ISSN 0280–5316
ISRN LUTFD2/TFRT--5572--SE

Speech Coding Using Orthonormal Basis Functions

Sven Hedlund

Department of Automatic Control
Lund Institute of Technology
January 1997

Department of Automatic Control
Lund Institute of Technology
Box 118
S-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
January, 1997

Document Number
ISRN LUTFD2/TFRT--5572--SE

Author(s)
Sven Hedlund

Supervisor
Brett Ninness and Björn Wittenmark

Sponsoring organisation

Title and subtitle
Speech Coding Using Orthonormal Basis Functions. (Talkodning med användning av ortogonala baser).

Abstract

Vocoders are coders that are designed to encode human speech efficiently by using analytical models of the human vocal system. The process of encoding involves dividing speech audio into many short segments, then finding the parameters that cause the model to generate the closest approximation to the sound in each segment. When using linear predictive coding, LPC, compression is accomplished by modeling the speech signal in each segment as an AR process and encoding the *model parameters* rather than the speech. Decoding consists of sending the parameters to the model to recreate the sequence of audio segments.

The adaptive all-pole filter used for linear prediction approximates the true physical configuration of the human vocal tract, but with the nasal tract left out. The contribution from the nasal tract suggests the need for a pole-zero model instead of the conventional one. In this report, we try to improve the traditional LPC by adding fixed zeros to the vocal tract filter. Since the zeros are fixed, they will not increase the parameters needed to represent the model. Experiments that are made to determine how successful these ideas are when it comes to perception by the human ear (i.e. sample, code, decode and listen), show that one can not achieve significantly better speech quality with any configuration of the added zeros.

Key words
Speech Coding, Vocal Tract Filter, Zeros

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
46

Recipient's notes

Security classification

Contents

1 Introduction 2
1.1 Acknowledgements . . . 3

2 Linear Predictive Coding of Speech 4
2.1 Source-Filter Model of Speech . . . 4
2.2 Linear Prediction Analysis of Speech . . . 5
2.2.1 Principles of Linear Prediction . . . 5
2.2.2 Determination of the Predictor Parameters . . . 6
2.2.3 Estimation of Remaining Parameters . . . 8
2.3 Speech Synthesis . . . 13

3 LPC Using a Pole-Zero Filter 17
3.1 Another Way to Look at Traditional LPC . . . 17
3.2 Introducing Zeros in the Vocal Tract Filter . . . 19
3.2.1 Previous Work on Pole-Zero Modelling for Speech Signals . . . 20
3.3 Finding the Zeros of the Vocal Tract Filter . . . 21
3.3.1 Trial-And-Error Search . . . 21
3.3.2 Systematic Search . . . 23
3.4 Analysis of the Poles of the Vocal Tract Filter . . . 27
3.4.1 Finding the Poles of the Vocal Tract Filter . . . 27
3.4.2 Using the Poles of the Vocal Tract Filter . . . 29
3.5 Finding the Zeros of the Vocal Tract Filter (second approach) . . . 33

4 Conclusions 40

Chapter 1

Introduction

Medium and low bit rate, high quality speech coders are in increasing demand to maximally utilize the channel capacity of a transmission medium especially in the case of cellular phone, since the number of subscribers is increasing every day. The aim of speech coding is to produce a compact representation of speech sounds such that when reconstructed it is perceived to be close to the original.

The frequency content of typical speech is within the range 300-3300 Hz. According to Nyquist's Sampling Theorem, speech has to be sampled at 6.6 kHz to avoid aliasing and ensure perfect reconstruction. However, 8 kHz has become the standard sampling frequency because it provides a margin of error. Uncompressed speech using 8-bit quantization requires a total bit rate of 64 kbits/s. This rate is far too high and can be reduced considerably through various compression schemes.

There are mainly two types of speech coders, waveform coders and vocoders. Waveform coders attempt to reproduce the waveform of the input signal. Waveform coders are generally designed to be signal independent so they can be used to code a wide variety of signals. However, if the signal is always a speech signal then it would be more efficient to use this knowledge when producing the signal.

Vocoders (VOice CODERS) are designed to encode human speech efficiently by using analytical models of the human vocal system. The process of encoding involves dividing speech audio into many short segments, then finding the parameters that cause the model to generate the closest approximation to the sound in each segment. The resulting sequence of parameter settings constitutes the encoded speech. Decoding consists of sending the parameter settings to the model to recreate the sequence of audio segments. Thus, the compression resulting from this method is accomplished by modeling speech as an AR process and encoding the *model parameters* rather than the speech. This method of speech compression and decompression is known as linear predictive coding, LPC, and has been found to give high compression ratios. Typical LPC techniques result in rates of 2.4 kbits/s, a considerable improvement over the uncompressed rate of 64 kbits/s.

LPC was in its simplest form used some twenty years ago and the coding method has continuously been further developed and combined with other ideas since then. In this report we return to the original LPC and try to improve it by extending one part of the

model without increasing the number of parameters needed to represent it. This might lead to a higher compression ratio without a significant increase of computational load.

The structure of this report is as follows. In chapter 2, the algorithms and mathematics behind traditional LPC are presented. We need to understand these, since the proposed new model is based on much of the traditional LPC. In chapter 3, the new model is presented and used in various testing schemes on speech samples in order to find the parameters that best suit the model and to see how high compression ratios we can get. Finally, conclusions on this investigation are made in chapter 4.

1.1 Acknowledgements

I would like to thank Brett Ninness at the Department of Electrical and Computer Engineering, University of Newcastle, Australia for having offered me this project and supervising me. I am grateful to The University of Newcastle and their division International Students Office as well as The Lund Institute of Technology and all the others that has made the "Study Abroad and Exchange Program" possible.

Chapter 2

Linear Predictive Coding of Speech

2.1 Source-Filter Model of Speech

To process the speech signal in an appropriate way it is normally beneficial to consider a model of the speech signal. Two approaches have been used for modelling of speech and its production. The first approach – which is considered in this research – is an attempt to model the speech production system as to consider the speech signal as the output of the model. The other approach is an attempt to model the speech directly (even sometimes regardless of the speech production system).

The vocal tract is usually modelled as a multitube lossless system, including both ends (i.e. glottis and lips). The preferred arrangement for the configuration of the tubes is the concatenation of acoustic tubes, which simplifies the analysis from the mathematical point of view. This structure well serves the vowels and unvoiced phonemes, however, a more realistic model for nasals is a multitube model with a branch form simulation of the nasal cavity.

A simple model for the excitation source that stimulates the vocal tract is either a periodic pulse generator or a random noise source, selected by a voiced/unvoiced switch. In this model the vocal tract and source are considered to be independent and without any interaction effects. This model is called the source-filter model of speech, and has widespread use in all speech processing applications. For processing purposes this model is usually implemented as in Fig. 2.1.

The spectral envelope of the synthesized speech follows the shape of the filter transfer function, which is normally extracted by spectral analysis from the speech signal. It should be noted that while the acoustic filter is formed by the vocal tract, the spectral envelope resulting from conventional spectral analysis methods normally contains the effects of both vocal tract and glottal pulse shape. Therefore, for voiced segments of speech the spectral envelope represents the combinational of vocal tract and glottal pulse shape.

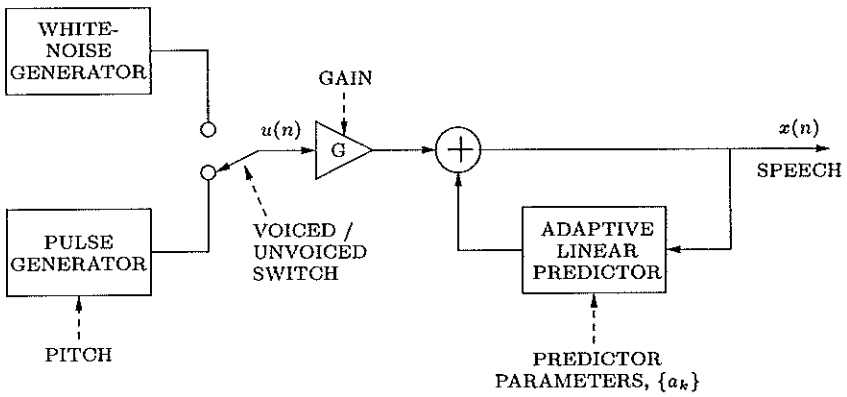


Figure 2.1: Block diagram of simplified model for speech production (LPC system).

2.2 Linear Prediction Analysis of Speech

The success of the source-filter model of speech in different fields of speech processing (coding, recognition and synthesis) is in large part due to the existence of an effective mathematical analysis method for extracting the filter information from a speech signal. This analysis method, which is called *linear prediction*, can efficiently exploit the linear dependencies between different samples of the signal. This section continues with a brief introduction to the principles of linear prediction analysis. A detailed description of linear prediction analysis can be found in [2]. Also, a classic tutorial by Makhoul et al. [3] about linear prediction is noteworthy.

2.2.1 Principles of Linear Prediction

While the history of least square estimation begins with Gauss, Wiener was the first person who used the term *linear prediction* [2]. In the linear prediction analysis method, each sample of a stationary signal is estimated by a linear combination of the other samples. For instance, the n-th sample of a sequence $(x(n))$ can be estimated as

$$\tilde{x}(n) = \sum_{k=1}^{p} a_k x(n-k) \tag{2.1}$$

where $\tilde{x}(n)$ and a_k denote the predicted signal and the predictor coefficients respectively. The order of the linear predictor is represented by p. The *prediction error* or *residual signal* is defined as the difference between the original sample and its estimated value, that is

$$e(n) = x(n) - \tilde{x}(n) = x(n) - \sum_{k=1}^{p} a_k x(n-k) \tag{2.2}$$

This equation is in the Z domain represented as

$$E(z) = X(z) - \sum_{k=1}^{p} a_k X(z) z^{-k} = X(z)A(z) \tag{2.3}$$

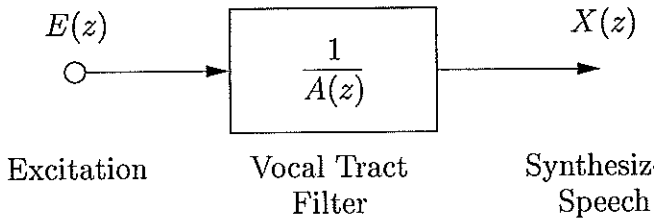


Figure 2.2: Speech production by linear prediction model

where

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k} \tag{2.4}$$

Equation 2.3 can be interpreted as the linear filtering of the signal $X(z)$ by the $A(z)$ filter. If the roots of this filter are assumed to be inside the unit circle (minimum-phase property [4]), the signal can be seen as the output of the *inverse filter* (an all-pole filter) when it is stimulated by the residual signal. That is

$$X(z) = A^{-1}(z)E(z) = H(z)E(z) \tag{2.5}$$

This process is shown in Fig. 2.2.

In linear prediction analysis the predictor coefficients are calculated in a way that the energy of the error signal is minimized, which is the same as the least-square solution.

2.2.2 Determination of the Predictor Parameters

Linear prediction analysis deals with deterministic or stationary stochastic signals. Thus, to use this method for the analysis of speech signals, first a segment which has this property should be selected. Normally, a frame of speech with a duration of 5 to 40 ms (or equivalently 40 to 320 samples for 8 kHz sampling frequency) is appropriate for this purpose. Two analysis methods will be reviewed here briefly.

Autocorrelation Method

Suppose the m-th frame of a speech sequence is to be analyzed and each frame has N samples. The frames overlap and the distance between two subsequent frames is D samples. Selection of this frame can be seen as multiplication of the speech signal by a rectangular window:

$$s(n) = w(n)x(n) \tag{2.6}$$

where $w(n)$ is zero outside the m-th frame and unity inside it. Hence $s(n)$ is zero outside the window. The error energy in this frame is equivalent to

$$E = \sum_{n=(m-1)D+1}^{(m-1)D+N} e^2(n) = \sum_{n=(m-1)D+1}^{(m-1)D+N} \left[s(n) - \sum_{k=1}^{p} a_k s(n-k) \right]^2 \tag{2.7}$$

Minimization of the error energy can be done by setting $\frac{\partial E}{\partial a_k} = 0$ for $k = 1, 2, \ldots, p$. This results in p linear equations for the unknown a_k:

$$\sum_{n=-\infty}^{\infty} s(n)s(n-i) = \sum_{k=1}^{p} a_k \sum_{n=-\infty}^{\infty} s(n-k)s(n-i) \qquad \text{for } i = 1, 2, \ldots, p. \tag{2.8}$$

Considering that the $s(n)$ is zero outside of the window, we have for the autocorrelation of $s(n)$

$$R(k) = \sum_{n=(m-1)D+k}^{(m-1)D+k+N-1} s(n)s(n-k) \qquad \text{for } k = 1, 2, \ldots, p \tag{2.9}$$

so that the equation 2.8 reduces to

$$\sum_{k=1}^{p} a_k R(i-k) = R(i) \qquad \text{for } i = 1, 2, \ \ldots, p. \tag{2.10}$$

The solution of this equation is discussed later. The residual error can be calculated as

$$E_{min} = R(0) - \sum_{k=1}^{p} a_k R(k) \tag{2.11}$$

where $R(0)$ is the energy in $s(n)$. The *prediction gain (PG)* is defined as

$$PG = \frac{R(0)}{E_{min}}. \tag{2.12}$$

A desirable feature of the autocorrelation method is the minimum-phase property of the linear prediction filter which results from it.

Since the rectangular window does not have a good frequency response (large sidelobes), other windows, such as the Hamming window, are normally preferred.

Covariance Method

The autocorrelation method suffers from the artificial boundary effects, because it uses a window before error minimization and this degrades its performance. By contrast, the covariance method does not need any windowing operation.

In this method the energy of the error in a frame is taken to be

$$E = \sum_{n=(m-1)D+1}^{(m-1)D+N} e^2(n) = \sum_{n=(m-1)D+1}^{(m-1)D+N} \left[x(n) - \sum_{k=1}^{p} a_k x(n-k) \right]^2. \tag{2.13}$$

This time, setting the $\frac{\partial E}{\partial a_k} = 0$ leads to another set of p linear equations

$$\sum_{k=1}^{p} a_k \phi(i,k) = \phi(i,0) \qquad \text{for } 1 \leq \ i \leq p, \tag{2.14}$$

where

$$\phi(i,k) = \sum_{n=(m-1)D+1}^{(m-1)D+N} x(n-k)x(n-i) \qquad \text{for } 0 \le i,k \le P. \tag{2.15}$$

The *covariance* analysis method draws its title from the fact that $\phi(i,k)$ is the element in the i-th row and k-th column of the covariance matrix.

