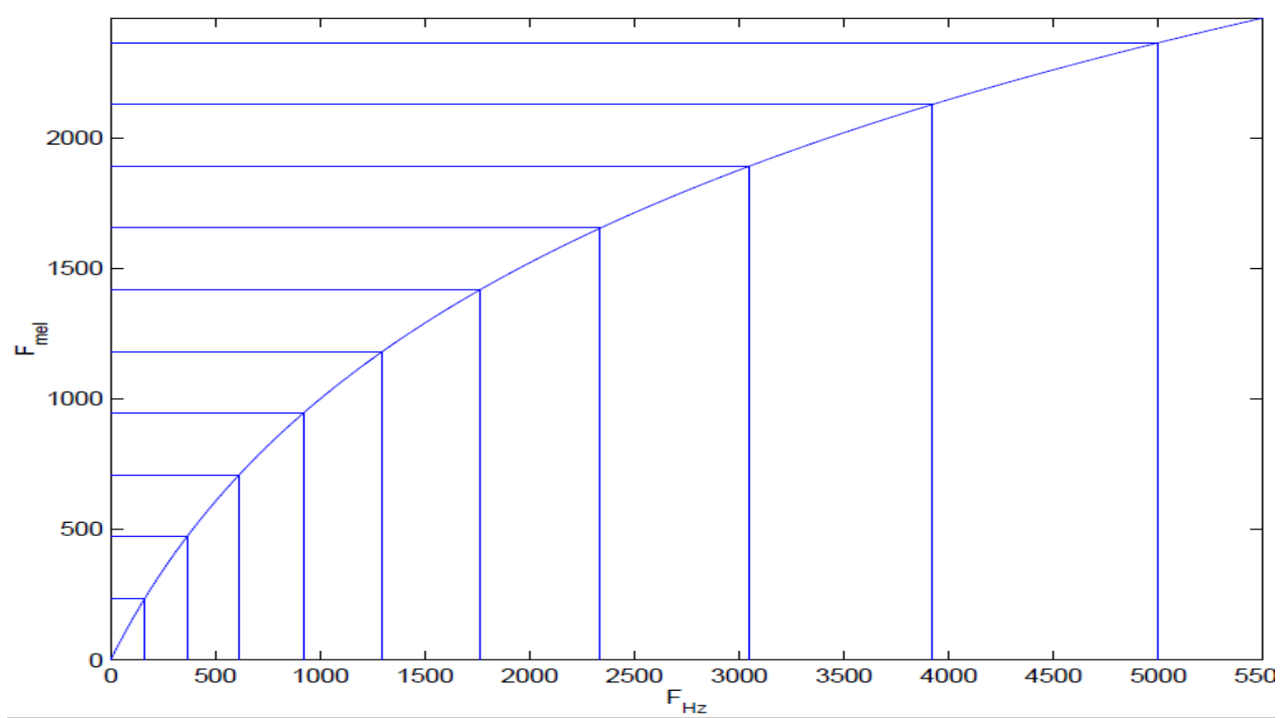


Figure: Mel scale

$$Mel(x) = 2595 \log\left(1 + \frac{x}{700}\right)$$

Feature extraction using MFCC (2)

3. The power spectrum is smoothened with a Mel spectral coefficients G[k]

$$G[k] = \sum_{n=0}^{\frac{N}{2}} \eta_{kn} |X[n]|^2 \quad ; k = 0,1,2, \ldots . K-1$$

where η_{kn} is the k-by-n matrix containing windowing coefficients , called filter bank.

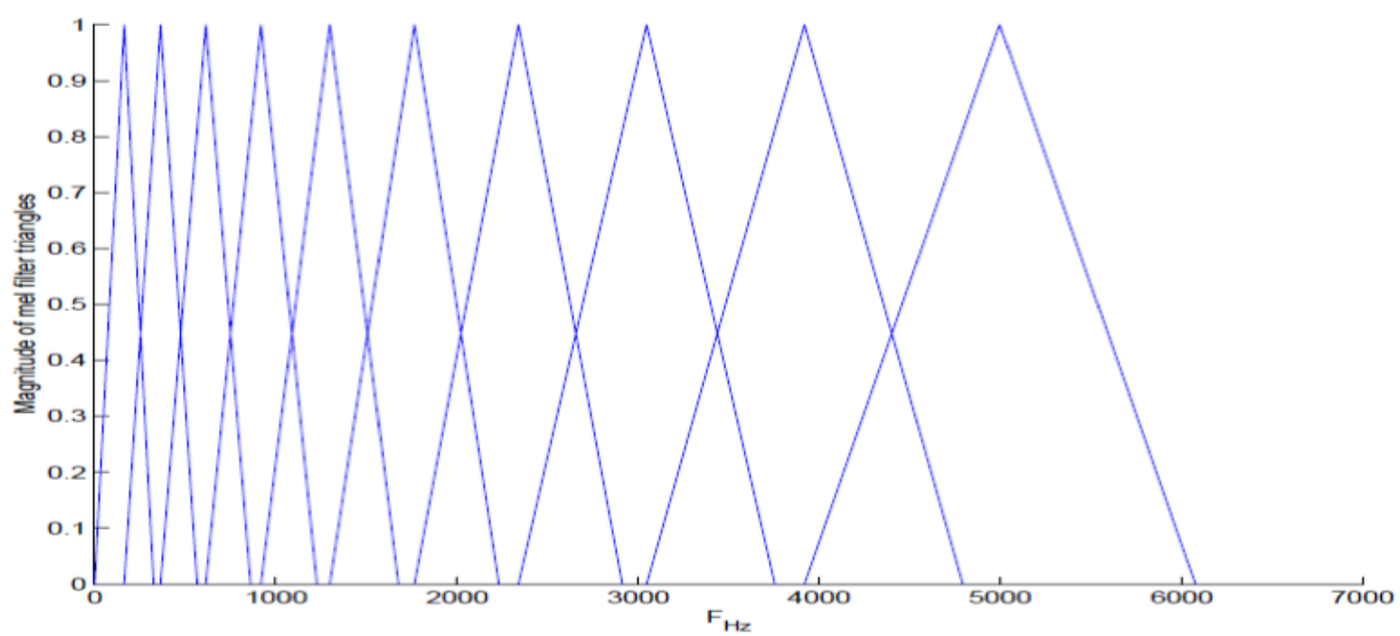


Figure: Filter bank

Feature extraction using MFCC (3)

4. log of Mel spectral coefficients are taken log(G[k]).
5. Discrete Cosine Transform (DCT) is done to compute the Mel-cepstrum c[q].

$$c[q] = \sum_{k=0}^{K-1} \log(G[k]) . \cos\left(\pi q(2k+1)/2K\right) ; q = 0,1,2,\ldots.,Q-1$$

The first few coefficients of Mel-cepstrum are taken and used for speaker recognition process.

- MFCC provide alternative representation for speech spectra which exploits auditory principles as well as decorrelating property of cepstrum.

2. Linear predictive cepstral ceofficients, LPCC

- The linear prediction method provides a robust, reliable, and accurate method for estimating the parameters that characterize the linear time-varying system representing vocal tract.
- The basic approach is to find set of predictor coefficients that will minimize the mean squared error over a short segment of speech waveform.

$$\sum_{k=1}^{p} \alpha_k \phi_n(i,k) = \phi_n(i,0) \qquad i=1,2,3....p$$

- There are two basic ways to solve above equation:

1. Covariance method
2. Autocorrelation method

Linear predictive cepstral coefficients, LPCC (2)

- In speech recognition the autocorrelation method is almost exclusively used because of its computational efficiency and inherent stability.
- Autocorrelation function is given by,

$$\sum_{k=1}^{p} \alpha_k R_n(|i-k|) = R_n(i) \qquad 1 \le i \le p$$

- The resulting equations are solved using Durbins recursive procedure.

Linear predictive cepstral coefficients, LPCC (3)

- LPC cepstral coefficients (*cm) are the extracted features* derived directly from LPC coefficients using recursions as:

$$c_0 = \ln G$$

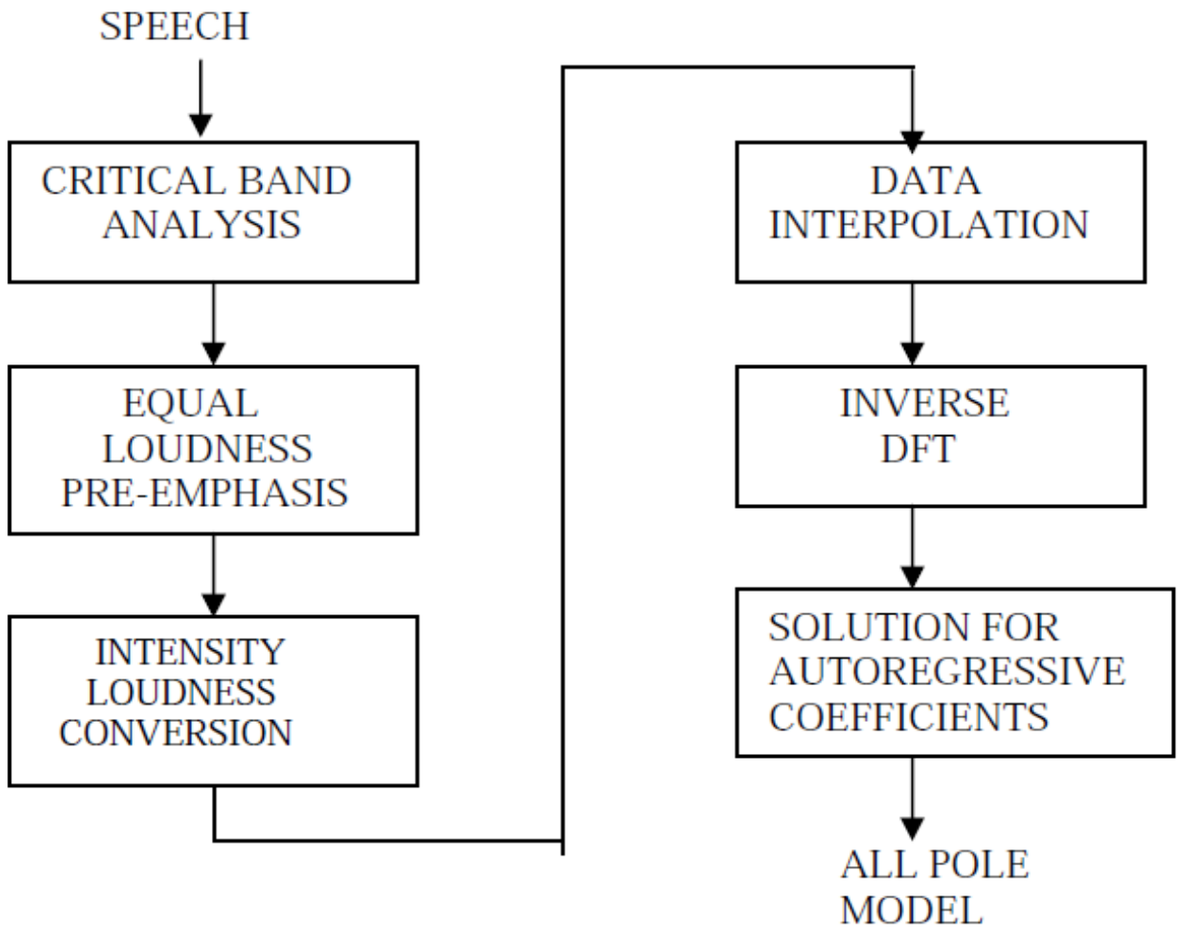
$$c_m = \alpha_m + \sum_{k=1}^{m-1} (k/m) c_k a_{m-k} \qquad 1 \le m \le p$$

$$c_m = \sum_{k=1}^{m-1} (k/m) c_k a_{m-k} \qquad m > p$$

- Where *G is the gain term in LPC model.*
- *This method is efficient, as it does not require explicit cepstral* computation. Hence combines decorrelating property of cepstrum with computational efficiency of LPC analysis.

