Noise suppression in speech

Group 844

30th May 2005
Abstract

This project develops an algorithm to suppress the noise from a noise contaminated speech signal for mobile phone users. The one microphone scenario is chosen with an assumption that noise is additive and that the speech signal and noise are uncorrelated. The spectral subtraction technique based on minimum statistics, is a noise power spectrum estimator and subtracts noise spectrum from noisy speech signal spectrum to get the estimate of noise suppressed speech spectrum. The spectral subtraction, the spectral manipulation and residual manipulation methods are implemented. The proposed algorithm is capable of tracking different noise levels, results in improvement of signal to noise ratio and perceptual quality and the intelligibility is maintained. The algorithm is analyzed for real time and tested according to objective evaluation criteria. The algorithm is chosen for mapping on an architecture of TMS320C6713 DSP processor.
Preface

This document reports on the work of group 844\textsuperscript{1} in the 8th semester. This report is organized into two parts, main report and appendix. The main report contains seven chapters. The first chapter starts as introduction and chapter 2 provides the overview of problem modeling and problem analysis of the project is clearly outlined. Third chapter is a review of requirements and specification. In fourth chapter, the design methodology of the project using $A^3$ model is described. In the chapter five, the algorithm of the project is described. Finally chapter 6 and chapter 7, provides the implementation and conclusion of the project. The second part of the report contains appendices with project relevant, like linear predictive coding, speech and noise properties and about DSP processor TMS320C6713. This report ends with Matlab programs. All the associated material, the Matlab code and C-programs along with project report are placed in the CD.

Aalborg, 30th May 2005.

\begin{center}
Raju Muchanthula
\end{center}

\textsuperscript{1}Student of Applied Signal Processing and Implementation (ASPI).
Contents

1 Introduction ........................................... 1

2 Problem Analysis ...................................... 2
   2.1 Problem Modeling ................................... 2
   2.2 Speech Production .................................. 2
   2.3 Process of Linear Predictive Coding ................. 4
   2.4 Noise Contaminated Signals ......................... 5

3 Requirements and specifications ......................... 7
   3.1 Application requirements ............................ 7
   3.2 Algorithm requirements .............................. 7
   3.3 System test ......................................... 10

4 System design ........................................ 11
   4.1 The A³-model ........................................ 11
   4.2 Application ......................................... 12
   4.3 Algorithm .......................................... 12
   4.4 Architecture ........................................ 14
   4.5 Test bed design ...................................... 14

5 Algorithm ............................................. 17
   5.1 Spectral Subtraction - Based on Minimum Statistics . 17
   5.2 Spectral Manipulation ................................ 21
   5.3 Residual Manipulation ............................... 22

6 Implementation ...................................... 24
   6.1 Overview ........................................... 24
   6.2 DSP Architecture Mapping ........................... 32
# Contents

7 Conclusions ................................................. 37
   7.1 Results ............................................. 37
   7.2 Conclusion ........................................... 37

A Linear Predictive Coding ................................. 38
   A.1 Introduction .......................................... 38
   A.2 Linear Prediction ...................................... 38
   A.3 All Pole Model ......................................... 39
   A.4 Autocorrelation Method ................................. 40
   A.5 Covariance Method ..................................... 41
   A.6 Levinson Durbin Algorithm ............................. 42

B Speech and Noise ............................................ 43
   B.1 Speech ................................................ 43
   B.2 Noise .................................................. 43

C DSP ............................................................ 45
   C.1 TMS320C6000 DSP platform ............................ 45
   C.2 Functional Units ........................................ 45
   C.3 Register File ........................................... 46
   C.4 TMS320C6713 DSP ....................................... 46
   C.5 C6713 Memory and Peripherals ......................... 47
   C.6 TMS320C6713 DSP Starter Kit (DSK) .................... 47
   C.7 Tools and Software ...................................... 48
   C.8 Algorithm Standard ..................................... 48
   C.9 Terminology ............................................ 48

Bibliography .................................................. 52
Introduction

The Advances in telecommunications over the last few decades has dramatically affected the way people live and communicate. Technological progress has made communication systems reliable and affordable and mobile communication has now become omnipresent. Resulting in the freedom and flexibility which introduces new challenges, one of the most prominent is being the suppression of background noise. Mobile phones often used in noisy environments, the result being additional unwanted noise signals are transmitted with the speaker’s voice over the channel. At the receiving end the speaker’s voice can become unintelligible amongst the noise. Researchers have been working for decades to suppress background noise as much as possible for GSM instrument users and to positively influence the intelligibility of speech in such noisy environments. Suppression of the background noise is important not only to improve the quality but also maintain the intelligibility of speech [10].

The problems with noise are quite complex. Noise is composed of several different sounds that are distinguishable from one another by temporal and spectral differences. Noise components are classified into different types. Convolution noise, the unwanted echo in hands free telephone system it can be called an acoustic noise or echo, which is removed or reduced using adaptive filters. The other types of noises are additive noise and noise due to non-linearities. Additive describes the nature of how a desired signal and the noise energy are summed at the receiver input (there are impairment that are multiplicative in nature).

Various methods were suggested overcoming the effect of noise on speech communications [10]. A single microphone scenario is used to analyze the speech data.

In microphone-based applications, which have a potential for incoming speech getting corrupted by ambient noise captured by the microphone. It is especially suitable for systems in which an acoustically isolated noise reference is not available, such as: Hands-free Cell Phone Kits, Speakerphones, Intercoms, Teleconferencing Systems, Headsets, As a front-end to a Speech Recognition system, Any microphone-based application that needs to eliminate undesired noise.

The scope of the project is to reduce the background noise sufficiently from a given noisy speech. To get the best noise reduction, an algorithm running in real time is chosen for the processing. The algorithm is too implemented on the Texas Instruments DSP Kit, TMS3206713 with suitable changes according to the architecture provided.
Problem Analysis

In this chapter, the production of speech will be introduced. Furthermore transmission of noise free and noise contaminated signals will presented. This leads to an overview of the system. The problem analysis consists of two sections. The first section concerned with problem modeling. The later sections deal with speech generation, linear prediction, solution to optimal predictor and transmission of noise contaminated signal.

2.1 Problem Modeling

The goal of this project is to develop an algorithm that can reduce the background noise to facilitate conversation in noisy environment. The proposed system uses one microphone for the input of the system. An illustration of the system can be seen in the figure 2.1. The input signal to the microphone is noisy speech. In the filter, an estimation of the background noise is subtracted from the signal picked by the microphone, so the output is noise suppressed speech signal. In this project the major factor but maintaining the perceptual quality and noise reduction are the main tasks.

In the following section, the speech generation and analysis are described before analyzing the problem. The production of speech can be modeled with three blocks [3, Ch.2 and 3]; the generator, the vocal tract and the radiation. These blocks are depicted in figure 2.2.

2.2 Speech Production

Physical speech production originates in the lungs, where the airflow begins. The actual sound is formed while air flows through the larynx and vocal tract. The larynx consists of the cricoids cartilage, vocal folds and arytenoids cartilage. The vocal tract can be divided
2.2. SPEECH PRODUCTION

into three areas: oral pharynx, nasal cavity and mouth. The tongue, lower jaw and lips have the most effect on the form of the vocal tract to produce distinct sounds.

![Block diagram of speech production.](image)

 depending on the activity status of the vocal folds, the sounds can be roughly divided into two extreme cases [6]. First, the vocal folds can open and close periodically, generating in this way a train of pulses (glottal pulse). This gives the sound its voiced nature, i.e. periodicity in time and harmonic structure in frequency. Second, the vocal folds can just be open, with the airflow forming turbulence between the folds and the sound very noise-like. Unvoiced sounds are formed this way. The excitation of the voiced sounds is usually modeled by the train of pulses, whereas unvoiced sounds use random noise excitation.

The figure 2.2, is an illustration of the applicability of linear prediction for speech modeling [5]. The vocal tract model $H(z)$ and lip radiation model $R(z)$ are excited by a discrete glottal excitation signal. For voiced speech the source excitation is an impulse train generator driving a glottal shaping filter and using local pitch period estimation. For unvoiced speech the source excitation comes from a random noise generator.

Now, the knowledge about the generator can be used to analyze the speech. Production of speech can be modeled as an AR-model, as the vocal tract can be modeled as an all-pole filter [11]. Since the signal from the generator is assumed white, the inverse of the vocal tract transfer function can be used to whiten the signals. The whitening filter can be found using linear prediction, which is covered in detail later in this chapter and in appendix A. For voiced sounds the glottis introduces two extra poles. It is desired to remove the effect of these poles, in order to focus the parameters of the AR-model on the vocal tract. The radiation can be modeled as a single-zero filter, which causes one of the poles to be canceled. A preemphasis filter with the transfer function

$$R(z) = 1 - z_0 \cdot z^{-1}$$  \hspace{1cm} (2.1)

with $0.9 < z_0 < 1$ will cancel the other pole [3, p. 330]. This filter must be applied prior to modeling the vocal tract.