It has been shown that the covariance method provides the optimum solution, and gives the same results as *minimum-variance* and *maximum likelihood* spectral estimation methods, if the residual error is assumed to have a Gaussian probability density function (PDF).

Solution Methods

Although a set of linear equations can be solved with any conventional mathematical methods, we are only interested in efficient solution techniques that use the properties of the linear equations derived from the linear prediction analysis.

First we look at the autocorrelation analysis method. The matrix representation of equation 2.10, which is called the *normal equation* or *Yule-Walker* equation, can be shown to be

$$\begin{bmatrix} R(0) & R(1) & R(2) & \cdots & R(p-1) \\ R(1) & R(0) & R(1) & \cdots & R(p-2) \\ R(2) & R(1) & R(2) & \cdots & R(p-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R(p-1) & R(p-2) & R(p-3) & \cdots & R(0) \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ R(3) \\ \vdots \\ R(p) \end{bmatrix} \tag{2.16}$$

The first matrix is both *toeplitz* and symmetric because of its structure. It is well known that a toeplitz matrix is *positive-definite*, and this is the reason for the minimum-phase property of the resulting filter. There is a particular algorithm for solving these normal equations known as the Levinson-Durbin [5] method, which iteratively calculates both the a parameters ($\{a_k\}$, $1 \le p \le P$) and the reflection coefficients. (The reflection coefficients is another set of parameters that is used to represent the vocal tract filter which has a nice physical interpretation, see for instance [4].) The complexity of this iterative method is much less than for other conventional methods.

2.2.3 Estimation of Remaining Parameters

The basic discrete-time model for speech production used in linear predictive analysis is depicted in Fig. 2.1 on page 5. In this model, the composite spectrum effects of radiation, vocal tract, and glottal excitation are represented by a time-varying digital filter whose steady-state system function is of the form

$$H(z) = \frac{X(z)}{U(z)} = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} \tag{2.17}$$

This system is excited by an impulse train for voiced speech or a random noise sequence for unvoiced speech. Thus, the parameters of this model are: voiced/unvoiced classification, pitch period for voiced speech, gain parameter G, and the a parameters of the digital filter. These parameters all vary slowly with time. In section 2.2.2 it was shown how to estimate the a parameters, but we also must learn how to estimate the remaining parameters. This can be done with some algorithms found in [6].

Pitch Period Estimation and Voiced/Unvoiced Classification

Numerous algorithms for estimating the pitch period from the short-time autocorrelation function representation have been proposed. An algorithm which has been implemented in digital hardware [7] is used (slightly modified) in this research. This algorithm serves it purposes well and also includes the possibility to decide which parts of the speech sample that actually contains speech and not just noise.

1. The speech signal (sampled at a rate of 8 kHz) is filtered with a 900 Hz lowpass digital filter:

2. Segments of length 37.5 msec (300 samples) are selected at 12.5 msec (100 samples) intervals. Thus, the segments overlap by 25 msec.

3. The average magnitude is computed with a 100 samples rectangular window, i.e.

$$M_n = \sum_{m=-\infty}^{\infty} |x_f(m)| w(n-m) \qquad \text{where } w(n) = \begin{cases} 1 & 0 \le n < 100 \\ 0 & otherwise \end{cases} \tag{2.18}$$

The peak signal level (of the *AMF, Average Magnitude Function*) in each frame is compared to a threshold determined by measuring the peak signal level (of the AMF) for 50 msec of background noise. If the peak signal level is above threshold, signifying that the segment is speech, not noise, then the algorithm procedes as follows; otherwise the segment is classed as silence and no further action is taken.

4. A clipping level is determined as a fixed percentage (e.g. 68%) of the minimum of the maximum absolute values in the first and last 100 samples of the speech segment, i.e.

$$clipping\ level = 0.68 \cdot min \left\{ \begin{array}{l} max\left[|x_f(n)|, |x_f(n+1)|, \ldots, |x_f(n+99)|\right], \\ max\left[|x_f(n+200)|, |x_f(n+201)|, \ldots, |x_f(n+299)|\right] \end{array} \right\} \tag{2.19}$$

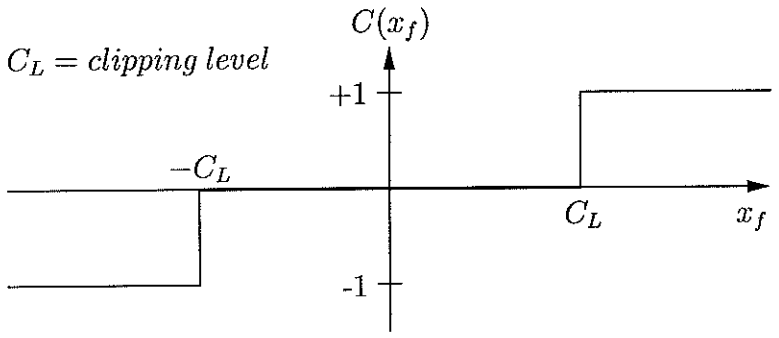


Figure 2.3: 3-level center clipping function.

5. Using this clipping level, the speech signal is processed by a 3-level center clipper (cf Fig. 2.3) and the autocorrelation function is computed over a range spanning the expected range of pitch periods. (The speech signal is clipped as to make the periodicity more prominent while suppressing other distracting features of the signal [6].)

6. The largest peak of the correlation function is located and the peak value is compared to a fixed threshold (e.g. 30% of $R_n(0)$). If the peak falls below threshold, the segment is classed as unvoiced and if it is above, the pitch period is defined as the location of the largest peak.

Having used this algorithm, a few obvious errors remain in the pitch contour. These errors can be effectively removed by a nonlinear smoothing method.

In most signal processing applications a linear smoother (or a linear filter) is generally used to eliminate the noise-like components of a signal. For some speech processing applications, however, linear smoothers are not completely adequate due to the type of data being smoothed. An example is the pitch contour you get from the algorithm shown above. An ordinary linear lowpass filter would not only fail to bring some errant points back into line but would severely distort the contour at the transition between voiced and unvoiced speech (represented as zero period). For such cases some type of nonlinear smoothing algorithm which can preserve signal discontinuities yet still filter out large errors is required. Although an ideal algorithm with these properties does not exist, a nonlinear smoother using a combination of running medians and linear smoothing (originally proposed by Tukey [8]) can be shown to have approximately the desired properties [9].

Different aspects of various combinations of median and linear smoothing are presented in [6]. In this research a combination smoother as a median of 5 followed by a median of 3 followed by a 3-point hanning window was found to be appropriate. It follows the changes in the input signal quite well while eliminating most of the noise in the signal.

Computation of the Gain for the Model [3]

It is reasonable to expect that the gain, G, could be determined by matching the energy in the signal with the energy of the linearly predicted samples. This indeed is true when appropriate assumptions are made about the excitation signal to the LPC system (cf. Fig. 2.1 on page 5).

It is possible to relate the gain constant G to the excitation signal and the error in prediction by rearranging Eq. 2.17 and referring back to Eq. 2.2. The excitation signal, $Gu(n)$, can be expressed as

$$Gu(n) = x(n) - \sum_{k=1}^{p} a_k x(n-k) \tag{2.20}$$

whereas the prediction error signal $e(n)$ is expressed as

$$e(n) = x(n) - \sum_{k=1}^{p} a_k x(n-k) \tag{2.21}$$

Hence,

$$e(n) = Gu(n) \tag{2.22}$$

i.e. the input signal is proportional to the error signal with the constant of proportionality being the gain constant, G.

Since Eq. 2.22 is only approximate (i.e. the filter of the model is not identical to the true system) it is generally not possible to solve for G in a reliable way directly from the error signal itself. Instead the more reasonable assumption is made that the energy in the error signal is equal to the energy in the excitation input, i.e.

$$G^2 \sum_{m=0}^{N-1} u^2(m) = \sum_{m=0}^{N-1} e^2(m) = E_{min} \tag{2.23}$$

At this point we must make some assumptions about $u(n)$ so as to be able to relate G to the known quantities, e.g. the a_k's and the correlation coefficients. There are two cases of interest for the excitation. For voiced speech it is reasonable to assume $u(n) = \delta(n)$, i.e. the excitation is a unit sample at $n = 0$.[1] For this assumption to be valid requires that the effects of the glottal pulse shape used in the actual excitation for voiced speech be lumped together with the vocal tract transfer function, and therefore both of these effects are essentially modelled by the time-varying linear predictor. This requires that the predictor order, p, be large enough to account for both the vocal tract and glottal pulse effects. For unvoiced speech it is most reasonable to assume that $u(n)$ is a zero mean, unity variance, stationary, white noise process.

Based on these assumptions we can now determine the gain constant G by utilizing Eq. 2.23. For voiced speech, we have as input $\delta(n)$. If we call the resulting output for this

[1]Note that for this assumption to be valid requires that the analysis interval be about the same length as a pitch period.

particular input $h(n)$ (since it is actually the impulse response of the system with transfer function $H(z)$ as in Eq. 2.17 we get the relation

$$h(n) = \sum_{k=1}^{p} a_k h(n-k) + G\delta(n) \tag{2.24}$$

It is readily shown that the autocorrelation function of $h(n)$, defined as

$$\tilde{R}(m) = \sum_{n=0}^{\infty} h(n)h(m+n) \tag{2.25}$$

satisfies the relations

$$\tilde{R}(m) = \sum_{k=1}^{p} a_k \tilde{R}(m-k) \qquad m = 1, 2, \ldots, p \tag{2.26}$$

and

$$\tilde{R}(0) = \sum_{k=1}^{p} a_k \tilde{R}(k) + G^2 \tag{2.27}$$

Since Eqs. 2.26 are identical to 2.10 it follows that

$$\tilde{R}(m) = R(m) \qquad 1 \leq m \leq p \tag{2.28}$$

Since the total energies in the signal ($R(0)$) and the impulse response ($\tilde{R(0)}$) must be equal we can use Eqs. 2.11, 2.23, 2.27 and 2.28 to obtain

$$G^2 = R(0) - \sum_{k=1}^{p} a_k R(k) = E_{min} \tag{2.29}$$

For the case of unvoiced speech, the correlations are defined as statistical averages. It is assumed that the input is white noise with zero mean and unity variance: i.e.

$$E[u(n)u(n-m)] = \delta(m) \tag{2.30}$$

If we excite the system with the random input $Gu(n)$ and call the output $g(n)$ then

$$g(n) = \sum_{k=1}^{p} a_k g(n-k) + Gu(n) \tag{2.31}$$

If we now let $\tilde{R}(m)$ denote the autocorrelation function of $g(n)$, then

$$\tilde{R}(m) = E[g(n)g(n-m)] = \sum_{k=1}^{p} a_k E[g(n-k)g(n-m)] + E[Gu(n)g(n-m)] = \sum_{k=1}^{p} a_k \tilde{R}(m-k) \qquad m \neq 0 \tag{2.32}$$

since $E[u(n)g(n-m)] = 0$ for $m > 0$ because $u(n)$ is uncorrelated with any signal prior to $u(n)$. For $m = 0$ we get

$$\tilde{R}(0) = \sum_{k=1}^{p} a_k \tilde{R}(k) + GE[u(n)g(n)] = \sum_{k=1}^{p} a_k \tilde{R}(k) + G^2 \tag{2.33}$$

since $E[u(n)g(n)] = E[u(n)(Gu(n) + \textit{terms prior to } n)] = G$. Since the energy in the response to $Gu(n)$ must equal the energy in the signal, we get

$$\tilde{R}(m) = R(m) \qquad 0 \leq m \leq p \tag{2.34}$$

or

$$G^2 = R(0) - \sum_{k=1}^{p} a_k R(k) \tag{2.35}$$

as was the case for the impulse excitation for voiced speech.

2.3 Speech Synthesis

In this investigation, we will always use a frame length (denoted N in section 2.2.2) of 300 samples and the frames overlap by 100 samples (denoted D). The parameter estimation is shown schematically in Fig. 2.4.

The speech signal is synthesized by means of the same parametric representation as was used in the analysis (cf. Fig. 2.1, p. 5). The control parameters supplied to the synthesizer are the pitch period, a binary voice/unvoiced parameter, the gain of the speech samples, and the predictor coefficients. The pulse generator produces a pulse of unit amplitude at the beginning of each pitch period. The white-noise generator produces uncorrelated uniformly distributed random samples with standard deviation equal to 1 at each sampling instant. The selection between the pulsegenerator and the white-noise generator is made by the voiced/unvoiced switch. The amplitude of the excitation signal is adjusted by the amplifier, G. The linearly predicted value of the speech is combined with the excitation signal to form the next sample of the synthesized speech signal.

The synthesizer control parameters are reset to their new values at the beginning of every pitch period for voiced speech and once every 10 msec for unvoiced speech. Since the control parameters are not determined *pitch-synchronously* (to be explained) in the analysis, new parameters are computed by suitable interpolation of the original parameters to allow pitch-synchronous resetting of the synthesizer.

