TEL AVIV UNIVERSITY אוניברסיטת תל-אביב
FACULTY OF ENGINEERING הפאקולטה להנדסה

Department of Electrical Engineering - Systems

ALGORITHMS FOR SINGLE MICROPHONE SPEECH ENHANCEMENT

Thesis submitted towards the degree of

"Master of Science in Electrical Engineering"

in Tel Aviv - University

by

SHARON GANNOT

April, 1995

TEL AVIV UNIVERSITY אוניברסיטת תל-אביב

FACULTY OF ENGINEERING הפאקולטה להנדסה

Department of Electrical Engineering - Systems

ALGORITHMS FOR SINGLE MICROPHONE SPEECH ENHANCEMENT

Thesis submitted towards the degree of

“Master of Science in Electrical Engineering”

in Tel Aviv - University

by

SHARON GANNOT

This research work was carried out at Tel-Aviv

University, in the department of electrical

engineering - systems, under the supervision of

Prof. EHUD WEINSTEIN

April, 1995

Acknowledgments

I would like to thank a lot my supervisor Prof. Ehud Weinstein who inspired me with his ideas and for his devoted guidance.
I appreciate very much the warm hospitality and the great help in achieving the ASR results given by Distinguished Professor Alan V. Oppenheim and his group in MIT, especially Shawn Verbout.
Very special thanks to Dr. Gershon Chazan for his fruitful ideas and his permanent interest in my work.
I am grateful to Thalia Koren who helped me so much in implementing the GUI, the MEX files, and more.
Thanks to Yona Eyal for her computer assistance. Thanks to Nadav Shulman for his LaTeX help, especially for the wonderful big hats.
Thanks to Irit Shaaya whom I have learned with most of the courses towards this degree. Thanks to Liron Frenkel for our fruitful discussion.
Warm thanks to Ofra Golani for her important remarks throughout the entire work.
Thanks to my brother Dr. Israel Gannot for his helpful advice.
Last but not least I express my warmest gratitude to my parents Yosef and Frieda Gannot who worked so hard all those years for giving me support and for planting the eager for knowledge.

Abstract

In this work we address the problem of single microphone speech enhancement.

We extend previous results by giving an extended parametric model for both the speech and noise signals. The speech is modeled by an AR process excited by a mixed innovation of both white noise and pitch sequence. The Noise is modeled as a different order AR process excited only by white noise. By this modeling we translate the speech enhancement problem into the Maximum-Likelihood estimation problem. Then, by an appropriate use the EM procedure, we achieve both parameter estimation and signal enhancement. The resulting algorithm has an intuitive structure. The signal is divided into segments, short enough for the stationary assumption to hold. In each segment the algorithm iterates between solving a decoupled set of equations for the speech and noise parameters, and an application of the Kalman Filter, or Fixed-Lag-Smoother.

In the presence of additive Gaussian noise, we suggest the use of Higher-Order-Statistics (HOS) techniques for robust estimation of the AR parameters. This use improves significantly the convergence behavior of the algorithm.

A Sequential/Adaptive solution (in which no segmentation is applied) is also derived, giving a computationally more efficient algorithm, but at the expense of performance degradation in the low SNR range.

Several objective and subjective tests are conducted for evaluating the proposed algorithms performance. The tests performed include informal speech quality ratings, distortion measures, Intelligibility scores achieved by human listeners, and recognition accuracy of an ASR system in the presence of actual noise. The proposed algorithms are proven to work well in all categories.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General

The problem of enhancing speech degraded by additive noise has received considerable attention in the literature since the mid-1970. Speech enhancement is desirable in wide variety of contexts. Consequently, a variety of approaches have been proposed and investigated (e.g. [21], [23], [16]).

The problem of Speech Enhancement can be divided to several problems and topics. Speech can be degraded by additive noise or by competitive speaker or by some distortion, such as microphone or telephone channel. We can process the speech with several microphones (two or more) or only by one. Another issue concerns the purpose of the enhancement. We may process the speech for a human listener in order to improve its quality (e.g. in noisy environments such as offices, streets, and motor vehicles), or to improve its intelligibility in harsh conditions (such as airports). Transcription of recorded tapes degraded by additive noise is also of interest. We may use it as a preprocessing mechanism for speech compression algorithms or as a front-end to Automatic Speech Recognition (ASR) systems. The type of algorithm depends on the application and problem issued.

We address the problem when only one microphone of degraded speech is available for processing. We also assume that the speech is degraded by additive noise. We are interested in speech quality enhancement, as well as in intelligibility improvement, and in ASR performance improvement.

Throughout the remainder of this chapter we will try to give a brief summary of existing algorithms.

1.2 Existing Methods

We will summarize several existing methods for single microphone speech enhancement. We will not describe all the existing algorithms, of course, but only methods that are related to our development, and algorithms that were implemented for comparison purposes. In each of the algorithms we will cite the advantages and drawbacks as stated by the authors. When ever possible we will try to give our opinion. Two of the algorithms were also implemented in MATLAB, for the purpose of comparison. The addressed methods can be roughly divided into the following areas:

Frequency domain methods

Time domain methods

Periodicity

Underlying model for speech production

1.2.1 Frequency domain methods

Several methods are developed for speech enhancement in which the incoming speech is divided into segments. Each segment is spectrally decomposed into a set of magnitudes and a set of phases. The noise reduction is achieved by appropriate adjustment of the set of spectral magnitudes. A waveform reconstruction process then combines the adjusted magnitudes and the unprocessed phases. All methods employ a speech activity detector, which detects when speech is present. At non-speech segments the background noise spectrum is estimated. The various methods - suggested by Weiss *et al.* [24], Boll [2], and others - differ primarily in their approaches to spectral magnitude adjustment. Boll's version, in which the spectral magnitude adjustment is achieved by subtracting the estimated noise spectrum from the corrupted speech spectrum, is the most popular version.

The method was tested by the author and by others [20]. The results include Spectrogram plots comparison, DRT tests and coarse measure of quality. The procedure was also repeated for a speech that was pre-processed by the algorithm prior to applying an LPC vocoder. The results indicates that the algorithm alone does not decrease intelligibility and does increase quality. When applied before the LPC bandwidth compressor, an intelligibility increase is noticed.

Nevertheless, there are significant drawbacks to the suggested algorithm. The main drawback is the residual noise characteristics. After the subtraction operation, especially in the absence of speech activity, the remaining residual signal will exhibit itself in the spectrum as randomly spaced narrow bands of magnitude spikes. In the time domain those spikes manifest themselves as a sum of tones. These audible effects can be reduced by thresholding operation, taking advantage of the frame-to-frame randomness of the noise. Another important drawback is the need of a very good speech-activity detector.

The Spectral Subtraction method (especially, Boll's version) has gained a lot of industrial interest. For that reason we implemented it for comparison purposes. The described residual tone phenomena was detected of course, and it is quite annoying. The algorithm performance degrades rapidly bellow SNR level of +5 dB.

A more detailed description of the algorithms may be found in Appendix C.

1.2.2 Time domain methods

We address here the algorithm suggested by Bernard Widrow *et al.* [34]. The algorithm is directed to minimize the square error between the estimated signal and the clean signal. This is a general algorithm, and it does not use the detailed structure of the speech production model. In spite of this limitation Widrow's algorithm is widely used for speech enhancement. Its main applications are in the two or more microphones case, or in one microphone case for speech corrupted by periodic noise. For that reason we implemented Widrow's algorithm for comparison purposes. Here we will give only a brief overview of the method. A more details description may be found in Appendix D.

The usual method of estimating a signal corrupted by additive noise is to pass it through a filter that tends to suppress the noise while leaving the signal relatively unchanged. The design of such filters is the domain of optimal filtering, which originated from the pioneering work of Wiener and was extended by the work of Kalman, Bucy and others. Filters used for the above purpose can be fixed or adaptive. The design of fixed filters is based on prior knowledge of both the signal and noise statistical characteristics. Adaptive filters, on the other hand, have the ability to adjust their own parameters automatically, and their design requires little or no *a priori* knowledge of the signal and noise characteristics.

Noise cancelling is a variation of optimal filtering. It makes use of an auxiliary or reference input derived from one or more sensors located at points in the noise field where the signal is weak or undetectable. In the one-microphone case, where no auxiliary sensor exist, we derive the reference input by applying a delay to the primary input.

The primary microphone detects both speech (desired) and noise (disturbing) signals. The secondary (reference) microphone detects a signal that is in correlation with one of the primary microphone components but not with the other. An adaptive filter is applied to the reference microphone. The output of the filter is subtracted from the primary microphone to yield an estimation of the uncorrelated signal. The output of the adaptive filter is an estimation of the correlated signal. The filter coefficients are adaptively changing in time in order to minimize the estimation error (which is the difference between the filtered reference signal and the disturbing signal).

The desired correlations are achieved via a modification to the DELAY value. We chose a value which is longer then the correlation length of one of the signals, but shorter then the other. Thus, if the noise is white, a short delay will cause the noise correlation to diminish while the speech correlation still exist. On the other hand, if we have a periodic corruption, a very long delay will do the job. This use of the DELAY implies that only signals with different correlation length can be separated.

We implemented this algorithm in our Lab. The algorithm works very well with a periodic noise, eliminating its component almost completely, without degrading the speech. With white noise (or worse, with a "colored noise"), the algorithm works only with sufficiently high SNR values ($5-10$ dB) and having a very annoying “barrel effect”. The speech is distorted and sounding like coming from inside a barrel.

1.2.3 Periodicity

waveforms of voiced sounds are periodic with a period that corresponds to the fundamental frequency, the *pitch*. Most of the speech energy is due to the this harmonic structure. So, a good practice may be to apply a comb filter, which passes only the harmonics, and attenuate the inter-harmonics frequency bands, which are due mostly to the noise. This idea was suggested by Shields. An improvement to this approach was suggested by Fraizer [29] and evaluated by Lim, Oppenheim and Braida [11]. The comb filter is constructed by several taps spaced pitch periods apart. A larger weight is given to the closest pitch impulses. Thus, the short-term periodicity of the speech signal is exploited in order to filter out the non-periodic components of the corrupted signal. Three, seven, or thirteen periods are checked. Special care is taken for abrupt changes in the pitch period (which is a common phenomena in speech) and to transition from voiced to unvoiced portion of speech. The exact pitch period is used (derived from clean speech). The SNR value considered are in the range from above -5 dB. Intelligibility tests are done for this algorithm. The intelligibility degrades with decreasing SNR value (as expected of course) and with applying a longer filter. The use of only three pitch periods is proven to be the best choice. It does not degrades the intelligibility significantly. On the other hand, Increasing the number of the pitch periods used increases the SNR improvement from 3.5 dB to 10 dB. It is important to emphasize, that trying to evaluate the pitch information from the corrupted speech, will cause a severe degradation in performance.

We note here that, in our opinion, the pitch information should not be used alone. There is an important information in the low level portion of the speech (unvoiced) too. We suggest to try and combine the pitch information, into more comprehensive model of speech.

1.2.4 Underlying model for speech production

We present here a family of algorithms that are based strongly on the underlying model of the speech production. All the algorithms that we will present here are based on the LPC model of speech, which is explained in chapter 2.

Lim and Oppenheim [12, 13] suggested an iterative scheme, which is based on the LPC model, for enhancing corrupted speech.

The algorithm tries to solve the Maximum A-posteriori (MAP) estimate of the speech parameters. Under the assumption of Gaussian excitation the MAP estimator of the LPC parameters given the observed clean data of the entire segment coincident with the Yule-Walker equations ("Covariance Method"). When only corrupted observations are given, the equations for solving the MAP estimator of become non-linear and in general difficult to solve.

The authors suggest to solve for the parameters by applying iterative procedure which requires only a solution of a set of linear equations. The iterative algorithm, referred to as linearized MAP (LMAP), begins with an initial estimate of the LPC parameters and then enhances the speech by an appropriate application of an optimal filter. A new estimate of the LPC parameters is obtained by the "correlation" method. The procedure iterates back and forth between enhancing the speech data given the parameters and the LPC parameters given the enhanced speech signal. Furthermore, under a stationary assumption (an infinite segment length) the enhancement operation can be performed via the non-casual Wiener filter.

The LPC parameters are calculated by inversion of the correlation matrix (assuming that the noise spectrum is known). The terms of this matrix can be calculated by a multiplication of the enhanced speech vectors, which is the LMAP version, or by a direct calculation, which constitutes the revised LMAP (RLMAP) algorithm.

The authors presents preliminary results about the enhancement achieved. It is worth noting that the LMAP algorithm converges more rapidly but after more than 2-3 iterations the formant bandwidth become very narrow, causing an unnatural sounding. The RLMAP converges to a point where the poles' bandwidth is comparable to those of the clean speech , but only after 10 iterations.

Several researchers tried to overcome the convergence drawbacks. Hansen *et. al.* [10] suggested to imply a set of vocal tract spectral constrains. Their modification is using inter-frame (across time) and intra-frame (across iterations) constrains to ensure speech-like characteristics of the enhanced speech. It has been shown that applying constrains on the the radial and angular movements of the poles across time and iterations can give substantial improvement in objective speech quality and in informal listening tests.An efficient technique for applying the constrains, based on the spectral pair (LSP) representation, is suggested. The LSP transformation results from modifying the LPC polynomial of the All-Pole speech model. The LSP coefficients introduce an easy way to find the vocal tract poles' location and bandwidth. The constrains include smoothing the location of the poles across time and iterations, preventing too fast movement, and down limiting the bandwidth of the poles to prevent an unnatural sounding caused by too narrow peaks. The algorithm that performs Lim and Oppenheim iterations was implemented including the discussed constrains. The results indicates improved Itakura-Saito distances from the clean speech (which is with good correlation with subjective tests) than achieved by the original Lim and Oppenheim algorithm and by Boll's algorithm. The best results over a wide range of SNR values (-5 dB to 10 dB) is obtained by applying both across-time and across-iterations constrains. This improvement is consistent over a large range of sound type (e.g. *Vowel, Nasal, Fricative, Glide, Liquid,* etc.). Furthermore, a consistent terminating point (number of iterations) over a wide range of SNR levels and sound types is achieved in the constrained algorithm. The resulted spectral shape did not suffer from the narrowed bandwidth of the formants occurred in the unconstrained approach. The algorithm was also checked as a preprocessor to an Isolated-Word speech recognition system. It proves to make a considerable improvements in SNR levels from 10 dB to 30 dB. It should be emphasized that the noise spectrum is calculated by using speech-free segments and it is not updated very frequently. This technique also suffers from a great computation complexity.

Masgrau *et. el.* suggested another modification to Lim and Oppenheim's algorithm [5, 6]. The modification takes advantage the blindness to uncorrelated

Gaussian noise possessed by the Higher-Order Cumulants. In Lim and Oppenheim algorithm the AR coefficients are estimated via the correlation function. This estimation suffers severely due to bias caused by the noise. The use of Cumulants decouples the speech from the noise, and exchange the bias in the estimation by a greater variance. Furthermore, the phenomena of narrow formants is more observable when using HOS than with second order statistics. In order to accommodate with this problem, the authors are using Higher Order Statistics (HOS) only in the first iteration and usual second order LPC analysis in the next iterations (resulting an hybrid algorithm). By starting with HOS we can use the benefits of robustness to noise and faster converges rate, and by continuing with second order statistics we have the benefit of statistically more stable algorithm.

Simulations indicates an improvement in the overall SNR and other distortion measures (like Itakure-Saito) compared to the conventional algorithm. This technique can help us also as a pre-processor for Speech Identification systems. The resulting AR parameters are more robust to noise.

A time-domain approach to signal enhancement is suggested by Weinstein, Oppenheim and Feder [7]. The procedure is based on the iterative Estimate-Maximize (EM) algorithm for maximum likelihood estimation. On each iteration, in the M-step of the algorithm, parameter values are estimated based on the signal estimates obtained in the E-step of the prior iteration. The E-step is then applied using these parameter estimates to obtain a refined estimate of the signal. The E-step is implemented in the time domain using a Kalman smoother. This approach enables the algorithm to avoid many of the computational difficulties with the prior Lim and Oppenheim frequency domain formulation. Furthermore, the time domain formulation leads naturally to time adaptive algorithm by replacing the Kalman Smoother with a Kalman filter, and in place of successive iterations on each data block, the algorithm proceeds sequentially through the data with exponential weighting applied to permit the algorithm to adapt to changes in the structure of the data. The resulted algorithm can be viewed as a time domain counterpart of the Lim and Oppenheim algorithm.

1.3 Proposed Algorithms Overview

This thesis is an extension to Weinstein, Oppenheim and Feder's algorithm. We develop two algorithms that exploit a more complicated model of the speech and noise. The periodicity of the speech is exploited by incorporation of the pitch innovation sequence. We also extend the noise model. The noise can be "colored" with arbitrary spectrum, instead of white, as in the previous algorithm. Attention is taken to the noise spectrum estimation, without using speech activity detector. The convergence behavior of the algorithm and an enhancement in its performance is achieved by using HOS for the LPC parameter estimation. Two algorithms are developed from the EM procedure for ML parameter estimation. The first one divides the signal into segments. In each segment iterations are made between both noise and speech parameter estimation (M-step), and signals enhancement (E-step). The M-step is implemented via two decoupled set of equations (for the speech and noise parameters). The equations are an extension of the Yule-Walker equations. The E-step is constructed by applying Kalman filter (or fixed -lag-smoother). The second algorithm has a sequential/adaptive structure achieved by replacing the iteration index by the time index. The resulting algorithm advance in time by applying Kalman filter and by sequentially estimating the signals parameters. Both algorithm have an intuitive form. The parameters of the signal are estimated using its current estimate, and the signal is enhanced by using the estimated parameters. This form implies a possible application of the algorithm for the competitive speaker separation problem.

The algorithms are checked intensively by applying subjective and objective tests, such as quality measure, intelligibility tests and ASR performance evaluation.

The work is arranged as follows. In chapter 2 the models of the speech and noise signals are presented and the enhancement problem is introduced. Chapters 3 and 4 includes both the **Iterative-Batch** and **Sequential** solutions for the problem. Chapter 5 is devoted to a preview of methods for evaluating the performance of speech systems. The performance of both algorithms is evaluated via Subjective and Objective tests in chapter 6. Conclusions and topics for further research are discussed in chapter 7.

Chapter 2

The Signal Model

The basic problem of interest is illustrated in Figure 2.1, where $s(t)$ represents the speech signal, $v(t)$ represents the noise, $z(t)$ represents the observed distorted signal, and $\hat{s}(t)$ represents the enhanced, or estimated speech signal. As with all the approaches mentioned in the introduction, the enhancement algorithm depends on the specific assumptions being made on the speech and the noise models. This will be the focus of this chapter.

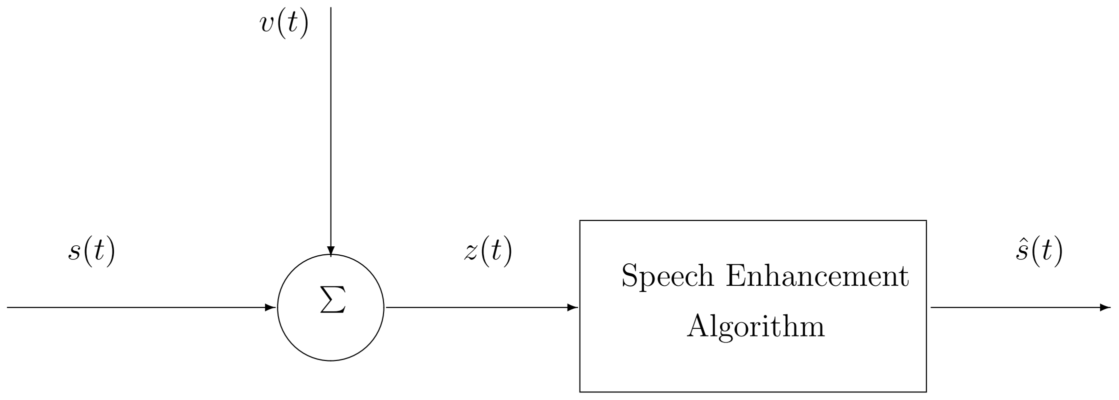


Figure 2.1: Problem Formulation

2.1 The Speech Production Model

Our description of the speech production model is based partly on Parson's book [30].

The speech mathematical model is based on physiological aspects concerning the speech production. The operation of speech production is usually divided into two functions, **excitation** and **modulation**. Excitation takes place mostly at the glottis but also at some other points; modulation is done by various organs of the vocal tract.

Excitation: Excitation is done in several ways, comprising phonation, whispering, frication, compression and vibration. **Phonation** is the oscillation of the vocal cords. When air is forced through the vocal cords, they vibrate. The opening and closing of the cords breaks the air stream up into pulses. The shape and duty cycle of these pulses depends strongly on the circumstances. The repetition rate of the pulses is termed *pitch.* At low level of air pressure oscillation may become irregular (repetition rate dropped by half, or pulses coming in pairs). Anyhow, speech sounds accompanied by phonation are called *voiced.* Other types of excitation are called *unvoiced.* These includes **Whispering**, where the vocal cords are almost closed, generating turbulence of air. Other types of unvoiced excitation are perceived as interruption of the air flow (and therefore can be also considered as modulation) forming *consonants* and syllable boundaries. If the vocal tract is constricted at any other point, the air flow past the constriction is turbulent. This sound is called **Fricative**. **Compression** occurs if the vocal tract is completely shut off at any point. Pressure builds up, and upon release, a small explosion will occur. This combination of silence followed by a short noise is called **plosive** in abrupt release and **affricate** in graduate release. **Vibration** occurs when air is forced through a closure other than the vocal cords, especially, the tongue. All *unvoiced* sounds have a wide-band noise-like characteristics.

Modulation: Modulation is the action of imposing information on the glottal output. Physiologically, the sound is modulated by moving the speech organs (mainly the tongue) in order to change the quality of the voice and to interpose additional sounds on the voice. Acoustically, the principal means of modulation is the operation of *filtering.* The glottal waveform is very rich in harmonics, and the vocal tract, like any acoustical tube, has natural frequencies, which are a function of its shape. These natural resonance are called *formants*. Those frequencies provide crucially important information about the vowels and some of the consonants. Additional types of modulations are various interruption and added noise bursts.

To summarize, and to make the necessary connection between physiologically aspects and our mathematical models, we will make some simple assumptions and modifications.

We can model the excitation as some mixture of train pulse and white noise. At each time period the excitation can be voiced or unvoiced or generally "more" voiced or "more" unvoiced.

The modulation, should be a filter. A reasonable choice might be an all-pole filter. There are several justification to this choice. First, we mentioned earlier that the speech spectrum (especially for voiced sounds), is characterized by resonance frequencies (formants), which can be modeled as poles. This all-pole model property is referred to as the "Spectral Matching" property. Second, a simplified "mechanical" model for the vocal tract can be of several "pipes" of different diameters connected together. Analysis of this model can bring us to the all-pole model. Third, and the most important, is that this model, although very simple, proves itself as a working model in several application, such as *speech compression* and speech recognition. The LPC parameters - which are, actually, the AR coefficients - and some related parameters form the basis of Automatic Speech Recognition systems. Those LPC parameters are changing in time to point out the non-stationarity of speech. Nevertheless, it is a usual practice to assume quasi-stationarity: the excitation and the filter parameters are assumed to be fixed during a short time period (about 20 mSec).

We shall conclude this summary with the a schematic figure 2.2. The excitation in this figure is modeled as a (weighted) sum of white noise and pulse train. This weighted sum can be replaced by a switch, which is controlled by a voiced/unvoiced decision. We prefer the more complicated version: the weighted sum. There are several unknown parameters in this model: the pole location (filter coefficients), the white noise level, the pitch period, and the glottal shape and gain.

