NTNU

Norwegian University of
Science and Technology

Speech Analysis for Automatic Speech Recognition

Noelia Alcaraz Meseguer

Master of Science in Electronics
Submission date: July 2009
Supervisor: Torbjørn Svendsen, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

The classical front-end analysis in speech recognition is a spectral analysis which produces features vectors consisting of mel-frequency cepstral coefficients (MFCC). MFCC are based on a standard power spectrum estimate which is first subjected to a log-based transform of the frequency axis (the mel transform), and then decorrelated using a modified discrete cosine transform.

An interesting issue is how much information relevant to speech recognition that is lost in this analysis. Thus, this project is concerned with synthesizing speech from different parametric representations (e.g. MFCC and linear prediction coefficients) and to conduct an investigation on the intelligibility of the synthesized speech as compared to natural speech.

Assignment given: 16. February 2009
Supervisor: Torbjørn Svendsen, IET

A mis padres, Antonio y Carmina,
y a mi hermano Sergio, sin ellos,
sin su incondicional apoyo y confianza,
nunca hubiese llegado hasta aquí.

ABSTRACT

The classical front end analysis in speech recognition is a spectral analysis which parameterizes the speech signal into feature vectors; the most popular set of them is the Mel Frequency Cepstral Coefficients (MFCC). They are based on a standard power spectrum estimate which is first subjected to a log-based transform of the frequency axis (mel- frequency scale), and then decorrelated by using a modified discrete cosine transform.

Following a focused introduction on speech production, perception and analysis, this paper gives a study of the implementation of a speech generative model; whereby the speech is synthesized and recovered back from its MFCC representations. The work has been developed into two steps: first, the computation of the MFCC vectors from the source speech files by using HTK Software; and second, the implementation of the generative model in itself, which, actually, represents the conversion chain from HTK-generated MFCC vectors to speech reconstruction.

In order to know the goodness of the speech coding into feature vectors and to evaluate the generative model, the spectral distance between the original speech signal and the one produced from the MFCC vectors has been computed. For that, spectral models based on Linear Prediction Coding (LPC) analysis have been used. During the implementation of the generative model some results have been obtained in terms of the reconstruction of the spectral representation and the quality of the synthesized speech.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my supervisor at NTNU, Professor Torbjørn Svendsen, for offering me the opportunity to make the Master Thesis with him, and for his indispensable guide during the execution of the Master Thesis.

I would also like to thank to the people at NTNU that, in one way or another, make possible the execution of this Thesis; hence, to extend my gratitude to my home university, *Universidad Politécnica de Valencia*, for giving me the chance of Erasmus year at NTNU, Trondheim.

I would like to express my sincere gratitude and love to my parents. Their support and confidence during all my life have made me as I am, and their effort has allowed me to be here completing my degree with the execution of this Master Thesis.

I wish to express my affection to my closest friends for being always next to me. Thanks to Micaela and, especially, Elena, Laura, Maria e Ica with who I have shared the best moments and have supported me in the difficult ones. Thanks to Natxete who has made wonderful the six years spent together at the university.

Finally, I would like to express my affection to all the friends that have made this Erasmus experience as unique and special as it has been. Specially, thanks to Nicolas, Josevi, Jose and Alberto who have been as my family here, making me feel like home. I wish to highlight my sincere gratitude and love to Alberto for the wonderful months spent here together, and for its support and confidence that have helped me so much during this period.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDIX

LIST OF ABREVATIONS

ASR	Automatic Speech Recognition
dB	Decibel
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
FT	Fourier Transform
HMM	Hidden Markov Model
HTK	Hidden Models ToolKit
IDCT	Inverse Discrete Cosine Transform
IFT	Inverse Fourier Transform
LP	Linear Prediction
LPC	Linear Predictive Coding
MFCC	Mel-Frequency Cepstral Coefficient

1. INTRODUCTION

This Master Thesis was developed at the Department of Electronics and Telecommunications (Faculty of Information Technology, Mathematics and Electrical Engineering) at NTNU University (Trondheim, Norway), from February 2009 to July 2009. The Master Thesis was called *Speech Analysis for Automatic Speech Recognition*.

This Master Thesis is connected to the research project SIRKUS[1]. The aims of SIRKUS project is to investigate structures and strategies for automatic speech recognition; both in terms of what type of linguistic units it uses as the basic unit (today, phonemes, which are perceptually defined, are used), which acoustic properties to look for in the speech waveform, or which classifier to use (Hidden Markov Models (HMM) are predominantly used today).

The classical front end analysis in speech recognition is a spectral analysis which parameterizes the speech signal into feature vectors. The most popular set of feature vectors used in recognition systems is the Mel Frequency Cepstral Coefficients (MFCC). They are based on a standard power spectrum estimate which is first subjected to a log-based transform of the frequency axis; it results in a spectral representation on a perceptually frequency scale, based on the response of the human perception system. After, they are decorrelated by using a modified discrete cosine transform, which allows an energy compaction in its lower coefficients.

An interesting issue is how much relevant information related to speech recognition is lost in this analysis. Thus, this Master Thesis is concerned with synthesizing speech from different parametric representations (MFCCs and Linear Prediction coefficients), and to conduct an investigation on the intelligibility of the synthesized speech as compared to natural speech.

According to this aim, the five principal objectives of the Master Thesis are:

1. Study speech analysis processing and theories based on speech production and speech perception.
2. Investigate on the implementation of MFCC computation in the Hidden Markov Toolkit (HTK), a standard research and development tool for HMM-based speech recognition.
3. Develop a speech generative model based on the implementation of the conversion chain from HTK-generated MFCC representations to speech reconstruction.
4. Employ objective measures for an intermediate evaluation of the generative model.
5. Present a subjective interpretation of the intelligibility of the synthesized speech.

[1] More information: www.iet.ntnu.no/projects/sirkus

All investigation works crash often with certain limitations which avoid a deeply study of the results obtained. In this Master Thesis, the main limitation has been the ignorance of the characteristics of source speech data which could make the recognition performance more difficult. In the other hand, the audio system used for listening synthesized speech was the audio system of a simple commercial laptop; so, slight differences within synthesized speech could not be identified.

To carry out of this Master Thesis, the report has been divided into five sections briefly described below.

The first one, in which the reader is, introduces the Master Thesis, its motivation and its objectives and limitations.

The documentation of this report starts in the second section. It is a presentation of the theoretical concepts in speech production, perception and analysis. Thus, this theoretical section pretends to give an essential background about the speech analysis involved in recognition tasks, in order to understand the basic principles in which the procedures and implementations carried out during this Master Thesis are based on.

The implementation of the speech generative model is explained in section three. This includes an investigation on the implementation of the MFCC computation in HTK, and a thorough explanation of the implementation of the generative mode, making relationships with the based theories.

Later, the results extracted during the implementation of the generative model are analyzed in section four. Also, an objective measure for intermediate evaluation of the generative model is performed.

Finally, the conclusions are drawn in section five.

2. THEORETICAL CONCEPTS FOR SPEECH ANALYSIS

The theoretical section pretends to give an essential background about the speech analysis involved in recognition tasks, in order to understand the basic principles in which the procedures and implementations carried out during this Master Thesis are based on.

The theoretical section is divided into three sections. In the first one, the speech signal and its characteristics are described; the second one is an introduction to front-end analysis for automatic speech recognition, where the important feature vectors of speech signal are explained; and the third is an approach of distance measures based on spectral measures for speech processing.

2.1. THE SPEECH SIGNAL

A brief introduction to how the speech signal is produced and perceived by the human system can be regarded as a starting point in order to go into the field of speech recognition.

The process from human speech production to human speech perception, between the speaker and the listener, is shown in Figure 1.

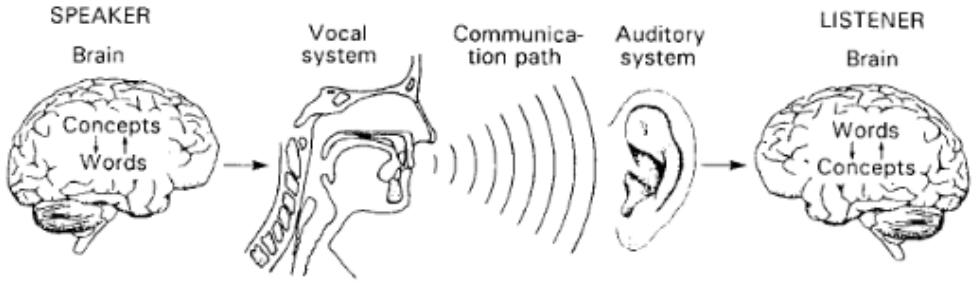

Figure 1: Human speech communication (Holmes & Holmes, 2001)

Speech recognition systems try to establish a similarity to the human speech communication system. A source-channel model for a speech recognition system is illustrated in Figure 2, proposed by Huang et al. (2001).

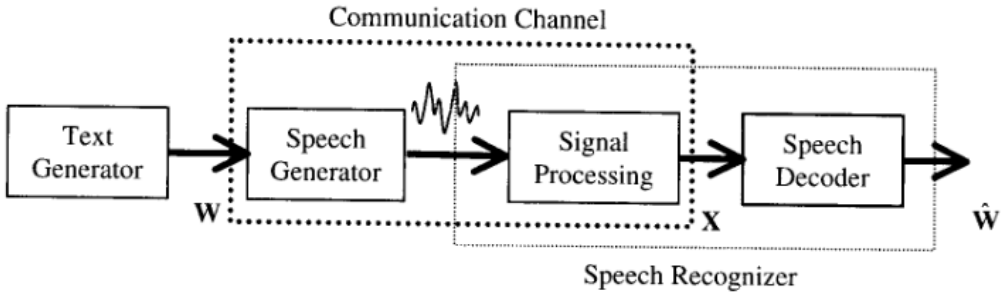

Figure 2: Source-channel model for a speech recognition system (Huang et al., 2001)

The different elements from the human communication system are related below to the modules or components of the source-channel model, giving a short explication of how human speech communication and speech recognition systems are performed.

The aim of human speech communication is to transfer ideas. They are made within the speaker's brain and then, the source word sequence *W* is performed to be delivered through her/his *text generator*. The human vocal system, which is modeled by the *speech generator* component, turns the source into the speech signal waveform that is transferred via air (a noisy *communication channel*) to the listener, being able to be affected by external noise sources. When the acoustical signal is perceived by the human auditory system, the listener's brain starts processing this waveform to understand its content and then, the communication has been completed. This perception process is modeled by the *signal processing* and the *speech decoder* components of the *speech recognizer*, whose aim is *to process and decode the acoustic signal X into a word sequence* $\hat{W}$*, which is hopefully close to the original word sequence W* (Huang et al., 2001).

Thus, speech production and speech perception can be seen as inverse processes in the speech recognition system.

2.1.1. Speech Production

As said before, it is important to know and to understand how humans generate the speech. Since a speech generative model, in addition to speech production knowledge, can itself form a useful basis of speech synthesis system. In this way, a schematic diagram of the human speech production apparatus is illustrated in Figure 3.

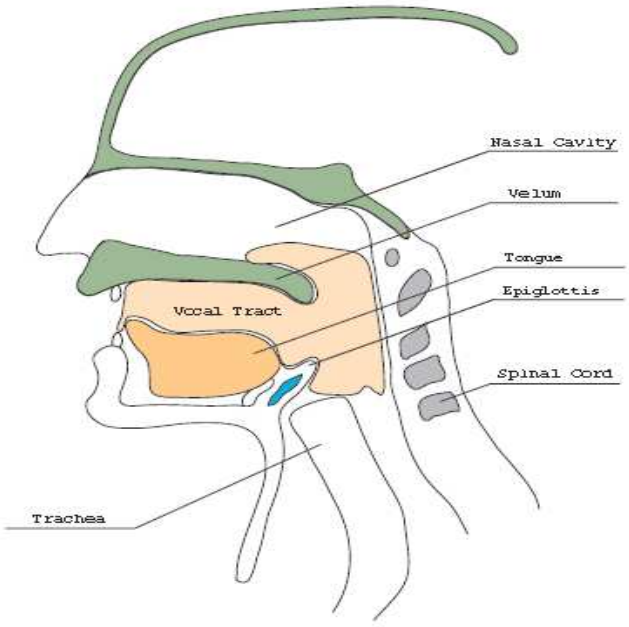

Figure 3: Human speech production apparatus

Speech is produced by air-pressure waves emanating from the mouth and the nostrils of a speaker, as it is defined by Huang et al. (2001).

The main organs involved into the speech production process are the lungs, larynx, nose and various parts of the mouth. The air expelled from the lungs is modulated in different ways to produce the acoustic power in the audio frequency range. After, the rest of the vocal organs, such as vocal cords, vocal tract, nasal cavity, tongue, and lips, modify the properties of the resulting sound to produce the speech waveform signal. These properties can be principally determinate thanks to the acoustical resonance process performed into the vocal tract. The main resonant modes are known as *formants*, being the two lowest frequency formants the most important ones in determining the phonetics properties of speech sounds.

This *resonant system can be viewed as a filter that shapes the spectrum of the source sound to produce speech* (Holmes & Holmes, 2001). This is modulated by *source-filter models of speech production*.

Source-filter Models of Speech Production

The source-filter model consists of an excitation signal that models the sound source, e(n); passing through all-pole filter[2], h(n); to produce the speech signal, s(n); as one can see in Figure 4.

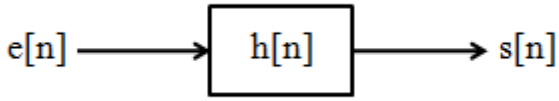

Figure 4: Basic source-filter of speech signal

The speech sounds can be presented in three states:

- Silence – No speech is produced.
- Unvoiced sounds – Vocal cords are not vibrating, resulting in no periodic random speech waveform.
- Voiced sounds – Vocal cords are tensed and vibrating periodically, resulting in a quasi-periodic[3] speech waveform.

[2] An all-pole filter is a filter whose transfer function contains only poles (roots of the denominator), without zeros (roots of the numerator).
[3] Quasi-periodic speech waveform means that the speech waveform can be seen as periodic over a short-time period (5-100 ms), where the signal is assumed stationary.

For voiced sounds, the excitation signal is an impulse train convolved with the glottal pulse (see Figure 5); while for unvoiced sounds, the excitation signal is random noise. Both of them with a gain factor G in order to control the intensity of the excitation.

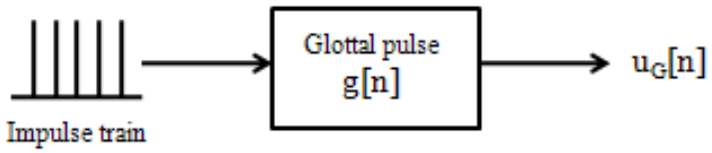

Figure 5: Glottal excitation model for voiced sound

For a complete source-filter model, as is shown in Figure 6, the glottal pulse, vocal tract and radiation have to be individually modeled as linear filter (Fant, 1960). The transfer function, V(z), represents the resonances of the vocal tract, and the transfer function, R(z), models the air pressure at the lips.

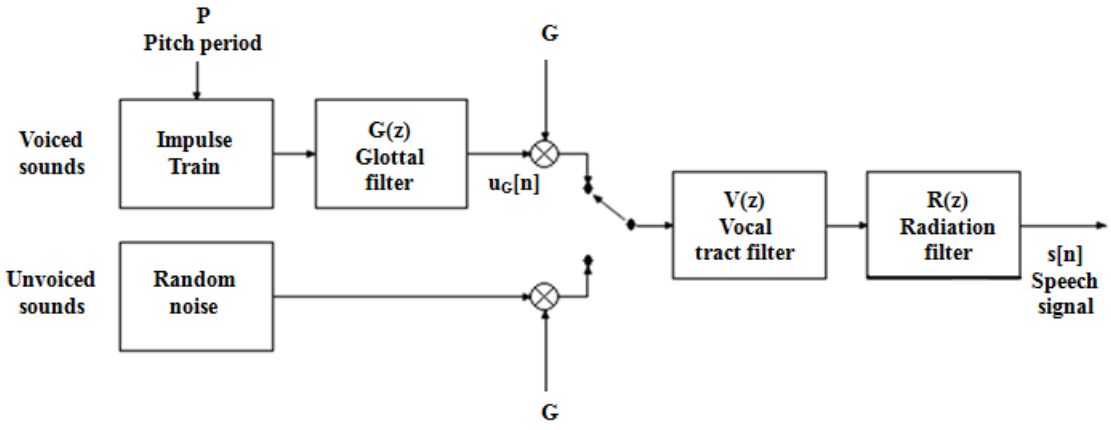

Figure 6: General discrete-time model of speech production

Combining G(z), V(z) and R(z), a single all-pole filter, H(z), is obtained, resulting in a new simple diagram shown in Figure 7.

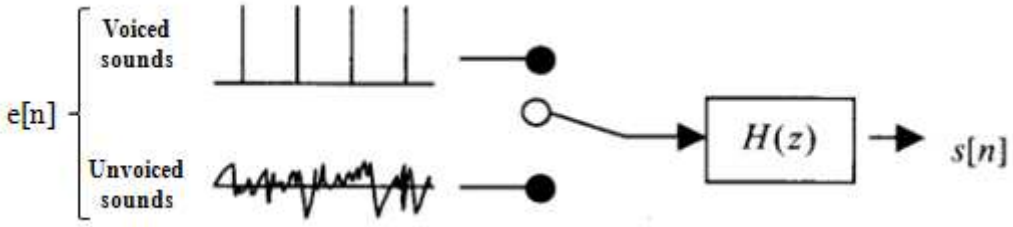

Figure 7: Source-filter model for speech production

The transfer function of H(z) is given by Eq. (1) (Mammone et al., 1996), where p is the filter's order. An enough number of poles give a good approximation for speech signals.

$$H(z) = G(z)V(z)R(z) = \frac{G}{1 - \sum_{i=1}^{p} a_i \, z^{-i}} \tag{1}$$

With this transfer function, a difference equation for synthesizing the speech samples s(n) can be proposed (Mammone et al., 1996):

$$s[n] = \sum_{i=1}^{p} a_i s(n-i) + \; Ge\,[n] \tag{2}$$

2.1.2. Speech Perception

Without going into any further details, this section just presents the auditory perception system and emphasizes its non-linear frequency response. Anymore details are not necessary for the understanding of this Master Thesis.

The auditory perception system can be split in two major components: the peripheral auditory system (ears), and the auditory nervous system (brain). The received acoustic pressure signal is processed by peripheral auditory system into two steps: firstly, it is transformed into a mechanical vibration pattern on the basilar membrane; and then, is represented by a series of pulses to be transmitted by the auditory nerve. Finally, the auditory nervous system is responsible for extracting the perceptual information.

The human ear, as shown in Figure 8, is made up of three parts: the outer ear, the middle ear, and the inner ear. The outer ear consists of the external visible part and the external auditory canal is where sound wave travels. The length of the auditory canal is such that performs as an acoustic resonator whose principal effect is to increase the ear's sensitivity to sounds in the 3-4 KHz range. When the sound arrives at the eardrum, it vibrates at the same frequency as the incoming sound pressure wave. The vibrations are transmitted through the middle ear. The main structure of the inner ear is the cochlea which communicates with the auditory nerve, driving a representation of sound to the brain. The cochlea can be seen as a filter bank, *whose outputs are ordered by location, so that a frequency-to-place transformation is accomplished. The filters closest to the cochlear base respond to the higher frequencies, and those closest to its apex respond to the lower* (Huang et al., 2001).

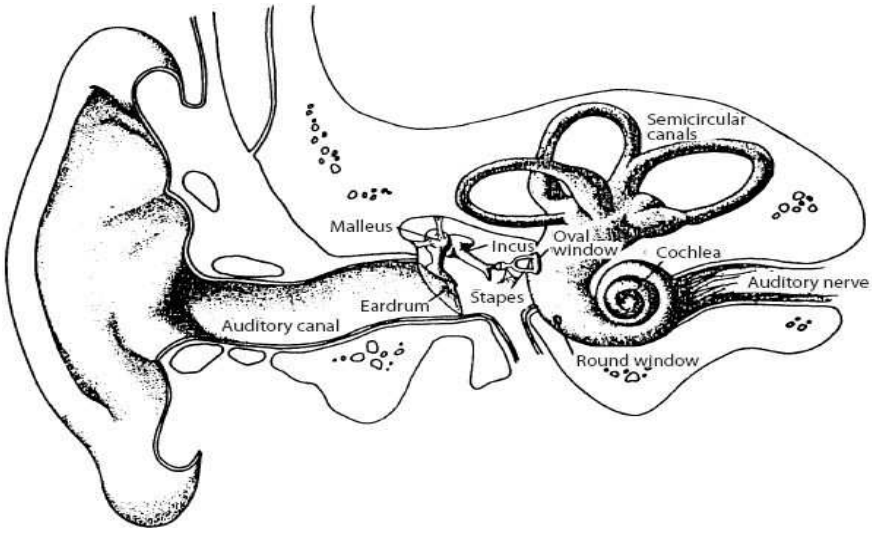

Figure 8: Peripheral auditory system

The main issue to model for one speech generative model is the nonlinear character of the human hearing system. That is why psychoacoustic experimental works have been undertaken to find frequency scales that can model the natural response of the human perceptual system.