3. Perceptually Based Linear Predictive Analysis (PLP)

- PLP analysis models perceptually motivated auditory spectrum by a low order all pole function, using the autocorrelation LP te



Perceptually Based Linear Predictive Analysis (PLP) (2)

- It involves two major steps: obtaining auditory spectrum, approximating the auditory spectrum by an all pole model.

$$\Omega(w) = 6 \ln \left((w/1200\pi) + \left((w/1200\pi) + 1 \right)^{0.5} \right)$$

Center frequency of kth critical band $\Omega_k = 0.994k$

They cover the frequency range $0 \le f \le 5$kHz ($0 \le \Omega \le 16.9$ Bark). Each filter is simulated by spectral weighing $c_k(w)$.The filter output is multiplied by equal loudness function $E(w_k)$ and converted into loudness domain using Stevens power law with exponent $r = 3$ as in[5]

$$Q_k = \left[E(w_k) \int_0^{\pi} c_k(w).P(w)\, dw \right]^{1/r}$$

Critical band weighing function is [5]

$$c_k(w) = 10^{(\Omega - \Omega k + 0.5)} \qquad \Omega \le \Omega_k - 0.5$$

$$= 1 \qquad \Omega_k - 0.5 \le \Omega \le \Omega_k + 0.5$$

$$= 10^{2.5(\Omega - \Omega k + 0.5)} \qquad \Omega_k + 0.5 \le \Omega$$

Perceptually Based Linear Predictive Analysis (PLP)(3)

- The output thus obtained is linearly interpolated to give interpolated auditory spectrum. The interpolated auditory spectrum is approximated by fifth order all pole model spectrum. The IDFT of interpolated auditory spectrum provides first six terms of autocorrelation function.
- The PLP analysis provides similar results as with LPC analysis but the order of PLP model is half of LP model. This allows computational and storage saving for ASR. Also it provides better performance to cross speaker ASR.

5.2.2 Deterministic sequence recognition

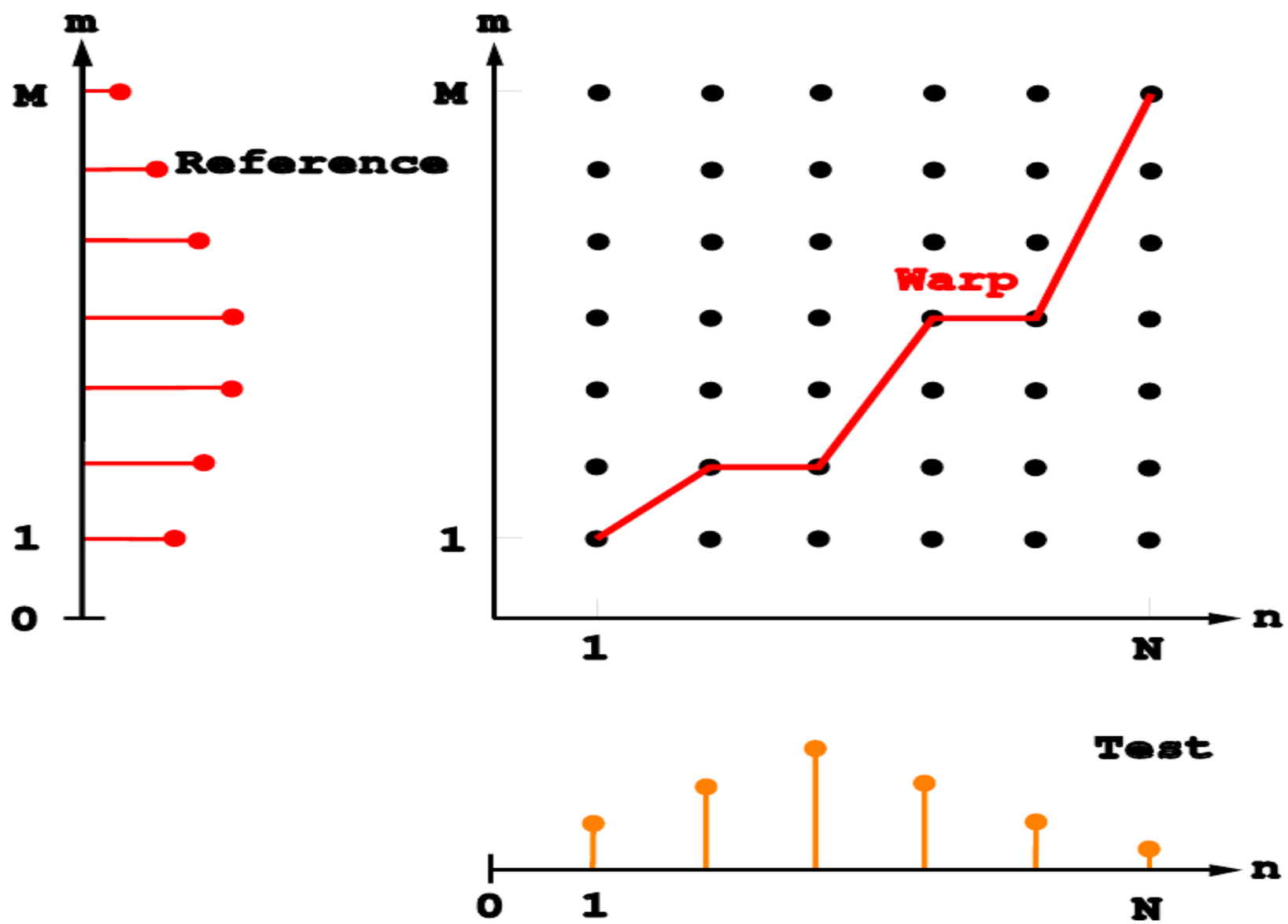
- Test pattern, $\boldsymbol{T}$, and reference patterns, $\{\boldsymbol{R}_1, \ldots, \boldsymbol{R}_V\}$, are represented by sequences of feature measurements
- Pattern similarity is determined by aligning test pattern, $\boldsymbol{T}$, with reference pattern, $\boldsymbol{R}_v$, with distortion $\mathcal{D}(\boldsymbol{T}, \boldsymbol{R}_v)$
- Decision rule chooses reference pattern, $\boldsymbol{R}^*$, with smallest alignment distortion $\mathcal{D}(\boldsymbol{T}, \boldsymbol{R}^*)$

$$\boldsymbol{R}^* = \arg\min_v \mathcal{D}(\boldsymbol{T}, \boldsymbol{R}_v)$$

- Dynamic time warping (DTW) is used to compute the best possible alignment warp, ϕ_v, between $\boldsymbol{T}$ and $\boldsymbol{R}_v$, and the associated distortion $\mathcal{D}(\boldsymbol{T}, \boldsymbol{R}_v)$

Signal template matching

- Frame wise comparison of unknown word and stored templates:



Dynamic Time Warping (DTW)

- Objective: an optimal alignment between variable length sequences $\boldsymbol{T} = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N\}$ and $\boldsymbol{R} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_M\}$
- The overall distortion $\mathcal{D}(\boldsymbol{T}, \boldsymbol{R})$ is based on a sum of local distances between elements $d(\boldsymbol{t}_i, \boldsymbol{r}_j)$
- A particular alignment warp, ϕ, aligns $\boldsymbol{T}$ and $\boldsymbol{R}$ via a point-to-point mapping, $\phi = (\phi_t, \phi_r)$, of length K_ϕ

$$\boldsymbol{t}_{\phi_t(k)} \Longleftrightarrow \boldsymbol{r}_{\phi_r(k)} \quad 1 \leq k \leq K_\phi$$

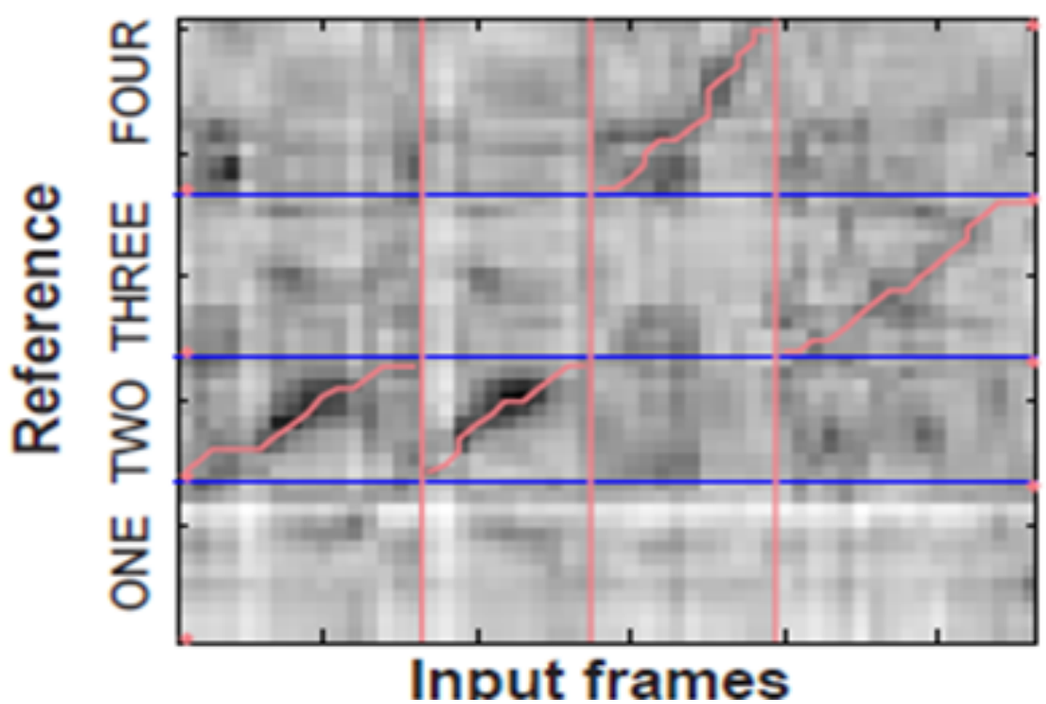
- The optimal alignment minimizes overall distortion

$$\mathcal{D}(\boldsymbol{T}, \boldsymbol{R}) = \min_{\phi} \mathcal{D}_\phi(\boldsymbol{T}, \boldsymbol{R})$$

$$\mathcal{D}_\phi(\boldsymbol{T}, \boldsymbol{R}) = \frac{1}{M_\phi} \sum_{k=1}^{K_\phi} d(\boldsymbol{t}_{\phi_t(k)}, \boldsymbol{r}_{\phi_r(k)}) m_k$$

DTW-based recognition

- **Reference templates for each possible word**
- **For isolated words:**
  - mark endpoints of input word
  - calculate scores through each template
  - choose best
- **For continuous speech**
  - one matrix of template slices;
    special-case constraints at word ends



5.2.3 Statistical sequence recognition

- Can be seen as extension of template-based approach, using more powerful mathematical and statistical tools .
- Collect a large corpus of transcribed speech recordings
- Train the computer to learn the correspondences (“machine Learning”)
- At run time, apply statistical processes to search through the space of all possible solutions, and pick the statistically most likely one.

Statistical sequence recognition (2)

- **DTW limited because it's hard to optimize**
  - interpretation of distance, transition costs?
- **Need a theoretical foundation: Probability**
- **Formulate recognition as MAP choice among models:**

$$M^* = \underset{M_j}{\operatorname{argmax}}\ p(M_j | X, \Theta)$$

  - X = observed features
  - M_j = word-sequence models
  - Θ = all current parameters

Statistical formulation (3)

- **Can rearrange via Bayes' rule (& drop $p(X)$):**

$$M^* = \underset{M_j}{\operatorname{argmax}}\ p(M_j | X, \Theta)$$

$$= \underset{M_j}{\operatorname{argmax}}\ p(X | M_j, \Theta_A)\, p(M_j | \Theta_L)$$

- $p(X \mid M_j)$ =
  likelihood of observations under model
- $p(M_j)$ = prior probability of model
- Θ_A = acoustics-related model parameters
- Θ_L = language-related model parameters

5.2.4 Recognition

- Given, a trained speech recognition and a test speech signal, the goal is to hypothesize the best sentence/ word sequence.

$$\widehat{\mathbf{W}} = \underset{\mathbf{W}}{argmax} \quad P(\mathbf{W}|\mathbf{A})$$

Where A is the extracted acoustic features from test da $\widehat{\mathbf{W}}$ is optimal word sequence .