This idea is shown in Fig. 2.5. In the uppermost row the data blocks are shown the way they are organized as they are received. Each block of data represents 100 samples, as this is in our case the distance between the analyzed frames. Assuming for a while that we only deal with voiced speech, we reproduce the speech signal by exciting a filter by a pulse train. We need to know the desired gain every time a new pulse reaches the filter, rather then knowing it at certain instances equally distributed in time. To generate a sample without unnatural discontinuities, the same applies for the filter parameters: it

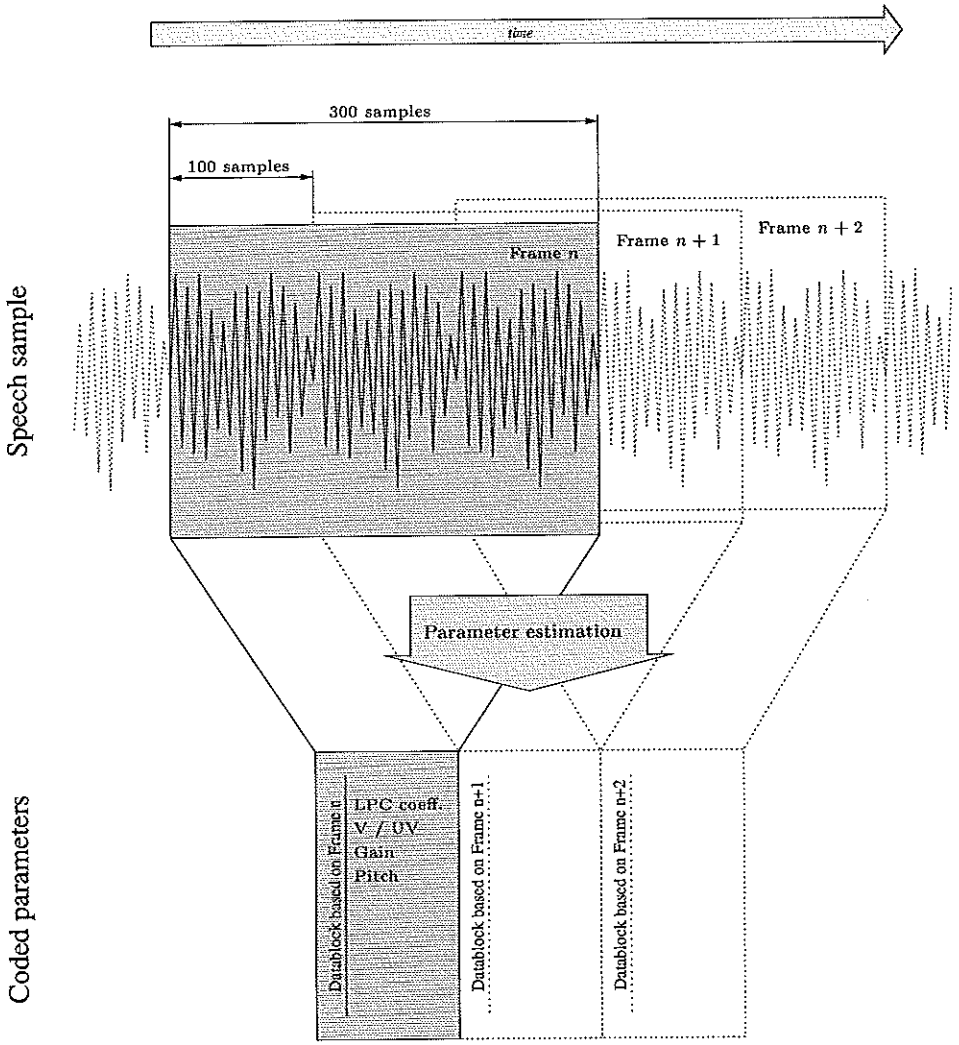


Figure 2.4: Analysis of speech

seems natural to re-tune the filter each time a new pulse is to enter, rather than changing the filter in the middle of an excitation. This pitch synchronization leads to the middle row of Fig. 2.5, where each (new) block of data now represents the amount of samples required to complete a pitch period. (Note that the item "Pitch" is not explicitly written in these blocks, since that information is given by the "length" of the box.)

In the case of unvoiced speech, the timing of the resetting of the control parameters does not affect the sound as the filter is then fed by white noise. To keep the filter updated, the parameters are interpolated to be changed every 10 msec.

The pitch period and the gain are interpolated geometrically (linear interpolation on a logarithmic scale). In interpolating the predictor coefficients, it is necessary to ensure the stability of the recursive filter in the synthesizer. The stability cannot, in general, be ensured by direct linear interpolation of the predictor parameters. One suitable method is to interpolate the first p samples of the autocorrelation function of the impulse response of the recursive filter. The autocorrelation function has the important advantage of having a one-to-one relationship with the predictor coefficients. Therefore, the predictor coefficients can be recomputed from the autocorrelation function. Moreover, the predictor coefficients derived from the autocorrelation function always result in a stable filter in the synthesizer [10].

The relationship between the predictor coefficients and the autocorrelation function is given by Eqs. 2.26 and 2.27, which enables us to compute the samples of the autocorrelation function from the predictor coefficients, and the predictor coefficients from the autocorrelation function. A computational procedure for performing these operations is outlined in [11].

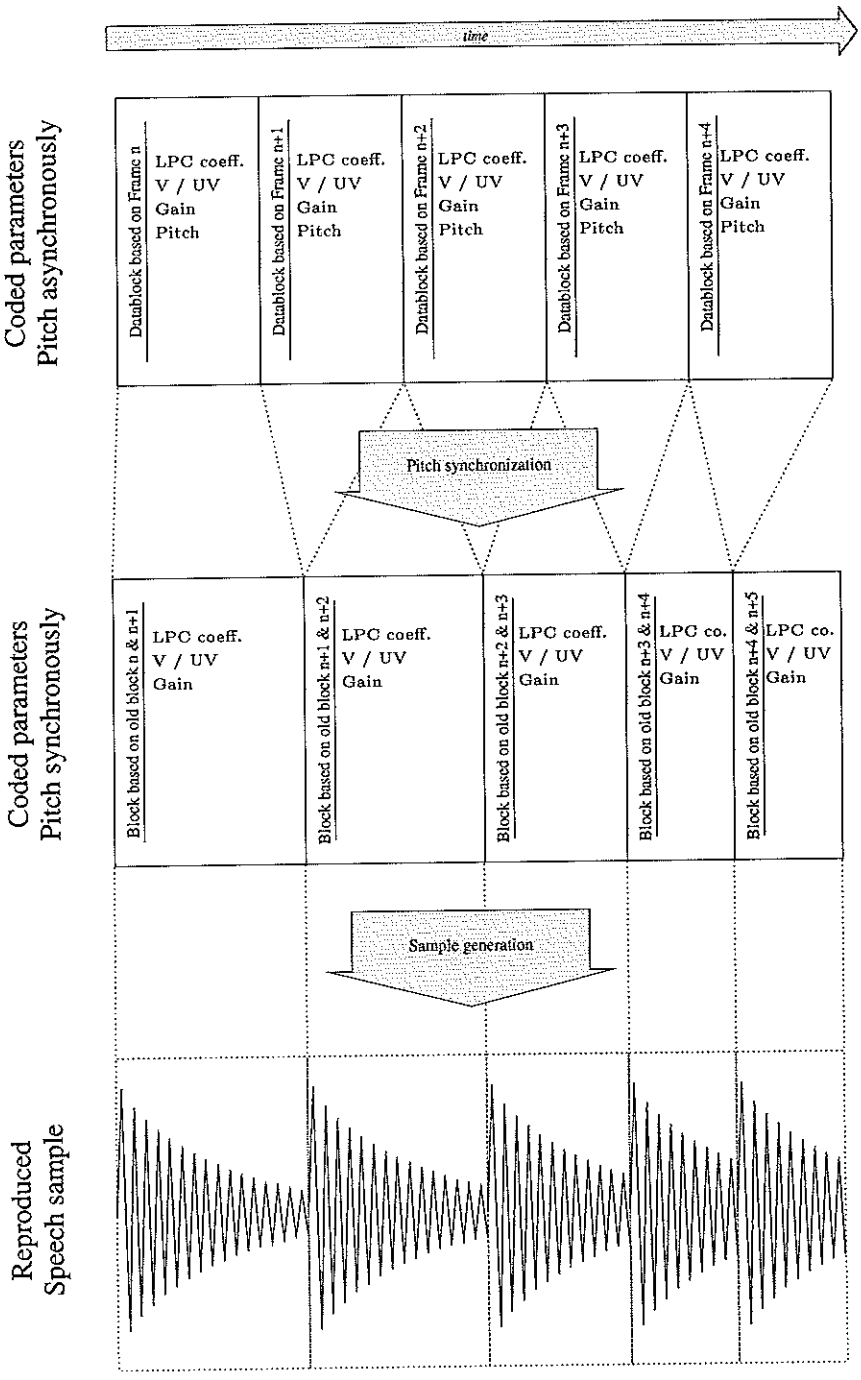


Figure 2.5: Synthesis of speech

Chapter 3

LPC Using a Pole-Zero Filter

3.1 Another Way to Look at Traditional LPC

Linear prediction can be seen as filtering a time series through a set of n basis functions to create a set of n new time series and by adding these series weighted by appropriate constants ($\{a_k\}$) trying to reconstruct the original time series. If we look back to Eq. 2.1:

$$\tilde{x}(n) = \sum_{k=1}^{p} a_k x(n-k) = \sum_{k=1}^{p} a_k q^{-k} x(n) \tag{3.1}$$

this equation describes how to predict the n-th sample from past samples. If we write Eq. 3.1 as

$$\tilde{x}(n) = \sum_{k=1}^{p} a_k B_k(q) x(n) = \sum_{k=1}^{p} a_k \Phi_k(n) \tag{3.2}$$

where $\Phi_k(n) = B_k(q)x(n)$ then it can easily be seen that the choice $B_k(q) = q^{-k}$, $1 \leq k \leq p$ gives Eq. 3.1, although other choices for the $B_k(q)$ are possible. Let us simplify the notation and rewrite this equation in matrix form:

$$\tilde{X} = \sum_{k=1}^{p} a_k \Phi_k \tag{3.3}$$

where $\tilde{X}$ is the predicted time series, and Φ_k is the new orthonormal time series used to build the prediction. They are now represented as column vectors:

$$\tilde{X} = \begin{bmatrix} \tilde{x}(1) \\ \tilde{x}(2) \\ \tilde{x}(3) \\ \vdots \\ \tilde{x}(N) \end{bmatrix} \qquad \Phi_k = \begin{bmatrix} B_k(q)x(1) \\ B_k(q)x(2) \\ B_k(q)x(3) \\ \vdots \\ B_k(q)x(N) \end{bmatrix}$$

The prediction of the time series is shown in Fig. 3.1. No matter how well we choose

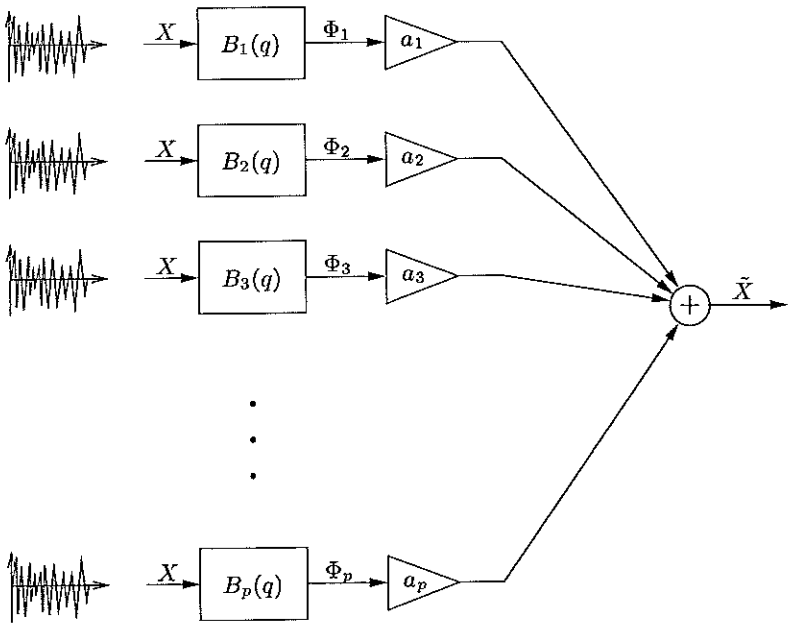


Figure 3.1: Block diagram of Linear Predictor

the a parameters, the predictor given by Eq. 3.1 is not likely to predict all the time series without error. The way we choose the a parameters is to choose a set that minimizes the error energy. If we define the error vector as

$$E \equiv X - \tilde{X} = X - \sum_{k=1}^{p} a_k \Phi_k = X - \Phi\theta \tag{3.4}$$

where

$$X = \begin{bmatrix} x(1) \\ x(2) \\ x(3) \\ \vdots \\ x(N) \end{bmatrix}, \quad \Phi = \begin{bmatrix} \Phi_1 & \Phi_2 & \Phi_3 & \dots & \Phi_p \end{bmatrix} \quad \text{and} \quad \theta = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix}$$

then we can write the total error energy as

$$W_E \equiv E^T E = (X - \Phi\theta)^T (X - \Phi\theta) = X^2 - 2X^T\Phi\theta + (\Phi\theta)^2 \tag{3.5}$$

where the square of a column vector, V^2 is defined as $V^T V$. Minimization of the energy is done by setting $\partial W_E / \partial a_k = 0$ for $1 \leq k \leq p$ or in matrix form

$$\frac{\partial}{\partial\theta} W_E = \frac{\partial}{\partial\theta} \left[X^2 - 2X^T\Phi\theta + (\Phi\theta)^2 \right] = -2X^T\Phi + 2(\Phi\theta)^T\Phi = 0 \tag{3.6}$$

Solving for θ gives

$$\Phi^T\Phi\hat{\theta} = \Phi^T X \qquad \text{or} \qquad \hat{\theta} = (\Phi^T\Phi)^{-1}\Phi^T X \tag{3.7}$$

and hence $\hat{\theta}$ is the set of a parameters that minimizes the error energy.

The interesting thing about this is that these calculations where made without any consideration of $B_k(q)$ and thus the result is independent of the form of these filters. Having made these calculations, we now know how to minimize the error energy for any set of filters provided that they are stable (i.e. the poles of $B_k(q)$ are located within the unit circle).

3.2 Introducing Zeros in the Vocal Tract Filter

As pointed out before, traditional LPC makes a prediction of next sample based on the p past samples:

$$x(n) = \sum_{k=1}^{p} a_k x(n-k) + e(n) = F(q)x(n) + e(n) \tag{3.8}$$

where

$$F(q) = a_1 q^{-1} + a_2 q^{-2} + \ldots + a_p q^{-p} \tag{3.9}$$

Though this turns out to work pretty well, it would be interesting if we could make a model that extends the prediction to include even more of the past samples without adding extra parameters needed to represent the model. The way to do this is to add some fixed poles to $F(q)$. This will in principle make $x(n)$ dependent on an infinite amount of past samples since

$$\frac{q}{q-\xi}x(k) = \frac{1}{1-\xi q^{-1}}x(k) = \left(\sum_{k=1}^{\infty}\xi^k q^{-k}\right)x(k) = \xi x(k-1) + \xi^2 x(k-2) + \xi^3 x(k-3) + \ldots$$

provided that the pole is stable, that is $|\xi| < 1$. $F(q)$ as described in Eq. 3.9 can actually be seen as having a set of poles placed in the origin since

$$F(q) = a_1 q^{-1} + a_2 q^{-2} + \ldots + a_p q^{-p} = \frac{a_1 q^{p-1} + a_2 q^{p-2} + \ldots + a_{p-1}q + a_p}{q^p}$$

Hence the way of extending the predictor is just a matter of moving out the poles to other parts of the unit disc.