Fletcher (1940) introduced for the first time the term of *critical bands*, pointing the existence of them in the cochlear response. Since that moment, several different experiments have been carried out to investigate critical band phenomena and to estimate critical bandwidth. There are two outstanding classes of critical band scales: Bark frequency scale and Mel frequency scale. Mel-frequency scale has been widely used in modern speech recognition system.

Mel Scale

Mel-frequency scale is a perceptually motivated scale (Stevens & Volkman, 1940) which is linear below 1 kHz, and logarithm above, with equal numbers of samples below and above 1 kHz. It represents *the pitch*[4] *(perceived frequency) of a tone as a function of its acoustics frequency* (Holmes, 2001).

One mel is defined as one thousandth of the pitch of a 1 kHz tone (Huang et al., 2001). Mel-scale frequency can be approximate by Eq. (3):

$$B(f) = 2595 \log_{10}(1 + {}^{f}/_{700}) \quad (3)$$

[4] Pitch, in psychophysics, is the perceptual correlate of the frequency of a sound wave. It means, *the pitch of a complex sound is related to its fundamental frequency, but the pitch is a subjective attribute* (Holmes & Holmes, 2001).

This non-linear transformation can be seen in Figure 9. It shows that equally spaced values on mel-frequency scale correspond to non-equally spaced frequencies. This is the inverse function of the Eq. (3) which is given by Eq. (4):

$$f = 700\left(10^{f_{mel}/2595} - 1\right) \tag{4}$$

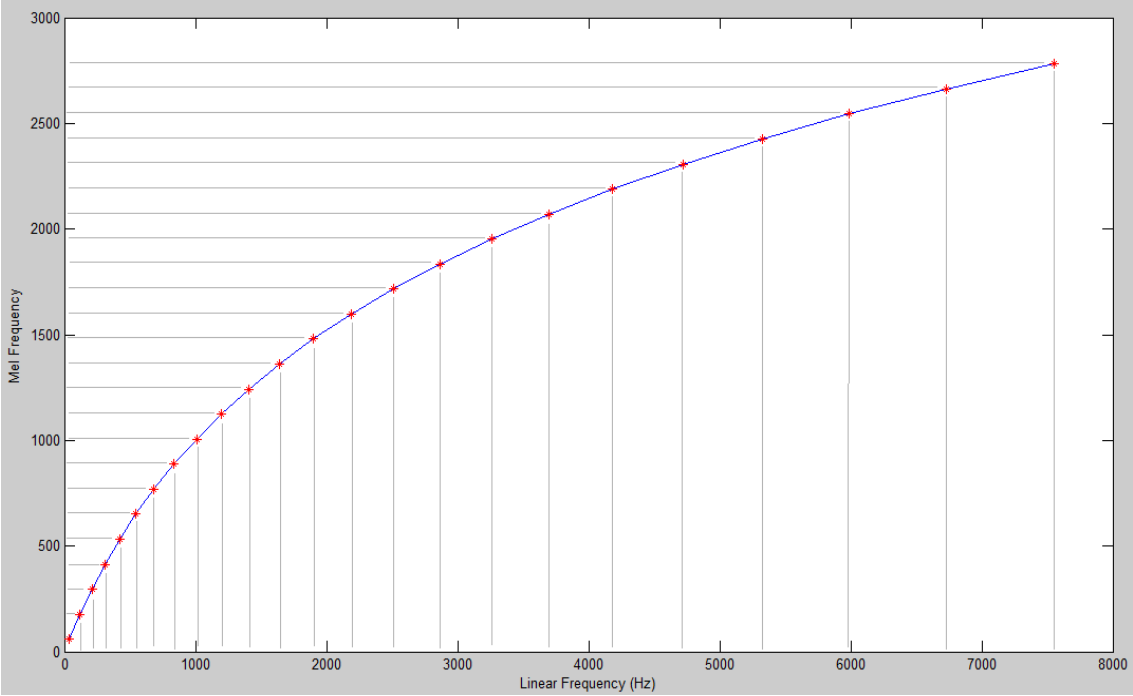

Figure 9: Mel-to-Linear Frequency scale transformation

So, it is hoped that mel scale more closely models the sensitivity of the human ear than a purely linear scale, and provides for greater discriminatory capability between speech segments.

2.1.3. Speech Signal Representation

Although some information about phonetic content can be extracted from waveforms plots, this is not useful in order to illustrate the properties of speech that are most important to the general sound quality or to perception of phonetic detail.

The large significance of resonances and their time variations, responsible for carrying the phonetic information, makes necessary to have some means of displaying these features. The short-time spectrum of the signal is more suitable for displaying such features.

Short-time Fourier Analysis

The short-time spectrum of the signal is the magnitude of a Fourier Transform of the waveform after it has been multiplied by a time window function of appropriate duration (Holmes & Holmes, 2001).

Thus, the short-time analysis is based on the decomposition of the speech signal into short speech sequences, called frames, and analysis of each one independently. For analyzing frames, the behavior (periodicity or noise-like appearance) of the signal in each one of them has to be stationary.

The width and the shape of the time window is one of the most important parameter in short-time Fourier analysis. In the figure below, the short-time spectrum of voiced speech obtained with rectangular window and Hamming window of 25ms and 10ms can be compared.

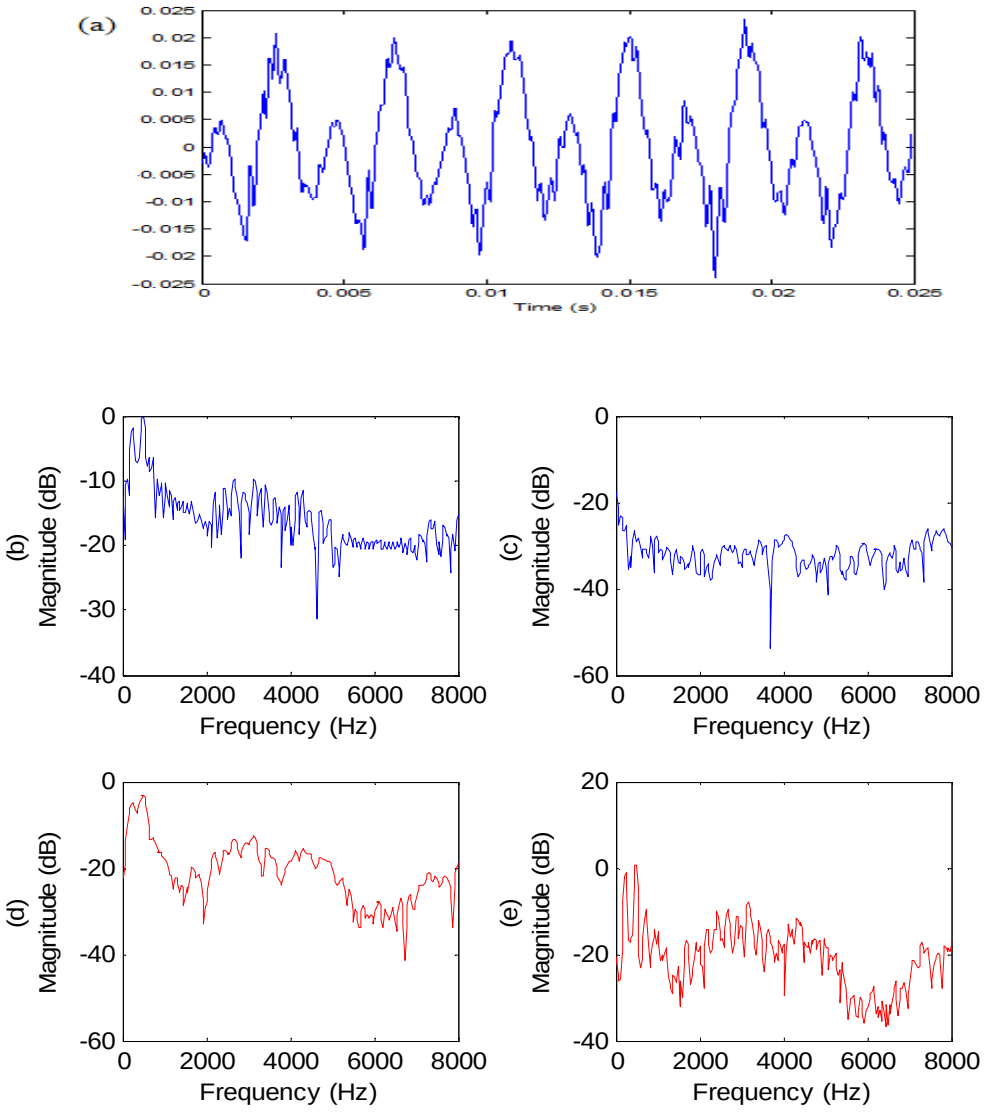

Figure 10: (a) Time signal and its Short-time spectrum obtained with: (b) 10ms rectangular window; (c) 25ms rectangular window; (d) 10 ms Hamming window; and (e) 25ms Hamming window.

One can conclude that for better stationary resolution, rectangular window is more appropriately; however, the Hamming window offers a better frequency resolution. In practice, window lengths are around 20 to 30 ms and the Hamming window is chosen. This choice is a compromise between the stationary assumption within each frame and the frequency resolution.

An efficient representation of the speech signal based on short-time Fourier analysis is spectrograms. A spectrogram of a time signal is a special two-dimensional representation that displays time in the horizontal axis and frequency in the vertical axis. Then, in order to indicate the energy in each time/frequency point, a grey scale is typically used, in which white represents low energy, and black, high energy (Huang et al., 2001). Sometimes, spectrograms can be represented by a color scale; as in Figure 11, where darkest blue parts represent low energy, and lightest red parts, high energy.

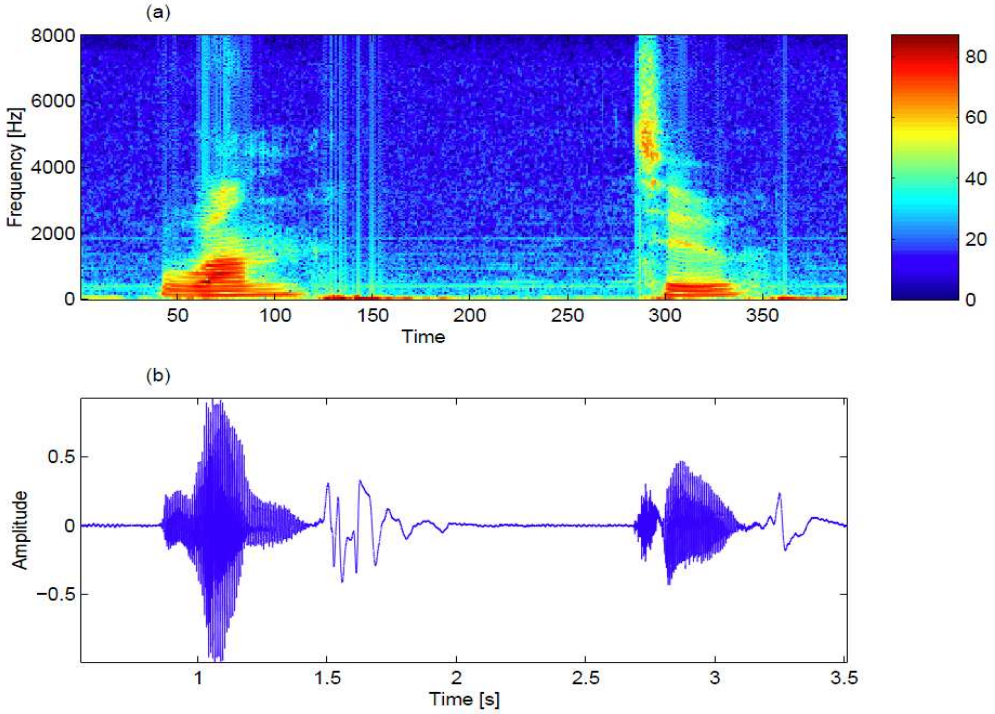

Figure 11: Spectrogram (a) of the speech waveform (b) (Nilsson & Ejnarsson, 2002)

Parametric Representation of the Spectral Analysis

When speech is produced in the sense of a time-varying signal, its characteristics can be represented via a parameterization of the spectral activity.

This speech representation is used by *front-end Automatic Speech Recognition systems*, where the frame sequence is converted into a feature vectors that contains the relevant speech information.

The main feature vectors are LPC coefficients, based on speech production models, and MFCC coefficients, based on speech perception models. They will be explained in the next section within the feature extraction process.

2.2. INTRODUCTION TO FRONT-END ANALYSIS FOR AUTOMATIC SPEECH RECOGNITION

Front-end analysis is the first stage of Automatic Speech Recognition (ASR), whereby the acoustic signal is converted into a sequence of acoustic feature vectors. Figure 12 illustrates the different stages that take place in the feature extraction process.

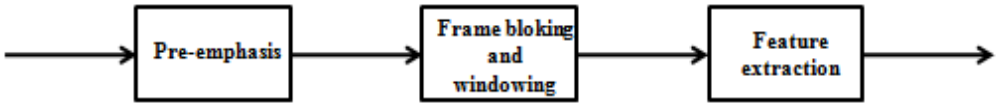

Figure 12: General feature extraction process

In this section, each stage of the above process will be explained in a subsection in order to draw a complete vision of the system.

Feature extraction stage is the most important one in the entire process, since it is responsible for extracting relevant information from the speech frames, as feature parameters or vectors. Common parameters used in speech recognition are *Linear Predictive Coding (LPC) coefficients*, and *Mel Frequency Cepstral Coefficients (MFCC)*. These parameters have been widely used in recognition system partly to the following reasons:

- The calculation of these parameter leads to a source-filter separation.
- The parameters have an analytically tractable model.
- Experience proves that these parameters work well in recognition applications.

Due to their significance, they will be described in two different subsections. Another subsection will be devoted to dynamic features. They are the *delta* and *acceleration* coefficients, that mean to add the first or second derivate approximation, respectively, to some feature parameters (LPC coefficients).

2.2.1. Pre-emphasis

In order to flatten speech spectrum, a pre-emphasis filter is used before spectral analysis. Its aim is to compensate the high-frequency part of the speech signal that was suppressed during the human sound production mechanism.

The most used filter is a high-pass FIR[5] filter described in Eq. (5), and whose transfer function corresponds to Figure 13.

$$H_{preem}(z) = 1 - a_{preem}z^{-1} \tag{5}$$

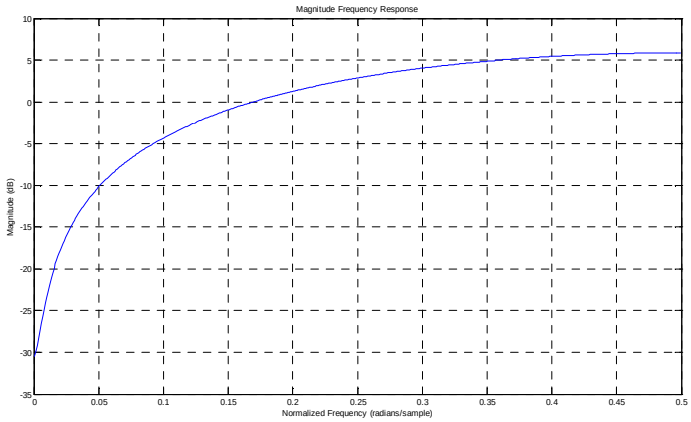

Figure 13: Pre-emphasis Filter, a=0.97.

2.2.2. Frame Blocking and Windowing

As explained in Section 2.1.3, the speech signal is divided into a sequence of frames where each frame can be analyzed independently and represented by a single feature vector. Since each frame is supposed to have stationary behaviour, a compromise, in order to make the frame blocking, is to use a 20-25 ms window applied at 10 ms intervals (frame rate of 100 frames/s and overlap between adjacent windows of about 50%), as Holmes & Holmes exposed in 2001. One can see this in Figure 14.

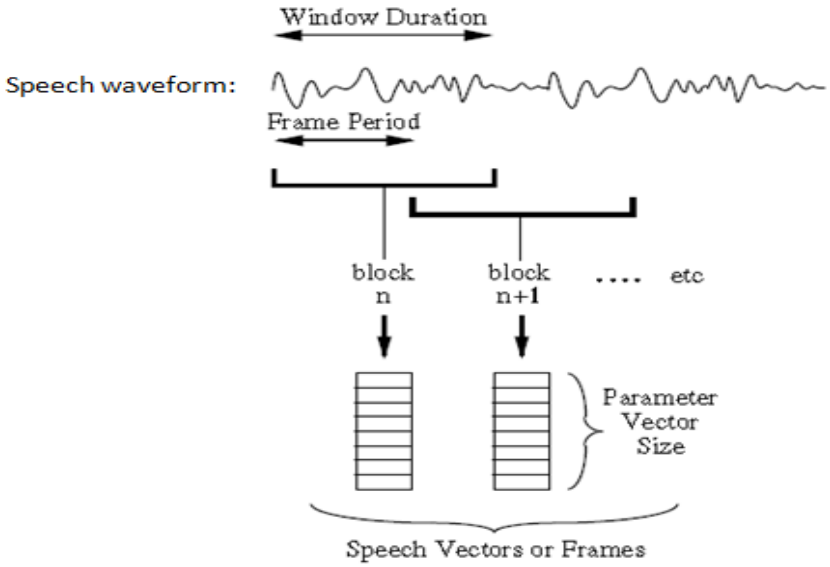

Figure 14: Frame blocking (Young et al., 2006)

[5] FIR = Finite Impulse Response.

In order to reduce the discontinuities of the speech signal at the edges of each frame, a tapered window is applied to each one. The most common used window is Hamming window, described in Eq. (6) and shown in Figure 15.

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi(n-1)}{\mathrm{N}-1}\right) \tag{6}$$

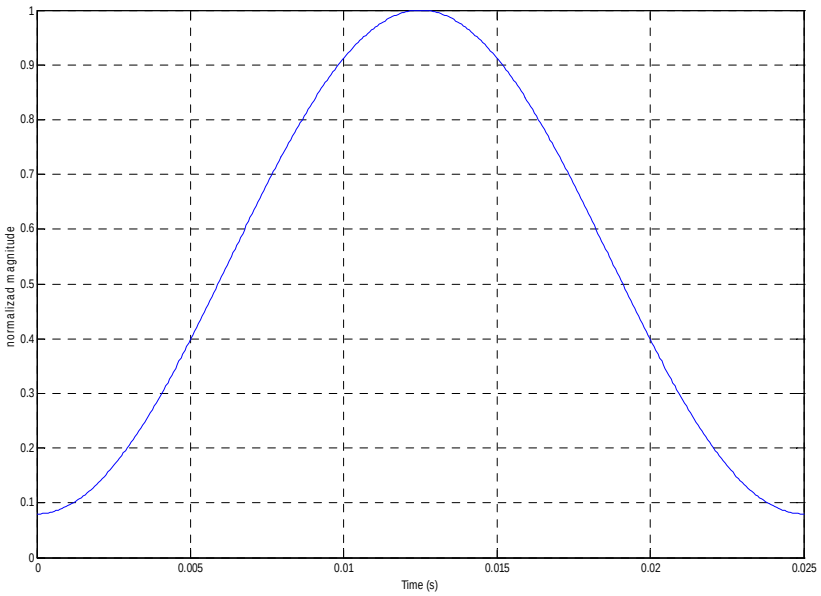

Figure 15: 25ms Hamming window (fs=16Khz)

2.2.3. Mel-Cepstrum

Davis & Mermelstein (1980) pointed the Mel Frequency Cepstrum[6] Coefficients (MFCC) representation as a beneficial approach for speech recognition (Huang et al., 2001). *The MFCC is a representation of the speech signal defined as the real cepstrum of a windowed short-time signal derived from the FFT of that signal* (Huang et al, 2001) which, is first subjected to a log-based transform of the frequency axis (mel-frequency scale), and then decorrelated using a modified Discrete Cosine Transform (DCT-II). Figure 16 illustrates the complete process to extract the MFFC vectors from the speech signal. It is to be emphasized that the process of MFCC extraction is applied over each frame of speech signal independently.

[6] Cepstrum is the inverse Fourier Transform of the log-spectrum. The name comes from to reversing the first syllable of the word spectrum and was invented by Bogert et al. (1963).