Using Bayes' rule, we can re-write $P(\mathbf{W}|\mathbf{A})$ as

$$P(\mathbf{W}|\mathbf{A}) = \frac{P(\mathbf{A}|\mathbf{W})P(\mathbf{W})}{P(\mathbf{A})}$$

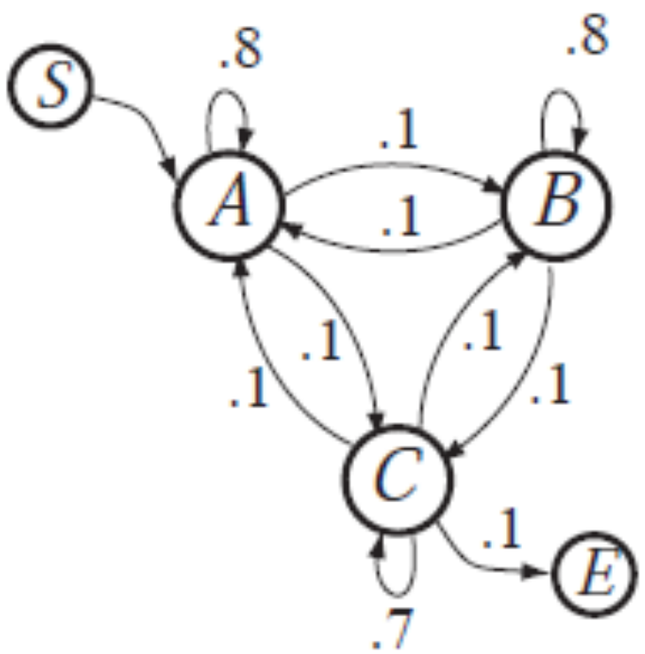
- P(A|W) = likelihood of feature, P(W) = prob. Of the word sequence, P(A) is priori prob. Which is independent of acoustic and language models.
- So, the probability of word sequence is product of prob. of the acoustic model and that of the language models.

Acoustic model

- Should be capable of modeling predictable variations of acoustic characteristics of speech as well as other variations due to speaker, channel etc.
- Most popular acoustic model is Hidden Markov Model.

Markov models

- A (first order) Markov model is a finite-state system whose behavior depends only on the current state
- E.g. generative Markov model:



$p(q_{n+1}\|q_n)$	q_{n+1}: S	A	B	C	E
q_n S	0	1	0	0	0
A	0	.8	.1	.1	0
B	0	.1	.8	.1	0
C	0	.1	.1	.7	.1
E	0	0	0	0	1

S A A A A A A A A B B B B B B B B B C C C C B B B B B B C E

5.2.5 Language models

- Role of language model is to drive the best sentence hypothesis subject to constraints of the language.
- Incorporates various types of linguistic information's.
- Lexicon - specify the sequence of phonemes which form valid words of the language.
- Syntax – describes the rules of combining words to form valid sentences.

Statistical language models (SLM)

- It estimates the probability distributions of various linguistic units or their composites.

n-grams : The simplest grammar that specifies the probability of the nth word in a sequence, given the previous n-1 words of the sequence.

Let W denotes the word sequence w, w2, …, wi . The probability of this sentence is

p(w) = p(w1) * p(w2lw1)*p(w3lw1,w2)*…*p(wilw1,w2, …,wi-1)

- it makes (n-1)th order Markovian approximation, so uses knowledge of only n-1 previous words.

n-grams

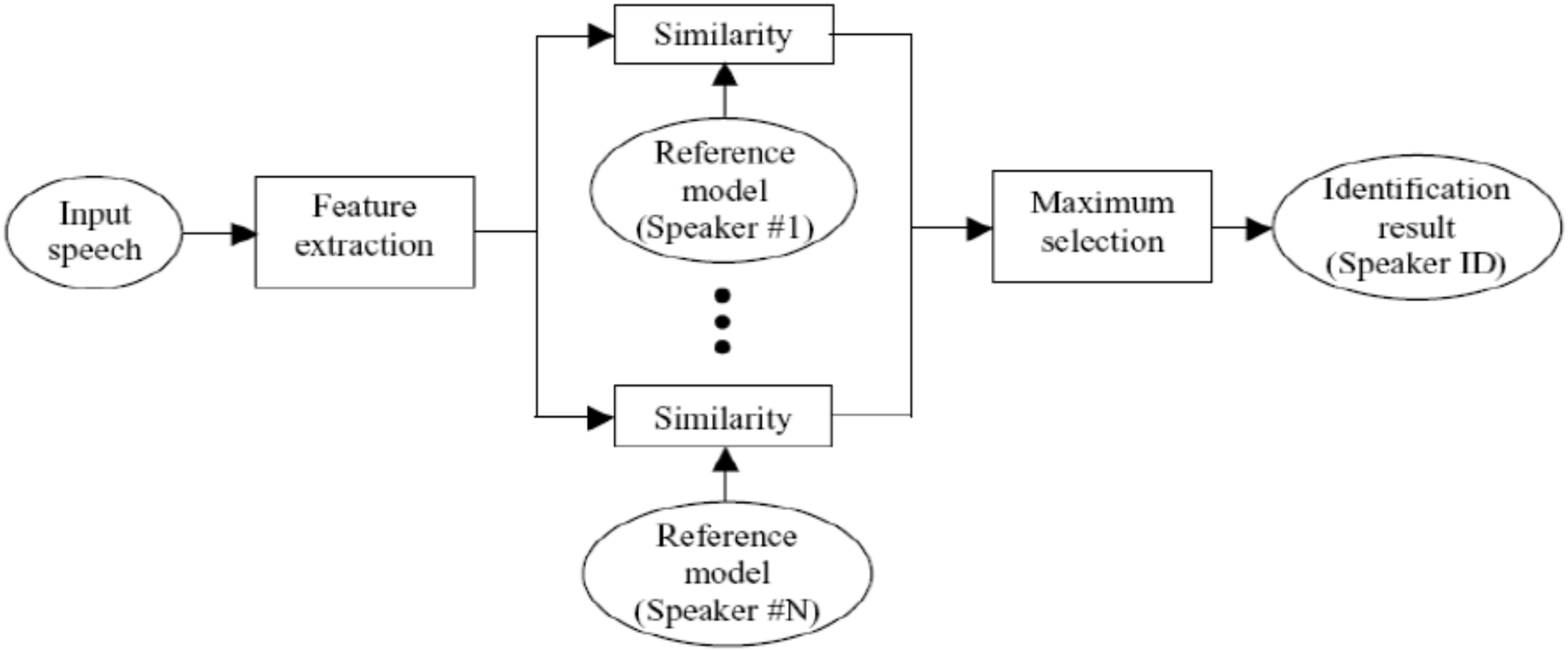
- Maximun likelihood estimate of n –gram can be written as:
- Unigram : p(w)=C(w)I $|V|$
- n- gram : p(wnlw1, w2,…,wn-1)= $\frac{C(w1,w2,...,wn)}{C(w1,w2,...,wn-1)}$
- C (w1,w2,…,wn-1,wn) is the count(frequency) of the word sequence w1, w2,……wn in the train corpus.

5.3 Speaker identification and verification

- Speech is one of the important biometric.
- Speaker recognition is the process of automatically recognizing who is speaking on the basis of individual information included in speech waves.
- This technique makes it possible to use the speaker's voice to verify their identity and control access to various services.

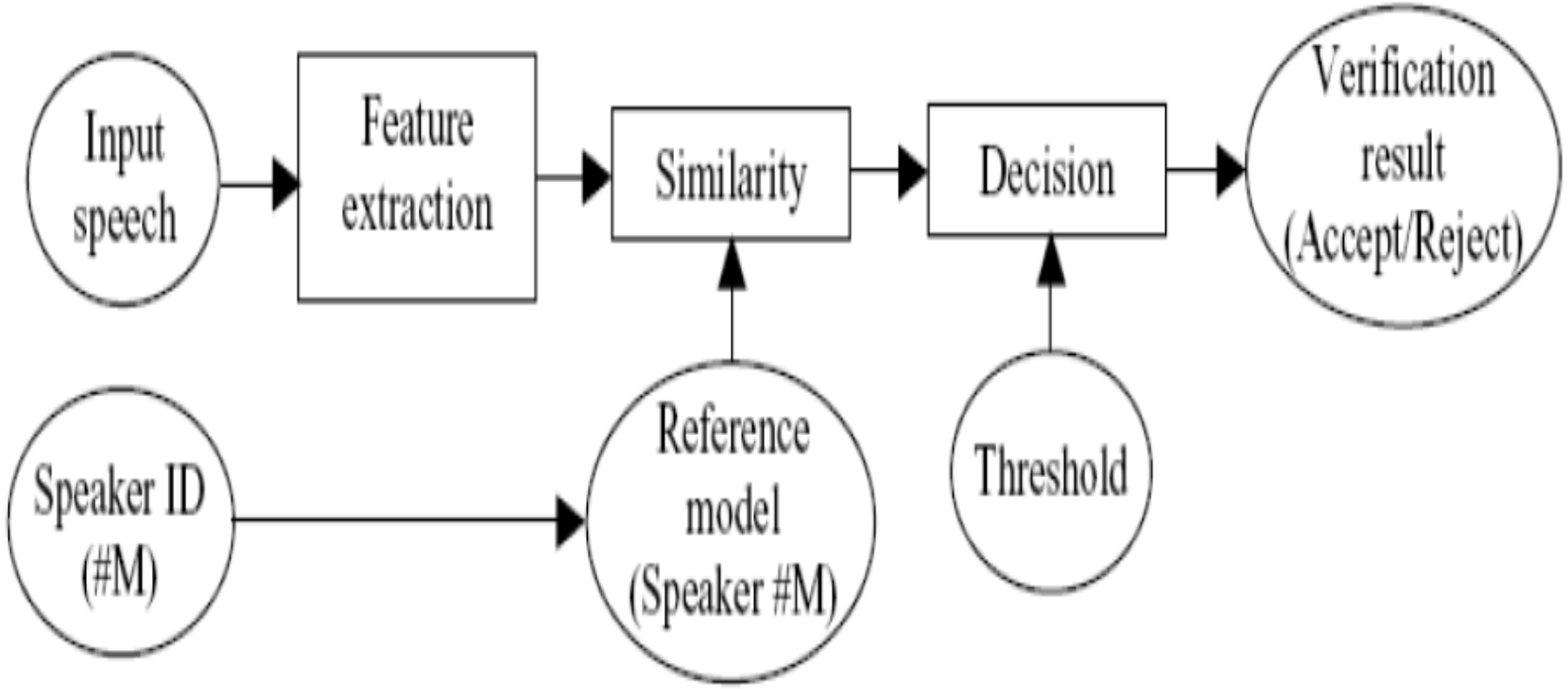
Principles of Speaker Recognition

- Speaker recognition can be classified into identification and verification.
- ***Speaker identification*** is the process of determining which registered speaker provides a

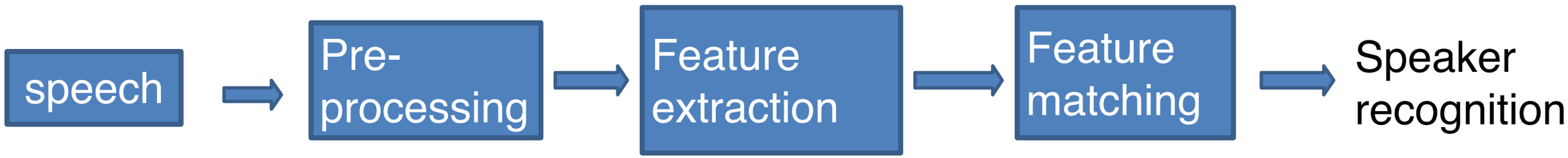


Principles of Speaker Recognition (2)

- ***Speaker verification***, on the other hand, is the process of accepting or rejecting the identity claim of a speaker.



System diagram



Speaker Recognition system

- Pre- processing and feature extraction for speaker recognition is similar to speech recognition.
- In speaker recognition,

  feature extraction => MFCC,

  feature matching => Gaussian Mixture Model (GMM), are exclusively used.

Gaussian Mixture Model (GMM)

- A Gaussian Mixture Model (GMM) is a parametric probability density function.
- GMM is based on clustering technique, where the entire set of experimental data set is modeled by a mixture of Gaussians.
- But unlike K-Means, GMMs are able to build soft clustering boundaries, i.e., points in space can belong to any class with a given probability.

