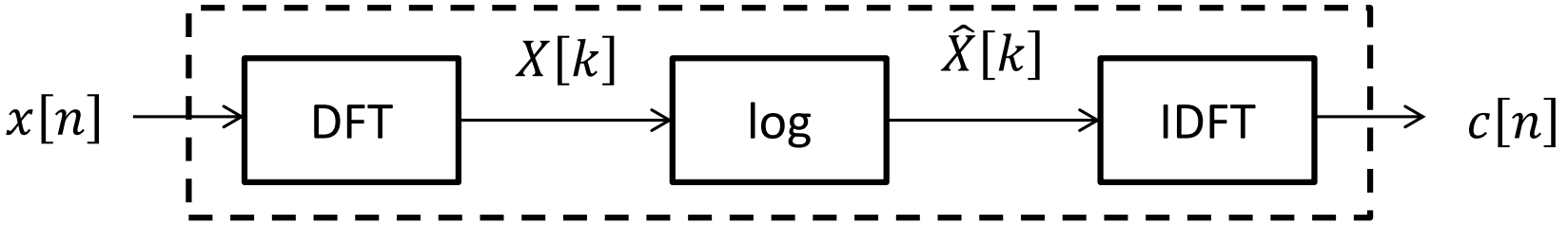
Week 4

Audio Analysis 3

Cepstrum

Inverse Fourier Transform of log magnitude spectrum of the signal!



Benefits of Cepstrum

- Log operation emphasizes periodicity of harmonics
- Cepstrum is useful in separating source and the filter
- Mainly used in speech and speaker recognition
- Cepstral coefficients are mostly uncorrelated

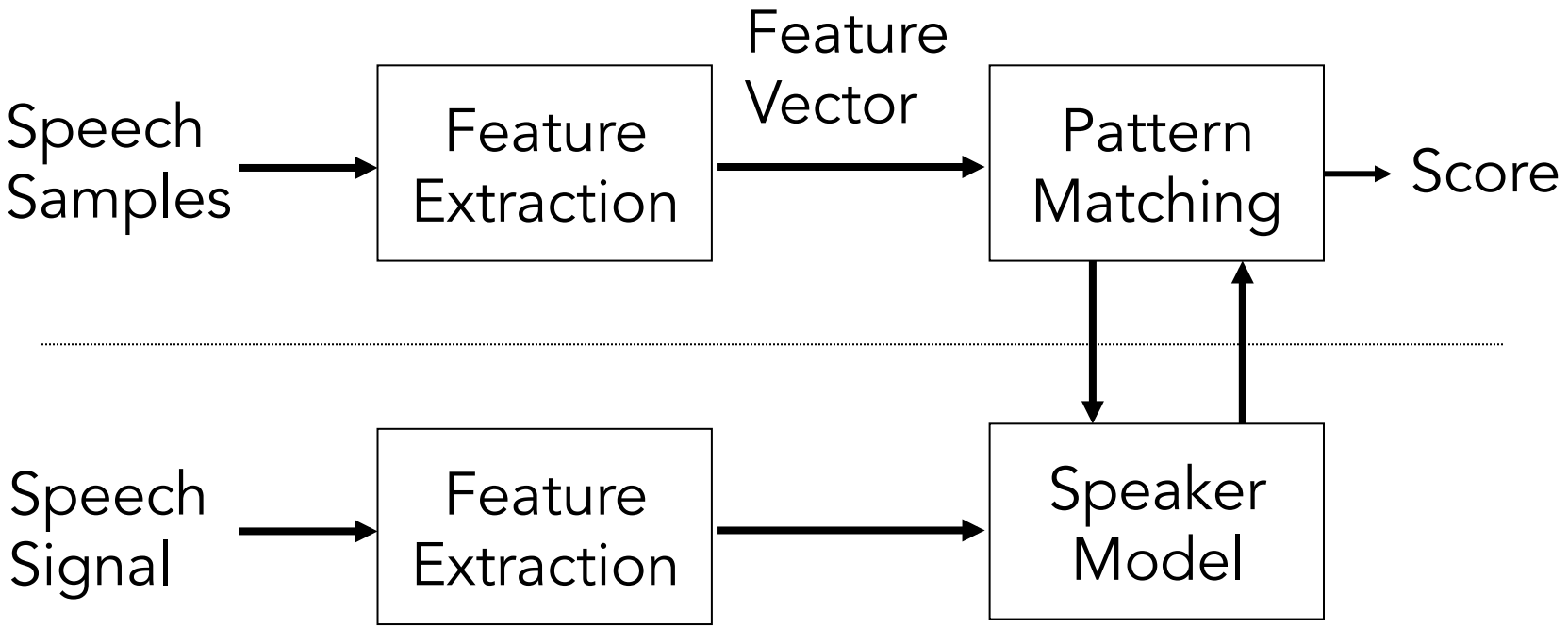
Further Audio Analysis

- Audio event detection
  - E.g. surveillance, sports events
- Movie content analysis
  - E.g. scene classification
- Music information retrieval
  - Tempo detection
  - Timbre detection
  - Rhythm analysis
  - Instrument detection
- Music genre detection
  - Rock, classic, Jazz
- Music source separation

