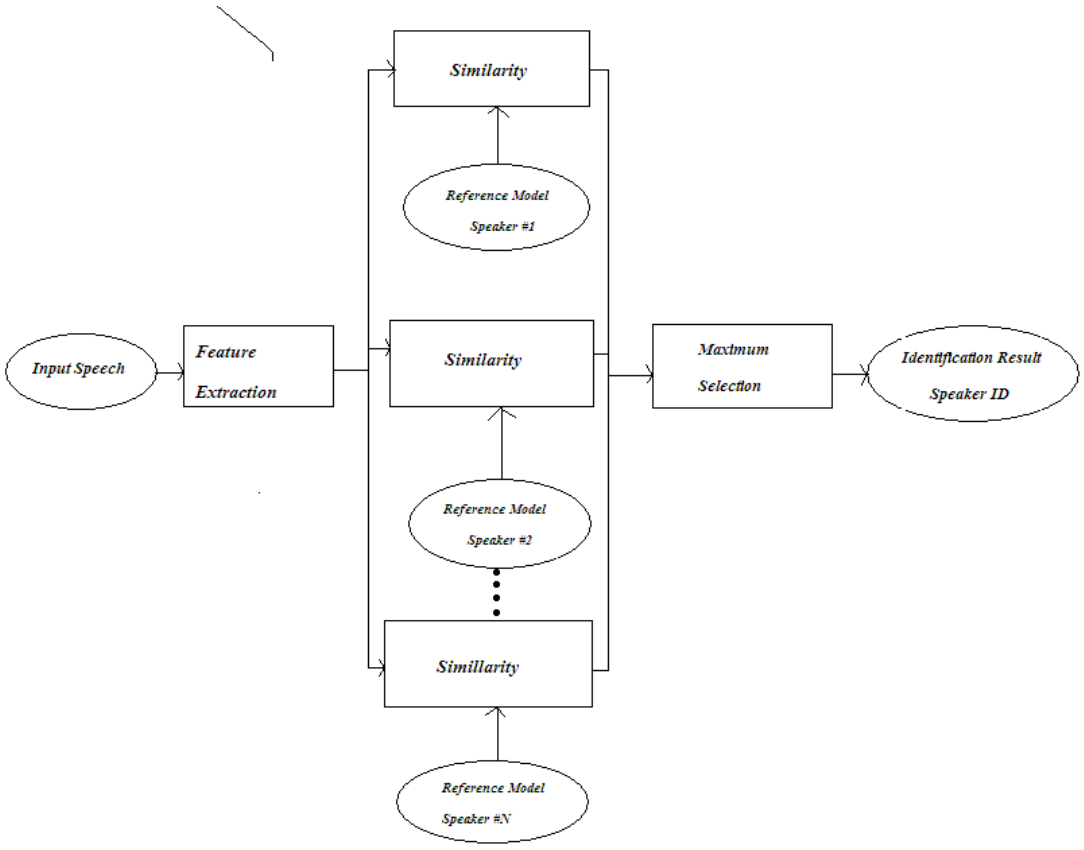
# Speaker Recognition
## Project Report

## Introduction:

Speaker recognition is basically divided into two-classification: speaker recognition and speaker identification and it is the method of automatically identify who is speaking on the basis of individual information integrated in speech waves. Speaker recognition is widely applicable in use of speaker's voice to verify their identity and control access to services such as banking by telephone, database access services, voice dialling telephone shopping, information services, voice mail, security control for secret information areas, and remote access to computer AT and T and TI with Sprint have started field tests and actual application of speaker recognition technology; many customers are already being used by Sprint's Voice Phone Card. Speaker recognition technology is the most potential technology to create new services that will make our every day lives more secured. Another important application of speaker recognition technology is for forensic purposes. Speaker recognition has been seen an appealing research field for the last decades which still yields a number of unsolved problems.

The main aim of this project is speaker identification, which consists of comparing a speech signal from an unknown speaker to a database of known speaker. The system can recognize the speaker, which has been trained with a number of speakers. Below figure shows the fundamental formation of speaker identification and verification systems. Where the speaker identification is the process of determining which registered speaker provides a given speech. On the other hand, speaker verification is the process of rejecting or accepting the identity claim of a speaker. In most of the applications, voice is use as the key to confirm the identities of a speaker are classified as speaker verification.




