Fourier Transform and MFCCs

Fourier transform decomposes the composite input signal into its constituents signals and frequencies. It decompose the signal into sinusoids and record the properties of each sinusoids as a complex numbers. We want to transfer the signal from the space of time domain to another domain - the frequency domain. Frequency components are the constituent signals that constitute the composite signal. The Fourier transform gives us some complex numbers in the complex array which are the coefficients the signal is multiplied. The number of components in the index frequency domain equal to the number of fast Fourier transform size.

Discrete Fourier Transform

Discrete Fourier Transform is an algorithm that deals with the discrete and periodic signals instead of continuous signals.

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{-2\pi ikn}{N}}$$

The discrete time signal is a vector $x(n)$ and that is being multiplied with different frequencies, which is the result of $e^{\frac{-i2\pi nk}{N}}$, the sin and cosine wave, for different values of n, and when we perform multiplication for the first component x(k), we get the inner product between the time signal and the frequencies (sin and cosine) at k = 0. The real part of the FFT is how well the time signal correlates to a cosine wave of a given frequency and the imaginary part correlates to the sine wave of the same frequency. Knowing both these things is necessary if we want to know the phase of the frequency in our original signal. Following the Euler's formula we can write it as: $e^{\frac{-2\pi ikn}{N}} = cos(\frac{2\pi nk}{N})$+ i $sin(\frac{2\pi nk}{N})$

$cos(\frac{2\pi nk}{N})$, the real part, represent the horizontal coordinate in the complex plane, contributing to the the magnitude of the frequency and i $sin(2\pi nkN)$, regulating the phase, represents the vertical coordinate. The 2π is a complete frequency cycle, ensuring one complete oscillation of the wave on a discrete time axis. nk/N determines the frequency of the wave. $i\ sin(\frac{2\pi nk}{N})$ refers to the imaginary part, product of real and imaginary unit i answered by its property $i^2 = -1$. The sine and cosine part they each contain different values.

Discrete Fourier Transform Matrix

A discrete Fourier transform matrix is a complex matrix whose matrix product with a vector computes the discrete Fourier transform of the vector. This is a **Vandermonde matrix**

which contains the complex exponentials. An N-point DFT is expressed as the multiplication of $X = Wx$, where x is the original input signal, W is the N-by-N square DFT matrix, corresponds to $e-\frac{i2\pi kn}{N}$, N is the length of total samples in the signal.

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_{K-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \dots & 1 \\ 1 & e^{-i2\pi/N} & e^{-i4\pi/N} \dots & e^{-i2\pi(N-1)/N} \\ 1 & e^{-i4\pi/N} & e^{-i8\pi/N} \dots & e^{-i2\pi2(N-1)/N} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & e^{-i2\pi(N-1)/N} & e^{-i2\pi2(N-1)/N} \dots & e^{-i2\pi(N-1)(N-1)/N} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

Multiplying the DFT matrix W times a signal vector x produces a column-vector $X = Wx$ in which the k th element X_k is the inner product of the k th DFT, with sinusoid x.

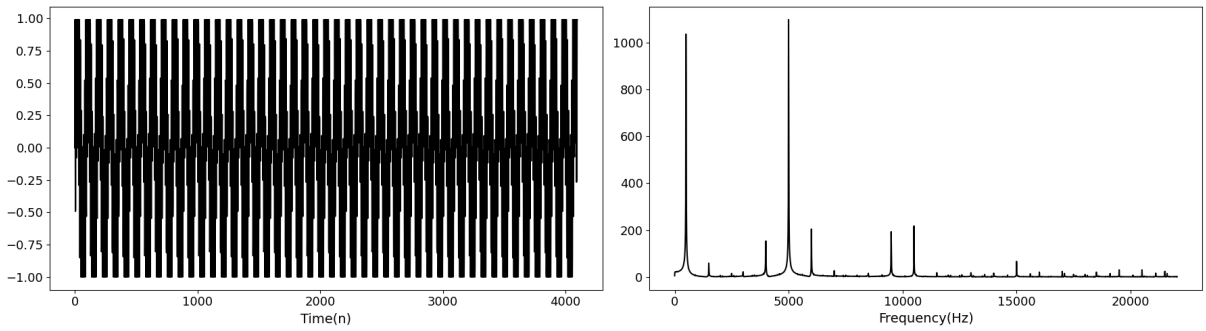


Figure 1: DFT of a 500 Hz and 5000 Hz mixed signal sampled at 44.1 kHz

In the Figure 1 above, we can see that there are two peaks or frequency components at k = 5000 Hz and another is nearly at 500 Hz, which is the first partial or the fundamental frequency. These two frequency components are the constituent signals for the given discrete time signal. In the Figure 1, the left part illustrates the distribution of time domain signal which are periodic.