This schematic description can be converted into a mathematical model which

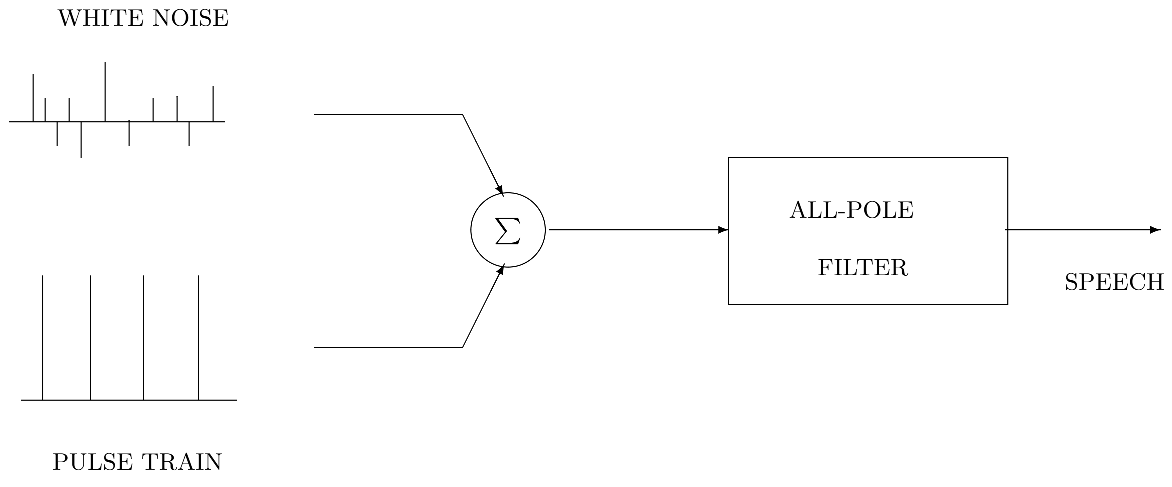


Figure 2.2: Schematic speech production model

include several parameters to be estimated later. Thus, the speech signal model can be written as in equation 2.1.

$$s(t) = -\sum_{k=1}^{p} \alpha_k s(t-k) + \sqrt{g_s} u(t) + A_s g(t). \tag{2.1}$$

Where, $\alpha_1, \alpha_2, \ldots, \alpha_p$ are the p AR coefficients, representing the vocal tract. This filter is derived by a mixed excitation (innovation sequence). The first component is a white, preferable non-Gaussian, zero-mean process, termed $u(t)$, with power $E\{u^2(t)\} = 1$.
The second component is an impulse train series $g(t) = \sum_{k=-\infty}^{\infty} d(t - kT_s - \phi_s T_s)$, where, T_s is the pitch period, ϕ_s is the phase of the first pulse relative to the beginning of the segment, and $d(t)$ is the glottal pulse shape. We will also denote g_s as the white noise power gain and A_s as the pitch series gain. The quasi-stationary assumption we have made, implies that the all of the parameters are constant through out a predefined segment. All the described parameters are shown in Fig 2.3.

2.2 The Noise Model

The noise model depends on its source. As there is a variety of efficient methods for suppressing the effect of narrow-band noise sources, we shall concentrate on wide-

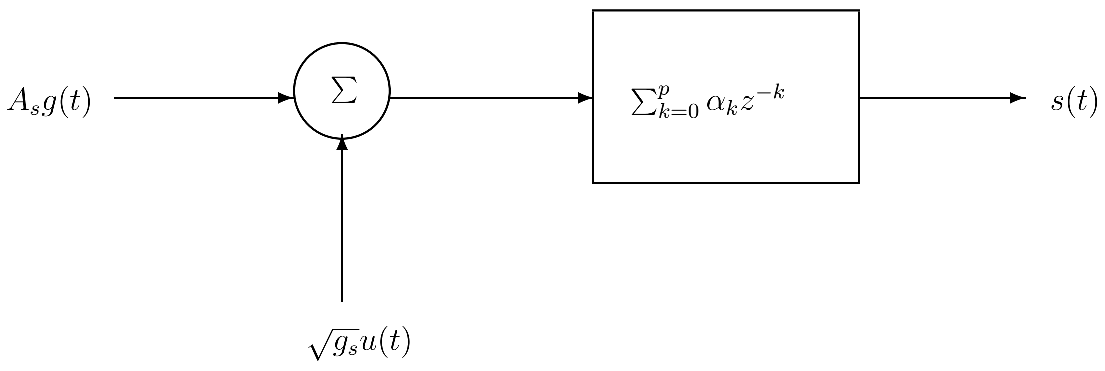


Figure 2.3: Mathematical model for Speech production

band noise sources. Furthermore, only wide-band noise sources that can be modeled as an autoregressive (AR) process will be discussed. This model, although limited, is broad enough for our purposes.

Thus, the noise signal model can be written as in equation 2.2.

$$v(t) = -\sum_{k=1}^{q} \beta_k v(t-k) + \sqrt{g_v} w(t) \tag{2.2}$$

Where $\beta_1, \beta_2, \ldots, \beta_q$ are the q AR coefficients. The noise innovation sequence, $w(t)$, is a white, zero mean, preferable Gaussian process with $E\{w^2(t)\} = 1$. g_v is the innovation power gain. The quasi-stationary assumption holds here also (we usually assume that the noise is changing slower than the speech). For example, The special case of white noise is treated as a private case of this model by choosing the filter order to be $q = 0$. All the described parameters are summarized in Fig 2.4.

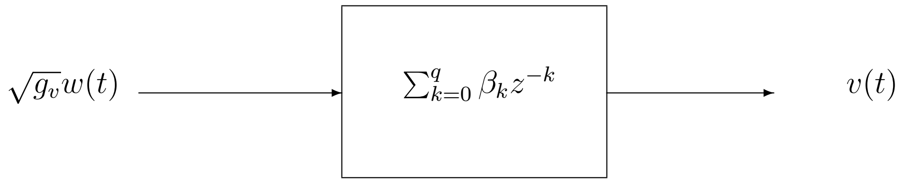


Figure 2.4: Mathematical model for Noise process

As a final remark we note that introducing a pitch series as another innovation of the noise signal can convert our problem into a speaker separation problem. We did not address this problem in this work.

Chapter 3

Algorithm Development

3.1 ML estimation via the EM procedure

Let $\mathbf{z} = \{z(t) : t = 1, 2, \ldots, N\}$ denote a vector of corrupted speech samples (observed data) possessing the p.d.f.$f_Z(z;\theta)$, where θ is the vector of unknown parameters modeling the speech and noise waveforms:

$$\theta = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \\ g_s \\ A_s \\ \phi_s \\ T_s \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_q \\ g_v \end{bmatrix} \tag{3.1}$$

The *Maximum-Likelihood* (ML) estimate of θ is obtained by solving

$$\hat{\theta}_{ML} = \arg\max_{\theta\in\Theta} \log f_Z(z;\theta) \tag{3.2}$$

Our objective is to estimate the clean speech samples $s(t)$ from the observed data. The use of the EM procedure for solving the ML problem enables us to get the speech estimate as a by-product of the parameter estimate.

The EM algorithm is described in appendix A. Here We will only quote the main results necessary for our development.

The EM algorithm is an iterative procedure for finding the ML parameter estimates. To apply the EM algorithm we need to define a "complete data" $\mathbf{y}$ which depends on the observed data ("incomplete data") $\mathbf{z}$ by:

$$\mathbf{z} = H(\mathbf{y}) \tag{3.3}$$

where $H(\cdot)$ is a non-invertible (many-to-one) transformation. Then, each iteration of the EM algorithm consists of the following steps:

E-step

$$Q(\theta, \theta^{(l)}) = E_{\theta^{(l)}} \left\{ \log f_Y(y; \theta) | z \right\} \tag{3.4}$$

M-step

$$\max_{\theta} Q(\theta, \theta^{(l)}) \rightarrow \theta^{(l+1)} \tag{3.5}$$

Intuitively, we get in the E-step an estimate of the "complete data" statistics given the "incomplete data", and based on the current estimate of the parameters (which is exactly the desired signal). In the M-step we solve the ML problem using the current estimate of the "complete data" instead of using the observed samples. The crucial point in any implementation of the EM algorithm is the definition of the "complete data". We will choose the "complete data" as a concatenation of the corrupted samples $z(t)$ and the clean speech samples $s(t)$. By this choice, we will obtain the estimate of the signal $s(t)$, as a by-product of the ML parameter estimation. For the purpose of signal enhancement, it is the signal estimate that we are interested in. Nevertheless, the parameters we estimate may be also useful in several application, such as Automatic Speech Recognition systems and speech compression algorithms.

This chapter organization is as follows. In the next section we will give a detailed development of the algorithm. Then special cases and several simplifications for the resulted algorithm will be discussed. We will conclude by addressing the problem of improving the AR parameter estimation, which improves the convergence behavior of the algorithm. Chapter 4 will give another version of the enhancement algorithm, namely a sequential/adaptive algorithm.

3.2 Detailed Algorithm Development

We will start by few assumptions. In order to compensate for the non-stationarity of the speech and noise, we divide the signal into (perhaps overlapping) segments, short enough for the stationarity assumption to hold. Standard choice for the segment length in speech signals is about 20 mSec. The noise segments may be even longer.

Assume also, for the time being, that the white-noise component of the speech innovation sequence is Gaussian, which causes the speech signal to have a Gaussian p.d.f.. We have already assumed that the noise signal Gaussian. We will see that under those assumptions the ML estimate coincidences with an optimal linear Minimum Mean Square Error (MMSE) estimation of the speech signal.

Denote by:

$$\mathbf{z} = \{z(t) : t = 1, 2, \ldots, N\} \tag{3.6}$$

the "incomplete data" segment, and by:

$$\mathbf{s} = \{s(t) : t = -p+1, -p+2, \ldots, N\} \tag{3.7}$$

a collection of speech samples which constitutes the "desired signal". N is the segment length and p is the speech AR order.

Define by:

$$\mathbf{y} = \begin{bmatrix} \mathbf{z} \\ \mathbf{s} \end{bmatrix} \tag{3.8}$$

the "complete data".

Invoking Bayes's rule,

$$f_Y(y;\theta) = f_S(s;\theta) \cdot f_{Z|S}(z|s;\theta) \tag{3.9}$$

Where θ is the vector of unknown parameters defined in equation 3.1. Taking the logarithm of both sides gives:

$$\log f_Y(y;\theta) = \log f_S(s;\theta) + \log f_{Z|S}(z|s;\theta) \tag{3.10}$$

From the assumption that the stochastic input is Gaussian and white and from the AR nature of the speech (2.1), we can write:

$$\log f_S(s;\theta) = \log f(s_{p-1}(0)) - \frac{N}{2}\log 2\pi g_s - \frac{1}{2g_s}\sum_{t=1}^{N}[s(t) + \alpha^T s_{p-1}(t-1) - A_s g(t)]^2 \tag{3.11}$$

where we defined a speech state vector as:

$$s_p(t) = \begin{bmatrix} s(t-p) \\ s(t-p+1) \\ \vdots \\ s(t) \end{bmatrix} \tag{3.12}$$

and $f(s_{p-1}(0))$ denotes the p.d.f.of $s_{p-1}(0)$, the initial condition of the frame for the speech signal. Than, encountering the fact that $\log f_{Z|S}(z|s;\theta) = \log f_V(v,\theta)$ and from the same assumptions about the noise (2.2), we can also write:

$$\log f_V(y;\theta) = \log f(v_{q-1}(0)) - \frac{N}{2}\log 2\pi g_v - \frac{1}{2g_v}\sum_{t=1}^{N}[v(t) + \beta^T v_{q-1}(t-1)]^2 \tag{3.13}$$

where we defined a noise state vector as:

$$v_q(t) = \begin{bmatrix} v(t-q) \\ v(t-q+1) \\ \vdots \\ v(t) \end{bmatrix} \tag{3.14}$$

and $f(v_{q-1}(0))$ denotes the p.d.f.of $v_{q-1}(0))$, the initial condition of the frame for the noise signal. Substituting 3.11 and 3.13 into 3.10 and assuming that the $\log f(s_{p-1}(0))$ and $\log f(v_{q-1}(0))$ are known from the previous segment (we can get them from the end of the previously enhanced segment), we obtain:

$$\begin{aligned} \log f_Y(y;\theta) &= C - \frac{N}{2}\log g_s - \frac{N}{2}\log g_v \qquad (3.15) \\ &- \frac{1}{2g_s}\sum_{t=1}^{N}[s(t) + \alpha^T s_{p-1}(t-1) - A_s g(t)]^2 \\ &- \frac{1}{2g_v}\sum_{t=1}^{N}[v(t) + \beta^T v_{p-1}(t-1)]^2 \end{aligned}$$

where C is a constant independent of the parameter vector θ. Taking the conditional expectation given the corrupted measurements $\mathbf{z}$ at a given parameter estimation $\theta^{(l)}$ gives:

$$
\begin{aligned}
Q(\theta, \theta^{(l)}) &= E_{\theta^{(l)}}\{\log f_Y(y;\theta)|z\} \\
&= -\frac{N}{2}\log g_s - \frac{N}{2}\log g_v \\
&\quad -\frac{1}{2g_s}\sum_{t=1}^{N} \widehat{[\epsilon_s(t) - A_s g(t)]^2}^{(l)} \\
&\quad -\frac{1}{2g_v}\sum_{t=1}^{N} \widehat{[\epsilon_v(t)]^2}^{(l)}
\end{aligned}
\tag{3.16}
$$

where we defined by:

$$
\epsilon_s(t) \overset{\triangle}{=} s(t) + \alpha^T s_{p-1}(t-1) \tag{3.17}
$$

$$
\epsilon_v(t) \overset{\triangle}{=} v(t) + \beta^T v_{q-1}(t-1) \tag{3.18}
$$

the estimates of the residual errors, and where for simplicity we used the notation:

$$
(\cdot)^{(l)} \overset{\triangle}{=} E_{\theta^{(l)}}\{\cdot|z\} \tag{3.19}
$$

Writing ϵ_s and ϵ_v explicitly:

$$
\begin{aligned}
&\widehat{[\epsilon_v(t)]^2}^{(l)} = \\
&\quad \widehat{v^2(t)}^{(l)} + 2\beta^T \, \widehat{v_{q-1}(t-1)v(t)}^{(l)} + \beta^T \, \widehat{v_{q-1}(t-1)v_{q-1}^T(t-1)}^{(l)} \beta
\end{aligned}
\tag{3.20}
$$

and

$$
\begin{aligned}
&\widehat{[\epsilon_s(t) - A_s g(t)]^2}^{(l)} = \\
&\quad \widehat{s^2(t)}^{(l)} + \alpha^T \, \widehat{s_{p-1}(t-1)s_{p-1}^T(t-1)}^{(l)} \alpha + A_s^2 g^2(t) \\
&\quad + 2\alpha^T \, \widehat{s_{p-1}(t-1)s(t)}^{(l)} - 2A_s g(t)\widehat{s(t)}^{(l)} - 2A_s g(t)\alpha^T \, \widehat{s_{p-1}(t)}^{(l)}
\end{aligned}
\tag{3.21}
$$

Thus, the computation of $Q(\theta, \widehat{\theta}^{(l)})$ (E-step in EM formulation) requires only the computation of the indicated conditional expectations (which in this case are estimates of the desired signal and its statistics).

Writing it down implicitly:

E-step: For $t = 1, 2, \ldots, N$ compute:

$$\widehat{s_p(t)}^{(l)} = \begin{bmatrix} \widehat{s_{p-1}(t-1)}^{(l)} \\ \widehat{s(t)}^{(l)} \end{bmatrix} \tag{3.22}$$

$$\widehat{s_p(t)s_p^T(t)}^{(l)} = \begin{bmatrix} \widehat{s_{p-1}(t-1)s_{p-1}^T(t-1)}^{(l)} & \widehat{s_{p-1}(t)s(t)}^{(l)} \\ \widehat{s(t)s_{p-1}^T(t)}^{(l)} & \widehat{s^2(t)}^{(l)} \end{bmatrix} \tag{3.23}$$

$$\widehat{v_q(t)}^{(l)} = \begin{bmatrix} \widehat{v_{q-1}(t-1)}^{(l)} \\ \widehat{v(t)}^{(l)} \end{bmatrix} \tag{3.24}$$

$$\widehat{v_q(t)v_q^T(t)}^{(l)} = \begin{bmatrix} \widehat{v_{q-1}(t-1)v_{q-1}^T(t-1)}^{(l)} & \widehat{v_{q-1}(t)v(t)}^{(l)} \\ \widehat{v(t)v_{q-1}^T(t)}^{(l)} & \widehat{v^2(t)}^{(l)} \end{bmatrix} \tag{3.25}$$

The maximization of $Q(\theta, \widehat{\theta}^{(l)})$ with respect to θ (M-step) is done by differentiation operation. First, note that the noise and speech parameters are completely decoupled. This means, that we can solve for the two signals separately, which is a very desirable behavior of the algorithm.

M-step (for noise):

$$\frac{\partial}{\partial \beta} Q(\theta, \widehat{\theta}^{(l)}) = 0 \;\;\Rightarrow\;\; \widehat{\beta}^{(l+1)} \tag{3.26}$$

$$\frac{\partial}{\partial g_v} Q(\theta, \widehat{\theta}^{(l)}) = 0 \;\;\Rightarrow\;\; \widehat{g_v}^{(l+1)} \tag{3.27}$$

M-step (for speech):

$$\frac{\partial}{\partial \alpha} Q(\theta, \widehat{\theta}^{(l)}) = 0 \quad \Rightarrow \quad \widehat{\alpha}^{(l+1)} \tag{3.28}$$

$$\frac{\partial}{\partial g_s} Q(\theta, \widehat{\theta}^{(l)}) = 0 \quad \Rightarrow \quad \widehat{g_s}^{(l+1)} \tag{3.29}$$

$$\frac{\partial}{\partial A_s, T_s, \phi_s} Q(\theta, \widehat{\theta}^{(l)}) = 0 \quad \Rightarrow \quad \widehat{A_s}^{(l+1)}, \widehat{T_s}^{(l+1)}, \widehat{\phi_s}^{(l+1)} \tag{3.30}$$

We will treat now the E-step and the M-step separately.

3.2.1 M-step

We will write down implicitly the operation involved in the maximization operations above.

Solving for the noise parameters involves only simple differentiation, and can be done analytically.

M-step (for noise):

$$\widehat{\beta}^{(l+1)} = -\left[\sum_{t=1}^{N} \widehat{v_{q-1}(t-1)v_{q-1}^T(t-1)}^{(l)}\right]^{-1} \sum_{t=1}^{N} \widehat{v_{q-1}(t-1)v(t)}^{(l)} \tag{3.31}$$

$$\widehat{g_v}^{(l+1)} = \frac{1}{N}\sum_{t=1}^{N}\left[\widehat{v^2(t)}^{(l)} + \widehat{\beta^T}^{(l+1)} \widehat{v_{q-1}(t-1)v(t)}^{(l)}\right] \tag{3.32}$$

which is actually the standard "covariance" solution for the LPC coefficients.

As for the speech parameters it involves a more complicated maximization. For unvoiced segments, the terms involving the pitch series diminishes and we obtain a set of equations similar in structure to the noise parameter maximization, but for voiced segments there is a coupling between the pitch parameters and the LPC parameters.

The same problem was addressed by Burshtein ([4]). He suggested an iterative procedure for decoupling those sets of parameters. We will use his algorithm here.

The algorithm attempts to find, jointly, the LPC parameters, the pitch parameters (period,level, phase) and the white noise level. The algorithm iterates between LPC parameter estimation and pitch determination from the residual error obtained.

The algorithm converges in few iterations to a point in which the residual error is more "white" than the residual obtained by the conventional LPC parameter estimation. Thus, in each of the EM iterations, indexed (l), we are making internal iteration, indexed (k), for the speech parameters.

Summarizing the **M-step** for the speech mathematically:

M-step (for speech): Obtain $\alpha^{(0)}$ by solving :

$$\alpha^{(0)} = -\left[\sum_{t=1}^{N} \widehat{s_{p-1}(t-1)s_{p-1}^T(t-1)}^{(l)}\right]^{-1} \sum_{t=1}^{N} \widehat{s_{p-1}(t-1)s(t)}^{(l)} \tag{3.33}$$

For $k = 0, \ldots, N_{itr} - 1$ do:

$$\eta(t) = \widehat{s(t)}^{(l)} + \alpha^{(k)T}\, \widehat{s_{p-1}(t-1)}^{(l)} \quad t = 0, 1, \ldots, N-1$$

$$\phi_s^{(k+1)}, T_s^{(k+1)} = \arg\max_{\phi_s, T_s} \frac{\sum_{t=0}^{N-1} g(t)\eta(t)}{\sqrt{\sum_{t=0}^{N-1} g^2(t)}} \tag{3.34}$$

$$A^{(k+1)} = \frac{\sum_{t=0}^{N-1} g^{(k+1)}(t)\cdot\eta(t)}{\sqrt{\sum_{t=0}^{N-1} [g^{(k+1)}(t)]^2}}$$

Obtain $\alpha^{(k)}$ by solving :

$$\left[\sum_{t=1}^{N} \widehat{s_{p-1}(t-1)s_{p-1}^T(t-1)}^{(l)}\right]\alpha^{(k)} = -\sum_{t=1}^{N} \widehat{s_{p-1}(t-1)s(t)}^{(l)} + \sum_{t=1}^{N} \widehat{s_{p-1}(t-1)}^{(l)} g^{(k+1)}(t) \tag{3.35}$$

END

After the last internal iteration we obtain the resulted maximization at the current EM iteration by:

$$\widehat{\alpha}^{(l+1)} = \alpha^{(N_{itr})} \tag{3.36}$$

$$\begin{aligned}\widehat{A_s}^{(l+1)} &= A_s^{(N_{itr})}\\ \widehat{\phi_s}^{(l+1)} &= \phi_s^{(N_{itr})}\\ \widehat{T_s}^{(l+1)} &= T_s^{(N_{itr})}\end{aligned}$$

and the estimation of the white noise innovation gain:

$$\begin{aligned}&\widehat{g_s}^{(l+1)} = \qquad (3.37)\\ &\frac{1}{N}\sum_{t=1}^{N-1}\Bigg[\widehat{s^2(t)}^{(l)} + (\widehat{\alpha}^{(l+1)})^T\ \widehat{s_{p-1}(t-1)s_{p-1}^T(t-1)}^{(l)}\ \widehat{\alpha}^{(l+1)}\\ &+(\widehat{A_s}^{(l+1)})^2(\widehat{g(t)}^{(l+1)})^2 + 2(\widehat{\alpha}^{(l+1)})^T\ \widehat{s_{p-1}(t-1)s(t)}^{(l)}\\ &-2\widehat{A_s}^{(l+1)}\widehat{g(t)}^{(l+1)}\widehat{s(t)}^{(l)} - 2\widehat{A_s}^{(l+1)}\widehat{g(t)}^{(()}\widehat{\alpha}^{(l+1)})^T\ \widehat{s_{p-1}(t)}^{(l)}\Bigg]\end{aligned}$$

We will not address here several practical consideration like the grid search of the exact pitch period, but some of them were implemented in our software.

The Author indicates that the LPC parameters obtained by this procedure are better features for an Automatic Speech Recognition system. This observation, makes this choice even more attractive for our application (we will address the Automatic Speech Recognition problem later in chapter 6).

To conclude the **M-step** formulation note that the M-step splits into two separate parts: the speech and noise. which form a decoupled set of equations. The noise equations are eventually the *Yule-Walker* solution for the AR parameters (using the "Covariance method"), where the first and second order statistics of the noise are substituted by their current estimate. The speech equations are more complicated and involves an iterative solution, which uses the estimation of the first and second order statistics of the speech. For both signals the expectation is realized by the summation operation, over the entire segment. This decoupling makes the algorithm very convenient for use. One can solve for the speech parameters without knowing anything about the noise parameters, and vice versa: to solve for the noise parameters without knowing the speech parameters.

3.2.2 E-step

We will develop now an implicit equation for implementing the **E-step**. Note that the equations 3.22, 3.23, 3.24, 3.25, involves conditional expectation given the observations of the segment at the current parameters value, which is the MMSE estimation of both the speech and the noise. This is also an intuitive form.

Therefore, we can implement those expectations by using the well known solution for the optimal linear MMSE filters, which will also be the general optimal solution under the Gaussian assumption. Remember, that this assumption holds only for the time being.

In Lim and Oppenheim's paper the non-casual Wiener filter was used. In Weinstein, Oppenheim and Feder's paper [7] - which is the basis of our development here - the Kalman smoother was used, which does not make any restrictions concerning stationarity. The problem with using the Kalman smoother is its more complicated form.