When reproducing the speech by sending a pulse train or white noise trough the vocal tract filter as shown in Fig. 2.2, the extended filter will be of the form

$$H(q) = \frac{1}{1-F(q)} = \frac{1}{1-\frac{F_{num}(q)}{F_{den}(q)}} = \frac{F_{den}(q)}{F_{den}(q) - F_{num}(q)} \tag{3.10}$$

where $F_{num}(q)$ and $F_{den}(q)$ represent the numerator and denominator of $F(q)$ respectively. Thus the fixed poles added to $F(q)$ (or moved out from the origin) emerge as zeros in the vocal tract filter, $H(q)$. At this stage two questions arise:

1. How shall we place the zeros of $H(q)$, or equivalently the poles of $F(q)$ to get the best possible approximation of the vocal tract filter?

2. Having placed the zeros, how shall we arrange the calculations to pick the set of filter parameters that minimize the error energy?

Since (provided that we use an appropriate representation of $F(q)$) much work was made in previous section to answer the second question, we start off with these calculations.

If we in some way include the zeros of the vocal tract filter as poles of the base functions, $B_k(q)$, described in Eq. 3.2 (to try to avoid confusion I will from now on refer to them as the zeros), we already know how to minimize the error energy.

So how do we choose the base functions? One desirable feature when constructing them is orthonormality, that is

$$\langle B_n, B_m \rangle = \frac{1}{2\pi} \int_{-\pi}^{\pi} B_n(e^{j\omega}) \overline{B_m(e^{j\omega})} d\omega = \begin{cases} 1 & \text{if } n = m \\ 0 & \text{if } n \neq m \end{cases} \tag{3.11}$$

Orthonormality is important since it leads to

- Improved numerics in solving the least squares normal equations [13].
- Parsimony in representation [14].
- Independence of parameters for broad band excitation signals [15].

In this research, a simple construction

$$B_k(q) = \left(\frac{\sqrt{1 - |\xi_k|^2}}{q - \xi_k} \right) \prod_{i=0}^{k-1} \left(\frac{1 - \bar{\xi}_i q}{q - \xi_i} \right) \tag{3.12}$$

as presented in [12] is used. It preserves orthonormality and furthermore provides a unifying formulation of all known system identification orthonormal systems (FIR, Laguerre and Kautz models). With this construction we can fix the poles, ξ_k $(0 \leq k < p)$, arbitrarily.

3.2.1 Previous Work on Pole-Zero Modelling for Speech Signals

At this stage it should be pointed out that it is not a new idea to use a pole-zero model of the vocal tract. Having searched different sources (e.g. databases such as INSPEC and Compendex) for research on models on the vocal tract filter, one can see that several attempts of including zeros in the vocal tract filter has been made, some of which have been reported as being successful. Those of the papers most relevant to this research are listed in [18] - [34].

Most researchers claim that the traditional LPC reproduces many sounds quite well. By applying LPC analysis to speech signals, we obtain all-pole type digital filters. Speech signals are assumed to be produced by filtering glottal excitation with these filters. Such

an all-pole filter can be directly derived from acoustic tube modeling of the vocal tract (c.f. for instance [6]). The overall shape of the model approximates the true physical configuration of the human vocal tract, *but* with the nasal tract left out. The contribution from the nasal tract suggests the need for a pole-zero model instead of the conventional one.

The vocal tract filters of the papers referred to above all include zeros that are a part of the adaptively changing parameters, i.e. they are re-estimated for each frame. The papers treat various methods of estimating the zeros — separately as well as simultaneously with the poles. The purpose is often to remove some of the adaptive poles of the all-pole filter of an order $p \approx 12$ and replace them by zeros of a pole-zero filter in order to reproduce nasal sounds more accurately.

It is quite possible to reproduce understandable speech using a predictor order as low as $p = 7$, though the resulting sound is quite synthetic. The aim of *this* research is to pick a *fixed* set of zeros in order to get better compression, that is adding zeros to the vocal tract filter without increasing the amount of parameters (we might even be able to decrease it) required to represent the speech.

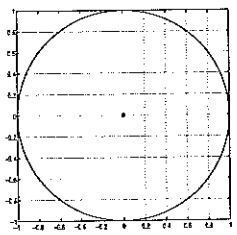
As previously pointed out, the zeros in traditional LPC are placed at the origin. Is there any reason to believe that this choice in general would give a better result than moving them to other positions within the unit disk? At least it is worth investigating the benefits from adding this extra degree of freedom, since we can not do worse: our new model includes the old one as a special case.

3.3 Finding the Zeros of the Vocal Tract Filter

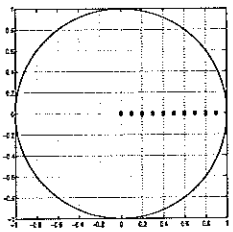
Let us return to the first question of the former section: How shall we place the zeros of the vocal tract filter to get the best possible approximation of the vocal tract filter?

3.3.1 Trial-And-Error Search

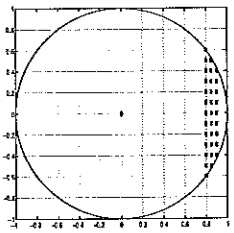
Before utilizing conventional "scientific" methods of analyzing the best zero configuration, let us try to find good positions (i.e. zeros that clearly result in a better reproduction than traditional LPC of the same order) moving poles manually. This might tell us what we can achieve using an enhanced filter.



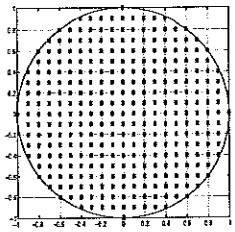
The first zero configuration is of course to place all the zeros at the origin in order to find out how well traditional LPC can do for various orders $1 \leq p \leq 10$. The lowest order where you can catch the words of the spoken sentence is $p \approx 5$, which implies that this is a good model order when deciding if better results are achieved by moving the zeros.



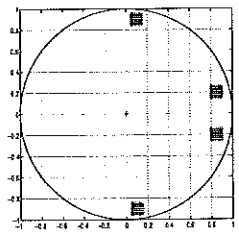
Using 5 zeros, one zero is moved out in the right half plane, further and further away from the origin. Not getting any significant difference, two zeros are moved out simultaneously (both at the same place).



Having found that the reproduced speech may sound a little bit better when two zeros are moved somewhere around $\xi = 0.85$, we examine this area carefully by moving around two complex conjugate zeros everywhere within the region $0.8 \leq \Re(\xi) \leq 0.9$, $|\xi| < 1$. (There are still 3 zeros left at the origin.) Having tried this, the author can notice a *slight* improvement for conjugate zeros at $\xi = 0.85 \pm 0.20i$. There was not a big difference but since that region still is quite interesting, we try to put *two* zero pairs around that position.



The outcome of the former attempt was not that successful. Noticing that we have basically been dealing with *one* degree of freedom (moving around *one* pair of conjugate zeros), we now extend the experiments by moving around *two* pairs of conjugate zeros, thus leaving only one zero at the origin. Since it would take too long just moving around these zeros manually and listen to every resulting reproduced sample, we are now unfortunately forced to rely on mathematic judgment. A MATLAB program is made that tries every combination of two conjugate pairs of zeros placed at the grid shown in the figure to the left. The different configurations are then sorted according to their resulting minimized error energy (as defined in Eq. 3.5, p. 18)



There were several good configurations (with a low value of W_E) with two zeros around $\xi = 0.85 \pm 0.20i$ (as earlier) and the other two around $\xi = 0.10 \pm 0.90i$. Once again we move the zeros manually, this time with one zero in each area shown in the figure. As both of these complex conjugate areas turned out to be quite interesting during other trivial experiments as well (not listed here), before leaving this trial-and-error scheme completely we try the following. Fix one of the zero pairs in the center of its region, then move the other zero pair (still keeping one zero at the origin) and try to find a suitable position just by listening. Fix the other pair of zeros and repeat the procedure.

The fact that none of the attempts outlined above led to any particular improvement forces us to find a more systematic method in our search for a suitable zero configuration.

3.3.2 Systematic Search

Since the attempt to get a better speech coding algorithm is based on the assumption that the "true" zeros of the vocal tract filter is not placed in the origin, we must find a way of estimating these zeros. *System Identification Toolbox*, written by L. Ljung to be used with MATLAB provides a useful means of doing this. The first approach to estimate the zeros is to fit an ARMA model to samples of speech, that is

$$A_{LL}(q)x(n) = C_{LL}(q)e(n) \qquad \text{or} \qquad y(n) = \frac{C_{LL}(q)}{A_{LL}(q)}e(n) \tag{3.13}$$

where the zeros of $C_{LL}(q)$ correspond to the zeros of the vocal tract filter. The index "LL" marks that these polynomials corresponds to the notation used in *L. Ljung's* Toolbox and should not be mixed up with the other polynomials used in this paper. One problem of using this is that the model relies on the input signal, $e(n)$, being white noise. Hence this model can only be used for estimating the zeros (as well as the poles) of stationary portions of *unvoiced* speech since this is the kind of speech that is thought of being driven by white noise.

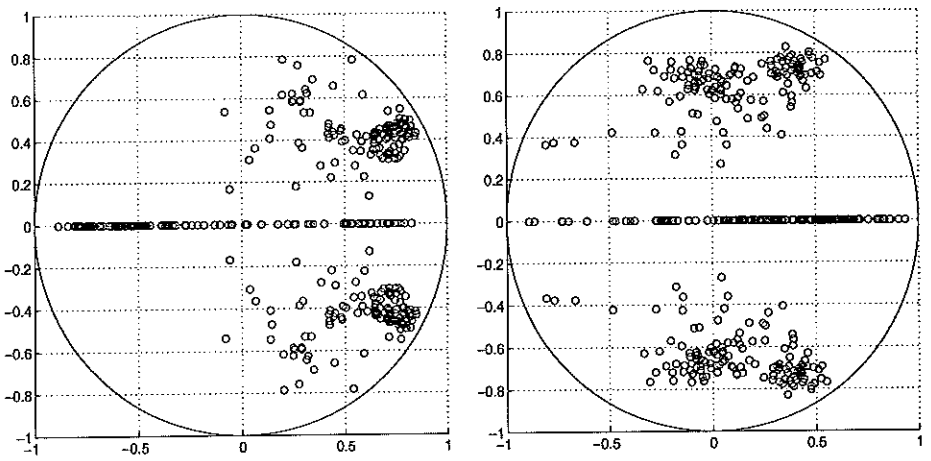
Analysis of Unvoiced Speech

To be able to analyze various sounds and make reliable conclusions, we need a big collection of sound samples, preferably from various speakers. There is an appropriate database for this, *DARPA TIMIT Acoustic-Phonetic Speech Database*, which contains a set of 420 talkers reading in total 4200 sentences. To each sampled sentence of this database there is a corresponding phonetic description in the *ARPABET* notation telling in what range of the sample to find different sounds, which means that we can extract many different samples of a certain sound. Table 3.1 shows a list of the set of ARPABET sounds [16], [17] with corresponding example words in which they occur.

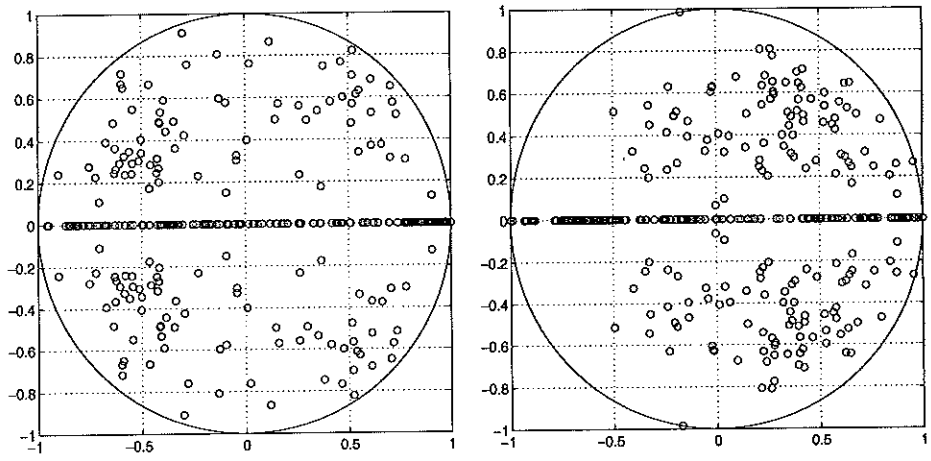
With these samples available we begin our analysis of unvoiced sound. To get a good estimation of the zeros of the vocal tract filter, the sample portions has to be of reasonable length, which restrict us to the *unvoiced fricatives*, that is *F, TH, S* and *SH* (a nice description of various classifications of sounds can be found in [17]).

The aim is to get a good vocal tract filter with some fixed zeros and some variable poles, but how many zeros and poles shall we use when estimating the model of Eq. 3.13? One way of finding an appropriate model order is to try some different orders and use cross validation to evaluate them: if we use a certain set of samples, $\{y_1(n)\}$, to *estimate* the parameters of the model of Eq. 3.13 by minimizing the squared error, $\sum_n e_1^2(n)$, a good model would also give a low value of $\sum_n e_2^2(n) = \sum_n \left((A_{LL}(q)/C_{LL}(q))y_2(n)\right)^2$ for another set, $\{y_2(n)\}$.

Having pursued this estimation/validation procedure for various unvoiced sound samples with $deg(A_{LL})$ and $deg(C_{LL})$ ranging from 1 to 6, one can see that an ARMA(2,3) model (i. e. with $deg(A_{LL}) = 2$ and $deg(C_{LL}) = 3$) seems to give a low squared error. That low order also has the benefit of not giving any numerically hard calculations. The zeros resulting from these estimations are shown in Figs. 3.2(a) - 3.2(d).



(a) "SH" (b) "S"



(c) "F" (d) "TH"

Figure 3.2: Zeros for various types of unvoiced sounds. For each sound type, several different samples are used. Poles and zeros are estimated (using an ARMA(2,3) model) *for each sample* and all the resultning zeros are plotted in the same graph.