Speech Audio Analysis

- Automatic speech recognition
  - Translating speech signal to text
- Speaker identification
  - Who is talking?
- Speaker verification
  - Is this the correct speaker?
- Speaker diarization
  - Who spoke when?
- Speech emotion recognition
  - Predict speaker's emotional state

Typical Speaker Recognition System



Speaker Models

- Text dependent
  - Speaker model is trained for particular text, e.g. Khul Ja Sim Sim
- Text Independent
  - No constraint of speech content

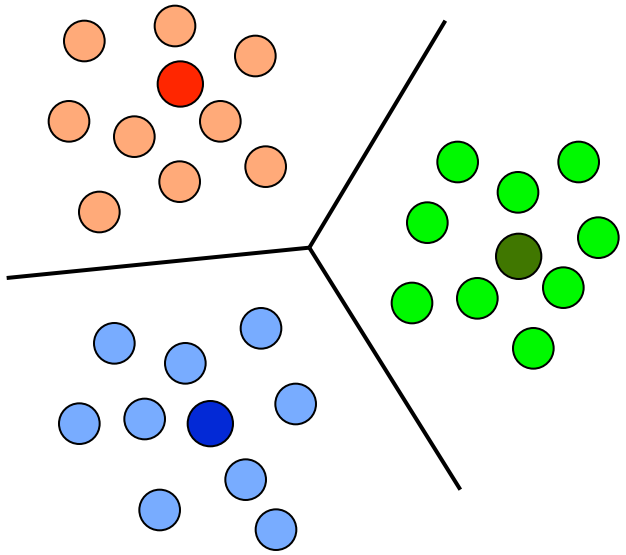
Features

- Cepstrum
- LPCC
- MFCC

Popular Speaker Models

- Vector Quantization (VQ)
- Gaussian Mixture Model (GMM)
- Hidden Markov Model (HMM)

Vector Quantization



10000 feature vectors to 16 codewords!

Linde–Buzo–Gray codebook generation

1. Guess the cluster centroids $C=\{c_1,c_2,\ldots,c_K\}$;
2. REPEAT

   - For each training vector x_j , find the nearest cluster centroid : $q_j = \arg\min_k \|x_j - c_k\|$

   - For each cluster k, re-calculate the cluster centroid from the vectors assigned to the cluster: $c_k = \text{mean } \{x_j | q_j=k\}$

   - UNTIL convergence

Obtaining Codebook

Input vectors: $S = \{\mathbf{x}_i \in R^d | \ i = 1, 2, \ \cdots, n\}$

Initial centroids: $C = \{\mathbf{c}_j \in R^d | \ j = 1, 2, \ \cdots, K\}$

Obtain clusters: $\mathbf{x}_i \in S_q \text{ if } \|\mathbf{x}_i - \mathbf{c}_q\|_p \ \leq \ \|\mathbf{x}_i - \mathbf{c}_j\|_p$

Update centroids: $\mathbf{c}_j = \frac{1}{|S_j|} \sum_{\mathbf{x}_i \in S_j} \mathbf{x}_i$

Calculate distortion: $D_k = \sum_{j=1}^{K} \sum_{\mathbf{x}_i \in S_j} \|\mathbf{x}_i - \mathbf{c}_j\|_p$

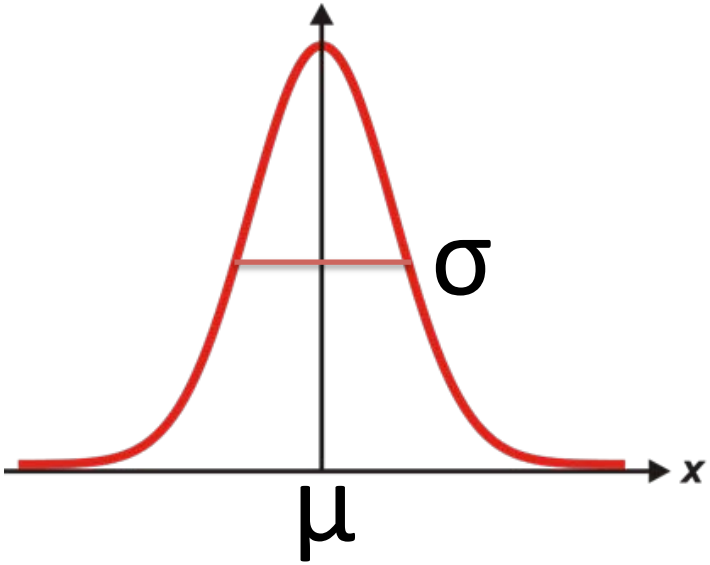
Repeat until distortion < threshold

The codebook: $C = \{\mathbf{c}_j \in R^d | \ j = 1, 2, \ \cdots, K\}$

Classification

- Generate speaker 1 codebook
- Generate speaker 2 codebook
- Take the input vectors and "quantize" using the codebooks
- The codebook with smaller distortion (distance from centroid) wins

Gaussian Mixture Model



Steps:

- Build one GMM for each speaker
- Calculate probability of test feature vector given each GMM
- The highest probability GMM wins