Adding the open set identification case in which a reference model for an unknown speaker may not exist can also modify above formation of speaker identification and verification system. This is usually the case in forensic application. In this circumstances, an added decision alternative, *the unknown does not match any of the models*, is required. Other threshold examination can be used in both verification and identification process to decide if the match is close enough to acknowledge the decision or if more speech data are needed.

Speaker recognition can also divide into two methods, text- dependent and text independent methods. In text dependent method the speaker to say key words or sentences having the same text for both training and recognition trials. Whereas in the text independent does not rely on a specific text being speak. Formerly text dependent

methods were widely in application, but later text independent is in use. Both text dependent and text independent methods share a problem however.

By playing back the recorded voice of registered speakers this system can be easily deceived. There are different technique is used to cope up with such problems. Such as a small set of words or digits are used as input and each user is provoked to thorough a specified sequence of key words that is randomly selected every time the system is used. Still this method is not completely reliable. This method can be deceived with the highly developed electronics recording system that can repeat secrete key words in a request order. Therefore T. Matsui and S. Furui have recently proposed the text dependent speaker recognition method

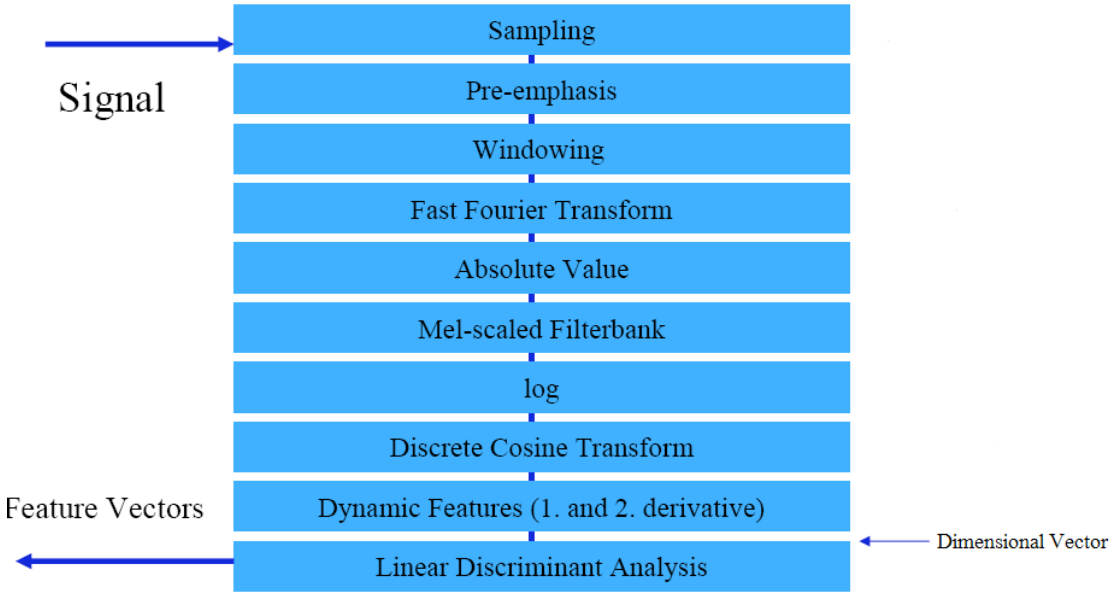
## Speech Feature Extraction:

In this project the most important thing is to extract the feature from the speech signal. The speech feature extraction in a categorization problem is about reducing the dimensionality of the input-vector while maintaining the discriminating power of the signal. As we know from the above fundamental formation of speaker identification and verification systems, that the number of training and test vector needed for the classification problem grows exponential with the dimension of the given input vector, so we need feature extraction.

But extracted feature should meet some criteria while dealing with the speech signal. Such as:

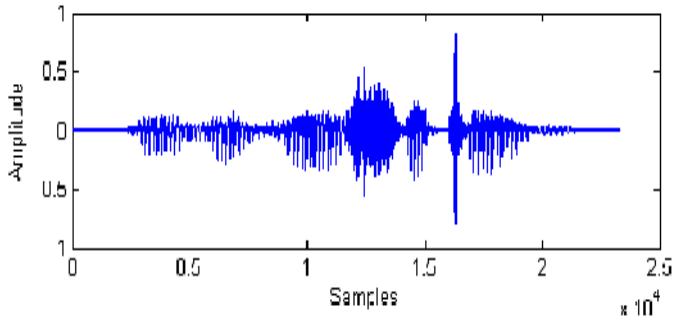
- Easy to measure extracted Speech features.
- Distinguish between speakers while being lenient of intra speaker variability's.
- It should not be susceptible to mimicry.
- It should show little fluctuation from one speaking environment to another.
- It should be stable over time.
- It should occur frequently and naturally in speech.