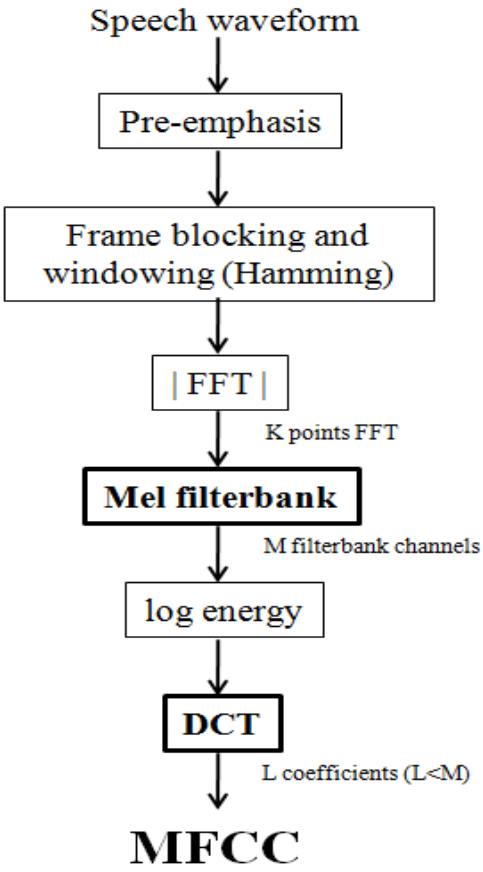

Figure 16: MFCC extraction process

After the pre-emphasis and the frame blocking and windowing stage, the MFCC vectors will be obtained from each speech frame. The process of MFFC extraction will be described below considering in any instant that all the stages are being applied over speech frames.

The first step of MFCC extraction process is to compute the Fast Fourier Transform (FFT) of each frame and obtain its magnitude. The FFT is a computationally efficient algorithm of the Discrete Fourier Transform (DFT). If the length of the FFT, is a power of two ($K=2^n$), a faster algorithm can be used, so a zero-padding to the nearest power of two within speech frame length is performed.

The next step will be to adapt the frequency resolution to a perceptual frequency scale which satisfies the properties of the human ears (Molau et al., 2001), such as a perceptually mel-frequency scale. This issue corresponds to Mel filterbank stage.

The filter-bank analysis consists of *a set of bandpass filter whose bandwidths and spacings are roughly equal to those of critical bands and whose range of the centre frequencies covers the most important frequencies for speech perception* (Holmes & Holmes, 2001).

The filterbank is a set of overlapping triangular bandpass filter, that according to mel-frequency scale, the centre frequencies of these filters are linear equally-spaced below 1 kHz and logarithmic equally-spaced above. The mel filterbank is illustrated in Figure 17**.** It is interesting to emphasize that these centre frequencies correspond to mel centre frequencies uniformly spaced on mel-frequency domain, as was shown in Figure 9 in Section 2.1.2.

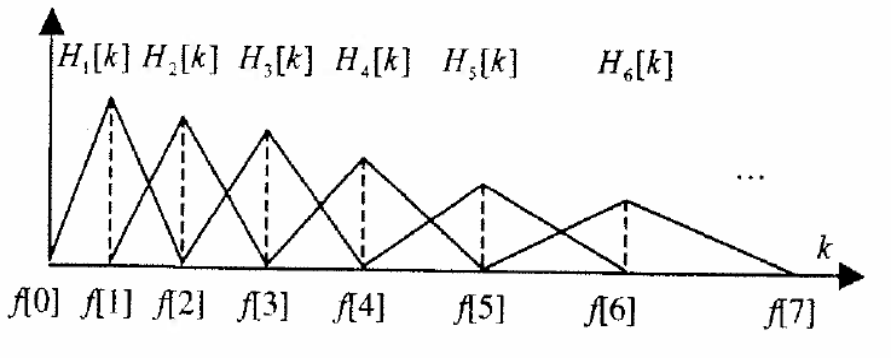

Figure 17: Mel filterbank (Huang et al., 2001)

Thus, the input to the mel filterbank is the power spectrum of each frame, $X_{frame}[k]$, such that for each frame a log-spectral-energy vector, $E_{frame}[m]$, is obtained as output of the filterbank analysis. Such log-spectral-energy vector contains the energies at centre frequency of each filter. So, the filterbank samples the spectrum of the speech frame at its centre frequencies that conform the mel-frequency scale.

Let's define $H_m[k]$ to be the transfer function of the filter m, the log-spectral-energy at the output of each filter can be computed as in Eq. (7) (Huang et al., 2001); where M (m=1, 2, ..., M) is the number of mel filterbank channels. M can vary for different implementations from 24 to 40 (Huang et al., 2001).

$$E[m] = \sum_{k=1}^{K-1} \ln[|X[k]|^2 H_m[k]] \qquad \text{m=1, 2, ..., M} \quad (7)$$

The choice of the filterbank energies as input of filterbank analysis has been widely used in early recognition system. However, another approaches based on further transformations have been nowadays proposed to gain substantial advantages respect the filterbank energies input (Holmes & Holmes, 2001).

Using the mel filterbank is subjected to two principal reasons:

- Smooth the magnitude spectrum such that the pitch of a speech signal is generally not presented in MFCCs.
- Reduce the size of the features involved.

The last step involved in the extraction process of MFCC is to apply the modified DCT to the log-spectral-energy vector, obtained as input of mel filterbank, resulting in the desired set of coefficients called Mel Frequency Cepstral Coefficients.

The most widely DCT used for speech processing is DCT-II because of its energy compaction, which results in its coefficients being more concentrated at lower indices than the DFT. This property allows approximating the speech signal with fewer coefficients (Huang et al., 2001).

In order to compute the MFCCs for one frame, the DCT-II is applied to the log-spectral-energy vector of such frame and is given by Eq. (8) (Young et al., 2006)[7]:

$$c_i = \sqrt{\frac{2}{\mathrm{M}}} \sum_{\mathrm{m}=1}^{\mathrm{M}} \mathrm{E_m} \cos\left(\frac{\pi \mathrm{i}}{\mathrm{M}}\left(\mathrm{m} - \frac{1}{2}\right)\right) \tag{8}$$

Cepstral coefficients have the property that both the variance and the average numerical values decrease as the coefficient index increases (Holmes & Holmes, 2001). The zero cepstral coefficient, c_0, is proportional to the mean of the log spectral energy channels and *provides an indication of overall level for the speech frame* (Holmes & Holmes, 2001).

As explained above, discarding the higher cepstral coefficients can be advantageous. In this way, the M filterbank channels can be become into only L MFCCs (L < M) used in the final feature vector. *The truncation of the cepstral sequence has a general spectral smoothing effect that is normally desirable because it tends to remove phonetically irrelevant detail* (Holmes & Holmes, 2001).

Although MFCC vectors is a beneficial approach as feature vectors for speech recognition, the extraction of them from speech signal involves much loss information due to the followings reasons:

- Phase information is removed at the magnitude operation.
- The filtering process reduces the initial frequency resolution obtained from the FFT and more spectral detail is lost due to the truncation from M to L (L<M) coefficients after the DCT stage.

2.2.4. Linear Prediction

In order to represent the short-time spectrum, there is another alternative to filterbank analysis based on deriving linear prediction coefficients which comes from Linear Predictive Coding (LPC) analysis (Holmes & Holmes, 2001). LPC analysis is an effective method to estimate the main parameters of speech signals.

[7] The HTK uses DCT-II to compute the MFCC.

In Section 2.1.1, the source-filter model for speech production was presented and finally schematized in Figure 7**.** The conclusion extracted was that an all-pole filter, H(z) in Figure 7, is a good approximation to estimate the speech signals. Its transfer function was described by Eq. (1). In this way, from the filter parameters (coefficients, $\{a_i\}$; and gain, G), the speech samples could be synthesized by a difference equation given by Eq. (2).

Thus, the speech signal resulting from Eq. (2) can be seen as *linear combination of the previous p samples. Therefore, the speech production model can be often called linear prediction model, or the autoregressive model* (Mammone et al., 1996). From here, *p*, in Eq. (1) and (2), indicates the order of the LPC analysis; and, the excitation signal, e[n], of the speech production model can be called *prediction error signal* or *residual signal* for LPC analysis.

The LPC coefficients, as well for MFFC coefficients, are obtained for each frame independently one of each other.

According to Eq. (2), the prediction error, E_m, for one frame can be defined in Eq. (9) as (Huang et al., 2001):

$$E_m = \sum_n e_m^2[n] = \sum_n \left(x_m[n] - \sum_{j=1}^{p} a_j x_m[n-j] \right)^2 \qquad (9)$$

where $x_m[n]$ is a frame of the speech signal and *p* the order of the LPC analysis. For one speech frame its LPC coefficients are estimated *as those that minimize the prediction error* E_m (Huang et al., 2001).

Estimating LPC coefficients from speech frame, the *orthogonality principle*[8] is assumed and the Yule Walker Equations are obtained:

$$\sum_{j=1}^{p} a_j \phi_m[i,j] = \phi_m[i,0] \qquad i=1, 2, ..., p \qquad (10)$$

where $\Phi_m[i,j]$ is the correlation coefficients defined as:

$$\phi_m[i,j] = \sum_n x_m[m-i] x_m[n-j] \qquad (11)$$

Solution of the *p* linear equations gives the *p* LPC coefficients that minimize the prediction error, such that the set of $\{a_i\}$ satisfies Eq. (2) to generate the speech signal through speech production model.

[8] *Orthogonality principle says that the predictor coefficients that minimize the predictor error are such that the error must be orthogonal to the past vectors* (Huang et al, 2001).

In order to resolve the Yule Walker Equations, different algorithms can be presented: *the covariance method*, *the autocorrelation method* and *the lattice method*. The algorithm that will be used in this Master Thesis will be the autocorrelation method.

The autocorrelation method corresponds to resolve a basic matrix equation expressed as Eq. (12), where R is the autocorrelation matrix of the speech signal ($R(i,j)=R_{xx}(|i-j|)$); r is the autocorrelation vector of the speech signal ($r(i)=R_{xx}(i)$) and a is the vector of the LPC coefficients.

$$R \times a = r \tag{12}$$

This matrix equation is resolved by *Levinson-Durbin recursion* algorithm in which the recursion finds the solution of all prediction coefficients of order less than p. In the computing of this algorithm, other intermediate variables, called *reflection coefficients, $\{k_i\}$*, are calculated.

Finally, Figure 18 illustrates the extraction process of the LPC coefficients.

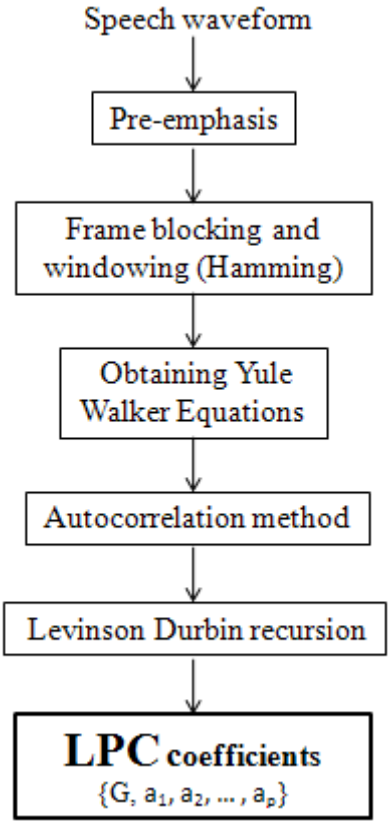

Figure 18: LPC coefficients extraction process

After the LPC analysis, the power spectrum of the speech frame can be calculated from its LPC parameters. Let's define A(z) to be the inverse transfer function of the filter given by Eq. (1) as:

$$A(z) = 1 - \sum_{i=1}^{p} a_i \, z^{-i} \tag{13}$$

From this inverse filter, A(z), a new speech synthesis model is proposed in Figure 19**,** which can be considered as inverse model of speech production model described on Figure 7**.**

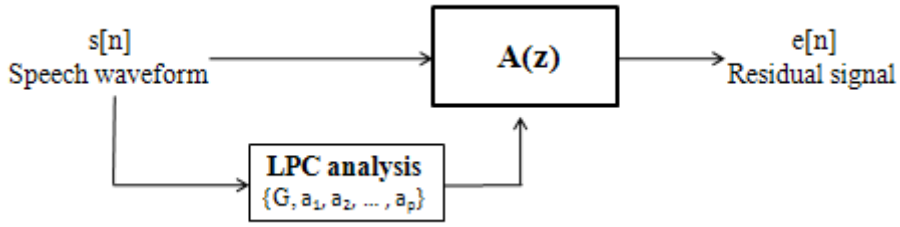

Figure 19: Synthesis LPC filter

The power spectrum of one signal can be obtained by passing one input signal through a filter. If the input signal is the speech signal and the filter is the inverse LPC filter A(z); the power spectrum of the output signal, in this case the residual signal or prediction error signal, can be obtained as:

$$S(\omega)|A(\omega)|^2 = \sigma_\omega^2 \tag{14}$$

Then, one can see that the power spectrum of the speech signal can be approximated by the *response of a sampled-data-filter, whose all-pole-filter transfer function is chosen to give a least-squared error in waveform prediction* (Holmes & Holmes, 2001). So, in Eq. (15), the power spectrum of the speech frame is obtained from its LPC coefficieents.

$$S(\omega) \approx \frac{\sigma_\omega^2}{|1 - \sum_{i=1}^{p} a_i \, e^{-j\omega i}|^2} \tag{15}$$

LPC analisys *produces an estimate smoothed spectrum, which much of the influence in the excitation removed* (Holmes & Holmes, 2001).

LPC-derived features have been used by many recognition systems, being its performance comparable whit the one obtained from recognizers using filterbank methods. However, later, LPC-derived cepstral coeffcients has begun to be considered since the addtion of ceptral tranformation improves recogntion performance (Holmes & Holmes, 2001).

Ending this section, It bears mentioning another set of features vector called *Perceptual Linear Prediction* (PLP) coefficients (Hermansky, 1990). PLP analysis is based on LPC analysis incorporating a non-linear frequency scale and other psychophysics properties of the human perception system.

PLP analysis is more similar to MFCC analysis, but the incorporation of more perceptual properties makes it more related to psychophysical results. In Table 1**,** the comparison between the properties of both methods can be seen.

Table 1: Comparison between the properties of MFCC and PLP coefficients

MFCCs	**PLP coefficients**
Cepstrum-based spectral smoothing	LPC-based spectral smoothing
Pre-emphasis applied to speech waveform	Pre-emphasis applied to spectrum
Triangular mel filterbank	Critical-band filterbank
Logarithmic amplitude compression	Cube root amplitude compression

2.2.5. Dynamic Features: Delta Coefficients

In order to improve the recognition performance, a new stage in the feature extraction process can be added, see Figure 20**.** *Dynamic features* consist of the incorporation of temporal derivatives to the feature vectors obtained in the last stage.

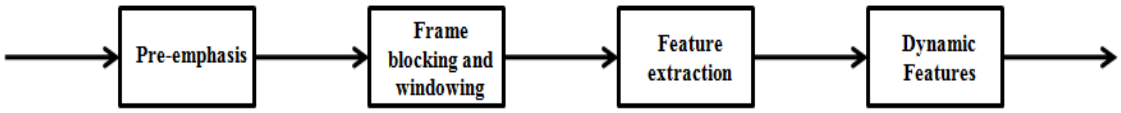

Figure 20: Feature vectors extraction and its dynamic features

As explained in Section 2.1.3**,** the speech signal is converted into a sequence of speech frames such that each one of them is assumed stationary in its short interval. Therefore, each frame can be analyzed independently and represented by an independent single feature vector.

In spite of the above assumption, *an acoustic feature vector representing part of a speech signal is highly correlated with its neighbors* (Holmes & Holmes, 2001). However, these correlations can be captured applying the dynamic features to the *static* feature vectors (such as MFFCs or LPC coefficients); since they can measure the change in the *static* features (Holmes & Holmes, 2001).

The dynamic features referred to the first order time derivatives are called as *delta coefficients* and to the second order time derivatives as *acceleration coefficients.*

The dynamics features can be computed *by simple differencing between the feature values for two frames either side of the current frame* (Holmes & Holmes, 2001). However, differencing method results more sensitive to random fluctuations in the original features and tends to be noisy. That is why another robust dynamic measure can be applied based on linear regression over a sequence of frames (Holmes & Holmes, 2001).

As said before, including the dynamics features (first-second order time derivatives) generally improves recognition performance. Delta features gives a large gain on this improvement while the acceleration features adds a smaller one.

The majority recognition systems incorporate dynamic features applied, generally, to a set of MFCC vectors or LPC coefficient vectors.

2.3. DISTANCE MEASURE FOR SPEECH PROCESSING: RMS LOG SPECTRAL MEASURE

After the front-end analysis in automatic speech recognition, thereby the speech signal is converted into a sequence of feature vectors; the next issue is to measure the quality of the recovered speech from such features vectors. The measures of interest in this Master Thesis are those called *distance measures*, especially *spectral distance measures* between the smoothed spectrums obtained from the feature vectors, LPC coefficients or MFCCs.

As explained in past sections, an estimate power spectrum of the speech frame can be obtained from its features vectors. LPC coefficients provide an estimate smoothed spectrum of the speech signal according to Eq. (15). In another hand, the MFCCs transform the FFT of the speech frame to a perceptually mel-frequency scale, using the mel filterbank method. The result is a smoothed magnitude mel spectrum in which the harmonics have been flattened in order to obtain the envelope of the spectrum of the original speech frame. Also the LPC coefficients can be derived from the MFFC coefficients and estimating the spectrum from Eq. (15).

Distance measures based upon transformations retain only the smoothed spectral behavior of the speech signal have been applied in recognition tasks (Gray et al., 1976*). So,* the *distance measure* called *root mean square (rms) log spectral measure* will be used in the implementation work of this Master Thesis, to measure the spectral distortion between the spectrums obtained from feature vectors.

RMS Log Spectral Measure

Let's define S_{xx} and S'_{xx} to be two spectral models. The error o difference between them is defined by (Gray & Markel, 1976):

$$V(\omega) = \ln S_{xx}(\omega) - \ln S'_{xx}(\omega) \tag{16}$$

In order to measure the distance between these spectral models, a set of L_p norms has been defined as d_p by (Gray & Markel, 1976):

$$(d_p)^p = \int_{-\pi}^{\pi} |V(\omega)|^p \frac{\mathrm{d}\omega}{2\pi} \tag{17}$$

the *rms log spectral measure* is defined when p takes the value of 2. The L_p norm is typically evaluated for smoothed spectra models, as the smoothed power spectrum computed from LPC coefficients. Then, the two spectral models S_{xx} and S'_{xx} are defined according to Eq. (15).

The L_p measures are related to decibel variations in the log spectra domain by using the multiplicative factor 10/ln(10) = 4.34.

3. IMPLEMENTATION: GENERATIVE MODEL OF SPEECH

The work developed in this Master Thesis consists of the implementation of a speech generative model; whereby the speech is synthesized and recovered from its MFCC representation. Synthesizing speech from parametric representations allows performing an investigation on the intelligibility of the synthesized speech as compared to natural speech. So, two steps were performed: the computation of the MFCCs vectors from the speech signal and the generative model in itself.

The tool used for computing the MFCCs from the speech signal in this Master Thesis was the *HTK Software Toolkit*, developed by the Cambridge University (Young et al., 2006). The speech signal processing, based on the conversion chain from HTK-generated MFCC representation to the generative model, was supported by the mathematical tool *Matlab*.

This conversion chain tends to simulate the inverse process to the feature extraction process described in Section 2.2, in order to recover the speech signal, and measure how much information relevant to speech recognition is lost in this analysis.

This section is divided into two sections. The first one offers an introduction to the HTK Software and an investigation on the implementation of the MFCC computation in HTK. The second one describes all the steps that take places in the generative model, making relationships with the theories-based, and giving an explanation of the approximation methods and algorithms used in its implementation. The results will be put forward in the Analysis of Results and Discussion Section.

3.1. MFCC COMPUTATION IN HTK

The initials of HTK correspond to *Hidden Markov Toolkit*, which is a standard research and development tool for building *Hidden Markov Models (HMM)* based on speech recognition.

The HMM has become one of the most powerful statistical methods for modeling the speech signals and its principles have been successfully used in automatic speech recognition (Huang et al., 2001). For the work developed in this Master Thesis, no more details about the concept of HMM are required.

The Software architecture of HTK is principally built over library modules with which HTK tools operate. The HTK tools are executed from commands into the operating system shell. These commands consist of a number of input files and optional input arguments (in order to control more in detail the behavior of the tool). Every tool uses a set of standard library modules that act as an interface between various files types and with outside world. These modules are, usually, customized by setting parameters in a configuration file. Finally, there is another set of parameters that are specified by using environment variables (Young, et al., 2006).

In the MFCC computation, the HTK tool *HCopy* and the *configuration file* play an important role in the parameterization of the speech signal into a sequence of feature vectors. The next sections will be focus in these items.

3.1.1. HCopy and Coding Speech Data to MFCC Vectors

HCopy is the tool of HTK responsible for copying and manipulating the speech files (Young et al., 2006).

By specifying an appropriate configuration file, HCopy can be seen as a speech coding tool, available to parameterize the speech signal into a sequence of feature vectors. Thus, HCopy parameterizes the source speech data according to the configuration file, and copies the target speech data into the output file. This is schematized in Figure 21.