The approach described here is model-based or parametric in contrast to a non-parametric approach. The advantage of this approach is that it is possible to manipulate e.g. the power spectrum or autocorrelation function without destroying the speech estimate since the information about the pitch period is still maintained in the residual.
2.3 Process of Linear Predictive Coding

Linear predictive coding is a way of representation of speech which can be transmitted efficiently through a digital channel. This method is based on linear predictive analysis. The basis is the vocal tract which is modeled as an all-pole filter. The purpose of LPC in speech is shown in figure 2.3. The analysis results a set of LP coefficients and a residual signal. Instead of transmitting the whole signal, the residual signal is encoded and transmitted together with LP coefficients. At receiving end the speech is synthesized by LP coefficients and residual signal.

![Figure 2.3: Block diagram of Process of LPC](image)

2.3.1 Linear Predictive Analysis

Linear prediction analysis of speech is one of the most important speech analysis technique. The assumption is that speech is short time or locally stationary for analysis. The analysis is an extraction of prediction Coefficients, i.e an determination of formants created by vocal tract. As shown in the figure 2.4 the speech is filtered with a prediction error filter. The predictor is to predict the actual sample based on linear combination of the past samples. If the predictor is optimal, the residue is spectrally white. The predictor optimization is described in appendix A.

![Figure 2.4: Block diagram of the prediction error filter.](image)

2.3.2 LPC synthesis

Till now the focus was on analyzing the signal with the purpose of identifying the parameters of a system, satisfying AR constraint by minimizing the prediction error. If the prediction error is 'white', then the estimated signal is good fit, so the synthesized signals having similar statistical properties as the original one. Then exciting the synthesis filter...
with the system function $H(z)$ using a white noise signal, the filter’s output will have a spectrum close to the original signal. The process is shown in the figure 2.5.

$$H(z) = \frac{1}{1 + \sum_{i=1}^{M} a_i z^{-i}} = \frac{1}{A(Z)} \quad (2.2)$$

In the equation 2.2, $a_i$ are the LP coefficients found from the original signal. The transfer function $1/A(z)$, $G$ is a gain factor and it is excited by the input signal $e(n)$, and thereby producing the speech signal $s(n)$.

![Figure 2.5: The inverse system identification problem.]

### 2.4 Noise Contaminated Signals

If the speech signal is corrupted by noise, the all-pole model no longer applied to approximate the speech spectrum estimation closely and simply. Assume that the speech signal is corrupted by an additive and uncorrelated background noise, then the signal model become a pole-zero model, is also called as autoregressive moving average (ARMA) model. In a system with pole-zero model the estimation of the parameters is inherently nondeterministic and nonlinear [6]. There are few solutions for pole-zero model like Gauss-Newton method, but it is not guaranteed for the optimal solution [2, p.326]. So spectral subtraction is considered as a solution to suppress the background noise.

#### 2.4.1 Proposed Solution

![Figure 2.6: LPC-transmission using spectral subtraction]

Spectral subtraction, a method to estimate the spectrum of a signal observed in additive, uncorrelated noise. The estimate is obtained by subtracting an estimate of the noise spectrum from the noisy signal spectrum. The estimation of the noise is difficult because of
the non stationarity of the process. Based on few assumptions speech is considered as locally stationary for analysis [12, p. 208]. The average noise power is approximately the same prior to speech activity as during speech activity. With these assumptions, the estimate of the signal spectrum $P_s(\omega)$ is obtained by [9, p.114]:

$$P_s(\omega) = P_x(\omega) - P_n(\omega),$$  \hspace{1cm} (2.3)

where $P_x(\omega)$ is the spectrum of the noise contaminated signal and $P_n(\omega)$ is the noise spectrum estimated during no speech activity. It is difficult to remove entire noise by this method, as the assumptions are not entirely correct. Some of the noise components are not additive resulting in a correlation between the speech and noise in someway. Due to these reasons, the performance of the method depends on the estimation of the noise spectrum [9].

The classical spectral subtraction method does not deal with signal phase information. Same noisy signal phase is used to reconstruct the estimate of the speech signal. Even the manipulation of the magnitude spectrum distorts the pitch period. Thus, another model is chose which is based on linear prediction. This model uses the envelope of the amplitude spectrum without considering the phase information. The proposed method i.e., spectral subtraction using LPC is shown in figure 2.6.
Requirements and specifications

As mentioned in the problem analysis, chapter 2 the project deals with the noise suppression. So the requirements for the development and implementation of the algorithm are mentioned in this chapter. With these desired requirements, the specifications for the algorithm implementation are discussed.

3.1 Application requirements

The system is only considered as a noise suppression device, which gives the two main objectives:

- The SNR of the signal should be improved.
- Intelligibility /perceptual quality should be maintained.

The boundaries specified for SNR are between 0 dB and 35 dB, as mentioned in [3, p.586]. The limit is considered so as the system can function efficiently.

3.2 Algorithm requirements

The system described in the Problem Analysis utilizes a model based description of the speech production. It is thereby possible to set up specific requirements to the parameters in this model. With starting point in figure 2.6, requirements to the different parameters will now be presented.

3.2.1 Frame characteristics

The speech is assured to be quasi-stationary resulting in the specification of the type of window and its length, amount of overlap between frames.

- **Frame Size**: The frame size had been set to 25\textit{ms}, if the analysis frame shorter than 20\textit{ms} results in roughness, while increasing the frame size decreases the musical
3.2. ALGORITHM REQUIREMENTS

(a) Illustration of frames, burst and overlap.

(b) Illustration of the adjustable delay between frames and bursts. As seen, the bursts form a time continuous signal.

Figure 3.1: The relation between frames, bursts and overlap.

noise considerably. If the frame size is too long, it results slurring [12]. The frame and burst are shown in the figure 3.1.

- **Window Selection:** The window determines the portion of the speech signal that is to be processed by zeroing out the signal outside the region of interest. The ideal window frequency response has a very narrow main lobe which increases the resolution, and decreases the side lobes (or frequency leakage). Since such a window is not possible in practice, a compromise is usually selected for each specific application.

There are many possible windows, such as rectangular, hanning, hamming window. The rectangular window has the highest frequency resolution due to the narrowest main lobe, but has the largest frequency leakage. Due to the high frequency leakage produced by the larger side lobes, rectangular windowed speech looks more noisy than hamming or hanning windowed speech. This undesirable high frequency leakage between adjacent harmonics tends to offset the benefits of the flat time domain response of the rectangular window. As a result, rectangular windows are not usually used for speech spectral analysis. On the other hand, the trapezoidal windows such as hamming and hanning window has smaller frequency leakage but lower resolution. So, they produce a smoother spectrum than the rectangular window [4].

Since LP model is determined on frame-by-frame basis, original signal should be segmented into frames. The segmented signal is also required to be windowed...
frame by frame. A properly windowed frame is easier to analyse in frequency domain. Additionally, frequency leakage of main lobe is reduced by trapezoidal window, such as hamming window or hanning windows. The window size is selected as length of the frame. Determination of window type should be done experimentally by doing Matlab simulations with different types of windows.

- **Frame overlap**: When sliding the window through the signal, the overlap is necessary to prevent discontinuities at frame boundaries. The amount of overlap is usually taken to be 20% of the frame size. For large frames, 10% may be excessive and might causes slurring of the signal. So amount of overlap will be determined from experimental results [12].

### 3.2.2 Analysis filter

The purpose of the analysis filter is to analyze the signal and identifying the parameters of a system, satisfying AR constraint by minimizing the prediction error. If the prediction error is 'white', then the estimated signal is good fit, so the synthesize signals having similar statistical properties as the original one. The objective of the analysis filter is to cancel the characteristics of the vocal tract in the observed signal and by it reproducing an estimate of the excitation signal at the output of the filter. This can also be considered as canceling the formants in the observed signal, which is the same as minimizing the error signal. In order to do this, it is necessary to have enough coefficients in the analysis filter to model the formants. In [3, fig. 5.7c] it is seen, that with a sampling frequency of 8 kHz, the residual energy is not further minimized when the prediction order exceeds 10. Furthermore, an algorithm for calculating the LP coefficients should be suited for real-time implementation. A natural choice would therefore be the Levinson-Durbin algorithm, which also applies to the LPC-block.

### 3.2.3 Manipulation of residual

In the case of a noise contaminated speech signal as the observed signal, the AR-model analysis filter will have a noisy residual as an output. Manipulation of this noisy residual should lead to an emphasis of the voiced segments i.e. the impulses in the residual.

### 3.2.4 FFT order

The length of the FFT must be long enough to give a sufficient frequency resolution. The preferred length is 256 samples. The minimum FFT order corresponding to a given frame size is adequate [12].