First, we represent the speech and noise equations 2.1, 2.2 in state-space form, as required by the Kalman equations:

$$s_p(t) = \Phi_s s_p(t-1) + G_s u(t) + D_s g(t) \tag{3.38}$$

$$v_q(t) = \Phi_v v_q(t-1) + G_v w(t) \tag{3.39}$$

$$z(t) = \begin{bmatrix} H_s^T & H_v^T \end{bmatrix} \begin{bmatrix} s_p(t) \\ v_q(t) \end{bmatrix} \tag{3.40}$$

where the state vector $s_p(t)$ is the $(p+1) \times 1$ vector of speech samples defined as in 3.12 by:

$$s_p(t) = \begin{bmatrix} s(t-p) \\ s(t-p+1) \\ \vdots \\ s(t) \end{bmatrix} \tag{3.41}$$

and the state vector $v_q(t)$ is the $(q+1) \times 1$ vector of noise samples defined as in 3.14 by:

$$v_q(t) = \begin{bmatrix} v(t-q) \\ v(t-q+1) \\ \vdots \\ v(t) \end{bmatrix} \tag{3.42}$$

Φ_s is the $(p+1) \times (p+1)$ speech transition matrix:

$$\Phi_s = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 1 \\ 0 & -\alpha_p & \cdots & \cdots & -\alpha_2 & -\alpha_1 \end{bmatrix} \tag{3.43}$$

and Φ_v is the $(q+1) \times (q+1)$ speech transition matrix:

$$\Phi_v = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 1 \\ 0 & -\beta_q & \cdots & \cdots & -\beta_2 & -\beta_1 \end{bmatrix} \tag{3.44}$$

G_s is the $(p+1) \times 1$ speech stochastic input vector:

$$G_s^T = [\ 0 \ \ \ldots \ \ 0 \ \ \sqrt{g_s}\ ] \tag{3.45}$$

G_v is the $(q+1) \times 1$ noise stochastic input vector:

$$G_v^T = [\ 0 \ \ \ldots \ \ 0 \ \ \sqrt{g_v}\ ] \tag{3.46}$$

H_s is the $(p+1) \times 1$ speech measurement vector:

$$H_s^T = [\ 0 \ \ \ldots \ \ 0 \ \ 1\ ] \tag{3.47}$$

H_v is the $(q+1) \times 1$ noise measurement vector:

$$H_v^T = [\ 0 \ \ \ldots \ \ 0 \ \ 1\ ] \tag{3.48}$$

D_s is the $(p+1) \times 1$ speech deterministic input vector:

$$D_s^T = [\ 0 \quad \dots \quad 0 \quad A_s\ ] \tag{3.49}$$

Define $X(t)$ to be the $(p+q+2) \times 1$ extended state vector:

$$X(t) \triangleq \begin{bmatrix} s_p(t) \\ \\ v_q(t) \end{bmatrix} \tag{3.50}$$

and G to be the $(p+q+2) \times 2$ extended stochastic input matrix:

$$G \triangleq \begin{bmatrix} G_s & 0 \\ \\ 0 & G_v \end{bmatrix} \tag{3.51}$$

and Φ to be the $(p+q+2) \times (p+q+2)$ extended state transition matrix:

$$\Phi \triangleq \begin{bmatrix} \Phi_s & 0 \\ \\ 0 & \Phi_v \end{bmatrix} \tag{3.52}$$

and H to be the $1 \times (p+q+2)$ extended stochastic input vector:

$$H \triangleq \begin{bmatrix} H_s^T & H_v^T \end{bmatrix} \tag{3.53}$$

and $U(t)$ to be the 2×1 extended stochastic input vector:

$$U(t) \triangleq \begin{bmatrix} u(t) \\ \\ w(t) \end{bmatrix} \tag{3.54}$$

and D to be the 2×1 extended deterministic input vector:

$$D \triangleq \begin{bmatrix} D_s \\ \\ 0 \end{bmatrix} \tag{3.55}$$

Thus, we can write compactly:

$$\begin{aligned} X(t+1) &= \Phi X(t) + GU(t) + Dg(t) \\ z(t) &= HX(t) \end{aligned} \tag{3.56}$$

This set of equations 3.57 is called "noise free" model, because the noise is not stated explicitly, but as another input. In order to prevent stability problems it is a common procedure to add a very low-level white noise to the measurement equation in 3.57. This addition eliminates the ill-conditioned matrix inversion, which could occur in an extremely high SNR values, but still does not effect the performance of the algorithm at all.

The Kalman smoothing equations can be stated now in terms of the previous definitions, and by defining for convenience:

$$\widehat{\mu}^{(l)}(t|n) \triangleq E_{\theta^{(l)}}\{x(t)|z(0), z(1), \ldots, z(n)\} \tag{3.57}$$

$$\widehat{P}^{(l)}(t|n) \triangleq E_{\theta^{(l)}}\{\left[\widehat{\mu}^{(l)}(t|t) - x(t)\right]\left[\widehat{\mu}^{(l)}(t|t) - x(t)\right]^T |z(0), z(1), \ldots, z(n)\} \tag{3.58}$$

to be the state-vector estimate and its related covariance matrix using n observed samples.

Clearly, the first and second order statistics in equation 3.22 3.23 3.24 3.25 can be expressed by:

$$\widehat{x(t)}^{(l)} = \widehat{\mu}^{(l)}(t|N) \tag{3.59}$$

$$\widehat{x(t)x^T(t)}^{(l)} = \widehat{\mu}^{(l)}(t|N)\widehat{\mu}^{(l)}(t|N)^T + \widehat{P}^{(l)}(t|N) \tag{3.60}$$

Note, that we used the entire segment (i.e. $n = N$).

Using the current estimation of the parameters vector, $\widehat{\theta}^{(l)}$, we can write the Kalman smoothing equations in three stages as follows:

Propagation Equation For t=1,2,. . . ,N:

$$\widehat{\mu}^{(l)}(t|t-1) = \widehat{\Phi}^{(l)}(t)\widehat{\mu}^{(l)}(t-1|t-1) + D\widehat{g(t)}^{(l)} \tag{3.61}$$

$$\widehat{P}^{(l)}(t|t-1) = \widehat{\Phi}^{(l)}(t)\widehat{P}^{(l)}(t-1|t-1)(\widehat{\Phi}^{(l)}(t))^T + GG^T \tag{3.62}$$

Updating Equation For t=1,2,...,N:

$$\widehat{\mu}^{(l)}(t|t) = \widehat{\mu}^{(l)}(t|t-1) + \widehat{k}^{(l)}(t)\left[z(t) - H^T\widehat{\mu}^{(l)}(t|t-1)\right] \tag{3.63}$$

$$\widehat{P}^{(l)}(t|t) = \widehat{P}^{(l)}(t|t-1) - \widehat{k}^{(l)}(t)H^T\widehat{P}^{(l)}(t|t-1) \tag{3.64}$$

where:

$$\widehat{k}^{(l)}(t) \triangleq \frac{\widehat{P}^{(l)}(t|t-1)H}{H^T\widehat{P}^{(l)}(t|t-1)H} \tag{3.65}$$

Smoothing Equation For t=N,N-1,...,1:

$$\begin{aligned}\widehat{\mu}^{(l)}(t-1|N) = &\\ &\widehat{\mu}^{(l)}(t-1|t-1) + \widehat{S}^{(l)}(t-1)\\ &\left[\widehat{\mu}^{(l)}(t-1|N) - \widehat{\Phi}^{(l)}\widehat{\mu}^{(l)}(t-1|t-1)\right]\end{aligned} \tag{3.66}$$

$$\begin{aligned}\widehat{P}^{(l)}(t-1|N) = &\\ &\widehat{P}^{(l)}(t-1|t-1) - \widehat{S}^{(l)}(t-1)\\ &\left[\widehat{P}^{(l)}(t|N) - \widehat{P}^{(l)}(t|t-1)\right])\widehat{S}^{(l)}(t-1)^T\end{aligned} \tag{3.67}$$

where:

$$\widehat{S}^{(l)}(t-1) \triangleq \widehat{P}^{(l)}(t-1|t-1)\left(\widehat{\Phi}^{(l)}\right)^T\left(\widehat{P}^{(l)}(t|t-1)\right)^{-1} \tag{3.68}$$

Note, that at each iteration the Kalman procedure is done sample by sample, but the various parameters update is done only once (by the **M-step** equations).

This concludes the entire Iterative-Batch algorithm. In the next sub-section we will make some simplifications that will yield a sub-optimal algorithm but easier to implement.

3.3 Simplifications and Special Cases

The algorithm we developed in the previous subsection is a very complicated one. There are several points that make the computational burden very high. We will try now to make some assumptions regarding the components of the algorithm in order to simplify its structure.

Smoothing The smoothing part of Kalman equations enforces the algorithm to repeat and process each data segment twice. A well known suboptimal procedure restrict the algorithm to be causal. Namely each data point is estimated using only data points from the beginning of the segments (using the correct initial conditions) till the currently estimated point (fixing $n = t$ in 3.57 and 3.58 instead of $n = N$). This procedure yields the the Kalman Filtering equations, which have a sequential form. In chapter 4, while trying to convert our algorithm to a completely sequential one we will use this simplification again. We decided that the benefit of using the smoothing equation does not justify the calculation burden, so we omitted it completely from our implementation.

To retain some of the smoothing benefits we used the **Fixed Lag Smoothing** procedure. Remember, that we are not estimating only one speech sample at a time, but a whole state-space vector. This vector contains, several speech samples as is evident from equation 3.69:

$$s_p(t) = \begin{bmatrix} s(t-p) \\ s(t-p+1) \\ \vdots \\ s(t) \end{bmatrix} \tag{3.69}$$

Thus, while $s(t)$ is estimated by casual filtering, $s(t-p)$ is estimated by means of fixed-lag-smoothing, using p samples ahead. The estimation accuracy of the smoother is better, without any computationally punishment.

Pitch information We saw that incorporating the estimation of the pitch series into our speech model, causes the LPC parameter estimation problem to be coupled with the parameter estimation. Those coupled equations were solved iteratively. Note, that the pitch information appears both in the **E-step**, as the deterministic input in Kalman propagation equations, and in the **M-step** in the extended version of Yule-Walker equations.

We suggest a possible simplification: merging the iterations. Due to the iterative structure of the parameter estimation, we suggest to use only one internal maximization iteration in the **M-step** and to use those parameters (including pitch information) in the **E-step**. In this way we are, in a sense, merging the iteration indices into one unified index of iteration. At each M-step we do not bring $Q(\theta, \theta^{(l)})$ to its maximum, but just increase its value. As a further simplification we can choose to omit the use the pitch information. We implemented in our algorithm the possibility to use the pitch information in either steps or in both of them or in none of them. Of course, using the pitch only in the E-step forces an existence of prior information about it. This is not a realistic assumption, but can be used for evaluation purposes. Omitting the pitch information from our algorithm at all will cause the speech parameter estimation equations to be similar in structure to the noise parameter estimation equations.

Choosing only one M-step iteration, minimizes the computational burden, on expense of the algorithm behavior.

As a final comment, we suggest to try to use some simpler pitch detection algorithms (e.g. [14], [19]). The use of the pitch is under further investigation now.

"Colored" noise We should note, that using the more complex structure for the noise (e.g. "colored" instead of "white") does not involves a great amount of computation. It just increases the size of the matrices involved slightly. For that reason, we are using the "colored" form of Kalman equations, which could yield a great improvement in the performance. Finally, it should be clear that choosing $q = 0$ for the noise AR model, will simplify the equation to the form suggested in [7]. So, our algorithm is a generalization of the previous algorithm.

3.4 Improving LPC Estimation

The problem of initializing the algorithm is a crucial one. As we saw, the algorithm we developed decouples the speech and noise estimation. So, theoretically we can solve the problem without any prior information about the noise characteristic, and we do not need to learn it from speech-free periods, as other algorithms do. But, it should be emphasized that the problem we address is a difficult one. The algorithm need some good starting point in order to proceed and process the speech and noise in a separated manner. Otherwise, it might converge to a local minima of the ML estimator. To prevent this phenomena we consider applying an improved initial estimate of the speech and noise parameters. We will address the speech and noise signals separately.

3.4.1 Noise Parameter Estimation

For the noise signal the initialization of the parameters depends on the Signal-To-Noise-Ratio (SNR). In low SNR range (which is the main interest of our work), the noise level is much stronger than the desired speech level. So, using the conventional Yule-Walker equations, which uses second-order statistics of the corrupted sample, may yield a very good estimate of the noise AR parameters.

The reason for that is obvious. Each covariance value of the corrupted samples is derived by the sum of the covariance values of the statistically independent speech and noise signals. Since the noise level is much stronger than the speech level, and since the spectral contents (and its Fourier Transform pair - the “correlation length”) are quite the same, the covariance values are mostly due to the noise. On the contrary, in high SNR range a prior averaged value of the noise parameters is necessary. The initial noise parameters can be estimated from speech-free segments at the beginning of the sentence, or from “silence” periods between sentences. The second approach requires the use of a very good speech activity detector, which is very hard to implement in adverse conditions.

3.4.2 Speech Parameters Estimation

As for the speech we need a different approach. For the SNR values of interest the correlation matrix used in the Yule-Walker set of equations are completely corrupted. So, we will need to develop a new set of equations. This development should use the different properties of the speech and the noise. We will try to use two distinguishing properties: **"Correlation length"** and **Gussianity**.

"Correlation length" The speech correlation length is assumed to be much longer than that of the noise. This is especially true for "white" noise, ("white" noise means zero correlation length). This case covers an interesting variety of problems, although not very common.

Gussianity The speech signal has no Gaussian p.d.f.. This fact was stated in several text books (e.g. [15]), and was also shown by several experimental results we achieved. In these experiments we divided the speech into short segments. We used common statistical tests to show that speech is quite far from Gussianity in both voiced and unvoiced segments. On the contrary, a large variety of noise signals can be modeled as having almost Gaussian p.d.f.. This is true probably because most noise production mechanism include a summation of several sources which interfere together to yield a signal with a tendency towards Gussianity. We exclude from our treatment periodic noise sources, that can be eliminated by very simple methods such as Widrow's algorithm. Thus, Gussianity, seems to be a very good distinguishing property in our case. We can exploit this separating feature by the use of Higher-Order-Statistics (HOS) techniques. So, a good practice is to give up the Gaussian assumption we made for the speech signal. HOS has gained an increasing interest in the last decade. HOS has begun to find wide applicability in many diverse fields; e.g. sonar, radar, biomedicine, seismic data, image reconstruction, adaptive filtering and blind equalization, and recently speech processing ([8], [28], [17] and a lot more). Definition and several properties of HOS can be found in Appendix B.

We will now develop methods that can use those two properties for improving the AR parameter estimation. Let, the corrupted speech samples be $z(t) = s(t) + v(t)$, where $s(t)$ is the speech signal, assuming now a simpler model that does not include the pitch innovation:

$$s(t) = -\sum_{k=1}^{p} \alpha_k s(t-k) + \sqrt{g_s} u(t) \tag{3.70}$$

and $v(t)$ is the noise signal (retaining the same model as in as 2.2):

$$v(t) = -\sum_{k=1}^{q} \beta_k v(t-k) + \sqrt{g_v} w(t) \tag{3.71}$$

Calculating the Cumulant series:

$$\begin{aligned} &\mathrm{cum}[z(t), z(t-l_1), z[t-l_2], \ldots, z(t-l_M)] = \\ &\quad \mathrm{cum}[\sqrt{g_s} u(t) - \sum_{k=1}^{p} \alpha_k s(t-k) + \sqrt{g_v} w(t) \\ &\quad -\sum_{k=1}^{q} \beta_k v(t-k), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] \end{aligned} \tag{3.72}$$

$$\begin{aligned} = \ & \mathrm{cum}[\sqrt{g_s} u(t), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] + \\ & \mathrm{cum}[-\sum_{k=1}^{p} \alpha_k s(t-k), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] + \\ & \mathrm{cum}[\sqrt{g_v} w(t), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] + \\ & \mathrm{cum}[-\sum_{k=1}^{q} \beta_k v(t-k), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] \end{aligned} \tag{3.73}$$

Where the last transition used the linearity property of the cumulants (B.1). Now, assuming $l_1, l_2, \ldots, l_M \geq 0$ causes the terms involving the innovation sequences $u(t)$ and $w(t)$ to diminish, by virtue of property B.3.

Further application of the linearity property of the cumulants gives:

$$\begin{aligned} = \ & -\sum_{k=1}^{p} \alpha_k \mathrm{cum}[s(t-k), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] \\ & -\sum_{k=1}^{q} \beta_k \mathrm{cum}(v(t-k), z(t-l_1), z(t-l_2), \ldots, z(t-l_M)] \end{aligned} \tag{3.74}$$

Splitting the terms that include $z(t-l_i)$ in 3.74 into its components and using again property B.3 gives:

$$\begin{aligned} = & -\sum_{k=1}^{p} \alpha_k \text{cum}[s(t-k), s(t-l_1), s(t-l_2), \ldots, s(t-l_M)] \\ & -\sum_{k=1}^{p} \alpha_k \text{cum}[s(t-k), v(t-l_1), v(t-l_2), \ldots, v(t-l_M)] \\ & -\sum_{k=1}^{q} \beta_k \text{cum}(v(t-k), s(t-l_1), s(t-l_2), \ldots, s(t-l_M)] \\ & -\sum_{k=1}^{q} \beta_k \text{cum}(v(t-k), v(t-l_1), v(t-l_2), \ldots, v(t-l_M)] \end{aligned} \tag{3.75}$$

Remember that $s(t)$ and $v(t)$ are statistically independent. So, using property B.3 once more causes the mixed terms to diminish. This gives us the final formula.

$$\begin{aligned} & \text{cum}[z(t), z(t-l_1), z[t-l_2], \ldots, z(t-l_M)] = \\ & \quad \text{cum}[s(t), s(t-l_1), s(t-l_2), \ldots, s(t-l_M)] \\ + & \quad \text{cum}[v(t), v(t-l_1), v(t-l_2), \ldots, v(t-l_M)] \\ = & -\sum_{k=1}^{p} \alpha_k \text{cum}[s(t-k), s(t-l_1), s(t-l_2), \ldots, s(t-l_M)] \\ & -\sum_{k=1}^{q} \beta_k \text{cum}(v(t-k), v(t-l_1), v(t-l_2), \ldots, v(t-l_M)] \end{aligned} \tag{3.76}$$

Now we should choose the order M and the lags $l_1, l_2, \ldots, l_M$ to accomplish our goals of distinguishing between speech and noise.

Several choices are available:

1. $M = 1$.

   Denote the first value of l_1 as L.

   Define by:

$$\begin{aligned} R_s(k-l) & \triangleq \text{cum}[s(t-k), s(t-l)] \\ R_v(k-l) & \triangleq \text{cum}[v(t-k), v(t-l)] \end{aligned}$$

covariance values of the speech and noise, respectively (we assume stationarity in each segment). Further assuming "white" noise and determining the value of L gives:

(a) $L = 1$. Choosing p consecutive values for l_1, and writing down equation 3.76 in matrix form gives:

$$\begin{bmatrix} R_s[0]+R_v[0] & R_s[-1] & \cdots & R_s[+1-p] \\ R_s[1] & R_s[0]+R_v[0] & \cdots & R_s[2-p] \\ \cdots & \ddots & \ddots & \cdots \\ \cdots & \cdots & \ddots & \ddots \\ R_s[p-1] & \cdots & \cdots & R_s[0]+R_v[0] \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{bmatrix} = - \begin{bmatrix} R_s[1] \\ R_s[2] \\ \vdots \\ R_s[p] \end{bmatrix} \tag{3.77}$$

Which is of course the original **Yule-Walker** equations.

(b) $L = p + 1$. Choosing again p consecutive values for l_1, and writing down equation 3.76 in matrix form gives:

$$\begin{bmatrix} R_s[p+1] & R_s[p] & \cdots & R_s[2] \\ R_s[p+2] & R_s[p+1] & \cdots & R_s[3] \\ \cdots & \ddots & \ddots & \cdots \\ \cdots & \cdots & \ddots & \ddots \\ R_s[2p] & \cdots & \cdots & R_s[p+1] \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{bmatrix} = - \begin{bmatrix} R[p+2] \\ R[p+3] \\ \vdots \\ R[2p+1] \end{bmatrix} \tag{3.78}$$

which is the well-known **Modified Yule-Walker**.

The trade-off between the equations is now evident. While the **original** equations suffers from a bias effect due to the "corruption" of the diagonal of the correlation matrix, the **modified** equations suffers from large estimation variance due to the decreasing statistical stability of higher-lag correlation estimates.

Finally, we should remark that the correlation sequence of "colored" noise become infinite (although decreasing rapidly). This forces a choice of even higher starting lag L causing unbearable estimation variance, or with the same choice of L, causing a bias effect. This noise characteristics makes the **modified Yule-Walker** equations useless.

2. $M = 2$. This choice of M gives us a set of third order cumulant based equations. Invoking property B.2 of the noise cumulants gives us "noise free" equations.

$$\begin{aligned} & \mathrm{cum}[s(t), s(t-l_1), s(t-l_2)] \\ = & -\sum_{k=1}^{p} \alpha_k \mathrm{cum}[s(t-k), s(t-l_1), s(t-l_2)] \end{aligned} \tag{3.79}$$

   Any choice of l_1, l_2 should be good. We made the choice:

$$\begin{aligned} 1 \leq l_1 \leq p \\ 0 \leq l_2 \leq l_1 \end{aligned} \tag{3.80}$$

   which were chosen in [17] and makes a reasonable compromise between selecting enough equations and maintaining statistical stability. This gives us a set of $\frac{p^2+3p}{2}$ over-determined equations for the parameters $\{\alpha_i : i = 1, 2, \ldots, p\}$. The energy level is estimated from the energy of the residual series obtained by applying the AR coefficients to the corrupted speech signal.

   This choice is applicable for all kinds of symmetric p.d.f.noise signals. So, in this sense, it covers a wide variety of noise signals. But it might be not a very good distinguishing property, since in our case the speech has also a quite symmetric p.d.f.. So, it is reasonable to use third order cumulant only when working with noise possessing a very precise symmetric p.d.f., such as sine wave. In this special case there exists very efficient algorithms, and choosing our algorithm seems as a very bad choice.

3. $M = 3$. This choice of M gives us a set of fourth order cumulant based equations. Invoking property B.2 of the noise cumulants gives us again "noise free" equations.

$$\mathrm{cum}[s(t), s(t-l_1), s(t-l_2), s(t-l_3)] \tag{3.81}$$

$$= \quad -\sum_{k=1}^{p} \alpha_k \text{cum}[s(t-k), s(t-l_1), s(t-l_2), s(t-l_3)]$$

Any choice of l_1, l_2, l_3 should be good. We made the choice:

$$\begin{aligned} 1 \leq l_1 \leq p \\ 0 \leq l_2 \leq l_1 \\ 0 \leq l_3 \leq l_2 \end{aligned} \tag{3.82}$$

which makes a reasonable compromise between selecting enough equations and maintaining statistical stability. This gives us a set of $\frac{11p+6p^2+p^3}{6}$ over-determined equations for the parameters $\{\alpha_i : i = 1, 2, \ldots, p\}$. The energy level is estimated from the energy of the residual series obtained by applying the AR coefficients to the corrupted speech signal. There, certainly, exist a smaller set of equations that has the same performance but this had not been checked in this work.

For conclusion we will make some final remarks. Although, we developed the approach of using Higher Order Statistics only for initialization purposes, it seems as a good idea to use it for all the iterations. The idea arise from the intuitive structure of the algorithm, which iterates between parameters estimation and optimal filtering. It is then suggested to replace the parameter estimation in each iteration by the methods that were developed in the subsection. We will address the influence of doing so in chapter 6. Here we will note only that the results indicate that this approach is wrong. Using HOS, even making a good feature selection to differentiate between noise and speech, has its problems. As was also noticed by several researcher (e.g. [5] [17]), the formants resulting from HOS based LPC estimation suffers from a tendency to become sharper (reduced bandwidth). Although this is a good phenomena in formant tracking, it causes the enhanced speech to sound unnatural. As we are interested in speech enhancement for a human listener we can not use the HOS at the higher iterations. So, we should restrict ourselves for using HOS only in the initialization iteration and then to use conventional second order estimate. In this way, we have the benefit of initializing the algorithm with a good AR parameter set, and yet retain the natural sounding obtained by the second order statistics (there are some other reasons for doing so, which will become clearer in a while). Another possible way for implementing the LPC estimate is to use a combination of several statistical orders in each iteration. We can choose a set of equations combined from all the Cumulants weighted in the correct way (via the Gauss-Markov theorem). The problem is to find empirically the correct weighting matrix. For this reason, we gave up doing so.

LPC parameters obtained by HOS was proven to be a better feature for Automatic Speech Recognition systems [17] [18]. It might be a reasonable thing to use the LPC parameters obtained from our algorithm (that was initialized by HOS estimation) as a feature in the ASR. We have not taken this approach yet. Instead, we concentrated in the use of the algorithm as a preprocessor for Automatic Speech Recognition systems. This matter is, surely, a subject of further research.