Fast Fourier Transform

The fast Fourier transform (FFT) is an algorithm that computes the discrete fourier transform (DFT) of a sequence and make it much memory efficient and faster. The most common FFT algorithm is the Cooley-Tukey algorithm based on the principle of divide and conquer. It recursively divides the input signal into smaller and smaller sub-problems halving the signal size in each step into even and odd indices until each group contains only a pair of samples. It performs two samples DFT on each pair of samples or groups in the signal repeatedly and combine the result. FFT algorithm significantly reduces the computational complexity, we can easily understand FFT with an example.

The formula of DFT can be written in a compact form as:

$$X_k = \sum_{n=0}^{N-1} x_n W_N^{kn}$$

where, W = -$\frac{i2\pi}{N}$, which is referred as twiddle factor.

We are assuming a signal, sample size = 8. The FFT algorithm break down a time series smaller signals or even and odd parts, known as "Decimation in Time'. The FFT algorithm break down the signal into stages termed as recursion.

$$(x_0 \quad x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7)$$
$$(x_0 \quad x_2 \quad x_4 \quad x_6) \quad (x_1 \quad x_3 \quad x_5 \quad x_7)$$
$$(x_0 \quad x_4)(x_2 \quad x_6)(x_1 \quad x_5)(x_3 \quad x_7)$$

The above input indices are not in linear order, FFT enables to produce the output in both linear and bit-reversed order. The above paired samples in bit reversed order which produces the frequency indices linear order.

Index	0	1	2	3	4	5	6	7
Binary	000	001	010	011	100	101	110	111
Bit-reversed binary	000	100	010	110	001	101	011	111
Bit-reversed index	0	4	2	6	1	5	3	7

Table 1: Bit-reversal process for FFT size N = 8

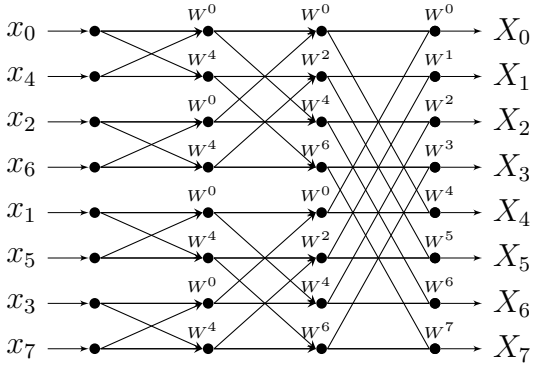
Decomposing the DFT into odd and even indices operation:

$$\begin{aligned} X_k &= \sum_{m=0}^{N/2-1} x_{2m} W_N^{2mk} + \sum_{m=0}^{N/2-1} x_{2m+1} W_N^{k(2m+1)} \\ &= \sum_{m=0}^{N/2-1} x_{2m} W_{N/2}^{mk} + W_N^k \sum_{m=0}^{N/2-1} x_{2m+1} W_{N/2}^{mk} \\ &= E_k + W_N^k O_k \end{aligned}$$

Where 2m refers to the even index and 2m+1 odd position numbers, m = 0.....$\frac{N}{2}$-1

Finally we ended up with two terms. This approach offers the advantage of concurrent computation for the even and odd indexed subsequences. The decomposition multiplies the odd indexed DFT with the twiddle factor W_N^k and add with the even indexed calculation to generate the first half of FFT and subtraction leads to the generation of second half FFT can be formulated as:

$$X_k = E_k + W_N^k O_k$$
$$X_{k+\frac{N}{2}} = E_k - W_N^k O_k$$



FFT calculates in stages, take only two point for calculation which take beautiful sweet shape of a butterfly. The successive stages of the FFT exhibit a doubling of the butterfly width, resulting in a geometric progression of widths 1, 2, 4 where the common ratio is 2. The exponents associated with the butterfly coefficients in successive stages of the FFT algorithm indicate a pattern directly related to the doubling of the $n \in [0 : N]$ backward. This pattern can be characterized as a geometric series, where the exponents at each stage are determined by recursively multiplying the series of the last stage by 2 and applying the modulo operation of N. In the above butterfly diagram, exponents for the last stage are respectively 0, 1, 2, 3, 4, 5, 6, 7. The exponents in the previous stage multiplied by 2 with the series of last stage are 0, 2, 4, 6, 8, 10, 12, 14 and modulo of N = 8, are 0, 2, 4, 6, 0, 2, 4, 6 and exponents in the first stage are 0, 4, 0, 4, 0, 4, 0, 4 which results from the modulo of 8 over 0, 4, 8, 12, 16, 20, 24, 28.