In this project we are using the Mel Frequency Cepstral Coefficients (MFCC) technique to extract features from the speech signal and compare the unknown speaker with the exist speaker in the database. Figure below shows the complete pipeline of Mel Frequency Cepstral Coefficients.




# Framing and Windowing:

As shown in the figure below the speech signal is slowly varying over time and it is called quasi stationery.




Above plot shows the word spoken by speaker. The recordings were digitised at f samples is equal to 11,025 samples per second and at 16 bits per sample. Time goes from left to right and amplitude is shown vertically. When the speech signal is examined over a short period of time such as 5 to 100 milliseconds, the signal is reasonably stationery, and therefore this signals are examine in short time segment, short time segments is referred to as a spectral analysis. This means that the signal is blocked into 20-30 milliseconds of each frame. And to avoid the loss of any information due to windowing adjacent frame is overlap with each other by 30

percent to 50 percent. As soon as the signal has been framed, each frame is multiplied with the window function *w(n)* with length N. The function below we are using is called hamming window function

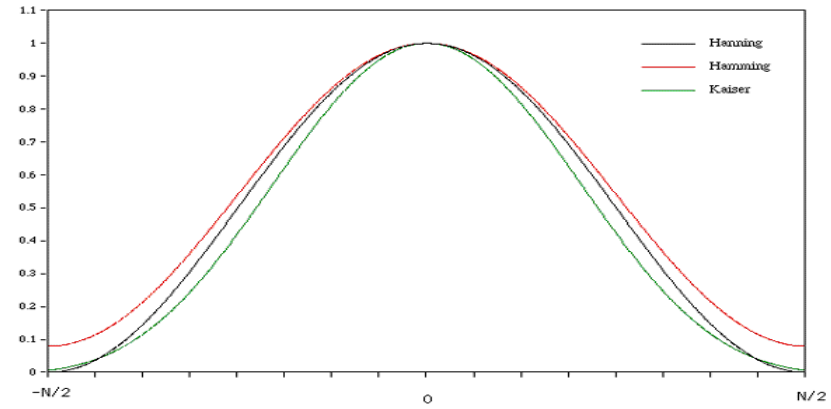
Where N = Length of the frame.

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

## Hamming Window:

Hamming window is also called the raised cosine window. The equation and plot for the Hamming window shown below. In a window function there is a zero valued outside of some chosen interval. For example, a function that is stable inside the interval and zero elsewhere is called a rectangular window, that illustrate the shape of its graphical representation. When signal or any other function is multiplied by a window function, the product is also zero-valued outside the interval. The windowing is done to avoid problems due to truncation of the signal. Window function has some other applications such as spectral analysis, filter design, and audio data compression such as Vorbis.

$$w(i) = 0.54 + 0.46 * \cos\left(\frac{2\pi i}{N}\right)$$




## Cepstrum:

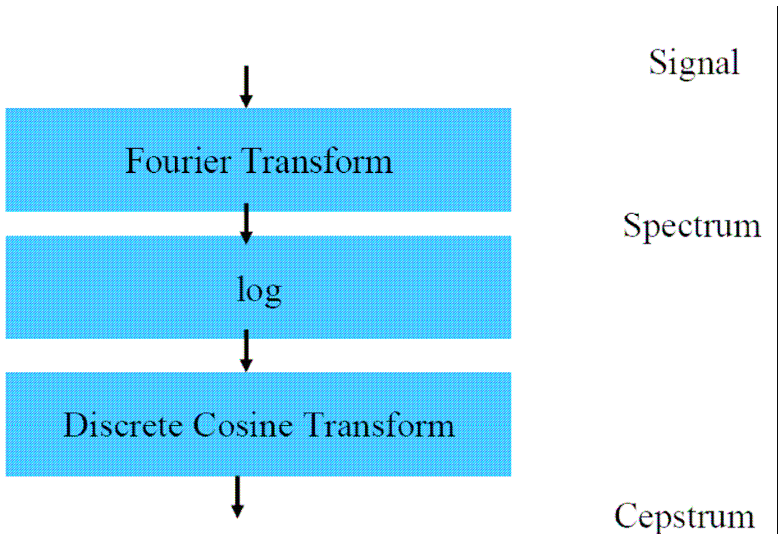
Cepstrum name was derived from the spectrum by reversing the first four letters of spectrum. We can say cepstrum is the Fourier Transformer of the log with unwrapped phase of the Fourier Transformer.