### 3.2.5 Power Spectral Subtraction

This block is the main SNR improving part of the system and the ideal goal is to entirely remove the noise spectrum from each analyzed frame, which is not possible. The output
3.3. SYSTEM TEST

should not be a time estimate of the speech signal, which is the classical usage of spectral subtraction, but instead an estimated power spectrum of the speech signal to be used in the IFFT-block in order to obtain the autocorrelation. From the autocorrelation, the LP coefficients are calculated.

The block should be able to estimate the noise spectrum under different types of noise and varying noise levels.

3.2.6 Manipulation of power spectrum

Since it is not possible to entirely remove the noise from the observed signal, manipulation of the estimated speech power spectrum is necessary. The results of this manipulation should be an estimated speech power spectrum in which the randomly varying parts due to noise is further reduced.

3.2.7 Synthesis filter

After spectral subtraction, power spectrum manipulation, and autocorrelation (ifft), a estimated time wave of speech is reconstructed using the manipulated LPC coefficients and manipulated noisy residual. Then the synthesis filter’s output will have a PSD close to the original signal as long as the prediction order 10 is adequate.

3.3 System test

The purpose of the system test is to test the system as a black box. There will be performed a test of the individual parts of the system.

The system test will be carried out with a noise contaminated speech signal. The test will show whether the requirements are maintained, with respect to SNR, intelligibility.
This chapter describes the system design. The design is based on Rugby model. In relation with rugby model the $A^3$ model, Application, Algorithm and Architecture are discussed. The application is described for real time implementation. Algorithm is divided into two parts, burst based and sample based. The architecture is Texas Instruments DSP, TMS320C6713 DSK [13].

4.1 The $A^3$-model

The Rugby model is a conceptual frame work, in which designs, design processes are expressed in order to analyze the problem. This model is used to evaluate the design covers all the domains at the various abstraction levels. Rugby model has four domains for design process, namely Computation, Communication, Data, and Time.

As shown in the figure 4.1 rugby model has starts with an idea or project proposal (higher abstraction level) and goes through four domains with different abstraction levels to develop the (lower abstraction level) final system. All abstraction levels are treated these domains resulting in the final design. In this project the system design is based on the $A^3$-model as shown in the figure 4.2. The model takes into consideration that the algorithm may be modified before moving to the architecture domain. Furthermore the move from the algorithm domain to the architecture domain i.e the feed back from the fixed architecture can be performed several times.

![Figure 4.1: The Rugby meta model in relation with the $A^3$-model.](image-url)
4.2 Application

The application of the project is to develop a noise suppression algorithm, which is must be an adaptive for give architecture. The system black box is illustrated in the figure 4.3. While considering the timing constraints, the system is expected to run as a real time system. For that latency must not exceed 30ms, in order to the audio or visual requirement. The input is a noise contaminated speech signal, $x[m]$, and the output is an estimate of the noise free speech signal, $\hat{s}[m]$.

![Figure 4.2: The A³-model.](image)

4.3 Algorithm

The computational complexity and desired output are considered as the most important factors for choosing the algorithm. Based on the problem analysis and the application description, the system is partitioned into processes. The processes are groupings of functionalities, which are defined in the problem analysis.

The system consists of different parts, that will be executed with different inputs. One part of the system is running as sample based execution, while the other two parts run as burst based and frame based execution. In the sample based part, circular buffers are available, providing the possibility of the frame overlap. The overview of the algorithm is shown in the figure 4.5. These different processes of the algorithm to be analyzed step by step. The processes and the interface between these are described in short terms in table 4.1.

The next step in the design process is a partitioning of the processes into functions. It is defines which processes are concurrent and therefore can be executed in parallel. It also describes how parameters are be passed between the functions.
<table>
<thead>
<tr>
<th>Process</th>
<th>Input</th>
<th>Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preemphasis filter</td>
<td>Speech samples, $x[m]$</td>
<td>Blocks of speech samples, either frame or burst.</td>
<td>Accounts from the effect of the lip/radiation from the mouth. The outputs are related as depicted in figure 3.1(b)</td>
</tr>
<tr>
<td>LPC</td>
<td>Block of data, frame</td>
<td>LP coefficients, $a$</td>
<td>Calculates the LP coefficients for the analysis filter.</td>
</tr>
<tr>
<td>Analysis filter</td>
<td>Block of data, burst, and LPC-coefficients, $a$</td>
<td>Noisy residual, $\hat{e}[m]$</td>
<td>Whitens the speech signal and creates the residual using prediction error filtering.</td>
</tr>
<tr>
<td>Residual manipulation</td>
<td>Noisy residual, $\hat{e}[m]$</td>
<td>Estimated residual, $\hat{e}[m]$</td>
<td>Suppresses noise in the residual signal.</td>
</tr>
<tr>
<td>Synthesis filter</td>
<td>Estimated residual, $\hat{e}[m]$ and LP coefficients, $\hat{a}$</td>
<td>Estimated speech samples, $\hat{s}_b[m]$</td>
<td>Synthesizes the speech signal from the residual and the LPC-coefficients.</td>
</tr>
<tr>
<td>Power spectrum</td>
<td>Frame of data, frame</td>
<td>Power spectrum, $P_s(w)$</td>
<td>Calculates the power spectrum of the speech frame.</td>
</tr>
<tr>
<td>Spectral subtraction</td>
<td>Power spectrum, $P_s(w)$</td>
<td>Noise suppressed power spectrum, $\tilde{P}_s(w)$</td>
<td>Suppresses noise in the power spectrum.</td>
</tr>
<tr>
<td>Spectrum manipulation</td>
<td>Noise suppressed power spectrum, $\tilde{P}_s(w)$</td>
<td>Estimated speech power spectrum, $P_s(w)$</td>
<td>Estimates the power spectrum using a model-based approach.</td>
</tr>
<tr>
<td>IFFT</td>
<td>Estimated speech power spectrum, $P_s(w)$</td>
<td>Autocorrelation function, $r_s(\tau)$</td>
<td>Calculates the autocorrelation from the power spectrum.</td>
</tr>
<tr>
<td>LPC</td>
<td>Autocorrelation function, $r_s(\tau)$</td>
<td>LP coefficients, $\hat{a}$</td>
<td>Calculates the LP coefficients from the autocorrelation function.</td>
</tr>
<tr>
<td>Deemphasis filter</td>
<td>Estimated burst of speech samples, $\hat{s}_b[m]$</td>
<td>Estimated speech samples, $s[m]$</td>
<td>The inverse of the preemphasis filter.</td>
</tr>
</tbody>
</table>

Table 4.1: Description of the processes and their interfaces.
4.4 Architecture

When mapping the algorithm to the architecture, several aspects need to be considered, like how to implement the chosen in an optimal manner for given architecture on different abstraction levels. The level of implementations are shown in the figure 4.4.

The basic algorithm implementation and testing to be done in MATLAB. Then the matlab code is converted into C. Considering the architecture the C and Matlab functions could be optimized with minimum changes. To facilitate functional debugging and easy to access to the internal variables of the algorithm. This should be ensure compatibility with the C-compiler included in CCS, allowing the code to be transferred with alternations.

In the next section a test bed will be designed, so that there is a proper way of testing the system.

4.5 Test bed design

To be able to test the algorithms, and thereby the system, a testbed design environment must be designed.

The design of the testbed is based on stepwise refinement, so that the system at the highest level of abstraction is tested first. A single process or function will be tested at a time, keeping the number of adjustable parameters to a minimum. It must be emphasized, that each process and function will have to fulfill the requirements in the requirements specification.
4.5. TEST BED DESIGN

Figure 4.5: The black box of the system.
First a Matlab model will be developed, so that a fully functioning mathematical model can be tested. The Matlab model gives the possibility to implement one module at a time, replacing the Matlab functions with C-functions and test the C-functions in connection with the already working Matlab-functions. Thereby, only one function is changed at a time, making it possible to track errors down to a single function.

The processes and their functions will be tested on the fixed architecture. This will be performed by writing some test-data in memory, and then running the function. The test data will be the same data that is used when testing the Matlab functions.

After the testbed has been designed, the design of each process in the algorithmic and architecture domain will be performed. The test of the process will performed in connection with this design.
The purpose of this chapter is to introduce the algorithms that will model and analyze the performance. As mentioned in system design this chapter is a division of the algorithm into number of blocks, named spectral subtraction, spectral manipulation and residual manipulations. For more related detail information on implementation see chapter 6.