Intuitively, we can understand that FFT will scale the magnitude of the sine wave, and which will lay down the circle in wave form, and the weights in each quadratic are similar except negative and positivity of things. The recursion will make use of these repetitively which make it efficient. The idea of Nth roots of unity help us to see it clearly.

The Nth roots of unity tells us each value of the complex coefficients with exponents N will give value 1. The powers of roots of unity are periodic above with total N periods, where we get the location for each weights in the circle dividing $\frac{2\pi}{N}$, since π refers to the number of radians in half circle or 180 degree. The properties of Nth roots of unity make the recursion work out properly. Squaring the Nth root of unity produces $\frac{N}{2}$ roots of unity, Nth roots are plus-minus paired $W(\frac{N}{2+K}) = -W^K$, we can easily find it on a unit circle, complex roots of unity contain the property of pairing. Thus, choosing positive-negative pairs, the even powers coincide with the odd ones and for each squared number, the negative value is the reflection of that positive number. The recursion won't work out if the exponents are not positive and negative pairs, complex numbers gently deal with it, recursion makes the evaluation to be performed $\frac{N}{2}$ times.

The basic computation performed at every stage is to take two complex numbers, for example, x_0, x_4 multiply x_4 by W_N^r and then add the product from a to form two new complex numbers X_0, X_1. The very first stage calculate four pairs then combined into 4 point split of two pairs butterflies next stage. Now four point splits the signal into four cosine and four sine components for the value of $k \in [0 : 3]$ which manipulates some of the earlier calculated DFT that resembles, e.g., earlier result of x_0, x_2 to X_2

The computation of butterflies include one complex multiplication which involves four real multiplies. The number of stages is $log_2 N$ and each stage has N/2 butterflies, so the total number complex multiplication is $4\frac{N}{2}log_2 N$ or $2Nlog_2 N$. Assume for a signal N = 262144, calculation needed $4N^2 \sim 27.5$ billion. To calculate in FFT, number of stages

$log_2 N = 18$, so required calculation $2 * 262144 * 18 \sim 9.43$ million nearly 29000 times faster.

Short-time Fourier transform (STFT)

Fourier transform averages the frequency distribution over time. The characteristics of signal for a specific portion diagnosed through STFT. The quality of frequency distribution, its changes, pattern and other properties confined within time illustrated through STFT. The Short-time Fourier transform divide the signal into sections and then slide the window function over the sections with a certain step size. We can compare the the activity of window function as weight parameters in typical neural network models or kernels in CNN, but the optimization continuously is abstract. Choosing the type of window and determining its size is important to reveal the pure frequency spectrum over time. The step size defines how much the analysis window is shifted along the time axis. A smaller step size leads to greater overlap between consecutive frames. The use of a window function in the STFT is crucial for mitigating spectral leakage. An abrupt truncating of signal into slices creates artificial discontinuities, creates a lot of noise that shows up in the high-frequency. e.g., rectangular window. The smoothly taper near the edge time information better with less noise compared to rectangular window like hanning or hamming window. The selection of a window function involves a trade-off between spectral leakage and bandwidth.

The formula for **STFT**:

$$X(m,k) = \sum_{n=0}^{N-1} x(n+mH)w(n)exp^{(-2\pi ikn/N)}$$

In the above formula, H refers to the hop size or step size which make the window slide for definite number of samples forward. The multiplication of window function with a portion of signal give us the windowed signal. m denotes the number of frames or overlapping segments. $w(n)$ refers to the window function which is a vector m^{th} length. The above equation extracts segments of signal x, computes the window signal, performs Fourier transform, and stores the computed FFT results for the current segment in the corresponding m^{th} column of the output matrix. The number K = N/2, k $\in$ [0:K], is the k^{th} Fourier coefficient for m^{th} time frame.