- Mathematically we can say Cepstrum of signal = FT(log(FT(the signal))+j2πm)

Where m is the interger required to properly unwrap the angle or imaginary part of the complex log function.

- Algorithmically we can say – Signal - FT - log - phase unwrapping - FT - Cepstrum.

For defining the real values real cepstrum uses the logarithm function. While for defining the complex values whereas the complex cepstrum uses the complex logarithm function. The real cepstrum uses the information of the magnitude of the spectrum.where as complex cepstrum holds information about both magnitude and phase of the initial spectrum, which allows the reconstruction of the signal. We can calculate the cepstrum by many ways. Some of them need a phase-warping algorithm, others do not. Figure below shows the pipeline from signal to cepstrum.




As we discussed in the Framing and Windowing section that speech signal is composed of quickly varying part e(n) excitation sequence convolved with slowly varying part $\theta$(n) vocal system impulse response.

$$s(n) = e(n) * \theta(n)$$

Once we convolved the quickly varying part and slowly varying part it makes difficult to separate the two parts, cepstrum is introduced to separate this two parts. The equation for the cepstrum is given below:

$$c_s(n) = \mathfrak{F}^{-1}\left\{log\left|\mathfrak{F}\{s(n)\}\right|\right\}$$

$\mathfrak{F}$ is the Discrete Time Fourier Transformer and $\mathfrak{F}^{-1}$ is the Inverse Discrete Time Fourier Transformer. By moving the signal from time domain to frequency domain convolution becomes the multiplication.

$$S(\omega) = E(\omega)\Theta(\omega)$$

The multiplication becomes the addition by taking the logarithm of the spectral magnitude

$$log|S(\omega)| = log|E(\omega)\Theta(\omega)| = log|E(\omega)| + log|\Theta(\omega)| = C_e(\omega) + C_\theta(\omega)$$

The Inverse Fourier Transform work individually on the two components as it is a linear

$$c_s(n) = \mathfrak{F}^{-1}\{C_e(\omega) + C_\theta(\omega)\} = \mathfrak{F}^{-1}\{C_e(\omega)\} + \mathfrak{F}^{-1}\{C_\theta(\omega)\} = c_e(n) + c_\theta(n)$$

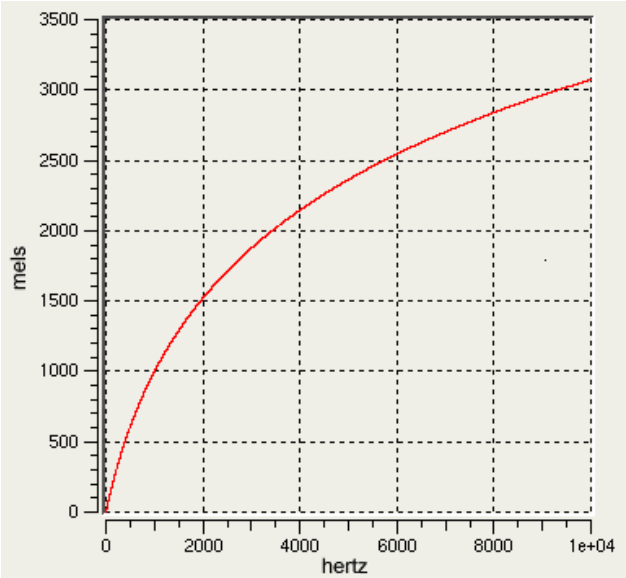
The domain of the signal *cs*(*n*) is called the quefrency-domain.

**Mel Frequency Cepstral Coefficients (MFCC):**

In this project we are using Mel Frequency Cepstral Coefficient. Mel frequency Cepstral Coefficients are coefficients that represent audio based on perception. This coefficient has a great success in speaker recognition application. It is derived from the Fourier Transform of the audio clip. In this technique the frequency bands are positioned logarithmically, whereas in the Fourier Transform the frequency bands are not positioned logarithmically. As the frequency bands are positioned logarithmically in MFCC, it approximates the human system response more closely than any other system. These coefficients allow better processing of data.

In the Mel Frequency Cepstral Coefficients the calculation of the Mel Cepstrum is same as the real Cepstrum except the Mel Cepstrum's frequency scale is warped to keep up a correspondence to the Mel scale.