ARPABET	Example	ARPABET	Example
IY	beat	NX	sing
IH	bit	P	pet
EY	bait	T	ten
EH	bet	K	kit
AE	bat	B	bet
AA	Bob	D	debt
AH	but	H	get
AO	bought	HH	hat
OW	boat	F	fat
UH	book	TH	thing
UW	boot	S	sat
AX	about	SH	shut
IX	roses	V	vat
ER	bird	DH	that
AXR	butter	Z	zoo
AW	down	ZH	azure
AY	buy	CH	church
OY	boy	JH	judge
Y	you	WH	which
W	wit	EL	battle
R	rent	EM	bottom
L	let	EN	button
M	met	DX	batter
N	net	Q	(glottal stop)

Table 3.1: A list of ARPABET sounds for American English

As one can see from these figures, the zeros of the sounds "SH" and "S" seems to be aligned to certain regions. When it comes to "F" and "TH", though, these zeros does not give much hope of finding a common pattern for unvoiced sound.

The facts that there are not that many unvoiced sounds that can be analyzed in a proper way, that they are not the biggest part of speech and that they are probably not the significant part when it comes to perception of speech by the human ear, implies that we have to get a model that can be used to analyze voiced sounds.

Analysis of Voiced Speech

As mentioned before, the model of Eq. 3.13 relies on the input signal being white noise. To be able to estimate the zeros of voiced speech, we must have a model that can adapt to the periodicity (the pitch period) of these sounds. A useful model for this purpose is the

Output Error structure:

$$x(n) = \frac{B_{LL}(q)}{F_{LL}(q)}u(n - nk) + e(n) \tag{3.14}$$

which uses an external signal, $u(n - nk)$, as input (nk will be explained later on). To use the features of this model in a proper way, we must have an input signal that incorporates the pitch period and leaves a residual signal, $e(n)$, as close to white noise as possible. If we look back to Eq. 2.20, it is easy to see that that relationship is not valid for synthesized speech since the input signal, $Gu(n)$, is simplified and thus not capable of reproducing the exact speech signal. The real relationship includes an error:

$$x(n) = \sum_{k=1}^{p} a_k x(n - k) + Gu(n) + e(n) \tag{3.15}$$

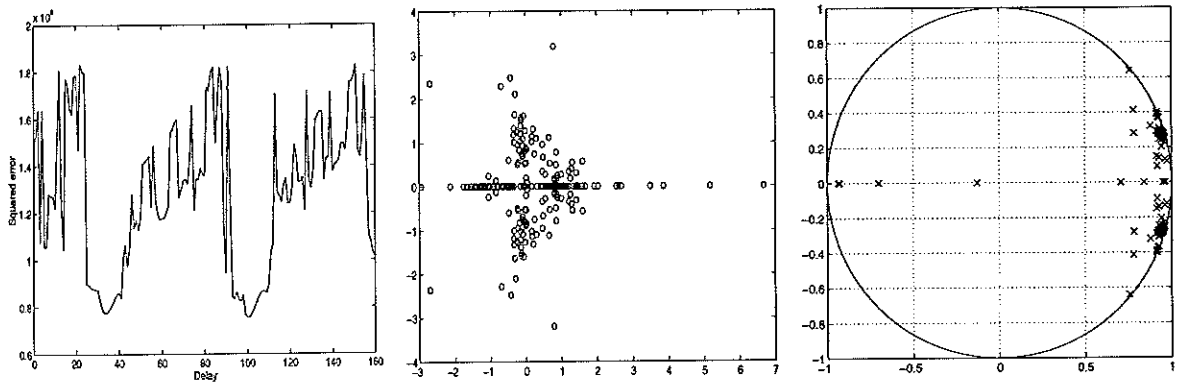
If we use the transfer function of Eq. 2.17 and modify it to contain the basis functions of Eq. 3.2 it is easy to see the connection to Eq. 3.14 (Remember that $B_k(q)$ now contatins a denominator that will correspond to $B_{LL}(q)$):

$$x(n) = \frac{1}{1 - \sum_{k=1}^{p} a_k B_k(q)} Gu_{nk}(n) + e(n) = \frac{B_{LL}(q)}{F_{LL}(q)}u(n - nk) + e(n) \tag{3.16}$$

(The error signal is obviously not the same in the former two equations.) This implies that we can use the input signal used for traditional LPC, $Gu_{nk}(n)$ as input signal to the output error model. One difficulty when using this input signal (a pulse train consisting of zeros except for a peak at the beginning of each pitch period), is that each data point of this signal is actually based on 500 samples of speech (due to the smoothing process described in section 2.2.3). This means that we don't know the best way to synchronize the input signal, that is what delay, nk, to use in Eq. 3.14. However, since the input signal is generated in that way, the question is if it makes any difference what delay we use. If we try some different values of nk and (by cross validation) calculate the corresponding squared error we get the result in Fig. 3.3(a). There is a difference between different values of nk, and one can also see that using input delays that are multiples of the pitch period gives the same result. Does this difference on the squared error give a significant difference on the estimated zeros and poles? To compare this we estimate the zeros for different values of nk and plot them in the same figure, which can be seen in Fig. 3.3(b). The poles are treated the same way in Fig. 3.3(c).

Remembering however that we just want a way of estimating the best representation of the vocal tract filter it seems natural to try different values of nk and choose the delay that minimizes the squared error. The result of an attempt to estimate poles like this is shown in Fig. 3.4. The data used for these estimation experiments are different long lasting vowels since they should best describe a long stationary voiced sample sequence.

As seen from these figures, various long lasting vowels seem to show patterns similar to each other. When plotted together though, one can see that the exact position of the zeros of the vowels can vary a lot.



(a) $\sum e^2(n)$ (b) Zeros (c) Poles

Figure 3.3: Results when using the output error model with various delays of the input signal. In b) and c) zeros and poles respectively are estimated using different delays and the results are plotted in the same figure.

As seen from Fig. 3.3, the positions of the zeros are very much depending on the delay of the input signal. This means that it could be difficult to synthesize the speech using the optimal zeros of the preceding analysis of the vocal tract filter. Since the input signal must be delayed a certain time (different time was used for each minimized squared error in Fig 3.4) for this filter to be valid, we have to find that delay in order to reliably reconstruct the speech. Moreover if we just miss the exact delay by one sample, the zeros will end up in completely different positions. An example of what happens with the filter estimations when the delay is shifted within the range $[nk_{optimal} - 5, nk_{optimal} + 5]$ is shown in Fig. 3.5.

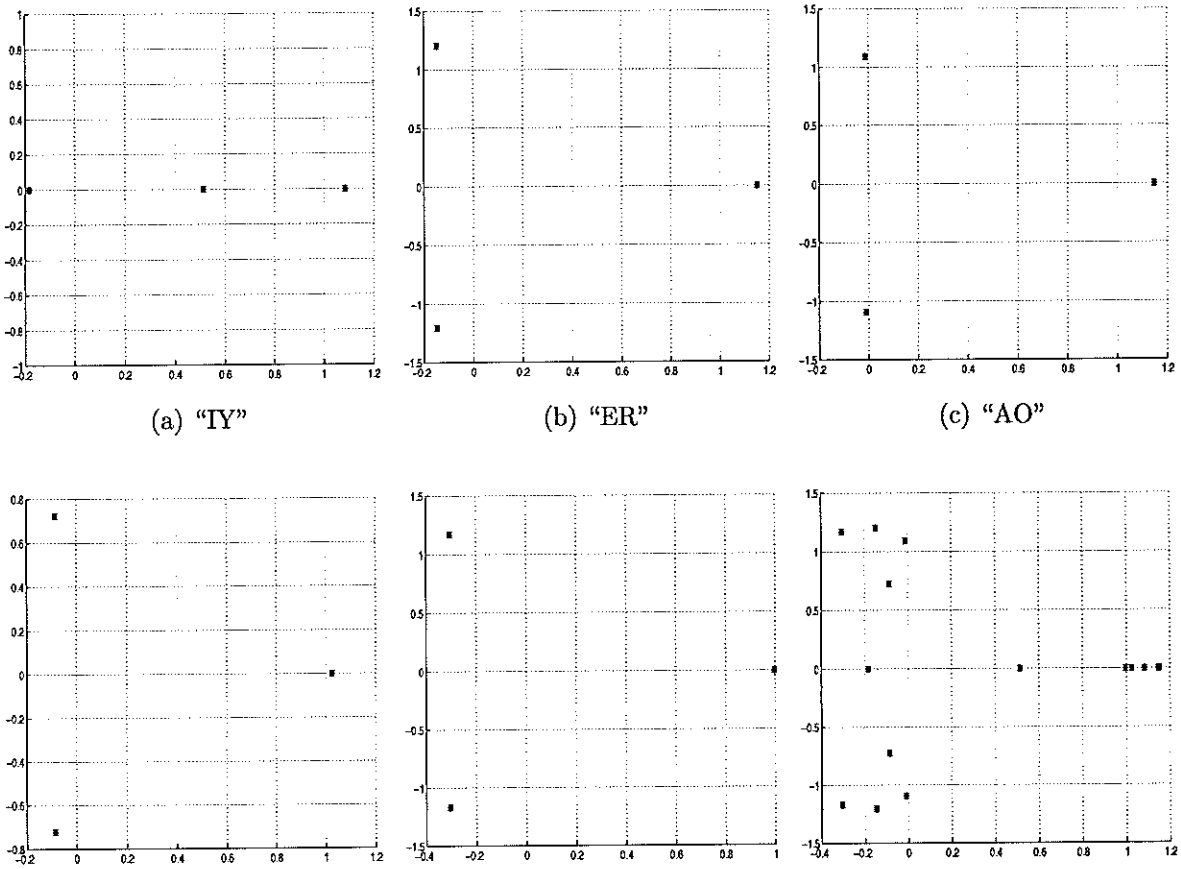
3.4 Analysis of the Poles of the Vocal Tract Filter

If we look back to Fig. 3.3(c), the poles of voiced speech seem to be more robust than the zeros when using different delays of the input signal, that is they don't move around that much. This is an interesting observation that implies that we should do a more accurate investigation on the position of the poles when using an input signal.

3.4.1 Finding the Poles of the Vocal Tract Filter

When estimating the *poles* of voiced speech we may consider using another model order than the one used with the output error structure so far ($deg(B_{LL}) = 3$, $deg(F_{LL}) = 2$). Using the estimation / validation procedure outlined in section 3.3.2, one can see that $deg(B_{LL}) = 1$ and $deg(F_{LL}) = 3$ gives a reasonable low squared error, while not having that many zeros interfering (as e.g. canceling) the estimation procedure.

We now proceed with the estimation of poles the same way that was done for the zeros in section 3.3.2, that is we extract many segments of a certain voiced sound from TIMIT,



(a) "IY" (b) "ER" (c) "AO"

(d) "UW" (e) "OW" (f) All together

Figure 3.4: Zeros for long lasting vowels

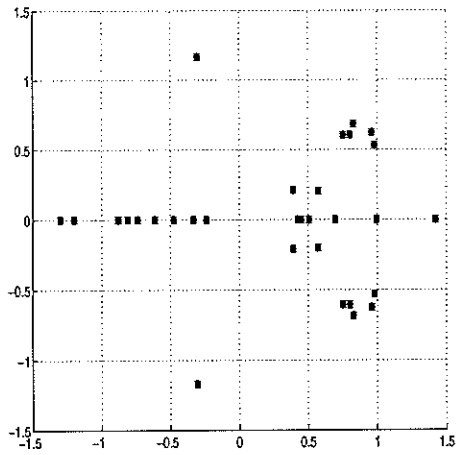


Figure 3.5: How the filter estimation of "OW" varies with delays close to the optimal one.

estimate the poles with the output error model for each segment and plot the results in the same figure to search for a pattern. Once again it is appropriate to start of with the analysis of long lasting vowels. The result is shown in Fig. 3.6.

One can certainly see a pattern in the position of these poles, which seems to be located close to the unit circle with a low imaginary part. Having seen these interesting results, we study diphthongs, which are sounds that can be thought of as gliding speech that starts at or near the articulatory position for one vowel and moves to or toward the position for another. These sounds are high energy, voiced sounds that should play an important role of the perception of speech as well as the vowels, though they might give a different result in the pole analysis as they suffer from the lack of absolute stationarity. The result is shown in Fig. 3.7.

3.4.2 Using the Poles of the Vocal Tract Filter

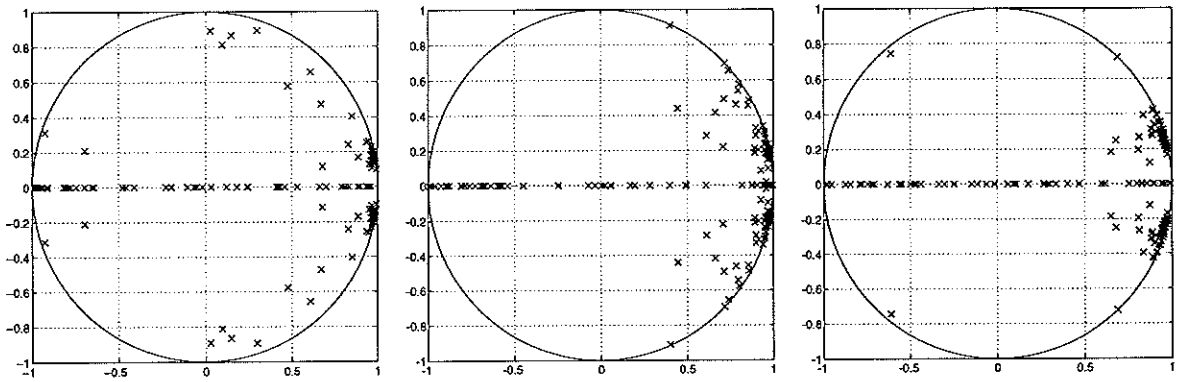
Having noticed this pattern of the poles of the vocal tract filter when we were looking for a pattern of the zeros, how can we use this knowledge?