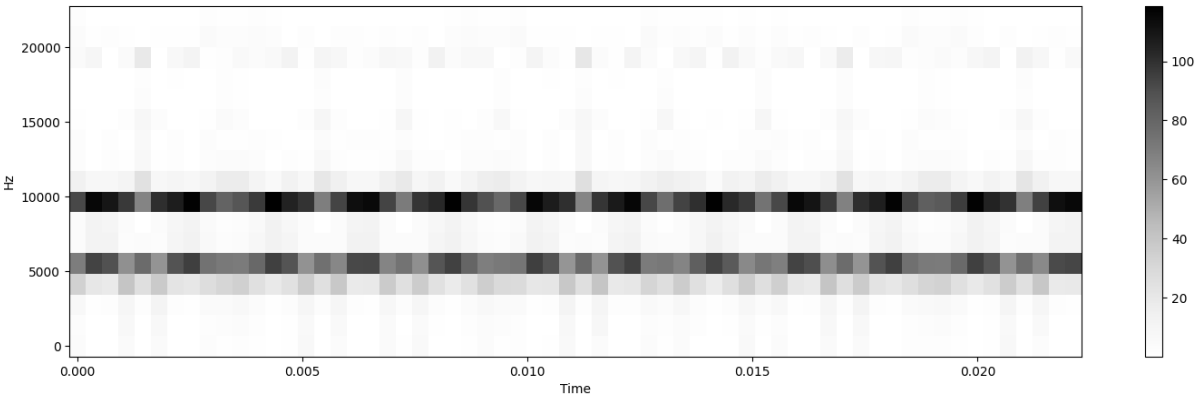


Figure 2: Spectrogram of the signal

This STFT analyzes an audio signal, Figure 2 composed of two frequencies: 5000 Hz and 10000 Hz. The signal is sampled, and the first 1024 samples are considered. STFT is performed on the signal using a FFT size 32 and a a hop size of 8. The time axis is divided into frames. The frequency axis of the spectrogram represents the frequency bins obtained from the FFT which is 17 in the Figure 2 and we can count this. A frequency bin is a point in the frequency domain representation of a signal that covers a specific range of frequencies. When we apply the FFT to a time-domain input, it divides the entire frequency range into discrete bins, each representing a specific frequency. Now the number of frequency bins or how much dense the frequency domain or what frequency range a single point contain depended on sampling rate of the signal and the total number of points in the FFT. The larger the number of FFT, the denser it becomes or shrink the bin size. For instance, if we take a sample rate 44100/512 FFT, we get a frequency bin size of 86, if we increase the FFT to 1024, the bin size would be 43. The time dimension is controlled by the total length of time signal and hop size.

Mel-frequency cepstral coefficients(MFCCs)

MFCCs are a popular audio feature extraction method identify the components of a audio signal and widely used in speech recognition or speech-to-text transformation. It follows the responsiveness of human hearing behavior or nature. Human voice is filtered by the shape of the vocal tract includes the nose, throat, lips, and tongue, and is responsible for producing sound of different features. The features contemplating can be extracted through the MFCCs which is not but the features or coefficients representing the envelops of the short time power spectrum and these envelopes evince the shape of the vocal tract itself. MFCCs extracted from the audio signal serve as the input observations for the Hidden Markov Model(HMM) or further operation for recognizing the sequences of sounds and ultimately understand the spoken word. The very first step for MFCCs starts with splitting the signals into frames. The frame size should not be very short because they don't provide enough data for reliable spectral estimation and a longer size contain too much signal changes. The typical size of frames lasts between 20 to 40 ms. Subsequently, the power spectrum is computed for each frame. The spectrum contains the frequency information for each frame as what cochlea do in case of human brain. The cochlea, a fluid-filled, spiral cavity within the inner ear, plays a crucial role in identifying the frequency components of sound which has more filters at low frequency and less filters at higher frequency. The basilar membrane within cochlea vibrate at specific locations, according to the frequency of input sound and propagates sound vibrations to the brain for interpretation. But, one drawback is that the frequency components lies very closely is not well recognizable by the cochlea and to address the problem frequency components are binned into parts covering certain areas of energy spectrum and the energy exist within each bin is summed. This is where we get introduced with filter bank, the filter bank hold the energy values of different locations of frequencies and to fix the width and size of filter bank we use the mel scale. Mel scale determines the width and sparsity of each filter bank in the periodogram.