The Mel scale was projected by Stevens, Volkmann and Newman in 1937.
The Mel scale is mainly based on the study of observing the pitch or frequency perceived by the human. The scale is divided into the units mel. In this test the listener or test person started out hearing a frequency of 1000 Hz, and labelled it 1000 Mel for reference. Then the listeners were asked to change the frequency till it reaches to the frequency twice the reference frequency. Then this frequency labelled 2000 Mel. The same procedure repeated for the half the frequency, then this frequency labelled as 500 Mel, and so on. On this basis the normal frequency is mapped into the Mel frequency. The Mel scale is normally a linear mapping below 1000 Hz and logarithmically spaced above 1000 Hz. Figure below shows the example of normal frequency is mapped into the Mel frequency.




$$m = 1127.01048 \log_e(1 + f/700) \quad \text{.....(1)}$$

$$f = 700(e^{m/1127.01048} - 1) \quad \text{.....(2)}$$

The equation (1) above shows the mapping the normal frequency into the Mel frequency and equation (2) is the inverse, to get back the normal frequency.

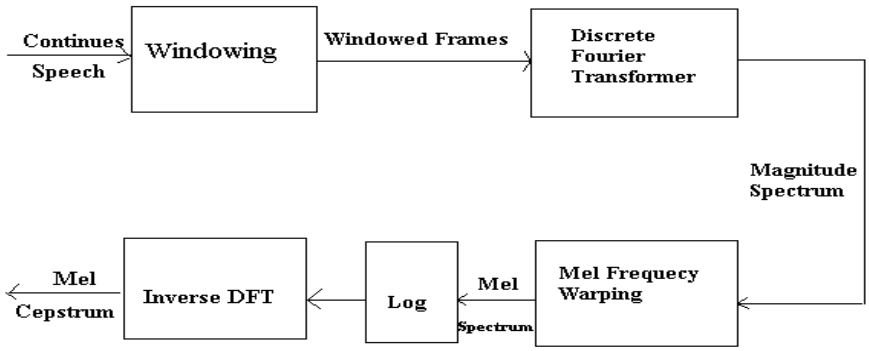



Figure above shows the calculation of the Mel Cepstrum Coefficients. Here we are using the bank filter to warping the Mel frequency. Utilizing the bank filter is much more convenient to do Mel frequency warping, with filters centered according to Mel frequency. According to the Mel frequency the width of the triangular filters vary and so the log total energy in a critical band around the center frequency is included. After warping are a number of coefficients.

$$Y(k) = \sum_{j=1}^{N/2} S(j) H_k(j)$$

Finally we are using the Inverse Discrete Fourier Transformer for the cepstral coefficients calculation. In this step we are transforming the log of the quefrench domain coefficients to the frequency domain. Where N is the length of the DFT we used in the cepstrum section.

$$c(n) = \frac{1}{N'} \sum_{k=0}^{N'-1} Y(k) e^{j\frac{k2\pi}{N'}n}$$

## Delta Cepstrum

Delta Cepstrum is used to catch the changes between the different frames. Delta Cepstrm defined as:

# Vector Quantization:

A speaker recognition system must able to estimate probability distributions of the computed feature vectors. Storing every single vector that generate from the training mode is impossible, since these distributions are defined over a high-dimensional space. It is often easier to start by quantizing each feature vector to one of a relatively small number of template vectors, with a process called vector quantization. VQ is a process of taking a large set of feature vectors and producing a smaller set of measure vectors that represents the centroids of the distribution.

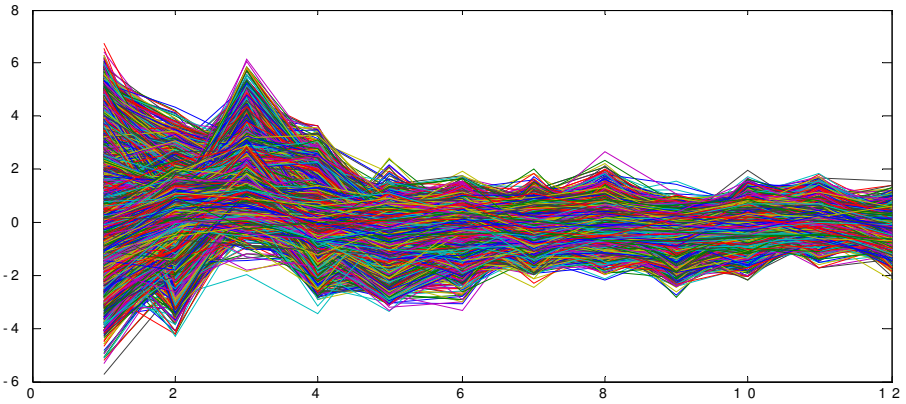



Fig 3.1 the vectors generated from training before VQ

The technique of VQ consists of extracting a small number of representative feature vectors as an efficient means of characterizing the speaker specific features. By means of VQ, storing every single vector that we generate from the training is impossible.