3.5 Summary

The Iterative-Batch solution for the Speech Enhancement problem was developed. The structure of algorithm is quite intuitive. It iterates between speech and noise parameter estimation and signal enhancement (the same structure of the Lim and Oppenheim algorithm [12]). The signal enhancement is done by the Kalman filter. The noise parameter estimation is done by solving the Yule-Walker equation, and for the speech, by solving an extended set of Yule-Walker equation, that takes the pitch into account. There is a complete decoupling between the speech and the noise parameter estimation equations. The initialization of the speech parameter is achieved by using HOS. The problem of non-stationarity of the speech and the noise was treated by dividing the signal into short enough segments. The resulting algorithm converges after 5-6 iterations on each data segment.

Chapter 4

Sequential/Adaptive Algorithm

4.1 Introduction

In the previous chapter we developed an **Iterative-batch** algorithm for solving the *Maximum-Likelihood* problem by applying the *Estimated-Maximize* solution.

Two problems arise from this algorithm. First, processing each segments several times causes an extensive computational burden. Second, the abrupt changes allowed at each segment border might cause an unnatural sounding of the resulted speech.

We suggest a modification (see also [7]), in which the iteration index is replaced by the time index. Note, that the Kalman filter we used in the simplified **E-step** of the Iterative-Batch solution has already a sequential structure. The **M-step** is the reason for the segmental structure of the **Iterative-batch** algorithm.

This chapter contribution is the development of a sequential/recursive algorithm for solving the second-order-statistics based Yule-Walker equations, by applying a sliding window to the correlation matrices..

Three types of algorithms for recursive solution are developed. One involves a matrix inversion for each sample, The second is similar to the RLS algorithm and the Third is similar to the LMS algorithm. For the time being, we do not use neither HOS nor Pitch information in our sequential algorithm. Use of HOS recursively is a topic for further research.

After using those recursive solutions we obtain a completely sequential algorithm, which updates the parameter estimate and uses the Kalman filter at each new sample. None of the samples are processed more then once. We obtain a computationally efficient algorithm at the cost of a possible degradation in performance.

4.2 Algorithm development

As we mentioned before the **E-step** of the **Iterative-Batch** can be used after omitting the deterministic input (we are not using the pitch information). Note also, that there are no segments in the sequential form, so the entire sentence is treated in one loop.

The Kalman filtering equation obtained is:

For $t = 1, 2, \ldots, T$:

Propagation Equation

$$\widehat{\mu}^{(l)}(t|t-1) = \widehat{\Phi}^{(l)}(t)\widehat{\mu}^{(l)}(t|t-1) \tag{4.1}$$

$$\widehat{P}^{(l)}(t|t-1) = \widehat{\Phi}^{(l)}(t)\widehat{P}^{(l)}(t-1|t-1)(\widehat{\Phi}^{(l)}(t))^T + GG^T \tag{4.2}$$

Updating Equation

$$\widehat{\mu}^{(l)}(t|t) = \widehat{\mu}^{(l)}(t|t-1) + \widehat{k}^{(l)}(t)\left[z(t) - H^T\widehat{\mu}^{(l)}(t|t-1)\right] \tag{4.3}$$

$$\widehat{P}^{(l)}(t|t) = \widehat{P}^{(l)}(t|t-1) - \widehat{k}^{(l)}(t)H^T\widehat{P}^{(l)}(t|t-1) \tag{4.4}$$

where:

$$\widehat{k}^{(l)}(t) \triangleq \frac{\widehat{P}^{(l)}(t|t-1)H}{H^T\widehat{P}^{(l)}(t|t-1)H} \tag{4.5}$$

All the matrices are defined as in chapter 3, and T is the entire sentence length.

Before developing a recursive parameter estimation we will state once again the Yule-Walker equations that were obtained in chapter 3. Note, that in this formulation, both the speech and the noise parameters have essentially the same structure, because both are assumed to be modeled as AR processes excited by a Gaussian white noise.

We need one more modification to those equations. In equations 3.31 3.32 we assumed stationarity of the noise signal (the same is applied for the speech signal).

For that reason the parameters are assumed to be fixed through out the entire segment-length. This was true for short segments, but now we are interested in a sequential solution which does not recognize the segmentation, but still has to consider the changing nature of the parameters. For that reason we incorporate a forgetting factor into the equations. The iteration index is, of course, omitted, and replaced by the time index. The normalization by the segment length N is replaced by an equivalent normalization term $\sum_{\tau=1}^{t} \lambda^{t-\tau}$, giving:

Speech parameters:

$$\widehat{\alpha}(t+1) = -\left[\sum_{\tau=1}^{t} \lambda_s^{t-\tau} \; \widehat{s_{p-1}(t-1)s_{p-1}^T(t-1)}\right]^{-1} \sum_{\tau=1}^{t} \lambda_s^{t-\tau} \; \widehat{s_{p-1}(t-1)s(t)} \tag{4.6}$$

$$\widehat{g}_s(t+1) = \frac{1}{\sum_{\tau=1}^{t} \lambda_s^{t-\tau}} \sum_{\tau=1}^{t} \lambda_s^{t-\tau} \left[\widehat{s^2(t)} + \widehat{\alpha}^T(t+1) \; \widehat{s_{p-1}(t-1)s(t)}\right] \tag{4.7}$$

Noise parameters:

$$\widehat{\beta}(t+1) = -\left[\sum_{\tau=1}^{t} \lambda_v^{t-\tau} \; \widehat{v_{q-1}(t-1)v_{q-1}^T(t-1)}\right]^{-1} \sum_{\tau=1}^{t} \lambda_v^{t-\tau} \; \widehat{v_{q-1}(t-1)v(t)} \tag{4.8}$$

$$\widehat{g}_v(t+1) = \frac{1}{\sum_{\tau=1}^{t} \lambda_v^{t-\tau}} \sum_{\tau=1}^{t} \lambda_v^{t-\tau} \left[\widehat{v^2(t)} + \widehat{\beta}^T(t+1) \; \widehat{v_{q-1}(t-1)v(t)}\right] \tag{4.9}$$

λ_s and λ_v are forgetting factors for the speech and noise, correspondingly, which compensates for the non-stationarity of the signals, by applying a sliding window to the correlation matrix.

λ_s and λ_v satisfies:

$$0 \le \lambda_s, \lambda_v \le 1 \tag{4.10}$$

While $\lambda = 1$ will guarantee maximum stability (no parameter change), $\lambda \approx 0$ will permit a quick parameter change. As the quantity $\frac{1}{1-\lambda}$ relates to the memory length of the estimator (in samples), we chose $\lambda_s, \lambda_v = 0.995$ (which relates approximately to 200 samples). Actually, we may choose λ_v to be closer to 1, due to the slower changing rate of the noise, but this has not been tried.

If we define, as in the **Iterative-Batch** algorithm, the state-vector statistics estimates by:

$$\begin{aligned} \widehat{x(t)} &= \widehat{\mu}(t|t) && (4.11) \\ \widehat{x(t)x^T(t)} &= \widehat{\mu}(t|t)\widehat{\mu}(t|t)^T + \widehat{P}(t|t) && (4.12) \end{aligned}$$

where

$$\widehat{x(t)} \triangleq \begin{bmatrix} \widehat{s_p(t)} \\ \\ \widehat{v_q(t)} \end{bmatrix} \qquad (4.13)$$

is the compound state-vector estimate, with the components:

$$\widehat{s_p(t)} = \begin{bmatrix} \widehat{s(t-p)} \\ \widehat{s(t-p+1)} \\ \vdots \\ \widehat{s(t)} \end{bmatrix} \qquad (4.14)$$

$$\widehat{v_q(t)} = \begin{bmatrix} \widehat{v(t-q)} \\ \widehat{v(t-q+1)} \\ \vdots \\ \widehat{v(t)} \end{bmatrix} \qquad (4.15)$$

and

$$\widehat{x(t)x^T(t)} \triangleq \begin{bmatrix} \widehat{s_p(t)s_p^T(t)} & \widehat{s_p(t)v_q^T(t)} \\ \\ \widehat{v_q(t)s_p^T(t)} & \widehat{v_q(t)v_q^T(t)} \end{bmatrix} \qquad (4.16)$$

is its second order statistics. Every term needed in the Yule-Walker equations can be calculated by the Kalman filter equations.

Now, a sequential solution for the equations 4.6, 4.7, 4.8, 4.9 will be developed. We suggest three sets of recursive solutions: **Recursion with matrix inversion**, **Recursion with Recursive Least Square (RLS)**, and **Recursion with Least Mean Square (LMS)**.

4.2.1 Recursion with Matrix inversion

We develop an algorithm for recursively updating the parameters which involves a new matrix inversion at each sample.

The algorithm is based on the fact that the pre-defined correlation terms can be computed recursively:

For convenience we define new matrices:

$$\begin{aligned} Q^s(t) &\triangleq \begin{bmatrix} Q^s_{11}(t) & Q^s_{12}(t) \\ Q^s_{21}(t) & Q^s_{22}(t) \end{bmatrix} \\ &\triangleq \sum_{\tau=1}^{t} \lambda_s^{t-\tau}\, \widehat{s_p(\tau)s_p^T(\tau)} \\ &= \lambda_s Q^s(t-1) + \widehat{s_p(t)s_p^T(t)} \end{aligned} \tag{4.17}$$

$$\begin{aligned} Q^v(t) &\triangleq \begin{bmatrix} Q^v_{11}(t) & Q^v_{12}(t) \\ Q^v_{21}(t) & Q^v_{22}(t) \end{bmatrix} \\ &\triangleq \sum_{\tau=1}^{t} \lambda_v^{t-\tau}\, \widehat{v_q(\tau)v_q^T(\tau)} \\ &= \lambda_v Q^v(t-1) + \widehat{v_q(t)v_q^T(t)} \end{aligned} \tag{4.18}$$

Using those definitions in 4.6, 4.7, 4.8, 4.9, we can solve for the speech parameters recursively:

$$\begin{aligned} \hat{\alpha}(t+1) &= -{Q^s_{22}}^{-1}(t)Q^s_{21}(t) \\ &= -{Q^s_{22}}^{-1}(t)\left[\lambda_s Q^s_{21}(t-1) + \widehat{s_{p-1}(t-1)s(t)}\right] \\ &= -{Q^s_{22}}^{-1}(t)\left[-\lambda_s Q^s_{22}(t-1)\hat{\alpha}(t) + \widehat{s_{p-1}(t-1)s(t)}\right] \\ &= {Q^s_{22}}^{-1}(t)\left\{[Q^s_{22}(t) - \widehat{s_{p-1}(t-1)s^T_{p-1}(t-1)}]\hat{\alpha}(t) - \widehat{s^T_{p-1}(t-1)s(t)}\right\} \\ &= \hat{\alpha}(t) - {Q^s_{22}}^{-1}(t)\left[\widehat{s_{p-1}(t-1)s^T_{p-1}(t-1)}\,\hat{\alpha}(t) + \widehat{s^T_{p-1}(t-1)s(t)}\right] \end{aligned} \tag{4.19}$$

$$\hat{g}_s(t+1) = \frac{1-\lambda_s}{1-\lambda_s^t}[Q_{11}^s(t) + \hat{\alpha}^T(t+1)Q_{21}^s(t)] \tag{4.20}$$

Where we used the equation:

$$\sum_{\tau=1}^{t} \lambda_s^{t-\tau} = \frac{1-\lambda_s}{1-\lambda_s^t} \tag{4.21}$$

and the same for the noise parameters:

$$\begin{aligned}
\hat{\beta}(t+1) &= -{Q_{22}^v}^{-1}(t)Q_{21}^v(t) \qquad (4.22)\\
&= -{Q_{22}^v}^{-1}(t)\left[\lambda_v Q_{21}^v(t-1) + \widehat{v_{q-1}(t-1)v(t)}\right]\\
&= -{Q_{22}^v}^{-1}(t)\left[-\lambda_v Q_{22}^v(t-1)\hat{\beta}(t) + \widehat{v_{q-1}(t-1)v(t)}\right]\\
&= {Q_{22}^v}^{-1}(t)\left\{[Q_{22}^v(t) - \widehat{v_{q-1}(t-1)v_{q-1}^T(t-1)}]\hat{\beta}(t) - \widehat{v_{q-1}^T(t-1)v(t)}\right\}\\
&= \hat{\beta}(t) - {Q_{22}^v}^{-1}(t)\left[\widehat{v_{q-1}(t-1)v_{q-1}^T(t-1)}\,\hat{\beta}(t) + \widehat{v_{q-1}^T(t-1)v(t)}\right]
\end{aligned}$$

$$\hat{g}_v(t+1) = \frac{1-\lambda_v}{1-\lambda_v^t}[Q_{11}^v(t) + \hat{\beta}^T(t+1)Q_{21}^v(t)] \tag{4.23}$$

Where we used the equation:

$$\sum_{\tau=1}^{t} \lambda_v^{t-\tau} = \frac{1-\lambda_v}{1-\lambda_v^t} \tag{4.24}$$

In the above equations we have a recursive form for the LPC parameters of both the speech and the noise. This form has an important drawback, it uses a new matrix inversion for each sample, which is very time consuming. For that reason we will develop a more efficient algorithm.

4.2.2 Recursion with RLS

We can overcome the drawback of the previous algorithm, by applying a recursion for the inverse of the correlation function, instead of using matrix inversion at each sample. This can be done only by assuming a special structure for that matrix. This kind of structure can be achieved if we give up the use of the covariance matrix, $\widehat{P}(t|t)$, obtained by the Kalman filter to yield the following approximated estimate:

$$\widehat{x(t)} = \widehat{\mu}(t|t) \tag{4.25}$$

$$\widehat{x(t)x^T(t)} = \widehat{\mu}(t|t)\widehat{\mu}(t|t)^T \tag{4.26}$$

This algorithm is quite similar to Ljung development [22].

First, define the normalization factor in equations 4.21 4.24 as:

$$\sum_{\tau=1}^{t} \lambda_s^{t-\tau} = \frac{1-\lambda_s}{1-\lambda_s^t} \triangleq \eta_s(t) \tag{4.27}$$

$$\sum_{\tau=1}^{t} \lambda_v^{t-\tau} = \frac{1-\lambda_v}{1-\lambda_v^t} \triangleq \eta_v(t) \tag{4.28}$$

These normalization factors can be calculated recursively:

$$\eta_s(t) = \lambda_s \eta_s(t-1) + 1 \tag{4.29}$$

$$\eta_v(t) = \lambda_v \eta_v(t-1) + 1 \tag{4.30}$$

Thus, we can define a correlation matrix with the previous definitions as:

$$R^s(t) \triangleq \frac{1}{\eta_s(t)} Q^s(t) \tag{4.31}$$

$$R^v(t) \triangleq \frac{1}{\eta_v(t)} Q^v(t) \tag{4.32}$$

Again $R^s(t)$ and $R^v(t)$ can be written in cells:

$$\begin{aligned} R^s(t) &\triangleq \begin{bmatrix} R^s_{11}(t) & R^s_{12}(t) \\ R^s_{21}(t) & R^s_{22}(t) \end{bmatrix} \\ &\triangleq \frac{1}{\eta_s(t)} \sum_{\tau=1}^{t} \lambda_s^{t-\tau} \widehat{s_p}(\tau)\widehat{s_p}^T(\tau) \\ &= \frac{1}{\eta_s(t)} \left[\lambda_s \eta_s(t-1) R^s(t-1) + \widehat{s_p(t)}\widehat{s_p(t)}^T \right] \\ &= (1 - \frac{1}{\eta_s(t)}) R^s(t-1) + \frac{1}{\eta_s(t)} \widehat{s_p(t)}\widehat{s_p(t)}^T \end{aligned} \tag{4.33}$$

$$R^v(t) \triangleq \begin{bmatrix} R^v_{11}(t) & R^s_{12}(t) \\ \\ R^v_{21}(t) & R^v_{22}(t) \end{bmatrix} \tag{4.34}$$
$$\triangleq \frac{1}{\eta_v(t)} \sum_{\tau=1}^{t} \lambda_v^{t-\tau} \widehat{v_q}(\tau) \widehat{v_q}^T(\tau)$$
$$= \frac{1}{\eta_v(t)} \left[\lambda_v \eta_v(t-1) R^v(t-1) + \widehat{v_q(t)} \widehat{v_q(t)}^T \right]$$
$$= (1 - \frac{1}{\eta_v(t)}) R^v(t-1) + \frac{1}{\eta_v(t)} \widehat{v_q(t)} \widehat{v_q(t)}^T$$

from these recursions we can derive the two following recursions:

$$\begin{aligned} R^s_{22}(t) &= (1 - \frac{1}{\eta_s(t)}) R^s_{22}(t-1) + \frac{1}{\eta_s(t)}) \widehat{s_{p-1}(t-1)} \widehat{s_{p-1}(t-1)}^T \\ R^s_{21}(t) &= (1 - \frac{1}{\eta_s(t)}) R^s_{21}(t-1) + \frac{1}{\eta_s(t)} \widehat{s_{p-1}(t-1)}\ \widehat{s(t)} \end{aligned} \tag{4.35}$$

$$\begin{aligned} R^v_{22}(t) &= (1 - \frac{1}{\eta_v(t)}) R^v_{22}(t-1) + \frac{1}{\eta_v(t)}) \widehat{v_{q-1}(t-1)} \widehat{v_{q-1}(t-1)}^T \\ R^v_{21}(t) &= (1 - \frac{1}{\eta_v(t)}) R^v_{21}(t-1) + \frac{1}{\eta_v(t)} \widehat{v_{q-1}(t-1)}\ \widehat{v(t)} \end{aligned} \tag{4.36}$$

With those relationship in hand we can proceed and calculate the parameter updating recursively by:

$$\hat{\alpha}(t) = (R^s_{22}(t))^{-1} R^s_{21}(t) \tag{4.37}$$
$$\begin{aligned} &= (R^s_{22})^{-1}(t) \left[(1 - \frac{1}{\eta_s(t)}) R^s_{21}(t-1) + \frac{1}{\eta_s(t)} \widehat{s_{p-1}(t-1)}\ \widehat{s(t)} \right] \\ &= (R^s_{22})^{-1}(t) \left[(1 - \frac{1}{\eta_s(t)}) R^s_{22}(t-1) \hat{\alpha}(t-1) + \frac{1}{\eta_s(t)} \widehat{s_{p-1}(t-1)}\ \widehat{s(t)} \right] \\ &= (R^s_{22})^{-1}(t) \left\{ \left[R^s_{22}(t) - \frac{1}{\eta_s(t)} \widehat{s_{p-1}(t-1)}\ \widehat{s(t)} \right] \hat{\alpha}(t-1) + \frac{1}{\eta_s(t)} \widehat{s_{p-1}(t-1)}\ \widehat{s(t)} \right\} \\ &= \hat{\alpha}(t-1) + \frac{1}{\eta_s(t)} (R^s_{22})^{-1}(t)\ \widehat{s_{p-1}(t-1)} \left[\widehat{s(t)} - \widehat{s_{p-1}(t-1)}^T\ \hat{\alpha}(t-1) \right] \end{aligned}$$

and,

$$\hat{\beta}(t) = (R_{22}^v(t))^{-1} R_{21}^v(t) \tag{4.38}$$
$$\begin{aligned}
&= (R_{22}^v)^{-1}(t) \left[(1 - \frac{1}{\eta_v(t)}) R_{21}^v(t-1) + \frac{1}{\eta_v(t)} \widehat{v_{q-1}(t-1)}\ \widehat{v(t)} \right] \\
&= (R_{22}^v)^{-1}(t) \left[(1 - \frac{1}{\eta_v(t)}) R_{22}^v(t-1) \hat{\beta}(t-1) + \frac{1}{\eta_v(t)} \widehat{v_{q-1}(t-1)}\ \widehat{v(t)} \right] \\
&= (R_{22}^v)^{-1}(t) \left\{ \left[R_{22}^v(t) - \frac{1}{\eta_v(t)} \widehat{v_{q-1}(t-1)}\ \widehat{v(t)} \right] \hat{\beta}(t-1) + \frac{1}{\eta_v(t)} \widehat{v_{q-1}(t-1)}\ \widehat{v(t)} \right\} \\
&= \hat{\beta}(t-1) + \frac{1}{\eta_v(t)} (R_{22}^v)^{-1}(t) \widehat{v_{q-1}}(t-1) \left[\widehat{v(t)} - \widehat{v_{q-1}(t-1)}^T\ \hat{\beta}(t-1) \right]
\end{aligned}$$

For simplification we define by:

$$L_s(t) \triangleq \frac{1}{\eta_s(t)} (R_{22}^s)^{-1}(t)\ \widehat{s_{p-1}(t-1)} \tag{4.39}$$

$$L_v(t) \triangleq \frac{1}{\eta_v(t)} (R_{22}^v)^{-1}(t)\ \widehat{v_{q-1}(t-1)} \tag{4.40}$$

"adaptation gains", and by:

$$\widehat{e_s(t)} \triangleq \widehat{s(t)} - \widehat{s_{p-1}(t-1)}^T\ \hat{\alpha}(t-1) \tag{4.41}$$

$$\widehat{e_v(t)} \triangleq \widehat{v(t)} - \widehat{v_{q-1}(t-1)}^T\ \hat{\alpha}(t-1) \tag{4.42}$$

the innovation terms.

With those notations the last recursion can be rewritten as:

$$\hat{\alpha}(t) = \hat{\alpha}(t-1) + L_s(t) e_s(t) \tag{4.43}$$

$$\hat{\beta}(t) = \hat{\beta}(t-1) + L_v(t) e_v(t) \tag{4.44}$$

There is still a matrix inversion for each sample in the last formulas, so we should further simplify the expression.