Humans are less sensitive to small energy change at high energy than small changes at a low energy level. Because, we human perceive sound in logarithmic fashion. The loudness of sound and intensity of energy are not the same thing. Loudness refers to the psychological perception of sound relative to the function of human hearing (pressure), whereas intensity is a physical quantity which measures the units of sound energy moving

through an area in a unit of time. The first is dependent on an organic perceptive element, whereas the latter is just an arbitrary measurement of physical phenomena moving through mass. In terms of sound pressure level, audible sounds range from the threshold of hearing at 0 dB to the threshold of pain which can be over 130 dB, up to 80 dB safe. An increase of 3 dB represents doubling or sound pressure while, it requires the sound increase of 10 dB in order to perceived twice loud. Lastly, we compute the DCT of the log filter bank which de-correlates the overlapped correlated filter bank and extracts the cepstral coefficients which are compact representations of the spectral envelope.

Implementation of MFCCs in Steps

The first task in MFCCs is to fix the size for each frame size and we already said the usual size of frames range from 20 ms to 40 ms. We taken the frame size of 25 ms which is decent.

Periodogram spectral estimate involves the calculation of power spectrum of each frames. We computed the periodogram using **Welch's method** which involves windowing the overlapped segments. We can find overlap percentage dividing the frame size by number of samples that overlap which is in our case was $1102/441 = 59.98$. After the data is split up into overlapping segments, the individual data segments have a window applied to them. We applied the hamming window function on each frames which has more impact on the data at center and results in loss of information, which the overlap mitigates. After multiplication of data frames with the window function we calculate the DFT for each of frame and computing the squared magnitude of the result, yielding power spectrum estimates for each frame, which are then averaged.

The formula of **hamming** window:

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

where, $n = 0, 1, 2...$N-1 is a vector.

The formula of **power spectrum** for each frame:

$$P_i(k) = \frac{1}{N} * |X_i(k)|^2$$

X_i(k) is the indicates the fourier tranform of i_{th} frame and $P_i(k)$ indicates the power spectrum and N is the length of frame or the number of samples in frame. We performed fourier transform of 512 sample size. We selected the first half of the spectrum to discard the redundancy and ended up with 257 coefficients.

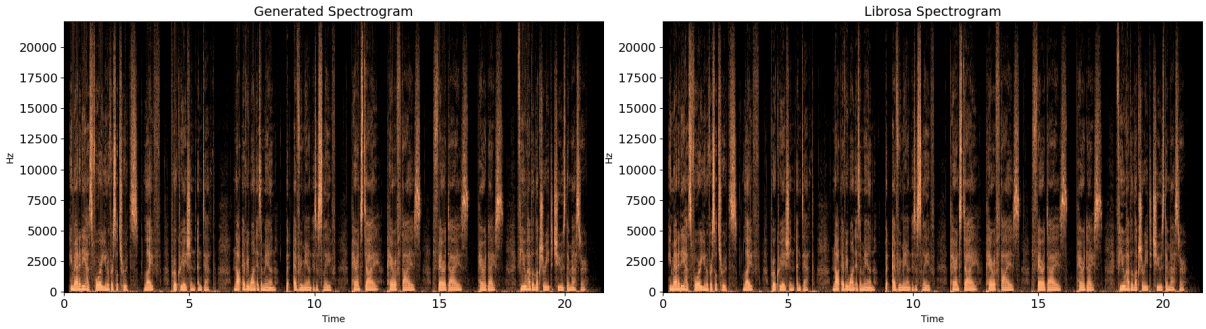


Figure 3: Periodogram Power Spectrogram

Then, The periodogram is converted into the mel filters. The minimum and maximum frequencies are converted from Hertz to the Mel scale using the formula:

$$\text{Mel(k)} = 2595 \cdot \log_{10}(1 + \frac{k}{700})$$

Then we generate evenly spaced mel points between the calculated minimum and maximum mel frequencies which in our case was 0 and half of sampling frequency 22050. The number of mel banks 40. The average use of filter bank number ranges from 26 to 40. Each filter bank is a vector length of 257 which are multiplied with the power spectrum. The generated evenly spaced mel points are converted back to Hertz using the inverse of the previous formula so that it can calculates the center frequency of each mel filter bank in terms of its corresponding bin index in the spectrum.