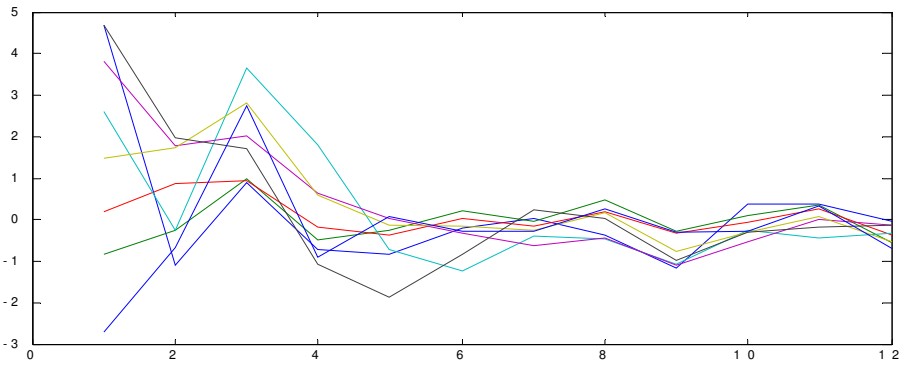



Fig 3.2 the representative feature vectors resulted after VQ

By using these training data features are clustered to form a codebook for each speaker. In the recognition stage, the data from the tested speaker is compared to the codebook of each speaker and measure the difference. These differences are then use to make the recognition decision.

## K-means Algorithm

The **K-means algorithm** is a way to cluster the training vectors to get feature vectors. In this algorithm clustered the vectors based on attributes into k partitions. It use the *k* means of data generated from gaussian distributions to cluster the vectors. The objective of the k-means is to minimize total intra-cluster variance, V.

$$V = \sum_{i=1}^{k} \sum_{j \in S_i} |x_j - \mu_i|^2$$

where there are *k* clusters $S_i$, $i$ = 1,2,...,*k* and $\mu_i$ is the centroid or mean point of all the points $x_j \in S_i$.

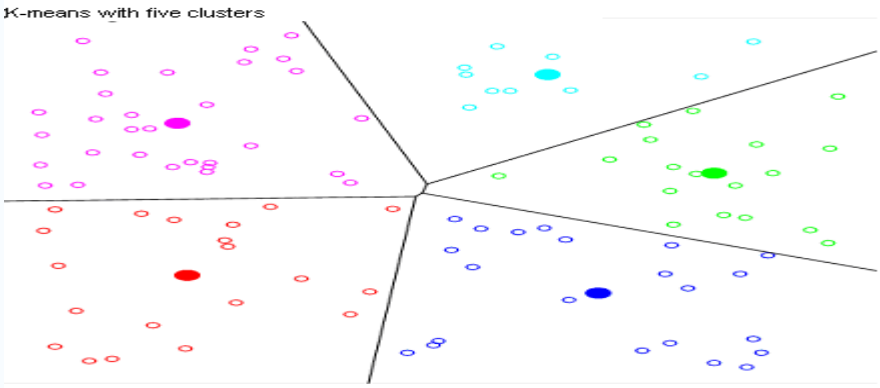



Fig 3.3 clusters in the k-mean algorithm

http://www.csit.fsu.edu/~burkardt/f_src/kmeans/test01_clusters.png

The process of k-means algorithm used least-squares partitioning method to devide the input vectors into k initial sets. It then calculates the mean point, or centroid, of

each set. It constructs a new partition by associating each point with the closest centroid. Then the centroids are recalculated for the new clusters, and algorithm repeated until when the vectors no longer switch clusters or alternatively centroids are no longer changed.

## Distance measure:

In the speaker recognition phase, an unknown speaker's voice is represented by a sequence of feature vector {x1, x2 ….xi), and then it is compared with the codebooks from the database. In order to identify the unknown speaker, this can be done by measuring the distortion distance of two vector sets based on minimizing the Euclidean distance.