Define by:

$$(R_{22}^s)^{-1}(t) \triangleq P_s(t) \tag{4.45}$$

$$(R_{22}^{v})^{-1}(t) \triangleq P_v(t) \tag{4.46}$$

the inverse of the correlation matrix main cell. Then by the first parts of 4.33, 4.34 those matrices may be written:

$$P_s(t) = \left[\frac{\eta_s(t)-1}{\eta_s(t)} P_s^{-1}(t-1) + \frac{1}{\eta_s(t)}\, \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T\right]^{-1} = \tag{4.47}$$
$$\frac{\eta_s(t)-1}{\eta_s(t)} P_s(t-1) - \gamma_s P_s(t-1)\, \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)$$

The last transition is by virtue of the Matrix inversion Lemma, so γ_s is given by:

$$\gamma_s = \frac{\eta_s(t)-1}{\eta_s(t)} \times \frac{1}{\eta_s(t)-1+\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)\ \widehat{s_{p-1}(t-1)}} \tag{4.48}$$

Rearranging terms gives:

$$P_s(t) = \frac{\eta_s(t)-1}{\eta_s(t)} \times \left[P_s(t-1) - \frac{P_s(t-1)\ \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)}{\eta_s(t)-1+\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)\ \widehat{s_{p-1}(t-1)}}\right] \tag{4.49}$$

The same can be applied for the noise signal giving:

$$P_v(t) = \frac{\eta_v(t)-1}{\eta_v(t)} \times \left[P_v(t-1) - \frac{P_v(t-1)\ \widehat{v_{q-1}(t-1)}\widehat{v_{q-1}(t-1)}^T\ P_v(t-1)}{\eta_v(t)-1+\widehat{v_{q-1}(t-1)}^T\ P_v(t-1)\ \widehat{v_{q-1}(t-1)}}\right] \tag{4.50}$$

An equivalent simplification can be developed for the “adaptation gains”. The speech “adaptation gain” is given by:

$$L_s(t) = \frac{1}{\eta_s(t)} P_s(t)\ \widehat{s_{p-1}(t-1)} \tag{4.51}$$
$$= \frac{1}{\eta_s(t)-1}\Bigg[P_s(t-1)\ \widehat{s_{p-1}(t-1)} -$$

$$\left. \frac{P_s(t-1)\ \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T}{\eta_s(t)-1+\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)\ \widehat{s_{p-1}(t-1)}} \times P_s(t-1)\ \widehat{s_{p-1}(t-1)}\right]$$

$$= \frac{1}{\eta_s(t)-1}\left[1-\frac{P_s(t-1)\ \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T}{\eta_s(t)-1+\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)\ \widehat{s_{p-1}(t-1)}}\right]P_s(t-1)\ \widehat{s_{p-1}(t-1)}$$

$$= \frac{1}{\eta_s(t)-1+\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)\ \widehat{s_{p-1}(t-1)}} \times P_s(t-1)\ \widehat{s_{p-1}(t-1)}$$

The same can be applied to the noise signal giving:

$$L_v(t) = \frac{1}{\eta_v(t)}P_v(t)\ \widehat{v_{q-1}(t-1)} \tag{4.52}$$

$$= \frac{1}{\eta_v(t)-1+\widehat{v_{q-1}(t-1)}^T\ P_v(t-1)\ \widehat{v_{q-1}(t-1)}} \times P_v(t-1)\ \widehat{v_{q-1}(t-1)}$$

Summarizing the above equations,

for the speech signal:

$$\hat{\alpha}(t) = \hat{\alpha}(t-1) + L_s(t)\ \widehat{e_s(t)} \tag{4.53}$$

$$\widehat{e_s(t)} = \widehat{s(t)} - \widehat{s_{p-1}(t-1)}^T\ \hat{\alpha}(t-1) \tag{4.54}$$

$$L_s(t) = \frac{P_s(t-1)\ \widehat{s_{p-1}(t-1)}}{\eta_s(t)-1+\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)\ \widehat{s_{p-1}(t-1)}} \tag{4.55}$$

$$P_s(t) = \frac{\eta_s(t)-1}{\eta_s(t)} \times \left[P_s(t-1) - \frac{P_s(t-1)\ \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T\ P_s(t-1)}{\eta_s(t)-1+\mathrm{trace}\{P_s(t-1)\ \widehat{s_{p-1}(t-1)}\widehat{s_{p-1}(t-1)}^T\}}\right] \tag{4.56}$$

and for the noise signal:

$$\hat{\beta}(t) = \hat{\beta}(t-1) + L_v(t)\ \widehat{e_v(t)} \tag{4.57}$$

$$\widehat{e_v(t)} = \widehat{v(t)} - \widehat{v_{q-1}(t-1)}^T \hat{\beta}(t-1) \tag{4.58}$$

$$L_v(t) = \frac{P_v(t-1)\,\widehat{v_{q-1}(t-1)}}{\eta_v(t) - 1 + \widehat{v_{q-1}(t-1)}^T P_v(t-1)\,\widehat{v_{q-1}(t-1)}} \tag{4.59}$$

$$\begin{aligned} P_v(t) = {} & \frac{\eta_v(t)-1}{\eta_v(t)} \times \\ & \left[P_v(t-1) - \frac{P_v(t-1)\,\widehat{v_{q-1}(t-1)}\widehat{v_{q-1}(t-1)}^T P_v(t-1)}{\eta_v(t) - 1 + \text{trace}\{P_v(t-1)\,\widehat{v_{q-1}(t-1)}\widehat{v_{q-1}(t-1)}^T\}} \right] \end{aligned} \tag{4.60}$$

We can, also, write a recursion for the residual series energy:

$$\begin{aligned} \hat{g}_s^2(t) = \sum_{\tau=1}^{t} \lambda_s^{t-\tau}\,\widehat{e_s^2(\tau)} = & \\ \left(1 - \frac{1}{\eta_s(t)}\right) \hat{g}_s^2(t-1) + \frac{1}{\eta_s(t)}\,\widehat{e_s^2(t)} & \end{aligned} \tag{4.61}$$

$$\begin{aligned} \hat{g}_v^2(t) = \sum_{\tau=1}^{t} \lambda_v^{t-\tau}\,\widehat{e_v^2(\tau)} = & \\ \left(1 - \frac{1}{\eta_v(t)}\right) \hat{g}_v^2(t-1) + \frac{1}{\eta_v(t)}\,\widehat{e_v^2(t)} & \end{aligned} \tag{4.62}$$

where $\widehat{e_s(t)}$ and $\widehat{e_v(t)}$ were defined above.

Note that the approximation made in 4.26 for the correlation matrix $\widehat{x(t)x^T(t)}$ is not very good one, especially for the signal which suffers from a low SNR level. This simplification is thus recommended (for computational efficiency) only when the Kalman covariance matrix is negligible (i.e. for the stronger signal). For high SNR value, it should be used for the speech signal, and for low SNR value - for the noise signal.

4.2.3 Recursion with LMS

The solution for the Yule-Walker equations 4.6, 4.7, 4.8, 4.9 can be found via a gradient search, that is directed to minimize the pre-specified cost function. This procedure is applied to each of the equations yielding the following updating equations:

$$\hat{\alpha}(t) = \hat{\alpha}(t-1) - \mu_s(t)\left[Q_{21}^s(t) + Q_{22}^s(t)\hat{\alpha}(t)\right] \quad (4.63)$$

$$\hat{\beta}(t) = \hat{\beta}(t-1) - \mu_v(t)\left[Q_{21}^v(t) + Q_{22}^v(t)\hat{\beta}(t)\right] \quad (4.64)$$

$$\hat{g}_s(t) = \hat{g}_s(t-1) - \mu_s(t)\left[\hat{g}_s(t) - \frac{1-\lambda_s}{1-\lambda_s^T}[Q_{11}^s(t) + Q_{12}^s(t)\hat{\alpha}(t)]\right] \quad (4.65)$$

$$\hat{g}_v(t) = \hat{g}_v(t-1) - \mu_v(t)\left[\hat{g}_{v(t)} - \frac{1-\lambda_v}{1-\lambda_v^T}[Q_{11}^v(t) + Q_{12}^v(t)\hat{\beta}(t)]\right] \quad (4.66)$$

Where μ_s and μ_v are step-size for the speech and noise gradient search, respectively, and $Q^s(t)$, $Q^v(t)$ are defined in equations 4.17, 4.18.

This procedure is very computationally efficient, but may suffer from larger mistakes. We did not used this approach in this work.

4.2.4 Recursion Summary

We developed three algorithms implementing a recursive solution for the Yule-Walker equations. As was stated by Ljung [22], all those recursive algorithm have the same structure:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + G(t)\epsilon(t) \quad (4.67)$$

An appropriate choice of the gain and direction matrix $G(t)$ and the error term $\epsilon(t)$, will give all the above equations. Ljung also suggested the application of a Kalman filter for the parameters for the case we have some a-priori information about the dynamics of the parameters. We do not use the Kalman Filter parameter estimation approach in this work.

Chapter 5

Evaluation of Speech Systems

Evaluation of the performance of speech systems is a difficult task, which depends strongly on the application. This problem was addressed widely in the literature (e.g. [23] [31] [16] [21]). The evaluation techniques involve subjective and objective tests. Among speech systems assessment categories are calculations of distortion measures, performance improvement of Automatic Speech Systems, and intelligibility scores achieved by human listeners.

This chapter gives a brief overview of the existing methods for speech systems evaluation.

5.1 Distortion measures

The problem of providing an objective measure of the improvement obtained by the speech enhancement system is as old as the research in the field. Finding an objective test that can predict the human opinion is a very difficult task. Several attempts has been made.

SNR improvement Computationally, this is the simplest test, but the most unreliable one. Let, $s(t)$, $z(t)$, $\hat{s}(t)$ be the clean, corrupted and enhanced speech signal, respectively, and T the sample size.

Define by:

$$\mathrm{SNR}_{in} = 10 \log_{10} \frac{\sum_{t=1}^{T} s^2(t)}{\sum_{t=1}^{T} [s(t) - z(t)]^2} \tag{5.1}$$

$$\text{SNR}_{out} = 10\log_{10}\frac{\sum_{t=1}^{T} s^2(t)}{\sum_{t=1}^{T}[s(t)-\hat{s}(t)]^2} \tag{5.2}$$

the SNR levels in the input and in the output of the evaluated Enhancer. Define by the difference:

$$\text{G} = \text{SNR}_{out} - \text{SNR}_{in} \tag{5.3}$$

the SNR improvement achieved by the enhancement algorithm (in dB).

It is important to emphasize that the improvement in SNR generally does not translate into improvement in speech quality and/or intelligibility.

Segmental SNR improvement A modification for the above test. In this method the speech signal is divided into segments, in each of which the speech is assumed to be more or less stationary. SNR is calculated for each segment, and then averaged over all the segments.

Thus:

$$\text{SEGSNR}_{in} = \frac{1}{M}\sum_{m=0}^{M-1} 10\log_{10}\left(\frac{\sum_{t=mK}^{(m+1)K-1} s^2(t)}{\sum_{t=mK}^{(m+1)K-1}[s(t)-z(t)]^2}\right) \tag{5.4}$$

$$\text{SEGSNR}_{out} = \frac{1}{M}\sum_{m=0}^{M-1} 10\log_{10}\left(\frac{\sum_{t=mK}^{(m+1)K-1} s^2(t)}{\sum_{t=mK}^{(m+1)K-1}[s(t)-\hat{s}(t)]^2}\right) \tag{5.5}$$

are the segmental SNR levels in the input and in the output of the evaluated Enhancer. K is the segment length and M is the number of segments in the tested sentences.

Once again, the difference defined by

$$\text{GSEG} = \text{SEGSNR}_{out} - \text{SEGSNR}_{in} \tag{5.6}$$

is the SEGSNR improvement achieved by the algorithm (in dB).

The value obtained by this way is more precise, since it is taking into account the non-stationarity of the speech signal.

Frequency Bands SNR A more meaningful test is the Speech Communication Index Meter (SCIM). In this test the signal is divided into frequency bands, in each of them SNR value is calculated. An auditory masking corrections is applied to those frequency bands to yield a representative figure.

Itakura-Saito distortion measure The Itakura-Saito distortion measure [9] is closely related to the "spectral matching" property of the LPC analysis. The success of LPC based vocoders suggests that the Itakura-Saito distortion measure is also subjectively meaningful distortion measure.

The Itakura-Saito distortion measure is defined by:

$$d_{IS} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{S(\omega)}{\hat{S}(\omega)} d\omega - \ln(g_s/\hat{g}_s) - 1 \tag{5.7}$$

where $S(\omega)$ and $\hat{S}(\omega)$ are the clean and processed speech spectra, respectively, and g_s, $\hat{g}_s$ are their relative gains.

The estimated spectrum of the speech is taken to be the AR spectral estimate:

$$\hat{S}(\omega) = \frac{\hat{g}_s}{\sum_{k=0}^{p} \hat{\alpha}_k \exp(-j\omega)} \tag{5.8}$$

5.2 Human Intelligibility

Subjective intelligibility tests can be categorized by the speech items tested and by the procedure used. The items can be phonemes, nonsense words, meaningful words, and sentences. A frequently used test to determine phoneme scores is the *rhyme* test. A rhyme test is a forced choice test in which the listener, after each word that is presented, has to select his response from a small group of visually presented alternatives. The alternatives only differ with respect to the phoneme at one particular position in the test word. A rhyme test is easy to apply and does not require much training of listeners. Frequently used rhyme tests are the Modified Rhyme Test (MRT, testing consonants and vowels), and the Diagnostic Rhyme Test (DRT, testing initial consonants only). The DRT is based on only two alternatives, which are based on testing single articulatory features. Studies have shown that the

DRT, because of the limited number of alternatives, is less sensitive and may force listeners to respond differently from their perceptual impression.

A more general approach is obtained by a test with an open response. Open response tests make use of short nonsense or meaningful words with a pre-defined consonants-vowels structure. The listener can respond with any combination of phonemes corresponding to the type of word defined beforehand. This procedure requires extensive training of the listeners.

Sentence intelligibility is measured by asking the listeners to estimate the percentage of words correctly heard on 0-100% scale. Sentence intelligibility saturates to 100% even at poor SNR conditions, so its effective range is small.

For example, in numbers test (which is the sentences we used for our intelligibility test), the effective SNR range is -9dB to -6dB for noise with a spectrum shaped according the long-term speech spectrum. Due to the high slope of the graph in this range this test may suffer from large deviation between listeners.

Another test is the Speech Reception Threshold (SRT) in which the listener has to recognize a word or sentence presented at a fixed level and masked by noise at a variable level. The noise level - where a 50% correct identification is achieved of words or sentence - is the SRT score. This test gives very good reproducible results.

5.3 Speech Quality

Speech quality may be determined by questionnaires or scaling methods, using one or more subjective scales, such as *overall impression, naturalness, noisiness, clarity.* Speech quality is normally used for sentences with high intelligibility. Quality rating is a more general method then the Intelligibility test, and it is used to evaluate the user's acceptance of the speech system. Some investigators claim that the quality rating reflects the total auditory impression of speech by the listener. The impression is frequently scaled into five discriminating levels: *bad, poor, fair, good, excellent.* Other types of scales are used, including: intelligibility, quality, acceptability, naturalness etc. For quality ratings, normal tests sentences or free conversation are used to obtain the listener's impression. The quality ratings is generally called Mean Opinion Score (MOS). The MOS does not give an absolute measure since the scales are not calibrated. Therefore it can be used only for rank ordering, or while using a reference conditions as an anchor.

We used a more loosely taken test, in which our listeners had to rank their impression between the corrupted speech and the enhanced speech, and sometimes to discriminate between several algorithms outputs.

5.4 Automatic Speech Recognition (ASR) performance

We have already mentioned that the human listening capability is very hard to test. Machines are easier clients. It is a well-known fact that the performance of ASR systems degrades rapidly, with increasing noise level. When the ASR is trained in different conditions then it is tested on (e.g. trained on noiseless speech data and tested on corrupted speech), the performance can be very poor. It is not a practical task to train the ASR for each working environments, so the problem should be solved differently.

One approach is to use a speech enhancement algorithm as a pre-processor for the recognizer. Another approach is to try to find better features or better distortion measures that is more robust to the working environment than the the original one.

Anyway, the algorithm performance can be calculated by the increase in the percentage of correct recognition after applying the algorithm, or by the amount of noise immunity gained by the algorithm (this point will be further discussed in chapter 6).

It should be noted that the performance is highly depended on the kind of distortion applied to the speech and the spectral shape of the noise added. So, when comparing several speech enhancement algorithms only relative performance could be discussed.

5.5 Comments

Some final comments. Another issue to be tested is the amount of listener fatigue. After a long period of listening to highly degraded speech the performance of the human listener is highly damaged. One possible benefit of speech enhancement algorithm, could be a reduction in this fatigue, even though the quality of the resulting speech is little worse.

The evaluation of the speech enhancement algorithms depends, also on some other factors.

The Listener The listener environment and experience have a big influence. Visual reception accompanying the hearing can help a lot. Good knowledge of the language is preferred.

The Task The performance depends strongly on the task performed. Detection of existence of speech is easier task than the understanding of the words. Performance is always better when the listener has a limited number of alternatives.

The speech signal Speech level, speech to noise ratio and linguistic or context effects have a lot of influence.

The Noise Masker The masking noise characteristics, especially the noise spectrum, is of great importance too.

So, to summarize, we can say that the comparison between algorithms can be done only under the same conditions.

Chapter 6

Experiments: Methodology and Results

6.1 Introduction

In this chapter we summarize the objective and subjective tests conducted in order to evaluate performance of the proposed algorithms and to compare them with two other widely used algorithms, namely, the **Spectral Subtraction** [2] and **Widrow's** algorithm [34]. Although those algorithms are not a state of the art algorithms, they have gained a lot of industrial interest, thus serving us as a good reference for classical methods both in frequency and in time domains.

The proposed **iterative-batch** and **sequential** algorithms and the **LMS** and **spectral subtraction** algorithms were implemented in MATLAB (with crucial parts in "C"). A Graphical User Interface (GUI) (see Appendix E) was implemented to control the software and to check the influence of the various parameters easily.

The influence of the various parameters of the algorithm on the subjective and objective tests will be discussed.

Our "experiments" were done on a **SPĄRC** station. A faster implementation, using a DSP device (**MOTOROLA 96000**), is under development now-days.

The subjective hearing tests were conducted via a high-quality audio system connected to our workstation. The objective tests were performed with an Automatic Speech Recognition (ASR) system developed in MIT university.

6.2 Experiment setup

In all our experiments we added a clean speech signal to a noise signal in several Signal-to-Noise-Ratios (SNR). The algorithms worked on the corrupted speech signals. The various parameters were changed via the GUI, and their influence has been checked.

6.2.1 Tests Scenario

Each test is defined by the speech and noise signals involved, the SNR conditions and the value of the the parameters of the algorithms.

Signal-To-Noise Ratio

We evaluate the input SNR level as in equation 5.1. This quantity gives us only a vague estimate of the speech quality, but it can be used as reference figure for discussion. A wide range of SNR values was taken, depending on the application:

Low level The range between -14dB and -10dB. This range was used only for intelligibility tests.

Mid level The range between -10dB and +5dB. Used for quality tests and ASR performance evaluation.

High level The range between +5dB and above. Used for ASR performance evaluation. Also used to check the degradation of the algorithm (if any) while reducing very low level noise.

Speech signal

In order to make a comprehensive test we need a large variety and a large quantity of speech signals. We used three groups of speech sentences. A lot more examples will be achieved after the conclusion of the DSP hardware implementation.

Free spoken sentences Long paragraphs of speech sentences recorded from the radio, both in Hebrew and English, and spoken by both male and female.

TIMIT database The speech sentences used for the ASR experiment was drawn from a standard data base known as TIMIT [25], which was produced jointly by MIT, SRI International, and Texas Instruments. The sentences comprising the TIMIT speech corpus were designed specifically to aid in the development and evaluation of phonetically-based ASR systems. The TIMIT utterances were recorded under very favorable acoustic conditions, and are therefore virtually free of distortion. Each speaker who participated in the recordings was placed in an anechoic room, equipped with a close-talking, noise-cancelling, headset-boom microphone, and instructed to read aloud, in a natural conversational voice, a sequence of preselected sentences. The spoken sentences were stored in digital form at a sampling rate of 16kHz, with each sample quantized to 16 bits. A total of 630 speakers participated in the recordings, each contributing ten sentences. Each speaker was associated with one of eight major dialect categories in American English. The text material in TIMIT corpus consists of three kinds of specially designed speaker prompts, identified in the data base as SA, SX, and SI sentences. We used SX sentences, which are phonetically compact sentences, designed to provide efficient and thorough coverage of phone pairs considered to be of particular interest for recognition problem. We used 10 sentences out of a total of 450 SX sentences. Each sentence we selected, was read by a different speaker (five male and five female).

Each utterance in the TIMIT data base is associated with four descriptive files: (1) a *waveform file*, which contains the digitized samples of recorded speech, (2) an *orthographic transcription file*, which contains the precise text of the spoken sentence, (3) a *word transcription file*, which contains a segmentation of the utterance into its component words, with beginning and ending waveform sample numbers provided for each word, and (4) a *phonetic transcription file*, which contains a segmentation of the utterance into its component phones, with beginning and ending waveform sample numbers provided for each phone.

Digits The ten English digits were recorded in our Laboratory. A series of digits drawn randomly from the those recordings were created. This separated digits data sets was used for intelligibility tests.

Noise signal

The algorithms performance was checked with three kinds of noise signals. Such variety is needed for a good evaluation of the algorithm over a wide range of noise signals. Every noise signal was multiplied by a gain factor to give the desired SNR value. “White” and “Colored” noise signals were used to check the benefit of using the more complicated “Colored” Kalman filter. Other noise signals were also used for checking the behavior of the algorithms on speech corrupted by non-stationary noise.

Artificial “white” noise We used a computer-generated Gaussian “white” noise. Although “White” noise is not physical, it is common in many signal processing application. We included it in our experiments because it is a good approximation in several important cases. This noise is, of course, stationary.

Artificial “colored” noise We created an artificial “colored” noise by passing a “white” Gaussian noise signal through an appropriate transfer function. We used an AR filter to color the noise. The actual shaping was left as an external parameter. In all experiments we have chosen LPF shaped spectrum, quite close in its frequency contents to the average spectrum of the speech. This noise is also stationary.

Actual noise The actual noise was recorded from a typical office environment: a computer fan. We have made spectral and statistical analysis to those recordings.

The short-term spectrum of this noise can be modeled as an AR process of order $q = 4$. The parameters (especially, the gain) are changing slowly across time, which make the noise non-stationary. Nevertheless, it can be viewed as quasi-stationary with a slower changing rate then the speech signal changing rate. The short-term spectrum is drawn in figure 6.1. Some statistical tests were also undertaken to analyze the noise p.d.f.. It was found to be close to Gaussian as shown in the “Normal plot” in figure 6.2. In “Normal”

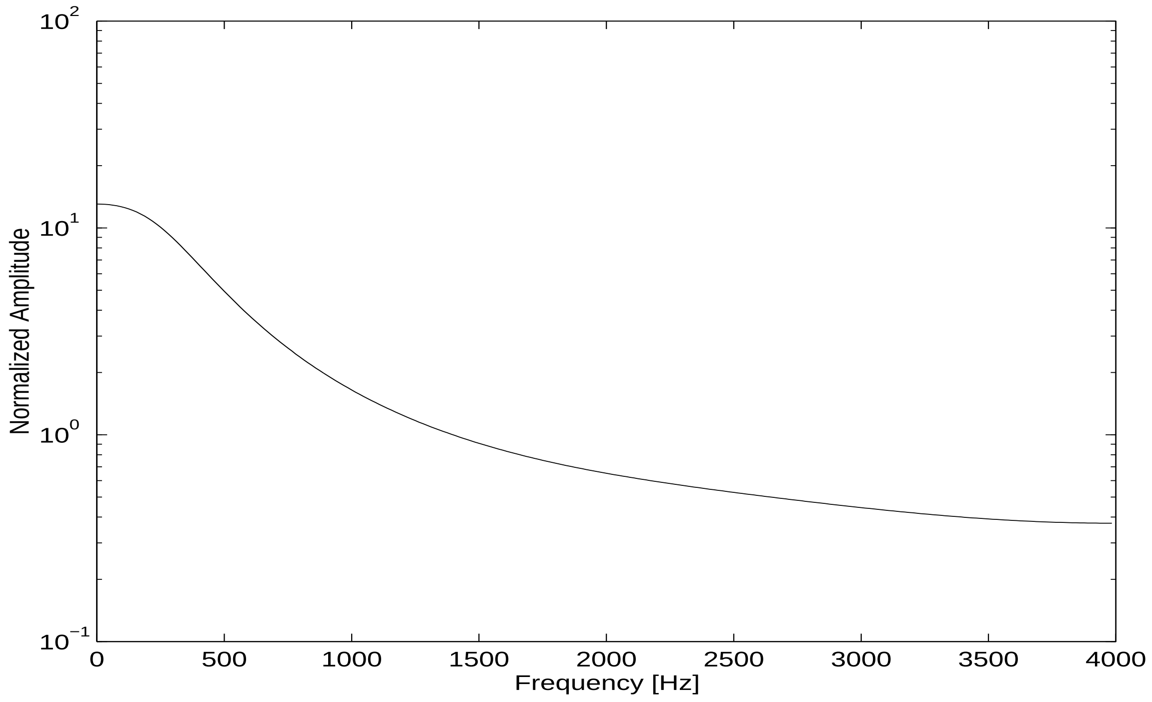


Figure 6.1: Short-term spectral envelope of actual noise segment, modeled as an AR(4) process

plot the empirical c.d.f.of the data is drawn, while the Y-axis of the graph is divided according to the Gaussian c.d.f.. A straight line indicates "Normal" p.d.f.. Forth Cumulant level was also computed and observed to be quite close to zero. This noise source is used to demonstrate the algorithm's ability to enhance speech corrupted by Gaussian noise.

6.2.2 Tests Performed

Each algorithm was evaluated by several tests, as shown in table 6.1.