5.1 Spectral Subtraction - Based on Minimum Statistics

The performance of speech signals degrades in the presence of noise. To reduce the degradation, a model is proposed to represent the spectral changes of speech signal uttered in noisy environments. The spectral subtraction method is a well-known noise reduction technique. The standard algorithms for spectral subtraction usually need a voice activity detector (VAD), such that the noise spectrum can be estimated during non-speech activity. But VAD’s performance is often degraded considerably under noise conditions. Most implementations and variations of the basic technique advocate subtraction of the noise spectrum estimate over the entire speech spectrum. In general noise is mostly colored and does not affect the speech signal uniformly over the entire spectrum. This chapter describes an alternative robust algorithm, which make use of minimum statistics while estimating the noise spectrum. The described algorithm is based on the article [8].

5.1.1 Overview

Spectral subtraction is a noise power estimator and subtraction rule which translates the SNR into a spectral weighting factor, such that low SNRs are attenuated and high SNRs are not modified. Spectral subtraction method is computationally efficient.

The assumption is that the power spectrum of signal is corrupted by uncorrelated noise, is equal to the sum of a speech spectrum and noise spectrum.

5.1.2 Terminology

Below are some of the more common terms that are encountered when discussing the spectral subtraction algorithm.

- **Spectral floor**: The spectrum components of the processed signal below a certain lower bound is called spectral factor.
- **Musical noise**: The noise which is generated in the spectral subtraction process.
5.1. SPECTRAL SUBTRACTION - BASED ON MINIMUM STATISTICS

- **Short time power**: The estimation of the power using the short-time.
- **Broadband noise**: In the frequency domain, a broadband noise has a continuous spectrum, that is present at all frequencies in a given range. This type of sound often referred to as noise because it usually lacks a discernible pitch.
- **Comb filter**: The filter has multiple pass bands and stop bands. It has a frequency response with a periodic function of $w$ and a period $2\pi/L$, where $L$ is a positive integer.

5.1.3 Basic idea

In the spectral subtraction, the minimum of subband noise power with a finite window to estimate the noise floor. A periodogram of $P_x(\lambda, k)$ is shown in the figure 5.1, the short time power estimate of noisy speech signal shows distinct peaks and valleys.

![Figure 5.1: Periodogram, $P_x(\lambda, k)$, for $f_s = 8$kHz ($N_{FFT} = 256, k = 25$) of the input signal](image)

The basic idea behind the algorithm is an utilization of these peaks and valleys. The peaks are assumed to speech activity and the valleys are used to obtain the estimation of subband noise power. To obtain the noise power estimates the data window for the minimum search must be large enough to bridge peaks of speech activity.

5.1.4 Algorithm overview

A block diagram describing the implemented algorithm for spectral subtraction is shown in figure 5.2.
5.1. SPECTRAL SUBTRACTION - BASED ON MINIMUM STATISTICS

The signal \( x[k] \) is assumed that the sampled and windowed signal, \( x[k] \) is sum of a zero mean speech signal \( s[k] \) and a zero mean noise signal \( n[k] \) is:

\[
x[k] = s[k] + n[k]
\] (5.1)

Further assumption is that, speech and noise are statically independent. The data window \( w[k] \) and the short-time fourier transform of \( x[k] \) is a given by,

\[
x_w[\lambda, k] = x[\lambda, k]w[k] \quad \text{for} \quad k = 1, \ldots, L_{\text{window}}
\] (5.2)

where \( \lambda \) indicates the time index, \( k \) indicates the frequency bin index and the discrete frequencies \( \Omega_k = \frac{2\pi k}{N_{\text{FFT}}} \), \( k = 0, 1, \ldots, N_{\text{FFT}} - 1 \). By computing the FFT of individual windowed frames of length \( N_{\text{FFT}} \). Then the short-time power spectrum from the equation 5.1 is derived as:

\[
P_x(\lambda, k) = P_s(\lambda, k) + P_n(\lambda, k)
\] (5.3)

where \( P_s(\lambda, k), P_n(\lambda, k) \) and \( P_n(\lambda, k) \) are the short time power spectrums of the noise contaminated speech, the speech and the noise signal respectively.

### 5.1.5 Subtraction Rule

To obtain \( |X(\lambda, k)|^2 \), the smoothed estimate of short time power of \( x_w[\lambda] \) with a first order recursive network is given by,

\[
|X(\lambda, k)|^2 = \gamma|X(\lambda - 1, k)|^2 + (1 - \gamma)|X(\lambda, k)|^2
\] (5.4)

Where \( \gamma \) is smoothing constant (\( \gamma \leq 0.9 \)). Now the aim is to estimate the short time magnitude spectrum of the clean speech signal, \( |\hat{S}(\lambda, k)| \). This can be done by using following subtraction rule:
5.1. SPECTRAL SUBTRACTION - BASED ON MINIMUM STATISTICS

\[
|\hat{S}(\lambda, k)| = \begin{cases} 
  k_{\text{subf}} \cdot P_n(\lambda, k) & \text{if } |X(\lambda, k)| \cdot Q(\lambda, k) \leq k_{\text{subf}} \cdot P_n(\lambda, k) \\
  |X(\lambda, k)| \cdot Q(\lambda, k) & \text{otherwise}
\end{cases}
\] (5.5)

\(k_{\text{subf}}\) is a spectral floor constant. It prevents the spectral components of \(P_x(\lambda, k)\) from descending below the lower bound \(k_{\text{subf}} \cdot P_n(\lambda, k)\). The variable gain factor \(Q(\lambda, k)\) is given by

\[
Q(\lambda, k) = 1 - \sqrt{\frac{k_{\text{osub}}(\lambda, k) \cdot P_n(\lambda, k)}{|X(\lambda, k)|^2}}
\] (5.6)

where \(k_{\text{osub}}(\lambda, k)\) is an oversubtraction factor, which is a function of the signal-to-noise ratio, \(\text{SNR}\).

5.1.6 Short time noise power estimation

An estimation of the short time noise power \(P_n(\lambda, k)\) is done from the short time power \(P_x(\lambda, k)\), which is estimated by filtering the instantaneous short time power \(|X(\lambda, k)|^2\), to obtain a smoothed estimate i.e.,

\[
P_x(\lambda, k) = \alpha P_x(\lambda - 1, k) + (1 - \alpha) |X(\lambda, k)|^2
\] (5.7)

where \(0.9 \leq \alpha \leq 0.95\) is a smoothing constant. The smoothing with a fixed parameter \(\alpha\) widens the peaks of speech activity of the smoothed estimate \(P_x(\lambda, k)\). This leads to inaccurate noise estimate. Using (5.7) \(P_n(\lambda, k)\) is estimated by finding a weighted minimum \(P_{x,\text{min}}(\lambda, k)\) within the last \(D\) frames of \(P_x(\lambda, k)\). The estimate is given by

\[
P_n(\lambda, k) = k_{\text{omin}} P_{x,\text{min}}(\lambda, k)
\] (5.8)

\(k_{\text{omin}}\) is a factor to compensate the bias of the minimum estimate. The minimum is obtained as a comparison between a minimum, calculated every \(M\) frames, of the last \(D\) frames of \(P_x(\lambda, k)\) given by

\[
P_{D,\text{min}}(\lambda, k) = \min_j \{P_x(\lambda, k)\} \quad j = i - D + 1, \ldots, i
\] (5.9)

and the actual value of \(P_X(\lambda, k)\). The minimum is obtained as the minimum of the last \(W\) minimums, that is \(D = W \cdot M\). By splitting up \(D\) the minimum is obtained after only \(M\) frames instead of after \(D\) frames.

5.1.7 Estimation of SNR

To control the oversubtraction factor \(k_{\text{osub}}(\lambda, k)\), the SNR in each subband is computed. If the SNR is high, the oversubtraction level is low and vice versa. An estimate of the SNR is:

\[
\text{SNR}(\lambda, k) = 10\log \left( \frac{P_X(\lambda, k) - \min(P_n(\lambda, k), P_x(\lambda, k))}{P_n(\lambda, k)} \right)
\] (5.10)
Large values of $k_{osub}$ leads to speech distortion. As shown in the figure 5.3, the optimal value of $k_{osub}$ is found for best noise reduction with less amount of musical noise is smaller for higher $SNR$. For $k_{osub} = 1$, the $SNR \geq 35$ dB and as increasing the $k_{osub}$ value the $SNR$ will be $SNR \leq -5$ dB.

The $SNR$ is calculated for each short-time frame, so the $k_{osub}$ value also varies for frame to frame. The actual value of $k_{osub}$ used in equation 5.6 is given by:

$$k_{osub} = k_0 - (SNR)/s, \quad -5 \leq SNR \leq 35,$$

where $k_{osub}$ is value of $k_0$ at $SNR = 0$ and $1/s$ is slope of the line.