For the experimental work of this Master Thesis, a set of 10 audio files on waveform format (*files.wav*) was given as source speech data. The texts of these utterances are contained in Appendix A**.** The speech files were taken at random from the TIMIT database[9]. The configuration file created was named *hcopy.conf* and was set-up to convert source waveform data to MFCC coefficients. Every output file, generated by HCopy, contains the MFCC vectors of its corresponding source waveform file. The source waveform files used and the output HCopy-generated MFCC files are listed in the Figure 21.

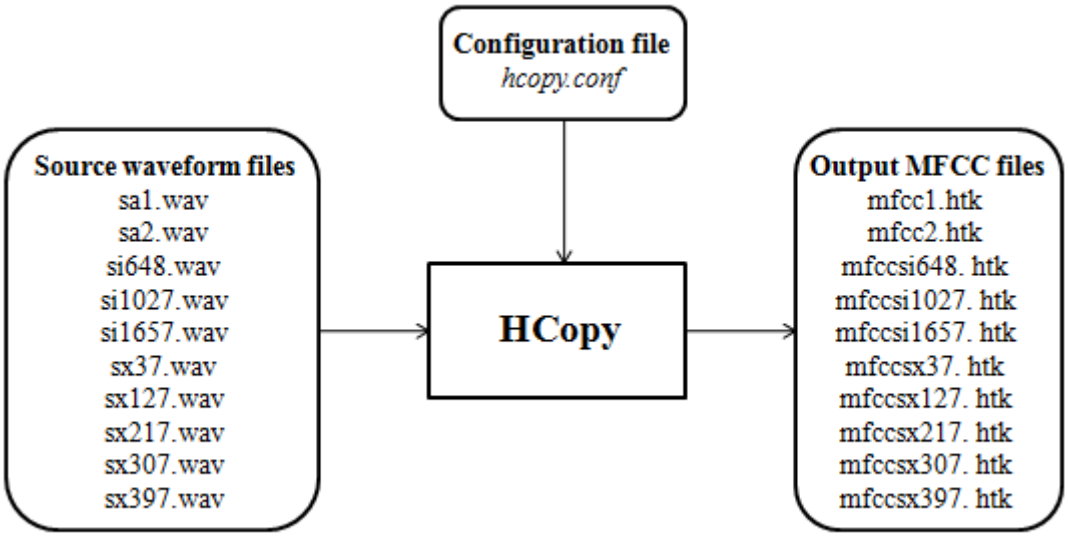

Figure 21: Parameterization of the speech data by HCopy; and list of the source waveform files and its corresponding MFCC files generated

[9] TIMIT database is a standard database of utterances examples for speech recognition experiments.

The configuration parameters within *hcopy.conf* file will specify the characteristics of the MFCC extraction process seen in Figure 16. Once it was created, the coding data from the source speech data to MFCC vectors was done by executing the following command line in a Linux shell:

HCopy –C hcopy.conf source-file mfcc-file

where *–C* is a standard option used to indicate that *hcopy.conf* file is the configuration file used. The *source-file* argument input indicates the file name that contains the source speech data; and, the *mfcc-file* argument input is the file name where HTK will copy the output MFCC vectors data. HCopy was run for every pair of the source waveform file and its output file.

3.1.2. Configuration File to Compute the MFCC Vectors

As said before, the configuration file created was called *hcopy.conf* and is added in Appendix B. The generative model was based on the way in that the MFCC vectors were computed by HTK; i.e., the whole speech signal processing and the characteristics of the generative model are controlled by configuration parameters.

The MFCC extraction process of Figure 16 will be followed in the description of the most important configuration parameters. Table 2 shows the setting of those configuration parameters related to such MFCC extraction process.

Table 2: Configuration parameters related to MFCC extraction process
(* expressed on units of 100ns)

Configuration parameters	Value	
SOURCEKIND	waveform	**MFCC extraction process**
SOURCERATE	625*	
TARGETKIND	mfcc_0	
PREEMCOEF	0.97	Pre-emphasis
TARGETRATE	100000*	Frame blocking and Hamming windowing
WINDOWSIZE	250000.0*	
USEHAMMING	true	
NUMCHANS	24	Filterbank and MFCC coefficients
NUMCEPS	12	

The *sourcekind* and *targetkind* are the configuration parameters used to indicate which parameterization should be done; since, they define the source and the target parameter kinds. Using *hcopy.conf* file, with the configuration above, the waveform data was converted to MFCC_0 using the C_0 as the energy component. The sample frequency of the source waveform data was set by using the configuration parameter *sourcerate* to 16 kHz.

As it was seen in Section 2.2.1, a pre-emphasis filter is required before spectral analysis. The value of this pre-emphasis filter coefficient in Eq. (5) was specified by setting the parameter *preemcoef* to 0.97.

The process of the frame blocking and Hamming windowing (see Figures 14 and 15) is described by the followings configuration parameters: *targetrate, windowsize* and *usehamming*. They were configured to apply a Hamming window of 25ms every 10ms (frame rate of 100frames/s), resulting an overlap of 15ms. If the frequency sample was 16 kHz, the size of the frames generated was of 400 samples (25ms * 16 kHz) with an overlap of the 60%. The Hamming window performed by HTK corresponds to the one described by Eq. (6).

Finally, for computing the MFCC coefficients, HTK *provides a simple Fourier Transform based filterbank* (Young et al., 2006) method and calculates the MFCCs using the DCT-II described by Eq. (8). The number of the filterbank channels was set by the configuration parameter *numchans* to 24. As it was explained in Section 2.2.3, they are equally spaced along the mel-frequency scale (see Figure 9). The number of the MFCC coefficients was specified by the configuration parameter *numceps* to 12.

In short, the source speech signal is passed through a first order pre-emphasis filter with a coefficient of 0.97. The FFT should use a Hamming window of 25ms with a frame period of 10ms. The filterbank has 24 channels and for each speech frame 13 components (12 MFCC coefficients plus the C_0 component) are generated and copied in the output file.

3.2. GENERATIVE MODEL

The generative model in itself is the conversion chain which synthesizes speech from HTK-generated MFCC representation.

All the speech processing involved within the implementation of the generative model was support by Matlab. In order to process speech signals, successful Matlab toolbox were incorporated, such as *Voicebox*[10] and *Auditory Toolbox*[11].

[10] Brookes, M., *Voicebox: Speech Processing Toolbox for Matlab* [on line], Imperial College, London, available on the World Wide Web: http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html.
[11] Slaney, M. (1998), *Auditory Toolbox* [on line], Interval Research Corporation, California, available on the World Wide Web: http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/.

The generative model was implemented in a Matlab function called *generative_model.m*, enclosed in Appendix C**.** The algorithm can be divided into five steps (schematized in Figure 22) in order to be explained.

Firstly, The MFCC files were exported to Matlab, extracting the MFCC vectors. Secondly, in order to further on implementing the source-filter model for speech production (see Section 2.1.1, Figure 7), the LPC coefficients were computed from the MFCCs vectors. In this process an inverse DCT (IDCT) had to be approximated. Thirdly, the source-filter model for speech production of Figure 7 was implemented. The filter was estimated by the LPC coefficients computed from MFCCs according to Eq. (1); and the excitation signal was modeled for voiced and unvoiced sounds. Finally, the speech signal produced was filtered by an inverse pre-emphasis filter (de-emphasized filter). Moreover, the dynamic features (delta coefficients) were added to LPC coefficients in order to achieve better performance recognition.

For the speech processing, a set of constants were defined according to the above configuration parameters. The based theories and investigations to implement the algorithms in *generative_model.m* function will be explained in detail.

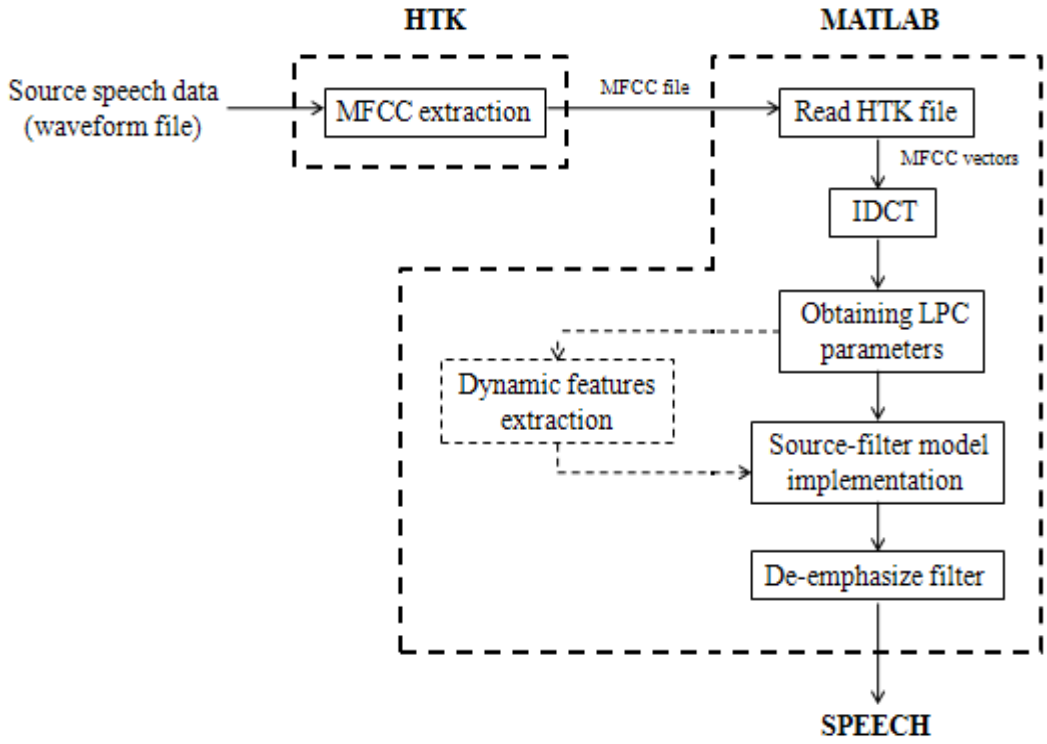

Figure 22: Conversion chain from HTK-generated MFCC representation to the generative model

The *generative_model.m* function contains other secondary functions with the algorithms of the different items to be implemented. They are: *idct_htk.m, mfcc2spectrum.m, mfcc2spectrum2.m, LPC_filter.m* and *deltacoef.m*. They are enclosed in Appendix D.

3.2.1. Conversion from MFCC Vectors to LPC Parameters

The process followed in the conversion from MFCC vectors to PC parameters is described below:

1. Read the MFCC vectors from the MFCC file generated by HTK.
2. Make the IDCT of MFCCs vectors to obtain the mel log power spectrum.
3. Exponentiation to obtain the mel power spectrum.
4. Convert the mel power spectrum to power spectrum on linear frequency domain.
5. Apply the inverse Fourier Transform (FT) to get the autocorrelation function estimate.
6. Solve the Yule Walker equations by autocorrelation method to obtain the LPC coefficients. They are the filter coefficients.

The entire process above is implemented in the functions *mfcc2spectrum.m* and *mfcc2spectrum2.m* (see the code in Appendix D). The difference between these two algorithms falls on that the extraction of the autocorrelation coefficients from the mel power spectrum was dealt by two different ways. This generated two approaches for the generative model that will be explained after. Each one can be chosen by an input argument in the *generative_model.m* function (see code of *generative_model.m* in Appendix C).

For reading the HTK-MFCC files, the function *readhtk.m* from Voicebox library was used. The MFCC vectors were stored in a matrix called *mfcc* of size [13, number-frames] (12 MFCC coefficients plus the C_0 component were generated per frame).

Inverse Discrete Cosine Transform

The IDCT had to be approximated since a direct transform was not possible because of the number of MFCC coefficients was lower than the number of filterbank channels.

The DCT employed by HTK to compute the MFFC coefficients is the DCT-II given by Eq. (8), in Section 2.2.3. It was explained that the DCT-II is applied to the log spectral energies at frequencies uniformly sampled on mel-frequency domain, to produce the MFCCs. So, the inverse transform of K points will provide K samples of mel log power spectrum at frequencies equally spaced on mel-frequency domain.

The inverse transform of DCT-II is given below, assuming that c_0 is computed in the same fashion as c_n, n>0.

$$S_{yy}(k) = \sqrt{\frac{2}{K}} \sum_{n=1}^{N} c(n) \cos\left(\frac{\pi k}{K}\left(n - {}^{1}/_{2}\right)\right) \qquad (18)$$

where N=13 is the number of MFCC coefficients including the c_0 component, and K is the number of points on mel-frequency domain required for the spectrum reconstruction. Comparing with the forward transform in Eq. (8), one can see that in Eq. (18) the term into the cosine is divided by K instead of N (K>N and K>M, where M is the number of filterbank channels). This fact acts like an interpolation in order to get a smoothed mel log power spectrum. That's why K has to be in the interval [129, 256] points. It was chosen to be 256 points (a number of points power of two allows an efficient algorithm for the inverse Fourier Transform).

The K points on mel-frequency domain represent the K centre frequencies of the mel filterbank. They are easily calculated dividing the mel bandwidth (calculate the mel frequency corresponding to the Nyquist sample frequency, $f_s/2$, by Eq. (3)) between K and taking the mel centre frequencies.

The algorithm that makes the inverse DCT, inspired as HTK performs the DCT-II, is written in the function *idct_htk.m* (see code in Appendix D) according to the method below. The *idct_htk.m* function is called from the functions *mfcc2spectrum.m* and *mfcc2spectrum2.m*.

The forward transform can be performed as a matrix multiplication c=AE, where E is the log spectral energies and A the transform matrix whose elements are defined as:

$$\{a_{ij}\} = \sqrt{\frac{2}{K}}\cos\left(\frac{\pi(i-1)}{K}\left(j - {}^{1}/_{2}\right)\right) \qquad \begin{matrix} \text{i=1,...,N} \\ \text{j=1,...,K} \end{matrix} \tag{19}$$

Then, the inverse transform matrix A^{-1} can be calculated by Eq. (20), where D is a diagonal matrix as D = {1/2, 1, 1, … , 1}.

$$A^{-1} = A^T D \tag{20}$$

Finally, if *c* is the vector of MFCC coefficients of one frame, the inverse DCT is performed as:

$$S_{yy} = A' D c \tag{21}$$

In short, the inverse DCT provides samples of mel log power spectrum uniformly spaced on the bandwidth of mel-frequency domain, which according to Eq. (4) corresponds to samples no uniformly spaced on linear-frequency domain.

Finally, the mel power spectrum is basically done by making the exponentiation to the log mel power spectrum.

Autocorrelation Function Estimate and LPC Coefficients

In Section 2.2.4, the LPC analysis was explained. According to this analysis, the LPC coefficients can be calculated by solving the Yule-Walker Equations described by Eq. (10), using the autocorrelation method. By this method, the Yule Walker Equations are transformed to a basic matrix equation expressed by Eq. (12), which needs that the autocorrelation function estimate is to be approximated. This matrix equation is resolved finally by the Levinson Durbin's algorithm to obtain the LPC coefficients.

It is studied that the power spectrum is the FT of the autocorrelation function. So, the autocorrelation coefficients of the speech signal are given by the inverse FT of its power spectrum as:

$$r[n] = \frac{1}{K}\sum S_{xx}(f)e^{j2\pi fn} \tag{22}$$

However, the mel power spectrum obtained by the inverse DCT has samples related to frequencies non uniformly spaced on the linear frequency scale. This fact makes that the inverse FT of Eq. (22) cannot be directly applied to the mel power spectrum.

As said before, the functions *mfcc2spectrum.m* and *mfcc2spectrum2.m* contains two different algorithms to approximate the autocorrelation function and solving this problem; so, they can define two different approaches for a generative model:

- Generative model 1: *mfcc2spectrum.m*

  The algorithm of this function is a lineal interpolation of the mel power spectrum in order to find the sample values at the frequencies uniformly spaced on linear frequency. With a mel power spectrum samples equally spaced on linear frequency, the inverse FT described by Eq. (22) can be performed to obtain the autocorrelation coefficients.
  The number of equally spaced linear frequency points was fixed up to 256 points in order to get a smooth spectral representation.

- Generative model 2: *mfcc2spectrum2.m*

  In this algorithm, instead of obtaining equally spaced samples of mel power spectrum, the inverse FT was applied to the non equally mel power spectrum samples considering the bandwidth at each mel frequencies to obtain the autocorrelation coefficients.
  In the figure of the filterbank representation in a linear frequency scale (see Figure 17), one can see that the filter bandwidth is wider as higher centre

frequency. So, each mel centre frequency has a different bandwidth on the linear frequency scale.

It was commented before that the K mel power spectrum samples provided by the inverse DCT are related at the centre frequencies of the filterbank on a linear frequency scale. Let's define Δf_k to be the bandwidth of the filter at the frequency f_k, the inverse FT can be applied directly to the mel power spectrum obtained considering the bandwidth at each frequency as:

$$r[n] = \frac{1}{K}\sum S_{xx}(f_k)e^{j2\pi f_k n}\Delta f_k \qquad (23)$$

This is the generalization of Eq. (22), in which the frequency increase, Δf_k, is normalized to one because of the samples are assumed to be uniformly spaced on a linear frequency domain.

When the autocorrelation coefficients are estimated, the matrix equation of Eq. (12) can be solved by the Levinson Durbin's algorithm to obtain the LPC coefficients. This algorithm is implemented by a Matlab function called *levinson.m* and returns the LPC coefficients and the filter gain. This function was used in the generative model algorithm for a p^{th} order LPC coefficients equal to 12 (see *mfcc2spetrum.m* and *mfcc2spectrum2.m* in Appendix D).

Finally, according to the explanation in Section 2.2.4, the power spectrum of every speech frame was computed from its LPC parameters as described in Eq. (15).

3.2.2. Implementation of Source-Filter Model for Speech Production

The source-filter model for speech production shown in Figure 7 was implemented into the *LPC_filter.m* function (see code in Appendix D). The filter was estimated by the LPC coefficients computed from MFCCs according to Eq. (1); and the excitation signal was modeled for voiced and unvoiced sounds, as it was described in Section 2.1.1. Some considerations that were supposed to be taken are presented below.

The excitation signal was modeled for voiced and unvoiced sounds as a pulse train and random noise, respectively. They were filtered separately, using the Matlab function *filter.m*, to produce the synthesized speech from different models of sound.

The speech is synthesized back from the LPC parameters sequence computed from the MFCC vectors. In this way, the excitation signal must be considered in speech windows or segments. Every segment acts as the excitation signal for one frame of the speech signal such that, the coefficients of the filter are the LPC parameters derived for this frame. Therefore, the sequence of excitation signal segments is passed through the LPC filter, which varies frame to frame, to produce the sequence of synthesized speech frames. They are consecutively concatenated to make up the synthesized speech signal.

Since the synthesized speech is performed by synthesizing successive segments of excitation signal, the LPC filter must avoid the discontinuities between the consecutive segments. For solving this problem, it was necessary to consider the use of the initial conditions of the filter in the Matlab function *filter.m*. Hence, in the change of the filter from one frame to the next one, the final conditions of one filter keep as the initial conditions of the next filter.

3.2.3. Incorporation of Dynamic Features: Delta Coefficients

As said before, the speech synthesis is performed by a LPC filter whose coefficients change frame to frame; and the final conditions of one filter are the initial conditions of the next filter. This avoids discontinuities in the filtered of the successive segments of the excitation signal. However, there is not a gradual transition between the LPC parameters of consecutive frames.

The incorporation of temporal derivatives to the *static* feature vectors, in this case to the LPC parameters, makes possible a continuous transition between the LPC parameters of consecutive frames. As said in Section 2.2.5, dynamic features can measure the change in the *static* features.

Delta coefficients are first order time derivatives and they were the dynamic features added in the generative model. The algorithm to approximate the delta coefficients was implemented in a Matlab function called *deltacoef.m*, enclosed in Appendix D. The algorithm was based on a lineal interpolation between the LPC parameters (filter gain and filter coefficients) of consecutive frames.

Each frame was divided into four subframes, whose LPC parameters were the lineal interpolation between the LPC parameters of the current frame and the next one. Since it was impossible to make an interpolation in the last frame; it was decided to repeat the value of its LPC parameters for the four last subframes. With this linear interpolation the total number of frames was increased by a factor of four. Hence, the change of the filter frame to frame was performed through intermediates frames, whose LPC parameters were the interpolation between the original frames such that, the transition between them was smoothed.

Whereas for the filter gain interpolation it was only necessary a lineal interpolation within 4 points between the consecutives filter gains, for the interpolation of the filter coefficients, some aspects had to be considered.