Subjective tests

We have not used a formal **quality** tests, such as **MOS**. Instead, we used some informal listening tests. Ten listeners were given about 40 free spoken sentences both in Hebrew and English, in mid-level SNR range and corrupted by the three kinds of noise signals. Every one of the listeners was to tell his opinion about the

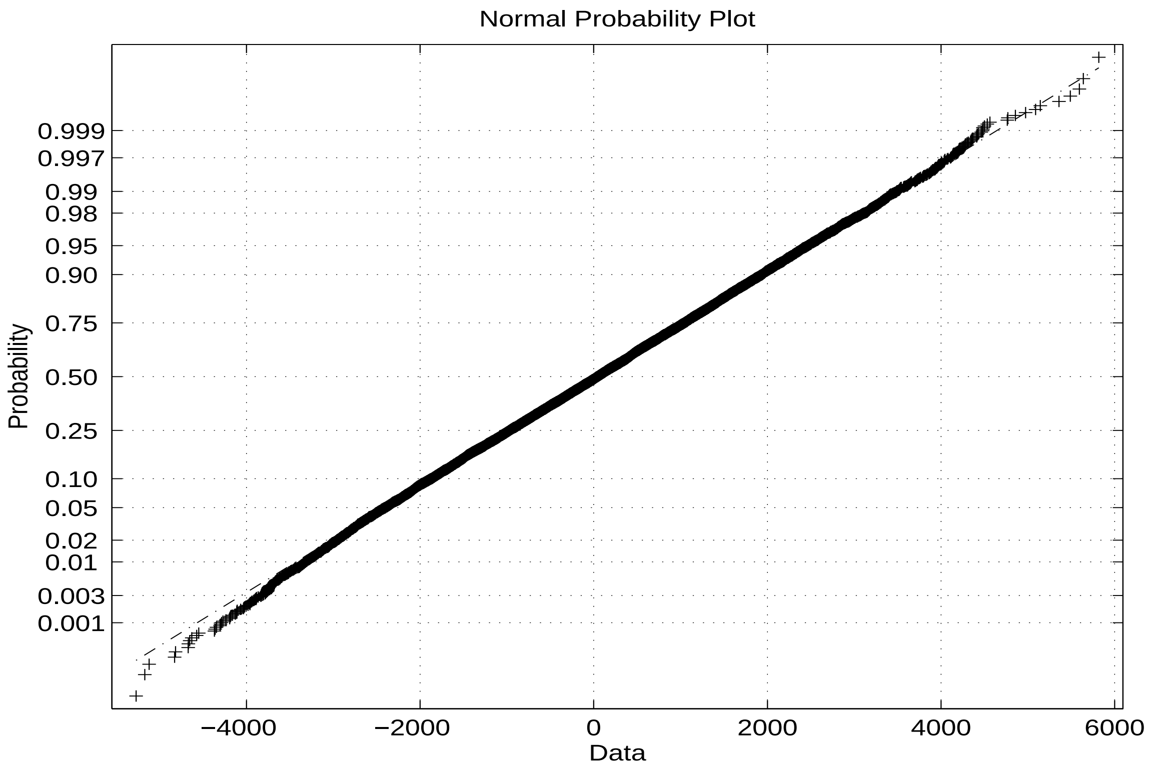


Figure 6.2: “Normal” plot of actual noise segment

quality of the different samples and to make a comparison between the corrupted speech samples and the enhanced samples. By **quality** the listeners addressed a various general aspects of the speech pleasantness and the level of the noise. They should pointed out what version of the sentences is preferred by them: the corrupted or the enhanced. It is worthwhile reminding that most algorithms, although reducing the noise level, distorts the speech. For that reason it is a difficult test to conduct.

	Quality	Intelligibility	SNR	ASR
Iterative-Batch	$\surd$	$\surd$	$\surd$	$\surd$
Sequential	$\surd$	$\surd$	$\surd$	
LMS	$\surd$		$\surd$	
Spectral-Subtraction	$\surd$		$\surd$	

Table 6.1: Tests conducted for each Algorithm

Objective tests

Those tests include **SNR** and **Segmental SNR** measurements, **Intelligibility** test, and **ASR** performance.

SNR and Segmental SNR measurements Although the SNR improvement has a limited meaning in speech processing, we used this figure to indicate an overall score. A more meaningful quantity is the SNR at each segment, that was also calculated.

Intelligibility tests Our listeners were given a highly corrupted (Low-level SNR values) speech sentences, and was requested to write down what they hear. This test used, especially, the separated digits sentences, but other sentences were used also. The percentage of correctly found words was the score given to the utterances.

Automatic Speech Recognition performance In this section we will present the experiments conducted with an Automatic Speech Recognition (ASR) system, developed by Victor Zue and other researchers from the Spoken Language Systems Group at the MIT Laboratory for Computer Science. We will not describe here the ASR system. The detailed description can be found in [26] [27] [32]. In our experiment the recognizer was configured to perform phone classification, while the exact phone borders were given a-priori.

We fed the recognizer with TIMIT database sentences corrupted by the (real-life) fan noise at mid and high SNR conditions. We used the ten sentences spoken by both male and female.

The proposed **Iterative-batch** algorithm was then applied as a pre-processor for the ASR system. A comparison between the ASR performance on the corrupted signal and on the algorithm outputs (the usual filtered output and the fixed-lag-smoothed output) was made. The difference between SNR values giving the same phone accuracy level was denoted as the "Noise Immunity" caused by the algorithm.

	Iterative-Batch	Sequential
HOS	$\surd$	
"colored" Filter	$\surd$	$\surd$
Noise Estim.	$\surd$	$\surd$
Smoothing	$\surd$	$\surd$
Overlapping	$\surd$	
Pitch	$\surd$	
Post-Filter	$\surd$	
AR order	$\surd$	$\surd$

Table 6.2: Parameters controlling the proposed algorithms

We did not conduct all the experiments for each of the algorithm. Thus, Intelligibility test was not conducted for the LMS and Spectral-Subtraction algorithms, because they do not work in the low-SNR range. ASR performance was done only for the proposed Iterative-Batch algorithm, because the ASR system was available to us only for a short period. One version of the Sequential algorithm was checked in MIT [33]. Results concerning the ASR performance of the Spectral-Subtraction can be found in several works (e.g. [10]). The SNR distortion measures are meaningless for the LMS and Spectral-Subtraction, due to the distortion caused by them. Table 6.1 summarize the tests that were conducted for each algorithm.

6.2.3 Algorithm Parameters

As mentioned before, various parameters influence the performance of the proposed algorithms. In table 6.2 we summarize those parameters and their availability in each of proposed the algorithms. The influence of each parameter on the algorithms behavior is discussed in subsection 6.4.

6.3 Experimental results

In this section we will summarize the results obtained by our experiments. We will list the results of all the algorithms as achieved by the various tests. In order to prevent a too much detailed description, we did not include results at all the different values of the parameters. We will quote only the best results that can be achieved by each algorithm at each test, and in section 6.4 we will discuss some general notes on the influence of each parameter.

6.3.1 Subjective results

As we mentioned before we used only informal listening tests.

All our listeners indicated that the quality of the speech processed by the Iterative-Batch algorithm is better then the quality of the corrupted speech in all the interesting SNR conditions from -10 dB to $+15$ dB. The listeners indicated a large reduction of noise level without any severe distortion to the speech signal.

No noticeable difference in speech quality has been stated by our listeners between the Sequential algorithm and the Iterative-Batch algorithm in the mid and high SNR conditions. At the lower level the Iterative-Batch solution supersede the Sequential one.

A comparison between the filtered output and the fixed-lag smoothed output shows the advantage of the filtered version. The listeners indicates that the fixed-lag smoothed output sounds slightly muffled.

The LMS algorithm (in its one-microphone form) works very well with periodic noise signals, eliminating it almost completely. But, while working with the types of noise we used the algorithm performance is much worse. The resulting speech suffers from a "barrel effect", which causes the speech to sound muffled. The algorithm should not be used in noise levels below 5 dB (at the low and mid SNR range). The **Spectral Subtraction** algorithm exhibits a great reduction of noise level but generates an annoying "musical tone" effect. The enhanced speech is followed by a sum of tones with fast shifting frequencies. The algorithm collapses in SNR values below -5 dB. We should also remember that we did not apply noise spectrum evaluation in speech-free segments, but instead, used a-priori knowledge. Applying the noise spectrum evaluation can degrade the algorithm performance even further.

Due to very long processing period, listener fatigue was not checked. But due to the amount of noise reduction, we think it will help in this field also. After the completion of the hardware implementation this problem will be addressed.

6.3.2 Objective results

Intelligibility results

This experiment was conducted with the digits data base and with free spoken sentences at low level SNR range. Only the Iterative-Batch algorithm worked in these harsh conditions. The results depends on the listener. Few of them (the more experienced) could understand each word from the corrupted utterance. While hearing the free spoken sentences most of the listeners indicate an increase of the number of words correctly detected from around 10% to 40%-50%, and while conducting the easier closed vocabulary test (digits) they achieved an improvement from 70%-80% to 90% of the number of digits detected. This result is quite important, because most of the known algorithms fail to work even at in higher SNR range. So, the results demonstrates one of the unique properties of the proposed Iterative-Batch algorithm.

SNR results

SNR improvement of 7-10 dB was achieved for input SNR in the low and mid-level range by the Iterative-Batch algorithm. The same algorithm achieved an improvement of 3-4 dB in the high SNR range.

The Sequential algorithm performed more of the same, with slight degradation in the low SNR range. In both cases the fixed-lag smoothed output had a better performance (1-2 dB) in all SNR ranges.

The LMS and Spectral-Subtraction algorithm, although reducing the noise level, distort the speech, and thus causing the SNR improvement test to be meaningless.

Automatic Speech recognition performance

The ASR results achieved by the Iterative-Batch algorithm both by the filtered and smoothed versions are summarized in table 6.3 and than viewed in a graphic manner in fig 6.3.

The parameters controlling the algorithm were adjusted to yield the best possible enhancement, but more improvement may be achieved by fine tuning. Anyway the parameters values were:

1. "Colored Kalman filter" - *on*
2. Speech AR order - 16
3. Noise AR order - 4
4. Overlapping - *off*
5. Pitch usage - *off*
6. Speech parameters initialization - via fourth order cumulant
7. Noise parameters initialization - from speech free segment

SNR[dB]	percentage			
	original	corrupted	filtered	smoothed
-5	73	25	29	29
0	73	32	38	43
5	73	38	48	54
10	73	48	52	

Table 6.3: ASR performance of Iterative-Batch Algorithm

From fig 6.3 it is clear that the recognizer performed on the "filtered" speech better than on the corrupted speech, and that the "fixed-lag smoothed" speech supersedes them both. We should remember that the SNR improvement of the fixed lag smoother is better than that of the simple filtering, although the speech quality is worse. This implies that an important score for the ASR might be the SNR improvement, certainly, more important than for a human listener.

From the graph we can define and measure an ASR oriented SNR improvement, or **Noise Immunity**: the difference between input SNR levels of the corrupted speech and the estimated speech that yield the same phone classification accuracy. This quantity can be measured by drawing an horizontal line in some pre-defined phone classification accuracy and measuring the length (in dB) between the intersection of this line with the three graphs.

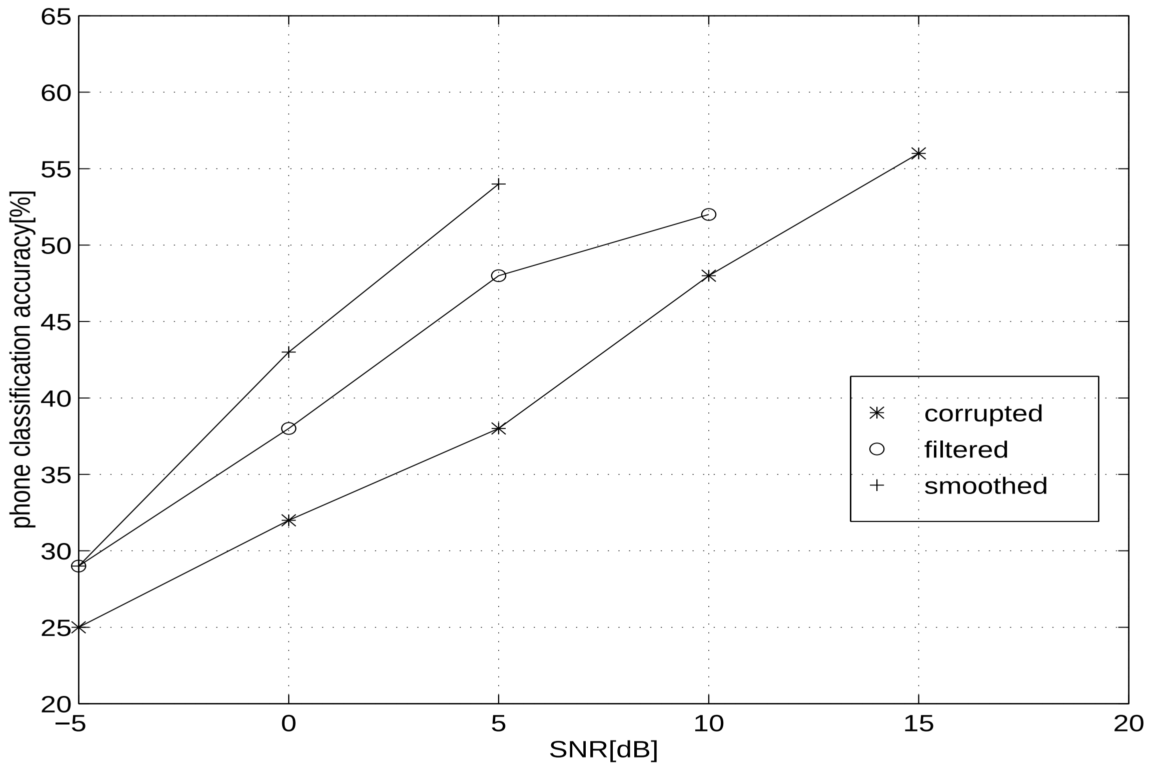


Figure 6.3: ASR performance of Iterative-Batch Algorithm

This improvement depends on the input SNR conditions, and on the added noise characteristics. For our experiment we can measure a $8 - 9$ dB improvement between the "smoothed" speech and the corrupted one for input SNR in the range of 0 to 10 dB, and $3 - 4$ dB for the lower range. For example, around 55% phone classification accuracy we have about 9 dB noise immunity achieved by the proposed Iterative-Batch algorithm.

We did not check word detection accuracy in this experiment. Since the graphs in the word case exhibit a threshold SNR level below it the performance degrades dramatically, the importance of the achieved improvement is emphasized. It can bring us above the threshold level.

The Sequential algorithm was not tested by us. The Sequential algorithm (implemented with gradient-search parameter update) was extensively tested by Verbout from MIT, and its performance can be seen in [33]. Although the tests were not conducted under the same conditions, our results seems to be slightly better. Results concerning the Spectral-Subtraction algorithm can be found in the literature, and they seems to be significantly inferior to the proposed algorithms.

6.4 Analysis of Experimental Results

Both the **Sequential** and **Iterative-Batch** algorithms are influenced by parameters that control their performance. In this section we will discuss this influence on the results presented in the previous sections of this chapter.

Sequential versus. Iterative-Batch Concerning quality the **Sequential** algorithm has equivalent performance to the **Iterative-Batch** algorithm at SNR range of -5 dB and above, but inferior at the lower range. We should remember that, the Sequential algorithm was developed as an approximation to the Iterative-Batch algorithm. So, this approximation is a good one in the discussed range, and the efficiency of the **Sequential** algorithm makes it preferable. Only when dealing with lower SNR values (which can occur in several important cases), we should choose the **Iterative-Batch** algorithm.

Using Higher Order Statistics We introduced in chapter 3 the use of improved techniques for estimating the speech parameters. The use of fourth-order-cumulant based equations has proven to supersede significantly the use of third-order-cumulant based equations. Both of them are superior to the conventional second-order-cumulant equations as well as the Modified Yule-Walker equations (which can only be applied in the "white" noise case). Note that this advantage exists, although the assumption, that the HOS of Gaussian noise diminishes, is not completely correct. Due to the finite segment length, even computer generated Gaussian noise has no zero Higher-Order-Cumulant.

We apply HOS based equations for the speech parameter estimation only in the first iteration (initialization step). This iteration is making the first (and good) separation between the speech and the Gaussian noise. After the first separation, the algorithm can converge to a correct solution, using 4-5 more conventional iterations, that separate the speech from noise but retaining the speech natural sounding. On the contrary, if we used only second order statistics for all the iterations (including the first), a fast converging algorithm (1-2 iteration) is resulted, but at low SNR range, the convergence point is inferior to the one achieved by the use of HOS. Trying to use HOS at other than the first iteration is causing the speech to sound unnatural. This effect is due to the reduced bandwidth of the formants, and it is supported also by other researchers (Palliwal & Sondhi [17], and Masgrau [5]). We suggest a possible explanation to this phenomena. As we mentioned, the second order statistics parameter estimation uses the covariance established from the Kalman filter in forming the Yule-Walker equations. Unfortunately, the Kalman filter is only optimal in MMSE (second-order sense). So, we do not have any error term to correct our cumulant-based equation in the same manner. Anyway, at this point we are using HOS only in the Iterative-Batch algorithm.

Estimating Noise Parameter In chapter 3 we introduced several approaches for the noise parameter estimation. In our experiments we skipped the use of speech activity detector due to the implementation difficulties. In the Iterative-Batch algorithm, at the high-level SNR range, the best results was obtained by initializing the parameters by values estimated at the beginning of the sentence, and by proceeding with "free" parameter estimation at the following iterations. This conclusion might be changed on the presence of highly non-stationary noise source. At low-level SNR range the best results was obtained by initializing the parameter estimate with values obtained by using the corrupted samples (which are, actually, "clean" noise samples), and proceeding as before. In each case, only second order statistics can be used for the noise. We assumed, in the algorithm development, that the noise is much more Gaussian than the speech (which is one of the main separating features we used). The Sequential algorithm is not iterating, so only the first iteration in the Iterative-Batch algorithm is performed.

"Colored Kalman filter" Using the "colored" version of Kalman filter, certainly improved the algorithm performance. The amount of SNR improvement is about $3-4$ dB (depending, of course, on the noise spectral shape). The slightly more complicated algorithm this use implies is not a large payment to pay. Even though knowing the exact noise parameters is preferred, estimating them together with the speech parameter yield a very good performance.

Fixed Lag Smoothing We stated before that we may use the Fixed-lag smoother instead of Kalman filter. As expected, an SNR improvement of $1 - 2$ dB is observed in the fixed lag smoothed version. The estimated speech quality as stated by our listeners is worse than the filtered version (they emphasized some "barrel" effect in the sound). On the contrary, the performance of the ASR system is improved significantly, as can be shown from figure 6.3. These contradicting results emphasizes the problem in evaluating a speech enhancement system. No "hard" decision can be done, and the answer for the question: "which algorithm is better?" - depends on the application.

Overlapping A common procedure in speech processing (especially, in **speech coders**) is to segment the speech into overlapping segments. By doing that, we can use long enough segments (for numerical reasons), but still compensate for the non-stationarity of the speech signal. Surprisingly enough, there is completely no difference between the original processing (without overlapping) and the suggested one. For that reason, we will, of course, skip this procedure.

Pitch Pitch seems like a very important information. Most of the energy of the speech signals comes from that periodic innovation. The proposed algorithm used the pitch in two ways: as a compensation value in estimating the speech LPC parameter, and as a "deterministic" input in the Kalman filter. Although promising, the results are doubtful. The pitch detector performance degrades quickly with decreasing SNR level, so the information we extract might be quite erroneous and can stop the algorithm convergence at all. we suggest to use the pitch information only on a very high SNR levels (in which we might not need it at all).

A further research should be done on that issue.

Post-Filter We suggest to use some simple post-filters after the algorithm. One simple choice is to apply a Band-Pass-Filter, which can "smoothen" the estimated speech and to filter out some residual noise caused by the algorithm. Use of the BPF causes a significant improvement in the quality of the **Iterative-batch** output. No improvement is observed in the **Sequential** algorithm, perhaps, because its "smooth" nature.

Another approach, that we suggested is to apply the two-microphone LMS algorithm with the corrupted signal in the primary input and the enhanced output in the reference input. Unfortunately, this approach proved to be a complete failure.

Noise and Speech AR orders The order of the AR models used in the Kalman filter is of course important. For the speech a common choice is $p = 10$ for 8 kHz sampled signal and $p = 12 - 14$ for 16 kHz sampled signal. we used those choices, and they proved to be good choices. The noise AR order was chosen in correspondence with the noise signal used. When the artificial noise degraded the speech, the noise AR order in Kalman filter equations was chosen to be similar to the signal AR order. When the actual noise was chosen, a choice of $q = 4$ in Kalman filter equations seems to be a good one.

6.4.1 Summary

In this chapter we introduced the experiments conducted and experimental results obtained by the proposed algorithms, and two other reference algorithms.

Both the **Spectral Subtraction** and the **LMS** algorithms are inferior to the proposed algorithms. Those algorithms covers only a proration of the SNR conditions that can be covered by the proposed algorithms. Even in the SNR values in which both the proposed algorithms and the reference algorithms work, the enhanced speech quality of the proposed algorithm is significantly better. One important reason for that is the distortion caused by the reference algorithms which is almost unheard in ours. Again, we will note that our algorithms works quite well in very low SNR level, lower then most algorithms do.

The proposed algorithms, especially the Iterative-Batch one, prove to work well in all the categories tested. The speech quality obtained, the detection accuracy rate of ASR system and the resulted SNR all demonstrate a significant improvement. Only very slight distortion effects are noticed. As indicated by several listeners,

		Quality	Intelligibility	SNR	ASR
Low SNR	Bat	good	+ 20-30 %	7-10 dB	
	Seq	quite good		4-6 dB	
	LMS	terrible			
	SSub	terrible			
Mid SNR	Bat	very good		7-10 dB	3-4 dB
	Seq	very good		7-10 dB	
	LMS	very Bad			
	SSub	Bad			
High SNR	Bat	Excellent		3-4 dB	8-9 dB
	Seq	Excellent		3-4 dB	
	LMS	very good			
	SSub	very good			

Table 6.4: Summary of Experimental Results

even Intelligibility tests indicates a noticeable improvement. A brief summary of the relevant results is shown in table 6.4.

Chapter 7

Conclusions and Topics for Further Research

In this work we developed, implemented evaluated and compared algorithms for speech enhancement. We dealt with the problem of enhancing speech degraded by additive noise and received with only one microphone.

The problem of speech enhancement has gained a lot of interest in the last three decades. For that reason many algorithms solving this problem have been suggested, but still with no complete success. Through out our research we tried to keep in mind the work of our predecessors. Our main concern is to enhance speech in a very harsh environment. In our development we tried to exploit as much information as we could on the speech and noise characteristics by giving quite general models for both signals. We used the well-known LPC model for the speech vocal tract, excited by a mixed innovation of both noise the pitch series. For the noise we used a different order LPC model, which can describe a large variety of actual noise signal. HOS techniques for AR parameter estimation is used to differ between the more Gaussian noise signal and the speech signal.

From the concept of EM we developed two families of algorithms, which have a quite intuitive structure. The algorithms iterates between decoupled estimation of speech and noise parameters estimation, and an optimal MMSE filter (the Kalman filter) application. The two families of algorithms derived from this concept are the **ITERATIVE-BATCH** algorithm, which divides the speech into segments, and performs several iterations on each segment, until convergence is achieved, and

SEQUENTIAL algorithm which replaces the iteration index by the time index to construct a completely recursive solution. Each of the algorithm uses several parameters that controls their behavior. The use of **HOS** and the **PITCH** series is only incorporated into the **ITERATIVE-BATCH** algorithm.

The algorithms were evaluated with both Objective and Subjective tests undertaken by Human listeners and by an ASR system.Comparison between the proposed algorithms and two widely used algorithms (Spectral Subtraction and LMS) were taken . Both the proposed algorithms work very well on a wider SNR range (-14 dB to 10 dB), suppressing noise almost without degrading the speech quality. At the extent of very low SNR conditions, we encountered an intelligibility improvement as indicated by several listeners. ASR system gained a noise immunity of about $8-10$ dB by using our algorithms as a preprocessor. The **ITERATIVE-BATCH** algorithm supersedes the **SEQUENTIAL** algorithm in the lower range of SNR values. The use of **HOS** improves the convergence behavior of the algorithm, and the resulting speech quality and intelligibility.

This research can be extended in several directions.

PITCH Although promising, the use of pitch information has doubtful results. Remember, that we have done several approximations in our implementation, to enable a reasonable processing time. After implementing the algorithms on a fast DSP processor, we suggest to try to give up our approximation and to implement the internal iterations for finding the pitch innovation. Other methods for pitch extraction may be incorporated as well.

CUMULANT We used in each iteration either Second-Order Statistics or HOS. Using a combination of several statistics weighted correctly may be considered, in order to get a more robust estimation of the AR coefficients.

Speaker separation Although, intended for speech enhancement, the structure of our algorithms implies that applying them for the speaker separation problem is worth checking. Remember that the speech and noise models are quite similar. By incorporating a pitch innovation sequence into the noise model, will result an identical speech and noise structure is resulted. The decoupling between the speech and noise parameter estimation equations can be exploited in the Competing Speaker Separation problem.

GOAL function We should remember that our algorithms are oriented for SNR improvement. As we noted before, SNR is not a representative quantity for speech perception. Perhaps a different goal function, that is more perceptually oriented (like the Itakura-Saito distance measure) can give a better improvement in speech quality and intelligibility.