### 5.2 Spectral Manipulation

After spectral subtraction, the estimated speech power spectrum still contains noise. This noise can be suppressed by spectral manipulations. The manipulation can be done in two ways:

- Spectral smoothing
- Floor setting

#### 5.2.1 Spectral Smoothing

The noise components, which abruptly comes up for a short time in the power spectrum are observed as noise. This type of noise can be reduced by spectral smoothing in time domain. For smoothing, the power spectrum is filtered with a lowpass filter. The designed lowpass filter should be able to suppress the short time varying components or noise components in the power spectrum without disturbing the formant frequencies and short time stationarity. The smoothed spectrum should be able to produce noise suppressed IPC
coefficients for synthesis. The smoothing in the time domains is capable of changing or modifying the amplitudes of the spectrum. This process can be done by filtering.

If the estimated speech spectrum contains rapidly or slowly varying frequency components compared to original speech spectrum, in such cases it is difficult to smooth the spectrum. This effect can be controlled by spectral smoothing in frequency domain. So the estimated speech spectrum is needed to be filtered with a lowpass filter.

By filtering the spectrum, it should capable to model the envelope of the estimated speech spectrum. In the filtering process, it is needed to maintain the frequency components should not be delayed. The same formats should retained even after smoothing.

![Figure 5.4: Autocorrelation of noisy residual signal](image)

5.2.2 Spectral Floor Setting

In the spectral subtraction method, subtracting an estimate of the noise power spectrum from the noisy speech power spectrum results in estimated speech power spectrum and setting the negative differences to zero [12].

5.3 Residual Manipulation

As described in system design, the algorithm is divided in two parts. Till now, the discussion and description was about first part, spectral subtraction and spectral manipulation with frame based execution. Now, the residual manipulation using burst based execution.

The ultimate challenge is trying maximum possible ways to suppress the background broadband noise and increase the SNR in which residual manipulation also plays major role.

The corrupted harmonics in noisy residue are regenerated by filtering with a comb filter. But the comb filter needs a pitch detector for detecting the pitch period in the residue.
This leads to a problem as sometimes it detects wrong pitch period and even it is quite complex to construct a comb filter to suppress the noise level in the residue. So in order to avoid the pitch detection, another method called autocorrelation is considered to reach the desired response.

The autocorrelation of a noisy residual signal is shown in the figure 5.4. The duration between the impulses is considered as pitch period. And the corresponding power spectrum is shown in the figure 5.5. The power spectrum clearly shows that the frequency components are amplified. Now, by filtering the spectrum, the noise level of the residual signal is reduced.
The purpose of this chapter is to describe the implementation of algorithm as mentioned in chapter 4 that we will analyze the performance. The purpose and functionality of each individual block of Matlab model is described. The implementation is performed on different development levels: First the basic model is in Matlab, C in Matlab and C in 6713DSK.

6.1 Overview

As discussed in design chapter to implement the chosen algorithm in an optimal manner. In this chapter the criteria is to fit the algorithm for a fixed architecture of 6713DSK. The system is tested for both functionality and performance.

6.1.1 Input Signal model

Different types of signals are considered for implementing the algorithm.

- Speech signal without noise.
- Speech signal with an additive uncorrelated noise.

The clean speech signal of male speaker who speaks a sentence “watch the log float in the wide river”. The additive noise signal is a pink noise. The figure 6.1 shows noise free signal and noisy speech signal.

6.1.2 Circular buffer

The circular buffers are used in the algorithm for real-time implementation. Digital signal processors which support circular buffers automatically generate and increment pointers for memory accesses which wrap to the beginning of the buffer when its end is reached, thus saving the time and instructions otherwise needed to ensure that the address pointer stays within the boundary of the buffer, and speeding the execution of repetitive DSP algorithms [14].
Figure 6.1: Speech Analysis
6.1.3 Frame, Burst and Windowing

The input signal of the system is a vector. The vector is made into frames with a length of 160 samples and 20 percent overlap between two consecutive frames. A frame of the input signal is shown in the figure 6.1.

The frame and burst relation is shown in the figure 3.1. 20 percent overlap of the frame is used as a delay for the analysis. The same delay is maintained for the synthesis.

Each frame is windowed using hanning window. A windowed frame with well shaped boundaries is shown in the figure 6.1.

6.1.4 Pre-emphasis filter

The pre-emphasis filter boosts the harmonic components of the input signal in order to get a better estimation. The purpose of the pre-emphasis filter is shown in the figure 6.2.

![Figure 6.2: Functionality of the preemphasis filter.](image)

6.1.5 LPC Analysis Filter

In linear prediction, the speech waveform is represented by a set of parameters of an all-pole model, called the linear predictive coefficients (LPC), which are closely related to speech production transfer function. The LPC analysis essentially attempts to find an optimal fit to the envelope of the speech spectrum from a given sequence of speech samples. The analysis filter gives a residue which is used as an excitation signal for synthesis.

In this algorithm, two LPC analysis filters with order 10 of a speech spectrum is shown in the block diagram 4.5. The first LPC analysis filter in figure 4.5 residue and the second LPC analysis filter coefficients are used for synthesis. Even the performance tested with LPC coefficients of second LPC filter and residue of the same filter is used for synthesis,
but gave poor results. The linear prediction to the signal spectrum with a 11-pole and sampled at 8 k\(z\) and residual signals are shown in figure 6.3. The signal power spectrum compared with residual power spectrum and auto-correlation of the residue are shown in 6.3.

![Figure 6.3: Linear prediction analysis and Residual signal](image)

(a) Linear prediction to the signal spectrum with a 11-pole and the signal sampled at 8kz

(b) A residual signal from the 25th burst

(c) Auto correlation of the residual signal from a 35th burst

(d) Signal power spectrum vs residual power spectrum

**Figure 6.3:** Linear prediction analysis and Residual signal

### 6.1.6 Spectral Subtraction

The spectral subtraction implementation is based on proposed algorithm 5. The power spectrum of each individual frame is calculated with the *FFT* length of 256. The *FFT* length should be greater than or equal to window length for the sufficient frequency resolution.
The main objective of the power spectral subtraction is to estimate the noise spectrum, $P_n(\lambda, k)$ and subtract it from the input signal power spectrum, $P_s(\lambda, k)$ to obtain the estimate of speech power spectrum, $\hat{P}_s(\lambda, k)$. The process should be able to track different types and levels of noise spectrum is shown in figure 6.4. The implementation of algorithm is described as flow chart is shown in figure 6.9. For this different parameters are considered while implementing.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias compensation factor</td>
<td>$O_{\text{min}}$</td>
<td>$1 \leq O_{\text{min}} \leq 1.5$</td>
</tr>
<tr>
<td>Spectral floor constant</td>
<td>$k_{\text{sub f}}$</td>
<td>$0.01 \leq k_{\text{sub f}} \leq 0.05$</td>
</tr>
<tr>
<td>Smoothing constant</td>
<td>$\alpha$</td>
<td>$0.9 \leq \alpha \leq 0.95$</td>
</tr>
<tr>
<td>Amplitude spectrum smoothing</td>
<td>$\gamma$</td>
<td>$\gamma \leq 0.9$</td>
</tr>
</tbody>
</table>

Table 6.1: Description of parameter range.

The performance evaluation of the algorithm is determined by certain parameters are shown in the table 6.1. These values are chosen on the basis of [Martin, 1994].

6.1.7 Spectral Manipulation

After the subtraction, as mentioned in algorithm, the spectral subtraction, spectral manipulation is necessary as mentioned in chapter 5. The manipulation is done in three steps. First step is the envelope estimation, which estimates the envelope of the smoothed spectrum inorder to describes the LPC model. Different types of filters are designed for envelope estimation and smoothing. The envelope estimation a kind of low-pass filtering is shown in 6.4.

The first order recursive filter is used for smoothing in frequency domain. The smoothing in frequency domain is carried out to suppress the frequency components which are introduced in envelope estimation is shown in 6.4. The next approach is smoothing in time domain is to smoothing the Periodogram is shown in figure 6.4, is similar to that of frequency smoothing.

6.1.8 IFFT

From the manipulated estimate of the speech power spectrum, the IFFT should calculated the autocorrelation for the signal frame. In figure 6.5, the calculated autocorrelation for the frame index $\lambda = 95$ is shown.
6.1. OVERVIEW

(a) Averaged estimated noise spectrum
(b) Estimated noise floor of noisy speech signal $(f = 8kHz, NFFT = 256)$
(c) Envelope estimation of estimated signal power spectrum
(d) Smoothed estimated power spectrum
(e) Periodogram of estimated power spectrum
(f) Estimated speech power spectrum after spectral manipulation.

Figure 6.4: Results of Spectral subtraction and spectral manipulation
6.1. OVERVIEW

Figure 6.5: Autocorrelation of an estimated speech frame calculated from the manipulated speech power spectrum estimate.

6.1.9 Residual Manipulation

The residual signal is result of the LPC analysis filter. The residual signal would be zero, as the prediction filter would exactly match all the poles there in. This phenomenon does not occur often, so a 10th-order LPC analysis filter is used in this algorithm implementation. The residue of analysis filter of an order of 10 is shown in the figure 6.1.