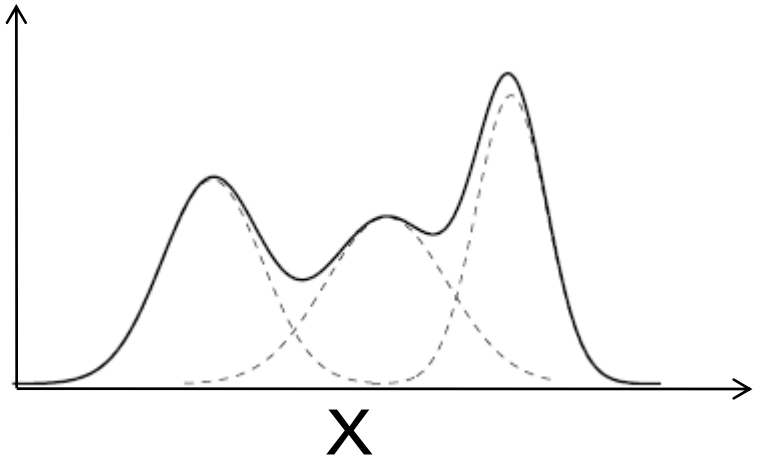
Let’s take a single feature

(5, 3, 7, 9, 2, 5, 3, 5, 4)

Soln1: Obtain μ and **σ** of the whole dataset!

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \mathrm{e}^{\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}}$$

Problem: the features may have multiple peaks!



Soln2: Obtain μ and **σ** for each peak!

$$p(x) = \sum_{j=1}^{M} w_j p_j(x)$$

where

$$p_j(x) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \mathrm{e}^{\left\{-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right\}}$$

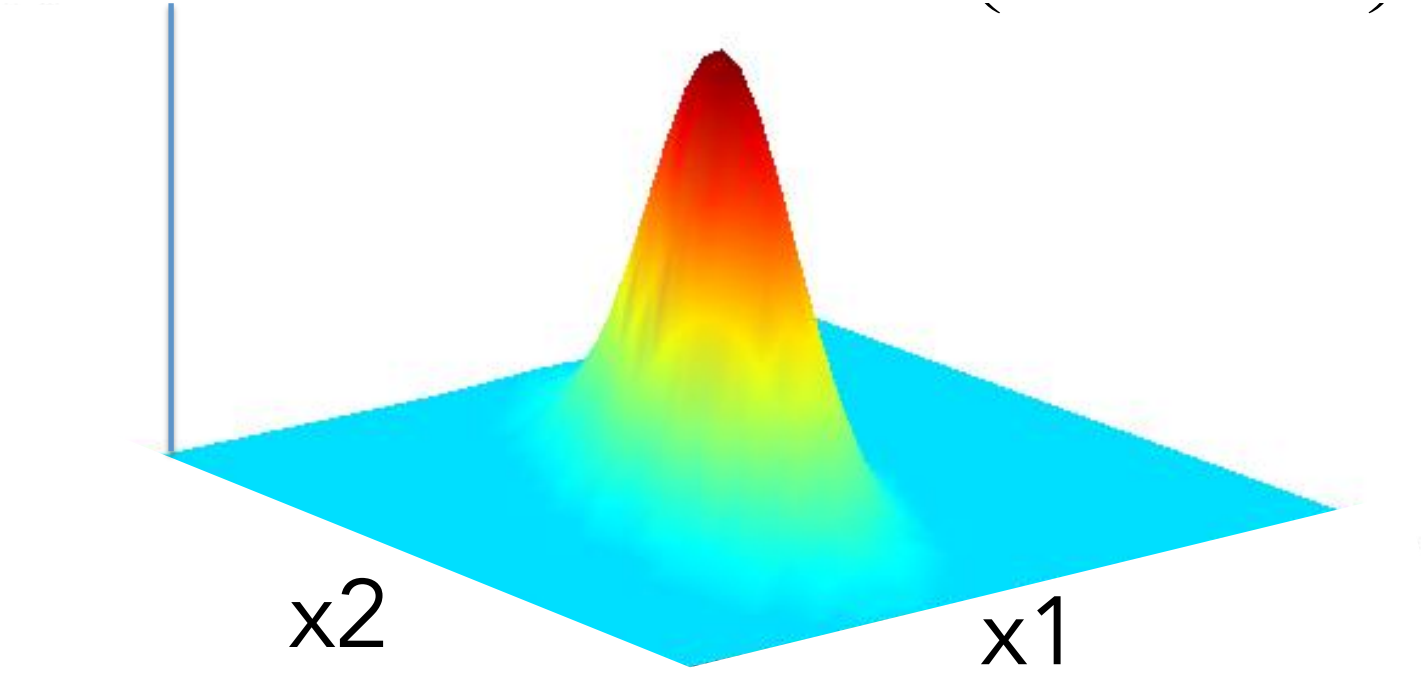
$$\sum_{j=1}^{M} w_j = 1$$

Problem: we have multiple features!