The lineal interpolation of the filter coefficients was performed between the umpteenth coefficient of the frame *n* and the umpteenth coefficient of the next frame *n+1*. Furthermore, the interpolation was not directly applied to the filter coefficients $\{a_i\}$, but to other coefficients called *reflection coefficients*, $\{k_i\}$. These coefficients are as well calculated in the computing of the Levinson-Durbin recursion algorithm as intermediate variables in the calculation of LPC coefficients. The reflection coefficients

are bounded by the range in Eq. (24); and this is a necessary and sufficient condition for all poles of the LPC filter to be inside the unit circle, guaranteeing a stable filter (Huang et al., 2001).

$$-1 < k_i < 1 \tag{24}$$

The fact of interpolating the reflection coefficients instead of the LPC filter coefficients is that they guaranty the stability of the filter after the interpolation. When one implements a linear interpolation of the reflection coefficients, *if the coefficient of both frames are in the range in* Eq. (24), *the linearly interpolated reflection coefficients also have that property, and thus the filter is stable* (Huang et al., 2001). The LPC filter coefficients do not have this property.

Thus, the LPC filter coefficients were converted to their corresponding reflection coefficients. Then, a lineal interpolation within 4 points was applied to them. Later, the linearly interpolated reflection coefficients were converted back to LPC filter coefficients, guarantying that the filter was still stable within all the news frames. The conversion from the filter coefficients to reflection coefficients and vice versa was performed by using the Matlab functions *lpcar2rf.m and lpcrfr2ar.m*, respectively. They belong to the Voicebox Matlab toolbox.

As said above, the result of the lineal interpolation was an increase of the total number of frames in a factor of 4, because the lineal interpolation could be considered as a division of each frame into four subframes. That is why some processing parameters had to be modified in the function *LPC_filter.m* to be able to be used for delta features. In this case, the mode of the performance of the *LPC_filter.m* function must be set to 1 (see code of *LPC_filter.m* in Appendix D). Then, the frame shift is a ¼ of the original frame shift.

3.2.4. De-emphasize Processing

When the features extraction process was presented in Section 2.2, it was explained that the speech signal is passed through a pre-emphasis filter before the spectral analysis. That is why, finally, the synthesized speech must be passed through a de-emphasize filter. It is a low-pass filter, whose transfer function (Eq. (25)) is the inverse to Eq. (5) and corresponds to Figure 23 (see that it is the inverse transfer function comparing with Figure 13).

$$H_{deem} = \frac{1}{H_{preem}(z)} = \frac{1}{1 - a_{preem} z^{-1}} \tag{25}$$

The de-emphasize coefficient must have an equal value than the pre-emphasis coefficient, that was set to 0.97.

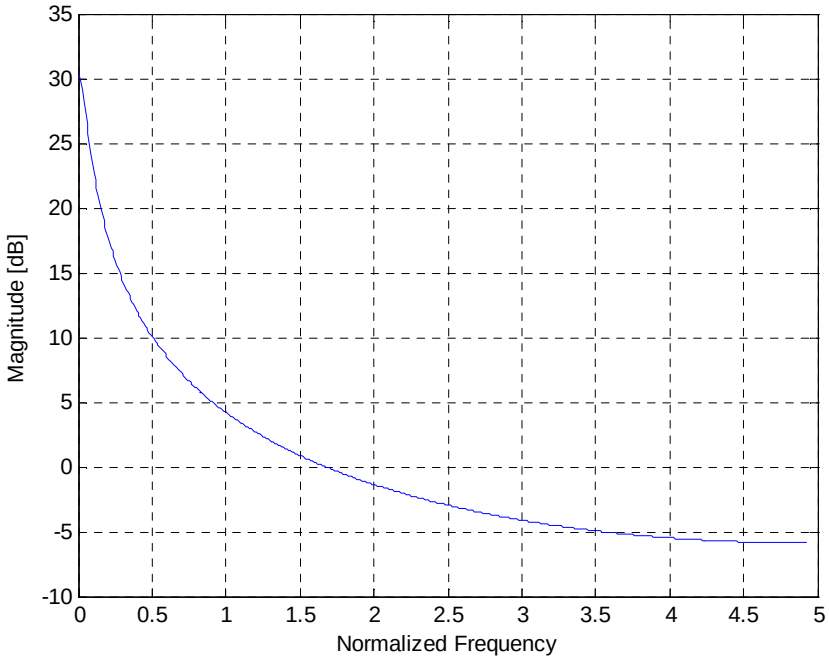

Figure 23: De-emphasized filter (a=0.97)

4. ANALYSIS OF RESULTS AND DISCUSSION

This section is an analysis of the results extracted in the implementation of the generative model described in the previous section. In short, the generative model returns the synthesized speech from its MFCC representation.

One way to make an objective evaluation of the generative model is to compute the spectral distance between the original signal and the one produced from the MFCC coefficients. That can be done by developing two spectral models based on the LPC coefficients from the original signal and the ones computed from the MFCCs. That is why; previously to the evaluation of the signal produced from the MFCC coefficients, a LPC analysis of the original waveform speech signal was developed.

In the other hand, for a subjective evaluation of the generative model, it is interesting to present an interpretation of the intelligibility of the synthesized speech versus the original signal.

This section is divided into five sections. Firstly, the LPC analysis of the waveform speech signal is presented. Secondly, a presentation of the MFCC files generated by HTK Software and an analysis of the MFCC vectors are done. In the third section, the two approaches for the generative model are compared and discussed. The fourth section is devoted to an evaluation of the parametric representation, based on the spectral distance measure. Finally, the intelligibility of the reconstructed speech by the generative model is commented and discussed.

4.1. LPC ANALYSIS OF THE WAVEFORM SPEECH SIGNAL

The algorithm for LPC analysis was implemented in a Matlab function called *waveform_analysis.m* and it is enclosed in Appendix D. This function follows the process shown in Figure 18, implementing the filtered of the speech signal through the pre-emphasis filter, the frame blocking and Hamming windowing and the LPC feature extraction.

The algorithm *waveform_analysis.m* was executed for a LPC analysis of 12^{th} order. The LPC parameters (filter gain, g; and LPC filter coefficients, $\{a_i\}$) were computed by using the Matlab function *proclpc.m*, which belongs to Matlab Auditory Toolbox.

In order to show the performance of the different steps involved in LPC extraction process, the following figures were executed for *sa1.wav* file. In Figure 24, the original speech waveform and how is affected after the pre-emphasis filter is illustrated. Figure 25 presents the effect of using a Hamming window, and Figure 26 shows the Linear Predictor spectrum of one frame as compared with its magnitude spectrum.

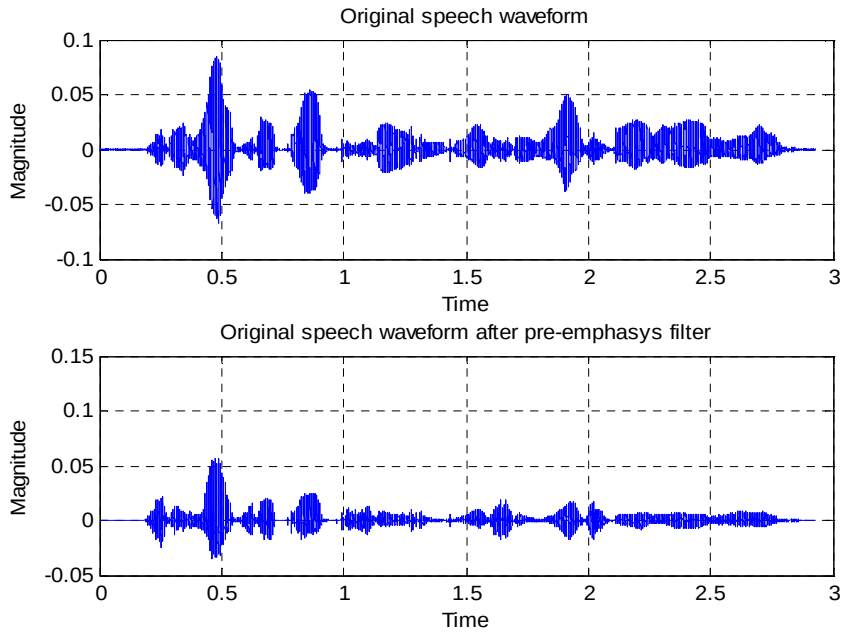

Figure 24: Original speech waveform and original speech waveform after the pre-emphasis filter with coefficient equal to 0.97 (*sa1.wav file)*

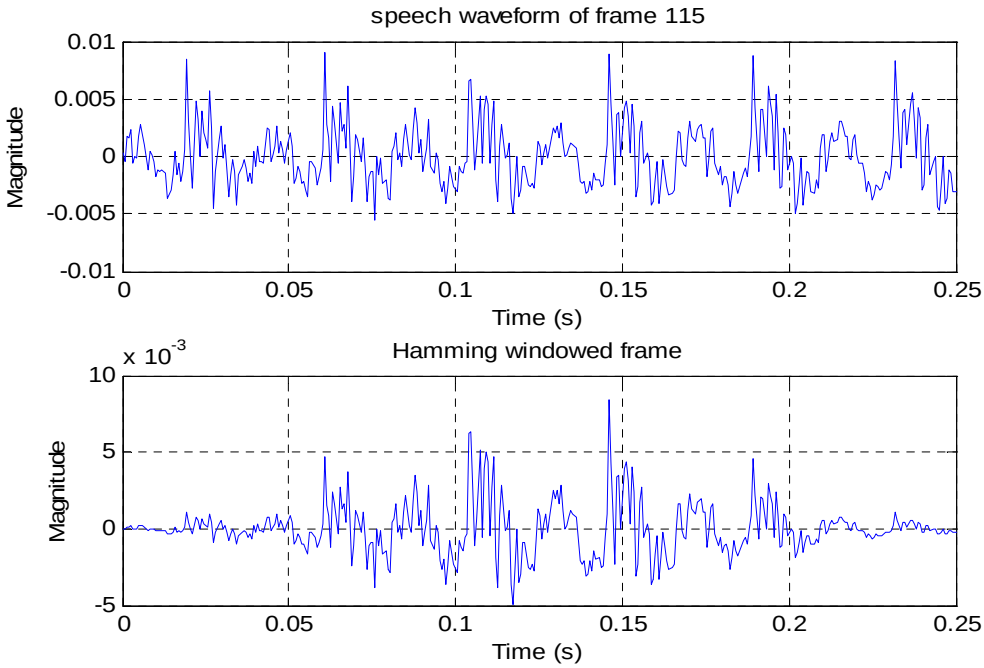

Figure 25: Effect of multiplying one speech frame by a Hamming window (frame 115 from *sa1.wav*)

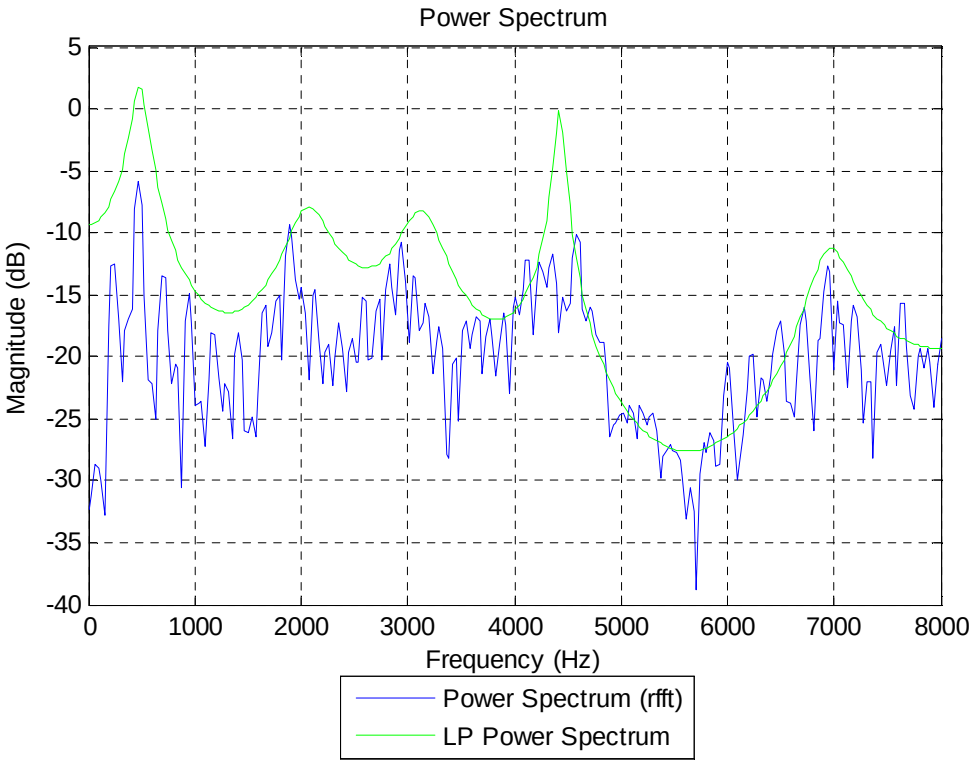

Figure 26: Comparison of the power spectrum computed from LPC coefficients with the original magnitude spectrum (frame 115 of *sa1.wav*)

As one can see in Figure 25, the use of a Hamming window makes that magnitude of the speech frame tapers from the centre of the window to the edges. This fact reduces the discontinuities of the signal at the edges of each frame.

Figure 26 shows the Linear Prediction (LP) power spectrum compared with the magnitude spectrum of a speech frame. One can see that the power spectrum computed from LPC coefficients is actually representing the *spectral envelope* of the magnitude spectrum of this frame. This spectral envelope marks the peaks of the formants of the speech frame.

More examples that illustrate this fact can be added. Figure 27 corresponds to the frames 84 and 176 of the same waveform file (*sa1.wav*).

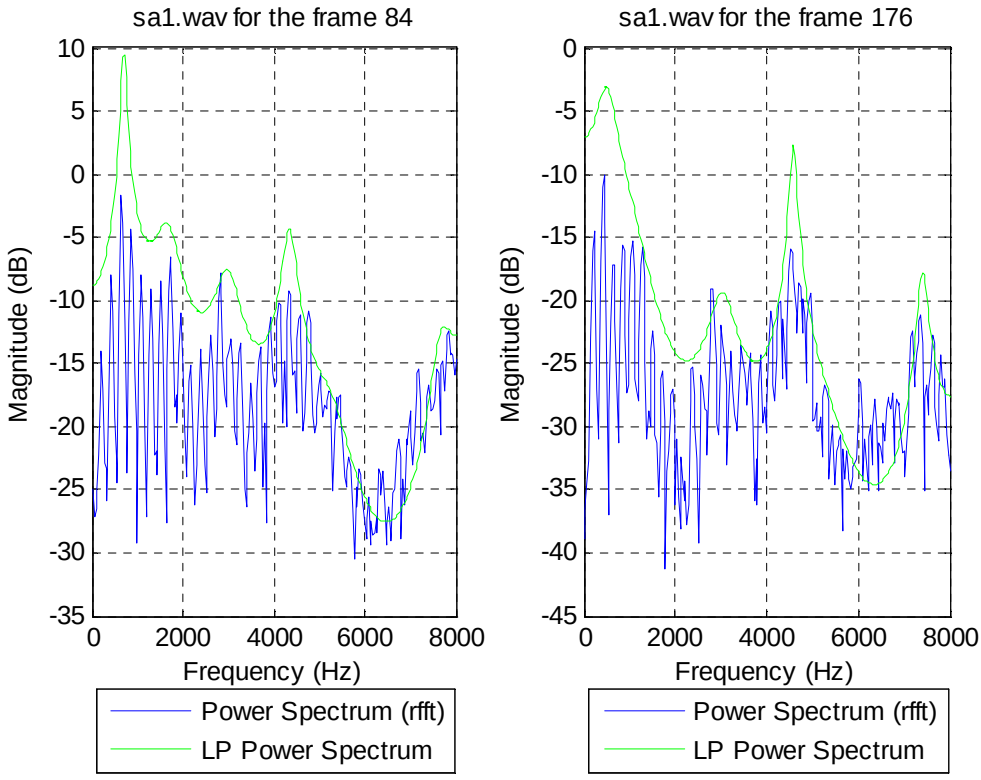

Figure 27: Comparison of the power spectrum computed from LPC coefficients with the original magnitude spectrum (frames 84 and 176 of *sa1.wav*)

4.2. MFCC VECTORS COMPUTED IN HTK

In the Section 3.1, the HCopy tool was presented as the tool to parameterize the speech signal into a sequence of feature vectors. The parameterization was defined by setting the configuration parameters of the configuration file, called *hcopy.conf* (enclosed in Appendix B).

In this section an analysis of the output files generated by HTK can been done in order to explore how HTK performs the output MFCC files. The set of source waveforms files given and the output MFCC files generated in HTK were listed in Figure 21. For example, the command line to create the corresponding output MFCC file of the source file *sa1.wav* was:

HCopy -C hcopy.conf sa1.wav mfcc1.htk

It is important, for the later processing of the MFCC vectors, to know how the structure of the output file executed by HTK is, and to check that the input conversions are being performed properly. For this task, there is a HTK tool which allows examining the contents of the speech data files, the HList tool (Young et al., 2006). The HList tool was executed to check the conversion performed in the previous *sa1.wav* file:

HList -C hcopy.conf -o -h -t -s 113 -e 115 sa1.wav

The options *–h* and *–t* are used to print the source header and the target header information. The options *–s* and *–e* indicate the range of sample vectors to display. The option *–o* is used to show the observation structure which identifies the role of each item in each sample vector (Young et al., 2006). The results are display below:

```
---------------------------------- Source: sa1.wav ----------------------------------
  Sample Bytes:  2        Sample Kind:   WAVEFORM
  Num Comps:     1        Sample Period: 62.5 us
  Num Samples:   46797    File Format:   WAV
------------------------------------- Target -------------------------------------
  Sample Bytes:  52       Sample Kind:   MFCC_0
  Num Comps:     13       Sample Period: 10000.0 us
  Num Samples:   290      File Format:   HTK
------------------------------- Observation Structure -------------------------------
x:      MFCC-1  MFCC-2  MFCC-3  MFCC-4  MFCC-5  MFCC-6  MFCC-7  MFCC-8  MFCC-9 MFCC-10
       MFCC-11 MFCC-12      C0
---------------------------------- Samples: 113->115 ----------------------------------
113:    -8.294  -4.822  -3.366 -15.631 -25.019 -17.790 -20.292  -0.808 -20.792  -4.385
       -15.564   4.213  56.708
114:    -7.577  -4.108   0.308 -13.606 -19.973 -15.594 -14.265   6.377 -16.892   2.171
       -10.880   7.017  57.463
115:    -7.040  -3.334   0.652 -14.712 -19.806 -14.623 -14.213   7.083 -16.690   4.210
       -10.035   5.303  56.754
---------------------------------------- END ----------------------------------------
```

The source header information confirms that the source file called *sa1.wav* contains *waveform* data with 2 byte samples and 46797 samples in total. The samples period is 62.5 μs which corresponds to a sample frequency of 16 kHz. With this data, one can know that the duration of the speech waveform file is the 2.923 seconds (46797samples/16kHz).

The target header information confirms that speech data have been parameterized to a sequence of 290 MFCC vectors, including, each one, the C_0 component as the energy component. Each MFCC vector contains 13 components and is 52 bytes in size. The frame period is 10 ms which corresponds to an output frame rate of 100 frames/second. Since the speech file is of 2.923 seconds, the number of frames and consequently the parameter vectors performed are 290 vectors.

The observation structure describes the structure of output data. One can see that the 13 components of the parameter vectors are grouped into 12 MFCC coefficients and the last component is the energy component, C_0. Finally, for this example, the values of the MFCC coefficients for three frames are displayed.

Some aspects have to be considered before processing MFCC vectors in Matlab. The output MFCC files place the C_0 component in the last position of the parameter vector. When the MFCC vectors are processed this component has to be changed to the first position into the parameter vector. In the other hand, it is worth highlighting that HTK considers the first sample vector with index 0 and Matlab does not have the index zero and starts in index 1.

Another example of the output MFCC files can be shown by using another source waveform file, *sx37.wav:*

HList -C hcopy.conf -o -h -t -s 40 -e 41 sx37.wav

```
---------------------------------- Source: sx37.wav ----------------------------------
  Sample Bytes:  2        Sample Kind:   WAVEFORM
  Num Comps:     1        Sample Period: 62.5 us
  Num Samples:   36250    File Format:   WAV
------------------------------------- Target -------------------------------------
  Sample Bytes:  52       Sample Kind:   MFCC_0
  Num Comps:     13       Sample Period: 10000.0 us
  Num Samples:   225      File Format:   HTK
------------------------------- Observation Structure -------------------------------
x:      MFCC-1  MFCC-2  MFCC-3  MFCC-4  MFCC-5  MFCC-6  MFCC-7  MFCC-8  MFCC-9 MFCC-10
       MFCC-11 MFCC-12      C0
---------------------------------- Samples: 40->41 -----------------------------------
40:    -12.614  -6.304 -19.824 -19.633   5.511  -4.476  -6.925  -1.930  -3.798  11.280
        -0.297  -4.124  47.195
41:     -7.308  -0.321 -11.791 -17.542   3.268 -13.308 -15.855  -9.029 -19.180  -5.943
        -8.695  -5.366  48.396
------------------------------------- END --------------------------------------
```

Energy Compaction within the MFCC Coefficients

As explained in Section 2.2.3, the MFCC coefficients are the DCT-II of the log-spectral-energies at the centre frequencies of the mel filterbank. The Fourier Transform of a speech frame is transformed to a mel-frequency scale by the filterbank analysis with M channels. The output of this process is the M log-spectral-energies at mel centre frequencies. The DCT-II allows an energy compaction in its lower coefficients. So, the use of the DCT-II makes that the M filterbank channels can be reduced to L (L<M) MFCC coefficients. This truncation into the cepstral components allows recovering a smoothed spectral representation in which phonetically irrelevant detail has been removed.