$$\text{Mel}^{-1}(m) = 700(10^{\frac{m}{2595}} - 1)$$

FFT bin for the for the corresponding mel point or Hertz:
array([0., 1., 3., 4., 6., 8., 10., 13., 15., 18., 21., 25., 28., 32., 37., 41., 47., 52., 58., 65., 72., 80., 89., 98., 108., 119., 131., 144., 159., 174., 191., 210., 230., 252., 275., 301., 329., 360., 393., 430., 469., 513.])
The last bin is set to value of FFT+1 size to ensure it covers the full range. Lastly, we create our filter bank in the sequential and repetitive manner where our first point becomes the starting for the filter bank, second become the peak and third again goes towards the ground. In the next step, the second number become the starting, third one is peak and fourth downward all the way up to the last point. At next step, we take the logarithm and DCT (Discrete Cosine Transform) of each filter bank array and and left with 40 log filter bank array. By applying the DCT, we emphasize the most significant components and reduce the impact of less important details. The coefficients we are left with after the DFT capture the shape of the energy distribution among the Mel frequency bins. The DCT reduces the dimensionality of the data, emphasizing the most significant aspects while minimizing the impact of less important details. The resulting coefficients, known as Mel Frequency Cepstral Coefficients (MFCCs), provide a compact yet informative representation of the audio signal's frequency characteristics.

The formula of **DCT**:

$$MFCC_p(i) = \sqrt{\frac{2}{N}} * \sum_{n=0}^{N-1} S_p(k) * cos\left(\frac{\pi}{2N}(2k-1)(i-1)\right)$$

The above formula, N denotes the number of columns in the log mel spectrum matrix, a cosine weight matrix is created (N, N), k refers to the column index and a dot product is made in our case the shape of log mel spectrum was (257, 40) and weight matrix resulted from the cosine function was (40, 40), which give output of (257, 40).

Lastly, liftering, the low order cepstral coefficients had its characteristics that were very sensitive to the spectral slope, while the high order parts were very sensitive to noise. Therefore, cepstral liftering was one of the standard techniques applied to minimize this sensitivity. Cepstral liftering aims to improve the accuracy used to recognize pattern matching, both speaker recognition and speech recognition.

Cepstral coefficient uses the following liftering window:

$$w(k) = \frac{1}{D} + sin\left(\frac{\pi n}{D}\right)$$

In the above formula, D is the liftering parameter, which regulates the properties of lifting window, 22 is common, higher value means wider and smoother window, de-emphasizes higher-order MFCCs coefficients more gradually, narrower heavily de-emphasizes higher-order coefficients.. k is the index of the cepstral coefficients. n number of cepstral coefficients 40 in our case, controls the shape of the lifting window. Ultimately, we kept 12 coefficients out of the 40 which are generally the most relevant and enough to represent the different phonemes.

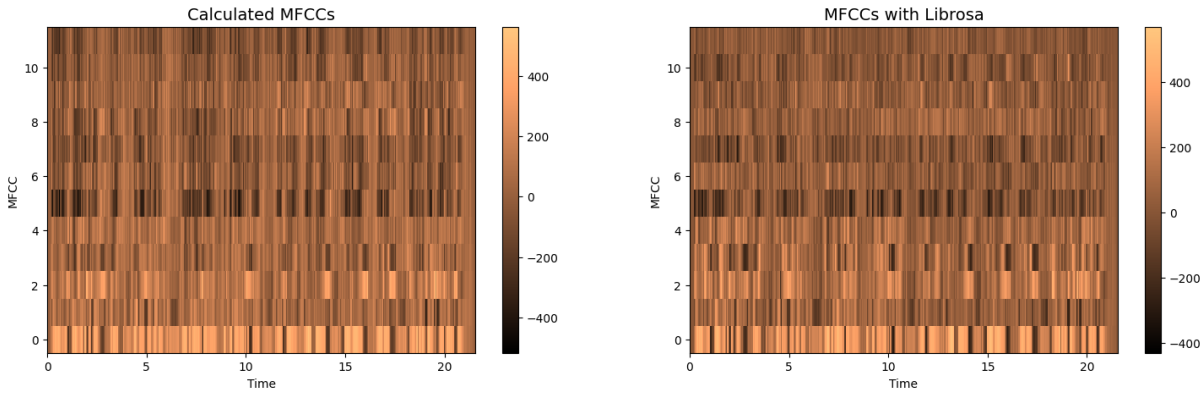


Figure 4: Mel-frequency cepstral coefficients(MFCCs)