Thus, the AR-modeling is not sufficient if the input signal is noise contaminated. So noise suppression is necessary in order to get better synthesized speech. The burst based execution is carried out to reduce the complexity in the process of noisy residue.

As proposed in algorithm, the autocorrelation method is considered to find the pitch period in the residue. and corresponding power spectrum is calculated. Now a first order lowpass filter is used to filter the spectrum. The filter responses is shown in the figure 6.6.

6.1.10 Synthesis filter

After finding the LPC coefficients from the autocorrelation, the estimated speech wave form is reconstructed using the results of analysis filter, the manipulated LPC coefficients and manipulated residual signal. It is called inverse system identification. The results of the of the synthesis filter is shown reffig:estimate. The spectrograms of clean speech signal, noisy spech signal and estimated speech signal are shown in the figure 6.7.

6.1.11 ANSI-C in MATLAB

The software is basically developed as a Matlab functions in ANSI-C. This allows easier access to the internal variables, and makes it possible to verify the results of the individual functions. The conversion of the software is based on simulations in Matlab. In the ANSI-
6.1. OVERVIEW

Figure 6.6: Residual manipulation results

(a) Auto correlation of the residual signal

(b) Periodogram of residual Signal

Figure 6.7: Spectrogram of speech, noisy speech and estimated speech signals

(a) Spectrogram of a noise free speech signal.

(b) Spectrogram of a noisy speech signal.
6.2 DSP Architecture Mapping

The basic architecture and functional units with functionalities of the DSP processor are explained in appendix C. The different blocks on TMS320C6713 DSK board which are used for LPC analysis implementation are shown figure 6.8. The processor and EDMA (Enhanced Direct Memory Access) can directly access the memory. EDMA uses buffers to read and process the data. The multi-channel serial port is used to communicate with external peripherals like AIC23 codec and LED’s. The address and data buses are used to communicate between the blocks [15].

6.2.1 ANSI-C on 6713DSK

The ANSI-C version, first executed on Matlab, then moved onto the 6713DSK with possible changes. The example programs were as for the reference to run on the 6713DSK. To optimize the ‘C’ code on DSK kit, one should have good knowledge of memory allocation, external peripheral port addresses and specific order of instruction execution. The code is optimized with the help of example programs from code composer studio. The Matlab code could not be worked out, so special Matlab functions were written like hanning window, framing, Levinson Durbin recursion algorithm, all pole filter, all zero filter.
The following is an example of all pole filter C code.

```c
void AllPole Filter( float *In, /* (i) In[0] to In[lengthInOut-1] contain filter input samples */
float *Coef, /* (i) filter coefficients (Coef[0] is assumed to be 1.0) */
int lengthInOut,/* (i) number of input/output samples */
int orderCoef, /* (i) number of filter coefficients */
float *Out /* (i/o) on entrance Out[-orderCoef] to Out[-1] contain the filter state, on exit Out[0] to Out[lengthInOut-1] contain filtered samples */
{
int n,k;
for(n=0;n<lengthInOut;n++)
*Out = Coef[0]*In[0];
for(k=1;k<=orderCoef;k++)
*Out += Coef[k]*In[-k];
}
Out++;
In++;
}
```

The aim is to implement the C programs on DSK. The TI C compiler (as with most compilers) differentiate C-source labels from assembly source labels by prepending an "_" to all C labels as it generates assembly code. The many leader files were added to the main program in C to run on DSK. CCS IDE (Integrated Development Environment) is made up of two parts

- Edit (and Build):
  Edit programs uses editor, to edit the programs and code generator tools to create code.
- Debug (and Load):
  Debug program is used to run the code.

Finally the program is loaded onto the DSK to test and confirm the operation of C program.
6.2. DSP ARCHITECTURE MAPPING

Compute power spectrum,

\[ P_X \]

Compute smoothed power

\[ P_{\text{smooth}} = \text{No} \]

No

\[ P_n = \min \]

Yes

\[ P_{\text{max}} \]

Compute \( P_{\text{max}} \)

Compute \( P_{\text{max}} \)

\[ S_m = X_m \cdot Q \]

if \( S_m \leq \text{Subf} \cdot P_{\text{max}} \)

Yes

Compute \( P_{\text{max}} \)

if \( P_x \leq P_{\text{max}} \)

Yes

\[ P_{\text{max}} = P_x \]

No

\[ P_n = \text{Omn} \cdot P_{\text{max}} \]

Compute SNR

\[ S_m = \text{Subf} \cdot P_{\text{max}} \]

Compute \( P_{\text{max}} \)

if \( P_x = 0 \)

Yes

\[ P_{\text{max}} = P_x \]

No

\[ P_{\text{max}} = P_x \]

Compute \( P_{\text{max}} \)

Compute \( P_{\text{max}} \)

\[ S_m = \text{Subf} \cdot P_n \]

Compute \( P_{\text{max}} \)

Compute \( P_{\text{max}} \)

\[ O_m = 7 \cdot SNR^1 / s \]

Compute \( P_{\text{max}} \)

Compute \( P_{\text{max}} \)

\[ P_{\text{max}} = P_x \]

Yes

\[ P_{\text{max}} = P_x \]

No

\[ P_{\text{max}} = P_x \]

Initialization

frame-counter = 0

\[ P_e = P_{\text{max}} \]

Load next frame

frame-counter += 1

Figure 6.9: Flow chart of spectral subtraction.
Figure 6.10: Flow chart of Basic Algorithm
6.2.2 Optimizing the Code

The C program is loaded onto the DSK to confirm the functionality as mentioned in one-day workshop student guide from TI. A new project directory is created on starting a new project. A new .cdb (Configuration database) file was created to control the range of CCS capabilities.

The created all files are added to project directory. On getting errors, break points were used for the correction of the code until the execution was successful.

After execution, the results were stored in memory were compared with that of Matlab results.

The Matlab converted C files and written C code along with Matlab code are placed in CD-Rom.
Conclusions

7.1 Results

The proposed algorithm was tested with noise free speech signals and additive and uncorrelated pink noise. After many experiments with different values, the smoothing factor of the power spectrum was set to $\alpha=0.95$. To smooth the minimum spectrum estimate, the pole was set at $\gamma=0.9$. The other parameters $komin = 1.5$, $ksubf = 0.02$, and $D = 100$ were set for the spectral subtraction. After spectral subtraction, the spectral manipulation, i.e. the envelope estimation, smoothing in both time and frequency domains and residual manipulations were done with suitable filter. The estimated noise reduced speech was observed with increased value of SNR at 0dB and 8dB. The perceptual quality was enhanced and at the same time intelligibility was maintained.

The second part of the project was about the study and the implementation onto TMS320C6713 DSK board. C code was written for LPC analysis and synthesis. Individual functions were tested onto the DSK board. Working environment of DSK was determined and the results observed were similar to that of the Matlab simulations results.

7.2 Conclusion

To conclude from the implementation results, the spectral subtraction method was used for noise reduction in speech. It is used as a good noise power estimator. More analysis was conducted using this technique and resulted in effective outcome. This technique helped in increase the signal-to-noise ratio (SNR) and perceptual quality. Spectral manipulation was also performed, but not much analysis was done, as not much change was observed in the result.

Residual manipulation played major role in order to maintain the intelligibility. This was performed with the burst based execution in order to produce better reconstruction of speech signal at synthesis filter.

It is recommended that new techniques are needed in order to suppress the noise in the residual signal along with burst based execution and also in spectral manipulation.

The C-code was written in two methods. The first method: Matlab converted C code was able to built on the board but while execution was terminated. The second method: The written C code gave good performance compared to that of Matlab generated code. The code was built and executed successfully.
A.1 Introduction

Linear Predictive Coding (LPC) is a method used for estimating speech parameters like pitch, formants, spectra, and vocal tract. LPC technique is based on linear predictive analysis. LPC removes redundancy in speech signals by forming/exploiting correlation in the speech. The analysis results in a set of AR coefficients, also known as whitening filter coefficients. The speech signal is passed through the whitening filter to obtain the residual signal, which looks like spectrally white. Finally the residual is encoded and transmitted together with the AR coefficients. This chapter describes the basic overview of linear prediction and different methods of approach to find the prediction coefficients and error minimization.

A.2 Linear Prediction

The basic principle of linear prediction is to predict a future value of a stationary discrete-time stochastic process, from a set of past or observed values [1, e.3,p.241]. The block diagram of linear prediction is shown in figure A.1.