All the time we have been looking for the zeros of the vocal tract filter in order to enhance the traditional LPC model (which uses an all-pole filter when reconstructing the speech signal). We generally estimate the filter according to the model given by Eq. 3.8:

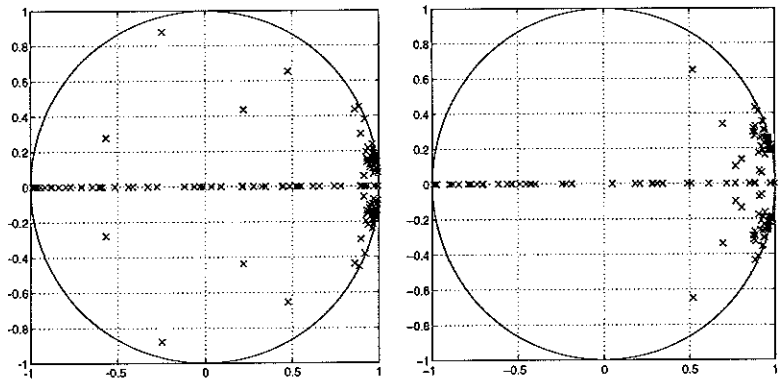
$$x(n) = F(q)x(n) + e(n) \tag{3.17}$$

where $F(q)$ is given by Eq. 3.9 for traditional LPC, while the new approach is to use the base functions given by 3.12. No matter what type of speech (voiced or unvoiced) the frame being analyzed contains, the reconstruction must be of the form

$$x(n) = \frac{1}{1 - F(q)}e(n) \tag{3.18}$$

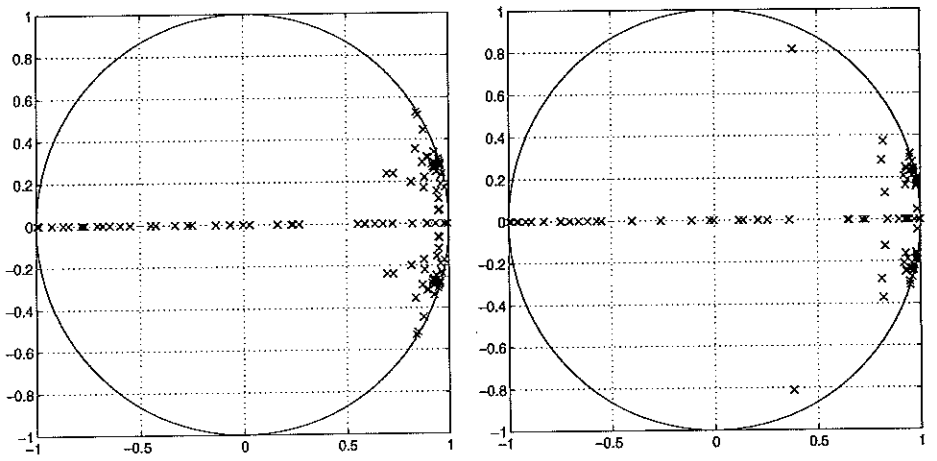


(a) "IY" (b) "ER" (c) "AO"

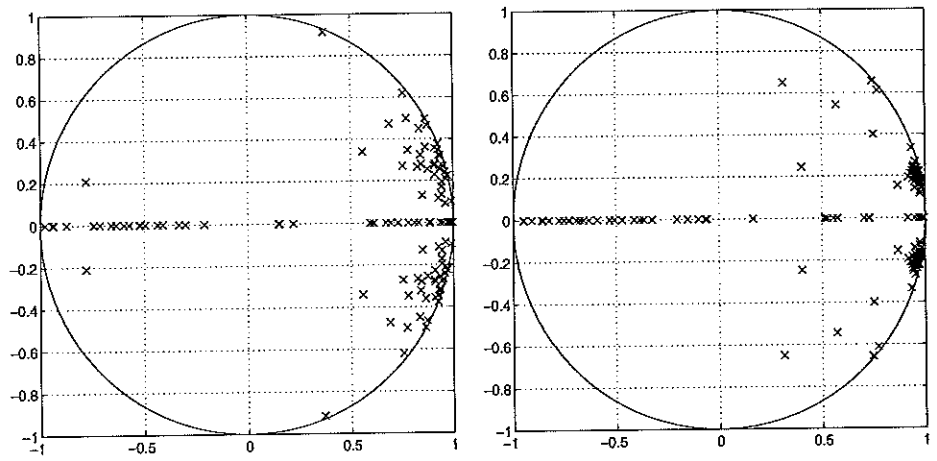


(d) "UW" (e) "OW"

Figure 3.6: Poles for long lasting vowels. For each sound type, several different samples are used. Poles and zeros are estimated *for each sample* and all the resultning poles are plotted in the same graph.



(a) "AY" (b) "OY"



(c) "AW" (d) "EY"

Figure 3.7: Poles for diphthongs. For each sound type, several different samples are used. Poles and zeros are estimated *for each sample* and all the resultning poles are plotted in the same graph.

where the residual error signal is either white noise (for unvoiced sound) or a pulse train (for voiced sound). When we started to analyze voiced sound however, the nature of the analysis tool (Ljung's Toolbox) forced us to produce an input signal as described by Eq. 3.14. The input signal used was the input signal created by traditional LPC algorithms. When we look at this equation and see what type of model we actually are estimating, it would be quite straight forward to reconstruct the speech sample by feeding the estimated filter $B_{LL}(q)/F_{LL}(q)$ with the input signal (recall that this is the signal that already is used in the synthesizer). Thus, for voiced sound, why not use the model

$$x(n) = F(q)u(n) + e(n) = \sum_{k=1}^{p} a_k B_k(q)u(n) \; + e(n) \tag{3.19}$$

instead?

A first approach to see if any improvement can be made with this model, is simply to try it on a speech sample and listen and compare the reconstructed speech with the result from traditional LPC. This is made with a fourth order model with the poles located at $0.97 \pm 0.18i$ and $0.95 \pm 0.27i$ (these locations seem reasonable if you inspect the graphs of Fig. 3.6). Unfortunately, having listened to this, the result is quite poor. It is easy to hear that the model of Eq. 3.19 gives worse result than traditional LPC. Why is that?

To gain some insight in the problem, we once again concentrate on a sample that is not that complex - a long lasting vowel (the longest segment of IY-sound found in TIMIT). To see how sensitive the reconstruction is when varying parameters of the model, we now visually inspect the predicted sample. Both the model of Eq. 3.14 and the model of Eq. 3.19 (used with the base functions of Eq. 3.12) use the least square technique to estimate the parameters of the filters, which means that if we estimate the zeros of $F_{LL}(q)$ and fix the poles of $F(q)$ at the same locations we should end up with the zeros of $F(q)$ estimated at approximately the same locations as the zeros of $B_{LL}(q)$. The predicted speech sample would also be about the same for the models. To avoid numerical problems we use a model with two zeros and three poles (the model of Eq. 3.19 always has one zero less than its number of poles). The result of the output error structure is shown in Fig. 3.8, where the dashed line is the predicted sample and the solid line is the original one.

The estimated poles are located at $0.9653 \pm 0.1555i$, -0.9744 and the zeros at -0.9730 and 0.8226. if we fix the poles at these locations using the new orthonormal base functions model, we end up with the same result (shown in Fig. 3.9(a)).

As pointed out in section 3.3.2, a problem when using an input signal trying to predict the next sample is how to synchronize the input signal with past samples. It could be interesting to see what happens if we change the delay time for the input signal. The result of this is shown in Fig. 3.9(b) and having seen this, it seems as the delay first used was only a fortunate guess. However, if you compare the predicted (dashed lines) samples to each other, the structure seems to be quite simular. The major difference is a delay for a couple of samples, which is not likely to sound that different. There must be another reason why this model doesn't seem to work.

What will happen if we disturb the pole positions, that is if we change them slightly from the positions resulting from the output error estimation? This was made to get the

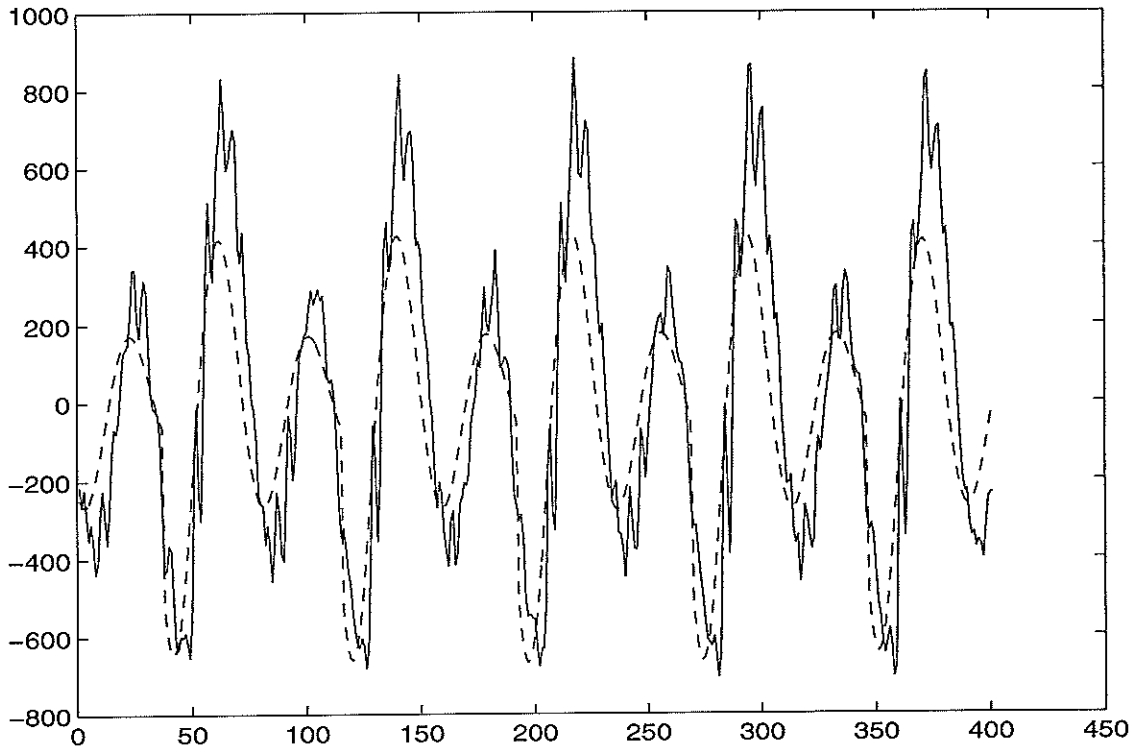


Figure 3.8: Predicted sample from output error model. The dashed line is the predicted sample and the solid is the original one.

graphs of Fig. 3.9(c) and 3.9(d). In Fig. 3.9(c) the poles were moved 2 % closer to the origin, while they in Fig. 3.9(d) were moved 2 % closer to the unit circle (they are still within the region of stability). From this we can conclude that the result is very sensitive to the pole placement and it is probably impossible to find some pole positions that would allow a fair reconstruction for all of the vowel sounds.

3.5 Finding the Zeros of the Vocal Tract Filter (second approach)

Althoug we didn't find a reliably suitable pattern of placing the zeros by estimating a pole-zero model in section 3.3, it still seems strange that placing the zeros at the origin would be the best choice of all possible choices as given by Eqs 3.2 and 3.12.

Since we are interested in placing the zeros so as to get a speech reconstruction that sounds better, rather than finding a model that explains the underlying physics behind speech production, one way of finding suitable zeros would be to move around a few zeros and compare the resulting samples.

A simple way of comparing the resulting coded/decoded speech samples for different pole positions is to visually inspect them and in some way guess which choice that would

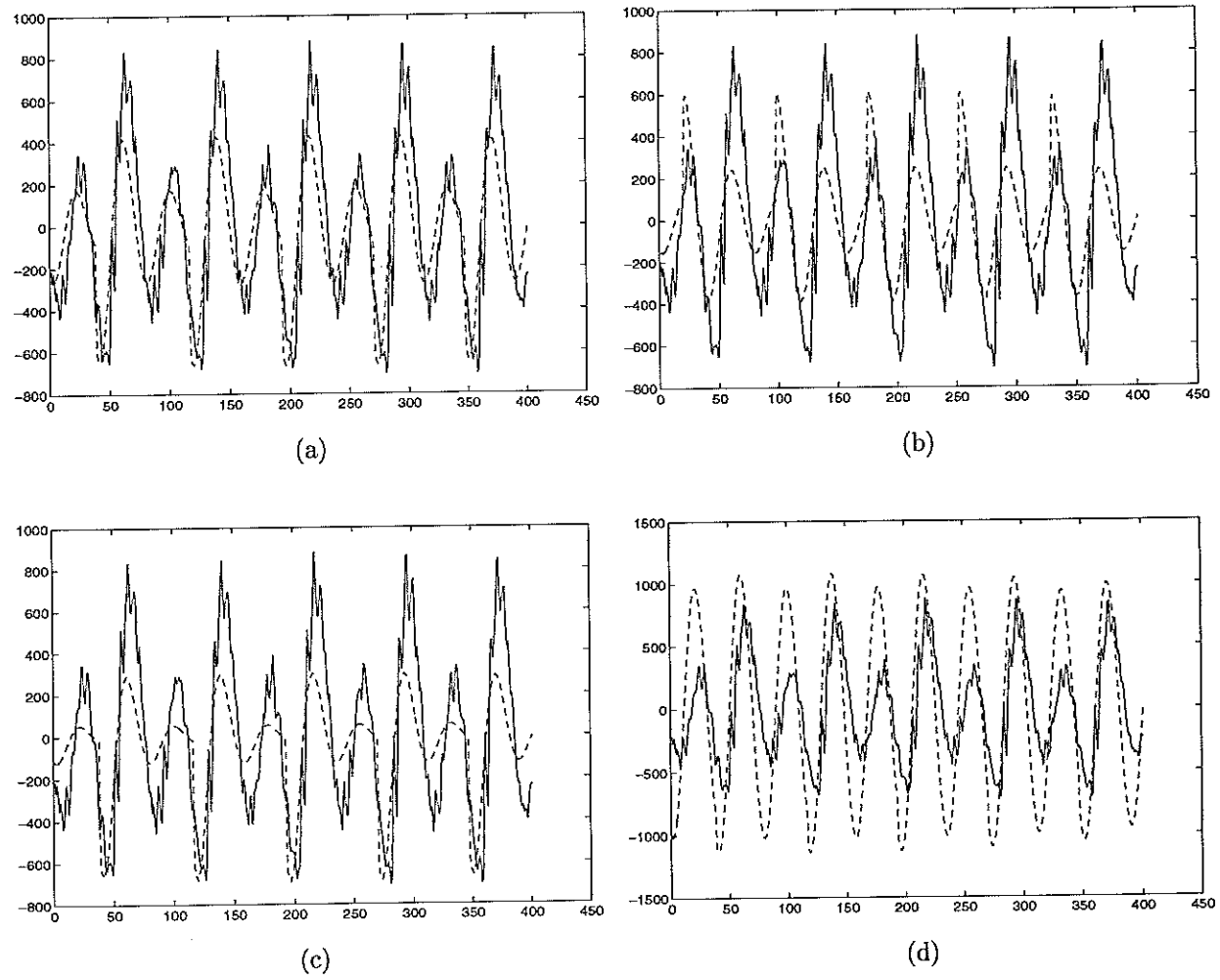


Figure 3.9: Predicted samples from new orthonormal base functions model. The dashed line is the predicted sample and the solid is the original one.