ASR The algorithms proposed were applied to an ASR system as a preprocessor. Thus, the more robust parameters estimated were used only for enhancing the speech, but were not used directly in the ASR system. This two stage procedure is only sub-optimal. Perhaps, the use of the more robust parameters within the ASR system may result a combined system with better noise immunity properties.

Appendix A

The EM Algorithm

The Estimate-Maximize (EM) algorithm [?] is an iterative method for finding *Maximum Likelihood* (**ML**) parameter estimates. It works with the notation of "complete data", and iterates between estimating the log-likelihood of the complete data using the observed ("incomplete") data and the current parameter estimate (**E-step**), and maximizing the estimated log-likelihood function (**M-step**) to obtain the new parameter estimate.

More specifically, let **z** denote the observed data, with the probability density function (p.d.f.) $f_Z(z;\theta)$, indexed by the vector of unknown parameters $\theta \in \Theta \subseteq \Re^k$. The ML estimate $\hat{\theta}_{ML}$ of θ is defined by:

$$\hat{\theta}_{ML} = \arg\max_{\theta \in \Theta} \log f_Z(z;\theta) \tag{A.1}$$

Let **y** denote the complete data, related to the observed (incomplete) data **z** by

$$H(y) = z \tag{A.2}$$

where $H(\cdot)$ is a non-invertible (many-to-one) transformation. With $f_Y(y;\theta)$ denoting the p.d.f.of **y**, and $f_{Y|Z}(y|z;\theta)$ denoting the conditional p.d.f. of **y** given **z**,

$$f_Y(y;\theta) = f_Z(z;\theta) f_{Y|Z}(y|z;\theta) \qquad \forall H(y) = z \tag{A.3}$$

Equivalently, taking the logarithm:

$$\log f_Z(z;\theta) = \log f_Y(y;\theta) - \log f_{Y|Z}(y|z;\theta) \tag{A.4}$$

Taking the conditional expectation given $\mathbf{z}$ at a parameter value θ' (that is, multiply both sides of A.4 by $f_{Y|Z}(y|z;\theta')$ and integrating over $\mathbf{y}$,

$$\log f_Z(z;\theta) = E_{\theta'}\left\{\log f_Y(y;\theta)|z\right\} - E_{\theta'}\left\{\log f_{Y|Z}(y|z;\theta)|z\right\} \tag{A.5}$$

where $E_{\theta'}\left\{\cdot|z\right\}$ denotes the conditional expectation given $\mathbf{z}$ computed using the parameter value θ'.

For convenience we define

$$\begin{aligned} Q(\theta,\theta') &= E_{\theta'}\left\{\log f_Y(y;\theta)|z\right\} && \text{(A.6)} \\ P(\theta,\theta') &= E_{\theta'}\left\{\log f_{Y|Z}(y|z;\theta)|z\right\} && \text{(A.7)} \end{aligned}$$

So that equation A.5 becomes

$$\log f_Z(z;\theta) = Q(\theta,\theta') - P(\theta,\theta') \tag{A.8}$$

By Jensen's inequality [1]

$$P(\theta,\theta') \leq P(\theta',\theta') \tag{A.9}$$

Therefore,

$$Q(\theta,\theta') > Q(\theta',\theta') \quad \text{implies} \quad \log f_Z(z;\theta) > \log f_Z(z;\theta') \tag{A.10}$$

The relation in A.10 forms the basis for the EM algorithm. Denote by $\theta^{(l)}$ the estimate of θ after l iterations of the algorithm. Then, the next iteration cycle is specified in two steps as follows:

[1] Jensen's inequality asserts that for any pair of p.d.f.'s $f(x)$ and $g(x)$ defined over the probability space Ω of points x,

$$\int_\Omega f(x)\log g(x)dx \leq \int_\Omega f(x)\log f(x)dx$$

E-step Compute

$$Q(\theta, \theta^{(l)}) = E_{\theta^{(l)}} \{\log f_Y(y; \theta) | z\} \tag{A.11}$$

M-step

$$\max_{\theta} Q(\theta, \theta^{(l)}) \rightarrow \theta^{(l+1)} \tag{A.12}$$

If $Q(\theta, \theta')$ is continuous both in θ and θ', the algorithm converges to a stationary point of the observed log-likelihood $\log f_Z(z; \theta)$, where the maximization in A.12 ensures that each iteration cycle increases the likelihood of the estimated parameters. Specifically, as in all "hill climbing" algorithms, the stationary point may not be the global maximum, and thus several starting points or an initial grid search may be needed.

We note that the transformation $H(\cdot)$ is not uniquely defined. Specifically, there are many complete data specifications $\mathbf{y}$ that will generate an observed $\mathbf{z}$. The final point of convergence of the EM algorithm is essentially independent of the complete data specification. However, the choice of $\mathbf{y}$ strongly affect the rate of convergence of the algorithm, and the computations involved.

Appendix B

Higher Order Statistics

In this appendix we will define moments and cumulants and the relationship between them. We also will state some of their common used characteristics.

Definition B.1 (Moments) *the $(n_1+n_2+\ldots+n_N)$-th order cross-moment of the random variables $x_1, x_2, \ldots x_N$ is:*

$$M\left(x_1^{n_1} \cdot x_2^{n_2} \cdot \ldots x_N^{n_N}\right) = \frac{\partial^{(n_1+n_2+\ldots+n_N)}}{(\partial s_1)^{n_1} \cdot (\partial s_2)^{n_2} \cdot \ldots (\partial s_N)^{n_N}} E\left\{\exp(s_1x_1+s_2x_2+\ldots+s_Nx_N)\right\}\big|_{s_1=s_2=\ldots=s_N=0} \tag{B.1}$$

Definition B.2 (Cumulants) *the $(n_1+n_2+\ldots+n_N)$-th order cross-cumulant of the stochastic variables $x_1, x_2, \ldots x_N$ is:*

$$\text{cum}\left(x_1^{n_1} \cdot x_2^{n_2} \cdot \ldots x_N^{n_N}\right) = \frac{\partial^{(n_1+n_2+\ldots+n_N)}}{(\partial s_1)^{n_1} \cdot (\partial s_2)^{n_2} \cdot \ldots (\partial s_N)^{n_N}} \log\left(E\left\{\exp(s_1x_1+s_2x_2+\ldots+s_Nx_N)\right\}\right)\big|_{s_1=s_2=\ldots=s_N=0} \tag{B.2}$$

There is a close relationship between the cumulants and moments. Presume $M(x_1) = 0$.

First Order

$$\text{cum}(x_1) = M(x_1) = E\{x_1\} \tag{B.3}$$

Second Order

$$\text{cum}(x_1, x_2) = M(x_1, x_2) = E\{x_1x_2\} \tag{B.4}$$

Third Order

$$\text{cum}(x_1, x_2, x_3) = M(x_1, x_2, x_3) = E\{x_1x_2x_3\} \tag{B.5}$$

Fourth Order

$$\begin{aligned}\mathrm{cum}(x_1,x_2,x_3,x_4) &= M(x_1,x_2,x_3,x_4) \\ &\quad -M(x_1,x_2)\cdot M(x_3,x_4) \\ &\quad -M(x_1,x_3)\cdot M(x_2,x_4) \\ &\quad -M(x_1,x_4)\cdot M(x_2,x_3)\end{aligned} \tag{B.6}$$

The following properties of cumulants can be verified (see [3]):

Property B.1 (Linearity) *For any set of constants c_i,*

$$\mathrm{cum}(\ldots,\sum_i c_i x_i,\ldots) = \sum_i c_i \mathrm{cum}(\ldots,x_i,\ldots) \tag{B.7}$$

Property B.2 (Gussianity) *If $(x_1,x_2,\ldots x_N)$ are jointly Gaussian random variables then,*

$$\mathrm{cum}(x_1,x_2,\ldots x_N) = 0 \tag{B.8}$$

$\forall N \geq 3$.

Property B.3 (statistically independence) *If $(x_1,x_2,\ldots x_N)$ can be divided into two or more statistically independent subsets, then*

$$\mathrm{cum}(x_1,x_2,\ldots x_N) = 0 \tag{B.9}$$

Appendix C

Spectral Subtruction Method

In this appendix we will describe briefly the Spectral Subtraction method. All those methods can be summarized by the following general form suggested by Weiss *et al.* [24]:

Denote the corrupted speech as $z(t)$:

$$z(t) = s(t) + v(t) \tag{C.1}$$

where, $s(t)$ is the speech signal and $v(t)$ is the noise signal. Than, the frequency transform of the speech is reconstructed by:

$$\begin{aligned} |\hat{S}(\omega)| &= |Z(\omega)|^a - kE\{|V(\omega)|^a\} && \text{(C.2)} \\ \hat{S}(\omega) &= |\hat{S}(\omega)| \exp(\theta_z(\omega)) && \text{(C.3)} \end{aligned}$$

where $\exp(\theta_z(\omega))$ is the corrupted phase. It was noted [16], that this choice of the phase is the best that can be done under these circumstances. This use of the phase is justified by the unimportance of short-term phase [1].

The time domain speech signal is than given by:

$$\hat{s}(t) = \mathcal{F}^{-1}\{\hat{S}(\omega)\} \tag{C.4}$$

a and k are free parameters. k controls the amount of noise reduction. a controls the spectral weighting. The choice $a = 1$ and $k = 1$ gives Boll's algorithm [2]. The choice $a = 2$ and $k = 1$ gives the *power spectrum subtraction* technique, which is also referred as the *correlation subtraction* technique. The expectation operation is performed by averaging signal segments during non-speech activity just prior to its application in the subtraction operation. After this magnitude adjustment, a secondary procedures such as half-wave rectification and adjacent frames smoothing are applied.

Appendix D

Widrow-Hopf LMS Method

A schematic draw of the noise cancelation problem in the one-microphone case is drawn in figure D.1.

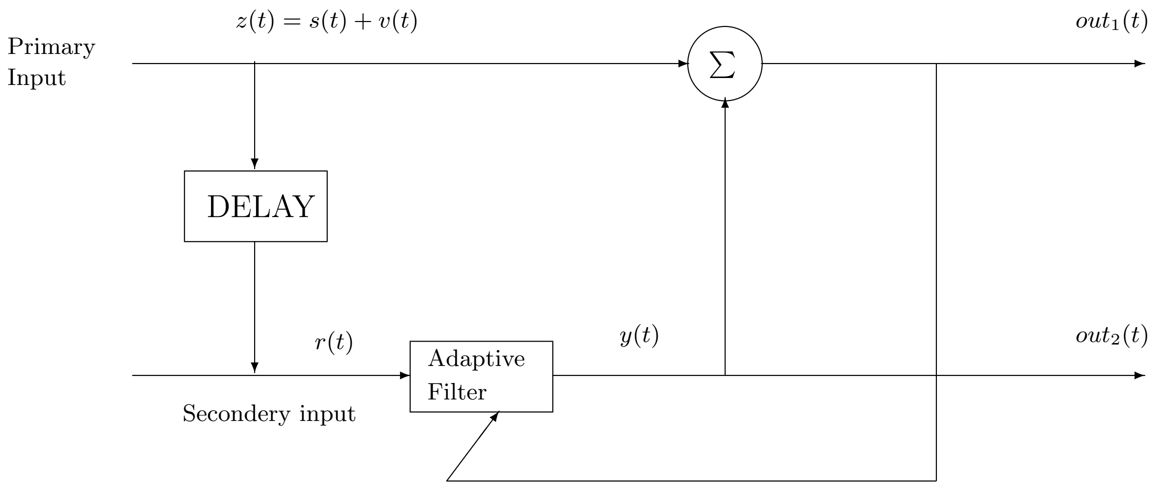


Figure D.1: The adaptive noise cancelling concept

Denote the primary microphone input as $z(t)$,

$$z(t) = s(t) + v(t) \tag{D.1}$$

Where $s(t)$ is the desired speech signal, and $v(t)$ is the disturbing noise signal. Denote the secondary microphone input by $r(t)$. In our case

$$r(t) = z(t - \mathcal{D}) \tag{D.2}$$

Where $\mathcal{D}$ is the amount of DELAY applied. This reference signal is filtered by the adaptive filter. Denote the filter output as $y(t)$. $y(t)$ is subtracted from the primary input. Denote the remaining residual error as $\epsilon(t)$:

$$\epsilon(t) = s(t) + v(t) - y(t) \tag{D.3}$$

$out_1(t)$ and $out_2(t)$ are the primary and secondary outputs of the algorithm, respectively.

$$out_1(t) \quad = \quad \epsilon(t) \tag{D.4}$$

$$out_2(t) \quad = \quad y(t) \tag{D.5}$$

If the filter output $y(t)$ is in correlation with the disturbing noise $v(t)$, $out_1(t)$ is a good approximation for the desired speech signal; and if the filter output $y(t)$ is correlated with the speech signal $s(t)$, $out_2(t)$ is the desired output. The desired correlations are achieved via a modification of the DELAY value. Thus, if the noise is white, a short delay will cause the noise correlation to diminish while the speech correlation still exist. In this case $out_2(t)$ will contain the desired speech. On the other hand, if we have a periodic corruption, a very long delay will do the job. In this case $out_1(t)$ will contain the desired speech. This use of the DELAY implies that only signals with different correlation length can be separated.

The adaptation mechanism involves minimization of the error signal. Our objective is, thus, to produce a system output, $out_1(t)$, that is the best Least-Squares fit to one of the primary input components, based on its reference, available in the secondary input. This objective is accomplished via adjusting the filter coefficient with the LMS algorithm. Minimizing the residual error energy is equivalent to minimizing the energy of the difference between the correlated signals.

Thus, in the case of white noise we apply:

$$\min E\{out_1^2(t)\} \quad = \quad E\{v^2(t)\} + \min E\{[s(t) - y(t)]^2\} \tag{D.6}$$

and in the case of periodic noise we apply:

$$\min E\{out_1^2(t)\} \quad = \quad E\{s^2(t)\} + \min E\{[v(t) - y(t)]^2\} \tag{D.7}$$

The solution for the minimization of the second term in Equations D.7, D.6 is the **Wiener-Hopf** set of equations. An adaptive solution to those equations is the basis of the Widrow's algorithm.

Define a vector of filter taps as:

$$W(t) = \begin{bmatrix} w_L(t) \\ w_{L-1}(t) \\ w_{L-2}(t) \\ \vdots \\ w_0(t) \end{bmatrix} \tag{D.8}$$

and a vector of reference input samples as:

$$X(t) = \begin{bmatrix} r(t-L) \\ r(t-L+1) \\ r(t-L+2) \\ \vdots \\ r(t) \end{bmatrix} \tag{D.9}$$

Than the **Widrow-Hopf LMS** algorithm is:

$$W(t+1) = W(t) + \mu\epsilon(t)X(t) \tag{D.10}$$

And the **Widrow-Hopf Normalized LMS** algorithm is:

$$W(t+1) = W(t) + \mu\epsilon(t)\frac{X(t)}{X^T(t)X(t)} \tag{D.11}$$

In the NLMS case the adaptation constant should satisfy:

$$0 \leq \mu \leq 1 \tag{D.12}$$

Appendix E

Graphical User Interface

E.1 General

The proposed algorithms are controlled by a "Graphical User Interface" (GUI) implemented by MATLAB software. All the external parameters can be changed via the GUI, to help the user in determining what is the influence of each of them on the algorithm performance. Several other utility software - such as file handling, or playing sound files via the the high quality sound player - can be also activated.

In this chapter we will demonstrate the GUI screens, and explain their various components. So, this chapter can be viewed as a fast manual for using the software.

The GUI is assembled by one permanent **MAIN** screen and one exchanging **ALGORITHM** screen. Several other windows can be opened for specific actions.

Before starting our detailed description we will explain what are the components of each GUI.

Push Buttons Clicking the mouse button on a push button causes MATLAB to perform a defined action.

Check Boxes Check Boxes let the user select one or more alternatives. Check Boxes act like toggle switches, indicating state of *on* or *off*. The state is *on* if the box is checked, and *off* if the box is not checked. Selecting a check box causes MATLAB to perform a defined action.

Radio Buttons Radio buttons let the user choose among mutually exclusive alternatives. Like check boxes, radio buttons act as toggle switches, indicating a state of either *on* or *off*. Selecting a Radio button causes MATLAB to perform a defined action.

Sliders Sliders let the user choose a value within a range of values. Sliders are analog devices which display their values graphically. The user can change the value by an indicator. Changing the value of a slider causes MATLAB to perform a defined action.

Pop-up Menus Pop-up menus let the user choose an item from a list. Choosing a pop-up menu item causes MATLAB to perform a defined action.

Editable Text Editable text controls let the user enter a string value to be used by the application. The user can accept, edit, delete, or replace an editable text value.

Pull-down menus Pull-down menus allow users to browse through and choose among options in an application. Menus consists of a menu bar, which displays the titles of available menus, and menu items. Menu items can be names of action, attributes, or windows.

E.2 Windows description

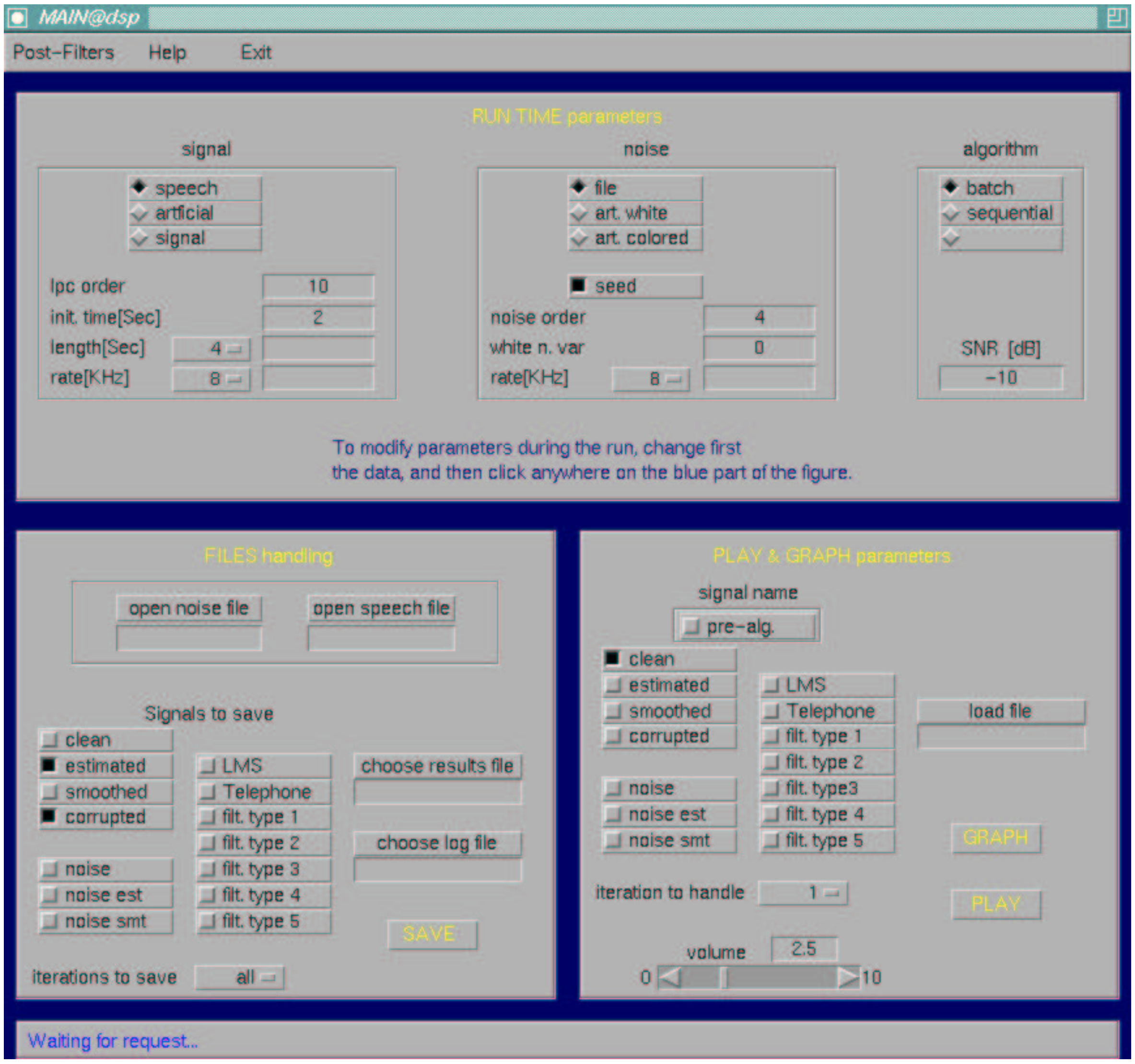
We will describe now each of the software windows and their action.

E.2.1 MAIN Screen

Figure E.1 demonstrates the main screen of the software control panel.

The screen is divided to four regions: **RUN TIME parameters**, **FILES handling**, **PLAY & GRAPH parameters**, and **Messages**. In addition there is a pull-down menus on the top of the screen.

RUN TIME parameters This region controls some general parameters concerning the signals involved and the algorithm performed. It is sub-divided into three sub-regions:



◆ speech

◆ file

■ seed

◆ batch

lpc order 10

init. time[Sec] 2

length[Sec] 4

rate[KHz] 8

noise order 4

white n. var 0

rate[KHz] 8

SNR [dB] −10

To modify parameters during the run, change first
the data, and then click anywhere on the blue part of the figure.

Signals to save: ■ estimated ■ corrupted

iterations to save all

■ clean

iteration to handle 1

volume 2.5

Waiting for request...

Figure E.1: GUI - MAIN Screen

signal Various parameters of the signal to be enhanced. A Radio Button selects between several signals available: speech, artificial AR process, and other signal (future use). If file is chosen the user should open a noise file (see FILES handling region). The LPC order, used by the Kalman filter will be chosen by the editable text field lpc order. From the file chosen, we can slice a segment for enhancing. The editable text field init time[Sec] will determine the initial time of the slice. the editable text and pop-up menu fields length[Sec] will determine the length of the slice. The speech sampling rate used through out the algorithm is determined via the pop-up menu and editable text fields rate[kHz].

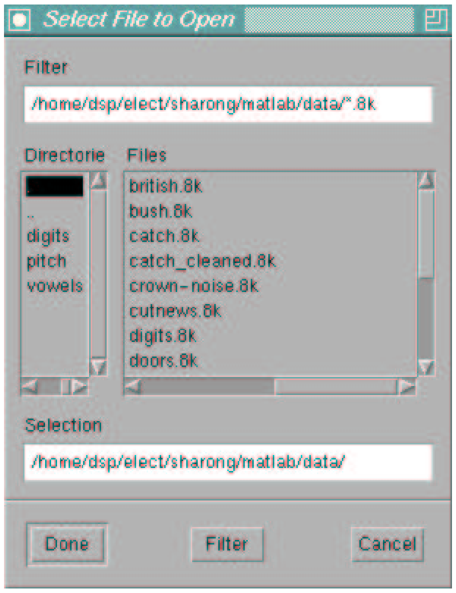
noise The specifications of the noise added to the signal are chosen via the noise region together with the **Noise Spectrum** window (See Noise Spectrum window section). Noise can be chosen to be one of three choices via a Radio button: a recorded file, an artificial Gaussian white noise, or an artificial Gaussian colored noise. If file is chosen the user should open a speech file (see FILES handling region). If colored noise is chosen, its AR order is defined by the editable text field noise order, and the AR coefficients values are determined via the Noise Spectrum window. The noise sampling rate used through out the algorithm is determined via the pop-up menu and editable text fields rate[kHz]. Additional white noise (that is used for stability reasons in the algorithm) can be added by determining its level via the editable text field white n var.

algorithm The algorithm used can be switched via the Radio button in the algorithm region. A choice between **BATCH** and **SEQUENTIAL** algorithms is currently available. The relative level of the speech and noise is determined by the editable text field SNR [dB].

FILES handling This region enables us to load speech and noise files (when necessary), and to save the results of the algorithm.

The window is divided into the following regions.

open noise/speech file The push buttons open noise file and open speech file



Filter
/home/dsp/elect/sharong/matlab/data/*.8k

Selection
/home/dsp/elect/sharong/matlab/data/

Figure E.2: GUI - Open File Screen

activates windows for file selection, as shown in figure E.2. The user can use the editable text fields instead.

choose log file All the messages appearing during the algorithm run (such as SNR value at each segment at each iteration) can be dumped into a log file. The file is selected via the choose log file push button in the usual file selection manner. The push button also enables the messages dump.