Gaussian Mixture Model (2)

A GMM is the weighted sum of M Gaussian component densities given by the equation

$$p(\vec{x}/\lambda) = \sum_{m=1}^{M} w_m \cdot g_m(\vec{x}/\vec{\mu}_m, C_m)$$

$g_m(\vec{x}/\vec{\mu}_m, C_m)$, m=1,2,…,M are the component densities where each component density is a k-dimensional Gaussian function (pdf) of the form

$$g_m(\vec{x}/\vec{\mu}_m, C_m) = \frac{1}{\sqrt{(2\pi)^K . |C_m|}} \exp\{-\frac{1}{2}(\vec{x} - \vec{\mu}_m).(C_m{}^{-1}(\vec{x} - \vec{\mu}_m))\}$$

Gaussian Mixture Model (3)

Where,

$\overrightarrow{\mu_m}$ is the mean vector of length k of mth Gaussian PDF,

C_m is the covariance matrix of k×k of mth Gaussian PDF

Thus the complete Gaussian Mixture Model is parameterized by mixture weights, mean vectors and covariance matrices for all component densities. The parameters are collectively represented by the notation,

$$\lambda = \{ w_m, \vec{\mu}_m , C_m \} , \text{m} = 1, 2,\ldots, \text{M} \tag{18}$$

These parameters are estimated in training section. For speaker recognition system, each speaker is represented by a GMM and is referred by his/her model λ.

GMM training

- goal of speaker model training is to estimate the parameters of the GMM that best match the distribution of training features vectors and hence develop a robust model for the speaker.
- Out of several techniques available for estimating the parameters of GMM, the system used **Maximum Likelihood (ML)** estimation or **Expectation-Maximization (EM)** technique to model the GMM.

Speaker identification

- The extracted feature vector from speech signal ,

  $X = \{x1, x2, x3, \ldots, xT\}$

- The rule to determine if *X* has come from speaker *s* can be stated as

  $p(\lambda_s \mid X) > p(\lambda_r \mid X) \quad r = 1, 2, \ldots, S \ (r \neq s)$

- for each speaker *s* in the speaker set, the classification system computed and found the value of *s* that maximizes $p(\lambda_s / X)$ according to

$$p(\lambda_s|X) = \frac{p(X|\lambda_s)p(\lambda_s)}{p(X)}$$

Speaker identification (2)

- if prior probability of each speaker is equal, then the term of $p(\lambda_s)$ can be ignored then

$$P(\lambda_s / X) = p(X / \lambda_s)$$

- Where, $$p(X|\lambda s) = \prod_{t=1}^{T} p(x_t|\lambda s)$$

- Practically, the individual probabilities will be low so $p(X / \lambda_s)$ will underflow, so they are calculated in log domain. So, The likelihood of any speaker having spoken the test data is:

$$L(\lambda_s) = \sum_{t=1}^{T} \ln(p(x_t|\lambda_s))$$

- the speaker of the test data was statistically chosen by,

$$speaker = max_{s=1}^{S} L(\lambda_s)$$

Speaker verification

- Its an hypothesis testing problem

  - based on the input speech observations, it must be decided whether the claimed identity of the speaker is correct or not.

The hypothesis test is given by,

- H0: the speaker is the claimed speaker
- H1: the speaker is an imposter

The decision is based on likelihood ratio

$$\frac{P(X/\lambda)}{P(X/\bar{\lambda})}$$

Where, $P(X/\lambda)$ is the likelihood that the utterance was produced by speaker model λ while $P(X/\bar{\lambda})$ is the likelihood that he utterance was produced by imposter model $\bar{\lambda}$.

Speaker verification (2)

Here, the imposter model $\bar{\lambda}$, also called as Universal Background Model (UBM), is obtained by training a collection of speech samples from a large no. of speakers, representative of the population of speakers.

The likelihood ratio is often expressed in logarithm as

$$\Delta(X) = \log\left(\frac{P(X/\lambda)}{P(X/\bar{\lambda})}\right) = \log P(X/\lambda) - \log P(X/\bar{\lambda})$$

The decision is made as follows:

If $\Delta(X) < \theta_T$, reject null hypothesis i.e. the speaker is an imposter.

If $\Delta(X) > \theta_T$, accept null hypothesis i.e. the speaker is the claimed one.

Here, threshold should be taken optimal.

Performance measurement in speaker verification

- determined by False Rejection Rate (FRR) and False Acceptance Rate (FAR).

1) False Rejection Rate(FRR)

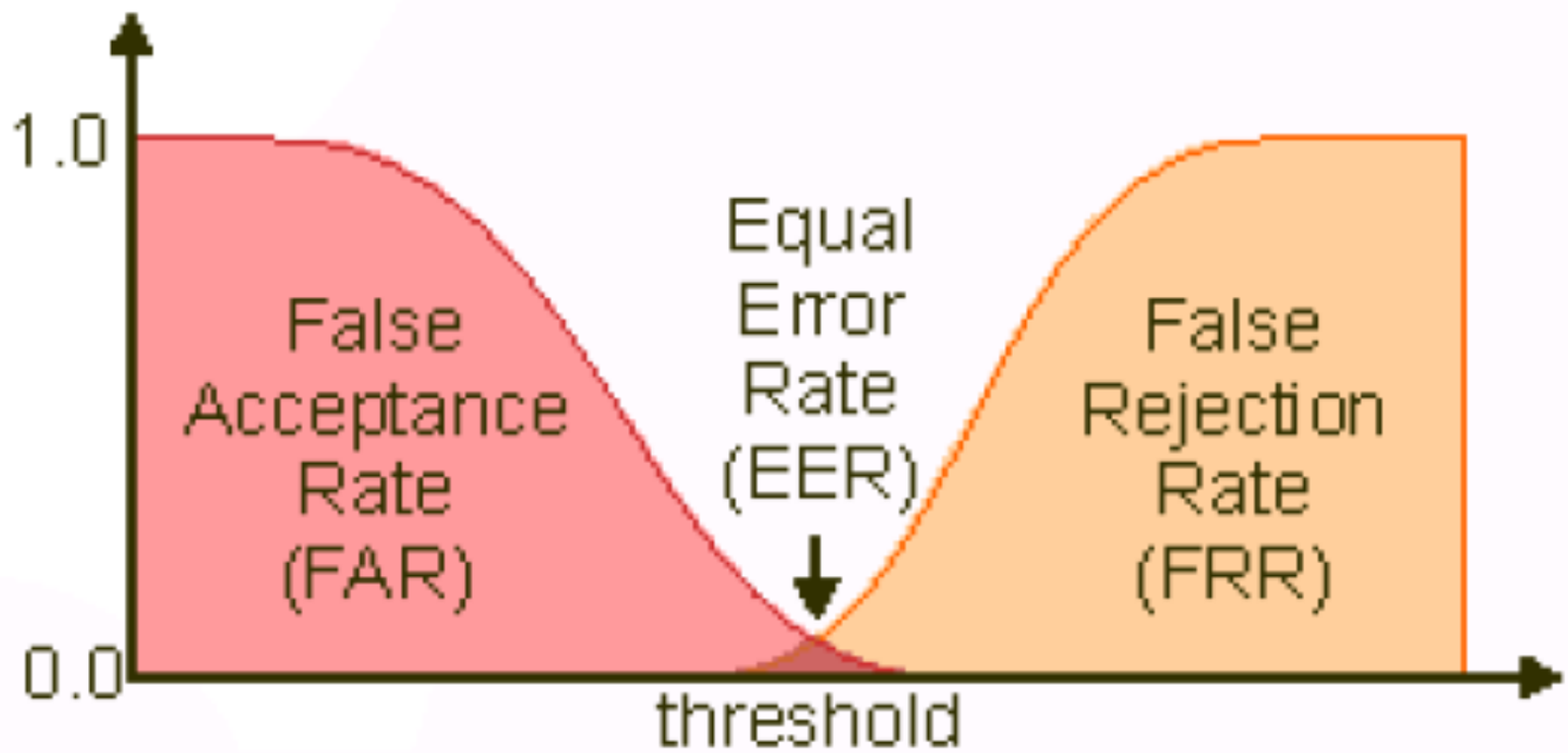
- likelihood that the system will incorrectly reject an access attempted by an authorized user i.e. ratio of the number of false rejections divided by the number of verification tests.

2) False Acceptance Rate(FAR)

- likelihood that the system will incorrectly accept an access attempt by an unauthorized user i.e. ratio of the number of false acceptances divided by the number of verification tests.

Performance measurement in speaker verification (2)

Total Error Rate (TER) is the combination of false rejection and false acceptance rate. And the requirement of the system is to minimize the Total Error Rate.



Voice response system

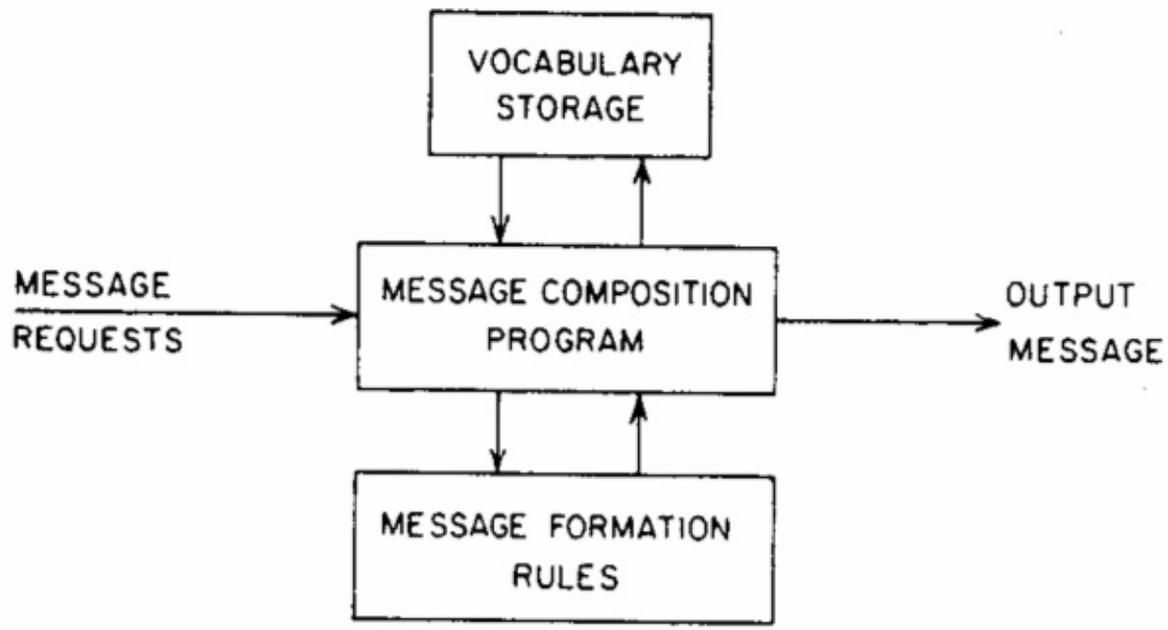


Fig. 9.1 Block diagram of a voice response system.

Figure 9.1 shows a block diagram of a general computer voice response system. The elements of a voice response system include:

1. Provision for storage of a vocabulary for the voice response system.
2. Rules for forming messages from elements of the vocabulary.
3. A program for composing voice response messages.

Since the objective of a voice response system is to produce speech utterances that are useful for communication with humans, intelligibility is of paramount importance. However, subjective factors such as quality and naturalness of the speech utterances have a great effect on the usefulness and acceptability of a voice response system. Thus, it is important that the three components of the voice response system be designed so that there is maximum potential for the production of highly intelligible, natural sounding speech.

Voice response system

In considering the choice of digital coding method for voice response applications it is helpful to consider three factors:

1. The information rate (bit rate) required for acceptable speech quality.
2. The complexity of the coding and decoding schemes.
3. The flexibility of the representation; i.e., the potential for modification of the vocabulary elements.