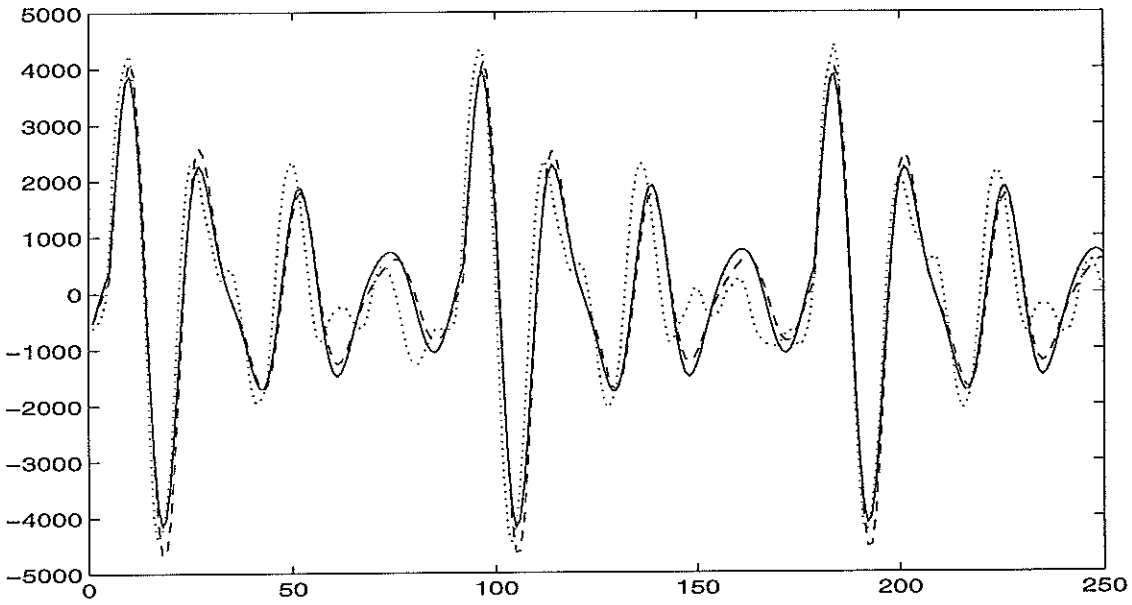


Figure 3.10: Predicted sample from fifth order orthonormal bases model (solid line) and tenth order FIR model (dashed line). The original speech sample is also included (dotted line). Sound "AO".

give the best sound. So how do we know what a "good sounding" sample is? Since LPC based on a tenth order FIR filter is commonly used today, we could use that coding technique to get a reconstructed sample that sounds reasonably good. If by using the new orthonormal bases function model we can obtain a similar sample using a lower order, say fifth order, then LPC compression ratios twice as good as the FIR case can be achieved. (It would not be exactly twice as good as the FIR case since the predictor coefficients are not the only parameters that are sent.) Once again we investigate one type of sound at a time, always using the longest stationary sequence we can find of that particular sound.

Having tried this for many different types of sound, it actually seems possible to achieve this for a certain set of poles ($0.78 \pm 0.24i$, $0.30 \pm 0.23i$ and 0.20)! Figs. 3.10 – 3.13 show a comparison between the reconstructed speech sample resulting from fifth order orthonormal bases model (solid line) and tenth order FIR model (dashed line). The original speech sample is also included (dotted line). As seen from these figures, the predictors does not always look the same as the original sample. The fifth order OB (orthonormal bases) model however seems to capture as much of the original sample as the tenth order FIR model. In those cases where the OB predictor differ from the FIR model, it even seems as if "the noisy part" is reduced by the OB function filter. In each case (not plotted here) the result is at least better than the one achieved by a fifth order FIR.

Since the thoughts presented above are based on intuition, we must confirm these results by listening to the samples generated by the fifth order OB model. Surprisingly, having listened to a long reconstructed speech sample (the sentences "A lathe is a big tool. Grab every dish of sugar." [35]), the result turns out to be quite poor. In some parts of the

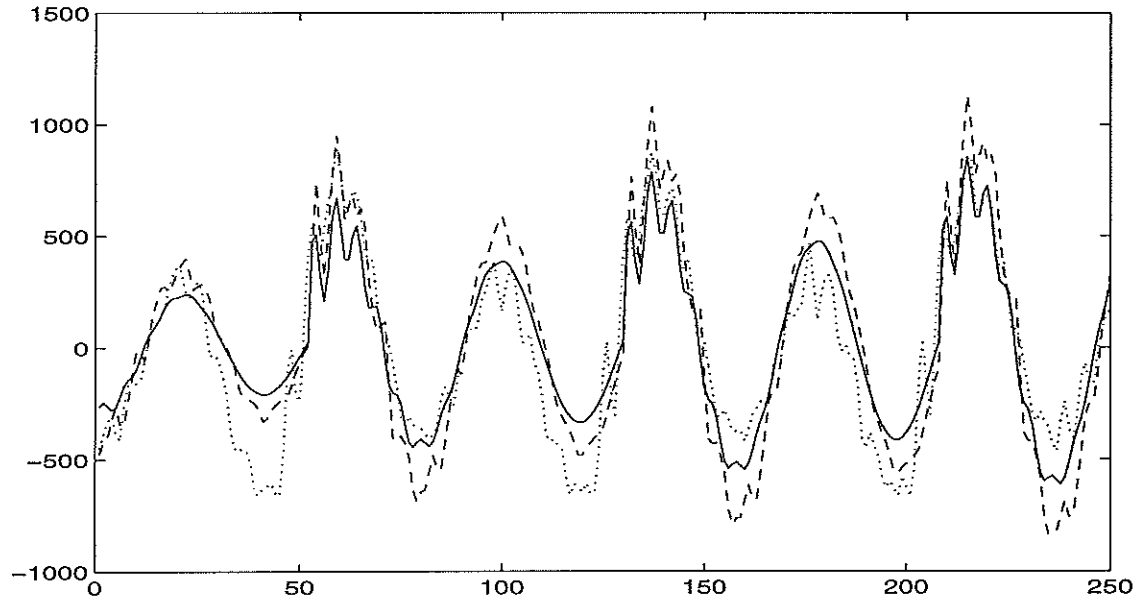


Figure 3.11: Predicted sample from fifth order orthonormal bases model (solid line) and tenth order FIR model (dashed line). The original speech sample is also included (dotted line). Sound "IY".

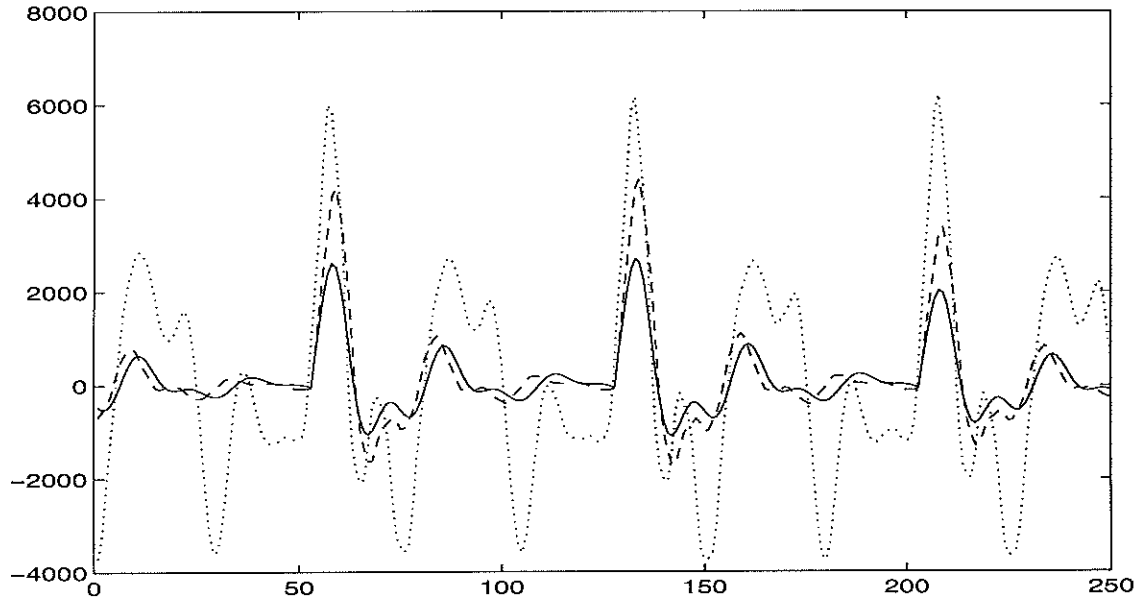


Figure 3.12: Predicted sample from fifth order orthonormal bases model (solid line) and tenth order FIR model (dashed line). The original speech sample is also included (dotted line). Sound "ER".

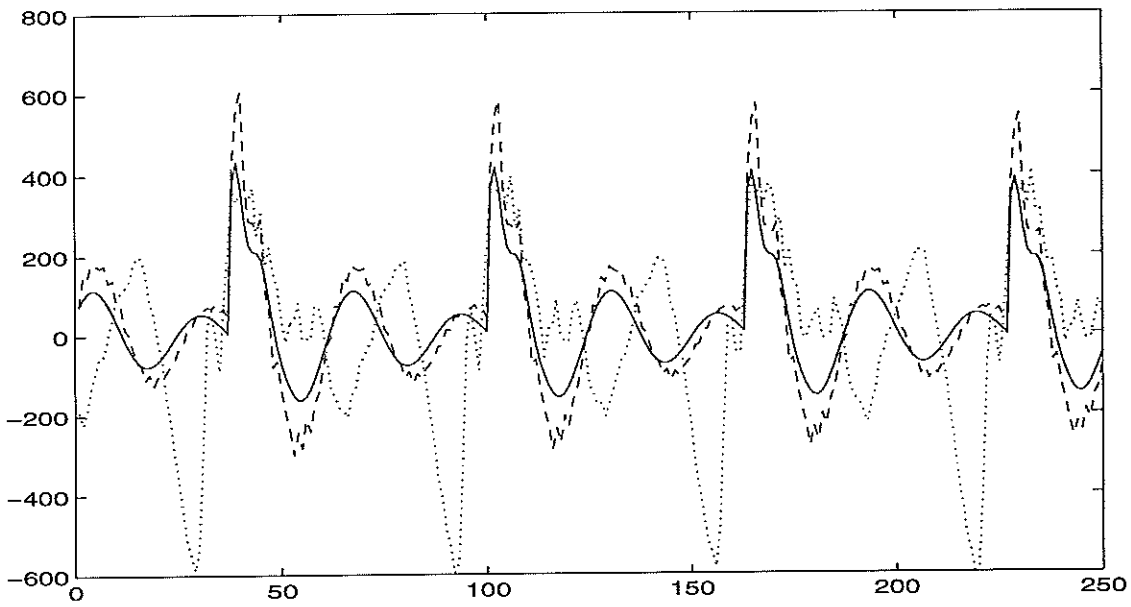


Figure 3.13: Predicted sample from fifth order orthonormal bases model (solid line) and tenth order FIR model (dashed line). The original speech sample is also included (dotted line). Sound "N".

sample, one cannot even find out the original words!

This large discrepancy in aural properties where there is very little difference in the visual properties seems quite surprising. To try and explain this discrepancy (for that particular set of chosen poles), we look at the frequency properties of the sample. Let us pick stationary parts of the sample at various positions and look at the frequency content in the original sample and the reconstructed fifth order OB sample and tenth order FIR sample. Something that all these parts have in common is that the original speech sample and the FIR sample have some frequencies around 2 kHz that is missing in the OB sample. This is seen in Fig. 3.14, where a typical frequency spectrum is plotted for the original speech sample and the reconstructed OB sample. (The peak in the spectrum of the original sample around 700 Hz that is not properly reconstructed is a special result from the analysis of this frame rather then a common phenomenon.)

If this is the reason why the reconstructed sample is at impaired clarity, then we should be able to change the frequency contents of the reconstructed speech by moving the poles. As pointed out in [36], each pole of a digital filter corresponds to a certain output. For instance a complex conjugate pole pair $\alpha e^{\pm i\beta}$ as shown in Fig. 3.15, corresponds to the term $A\alpha^n cos(\beta n + \theta)$ in the output $y(nT)$ (n is the index of the sample, T is the sampling period).