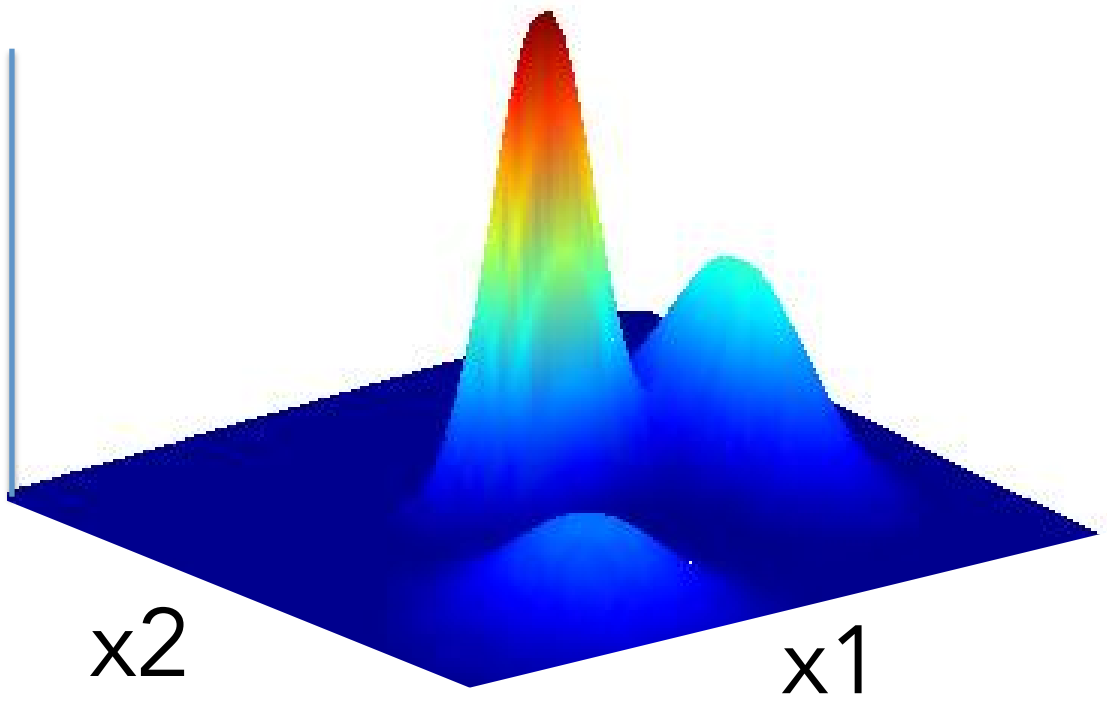
$$\left(\begin{bmatrix}4\\2\\3\\5\\2\\9\end{bmatrix}\begin{bmatrix}4\\2\\5\\7\\4\\3\end{bmatrix}\begin{bmatrix}6\\5\\2\\8\\4\\6\end{bmatrix}\begin{bmatrix}4\\5\\2\\6\\7\\4\end{bmatrix}\begin{bmatrix}3\\4\\5\\2\\7\\5\end{bmatrix}\begin{bmatrix}6\\5\\2\\8\\4\\6\end{bmatrix}\begin{bmatrix}4\\5\\2\\6\\7\\4\end{bmatrix}\begin{bmatrix}3\\4\\5\\2\\7\\5\end{bmatrix}\begin{bmatrix}6\\5\\2\\8\\4\\6\end{bmatrix}\begin{bmatrix}4\\5\\2\\6\\7\\4\end{bmatrix}\begin{bmatrix}3\\4\\5\\2\\7\\5\end{bmatrix}\right)$$

Soln: Use multi-variate Gaussian

2D-Gaussian (1 peak)



2D-Gaussian (3 peaks)