The Euclidean distance is the "ordinary" distance between the two points that one would measure with a ruler, which can be proven by repeated application of the Pythagorean Theorem. The formula used to calculate the Euclidean distance can be defined as following:

The Euclidean distance between two points P = (p1, p2…pn) and Q = (q1, q2...qn),

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

The speaker with the lowest distortion distance is chosen to be identified as the unknown person.

## Result:

For example, we are going to test speech wave file made by Brian, which called ‘test_brian.wav’. Assume we do not know the speaker is Brian at the beginning. Therefore we need to apply the wav. file into our speaker recognition system to find out who the speaker is. We run the program twice in order to get a more accurate result. The Matlab codes are provided as following:

% First run
*>> speakerID('test_brian')*
*Loading data...*
*Calculating mel-frequency cepstral coefficients for training set...*
*Harry*
*Carli*
*Brian*
*In___*
Hojin
*Performing K-means...*
*Calculating mel-frequency cepstral coefficients for test set...*
*Compute a distortion measure for each codebook...*
*Display the result...*
*The average of Euclidean distances between database and test wave file*
*Harry*
*7.0183*
*Carli*
*10.0679*
Brian
*5.9630*
*In___*
*8.4237*
*Hojin*
*7.6526*
*The test voice is most likely from*
*Brian*

% Second run

*>> speakerID('test_brian')*

*Loading data...*

*Calculating mel-frequency cepstral coefficients for training set...*

*Harry*

*Carli*

*Brian*

*In___*

*Hojin*

*Performing K-means...*

*Calculating mel-frequency cepstral coefficients for test set...*

*Compute a distortion measure for each codebook...*

*Display the result...*

*The average of Euclidean distances between database and test wave file*

*Harry*

*6.9995*

Carli

*9.9876*

*Brian*

*5.8339*

*In___*

*8.7075*

*Hojin*

*7.6390*

*The test voice is most likely from*

*Brian*

From the above outputs we had in Matlab, we got 5 measurements for each run, whic are the calculated Euclidean distances between the test wave file and codebooks from the database. We can see that, compare to the codebooks in the database; both calculated distortion distance of Brian have the smallest values, which are 5.9630 and 5.8339. Therefore, we can conclude that the speak person is Brian according to the

theory: “the most likely speaker’s voice should have the smallest Euclidean distance compared to the codebooks in the database”.

## Conclusion:

The goal of this project was to create a speaker recognition system, and apply it to a speech of an unknown speaker. By investigating the extracted features of the unknown speech and then compare them to the stored extracted features for each different speaker in order to identify the unknown speaker.

The feature extraction is done by using MFCC (Mel Frequency Cepstral Coefficients). The function ‘melcepst’ is used to calculate the mel cepstrum of a signal. The speaker was modeled using Vector Quantization (VQ). A VQ codebook is generated by clustering the training feature vectors of each speaker and then stored in the speaker database. In this method, the K means algorithm is used to do the clustering. In the recognition stage, a distortion measure which based on the minimizing the Euclidean distance was used when matching an unknown speaker with the speaker database.

During this project, we have found out that the VQ based clustering approach provides us with the faster speaker identification process.

# MATLab Codes:

## speakerID.m

```
function speakerID(a)
% A speaker recognition program. a is a string of the filename to be tested
% against the database of sampled voices and it will be evaluated whose
% voice it is.
% - Example -
% to test a 'test.wav' file then,
% >> speakerID('test.wav')
%
% - Reference -
% Lasse Molgaard and Kasper Jorgensen, Speaker Recognition,
% www2.imm.dtu.dk/pubdb/views/edoc_download.php/4414/pdf/imm4414.pdf
%
% Mike Brooks, VOICEBOX, Free toolbox for MATLab,
% www.ncl.ac.uk/CPACTsoftware/MatlabLinks.html
% disteusq.m enframe.m kmeans.m melbankm.m melcepst.m rdct.m rfft.m from
% VOICEBOX are used in this program.



test.data = wavread(a);                         % read the test file
name = ['Harry';'Carli';'Brian';'In___';'Hojin'];   % name of people in the database

fs = 16000;             % sampling frequency
C = 8;                  % number of centroids

% Load data
disp('Loading data...')
[train.data] = Load_data(name);

% Calculate mel-frequecny cepstral coefficients for training set
disp('Calculating mel-frequency cepstral coefficients for training set...')
[train.cc] = mfcc(train.data,name,fs);

% Perform K-means algorithm for clustering (Vector Quantization)
disp('Performing K-means...')
[train.kmeans] = kmean(train.cc,C);

% Calculate mel-frequecny cepstral coefficients for training set
disp('Calculating mel-frequency cepstral coefficients for test set...')
test.cc = melcepst(test.data,fs,'x');

% Compute average distances between test.cc with all the codebooks in
% database, and find the lowest distortion
disp('Compute a distortion measure for each codebook...')
[result index] = distmeasure(train.kmeans,test.cc);

% Display results - average distances between the features of unknown voice
% (test.cc) with all the codebooks in database and identify the person with
% the lowest distance
disp('Display the result...')
dispresult(name,result,index)
```