Typical applications of computer voice response systems

1. A system for producing vocal instructions for wiring communications equipment
2. A directory assistance system
3. A stock price quotation system
4. A data set testing information system
5. A flight information system
6. A speaker verification system

Voice response system

Speech synthesis

- What is the task?
  - Generating natural sounding speech on the fly, usually from text
- What are the main difficulties?
  - What to say and how to say it
- How is it approached?
  - Two main approaches, both with pros and cons
- How good is it?
  - Excellent, almost unnoticeable at its best
- How much better could it be?
  - marginally

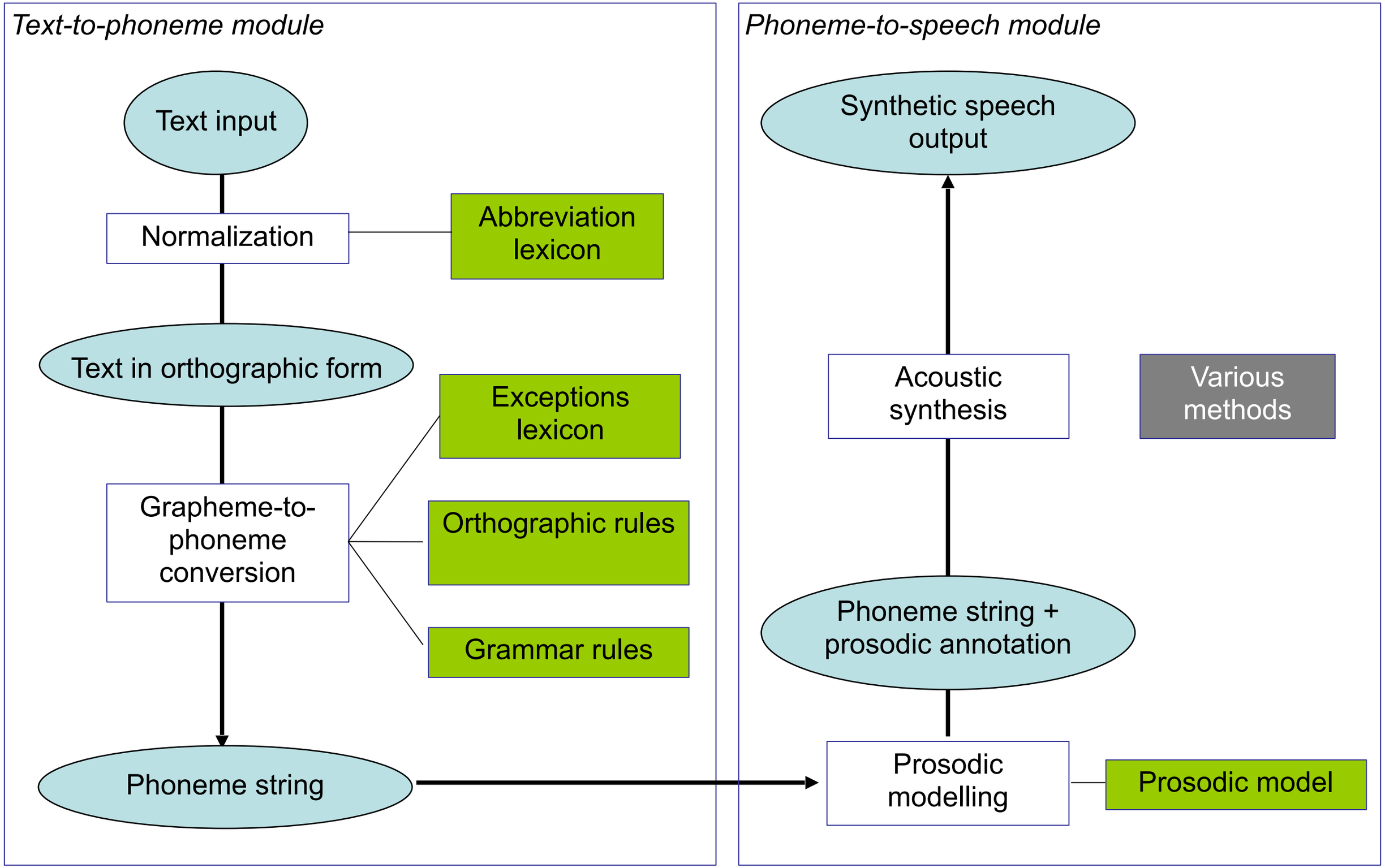
Input type

- Concept-to-speech vs text-to-speech
- In CTS, content of message is determined from internal representation, not by reading out text
  - E.g. database query system
  - No problem of text interpretation

Text-to-speech

- What to say: text-to-phoneme conversion is not straightforward
  - *Dr Smith lives on Marine Dr in Chicago IL. He got his PhD from MIT. He earns $70,000 p.a.*
  - *Have toy read that book? No I'm still reading it. I live in Reading.*
- How to say it: not just choice of phonemes, but allophones, coarticulation effects, as well as prosodic features (pitch, loudness, length)

Architecture of TTS systems



Text normalization

- Any text that has a special pronunciation should be stored in a lexicon
  - Abbreviations (Mr, Dr, Rd, St, Middx)
  - Acronyms (UN but UNESCO)
  - Special symbols (&, %)
  - Particular conventions (£5, $5 million, 12°C)
  - Numbers are especially difficult
    - 1995 2001 1,995 ₩236 3017 233 4488

Grapheme-to-phoneme conversion

- English spelling is complex but largely regular, other languages more (or less) so
- Gross exceptions must be in lexicon
- Lexicon or rules?
  - If look-up is quick, may as well store them
  - But you need rules anyway for unknown words
- MANY words have multiple pronunciations
  - Free variation (eg *controversy*, *either*)
  - Conditioned variation (eg *record*, *import*, weak forms)
  - Genuine homographs

Grapheme-to-phoneme conversion

- Much easier for some languages (Spanish, Italian, Welsh, Czech, Korean)
- Much harder for others (English, French)
- Especially if writing system is only partially alphabetic (Arabic, Urdu)
- Or not alphabetic at all (Chinese, Japanese)

Syntactic (etc.) analysis

- Homograph disambiguation requires syntactic analysis
  - *He makes a record of everything they record.*
  - *I read a lot. What have you read recently?*
- Analysis also essential to determine appropriate prosodic features

Prosody modelling

- Pitch, length, loudness
- Intonation (pitch)
  - essential to avoid monotonous robot-like voice
  - linked to basic syntax (eg statement vs question), but also to thematization (stress)
  - Pitch <u>range</u> is a sensitive issue
- Rhythm (length)
  - Has to do with pace (natural tendency to slow down at end of utterance)
  - Also need to pause at appropriate place
  - Linked (with pitch and loudness) to stress

Speech Synthesis: A Basic Overview

- Speech synthesis is the generation of speech by machine.
- The reasons for studying synthetic speech have evolved over the years:

1. Novelty
2. To control acoustic cues in perceptual studies
3. To understand the human articulatory system
   - “Analysis by Synthesis”
4. Practical applications
   - Reading machines for the blind, navigation systems

Speech Synthesis: A Basic Overview

- There are four basic types of synthetic speech:

1. Mechanical synthesis
2. Formant synthesis
   - Based on Source/Filter theory
3. Concatenative synthesis
   - = stringing bits and pieces of natural speech together
4. Articulatory synthesis
   - = generating speech from a model of the vocal tract.

1. Mechanical Synthesis

- The very first attempts to produce synthetic speech were made without electricity.
  - = **mechanical synthesis**
- In the late 1700s, models were produced which used:
  - reeds as a voicing source
  - differently shaped tubes for different vowels

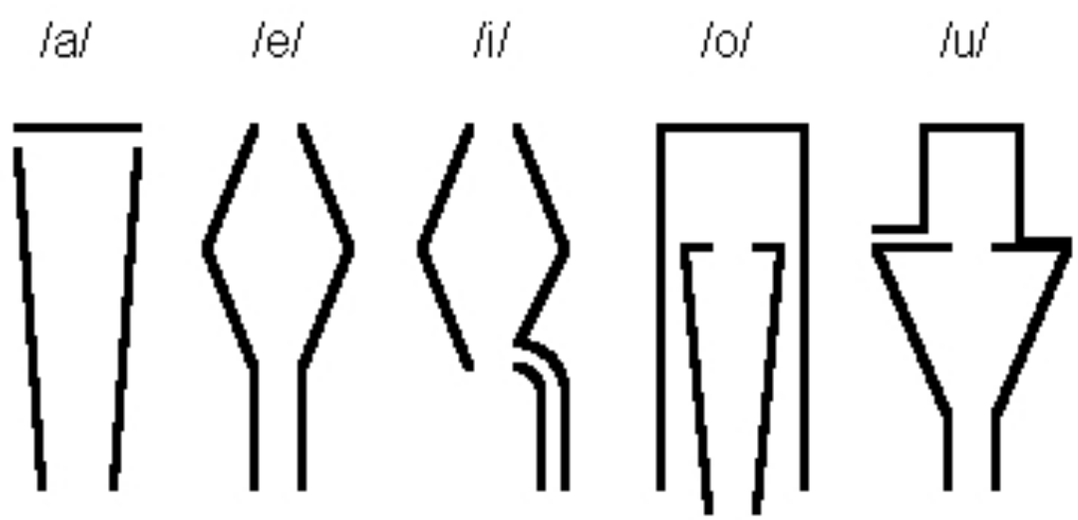
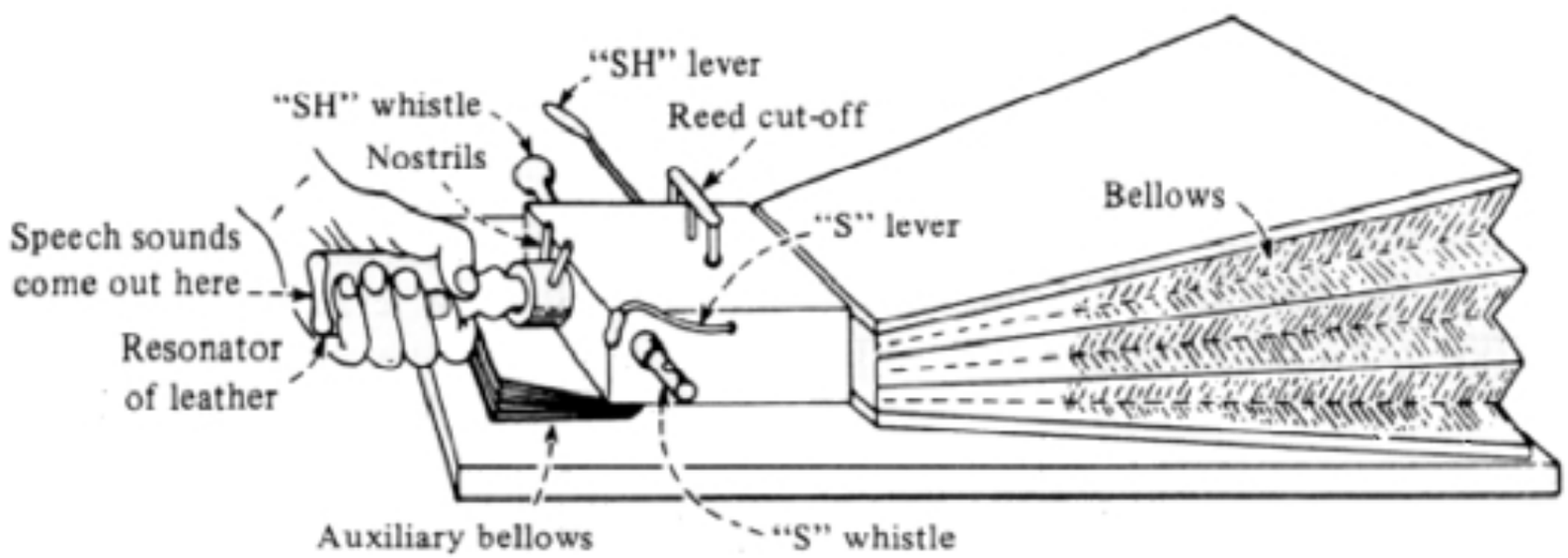
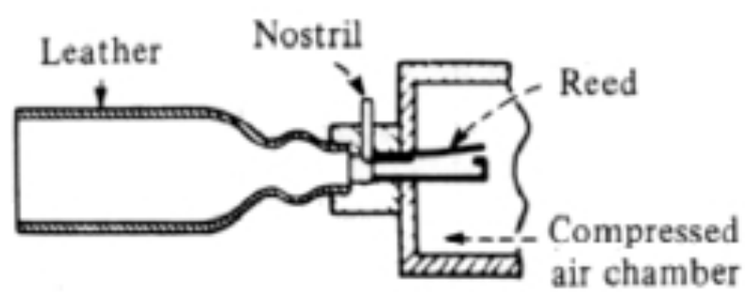


Fig. 2.1. Kratzenstein's resonators (Schroeder 1993).