Multi-variate Gaussian

$x \Longrightarrow \vec{x}$ Feature vector

$\mu \Longrightarrow \vec{\mu}$ Mean vector

σ^2 equivalent $\sum$ Covariance matrix

Probability of a Feature Vector

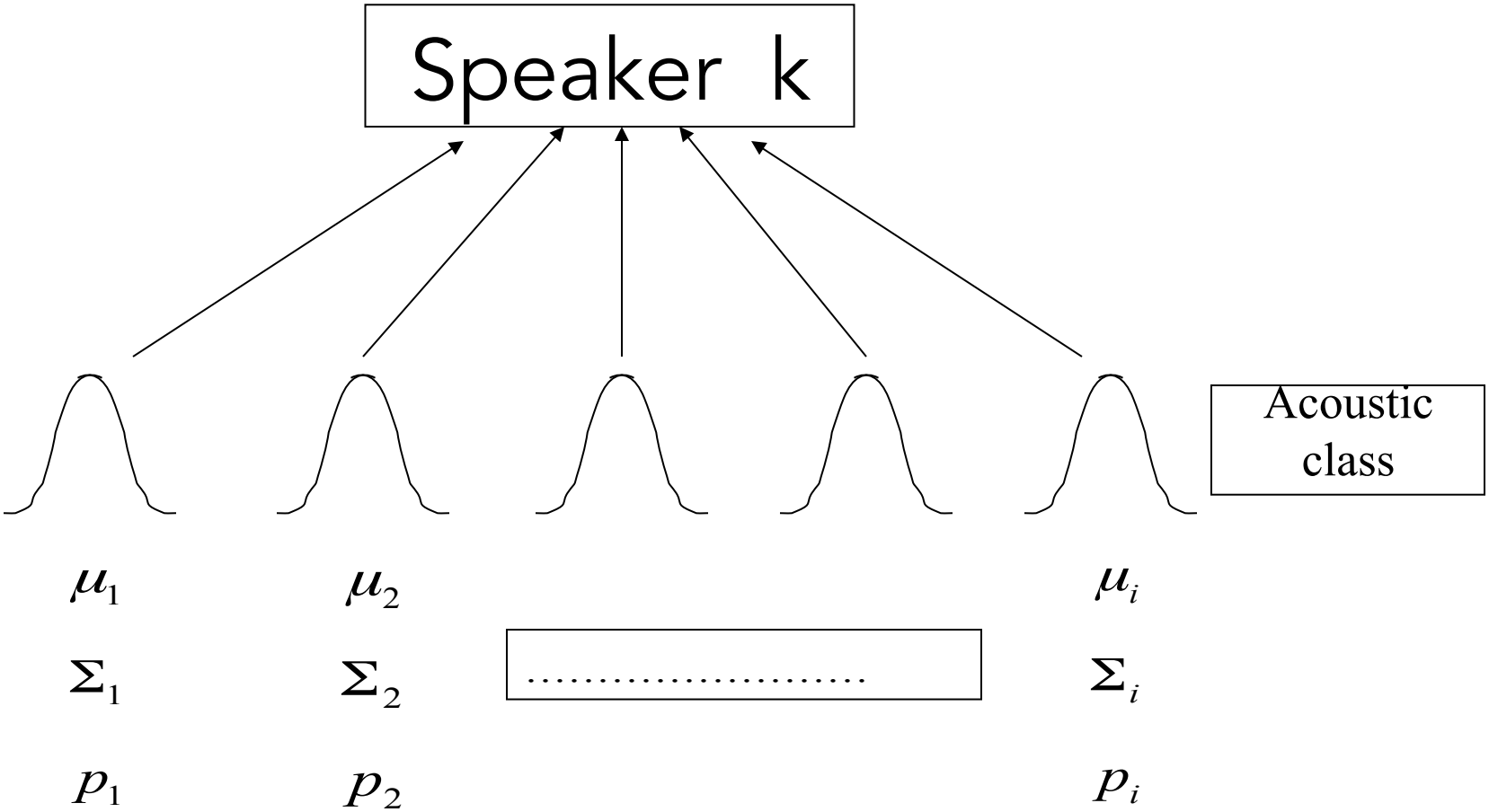
$$\vec{x} = \{x_1, x_2 \ldots x_D\}$$

$$p_j(\vec{x}) = \frac{1}{(2\pi)^{D/2} \left|\Sigma_j\right|^{1/2}} \exp\left\{-\frac{1}{2}(\vec{x} - \vec{\mu}_j)' \Sigma_j^{-1} (\vec{x} - \vec{\mu}_j)\right\}$$

$$p(\vec{x} / S) = \sum_{j=1}^{M} w_j p_j(\vec{x})$$

$$S = \{w_j, \mu_j, \Sigma_j\} \quad \text{where } j = 1, 2, 3 \ldots M$$

Each Gaussian component models an acoustic class



Speaker Identification

A group of speakers S = {1,2,…,S} is represented by GMM's $\lambda_1, \lambda_2, \ldots, \lambda_S$

$$\hat{S} = \arg\max_{1 \le k \le S} \Pr(\lambda_k \mid \vec{x}) = \arg\max_{1 \le k \le S} \frac{p(\vec{x} \mid \lambda_k)\Pr(\lambda_k)}{p(\vec{x})}$$

$$\hat{S} = \arg\max_{1 \le k \le S} p(\vec{x} \mid \lambda_k) \xrightarrow{take\log} \hat{S} = \arg\max_{1 \le k \le S} \sum_{j=1}^{M} \log p_j(\vec{x} \mid \lambda_k)$$

$$p(\vec{x} \mid \lambda_k) = \sum_{j=1}^{M} w_j p_j(\vec{x})$$

Problem

- You are at at a party in a club
- You hear an audio signal (e.g., a song)
- You want to quickly know more about it

How do you say two audio objects are same or similar?

Solution

1. Record an excerpt of audio
2. Match with your database
3. Get the information

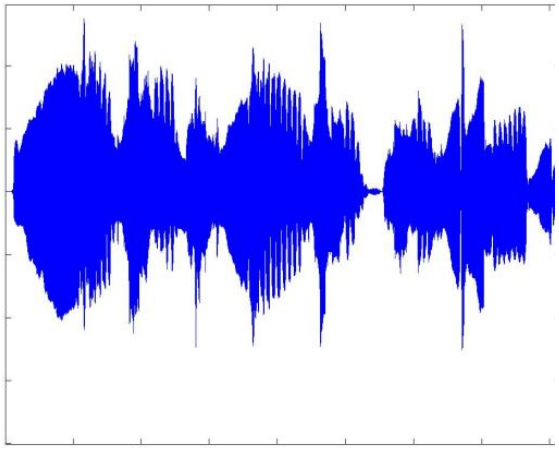
Problem

- You cannot compare songs directly
- Inefficient
- Not robust

Obtain fingerprint of the audio and only match fingerprints!