# Load_data.m

```
function [data] = Load_data(name)
% Training mode - Load all the wave files to database (codebooks) %

data = cell(size(name,1),1);

for i=1:size(name,1)
   temp = [name(i,:) '.wav'];
   tempwav = wavread(temp);
   data{i} = tempwav;
end
```

# mfcc.m

```
function [cepstral] = mfcc(x,y,fs)
% Calculate mfcc's with a frequency(fs) and store in ceptral cell. Display
% y at a time when x is calculated

cepstral = cell(size(x,1),1);

for i = 1:size(x,1)
   disp(y(i,:))
   cepstral{i} = melcepst(x{i},fs,'x');
end
```

# kmean.m

```
function [data] = kmean(x,C)
% Calculate k-means for x with C number of centroids

train.kmeans.x = cell(size(x,1),1);
train.kmeans.esql = cell(size(x,1),1);
train.kmeans.j = cell(size(x,1),1);

for i = 1:size(x,1)
   [train.kmeans.j{i} train.kmeans.x{i}] = kmeans(x{i}(:,1:12),C);
end

data = train.kmeans.x;
```

# distmeasure.m

```
function [result,index] = distmeasure(x,y)

result = cell(size(x,1),1);
dist = cell(size(x,1),1);
mins = inf;

for i = 1:size(x,1)
   dist{i} = disteusq(x{i}(:,1:12),y(:,1:12),'x');
   temp = sum(min(dist{i}))/size(dist{i},2);
   result{i} = temp;
```

```
    if temp < mins
        mins = temp;
        index = i;
    end
end
```

# dispresult.m

```
function dispresult(x,y,z)

disp('The average of Euclidean distances between database and test wave file')
color = ['r'; 'g'; 'c'; 'b'; 'm'; 'k'];
for i = 1:size(x,1)
    disp(x(i,:))
    disp(y{i})
end
disp('The test voice is most likely from')
disp(x(z,:))
```

## References:


1. Mike N. & Wei W. 2004, "Speaker Recognition", available at: http://cslu.cse.ogi.edu/HLTsurvey/ch1node47.html [viewed on 15th Sept 2006]

2. "Speech Production" Available at: http://www.ise.canberra.edu.au/un7190/Week04Part2.htm [viewed on 15th Sept 2006]

3. Wikipedia, "Window function", available at: http://en.wikipedia.org/wiki/Window_function [viewed on 25th sept 2006]

4. Wikipedia, "Cepstral Concept", available at: http://en.wikipedia.org/wiki/Cepstrum#Cepstral_concepts [viewed on 25th sept 2006]

5. "feature extraction technique", available at: http://www.lsv.uni-saarland.de/dsp_ss05_chap9.pdf [veiwed on 27th sept 2006]

6. Wikipedia, "Euclidean distance", available at: http://en.wikipedia.org/wiki/Euclidean_distance [viewed on 17th Oct 2006]

7. "K-Means least-squares partitioning method" available at: http://www.bio.umontreal.ca/casgrain/en/labo/k-means.html [viewd on 5th Oct 2006]

8. "The k-means algorithm" available at:

http://www.cs.tu-bs.de/rob/lehre/bv/Kmeans/Kmeans.html [viewed on 7th oct 2006]

9. "K means Analysis" available at: http://www.clustan.com/k-means_analysis.html [viewed on 16th oct]

10. Wikipedia "Mel frequency cepstral coefficient" available at: http://en.wikipedia.org/wiki/Mel_frequency_cepstral_coefficient [viewed on 15th oct 2006]

11. Wikipedia "Mel Scale" available at: http://en.wikipedia.org/wiki/Mel_scale [viewed on 18 th oct 2006]