Despite of the MFCC computation was performed by using 24 filterbank channels (see configuration parameters in Appendix B); a mel power spectrum of a speech frame can be computed from its 13 MFCCs by using the inverse DCT-II (function *idct_htk.m* enclosed in Appendix D). This is illustrated in Figure 28.

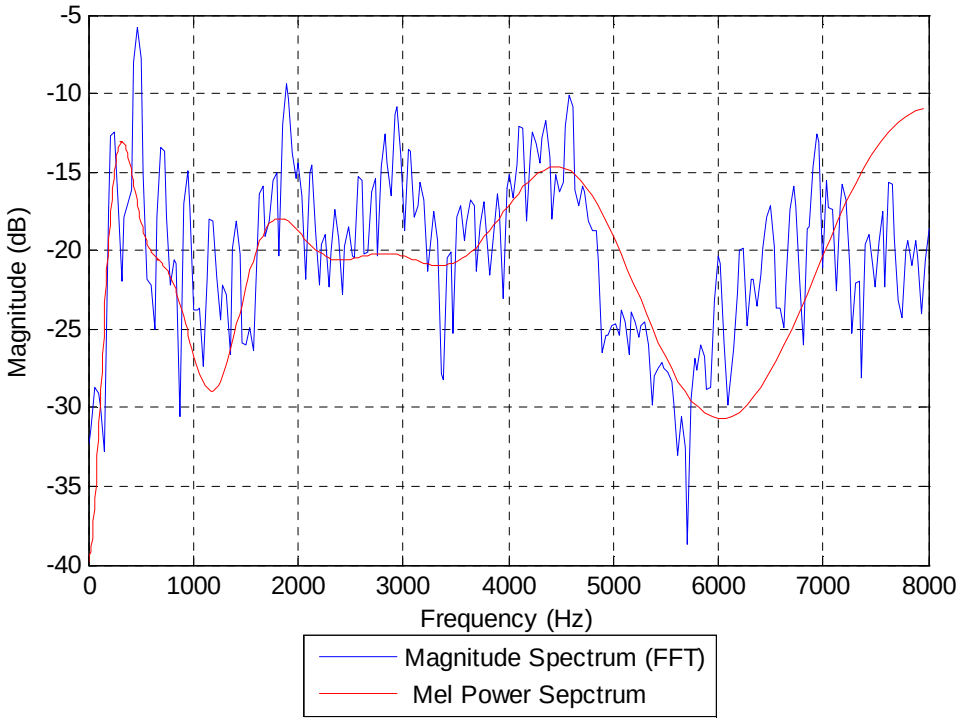

Figure 28: Mel power spectrum of one speech frame compared with its magnitude spectrum (frame 115 from *sa1.wav*)

Figure 28 demostrates that the mel power spectrum is the smoothed spectral envelope of the magnitude spectrum of the speech frame. In this case, the harmonics of the speech spectrum are flattened because of, the reduction of the frequency resolution performed wtihin mel-filterbank analysis and, the truncation of higher-order coeffcients in the DCT-II computation.

4.3. ANALYSIS OF TWO APPROACHES FOR THE GENERATIVE MODEL

Section 3.2.1 deals the computation of the LPC coefficients from the MFCC vectors. It was seen that the LPC coefficients come from the solution of the Yule Walker equations. They can be solved by the autocorrelation method, for which the autocorrelation coefficients must be calculated. In this point, two approaches were proposed to estimate the autocorrelation coefficients based on the IFT of the mel power

spectrum. These approaches were implemented in the algorithms *mfcc2spectrum.m* and *mfcc2spectrum2.m* (see code in Appendix D).

In this section, the results of both algorithms will be exposed and discussed. So, the power spectrum computed from the LPC parameters as compared with the mel power spectrum will be plotted by executing both algorithms.

Figure 29 is obtained by executing the *mfcc2spectrum.m* function. This algorithm makes a linear interpolation of the mel power spectrum to get samples uniformly spaced in a linear frequency scale in order to use the inverse Fourier Transform of Eq. (22).

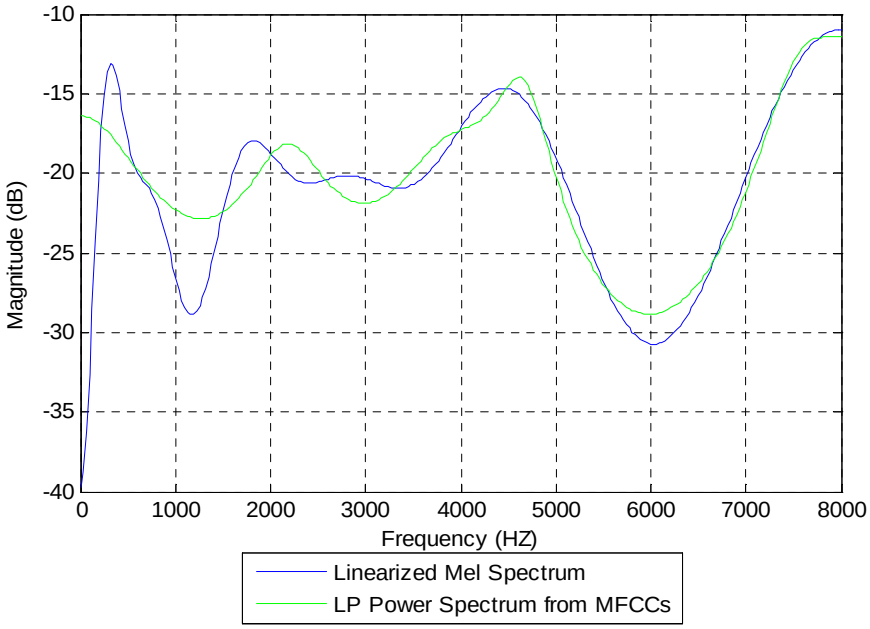

Figure 29: LP power spectrum computed from MFCCs by generative model 1: *mfcc2spectrum.m* (frame 115 from *sa1.wav*)

One can see that the LP power spectrum computed from MFCC coefficients is approximated to the mel power spectrum. Both of them represent the spectral envelope of the magnitude spectrum of the speech frame.

Following Figure 30 is obtained by executing the *mfcc2spectrum2.m* function. This algorithm applies the inverse Fourier Transform of Eq. (23) directly to the mel power spectrum at frequencies on a mel scale considering their bandwidth.

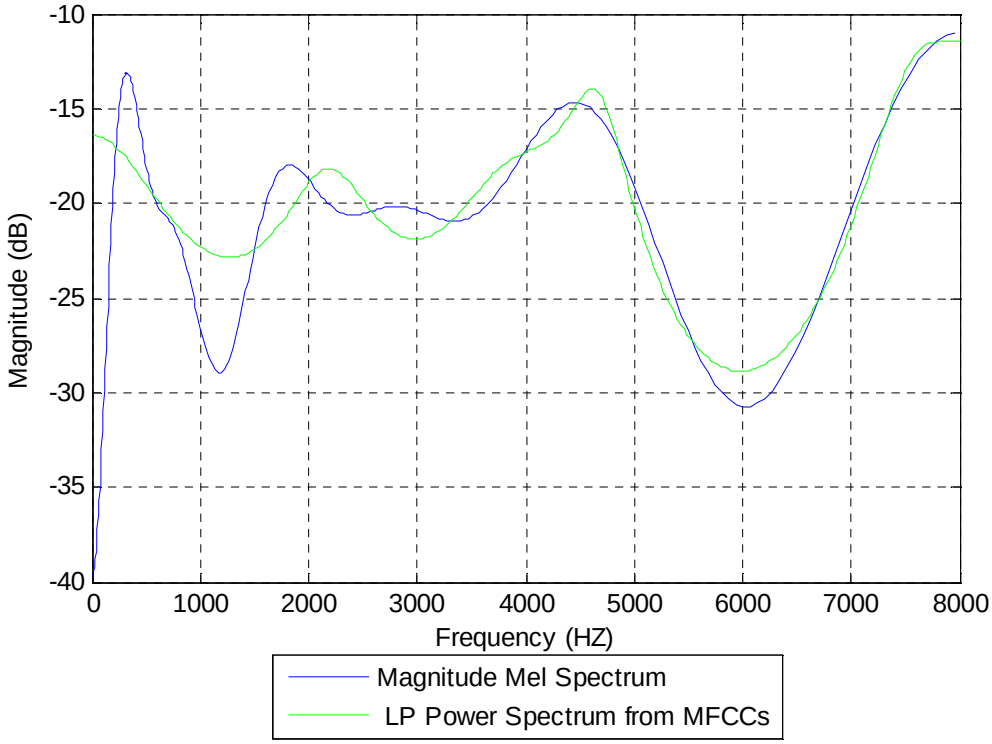

Figure 30: LP power spectrum computed from MFCCs by generative model 2: *mfcc2spectrum2.m* (frame 115 from *sa1.wav)*

The results are equal to the ones obtained with the first algorithm, since both of them can be considered as a linear interpolation in which, finally, the mel power spectrum samples have to correspond to a determined frequency separation.

The algorithm of *mfcc2spectrum2.m* is faster than the *mfcc2spectrum.m*. That is because of, the first one computes the autocorrelation coefficients of one frame in one matrix multiplication; whereas, the second one has to make one linear interpolation for each equally-spaced frequency sample. That is why; the results of the generative model will be performed by using the *mfcc2spectrum2.m* algorithm.

4.4. SPECTRAL DISTANCE MEASURE

As was introduced before, the goal of the generative model is to implement a system or method to be able to synthesize speech from its MFCC parametric representation. The goodness of the synthesized speech can be measured by computing the spectral distance between the original signal and the one produced from the MFCC coefficients. For that, the two spectral models used were the one obtained from the LPC coefficients computed from the original signal and the one obtained from the LPC coefficients computed from the MFCCs.

The spectral distance measure and its L_2 spectral norm (*rms log spectral distance*) were explained in Section 2.3. An algorithm to measure the *spectral distance* between two spectral models was implemented in a Matlab function called *spectral_distance.m*; and it is enclosed in Appendix E. The algorithm follows Eq. (16) and Eq. (17).

Several examples will be given to show a graphical comparison between the two spectral models.

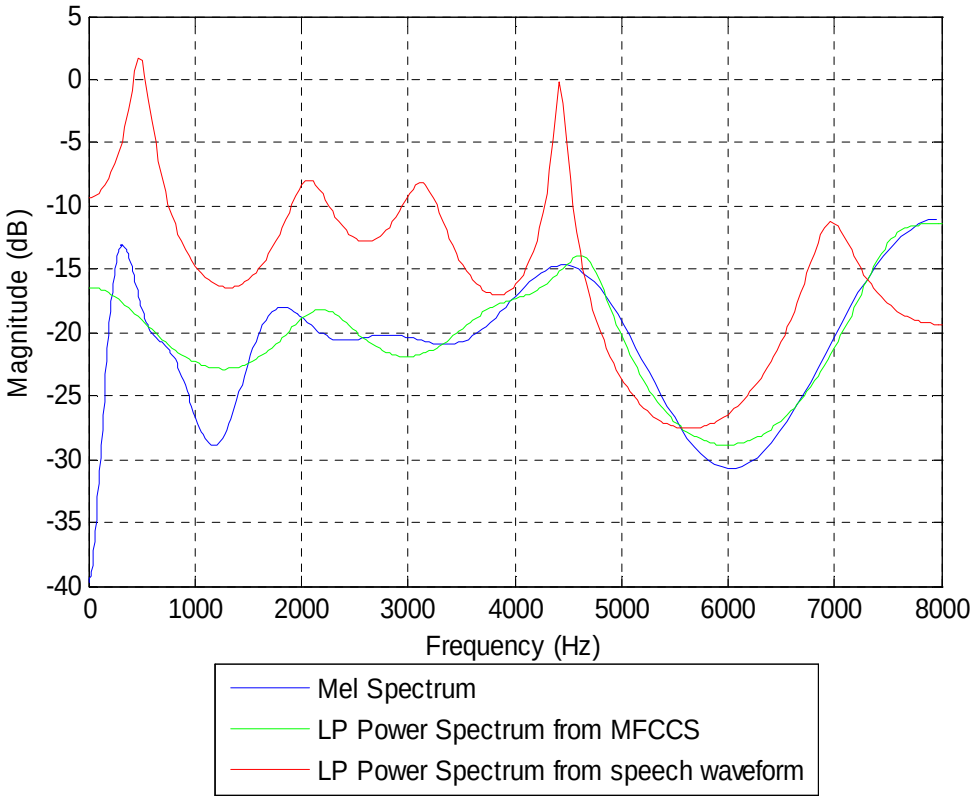

Figure 31: Comparison of spectral models from the original speech waveform and from the MFCC vectors (fame 115 from *sa1.wav*)

It was said before that the LP power spectrum computed from speech waveform as well as from MFCCs coefficients, represented the spectral envelope of the magnitude spectrum of the speech frame. However, in Figure 31, one can see that the harmonics or formants peaks are marked in the LP power spectrum from speech waveform whereas, they are more flattened when is computed from the MFCCs coefficients. This gives a spectral distortion between them of 0.87dB.

Another example can be shown by using the *si648.m* file. Figure 32 illustrates the comparison of the LP spectrums whose spectral distortion computed is of 0.35 dB.

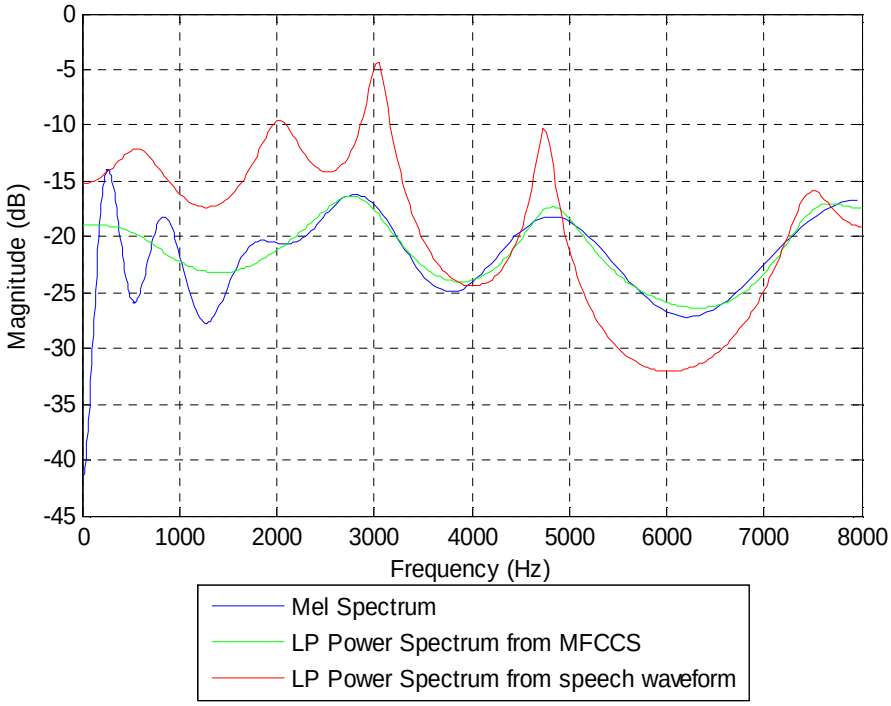

Figure 32: Comparison of spectral models from the original speech waveform and from the MFCC vectors (fame 133 from *si648.wav*)

The generative model can be evaluated more in detail by computing the spectral distance for every frames of each speech waveform file. Hence, it is possible to give an overview of the minimum and maximum spectral distances that were computed by the model. Also, the mean spectral distortion of every speech file is calculated. Table 3 shows the results of these measures.

Table 3: Study of spectral distortion computed between LP power spectrum from original waveform speech signal and the one computed from MFCCs

Source waveform files	Minimum spectral distortion (dB)	Maximum spectral distortion (dB)	Mean spectral distortion (dB)
sa1.wav	0.11	2.09	0.76
sa2.wav	0.14	2.11	0.67
si648.wav	0.09	1.67	0.57
si1027.wav	0.11	2.12	0.62
si1657.wav	0.09	1.71	0.67
sx37.wav	0.10	2.18	0.58
sx127.wav	0.14	2.56	0.68
sx217.wav	0.11	1.83	0.77
sx307.wav	0.16	2.01	0.61
sx397.wav	0.10	1.93	0.72

From the above table, one can extract that he minimum spectral distortion computed is 0.09 dB and the maximum is 2.56 dB. So, the results of the generative model depend of the utterances which have to be synthesized. If one computes the mean of the mean spectral distortion of every speech file can give a mean estimate of the generative model. Doing that, it is possible to say that the generative model has a spectral distortion mean of 0.66 dB. This mean depends strongly on the speech data that were used for the experimental results.

4.5. STUDY OF THE INTELLIGIBILITY OF THE RECONSTRUCTED SPEECH

This section is proposed to give an interpretation of the intelligibility of the reconstructed speech.

The speech synthesis is based on the implementation of a source-filter model for speech production (Figure 7). As explained in Section 3.2.2, for the generative model implementation, the filter was estimated by using the LPC coefficients computed from the MFCCs; and the excitation signal was modeled for voiced and unvoiced sounds.

After, adding to the generative model, two tests were proposed to synthesize speech from other two different excitation signals:

- Predictor error signal or residual signal of the LPC analysis of the waveform speech (*test1.m* in Appendix F).
- A mixed model for voiced and unvoiced sounds based on the pitch information of the original waveform speech (*test2.m* in Appendix F).

Thus, this section is divided into two. The first one presents an interpretation of the intelligibility of the reconstructed speech when unvoiced and voiced excitation signals are used; and in the second one, the results of the tests for different excitation signals are discussed.

4.5.1. Speech Synthesis from Voiced and Unvoiced Excitation Signals

The excitation signal models unvoiced sounds as a random noise and the voiced sounds as a pulse train repeating at a fixed constant pitch. The difference in the synthesized speech, when it is computed from the two different excitations, will be discussed in this section.

The speech synthesis was performed from its MFCC representation; in which much information is lost in the extraction process. So, as said in the previous sections, the recovered spectrum is smoothed since the harmonics and formants of the speech signal are flattened in that process. This fact made that resulting synthesized speech

sound as monotone voice. Moreover, the subjective quality of the speech was limited by annoying buzzes, thumbs and tonal noises.

In the experimental work, when the speech was synthesized from a white random noise (all unvoiced excitation), it sound as whispered voice. Thumps could be perceived due to the erroneous noise burst within voiced segments. Whereas, when the speech was synthesized from a pulse train (all voiced excitation), it seemed to be affected by a tonal noise, perceived as hums or droning sounds which degraded the subjective quality. In both cases, the resulting voice tended to be a monotone voice, as was concluded above. This perceived noise and distortion made that the resulting voice was not clearly understandable.

In order to compare the resulting synthesized speech, the LPC filter was also implemented from the LPC coefficients of the original speech waveform. In this case, the speech synthesized sound much better and understandable. The following Figures, 33 and 34, show the synthesized speech waveform by using the LPC filter implemented from the LPC coefficients computed from MFCC vectors, as compared to the one resulting by using the LPC filter implemented from the LPC coefficients from the original speech waveform. The original speech waveform signal is plotted in Figure 24.