Audio Fingerprinting

- Short summary of audio object using a limited number of bits



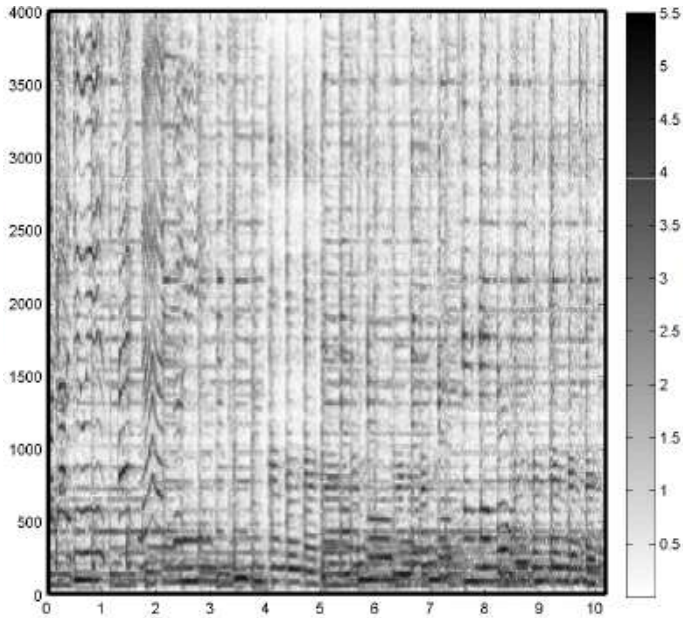
0
1
0
0
1
1
1
0

Fingerprinting Requirements

1. The fingerprint should be compact
2. It should be discriminative
3. It should be robust to usual audio degradations

Shazam uses fingerprinting work by Avery Wang!

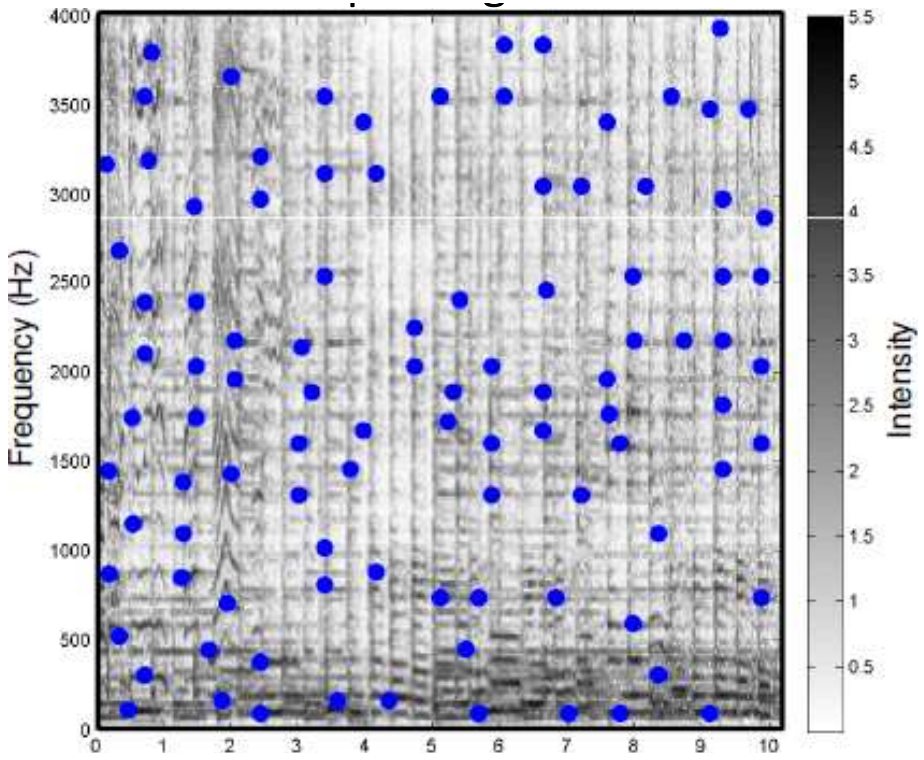
Transform audio to spectrogram



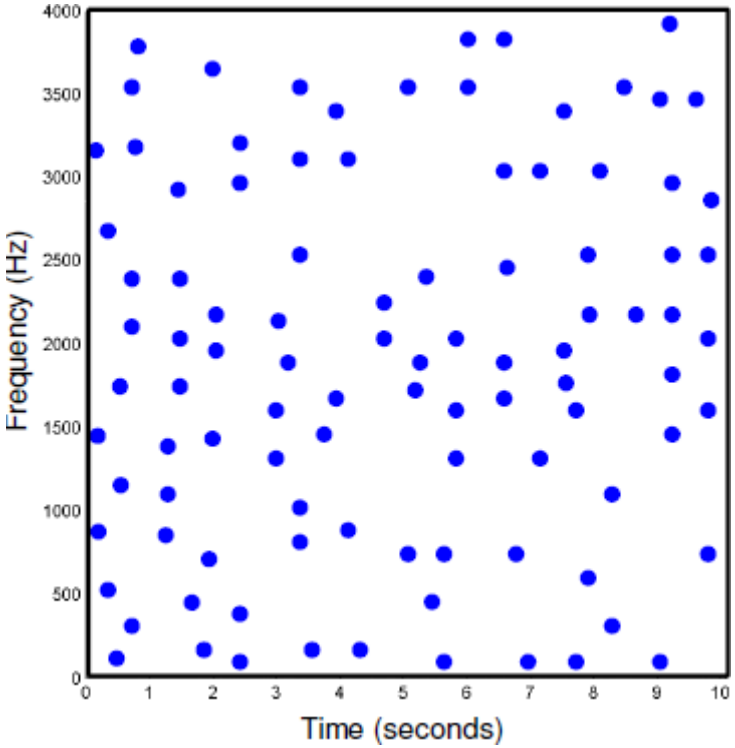
Find peaks in the spectrogram using some criteria!