Mechanical Synthesis, part II

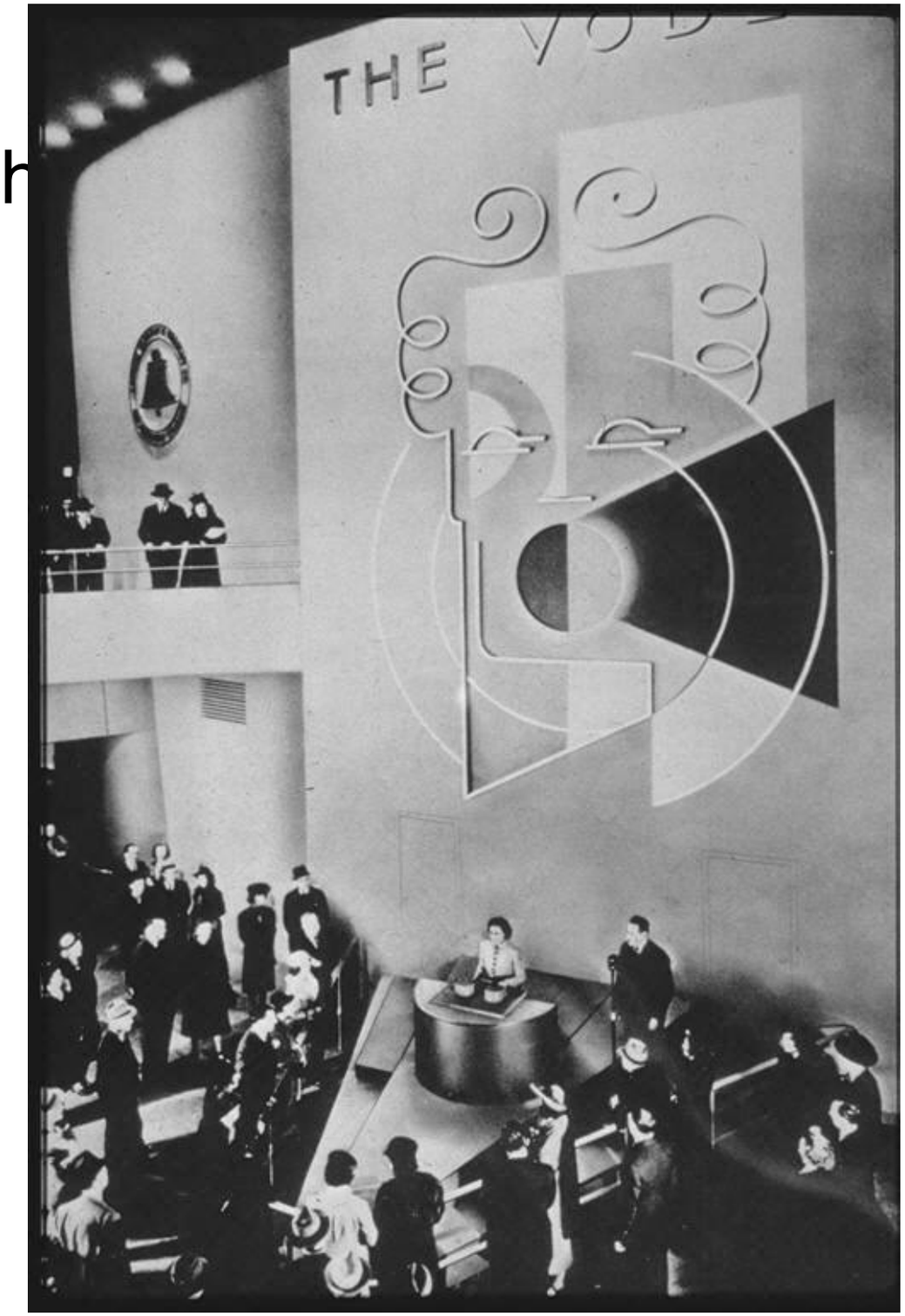
- Later, Wolfgang von Kempelen and Charles Wheatstone created a more sophisticated mechanical speech device…
  - with independently manipulable source and filter mechanisms.



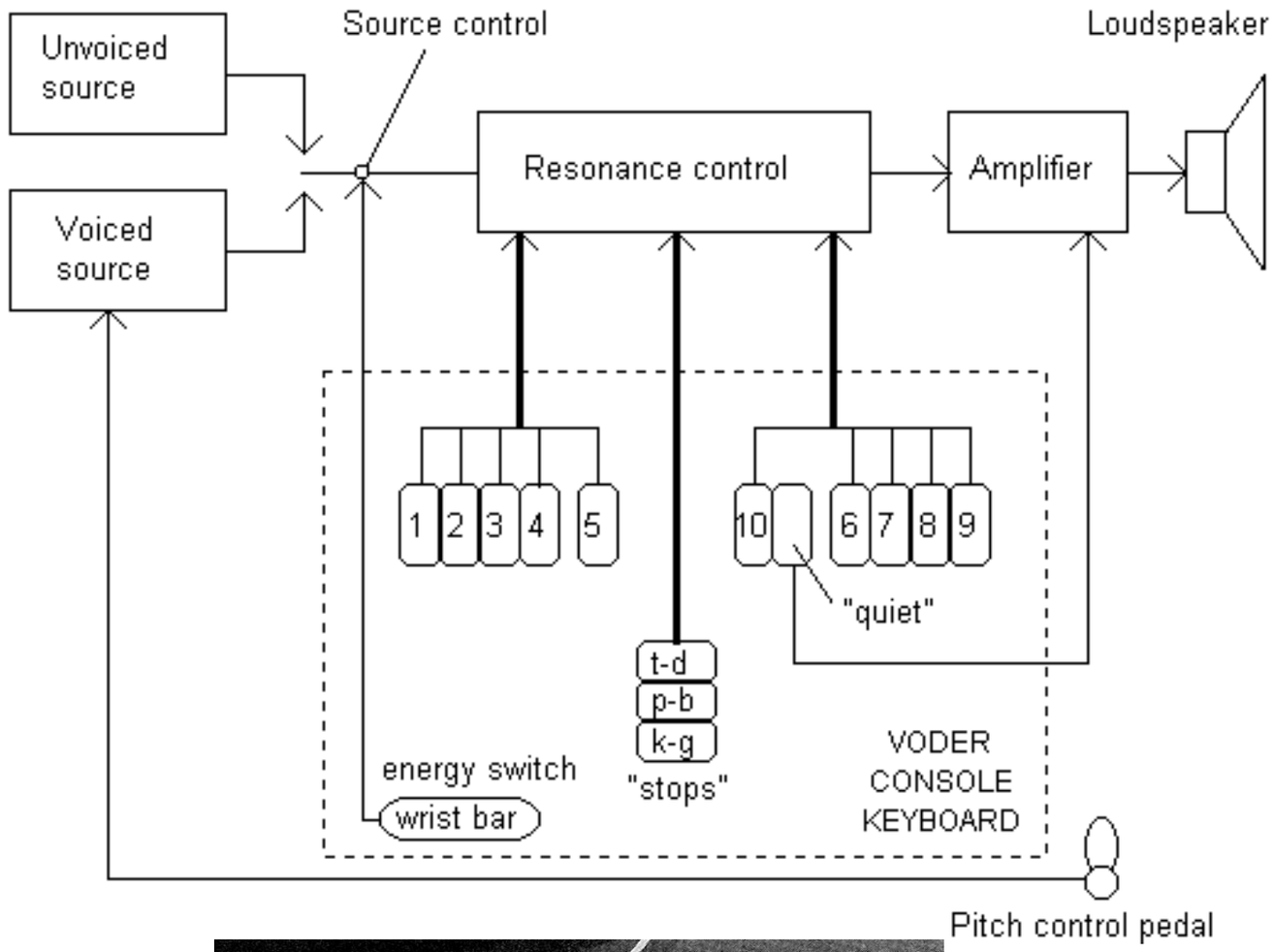


The Voder

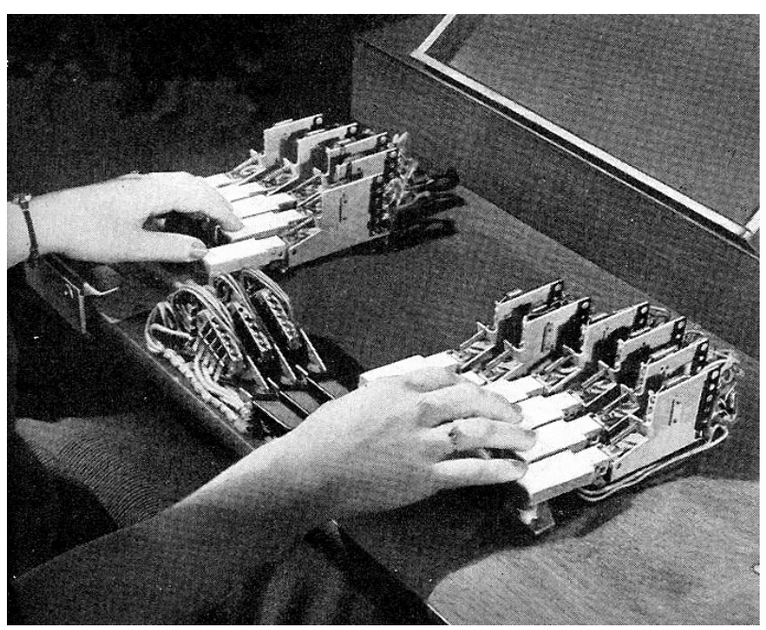
- The next big step in speech synthesis was to generate speech electronically.
- This was most famously demonstrated at the New York World's Fair in 1939 with the **Voder**.
- The Voder was a **manually controlled** speech synthesizer.
  - (operated by highly trained young women)



Voder Principles



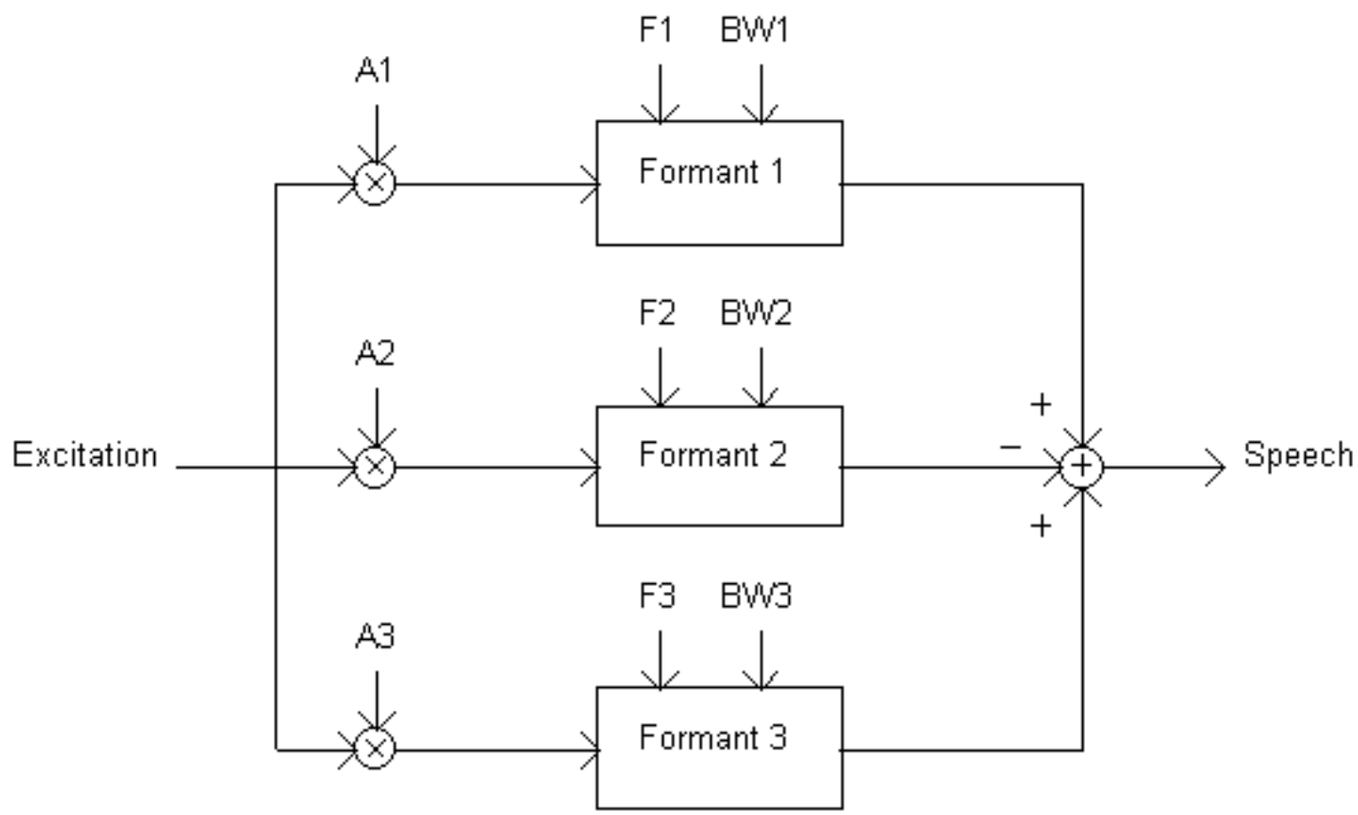
- The Voder basically operated like a **vocoder**.
- Voicing and fricative **source** sounds were filtered by 10 different resonators…
- each controlled by an individual finger!