Signals to save Each of the signals produced by the algorithm can be saved in a result file. The file name is selected via the push button choose result file. The name of the signals to be saved is chosen via the group of check boxes. If several iterations exist for part of the signal, the iteration to be saved is selected via the pop-up menu iterations to save. After selecting the appropriate signals the actual saving is done by the push button SAVE.

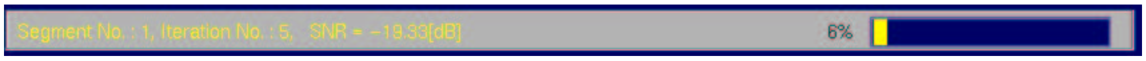
PLAY & GRAPH parameters This region enables to examine the results of the algorithm. It is possible to **hear** the signals or to **see** them. The push button PLAY activates our high quality DAT machine. The editable text field together with the slider volume are used for controlling the volume of the sound.

Activating the GRAPH push button open another window, which contains graphs of all the selected signals. Selecting a signal is done by the same manner as in the **FILES handling** region.

The signals can be either the results of the last run or signals loaded from a file by the load file push button.

The check box pre-alg starts a short run of the initialization part of the algorithm. After activating this button the user can check the corrupted signal before applying the algorithm.

Messages All the responses of the software to the actions of the user, and the intermediate results appear on the **Messages** region. The progress of the algorithm can be viewed graphically - by a bar - and mathematically - by percentage display. Both means appear in the Messages region (see figure E.3).



```
Segment No. : 1, Iteration No. : 5,   SNR = −19.33[dB]          6%
```

Figure E.3: GUI - Messages Region

pull-down menu A pull-down menu is located on the top bar of the MAIN screen. Several options are available.

Post-Filters The user can choose between several post-filter to apply to the results of the algorithm. “LMS” and “Telephone” are available. Choosing one of the choices activates an appropriate screen.

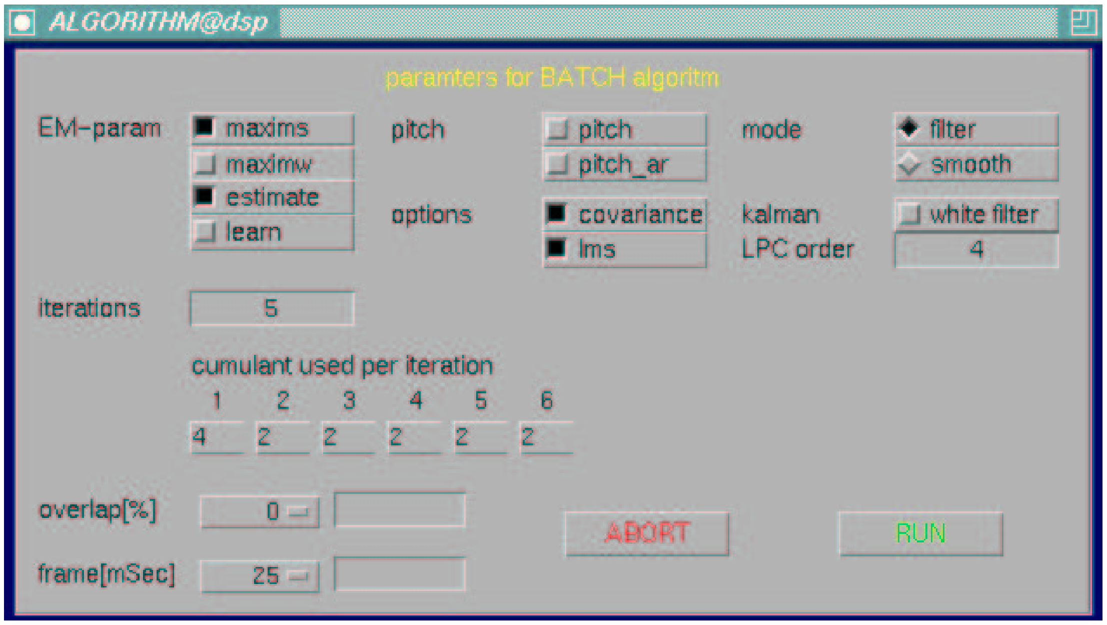
Help “Help” pages are not available yet.

Exit Enables the user to quit the software package. A confirmation screen is opened.

E.2.2 ALGORITHM Screen

This screen contains special parameters concerning the algorithm used. The screen has two different phases, The **BATCH** and the **SEQUENTIAL**, which can be switched via the radio button in the MAIN screen.

parameters for BATCH algorithm Figure E.4 demonstrates the **BATCH** algorithm screen of the software control panel. There are various options.



ALGORITHM@dsp

paramters for BATCH algoritm

EM-param: [✓ maxims] maximw [✓ estimate] learn

pitch: pitch, pitch_ar

mode: [✓ filter] smooth

options: [✓ covariance] [✓ lms]

kalman: white filter

LPC order: 4

iterations: 5

cumulant used per iteration

1	2	3	4	5	6
4	2	2	2	2	2

overlap[%]: 0

frame[mSec]: 25

Figure E.4: GUI - BATCH Algorithm Screen

The EM-param group contains the check boxes maxims - which determines whether to apply maximization of the speech parameters during the **M-step**, maximw - which determines whether to apply maximization of the noise parameters during the **M-step**, estimate - which determines whether to apply the Kalman filter in the **E-step**, and the learn - which is used for learning the the parameters from the clean speech, in order to evaluate the upper-bound on the performance.

The pitch group is used for determining whether to use pitch information in either the Kalman filter (check box pitch), or in the Yule-Walker equations (check box pitch_ar), or in both.

In the options group the covariance check box determines whether the covariance matrix from Kalman filtering is used. The lms check box is not in use.

The mode group is not in use.

The kalman check box determines whether the Kalman filter is using the standard equations, assuming white noise, or the more complicated equations using the colored noise structure. if “colored” Kalman filter is used, than the editable text field LPC order is determining the AR model of the noise.

The editable text field iterations, is responsible for the number of iterations used in the EM algorithm. In each iteration a pop-up menu determines what is the cumulant used. There are five options: Fourth order cumulants, Third order cumulants, second order cumulants, Modified second order cumulants, or mix order cumulants (not valid).

The pop-up menu and editable text fields overlap[%] is responsible for the percentage of overlap between adjacent segments used.

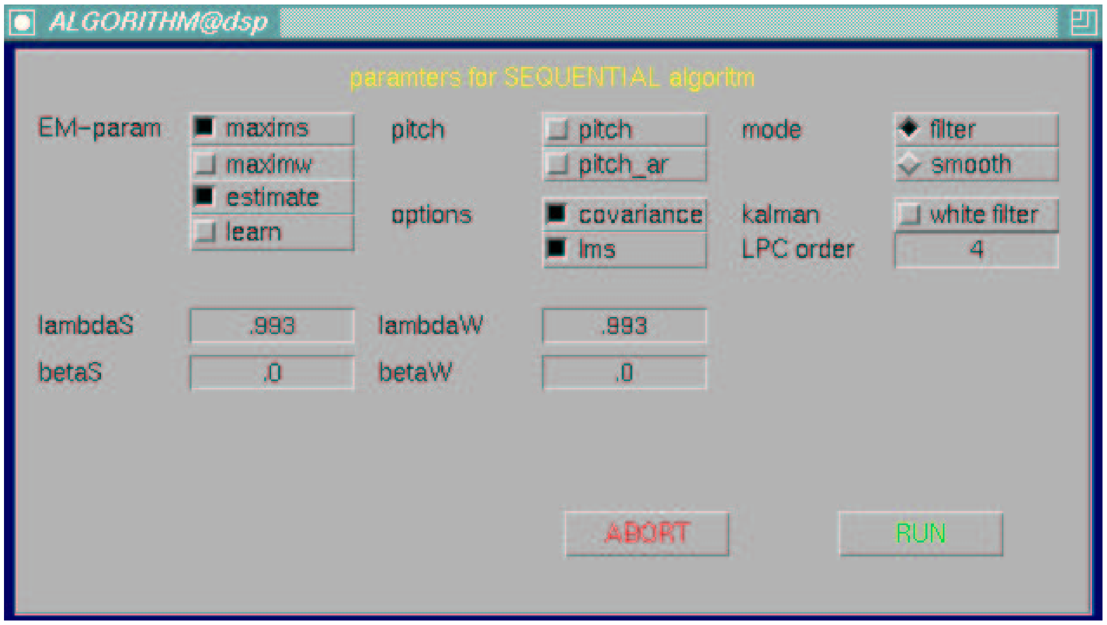
Frame length is determined by both the pop-up menu and the editable text frame[mSec] fields.

The RUN and ABORT push buttons controls the flow of the program.

parameters for SEQUENTIAL algorithm Figure E.5 demonstrates the **SEQUENTIAL** algorithm screen of the software control panel.

This screen is rather similar to the previous BATCH screen. There are some differences.

An editable text fields lambdaS and lambdaW control the forgetting factor of the SEQUENTIAL algorithm (both recursive estimation). The editable text fields betaS and betaW are for future use (can be used for the LMS recursion).

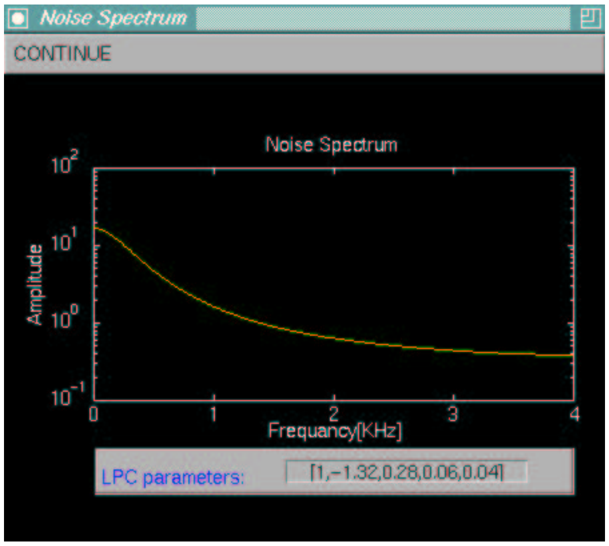

ALGORITHM@dsp

paramters for SEQUENTIAL algoritm

EM-param	■ maxims	pitch	☐ pitch	mode	◆ filter
	☐ maximw		☐ pitch_ar		◇ smooth
	■ estimate	options	■ covariance	kalman	☐ white filter
	☐ learn		■ lms	LPC order	4
lambdaS	.993	lambdaW	.993		
betaS	.0	betaW	.0		

ABORT RUN

Figure E.5: GUI - SEQUENTIAL Algorithm Screen


Noise Spectrum

CONTINUE

LPC parameters: [1,−1.32,0.28,0.06,0.04]

Figure E.6: GUI - Noise Spectrum Screen

E.2.3 Noise Spectrum Screen

At the initial stage of the algorithm the user is prompted with a graph of the Noise Spectrum. When an artificial colored noise is chosen, the user can change its AR coefficients to achieve a desired spectrum . The menu item CONTINUE enables the user to either continue the run or to print out the resulting spectrum. Figure E.6 demonstrates the window.

E.2.4 GRAPH Screen

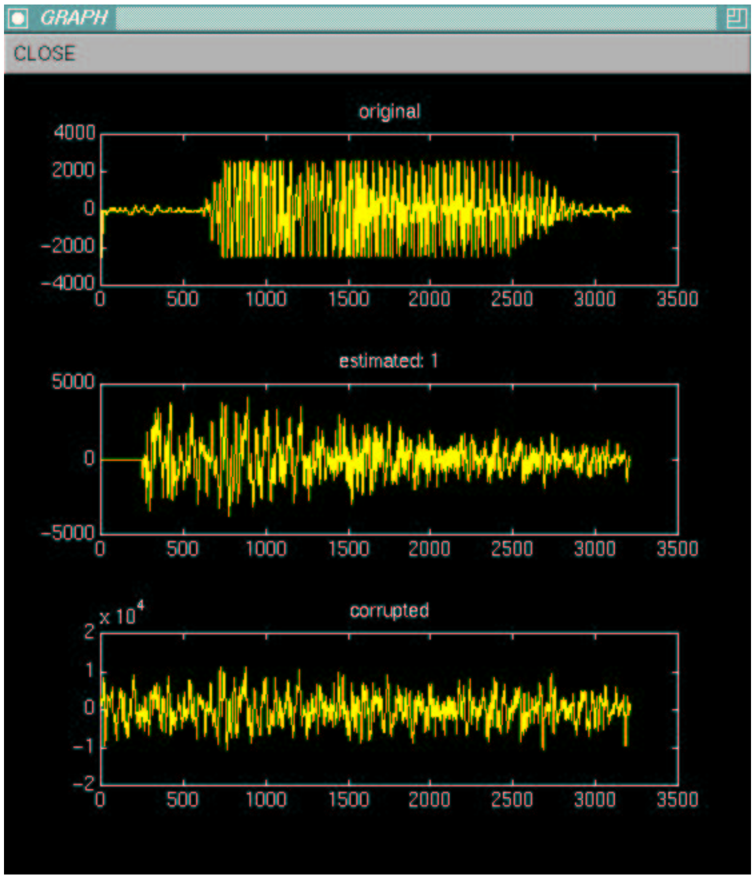
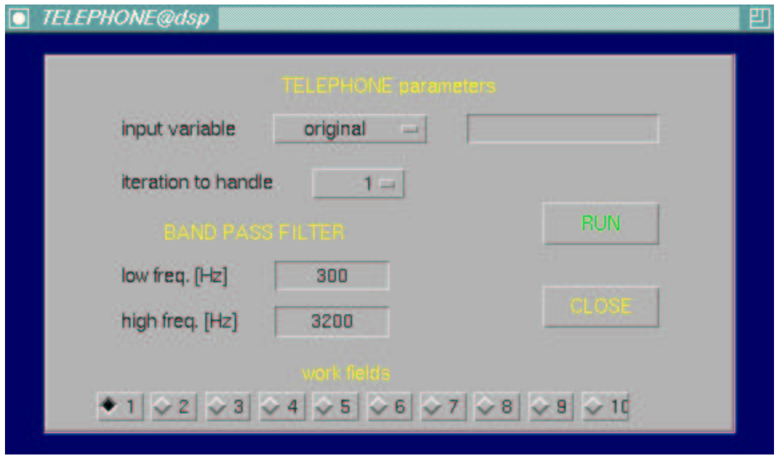


Figure E.7: GUI - Graph Screen

The GRAPH window can be opened by activating the GRAPH push button in the MAIN window. The graphs of all the selected signals are displayed in this window. If a signal which has several iterations is chosen, only the selected iteration is being displayed. If “LMS” or “Telephone” signals are chosen, only the selected work field are displayed. Figure E.7 demonstrates the window.

E.2.5 Post-filters Screen

As mentioned before pre-algorithms can be applied on the results of the algorithm. “LMS” or “Telephone” algorithms are currently available.



input variable original

iteration to handle 1

low freq. [Hz] 300

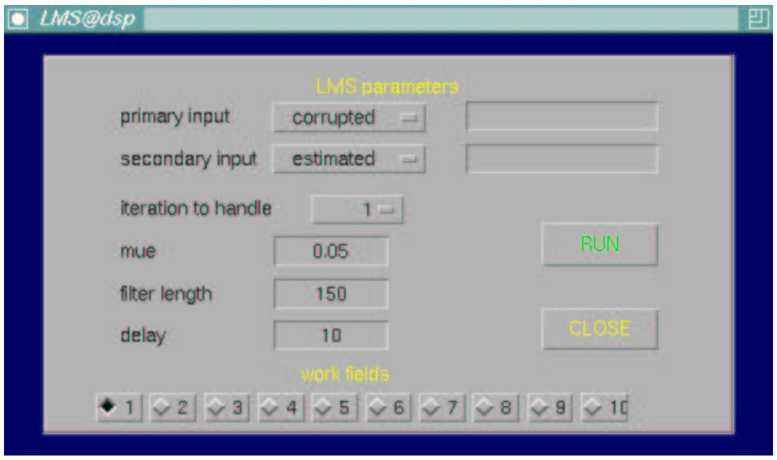
high freq. [Hz] 3200

Figure E.8: GUI - Telephone Parameters Screen

Telephone Figure E.8 demonstrates the The “Telephone parameters” screen. This is actually a Band-Pass filter with user defined cut-off frequencies.

The signal to be filtered is selected via the pop-up menu input variable and the iteration to handle pop-up menu determines which iteration to filter. The cut-off frequencies are determined in the editable text fields low freq.[Hz] and high freq.[Hz]. The work fields radio button selects one temporary variable to store the results in. The selected work field is the one presented in the GRAPH window, and being played by the PLAY command. The push button RUN and close are obvious.

LMS Figure E.9 demonstrates the The “LMS parameters” screen. This window is the control panel of the LMS algorithm that can be applied as a post-filter for our algorithms. the primary input and secondary input pop-up menus selects the signals to be handled. The various parameters of the the LMS algorithms



primary input: corrupted
secondary input: estimated
iteration to handle: 1
mue: 0.05
filter length: 150
delay: 10

Figure E.9: GUI - LMS Parameters Screen

are determined via the editable text fields mue, filter length, and delay. The delay is used only in ONE MICROPHONE mode.

E.2.6 Exit Screen

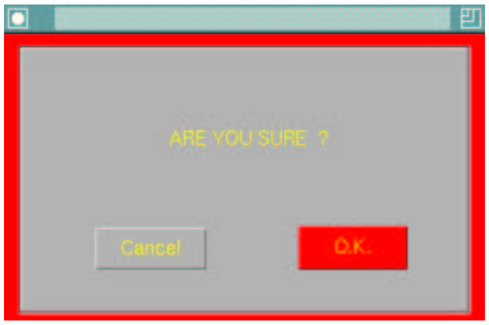


Figure E.10: GUI - EXIT Screen

After choosing Exit from the pull-down menu an Exit screen is opened for confirmation. The Exit screen is demonstrated in Figure E.10.

Bibliography

[1] Alan V. Oppenheim and Jae S. Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529–541, May 1981.

[2] S. F. Boll. Suppression of acoustic noise in speech using spectral subtraction. In Jae S. Lim, editor, *Speech Enhancement*, Alan V. Oppenheim series, pages 61–68. Prentice-hall, 1983.

[3] D.R. Brillinger. *Time Series, Data Analysis and Theory.* San Francisco, CA: Holden-day, 1981.

[4] David Burshtein. Joint modeling and maximum-likelihood estimation of pitch and linear prediction coefficient parameters. *J. Acoustic Society of America*, 3:1531–1537, Mar. 1992.

[5] E. Masgrau J. Salavedra A. Moreno and A. Ardanuy. Speech enhancement by adaptive wiener filtering based on cumulant ar modeling. In Michel Grenie and Jean Claude Junqua, editors, *Speech Processing in Adverse conditions*, chapter Speech Analysis and speech enhancement, pages 143–146. 1992.

[6] E. Masgrau José A. Rodrígez-Fonollosa and Antonio Ardanuy. Enhancement of speech by using higher-order spectral modeling. In J. Vandewalle R. Boite M. Moonen and A. Oostterlinck, editor, *Signal Processing VI:Theories and Applications*, pages 307–310. Elsevier Science Publishers B.V., 1992.

[7] E. Weinstein A.V. Oppenheim and M.Feder. Signal Enhancement Using Single and Multi-Sensor Measurement. Technical report, MIT, Nov. 4 1990.

[8] Georgios B. Giannakis and Jerry M. Mendel. Identification of nonminimum phase systems using higher order statistics. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):360–377, Mar. 1989.

[9] Robert M. Gray Andrés Buzo Augustine H. Gray and Yasuo Matsuyama. Distortion measures for speech processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-28(4):367–376, Aug. 1980.

[10] John H. L. Hansen and Mark A. Clements. Constrained iterative speech enhancement with application to speech recognition. *IEEE transactions on signal processing*, 39(4):795–805, Apr. 1991.

[11] Jae S. Lim Alan V. Oppenheim and Louis D. Braida. Evaluation of an adaptive comb filtering method for enhancing speech degraded by white noise addition. In Jae S. Lim, editor, *Speech Enhancement*, Alan V. Oppenheim series, pages 88–92. Prentice-hall, 1974.

[12] Jae S. Lim and Alan V. Oppenheim. All-pole modeling of degraded speech. *IEEE Transaction on Acoustic,Speech and Signal Processing*, 26(3):197–210, Jun. 1978.

[13] Jae S. Lim and Alan V. Oppenheim. Enhancement and bandwidth compression of noisy speech. In Jae S. Lim, editor, *Speech Enhancement*, Alan V. Oppenheim series, pages 7–25. Prentice-hall, 1983.

[14] James D. Wise James Caprio and Thomas W. Parks. Maximum likelihood pitch estimation. *IEEE transactions on Acoustics, Speech and Signal Processing*, 24(5):418–423, Oct. 1976.

[15] N.S. Jayant and P. Noll. *Digital Coding of Waveforms.* Prentice-Hall,Engelwood Cliffs, 1984.

[16] John Makhoul Thomas H. Crystal David M. Green, Douglas Hogan Robert J. McAulay David B. Pisoni, Robert D. Sorkin and Thomas G. Stockham. Removal of noise from noise-degraded speech signals. In *Panel on removal*

of noise from a speech/noise signal. Committee on Hearing, Bioacoustics and Biomechanics, National Research Council, National Academy press, 1989.

[17] K.K. Paliwal and M.M. Sondhi. Recognition of noisy speech using cumulant based linear prediction analysis. *Int. Conf. on Acoustics, Speech and Signal Proc.*, pages 429–432, 1991.

[18] Larry J. Trent Charels M. Rader and Douglas A. Reynolds. Using higher order statistics to increase the noise robustness of a speaker identification system. *Esca workshop on automatic recognition, identification and verification*, pages 221–224.

[19] Lawrence R. Rabiner Michael J. Cheng Aaron E. Rosenberg and Carol A. McGonegal. A comparative performance study of several pitch detection algorithms. *IEEE transaction on Acoustics,Speech and Signal Processing*, 24(5):399–418, Oct. 1976.

[20] Jae S. Lim. Evaluation of a correlation subtraction method for enhancing speech degraded by additive white noise. In Jae S. Lim, editor, *Speech Enhancement*, Alan V. Oppenheim series, pages 83–84. Prentice-hall, 1983.

[21] Jae S. Lim. *Speech Enhancement.* Prentice-Hall, 1983.

[22] Lenhart Ljung. *System Identification.* Prentice Hall, 1988.

[23] Michel Grenie and Jean Claude Junqua, editor. *Speech processing in adverse conditions.* ETRW, Nov. 1992.

[24] M.R. Weiss and E. Aschkenasy. Study and development of the intel technique for improving speech intelligibility. Technical Report NSC-FR/4023, Nicolet Scientific Corp., Dec. 1974.

[25] National Institute of standards and Technology. The DARPA TIMIT acoustic-phonetic continuous speech corpus. CD-ROM NIST Speech Disc 1-1.1, Oct. 1991.

[26] V. Zue J. Glass D. Goodine M. Phillips and S. Seneff. The SUMMIT speech recognition system: Phonological modelling and lexical access. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 49–52, Albuquerque NM, Apr. 1990.

[27] V. Zue J. Glass M. Phillips and S. Seneff. Acoustic segmentation and phonetic classification in the SUMMIT speech recognition system. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 389–392, Glasgow, Scotland, 1989.

[28] Mysore R. Raghuveer and Chrysostomos L. Nikias. Bispectrum estimation: A parametric approach. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(4):1213–1230, Oct. 1985.

[29] Ronald H. Fraizer Siamak Samsam Louis D. Braida and Alan V. Oppenheim. Enhancement of speech by adaptive filtering. In Jae S. Lim, editor, *Speech Enhancement*, Alan V. Oppenheim series, pages 85–87. Prentice-hall, 1974.

[30] Thomas W. Parsons. *Voice and Speech Processing*, chapter Speech generation and perception, pages 59–83. McGraw-Hill, 1987.

[31] Thomas W. Parsons. *Voice and Speech Processing.* McGraw-Hill, 1987.

[32] J. Polifroni V. Zue J. Glass D. Goodine H. Leung M. Phillips and S. Seneff. Recent progress on SUMMIT system. In *Proceedings of the Speech and Natural Language Workshop*, pages 380–384, Hidden Valley, PA, 1990.

[33] Shawn M. Verbout. Signal enhancement for automatic recognition of noisy speech. Technical Report RLE 584, MIT, May 1994.

[34] B. Widrow, J.R. Glover Jr., J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeider, E. Dong Jr., and R.C. Goodlin. Adaptive Noise Cancelling: Principals and Applications. *Proceeding of the IEEE*, 63(12):1692–1716, Dec. 1975.