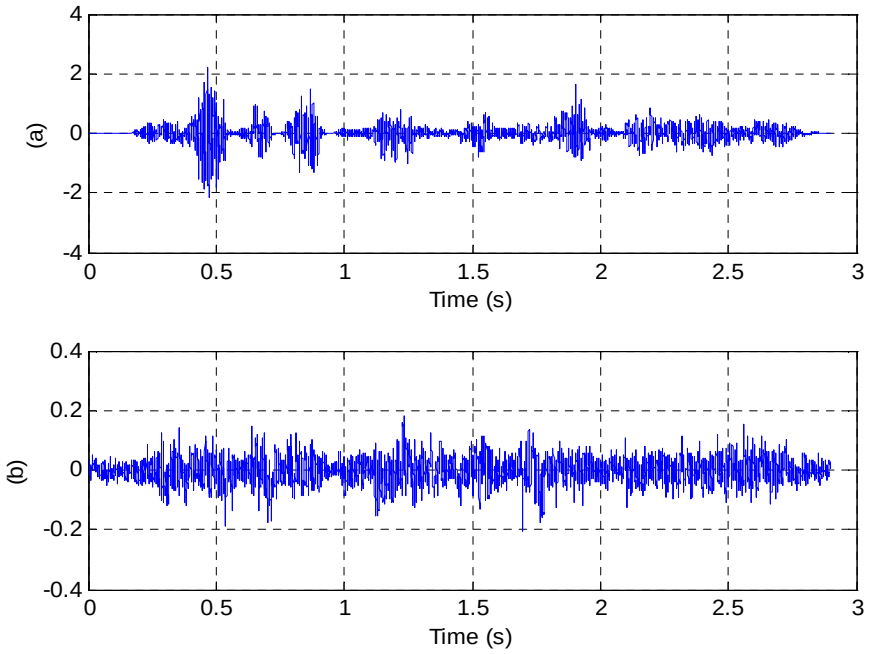

Figure 33: Synthesized speech from an unvoiced excitation signal when the filter is implemented by (a) the LPC coefficients computed from the original speech waveform and (b) the LPC coefficients computed from the MFCCs vectors (*sa1.wav file*)

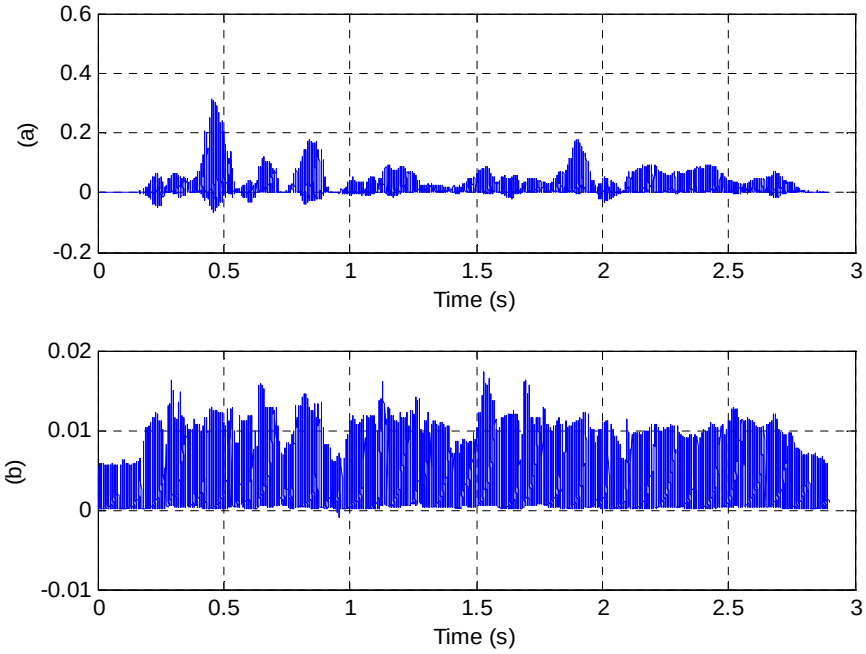

Figure 34: Synthesized speech from an voiced excitation signal when the filter is implemented by (a) the LPC coefficients computed from the original speech waveform and (b) the LPC coefficients computed from the MFCCs vectors (*sa1.wav file*)

One can see, from the above figures, a greater distortion is presented when the filter model is implemented from the MFCC representation. The synthesized speech is clearer when is performed from the LPC coefficients than when is performed by transforming the MFCCs into LPC coefficients that carry out more approximations. However, it has to be emphasized that the MFCC representation contains more perceptually information than the LPC coefficients.

Further on, Figures 35 and 36 show the spectrograms of the synthesized speech waveforms as compared with the spectrogram of the original speech waveform.

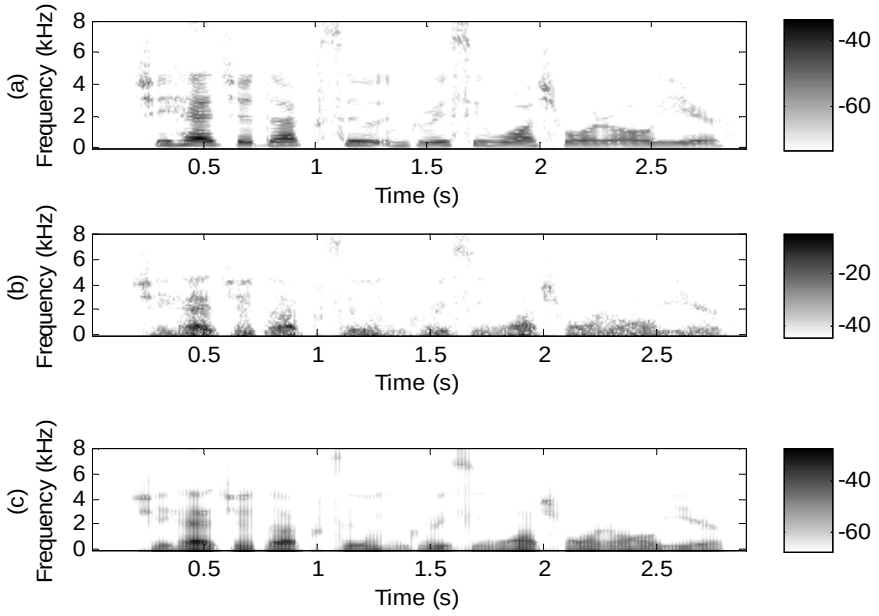

Figure 35: Spectrogram of (a) original speech waveform; and synthesized speech from (b) unvoiced excitation signal and (c) voiced excitation signal. The filter is implemented with LPC parameters computed form original speech waveform (*sa1.wav*)

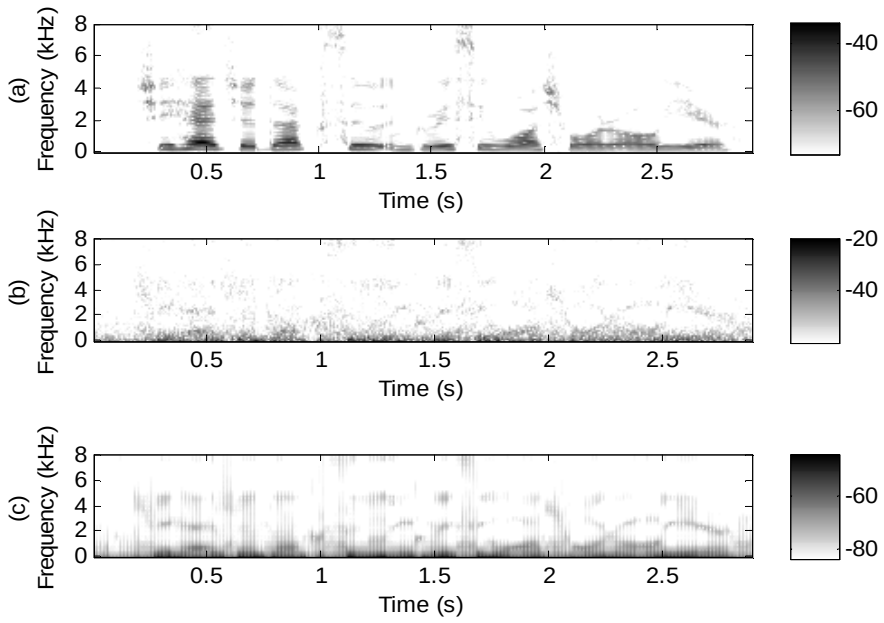

Figure 36: Spectrogram of (a) original speech waveform; and synthesized speech from (b) unvoiced excitation signal and (c) voiced excitation signal. The filter is implemented with LPC parameters computed from MFCC vectors (*sa1.wav*)

As was said in Section 3.2, the delta coefficients for the LPC coefficients were added in order to achieve a better recognition performance. These coefficients were approximated by a linear interpolation as described in Section 3.2.3. It was expected that the incorporation of these features caused better subjective quality or intelligibility of the synthesized speech. However, the improvement could not be noted. The synthesized speech sound like the one computed with only LPC coefficients or even worse in some cases. This can be due to both non efficient implementation of the LPC filter for delta coefficients or a poor development in the approximation method of the delta coefficients.

4.5.2. Speech Synthesis from Different Excitation Signals

As presented above, the two excitation signals proposed were:

- Predictor error signal or residual signal of the LPC analysis of the waveform speech (*test1.m* in Appendix F). The residual signal is obtained when the original speech signal is filtered through the LPC filter, whose coefficients are those that minimize the prediction error (see Figure 19)**.**
- A mixed model for voiced and unvoiced sounds based on the pitch information of the original waveform speech (*test2.m* in Appendix F).

For the first test, the predictor error or residual signal was calculated by using the Matlab function called *proclpc.m* from Auditory Toolbox. It returns the predictor error for every frame of the speech signal. The synthesized speech from this excitation signal could be heard more clear although not as a natural voice. This is because of the speech is synthesized from a *real* excitation signal.

For the second test, the estimate of the pitch of the signal was also calculated by the *proclpc.m* function; whose algorithm for that is based on finding the peak in the residual's autocorrelation for each frame.

The frames of the speech signal were classified into voiced frames or unvoiced frames considering the value of the pitch estimate computed. Frames with a pitch value of zero were considered as unvoiced frames; in the other hand, frames with different values of zero were labeled to voiced frames.

Thus, knowing the pitch of the signal on each frame, they could be classified as voiced or unvoiced frames and then the appropriate voiced or unvoiced excitation signal must be used for each frame. That reduces the tonal noise or thumps that appear when unvoiced frames are synthesized from voiced segments and vice versa. Anyway, this synthesized speech sound more closely to the one obtained from all unvoiced excitation signal.

5. CONCLUSION

The work developed in this Master Thesis consisted of the implementation of a speech generative model; whereby the speech is synthesized and recovered from its MFCC representation. Synthesizing speech from parametric representations allows performing an investigation on the intelligibility of the synthesized speech as compared to natural speech.

The first part of the implementation work consisted of extracting the MFCCs feature vectors from a set of speech waveform files. In the HTK Software, the feature parameterization of speech was performed according to the parameter settings in the configuration file. After, the generative model implemented the conversion chain from HTK-generated MFCC vectors to speech reconstruction.

During the MFCC extraction process, much relevant information was lost due to reduction of the spectral resolution in the filterbank analysis and the next truncation into the MFCC components. However, that allowed recovering a smoothed spectral representation in which phonetically irrelevant detail had been removed. For that, the log mel power spectrum could be computed from its MFCCs by an inverse DCT. This mel power spectrum actually represented the envelope of the magnitude spectrum, where the harmonics appeared flattened.

In the generative model implementation was necessary to derive LPC coefficients from MFCC vectors. In one hand to implement the source-filter model for speech production; and in the other hand, to compute a spectral model that could be compared with the one derived directly from the original speech waveform.

Previously to the subjective evaluation of the generative model, the goodness of the synthesized speech was measured by computing the spectral distance between the original signal and the one produced from the MFCC coefficients. The two spectral models used were the one obtained from the LPC coefficients computed from the original signal, and the one obtained from the LPC coefficients computed from the MFCC coefficients. In this evaluation was extracted that the minimum spectral distortion computed was of 0.09 dB and the maximum one was of 2.5 dB. A spectral distortion mean of the generative model was calculated with a result of 0.66 dB. Although it seems a good result, even regarded as transparent quality, the final results obtained within speech synthesis indicated a strong distortion which avoided the entire intelligibility of reconstructed speech.

Both spectral models were also compared graphically. As the mel power spectrum, both LP spectral models represented the envelope of the magnitude spectrum. Whereas, the LP power spectrum computed from MFCC coefficients was really approximate to the smoothed mel power spectrum; the LP power spectrum computed from the LPC coefficients of the original signal allowed the representation of the formants peaks.

The source-filter model for speech production was implemented; where the filter was estimated by the LPC coefficients computed from the MFCCs vectors, and the excitation signal was modeled for voiced and unvoiced sounds.

In the experimental work, when the speech was synthesized from a white random noise (all unvoiced excitation), it sound as whispered voice. Thumps could be perceived due to the erroneous noise burst within voiced segments. Whereas, when the speech was synthesized from a pulse train (all voiced excitation), it seemed to be affected by a tonal noise, perceived as hums or droning sounds that degraded the subjective quality. In both cases, the resulting voice tended to be a monotone voice because of the smoothing of the harmonics and formants in the LP power spectrum computed from the MFCC coefficients. This perceived noise and distortion made that the resulting voice was not clearly understandable.

In order to compare the resulting synthesized speech, the LPC filter was also implemented by the LPC coefficients of the original speech waveform. In this case, the speech synthesized sound much better and understandable. However, it has to be emphasized that the MFCC representation contains more perceptually information than the LPC coefficients.

Delta coefficients for the LPC coefficients were added in order to achieve a better recognition performance. It was expected that the incorporation of these features caused better subjective quality of the synthesized speech. However, the synthesized speech sound like one computed with only LPC coefficients, even worse in some cases. This could be due to both non efficient implementation of the LPC filter for delta coefficients or a poor development in the approximation method of the delta coefficients.

Finally, two tests were proposed to study the synthesized speech from other excitation signals. In the first one, the excitation signal used was the predictor error or residual signal obtained within the LPC analysis of the waveform speech. In the second one, the excitation signal used consisted of a mixed model for voiced and unvoiced sounds based on the pitch information of the original waveform speech

For the first test, the synthesized speech could be heard more clear although not as a natural voice. This was because of the speech was synthesized from a *real* excitation signal more closely for achieving the original speech waveform.

For the second test, the frames were classified as voiced or unvoiced frames and then, synthesized from the appropriate voiced or unvoiced excitation signal. That reduced the tonal noise or thumps that appeared when unvoiced frames were synthesized from voiced segments and vice versa. Anyway, this synthesized speech sound more closely to the one obtained from all unvoiced excitation signal.

REFERENCES

Bogert, B., Healy, M. & Tukey, J. (1963), *The Quefrequency Analysis of Time Series for Echoes*, Proc. Symp. on Time Series Analysis, New York, J. Wiley: 209-243.

Brookes, M., *Voicebox: Speech Processing Toolbox for Matlab* [on line], Imperial College, London, Available on the World Wide Web: http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html.

Darch, J., Milner, B. & Shao, X., *Formant Prediction from MFCC Vectors*, University of East Anglia, Norwich.

Davis, S. & Mermelstein, P. (1980), *Comparison of Parametric Representation for Monosyllable Word Recognition in Continuously Spoken Sentences*, IEEE Transactions on Acoustic, Speech and Signal Processing, 28(9): 357-366.

Fant, G. (1996), *Acoustic Theory of Speech Production*, Mounton & Co., Gravenhage, Netherland.

Fletcher, H. (1940), *Auditory Patterns*, Reviews of Modern Physics, January 1940(12): 47-65.

Gray Jr., A.H. & Markel, J. D. (1976), *Distance Measures for Speech Processing*, IEEE Transactions on Acoustics, Speech and Signal Processing, October 1976(5): 380-391.

Hermansky, H. (1990), *Perceptual Linear Predictive (PLP) Analysis of Speech*, Journal of the Acoustic Society of America, 1990(87): 1738-1752.

Holmes, J. & Holmes, W. (2001), *Speech Synthesis and Recognition*, 2th ed., Tailor & Francis, London.

Huang, X., Acero, A. & Hon, H. (2001), *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall PTR, New Jersey.

Lippmann, R. (1997), *Speech Recognition by Machines and Humans*, Speech Comunication, Elsevier, April 1997(22): 1-15.

Mammone, R. J., Zhang, X. & Ramachandran, R. P. (1996), *Robust Speaker Recognition - A Feature-based Approach*, IEEE Signal Processing Magazine, September 1996: 58-71.

McCree, A. V. & Barnwell III, T. P. (1991), *A New Mixed Excitation LPC Vocoder*, Georgia Institute of Technology, Atlanta, 1991: 593-596.

Molau, S., Pitz, M., Schlüter, R. & Ney, H. (2001), *Computing Mel-Frequency Cepstral Coefficients on the Power Spectrum*, IEEE International Conference on Acoustics, Speech and Signal Processing, Germany, 2001: 73-76.

Nilsson, M. & Ejnarsson, M. (2002), *Speech Recognition Using Hidden Markov Model – Performance Evaluation in Noisy Environment*, Blekinge Institute of Technology Sweden.

Picone, J. W. (1993), *Signal Modeling Techniques in Speech Recognition*, Proc. IEEE, Japan, 81(9): 1215-1247.

Slaney, M. (1998), *Auditory Toolbox* [on line], Interval Research Corporation, California, Available on the World Wide Web: http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/.

Stevens, S. S. & Volkman, J. (1940), *The Relation of the Pitch to Frequency* , Journal of Psychology, 1940(53): 329.

Vergin, R. (1998), *An Algorithm for Robust Signal Modelling in Speech Recognition*, IEEE Transactions on Speech and Audio Processing: 969-972.

Vergin, R., O'Shaughnessy, D. & Farhat, A. (1999), *Generalized Mel Frequency Cepstral Coefficients for Large-Vocabulary Speaker-Independent Continuous-Speech Recognition*, IEEE Transactions on Speech and Audio Processing, 7(5): 525-532.

Young, S. (2008), *HMMs and Related Speech Recognition Technologies*, Springer Handbook of Speech Processing, 2008: 539-557.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V. & Woodland, P. (2006), *The HTK Book Version 3.4*, Cambridge University, Cambridge 2006, Available on the World Wide Web: http://htk.eng.cam.ac.uk.

APPENDIX

APPENDIX A: TEXTS OF UTTERANCES OF SPEECH DATA

1) sa1.wav: *She had your dark suit in greasy wash water all year.*
2) sa2.wav: *Don't ask me to carry an oily rag like that.*
3) si648.wav: *A sailboat may have a bone in her teeth one minute and lie becalmed the next.*
4) si1027.wav: *Even then, if she took one step forward he could catch her.*
5) si1657.wav: *Or borrow some money from someone and go home by bus?*
6) sx37.wav: *Critical equipment needs proper maintenance.*
7) sx127.wav: *The emperor had a mean temper.*
8) sx217.wav: *How permanent are their records?*
9) sx307.wav: *The meeting is now adjourned.*
10) sx397.wav: *Tim takes Sheila to see movies twice a week.*

APPENDIX B: CONFIGURATION FILE hcopy.conf

```
SOURCEKIND = WAVEFORM
SOURCEFORMAT = WAVE
SOURCERATE = 625

ZMEANSOURCE = FALSE
TARGETKIND = MFCC_0
#TARGETFORMAT = HTK
 TARGETRATE = 100000

#SAVECOMPRESSED = TRUE
#SAVEWITHCRC = TRUE

 WINDOWSIZE = 250000.0
 USEHAMMING = TRUE
 PREEMCOEF = 0.97

#USEPOWER = FALSE
 NUMCHANS = 24
#LOFREQ = -1.0
#HIFREQ = -1.0

#LPCORDER = 12
#CEPLIFTER = 22
#NUMCEPS = 12

#RAWENERGY = TRUE
#ENORMALISE = TRUE
#ESCALE = 1.0
#SILFLOOR = 50.0

#DELTAWINDOW = 2
#ACCWINDOW = 2
#SIMPLEDIFFS = FALSE

#USESILDET = TRUE
#SPEECHTHRESH = 0.0
#SILTHRESH = 0.0
#MEASURESIL = TRUE

#OUTSILWARN = TRUE
#SILMEAN = 0.0
#SILSTD = 0.0
#AUDIOSIG = 0
#V1COMPAT = FALSE
#VQTABLE = ""
```

APPENDIX C: GENERATIVE MODEL

```
function generative_model(filewav,filehtk,nf,model)

%--------------------------------------------------------------------
%   Name:   generative_model.m
%   Author: Noelia Alcaraz
%
%   Description:    Implement the conversion chain from HTK-generated
%                   MFCC representations to generative model.
%--------------------------------------------------------------------

% Set processing parameters
fs=16000;   %Frequency samples
Nshift=160; %Frame period of 10ms   --> 10ms*16KHz=160 samples
Nfrm=400;   %Frame length of 25 ms  --> 25,s=16jHz=400 samples
preem=0.97; %Pre-emphasis coefficient
p=12;       %LPC Filter order.
M=24;       %Number of filterbank channels
Nfreq=256;  %Number of frequency points for spectral representation

f=fs*(0:Nfreq-1)/(2*(Nfreq-1));

%LPC Analysis and representation from the original speech waveform.
 [x,fs,X,aCoef,G,LPspectrum]=waveform_analysis(filewav,nf);

%Generative Model:
%Read the MFCCs HTK file (mfcc vectors).
%Convert the MFCCs to Power Spectrum.

%Generative Model 1:
if(model==1)
    %Find samples of Magnitude Mel Spectrum at uniformly spaced linear
    %frequencies by linear interpolation method: (Syy)
    %Produce the LPC parameters (g,aa) and the Power Spectrum (Sxx).
    [mfcc,Syy,Sxx,aa,g]=mfcc2spectrum(filehtk,M,p,nf);
    figure
    plot(f,10*log10(abs(Syy(:,nf))))

%Generative Model 2:
else
    %Obtain the LPC parameters (g,aa) from MFCCs by applying the IFFT
    %to the Mel Spectrum on mel scale
    %( non-equally spaced linear frequencies)
    %considering the bandwidth at each mel frequency.
    %Obtain the Power Spectrum (Sxx) from these LPC coefficients.
    [mfcc,ymel,fsamp,Sxx,aa,g]=mfcc2spectrum2(filehtk,M,p,nf);
    figure
    plot(fsamp,10*log10(abs(ymel(:,nf))))

end

%Plot Power Spectrum
hold on
plot(f,Sxx(:,nf),'g')
plot(f,LPspectrum(:,nf),'r')
grid
xlabel('Frequency (Hz)')
ylabel('Magnitude (dB)')
```

```
legend('Mel Spectrum','LP Power Spectrum from MFCCS','LP Power
        Spectrum from speech waveform','Location','SouthOutside')


%Implementation the LPC Filter.
%The excitation signal is filtered to obtain the speech signal.

%1. Speech signal from LPC analysis of original waveform.
[sw,sv]=LPC_filter(G,aCoef',x,Nshift,Nfrm,p,2);

%2. Speech signal from LPC from the generative model 1 or 2
[sw2,sv2]=LPC_filter(g,aa,x,Nshift,Nfrm,p,2);

%3. First Time derivatives of LP parameters of generative model 1 or 2
[gd,ad]=deltacoef(g,aa);
[sw2_d,sv2_d]=LPC_filter(gd,ad,x,Nshift,Nfrm,p,1);


% de-emphasize
sw=filter(1,[1 -preem],sw);
sv=filter(1,[1 -preem],sv);

sw2=filter(1,[1 -preem],sw2);
sv2=filter(1,[1 -preem],sv2);

sw2_d=filter(1,[1 -preem],sw2_d);
sv2_d=filter(1,[1 -preem],sv2_d);

%Pay
soundsc(sw,fs);
soundsc(sv,fs);

soundsc(sw2,fs);
soundsc(sv2,fs);

soundsc(sw2_d,fs);
soundsc(sv2_d,fs);
```