- Only about 1 in 10 had the ability to learn how to play the Voder.

2. Formant Synthesis

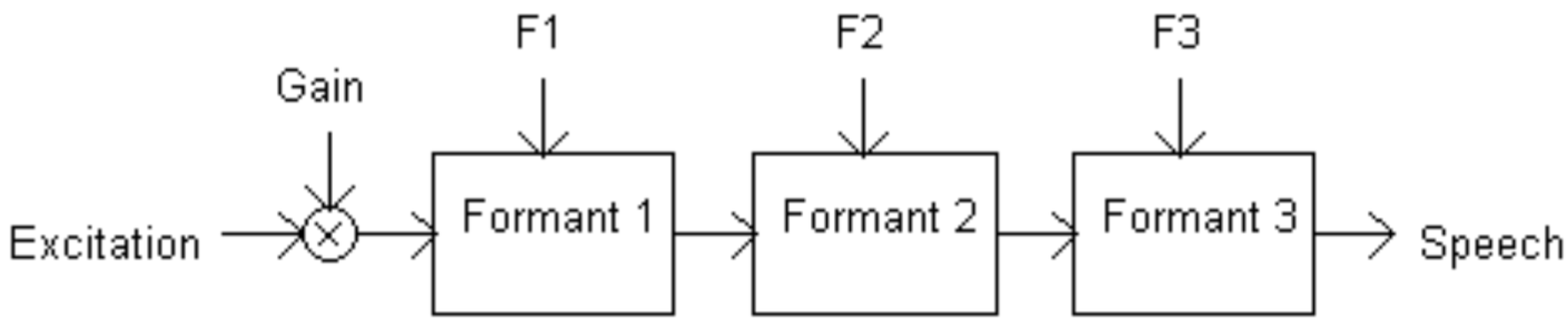
- The next synthesizer was PAT (Parametric Artificial Talker).
- PAT was a parallel **formant** synthesizer.
- Idea: three formants are good enough for intelligble speech.



- Subtitles: What did you say before that? Tea or coffee? What have you done with it?

2. Formant Synthesis, part II

- Another formant synthesizer was OVE, built by the Swedish phonetician Gunnar Fant.
- OVE was a **cascade** formant synthesizer.



- In the ‘50s and ‘60s, people debated whether parallel or cascade synthesis was better.
- Weeks and weeks of tuning each system could get much better results:

Synthesis by rule

- The ultimate goal was to get machines to generate speech automatically, without any manual intervention.
  - synthesis by rule
- A first attempt, on the Pattern Playback:

(I painted this by rule without looking at a spectrogram. Can you understand it?)

- Later, from 1961, on a cascade synthesizer:
  - Note: first use of a computer to calculate rules for synthetic speech.
- Compare with the HAL 9000:

LPC Synthesis

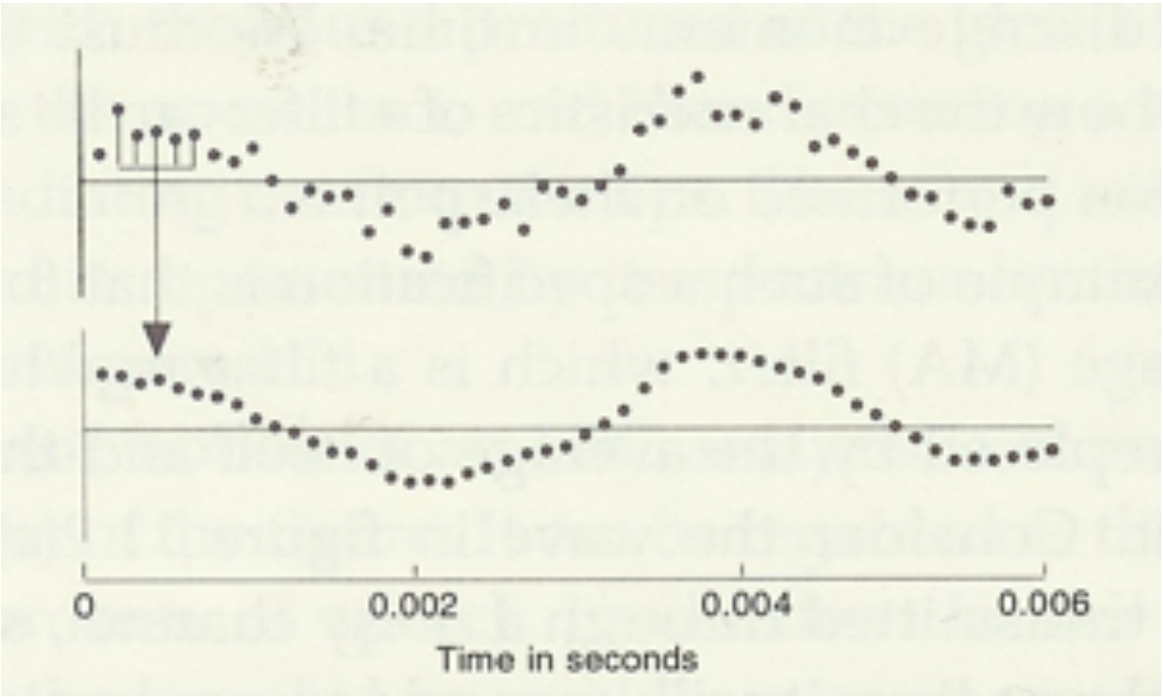
- Another method of formant synthesis, developed in the '70s, is known as **Linear Predictive Coding** (LPC).
- Here's an example:

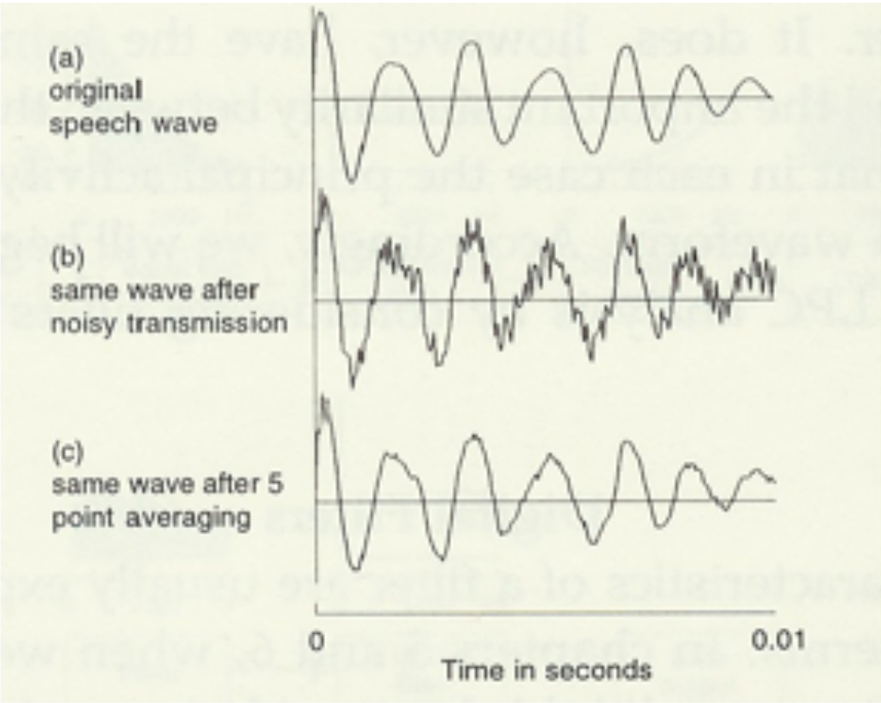
- As a general rule, LPC synthesis is pretty lousy.
  - But it's cheap!
- LPC synthesis greatly reduces the amount of information in speech…

- To recapitulate childhood: http://www.speaknspell.co.uk/

Filters + LPC

- One way to understand LPC analysis is to think about a **moving average filter**.
- A moving average filter reduces noise in a signal by making each point equal to the average of the points surrounding it.

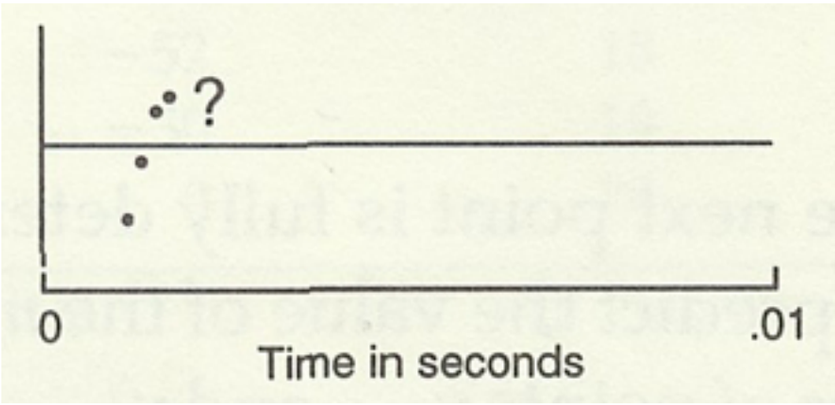


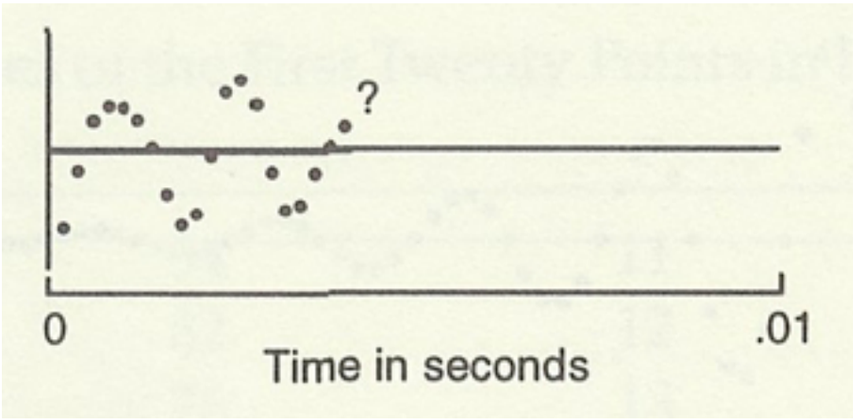


$$y_n = (x_{n-2} + x_{n-1} + x_n + x_{n+1} + x_{n+2}) / 5$$

Filters + LPC

- Another way to write the smoothing equation is
  - $y_n = .2*x_{n-2} + .2*x_{n-1} + .2*x_n + .2*x_{n+1} + .2*x_{n+2}$
- Note that we could **weight** the different parts of the equation differently.
  - Ex: $y_n = .1*x_{n-2} + .2*x_{n-1} + .4*x_n + .2*x_{n+1} + .1*x_{n+2}$
- Another trick: try to predict **future** points in the waveform on the basis of only previous points.