![Figure A.1: Block diagram of Linear Prediction](image)

As shown in the figure $x[n]$ is a signal, considered as an input to a linear filter. The output of the predictor is denoted by $\hat{x}[n]$. It is the estimation of the signal $x[n]$. The error $e[n]$ is the difference between $\hat{x}[n]$ and $x[n]$. This is called linear estimation problem. In such a problem the selection of a filter is designed to minimize the error $e[n]$. Now the aim is to estimate future values from observed values of $x[n]$, then the estimation called as linear prediction. To estimate future values of $x[n]$, the estimation is just obtaining the linear prediction coefficients which are used in speech coding. A mathematical model of
linear prediction is described in-order to obtain the predictor coefficients and minimize the prediction error.

Consider a system with output $x[n]$ with some unknown input $u[n]$. A linear prediction of order $p$ is to estimate the value $x[n]$ as a linear combination of the $P$ previous observations $x[n-1], x[n-2],..., x[n-p]$. So the following equation is [6]:

$$x[n] = -\sum_{k=1}^{p} a_k x[n-k] + G \sum_{l=0}^{q} b_l u[n-l],$$  \hspace{1cm} (A.1)

where $a_k$, $b_l$, and gain $G$ are parameters of the system. Equation A.1 indicates that output $x[n]$ is a linear function of present, past inputs and past outputs. Then the signal $x[n]$ is predictable from linear combination of past outputs and inputs. The equation A.1 in frequency domain the transfer function is $H(z)$ of the system.

$$H(z) = \frac{X(z)}{U(z)} = \frac{GB(z)}{A(z)},$$  \hspace{1cm} (A.2)

where $X(z)$ and $U(z)$ are the z-transform of $x[n]$ and $u[n]$. $H(z)$ is the pole-zero model. The pole-zero model is also called as autoregressive moving average (ARMA) model.

A system with pole-zero model the estimation of the parameters is inherently non-deterministic and nonlinear. But the all-pole model can approximate the speech spectrum estimation closely and simply. So all-pole model is preferred in LPC [6].

### A.3 All Pole Model

A system only with poles, that is $b_l = 0, 1 \leq l \leq q$, and $b_0 = 1$ in equation A.1, then the system is referred to the all-pole model or auto-regressive (AR) model. Then the equation A.1 reduce to:

$$x[n] = -\sum_{i=1}^{p} a_i x[n-i] + Gu[n],$$  \hspace{1cm} (A.3)

where $G$ is the gain factor. The transfer function $H(z)$ reduce to an all-pole transfer function.

$$H(z) = \frac{X(z)}{U(z)} = \frac{G}{1 + \sum_{i=1}^{p} a_i z^{-1}} = G \frac{1}{A(z)}.$$  \hspace{1cm} (A.4)

If the current input signal is unknown, which is the common case in most of the applications, then it can only approximately predict the signal $x[n]$ from a linear weighted summation of past samples. This approximation is denoted as $\hat{x}[n]$ and is expressed as:

$$\hat{x}[n] = -\sum_{k=1}^{p} a_k x[n-k].$$  \hspace{1cm} (A.5)

The error $e[n]$ between the actual value $x[n]$ and the predicted value $\hat{x}[n]$, is:

$$e[n] = x[n] - \hat{x}[n] = x[n] + \sum_{k=1}^{p} a_k x[n-k],$$  \hspace{1cm} (A.6)
where $e[n]$ known as the residual. The prediction coefficients $a_k$ should be selected to in order to minimize the total squared error. To obtain the coefficients the method of least squares is applied. Then the equation is:

$$E_n = \sum_n e_n^2[n] = \sum_n \left[ x[n] + \sum_{k=1}^{p} a_k x[n-k] \right]^2, \quad 0 \leq n \leq N - 1. \quad (A.7)$$

The error is minimized by setting the error:

$$\frac{dE_n}{da_i} = 0, \quad 1 \leq i \leq p. \quad (A.8)$$

From the equations A.7 and A.8 gives set of equations:

$$\sum_{k=1}^{p} a_k \sum_n x[n-k]x[n-i] = -\sum_n x[n]x[n-i], \quad 1 \leq i \leq p. \quad (A.9)$$

Equation A.9 is called least squares terminology. The equation A.9 forms a set of $p$ linear equations, that can be solved to find the predictor coefficients $a_k, 1 \leq i \leq p$ and also to minimizes the $E_n$ (equation A.7). The total minimum squared error denoted by $E_p$, is obtained by substituting the equation A.7 in A.9.

$$E_p = \sum_n (x[n])^2 + \sum_{k=1}^{p} a_k \sum_n x[n]x[n-k]. \quad (A.10)$$

Now, the coefficients $a_k$ are obtained by solving equation A.10. Equation A.10 can be solved using two different approaches. One is the autocorrelation and the another one is covariance method.

### A.4 Autocorrelation Method

In this method, the assumption is that the error in equation A.7 is minimized over the infinite duration. The signal is considered a finite duration ($N$ samples) to find the autocorrelation [6]. The autocorrelations are calculated from the signal using available $N$ samples. Now, to solve the prediction coefficients $a_k$, the equations A.9 and A.10 in-terms of autocorrelation functions are:

$$\sum_{k=1}^{p} a_k R(i-k) = -R(i). \quad (A.11)$$

$$E_p = R(0) + \sum_{k=1}^{p} a_k R(k), \quad 1 \leq i \leq p. \quad (A.12)$$
The equation system can be expressed with the matrix:

\[
\begin{pmatrix}
R(0) & R(1) & R(2) & \ldots & R(p-1) \\
R(1) & R(0) & R(1) & \ldots & R(p-2) \\
R(2) & R(1) & R(0) & \ldots & R(p-3) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
R(p-1) & R(p-2) & R(p-3) & \ldots & R(0)
\end{pmatrix}
\begin{pmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_p
\end{pmatrix}
= 
\begin{pmatrix}
R(0) \\
R(1) \\
R(2) \\
\vdots \\
R(p)
\end{pmatrix}
\]

(A.13)

where \(R(i-k)\) is a symmetric autocorrelation matrix and \(R(i)\) is an autocorrelation function of signal \(x[n]\), it is also an even function. So this method is called the autocorrelation method. Here signal \(x[n]\) is multiplied by a window, the signal \(x[n] = 0\) outside the window \((0 \leq n \leq N - 1)\). The windowed signal spectrum is smoothed version of original signal and the smoothness depends on the shape and size of the window. Because of windowing the LPC coefficients results a smoothed spectrum. This method guarantees stable filters [3]. The equation A.13 is solved by taking the advantage of the symmetry and the fact that the elements across the diagonal are all identical (Toeplitz matrix), so the more efficient Levinson-Durbin algorithm can be used to compute the matrix inversion.

## A.5 Covariance Method

This is an alternative method to solve the predictor coefficients \(a_k\), \(1 \leq k \leq p\) and also to minimize the error \(E_p\). This method use much smaller samples compared to autocorrelation method, to minimize the error \(E_p\) over a finite duration \(1 \leq n \leq N - 1\). The equations A.9 and A.10 in terms of covariance functions are [6]:

\[
\sum_{k=1}^{p} a_k \phi_{ki} = -\phi_{i0}, \quad 1 \leq i \leq p
\]

(A.14)

\[
E_p = \phi_{00} + \sum_{k=1}^{p} a_k \phi_{0k},
\]

(A.15)

where

\[
\phi_{ik} = \sum_{n=0}^{N-1} x[n-i]x[n-k],
\]

(A.16)

where \(\phi_{ki}\) is a symmetric covariance matrix. By solving the above covariance normal equation set in matrix form as:

\[
\begin{pmatrix}
\phi_{11} & \phi_{12} & \phi_{13} & \ldots & \phi_{1p} \\
\phi_{21} & \phi_{22} & \phi_{23} & \ldots & \phi_{2p} \\
\phi_{31} & \phi_{32} & \phi_{33} & \ldots & \phi_{3p} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\phi_{p1} & \phi_{p2} & \phi_{p3} & \ldots & \phi_{pp}
\end{pmatrix}
\begin{pmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_p
\end{pmatrix}
= 
\begin{pmatrix}
\phi_{10} \\
\phi_{20} \\
\phi_{30} \\
\vdots \\
\phi_{p0}
\end{pmatrix}
\]

(A.17)
The matrix equation A.17 is symmetric but not Toeplitz matrix (the elements across the diagonal are different). This method cannot guarantee stable filters. There is no efficient algorithm to make the matrix inversion for covariance matrix equation.

### A.6 Levinson Durbin Algorithm

The Levinson Durbin (L-D) algorithm describes a direct method for computing the predictor coefficients \( a_k \) and mean square error \( E_n \) for a order \( p \) by solving the augmented Wiener-Hopf equations. The method is recursive in nature and by using Toeplitz structure of the correlation matrix (equation A.13) of a tap input of the filter. This known as Levinson Durbin algorithm [1, e.3,p.198].