APPENDIX D: FUNCTIONS USED IN THE GENERATIVE MODEL

Appendix D.1: *waveform_analysis.m*

```
function [x,fs,X,aa,G,Sxx]=waveform_analysis(filesignal,nf)

%-------------------------------------------------------------------------
%   Name:   waveform_analysis.m
%   Author: Noelia Alcaraz
%
%   Description:    Represent the Frequency Response (X) of the frame
%                   number nf and is compare with the Power Spectrum
%                   (Sxx) of the such frame obtained from LPC analysis
%                   of the original speech waveform.
%-------------------------------------------------------------------------

%Data from the configuration file of HTK to generate the MFCC_C0
Nfreq=256;  %number of frequency points in spectral representation
p=12;       %LPC analysis order
Tfrm=25;    %frame size for analysis (ms)
Tshft=10;   %frame shift for analysis (ms)
preem=0.97; %pre-emphasis coefficient

%Read the speech signal
[x,fs,wmode,fidx]=readwav(filesignal);
Nfrm=Tfrm*fs/1000;      %number of samples of frame size (400)
Nshft=Tshft*fs/1000;    %number of samples of frame shift (160).

% Pre-emphasis filter.
preem=0.97;
xpre=filter([1, -preem], 1, x);

%Frame blocking.
%There is an overlape of 15ms (400-160=240 samples).
frames=enframe(xpre,Nfrm,Nshft);
frames=frames'; %one frame per column
[lframe,numframes]=size(frames);

%Entire process is applied over each frame.

%Hamming Windowing
xw=[];
w=hamming(lframe);
for i=1:numframes
    xw(:,i)=frames(:,i).*w;
end

%fft of each frame
X=10*log10(abs(rfft(xw,510,1)));

%LPC analysis of original speech waveform.
Sxx=[];
[aa,e,P,G]=proclpc(x,fs,p,Tshft,Tfrm,preem);
for i=1:numframes
dbspec=lpcar2db(aa(:,i),Nfreq-2);
dbspec=dbspec+10*log10(G(i));
Sxx=[Sxx dbspec];
end
```

```
t=0:1/fs:(length(x)-1)/fs;
tframe=linspace(0,0.25,lframe);
f=fs*(0:Nfreq-1)/(2*(Nfreq-1));

%1. Plot Speech Waveform Signal
figure
subplot(2,1,1)
plot(t,x)
title('Original speech waveform')
xlabel('Time (s)')
ylabel('Magnitude')
grid
subplot(2,1,2)
plot(t,xpre)
title('Original speech waveform after pre-emphasys filter')
xlabel('Time (s)')
ylabel('Magnitude')
grid

%2. Plot the pre-processing steps of the selected frame nf
figure
subplot(2,1,1)
plot(tframe, frames(:,nf))
title(sprintf('speech waveform of frame %d',nf));
xlabel('Time (s)')
ylabel('Magnitude')
grid
subplot(2,1,2);
plot(tframe,xw(:,nf))
title('Hamming windowed frame');
xlabel('Time (s)')
ylabel('Magnitude')
grid

%3. Plot the Power Spectrum of the selected frame nf
figure
plot(f,X(:,nf));
hold on
plot(f,Sxx(:,nf),'g')
grid
title('Power Spectrum')
xlabel('Frequency (Hz)')
ylabel('Magnitude (dB)')
legend('Power Spectrum (rfft)','LP Power Spectrum', 'Location',
        'SouthOutside')
```

Appendix D.2: *mfcc2spectrum.m*

```
function [mfcc,Syy,Sxx,aa,g]=mfcc2spectrum(filehtk,M,p,nf)

%--------------------------------------------------------------------
%   Name:   mfcc2spectrum.m
%   Author: Noelia Alcaraz
%
%   Description:    Calculate the Power Spectrum from MFCCs vectors
%   Input parameters:
%       M -->   Number of filterbank channels
%       p -->   Order of LPC analysis
%--------------------------------------------------------------------
```

```
%Initial values
fs=16000;   %sample frequency
Nfreq=256;  %number of frequency points in spectral representation
K=256;      %number of mel-frequency points in spectral reconstruction

%Obtaining the Mel-Frequency Cepstral Coefficients from HTK program.
[d,fp,dt,tc,t]=readhtk(filehtk);
mfcc=d';
C0s=mfcc(end,:);
mfcc=[C0s; mfcc(1:end-1,:)];
[ncof,nframes]=size(mfcc);

Syy=zeros(Nfreq,nframes);

%Convert MFCCs to Log Magnitude Mel Spectrum
const=log(32768)*sqrt(M/K);
coeffadj=0.5*ncof/M;
ylogmel=coeffadj*idct_htk(mfcc,K)-const;

%Calculate Mel Power Spectrum
ymel=exp(2*ylogmel);

%K equally-spaced samples on mel scale converted to linear frequency
scale
fcnt=mel2frq(frq2mel(fs/2)*((1:K)-0.5)/K);

%Nfreq equially-spaced points on linear frequency scale.
flinear=linspace(0,fs/2,Nfreq);

%Find the value of Mel Power Spectrum of frame j for samples
%equally-spaced on linear frequency scale.
    for i=1:Nfreq
        dif=fcnt-flinear(i);
        [value,pos]=min(abs(dif));
         value=dif(pos);

        if(flinear(i)<fcnt(1))
            ylinear(i)=ymel(1,j);
        elseif (flinear(i)>fcnt(end))
            ylinear(i)=ymel(end,j);
        elseif (value==0)   %fcnt(pos)=flinear(i)
            ylinear(i)=ymel(pos,j);
        elseif (value<0)    %flinearl(i) --> [fmel(pos),fmel(pos+1)]
            ylinear(i)=interp1([fcnt(pos),fcnt(pos+1)],
                       [ymel(pos,j),ymel(pos+1,j)],flinear(i));
        else                %flinearl(i) --> [fmel(pos-1),fmel(pos)]
            ylinear(i)=interp1([fcnt(pos-1),fcnt(pos)],
                       [ymel(pos-1,j),ymel(pos,j)],flinear(i));
        end
    end
    Syy(:,j)=ylinear;
  end

%The Autocorrelation Coefficients are obtained applying the IDFT over
%Mel Power Spectrum on linear frequency scale.
r=irfft(Syy);
[aa,g]=levinson(r,p); %aa(nframes,p+1)
```

```
%Calculate Power Spectrum
Sxx=zeros(Nfreq,nframes);
for i=1:nframes
ff=rfft(aa(i,:).',2*(Nfreq-1)).';
Sxx(:,i)=-10*log10((1/g(i))*real(ff.*conj(ff)));
end

%Plot Spectra for the selected frame nf
%1.Plot Linearized Mel Spectrum
figure
plot(flinear,10*log10(abs(Syy(:,nf))))
ylabel('Magnitude (dB)')
xlabel('Frequency (HZ)')
hold on
grid
%2.Plot Spectrum produced by transforming MFCCs to LP coefficients
plot(flinear,Sxx(:,nf),'g')
legend('Linearized Mel Spectrum','LP Power Spectrum from MFCCs',
       'Location', 'SouthOutside')
```

Appendix D.3: *mfcc2spectrum2.m*

```
function [mfcc,ymel,fsamp,Sxx,aa,g]=mfcc2spectrum2(filehtk,M,p,nf)

%--------------------------------------------------------------------
%   Name:   mfcc2spectrum2.m
%   Author: Noelia Alcaraz
%
%   Description:    Calculate the Power Spectrum from MFCC vectors
%--------------------------------------------------------------------

%Initial values
fs=16000;   %sample frequency.
Nfreq=256;  %number of frequency points in spectral representation.
K=256;      %number of mel-frequency points in spectral reconstruction

%Obtaining the Mel-Frequency Cepstral Coeficients from HTK  program.
[d,fp,dt,tc,t]=readhtk(filehtk);
mfcc=d';
C0s=mfcc(end,:);
mfcc=[C0s; mfcc(1:end-1,:)];
[ncof,nframes]=size(mfcc);


%Convert  MFCCs to Log Magnitude Mel Power Spectrum
const=log(32768)*sqrt(M/K);
coeffadj=0.5*ncof/M;
ylogmel=coeffadj*idct_htk(mfcc,K)-const;

%Calculate Magnitude Mel Power Spectrum
ymel=exp(2*ylogmel);

%In order to obtain the LP coefficients from MFCCs, The IFFT will be
%applied to the mel spectrum over mel frequency scale considering
%the bandwidth corresponding to each mel frequency.

%Find the size of the equisized mel bins in Hz
melbnd=(0:K)*frq2mel(fs/2)/K;
```

```
fdelta=zeros(1,K);
for i=1:K
    fdelta(i)=mel2frq(melbnd(i+1))-mel2frq(melbnd(i));
end
fdelta=fdelta/(fs/2);
%Find the center frequency of the mel bins
melcnt=((1:K)-0.5)*frq2mel(fs/2)/K;
fsamp=mel2frq(melcnt);
%Calculate the inverse DFT transform matrix
A=cos(((0:K-1)'*fsamp*(2*pi/fs)));
%Do the inverse transformation (weighted by bin size)
% to obtain the autocorrelation coefficients.
for i=1:nframes
    r(:,i)=A*(fdelta'.*ymel(:,i));
end
% Levinson recursion to obtain LP coefficients
[aa,g,k]=levinson(r,p);
% Calculate power spectrum
for i=1:nframes
    ff=rfft(aa(i,:).',2*(Nfreq-1)).';
    Sxx(:,i)=-10*log10((1/g(i))*real(ff.*conj(ff)));
end

f=fs*(0:Nfreq-1)/(2*(Nfreq-1));

%Plot Spectra for the selected frame nf
%1.Plot Magnitude Mel Spectrum
figure
plot(fsamp,10*log10(abs(ymel(:,nf))),'r')
ylabel('Magnitude (dB)')
xlabel('Frequency (HZ)')
hold on
%2.Plot Power Spectrum by converting the MFCCs to LP coefficients
plot(f,Sxx(:,nf),'g')
grid
legend('Magnitude Mel Spectrum',' LP Power Spectrum from MFCCs',
       'Location', 'SouthOutside')
```

Appendix D.4: *LPC_filter.m*

```
function [sw,sv]=LPC_filter(g,a,x,Nshift,Nfrm,p,mode)

%--------------------------------------------------------------------
%   Name:   LPC_filter.m
%   Author: Noelia Alcaraz
%
%   Description: Implement source-filter model for speech production.
%                Generate speech from the LP coefficients when
%                excitation signal modulates voiced sounds, sw, and
%                unvoiced sounds, sw.
%   Input parameters:
%       g -->   Filter Gain
%       a -->   a(:,p+1)LP Coefficients
%       x -->   speech waveform.
%       mode --> 1 filter is implemented by delta coefficients.
%            --> 2 filter is implemented by LPC coefficients.
%--------------------------------------------------------------------

[nframes,ncoef]=size(a);
```

```
%Unvoiced excitation (white noise)
Zi=zeros(p,1); %initial conditions
Zf=zeros(p,1); %final conditions

if mode==1
    Nshift=Nshift/4;
    shft=(Nfrm-Nshift)/2-2;
    shft=round(shft/4);
else
    shft=(Nfrm-Nshift)/2-2;
end

exw=randn(size(x,1)+Nshift,1);
j=1;
for i=1:nframes
    [sw(j:j+shft),Zf]=filter(g(i),a(i,:),exw(j:j+shft),Zi);
    j=j+shft+1;
    shft=Nshift-1;
    Zi=Zf;
end

%Voiced excitation with fixed F0 (pulse train)
Zi=zeros(p,1);

if mode==1
    shft=(Nfrm-Nshift)/2-2;
    shft=round(shft/4);
else
shft=(Nfrm-Nshift)/2-2;
end

exv=zeros(size(x,1)+Nshift,1);
for i=1:120:size(exv,1)
    exv(i)=1;
end
j=1;
for i=1:nframes
    [sv(j:j+shft),Zf]=filter(g(i),a(i,:),exv(j:j+shft),Zi);
    j=j+shft+1;
    shft=Nshift-1;
    Zi=Zf;
end

end
```

Appendix D.5: *deltacoef.m*

```
function [gd,ad]=deltacoef(g,a)

%-------------------------------------------------------------------
%   Name:   deltacoef.m
%   Author: Noelia Alcaraz
%
%   Description:    Delta coefficients are the first time derivatives
%                   that can be obtained by polynomial approximation
%                   (linear interpolation).
%-------------------------------------------------------------------
```

```
[nframes,ncoef]=size(a);

%The interpolation of the gain, g:
x=1:nframes;
xi=1:0.25:nframes;
gd=interp1(x,g,xi);
% gd will now have (N-1)*4-1 samples, i.e. that all original frames
% except the last one will need to be divided in 4 subframes; while
% for the last frame its gain value will be repeated for 4 times.
gd=[gd gd(end) gd(end) gd(end)];

%Interpolation of the filter coefficients.

%First a conversion from autoregressive to reflection coefficients is
% needed to ensure the filter stability after the interpolation.
rf=lpcar2rf(a); %rf(:,p+1)  Reflection coefficients with rf(:,1)=1
%Interpolation of reflection coefficients
rfi=zeros(4*nframes,ncoef);
rfi(:,1)=1;
for i=2:ncoef
y=interp1(x,rf(:,i),xi);
rfi(:,i)=[y';rf(end, i);rf(end,i); rf(end,i)];
end

%Finally, convert the reflections to autoregressive coefficients.
[ad,arp,aru,gr]=lpcrf2ar(rfi);
```

Appendix D.6: *idct_htk.m*

```
function y=idct_htk(x,K)

%-------------------------------------------------------------------------
%   Name:   idct_htk.m
%   Author: Noelia Alcaraz
%
%   Description:    Backward DCT as used by HTK (DCT-II)
%                   y=A'*D*x, where
%                   a(i,j)=sqrt(2/K)*cos(pi*(i-1)*(j-1/2)/K)
%                   and D is a diagonal matrix, diag([0.5 1 … 1]).
%                   Produces K log power spectrum samples from the
%                   input cepstral vector x, which is a (Nx1) column
%                   vector
%-------------------------------------------------------------------------

N=size(x,1);
A=sqrt(2/K)*cos(((0:N-1)'*((1:K)-0.5))*(pi/K));
D=diag([0.5 ones(1,N-1)]);
y=A'*D*x;
```

APPENDIX E: SPECTRAL DISTANCE MEASURE

```
function d2db=spectral_distance(LPspectrum,Sxx)

%---------------------------------------------------------------------
%   Name:   spectral_distance.m
%   Author: Noelia Alcaraz.
%
%   Description:    distance measure for speech recognition based on
%                   rms Log Spectral measure.
%   Input parameters:(two spectral models)
%       LPspectrum --> LP Power Spectrum from original speech waveform
%       Sxx        --> LP Power Spectrum by transforming MFCCs into LP
%                      parameters
%   Output parameters:
%       d2db       --> spectral distance or distortion in dB.
%---------------------------------------------------------------------

[Nfreq,nframes]=size(Sxx);

%The error or difference between the spectra models.
V=log(LPspectrum)-log(Sxx);

%In order to measure the distance between the spectral models, the Lp
%norm is chosen. For p=2, the rms log spectral measure is defined by:

L2=sum(abs(V).^2)/(Nfreq);
d2db=sqrt(L2);

%As the spectra models are in db-spectra domain, the spectral
%distortion is obtained in dB.
```

APPENDIX F: EXCITATION SIGNAL TEST

Appendix E.1: *test1.m*

```
function test1(filewav,filehtk,nf)

%--------------------------------------------------------------------
%   Name:   test1.m
%   Author: Noelia Alcaraz
%
%   Description:    Modify the generative model using as excitation
%                   signal the residual signal obtained with LPC
%                   parameters from original waveform signal.
%                   Test will use generative model 2.
%--------------------------------------------------------------------

%Data from the configuration file of HTK to generate the MFCC_C0
p=12;       %LPC analysis order
Tfrm=25;    %frame size for analysis (ms)
Tshft=10;   %frame shift for analysis (ms)
preem=0.97; %pre-emphasis coefficient
M=24;       %Number of filterbank channels

%Read the speech signal
[x,fs,wmode,fidx]=readwav(filewav);

%LPC analysis to get the residual signal
[aCoef,e,P,G]=proclpc(x,fs,p,Tshft,Tfrm,preem);
    %e --> LPC residual. One column of fs*Tfrm samples representing
    %the excitation or residual of the LPC filter for one frame.

%MFCCs and LPC parameters for the LPC filter.
[mfcc,ymel,fsamp,Sxx,aa,g]=mfcc2spectrum2(filehtk,M,p,nf);
[ncof,nframes]=size(mfcc);

%Implementation LPC-filter (Modified)
Zi=zeros(p,1); %initial conditions
s_resid=[];
for i=1:nframes
    [sre,Zf]=filter(g(i),aa(i,:),e(:,i),Zi);
    s_resid=[s_resid; sre];
    Zi=Zf;
end

% de-emphasize
s_resid=filter(1,[1 -preem],s_resid);

% Play
soundsc(s_resid,fs);
```

Appendix E.2: *test2.m*

```
function test2(filewav,filehtk,nf)
%--------------------------------------------------------------------
%   Name:   test2.m
%   Author: Noelia Alcaraz
%
```

```
%   Description:    Modify the generative model using as excitation
%                   signal one created from the pitch information.
%                   Test will use generative model 2.
%-------------------------------------------------------------------

%Data from the configuration file of HTK to generate the MFCC_C0
p=12;       %LPC analysis order
Tfrm=25;    %frame size for analysis (ms)
Tshft=10;   %frame shift for analysis (ms)
preem=0.97; %pre-emphasis coefficient
M=24;       %Number of filterbank channels

%Read the speech signal
[x,fs,wmode,fidx]=readwav(filewav);
Nfrm=Tfrm*fs/1000;      %number of samples of frame size
Nshft=Tshft*fs/1000;    %number of samples of frame shift.

%LPC analysis to get the residual signal
[aCoef,e,P,G]=proclpc(x,fs,p,Tshft,Tfrm,preem);
    %pitch - A frame-by-frame estimate of the pitch of the signal,
    %calculate by finding the peak in the residual's autocorrelation

%MFCCs and LPC parameter for the LPC filter.
[mfcc,ymel,fsamp,Sxx,aa,g]=mfcc2spectrum2(filehtk,M,p,nf);
[ncof,nframes]=size(mfcc);

%Implementation LPC-filter (Modified)
Zi=zeros(p,1); %initial conditions
shft=(Nfrm-Nshft)/2-2;
exw=randn(size(x,1)+Nshft,1); %unvoiced excitation
exv=zeros(size(x,1)+Nshft,1); %voiced excitation
for i=1:120:size(exv,1)
    exv(i)=1;
end
j=1;
for i=1:nframes
    if P(i)==0
        %unvoiced excitation
        [s_pitch(j:j+shft),Zf]=filter(g(i),aa(i,:),exw(j:j+shft),Zi);
    else
        %voiced excitation
        [s_pitch(j:j+shft),Zf]=filter(g(i),aa(i,:),exw(j:j+shft),Zi);
    end
    j=j+shft+1;
    shft=Nshft-1;
    Zi=Zf;
end

% de-emphasize
s_pitch=filter(1,[1 -preem],s_pitch);

% Play
soundsc(s_pitch,fs);
```