This means that if we want to be sure of including a frequency around 2 kHz, we should include a pole pair with $\beta = 2\pi\frac{f}{f_s} = 2\pi\frac{2000}{8000} = \frac{\pi}{2}$. Having tried different pole positions again (now with the constraint of one complex conjugate pair of poles lying on the imaginary

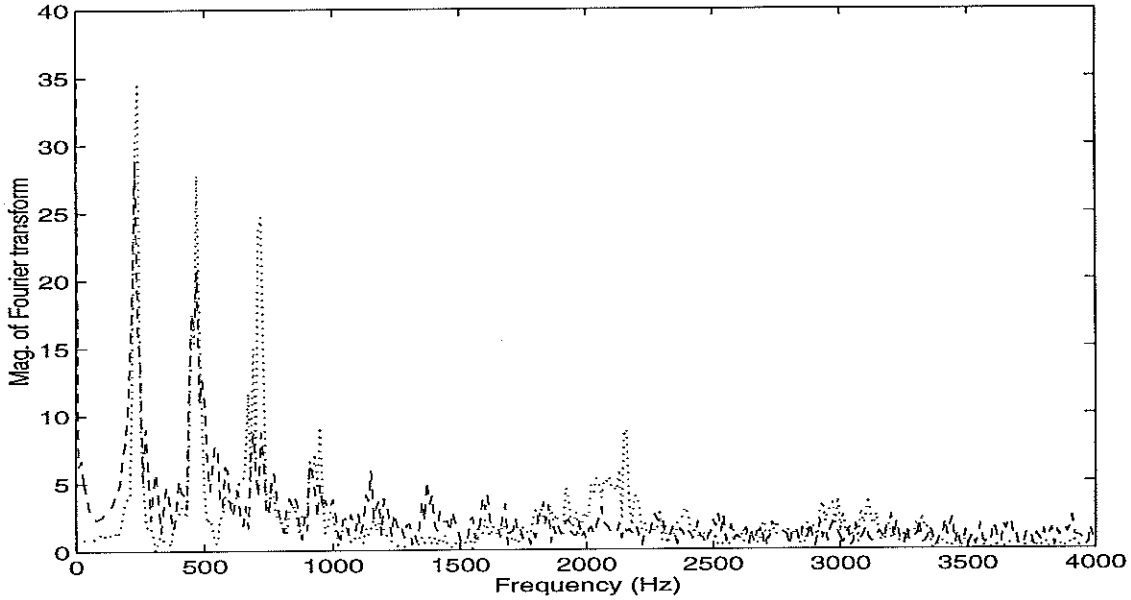


Figure 3.14: Frequency spectrum of original speech signal (dotted) and predicted signal (solid). Poles at $0.78 \pm 0.24i$, $0.30 \pm 0.23i$ and 0.20

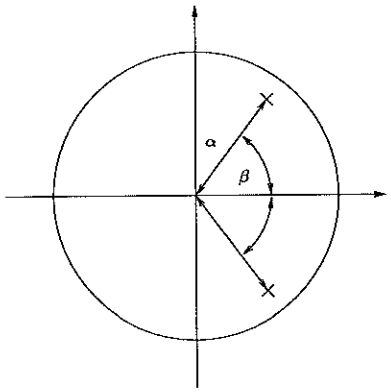


Figure 3.15: Plot of a complex conjugate pair of poles.

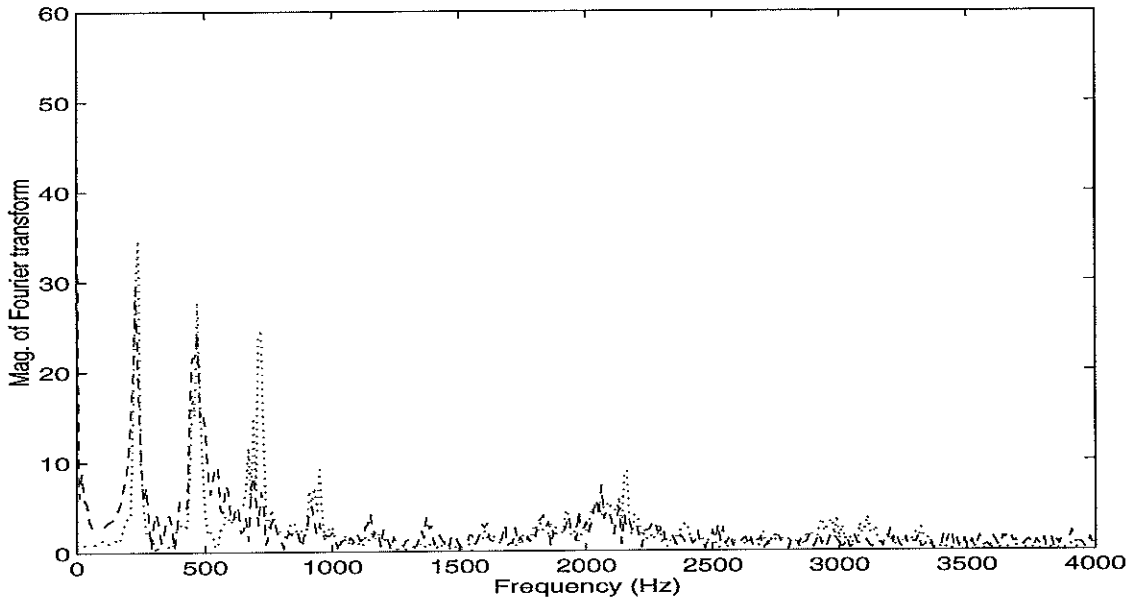


Figure 3.16: Frequency spectrum of original speech signal (dotted) and predicted signal (solid). Poles at $\pm 0.8i$, $0.30 \pm 0.23i$ and 0.20

axis), $\pm 0.8i$, $0.30 \pm 0.23i$ and 0.20 seems to be reasonably good. That this choice of pole placement improves the spectrum of the reconstructed sample can easily be seen from Fig. 3.16.

However, having listened to the reconstructed sample, the result might be somewhat better than the one produced by a fifth order FIR filter, but we are still far away from a tenth order FIR.

At this stage it could be interesting to investigate the importance of certain frequencies. For instance does it really matter if we remove the frequencies mentioned earlier from the original sample? To answer this question we filter out the frequencies between 1600 Hz and 2400 Hz. Having listened to the result of this, there is not much of a difference from the original sample. The matter of fact is that one can actually filter out all frequencies above 1600 Hz and (even if the sound quality is quite poor) still be able to determine what the spoken sentences are. Hence it seems hard to connect the substantial parts of speech to any special frequency intervals.

Chapter 4

Conclusions

Various ideas on how to extend the simple but powerful traditional LPC method was investigated in this report. Experiments that were made to determine how successful these ideas are when it comes to perception by the human ear (i.e. sample, code, decode and listen), gave us some interesting information. A comparison between visual and aural inspection of speech generated by a coding/decoding procedure showed that it is very difficult to find criteria for "good sounding" speech samples that does not include listening to them. It would be desirable (in order to omit some time consuming procedures of the analysis of the result) to be able to make conclusions about the accuracy of the reconstruction of a speech sample by pure visual inspection in time or frequency domain. The search for a coder with a higher compression ratio would be even more straight forward if we had a *mathematic* expression that would describe the speech quality.

Another and even more important discovery is the limitation of the linear predictive coder. You are not able to improve the performance of this coder significantly without using more sophisticated methods. The zeros of the vocal tract filter in traditional LPC are placed at the origin, which is a special case of all possible configurations. Since all speech coding methods based on linear prediction use such a filter (when not using adaptive zeros), we investigated the effect of moving the zeros out from the origin. The somewhat surprising result was that you can not achieve significantly better speech quality with any other fixed configuration.

The explanation of not being able to improve the reconstructed speech by adding the proposed extra degrees of freedom is found in the literature. The proposed new model moves the zeros of the vocal tract filter out from the origin, which is needed in order to model the nasal cavity (As pointed out in section 3.2.1). The contribution from zeros to other sounds is negligible as the traditional LPC using an all-pole filter reproduce them quite well. Thus the benefit from introducing zeros in the vocal tract is to reproduce nasal sounds. The nasal sounds though are complex and do not have that smooth regularity that is easily found in, for instance, long lasting vowels, which implies that if we also want to reproduce the nasal sounds with clarity we must include them among the *adaptive* parameters. To fix them at certain positions will not carry us all the way.

A drawback by using linear prediction is that as the bit rate decreases, the reconstructed

signal is very noisy since the accuracy of waveform matching decreases. Waveform coders (that were mentioned in the introduction of this report) constitute another branch of speech coding where filters based on orthonormal basis functions can be used. In sub-band coding, which is a waveform coding technique carried out in the frequency domain, the signal is passed through a bank of bandpass filters. Each sub-band is then low pass translated and the sampling rates are reduced to the Nyquist rate for each band. The benefit of this technique is that the number of bits assigned to each band (and hence the quantization noise) can be varied according to the perceptual importance of the band. At the receiver the sampling rates are increased, the bands are modulated back to their original positions and finally summed to produce the output speech.

There is a promising ongoing research in letting *wavelet bases* represent the bandpass filters and via some programming experiments, the author has verified that it is quite simple to (by using wavelet filters) implement a rough coder for speech compression. Since this type of coder is signal independent, wavelets can also be used to compress images and it is possible that wavelet-based compression will be employed in High-Definition Digital Television in the future.

Bibliography

[1] H. R. S. Mohammadi, *Efficient Coding Of The Short-Term Speech Spectrum* Iran University of Science and Technology, 1995.

[2] J. D. Markel and A. H. Gray, *Linear Prediction of Speech*, Springer-Verlag, New York 1976.

[3] J. Makhoul, "Linear Prediction: a Tutorial Review", *Proc. IEEE*, vol. 63, no 4, pp. 561-580, Apr. 1975.

[4] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, New Jersey, USA, 1975.

[5] J. Durbin, "The fitting of time series models", *Review of Institute for International Statistics*, vol. 238, pp. 233-243, 1960.

[6] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*, Prentice-Hall, Inc., New Jersey 1978.

[7] J. J. Dubnowski, R. W. Schafer, and L. R. Rabiner, "Real-Time Digital Hardware Pitch Detector", *IEEE Trans. Acoust., Speech, and Signal Proc.*, Vol. ASSP-24, No. 1 pp. 2-8, February 1976.

[8] J. W. Tukey, "Nonlinear (Nonsuperposable) Methods for Smoothing Data", *Congress Record, 1974 EASCON*, p. 673, 1974.

[9] L. R. Rabiner, M. R. Sambur, and C. E. Schmidt, "Applications of a Nonlinear Smoothing Algorithm to Speech Processing", *IEEE Trans. Acoust., Speech and Signal Proc.*, Vol. ASSP-23, No. 6, pp. 552-557, December 1975.

[10] U. Grenander and G. Szegö, *Toeplitz Forms and Their Applications* (Univ. California Press, Berkeley, 1958), p. 40.

[11] B. S. Atal and S. L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction of the Speech Wave", *J. Acoust. Soc. Am.*, Vol. 50, pp. 637-655, 1971.

[12] B. Ninness and F. Gustafsson, *A unifying construction of orhonormal bases for system identification*, in Proceedings of the 33rd IEEE Conference on Decision and Control, December 1994, pp. 3388-3393.

[13] B. Wahlberg, *System identification using Laguerre models*, IEEE Transactions on Automatic Control, AC-36 (1991), pp. 551-562.

[14] B. Wahlberg and L. Ljung, *Design variables for bias distribution in transfer function estimation*, IEEE Transactions on Automatic Control, AC-31 (1986), pp. 134-144.

[15] L. Ljung, *System Identification: Theory for the User*, Prentice-Hall, Inc., New Jersey, 1987.

[16] J. E. Shoup, "Phonological Aspects of Speech Recognition," 125-138, Ch.6 in *Trends in Speech Recognition*, W. A. Lea, Ed., Prentice-Hall, Englewood Cliffs, NJ, 1980.

[17] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Inc., New Jersey 1993.

[18] Juin-Hwey Chen, Gersho A, *Adaptive postfiltering for quality enhancement of coded speech.*, IEEE Transactions on Speech and Audio Processing, vol. 3, no. 1, pp. 59-71, Jan. 1995

[19] Hong Chae Woo, Gibson J.D., *Low delay tree coding of speech at 8 kbit/s.*, IEEE Transactions on Speech and Audio Processing, vol. 2, no. 3, pp. 361-370, July 1994

[20] Miseki K., Akamine M., *Adaptive bit-allocation between the pole-zero synthesis filter and excitation in CELP.*, 1991 International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. 229-232, 1991

[21] Flanagan J.A., Murray B., Fagan A.D., *Pole-zero code excited linear prediction.*, Sixth International Conference on Digital Processing of Signals in Communications (Conf. Publ. No. 340) London, UK, pp. 42-47, 1991

[22] Spanias A.S., Mikhael W.B., *An adaptive bit allocation scheme for coding of speech signals using partial sets of orthogonal functions.*, Proceedings of the 32nd Midwest Symposium on Circuits and Systems, vol. 1, pp. 598-601, 1990

[23] Spanias A.S., *A pole-zero adaptive algorithm for speech processing.*, Ninth Annual International Phoenix Conference on Computers and Communications, p. 894, 1990.

[24] Lee K.Y., Evans B.G., *Combined optimization of excitation and filter parameters in analysis-by-synthesis coders.*, Eurospeech 89. European Conference on Speech Communication and Technology, vol. 2, pp. 501-504, 1989

[25] Thomson D.L., *Parametric models of the magnitude/phase spectrum for harmonic speech coding.*, 1988 International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. 378-381, 1988

[26] Ramamoorthy V., Jayant N.S., Cox R.V., Sondhi M.M., *Enhancement of ADPCM speech coding with backward-adaptive algorithms for postfiltering and noise feedback.* IEEE Journal on Selected Areas in Communications, vol. 6, no. 2, pp. 364-382, Feb. 1988

[27] Koo B., Gibson J.D., *Experimental comparison of all-pole, all-zero, and pole-zero predictors for ADPCM speech coding.*, IEEE Transactions on Communications, vol. COM-34, no. 3, pp. 285-290, March 1986

[28] De Lima Araujo A.M., Alcaim A., Boisson De Marca J.R., *Independent analysis of quantiser and predictor used in CCITT 32 kbit/s ADPCM algorithm.*, Electronic Letters, vol. 22, no. 4, pp. 220-221, 13 Feb. 1986

[29] Steiglitz K., Dickinson B., *The use of time-domain selection for improved linear prediction.*, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASSP-25, no. 1, pp. 34-39, Feb. 1977

[30] Morikawa H., Fujisaki H., *Adaptive Analysis of Speech Based on a Pole-Zero Representation*, IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. ASSP-30, no. 1, Feb. 1982

[31] Gibson J.D., *Speech Prediction Algorithms Based Upon ARMA Filters*, IEEE International Symposium on Circuits and Systems, San Jose, CA, 5-7 May 1986, pp. 281-284.

[32] S. Yim, D. Sen and W. H. Holmes, *Comparison of ARMA modelling methods for low bit rate speech coding*, IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. I/273-276, 1994

[33] I.-T. Lim and B. G. Lee, *Lossless Pole-Zero Modeling of Speech Signals*, IEEE Trans. Speech Audio Processing, vol. 1, no. 3, pp. 269-276, July 1993.

[34] I.-T. Lim and B. G. Lee, *Lossy Pole-Zero Modeling for Speech Signals*, IEEE Trans. Speech Audio Processing, vol. 4, no. 2, pp. 81-88, March 1996.

[35] F. Alajaji, N. Phamdo, and Tom Fuja, "Channel Codes That Exploited the Residual Redundancy in CELP-Encoded Speech,' submitted to IEEE Transactions on Speech and Audio Processing, May 1995.

[36] P. M. DeRusso, R. J. Roy, and C. M. Close, *State Variables for Engineers*, John Wiley & Sons, Inc., 1965