- Density
- Energy difference

Finding Peaks

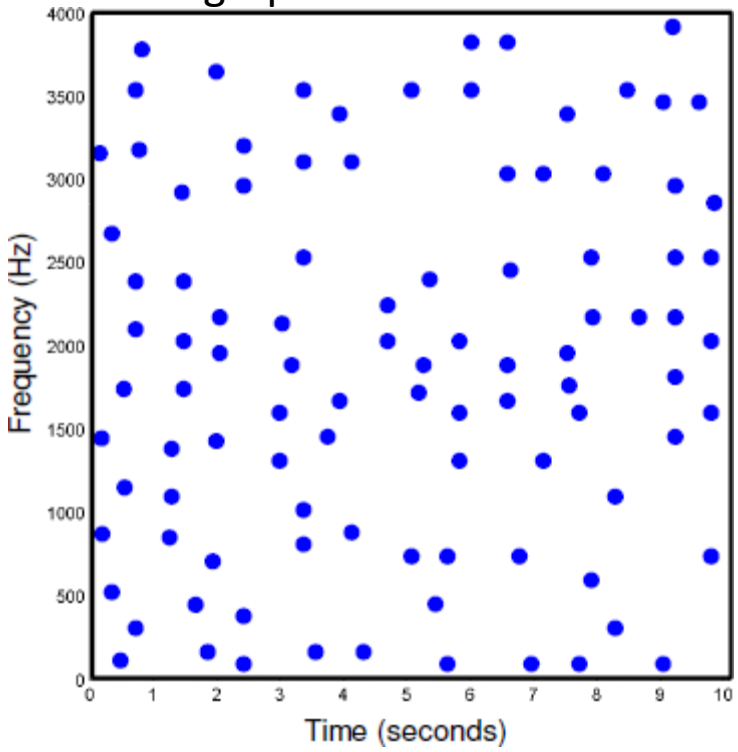


Audio Fingerprint

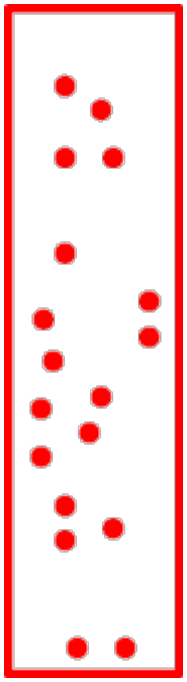


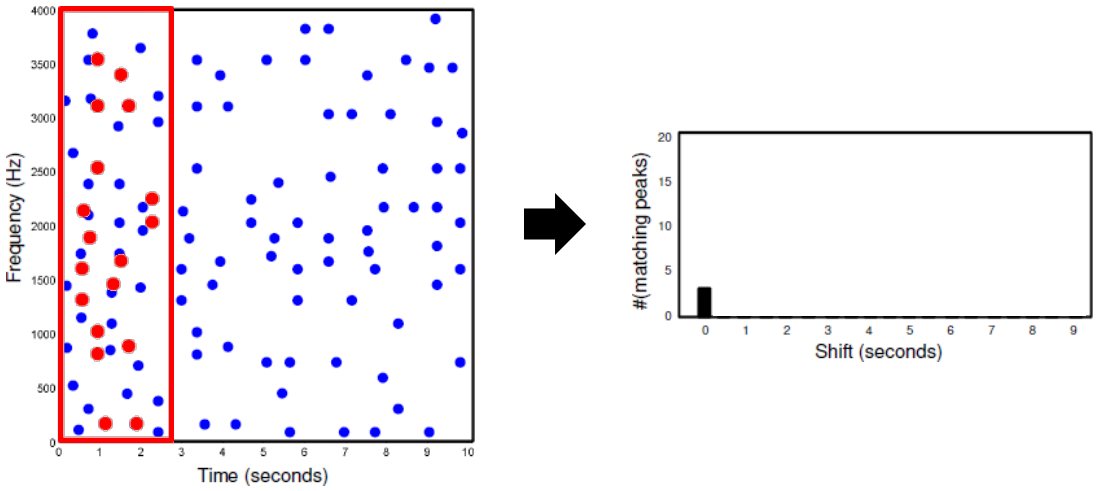
Fingerprint Matching

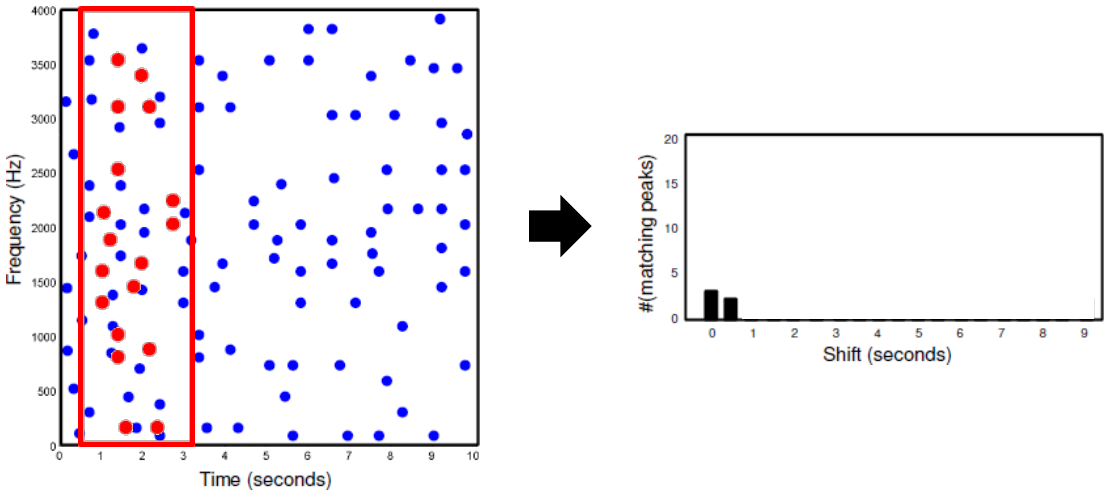
Fingerprint of a reference

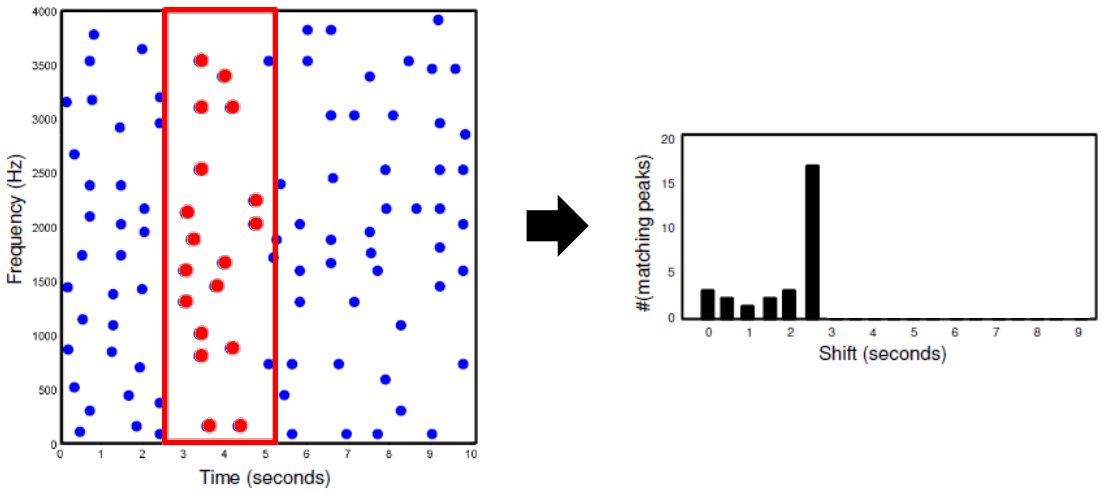


Fingerprint of the query

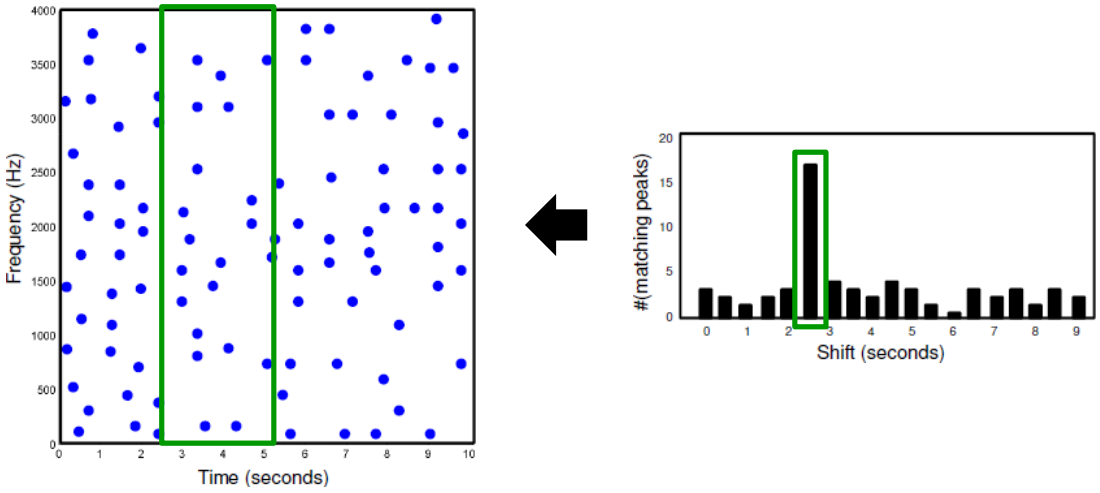



Frequency (Hz)
4000
3500
3000
2500
2000
1500
1000
500
0
0 1 2 3 4 5 6 7 8 9 10
Time (seconds)
#(matching peaks)
20
15
10
5
0
0 1 2 3 4 5 6 7 8 9
Shift (seconds)


Frequency (Hz)
4000
3500
3000
2500
2000
1500
1000
500
0
0 1 2 3 4 5 6 7 8 9 10
Time (seconds)
#(matching peaks)
20
15
10
5
0
0 1 2 3 4 5 6 7 8 9
Shift (seconds)


Frequency (Hz)
Time (seconds)
#(matching peaks)
Shift (seconds)

A high count indicates match!



More fingerprinting

- Divide into frames (11.6ms)
- Obtain a vector for a set of frames (256)
- Obtain bit vector for each frame (32 bits)
- Compare the bit pattern with reference using Hamming distance