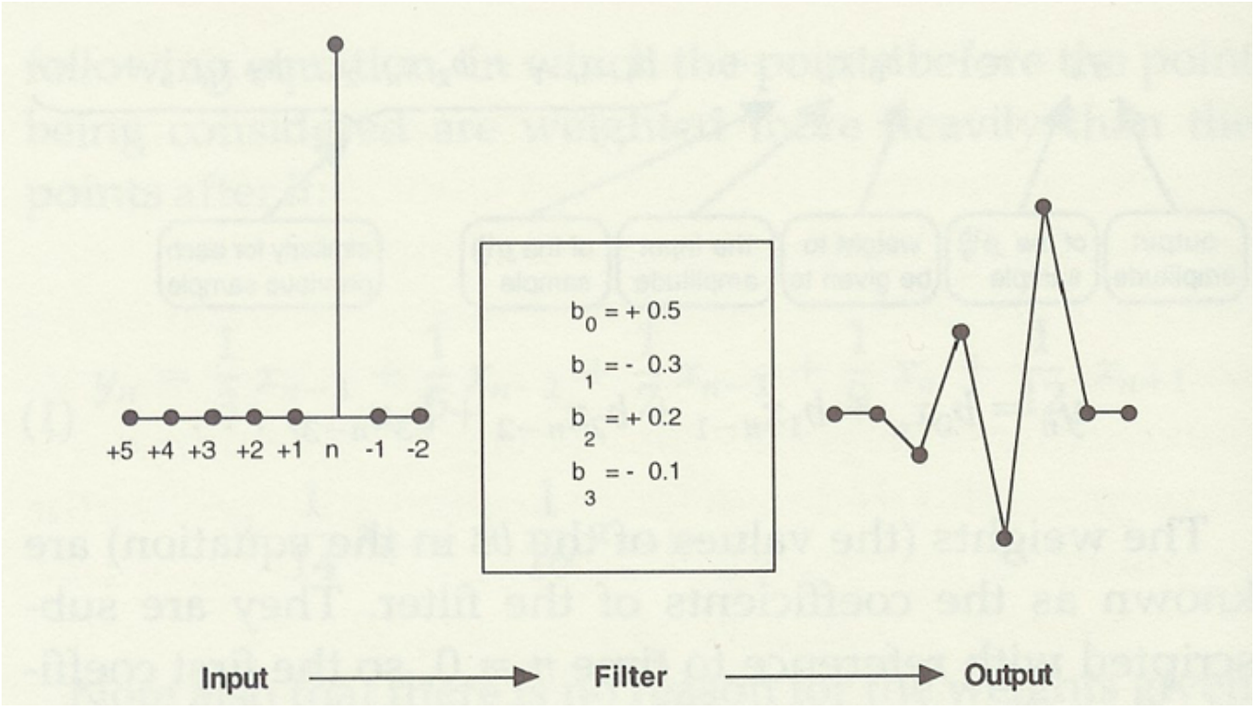
- Objective: find the combination of weights that predicts future points as perfectly as possible.

Deriving the Filter

- Let's say that minimizing the prediction errors for a certain waveform yields the following equation:
  - $y_n = .5*x_n - .3*x_{n-1} + .2*x_{n-2} - .1*x_{n-3}$
- The weights in the equation define a **filter**.
- Example: how would the values of y change if the input to the equation was a **transient** where:
  - at time n, x = 1
  - at all other times, x = 0
- Graph y at times n to n+3.

Decomposing the Filter

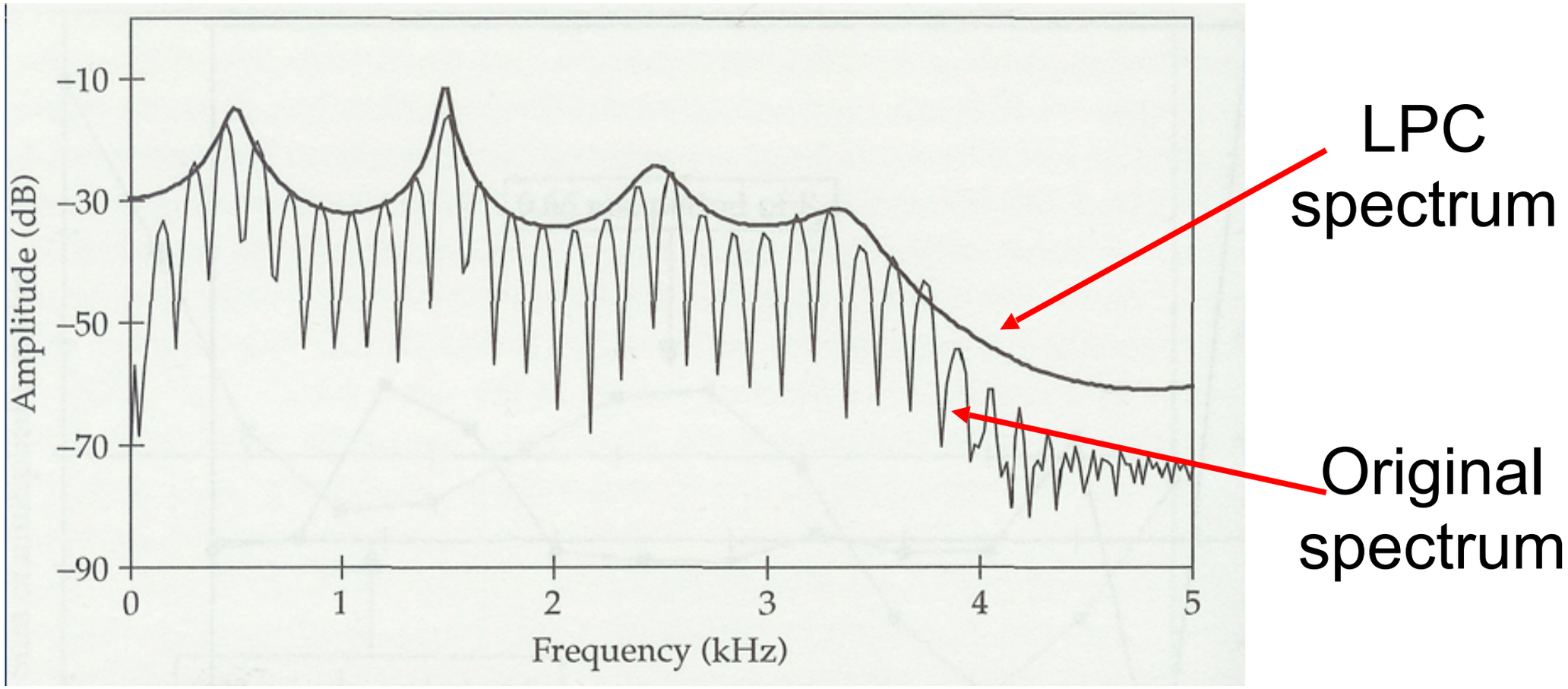
- Putting a transient into the weighted filter equation yields a new waveform:



- The new equation reflects the weights in the equation.
- We can apply Fourier Analysis to the new waveform to determine its spectral characteristics.

LPC Spectrum

- When we perform a Fourier Analysis on this waveform, we get a very smooth-looking spectrum function:



- This function is a good representation of what the vocal tract **filter** looks like.

LPC Applications

- Remember: the LPC spectrum is derived from the weights of a linear predictive equation.
- One thing we can do with the LPC-derived spectrum is estimate formant frequencies of a filter.
  - (This is how Praat does it)
- Note: the more weights in the original equation, the more formants are assumed to be in the signal.
- We can also use that LPC-derived filter, in conjunction with a voice source, to create synthetic speech.
  - (Like in the Speak & Spell)

3. Concatenative Synthesis

- Formant synthesis dominated the synthetic speech world up until the ‘90s…
  - Then **concatenative synthesis** started taking over.
- Basic idea: string together recorded samples of natural speech.
- Most common option: “diphone” synthesis
  - Concatenated bits stretch from the middle of one phoneme to the middle of the next phoneme.
- Note: inventory has to include all possible phoneme sequences
  - = only possible with lots of computer memory.

Concatenated Samples

- Concatenated synthesis tends to sound more natural than formant synthesis.
  - (basically because of better voice quality)
- Early (1977) combination of LPC + diphone synthesis:

- LPC + demisyllable-sized chunks (1980):
- More recent efforts with the MBROLA synthesizer:

- Also check out the Macintalk Pro synthesizer!

Recent Developments

- Contemporary concatenative speech synthesizers use **variable unit selection**.
- Idea: record a huge database of speech…
  - And play back the largest unit of speech you can, whenever you can.
- Interesting development #2: synthetic voices tailored to particular speakers.
- Check it out:

4. Articulatory Synthesis

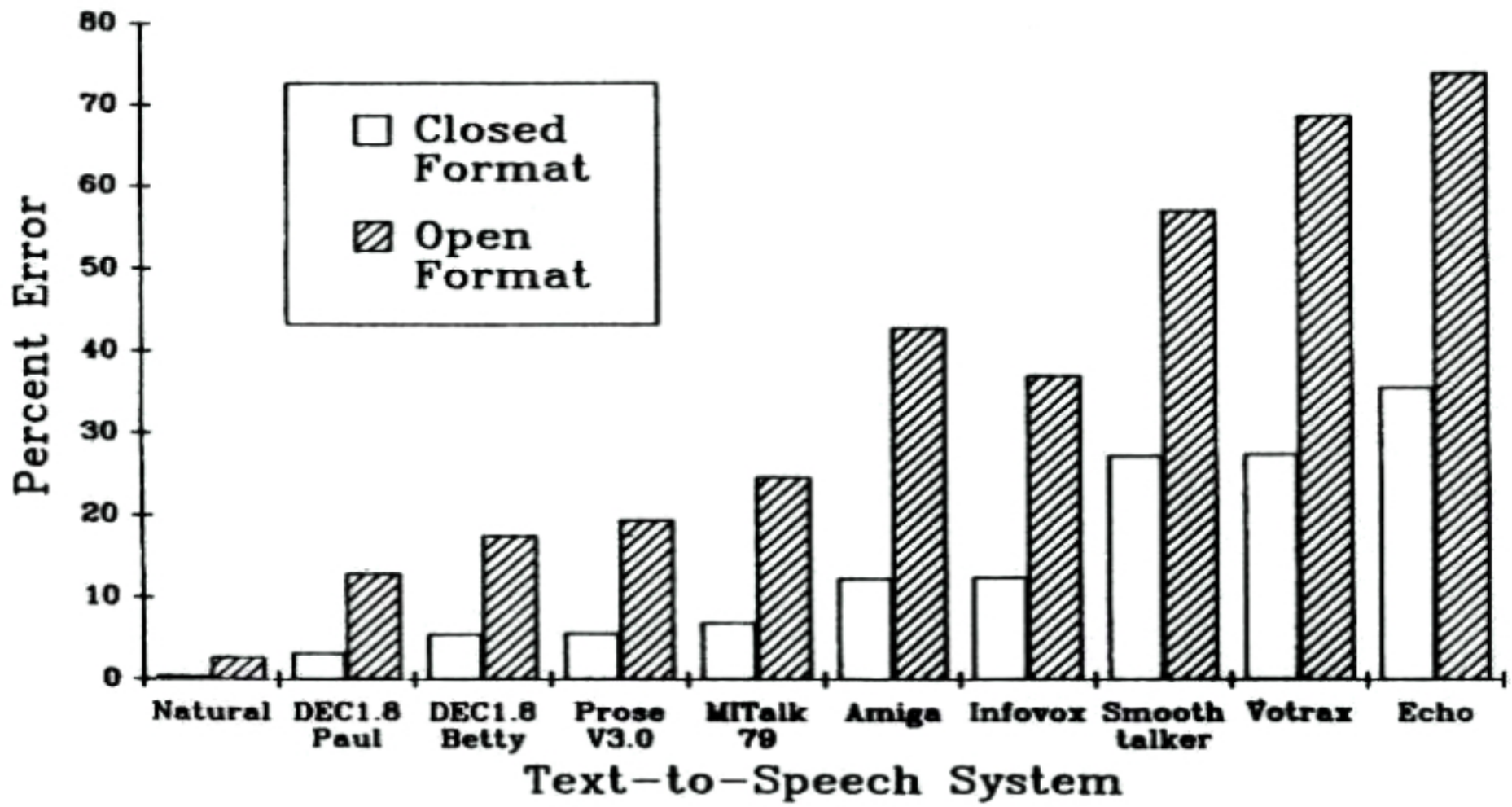
- Last but not least, there is articulatory synthesis.
  - Generation of acoustic signals on the basis of models of the vocal tract.
- This is the most complicated of all synthesis paradigms.
  - (we don’t understand articulations all that well)
- Some early attempts:
- Paul Boersma built his own articulatory synthesizer…
  - and incorporated it into Praat.

Synthetic Speech Perception

- In the early days, speech scientists thought that synthetic speech would lead to a form of “super speech”
  - = ideal speech, without any of the extraneous noise of natural productions.
- However, natural speech is <u>always</u> more intelligible than synthetic speech.
  - And more natural sounding!
- But: perceptual learning is possible.
  - Requires lots and lots of practice.
  - And lots of variability. (words, phonemes, contexts)
- An extreme example: blind listeners.

More Perceptual Findings

1. Reducing the number of possible messages dramatically increases intelligibility.



More Perceptual Findings

2. Formant synthesis produces better vowels;
   - Concatenative synthesis produces better consonants (and transitions)
3. Synthetic speech uses up more mental resources.
   - memory and recall of number lists
4. Synthetic speech perception is a lot easier for native speakers of a language.
   - And also adults.
5. Older listeners prefer slower rates of speech.