The Toeplitz matrix equation A.13 is solved by using L-D algorithm, which is efficient method to exist to invert such matrices. To determine the values of the linear prediction parameters at iteration \( i \) by \( a^i_k \) and the error \( E^i_n \) for \( 1 \leq i \leq p \). The recursive procedure is:

Initially \( E_0 = R(0) \) and \( a_0 = 0. \)

\[
\begin{align*}
    k_i &= \frac{R(i) + \sum_{j=1}^{i-1} a^{(i-1)}_j R(r-j)}{E^{i-1}}, \quad (A.18) \\
    a^i_j &= k_i, \quad (A.19) \\
    a^i_j &= a^{(i-1)}_j + k_i a^{(i-1)}_{i-j}, \quad 1 \leq j \leq i-1 \quad (A.20) \\
    E_i &= (1 - K^2_i) E_{i-1}. \quad (A.21)
\end{align*}
\]

The parameters \( k_i \) are known as the reflection coefficients.

The equations A.19 and A.21 are solved recursively in each iteration. In each cycle the iteration \( i \), the coefficients \( a^i_k \) for \( 1 \leq i \leq p \) gives the optimal \( p \)th order linear predictor and error \( E_n \) is reduced by a factor of \( (1 - k^2_i) \) [6]. The filter with L-D algorithm produces is in minimum phase and stable. L-D algorithm is computationally efficient.
Speech and Noise

This chapter is introduced to describes the properties of speech and noise. In the speech section it describes about formants and pitch period. In the noise section, different noise types and its characteristics are described.

B.1 Speech

Speech is used to communicate information from a speaker to a listener. The human speech production begins with an idea or thought that the speaker wants to convey to a listener. The speaker produces an acoustic sound pressure wave by a series of neurological process and muscular movements that is received by a listener auditory system.

Speech signals are non-stationary and at best they can be considered as quasi-stationary, over the short segments, typically 5-30 msec. The statistical and the spectral properties of speech are thus defined over a short segment. Speech can be generally classified as voiced, unvoiced or mixed. Voiced speech is quasi-periodic in the time domain and harmonically structured in frequency domain while unvoiced speech is random like and broadband. In addition, the energy of voiced segments is generally higher than the energy of unvoiced segments.

The short time spectrum of voiced speech is characterized by its fine and formant structure. The fine harmonic structure is a consequence of quasi-periodicity of speech and may be due to the vibrating vocal cords. The frequency of periodic pulses is referred to as the fundamental frequency or pitch. The formant structure (Spectral envelope) is due to the interaction of the source and vocal tract. The spectral envelope is characterized by a set of peaks which are called formants. The formants are the resonant modes of vocal tract. These formants are quite important both in speech synthesis and perception.

B.2 Noise

The noise can be defined as complex sound waves that are aperiodic, that is, the sound waves with irregular vibrations and no definite pitch. In other words, noise is defined as a unwanted signal that interferes with the detection of or quality of another signal [16].

The noise is classified into different colors according to their spectral properties. White noise power density is constant over a finite frequency range. The next most commonly used color is pink noise. Its frequency spectrum is not flat, but has equal power in bands that are proportionally wide. Pink noise is perceptually white. That is, the human auditory
B.2. NOISE

system perceives approximately equal magnitude on all frequencies. The power density decreases by -3 dB per octave (density proportional to \(1/f\)). Brown noise is similar to pink noise, but with a power density decrease of -6 dB per octave with increasing frequency (density proportional to \(1/f^2\)) over a frequency range which does not include DC. There are also many "less official" colors of noise such as red, orange, green and black.
This chapter includes DSP processor architecture, development tools and issues related to real-time processing considered in this project. The TM320C6713 DSP starter kit with high precision applications based on TI’s TM320C6000 floating point DSP generation. The TMS320 DSP family offers the most extensive selection of DSPs available, with a balance of general-purpose and application-specific processors to suit user needs. There are distinct Instruction Set Architectures that are completely code-compatible within platforms.

C.1 TMS320C6000 DSP platform

The C6000 DSP platform offers fast DSPs running at clock speeds up to 1 GHz. The platform consists of the TMS320C64x and TMS320C62x fixed-point generations as well as the TMS320C67x floating-point generation. The C6000 DSP platform’s performance ranges from 1200 to 8000 MIPS (Million Instructions per Second) for fixed-point and 600 to 1350 MFLOPS (Mega Floating Point Instructions per Second) for floating point [15].

Basic C6000 CPU Architecture:

- Functional Units
- Register File
- Memory and Peripheral

C.2 Functional Units

It contains eight independent functional units as shown in the figure C.1. All eight functional units can receive their own 32 bit instruction on every cycle, i.e. it can execute eight instructions in parallel.

- **.D unit (.D1,.D2)** A 32-bit loads and stores, add, subtract, linear and circular address calculations.
- **.M unit (.M1,.M2)** It performs 16x16-bit integer or 32x32-bit floating point multiply operations in the hardware.
- **.L unit (.L1,.L2)** A 32/40-bit arithmetic and compare, 32 bit logic operations. It performs conversion operations.
C.3 Register File

The variables operated upon by the CPU are stored in register file. There are two register files. Register file "A" (A0-A15/31) and Register file "B" (B0-B15/31) of 16 or 32 registers each, depending upon which C6000 CPU is using.

C.4 TMS320C6713 DSP

The TMS320C6713 is the floating-point DSP generation in the TMS320C6000 DSP platform. The C6713 DSP also features a two-level cache and VLIW (very-long instruction word) architecture. The C6713 DSP operating at 225 MHz, delivers up to 1350 million floating-point operations per second (MFLOPS), 1800 million instructions per second (MIPS), and with dual fixed-/floating-point multipliers up to 450 million multiply-accumulate operations per second (MMACS). The C6713 DSP has sufficient bandwidth to support all 16 serial data pins transmitting a 192 kHz stereo signal.
C.5 C6713 Memory and Peripherals

The c6x family of DSPs has a single large 32-bit address space. The address space is split between on-chip memory, on-chip peripheral registers and external memory. All memory is byte addressable and program code and data can be mixed freely. The C6713 also has 4kB program and data caches to improve performance when accessing external code and data [15]. The figure C.2 memory map for the C6713 DSK showing how the address space is used. The internal memory starts at the beginning of the address space with most of the either reserved or used for peripheral registers. The EMFI starts at address 0x80000000 and spans the next 1 GB of the address space. It is divided into 4 equally sized regions each with a dedicated chip enable signal (CE0-CE3). The on-board memory, programmable CPLD (complex programmable logic device) registers and add-on daughter cards are all connected through EMIF (external memory interfaces).

C.6 TMS320C6713 DSP Starter Kit (DSK)

The TMS320C6713 DSP Starter Kit (DSK, developed for high precision applications based on TMS320C6000 floating point DSP generators. Like audio, medical imaging, test and instrumentation. The C6713 DSK includes 8MB of on-board SDRAM and an emulation header and 12C interfaces.

The DSK includes the Fast Run Time Support libraries and utilities such as Flash burn to program flash, Update Advisor to download tools, utilities and software and a power on self test and diagnostic utility to ensure the DSK is operating correctly. The hardware
features of the TMS320C6713 DSK board include:

- C6713 DSP Development Board with 512K Flash and 8MB SDRAM
- High-quality 24-bit stereo codec
- Four 3.5mm audio jacks for microphone, line in, speaker and line out
- Expansion port connector for plug-in modules

C.7 Tools and Software

For DSP product development, the TMS320 DSP family is supported by user eXpress-DSP Real-Time Software Technology that includes Code Composer Studio integrated development environment, DSP/BIOS Real-time software kernel, TMS320 DSP Algorithm Standard.

C.8 Algorithm Standard

The TMS320 DSP Algorithm Standard is a single, standard set of coding conventions and application programming interfaces (APIs) for algorithm creators. The standard includes algorithm programming rules that enable interoperability between different types of algorithms.

C.9 Terminology

Below are some of the more common terms that encounter when discussing the TMS320 DSP Algorithm Standard.

- **Algorithm**: A module of code that consumes a data stream, processes it, and outputs a resultant stream. Examples include vocoders, modems, audio compression, video decompression, etc.
- **Reference Framework**: The "glue" code that holds together the drivers, the algorithms, resource managers, and DSP kernel. Reference Frameworks start out as application-agnostic. Upon the addition of application-specific algorithms, the Framework takes on an application-specific nature.
- **DSP Kernel**: A low-level software layer that provides hardware abstraction and manages low level physical resources. It provides threading; interrupt support, pipes, signals, and several other functions. In addition, DSP/BIOS (Basic Input Output System) offer data logging and statistical accumulation that enable real-time analysis of the system.
• **Application**: It depends on the use of some or all of the other components. If a user writes all the code from scratch including a kernel, algorithms, and a framework, then the entire software system may be described as the application. However, in an environment where DSP/BIOS, a reference framework, and COTS (Commercial off-the-shelf) algorithms have been deployed, the application programmer uses the APIs (Application Program Interface) for the controlling framework.


